The GROMOS Software for (Bio)Molecular Simulation

Volume 1: About the GROMOS package: Overview

January 9, 2021

Contents

CHAPTER 1

What is GROMOS

GROMOS is an acronym of the GROningen MOlecular Simulation computer program package, which has been developed since 1978 for the *dynamic modelling of (bio)molecules*, until 1990 at the University of Groningen, The Netherlands, and since then at the ETH, the Swiss Federal Institute of Technology in Zürich, Switzerland. Until 2013 its development was driven by the research group of Wilfred van Gunsteren. Currently, the development is shared between him and the research groups of Philippe Hünenberger and Sereina Riniker at the ETH and of Chris Oostenbrink at the Institute for Molecular Modeling and Simulation of the University of Natural Resources and Life Sciences in Vienna, Austria.

Since the last official release of the GROMOS software and manual in 1996, called GROMOS96, no comprehensive release occurred till 2011. Yet the GROMOS software has seen a steady development since 1996, see e.g. Christen *et al.* J. Comput. Chem. **26** (2005) 1719. The programming language has been changed from FORTRAN to C++, the documentation has been put into electronic form, and many new features have been included in the software. In spring 2011 an official version of the C++ software was released and this volume summarizes the basic principles underlying the development of the GROMOS software.

The GROMOS software has been developed as a research vehicle of the van Gunsteren research group, which was characterized by a changing composition of members that are interested in both methodological developments and a variety of applications of simulation methods. This dictated the following principles for development:

- Transparency of code, so that modification is easy.
- Molecular architecture, so that parts of it can be used in changing combinations and functions may be replaced by ones written by the user.
- Independence of the code of the force field that is used.
- Independence of the code of the units of physical quantities and constants.
- Independence of the code of the computer hardware that is available.

The criteria for inclusion of new features into GROMOS are, ordered according to decreasing importance:

1. Research and teaching interests of the research groups developing the code.
2. Scientific importance
3. Demonstrated usefulness or efficiency
4. Well-defined and correct formulae and algorithms
5. Extent of use by the scientific community
6. Ease of implementation
7. Computational efficiency

CHAPTER 2

The GROMOS force fields

The GROMOS software is to be distinguished from the GROMOS force fields for biomolecular systems.

The quality of the interaction function or force field that describes the forces between the atoms of a biomolecular system is of decisive importance for the predictive power of MD simulations. Therefore, we have over the past decades spent much effort to gradually improve the GROMOS force field whenever results of simulation applications pointed at force field deficiencies. The first set of non-bonded GROMOS force field parameters dates from 1984,[1] while the bonded parameters were taken from ref.[2]. Since then, the force field has continuously been improved and refined.[3–15] The most widely used versions of the GROMOS force field are the GROMOS 37C4 force field of 1985, the GROMOS 43A1 force field of 1996[4,16] and the GROMOS 45A3 force field of 2001.[6] The currently used versions are the 45A4 parameter set,[7,9,10] the 53A5/6,[8] the 54A7,[12,14] and the 54A8 one.[15] In parallel to the development of force field parameters for biomolecules, solvent models that are consistent with the GROMOS biomolecular force field were developed for much used (co-)solvents:[17] water,[18,19] methanol,[20] DMSO,[21] chloroform,[22] carbontetrachloride,[23] urea,[24] acetonitrile.[25] A polarisable force field is under development,[26–37] as are supra-molecular coarse-grained ones for water[38–40], co-solvents[41–43] and lipids.

CHAPTER 3

GROMOS functionalities and documentation

GROMOS has the following *basic capabilities*.

1. Simulation of biomolecules or arbitrary molecules using the molecular dynamics (MD) or stochastic dynamics (SD) methods.
2. Analysis of molecular configurations and velocities or energies obtained by computer simulation or model building based on experimental (X-ray, NMR) data.

The GROMOS software manuals that accompanied the major releases of 1987 and 1996 are

W.F. van Gunsteren and H.J.C. Berendsen
Groningen Molecular Simulation (GROMOS) Library Manual
Biomos, Groningen, The Netherlands, 1987, pp. 1-221

W.F. van Gunsteren , S.R. Billeter, A.A. Eising, P.H. Hünenberger, P. Krüger, A.E. Mark, W.R.P. Scott and I.G. Tironi
Biomolecular Simulation: The GROMOS96 Manual and User Guide
Vdf Hochschulverlag AG an der ETH Zürich, Zürich, Switzerland, 1996, pp. 1-1042

The current GROMOS manual and user guide exists of 9 volumes:

The GROMOS Software for (Bio)Molecular Simulation
Volume 1: About the GROMOS Package: Overview
Volume 2: Algorithms and Formulae for Modelling of Molecular Systems
Volume 3: Force Field and Topology Data Set
Volume 4: Data Structures and Formats
Volume 5: Program Library Manual
Volume 6: Technical Details
Volume 7: Tutorial with Examples
Volume 8: Installation Guide
Volume 9: Index

The functionalities of GROMOS 87, GROMOS96 and GROMOS 05 have been summarized in

- W.R.P. Scott and W.F. van Gunsteren
  The GROMOS Software Package for Biomolecular Simulations
  In: Methods and Techniques in Computational Chemistry: METECC-95, E. Clementi and G. Corongiu editors, STEF, Cagliari, Italy, 1995, pp. 397-434.
- W.R.P. Scott, P.H. Hünenberger, I.G. Tironi, A.E. Mark, S.R. Billeter, J. Fennen, A.E. Torda, T. Huber, P. Krüger and W.F. van Gunsteren
  The GROMOS Biomolecular Simulation Package
  J. Phys. Chem. A 103 (1999) 3596-3607
- M. Christen, P.H. Hünenberger, D. Bakowies, R. Baron, R. Bürgi, D.P. Geerke, T.N. Heinz, M.A. Kastenholz, V. Kräutler, C. Oostenbrink, C. Peter, D. Trzesniak and W.F. van Gunsteren
  The GROMOS Software for Biomolecular Simulation: GROMOS 05
  J. Comput. Chem. 26 (2005) 1719-1751

The architecture and different functionalities of the current version of GROMOS, GROMOS 11, are described in the following papers:

- N. Schmid, C.D. Christ, M. Christen, A.P. Eichenberger and W.F. van Gunsteren
  Architecture, Implementation and Parallelisation of the GROMOS Software for Biomolecular Simulation
  Comp. Phys. Commun. 183 (2012) 890-903
- A.P.E. Kunz, J.R. Allison, D.P. Geerke, B.A.C. Horta, P.H. Hünenberger, S. Riniker, N. Schmid and W.F. van Gunsteren
  New Functionalities in the GROMOS Biomolecular Simulation Software
  J. Comput. Chem. 33 (2012) 340-353
- S. Riniker, C.D. Christ, H.S. Hansen, P.H. Hünenberger, C. Oostenbrink, D. Steiner and W.F. van Gunsteren
  Calculation of Relative Free Energies for Ligand-Protein Binding, Solvation and Conformational Transitions using the GROMOS Biomolecular Simulation Software
  J. Phys. Chem. B 115 (2011) 13570-13577
- N. Schmid, J.R. Allison, J. Dolenc, A.P. Eichenberger, A.P.E. Kunz, and W.F. van Gunsteren
  Biomolecular Structure Refinement using the GROMOS Simulation Software
  J. Biomol. NMR 51 (2011) 265-281
- A.P. Eichenberger, J.R. Allison, J. Dolenc, D.P. Geerke, B.A.C. Horta, K. Meier, C. Oostenbrink, N. Schmid, D. Steiner, D. Wang and W.F. van Gunsteren
  The GROMOS++ Software for the Analysis of Biomolecular Simulation Trajectories
  J. Chem. Theory Comput. 7 (2011) 3379-3390
- S.J. Bachmann, W.F. van Gunsteren
  On the compatibility of polarisable and non-polarisable models for liquid water
  Mol. Phys. 112 (2014) 2761-2780
- N. Hansen, F. Heller, N Schmid, W.F. van Gunsteren
  Time-averaged order parameter restraints in molecular dynamics simulations
  J. Biomol. NMR 60 (2014) 169-187

The GROMOS C++ code is documented in the code in the form of a doxygen documentation. It is accompanied by make files, etc. and by example files.

CHAPTER 4

Examples of application of GROMOS

During the past forty years the computer has taken an increasingly prominent position in science. This is due to the rapid increase of computer power. Every five to six years the ratio of performance to price has increased by a factor of ten. This development has paved the way for simulating in atomic detail a variety of physical processes on a computer. Computer simulation is a powerful tool to predict molecular properties that are inaccessible to experiments once the reliability of the molecular models, force fields and computational procedures has been established by comparison of simulated properties with known experimental ones. This may lead to the design of substances or molecules that possess specific properties useful in practical applications. Here, one may think of applications in drug or vaccine design, in protein engineering or in material science. The common approach to modelling a molecular system on a computer is a static one. For example, quantum calculations yield a particular charge distribution; Molecular Mechanics calculations yield one or a few minimum energy conformations of a molecule; on a graphics device molecules are studied in terms of fixed conformations.

However, a molecular system at room temperature is by no means of static character. A system of interacting atoms traverses multiple minima of the potential energy surface. One would like to know the multidimensional distribution function of all atomic coordinates and its development in time. This knowledge can never be complete. Only parts of configuration space can be searched for relevant low (free) energy conformations. The computer simulation technique of Molecular Dynamics provides the possibility to scan that part of configuration space that is accessible to the molecular system at the given temperature.

Static modelling techniques are completely inadequate to describe the properties of a system in a number of applications. Examples are the behaviour of liquid water and its influence on the conformation of a solute, and the calculation of quantities like entropy and free energy. The latter determine such properties as the binding strength of small drug molecules to large acceptor molecules, which is crucial in the process of drug design.

Dynamic modelling techniques are therefore a very promising tool in the field of (bio)molecular chemistry and physics. Below we sketch a few applications of the GROMOS software and force fields.

4.1. Analysis: Calculation of dielectric permittivity and relaxation time

The static dielectric permittivity $\epsilon(0)$ and the Debye relaxation time τ_D of a molecular liquid can be obtained from non-equilibrium MD simulations of the liquid in which a homogeneous static external electric field $\mathbf{E}^{ext}$ is switched on at $t = t_0$.[44] Upon switching on $\mathbf{E}^{ext}$ along the z-axis at $t = t_0$, the z-component P_z of the polarisation $\mathbf{P}$ will increase from its initial value $P_z(t_0)$, which values are Gaussian distributed around $P_z = 0$, to a steady-state value $P_z(t = \infty)$. For a Debye dielectric medium, this build-up will be exponential,

$$\langle P_z(t)\rangle_{t_0} = \langle P_z(t=\infty)\rangle_{t_0}\left[1 - e^{-(t-t_0)/\tau_P}\right]. \tag{4.1}$$

The value of $P_z(t = \infty)$ will be larger for larger E_z^{ext}, but different field strengths E_z^{ext} should yield the same τ_P, as long as E_z^{ext} is not too small and not too large. The static dielectric permittivity of the molecular model is then

$$\epsilon(0) = 1 + 4\pi\frac{P_z(t=\infty)}{E_z^{ext}} \tag{4.2}$$

and the Debye relaxation time is

$$\tau_D = \frac{\epsilon(0) + 2 + C_{rf}(\epsilon(0) - 1)}{3}\tau_P \tag{4.3}$$

in which C_{rf} is a constant depending on the dielectric permittivity ϵ_{cs} of the medium inside the cut-off sphere with radius R_{rf} and the dielectric continuum outside the cut-off sphere is characterised by a dielectric permittivity ϵ_{rf} and an inverse Debye screening length κ_{rf} .

The results for three different system sizes of a cubic box of liquid simple-point-charge (SPC) water are shown in Tab. 4.1 and Fig. 4.1. While the variation of $P_z(t=\infty)$ decreases with increasing system size due to better statistics, the average relaxation is independent of system size, and so are the values obtained for τ_D and $\epsilon(0)$.

Number of H_2O	$\epsilon(0)$	E_z^{ext}	τ_D	E_z^{ext}	τ_D
1024	63	0.03	6.1	0.05	6.0
5384	67	0.03	6.6	0.05	6.0
12800	64	0.03	6.3	0.05	6.0

TABLE 4.1. Calculated values for the relative static dielectric permittivity $\epsilon(0)$ and the Debye dielectric relaxation time τ_D (ps) at 298 K and 1 atm for water using three different system sizes and two different electric field strengths.[44] The electric field strengths (enm^{-2}) were chosen such that they are as large as possible while being in the linear-response regime.

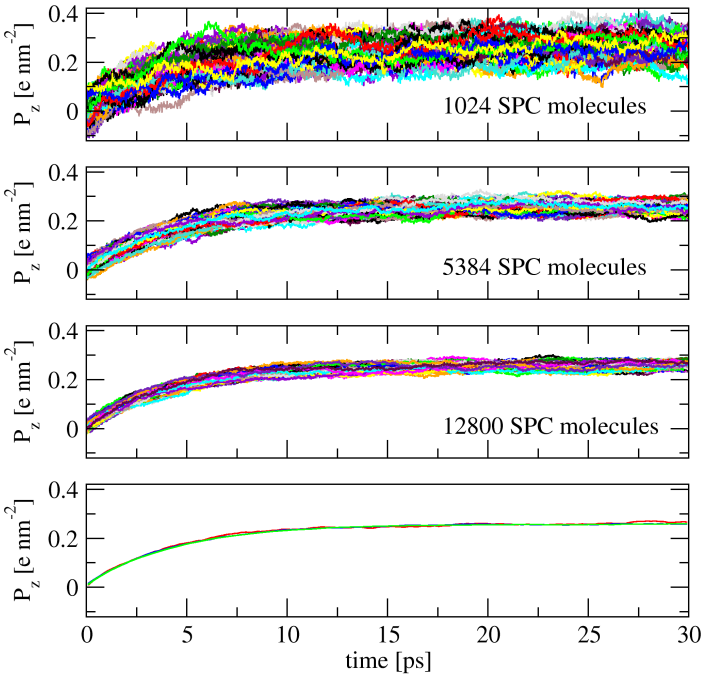


FIGURE 4.1. Polarisation $P_z(t)$ for 100 non-equilibrium MD simulations[44] of liquid water using three different system sizes, 1024, 5384 and 12800 SPC molecules, after switching on an electric field $E_z^{ext} = 0.05$ e nm^{-2} at $t_0 = 0$. The averages over the 100 trajectories are shown in red (1024 molecules), blue (5384 molecules) and green (12800 molecules) in the lowest panel.

4.2. Simulation of polypeptide folding using a polarisable solvent

Folding and unfolding of β-peptides has been studied extensively by molecular dynamics simulation. In these simulations, a non-polarisable model for the solvent, mostly methanol, was used. If a polarisable

solvent is used, the agreement with the experimental data from NMR is slightly improved, see Fig. 4.2. In the polarisable solvent the helical structure of the 7-residue β-peptide, which has a large dipole moment, is stabilised[45]. This means that the introduction of electronic polarisability into the solvent model appears of importance to a proper description of folding equilibria if these are determined by competing solute conformations that have different dipole moments.

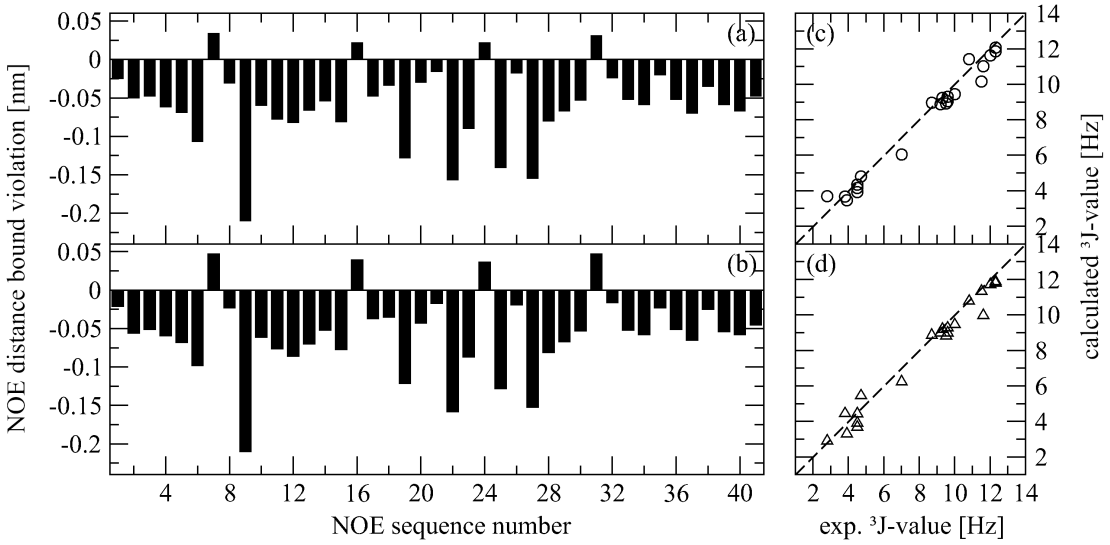


FIGURE 4.2. Comparison of r^{-6} averaged NOE distances and average ^3J-coupling constants as obtained from simulations (at 340 K, 1 atm) and experimental data of a 7-residue β-peptide.[45] Panels (a,c): in polarisable methanol (b,d): in non-polarisable methanol.

4.3. Properties of coarse-grained models for solvents: H_2O and co-solvents

The development of coarse-grained (CG) models that represent the important features of compounds is essential to overcome limitations in time scale and system size currently encountered in atomistic molecular dynamics simulations. Since the solvent interactions account for most of the computational effort in a biomolecular simulation, coarse-graining of the solvent model significantly enhances the efficiency of a simulation. In Fig. 4.3 and Fig. 4.4, some thermodynamic properties of mixtures of CG DMSO and CG MeOH with CG H_2O[38] are shown. Apart from the energy of mixing, the trends as function of mole fraction are reproduced. A change of DMSO-H_2O and MeOH-H_2O Lennard-Jones interaction would be sufficient to obtain negative values for ΔU_{mix}.

4.4. Enhancing the configurational sampling of ions

While configurational sampling of a liquid is relatively easy, due to the fact that it consists of many identical molecules that may exchange their position in space, the conformational sampling of a protein is much more difficult due to the connectivity of its covalent topology which impedes a fast exchange of atom positions. Sampling the configurational distribution of ions around a protein is a challenge that lies somewhere between these two cases. The sampling of ionic degrees of freedom can be considerably enhanced by using the technique of adiabatic decoupling the motion of the ionic degrees of freedom from that of the water solvent and then raising the temperature of the ions. The ionic diffusion turned out to be 15 times larger while keeping the distribution of the water molecules around the ions unaltered with respect to the standard temperature simulation, see Tab. 4.2.[46]

4.5. Calculation of protein-ligand binding free enthalpies

The relative free enthalpy of binding of different inhibitors to a protein can be obtained using enveloping distribution sampling (EDS), which is a computationally efficient alternative to the method of thermodynamic integration. In Fig. 4.5, the binding free enthalpies of three inhibitors of the protein

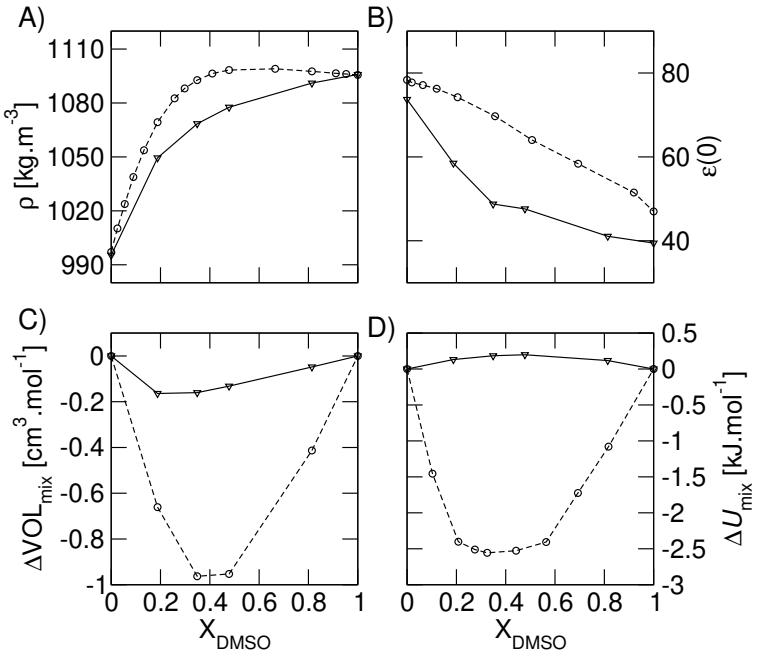


FIGURE 4.3. Thermodynamic properties of coarse-grained (CG) DMSO:H_2O mixtures at 298 K and 1 atm, as a function of the mole fraction of DMSO, X_{DMSO}, from MD simulations: (A) densities ρ, (B) dielectric permittivities $\epsilon(0)$, (C) excess volume of mixing ΔVOL_{mix} and (D) excess potential energy of mixing ΔU_{mix}. Experimental data are shown in solid lines and results of the CG simulations are in dashed lines.

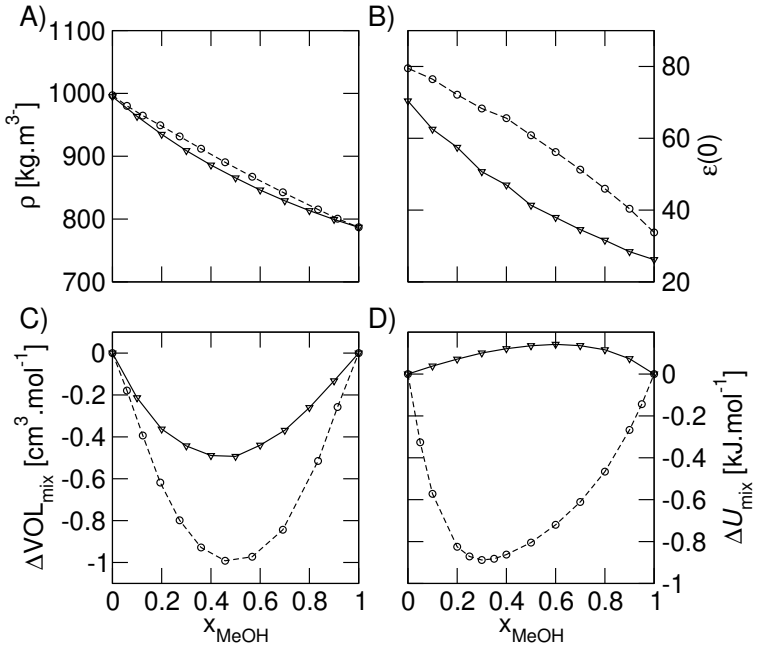


FIGURE 4.4. Thermodynamic properties of coarse-grained (CG) MeOH:H_2O mixtures at 298 K and 1 atm, as a function of the mole fraction of MeOH, X_{MeOH}, from MD simulations: (A) densities ρ, (B) dielectric permittivities $\epsilon(0)$, (C) excess volume of mixing ΔVOL_{mix} and (D) excess potential energy of mixing ΔU_{mix}. Experimental data are shown in solid lines and results of the CG simulations are in dashed lines.

s_m	s_T	$D_{Ca^{2+}}$ $[10^{-9}\mathrm{m^2s^{-1}}]$	$D_{SO_4^{2-}}$ $[10^{-9}\mathrm{m^2s^{-1}}]$	$\Delta g_{Ca^{2+}OW}\cdot 100$	$\Delta g_{SO_4^{2-}OW}\cdot 100$
1	1	1.06	1.23	0.00	0.00
	2	2.01	2.71	4.20	2.93
	3	4.11	6.03	8.59	6.71
	5	24.39	66.98	10.10	8.23
100	1	1.44	1.80	1.21	1.23
	2	13.22	16.89	1.25	1.20
	3	25.87	35.25	1.55	1.60
	5	51.64	74.56	1.60	2.11
200	1	1.23	1.46	1.34	1.11
	2	7.84	9.66	1.07	1.24
	3	15.27	18.87	1.45	1.40
	5	28.95	38.25	1.41	2.06
500	1	1.13	1.14	1.09	1.14
	2	4.22	4.48	1.18	1.18
	3	7.00	8.26	1.58	1.24
	5	13.19	15.83	1.47	1.97
1000	1	0.83	0.74	1.64	1.05
	2	2.33	2.51	1.75	1.19
	3	4.04	4.29	1.40	1.30
	5	7.43	7.96	1.70	1.54

TABLE 4.2. Configurational and dynamic properties of Ca^{2+} and SO_4^{2-} ions in aqueous solution from differently strong (s_m) adiabatically decoupled simulations in which the temperature of the ions is increased by different amounts (s_T). s_m: mass scaling factor, s_T: temperature scaling factor, D: diffusion coefficient, Δg: radial distribution difference.[46]

phenylethanolamine N-methyltransferase (PNMT) obtained using EDS and a GROMOS force field are compared to experimental data.[47] Excellent agreement with experiment is found.

4.6. Structure refinement based on NMR data

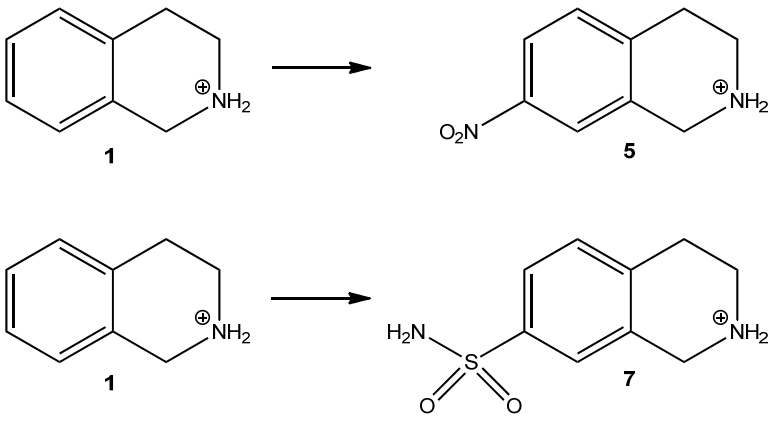
In structure refinement of proteins based on NMR data it is tried to find a single structure or a set of structures that reproduces the measured values of quantities such as NOE atom-atom distances bounds or ^{3}J-couplings. However, this is not always possible due to the presence of different conformers in solution. Such a case is illustrated in Fig. 4.6, where the set of NMR model structures deposited in the protein data bank for the peptide GCN4-p1 does not agree with the experimental NOE and ^{3}J-coupling data for this molecule. Using time-averaging refinement and a GROMOS force field the data can much better be reproduced[48,49].

4.7. Water configurations and mobility in the pore of a membrane protein

In Fig. 4.7, structures of the membrane protein OmpX embedded in a DMPC bilayer take from a simulation are shown[50], and Fig. 4.8 shows the water molecules trapped inside the β-barrel: a very stable salt-bridge and hydrogen-bond network exists in the barrel which inhibits a water flux.

4.8. Computer time required for MD simulation

Molecular dynamics computer simulations can be rather computer time demanding. In order to obtain an impression of the computing effort needed to simulate various biomolecular systems some benchmark data for GROMOS are given in Fig. 4.9.



Pert.	$\Delta G^{complex}$	ΔG^{free}	$\Delta\Delta G^{binding}$(calc)	$\Delta\Delta G^{binding}$(exp)
1-5	-4.0 ± 0.9	6.2 ± 0.1	-10.2 ± 1.0	-10.3
1-7	-352.0 ± 2.0	-343.6 ± 0.2	-8.4 ± 2.2	-9.6

FIGURE 4.5. Perturbation between inhibitors 1 and 5, and 1 and 7 of the protein PNMT in water for the bound (complex) and unbound (free) ligands with the corresponding free enthalpy differences and the resulting relative binding free enthalpies as obtained from EDS simulations.[47]

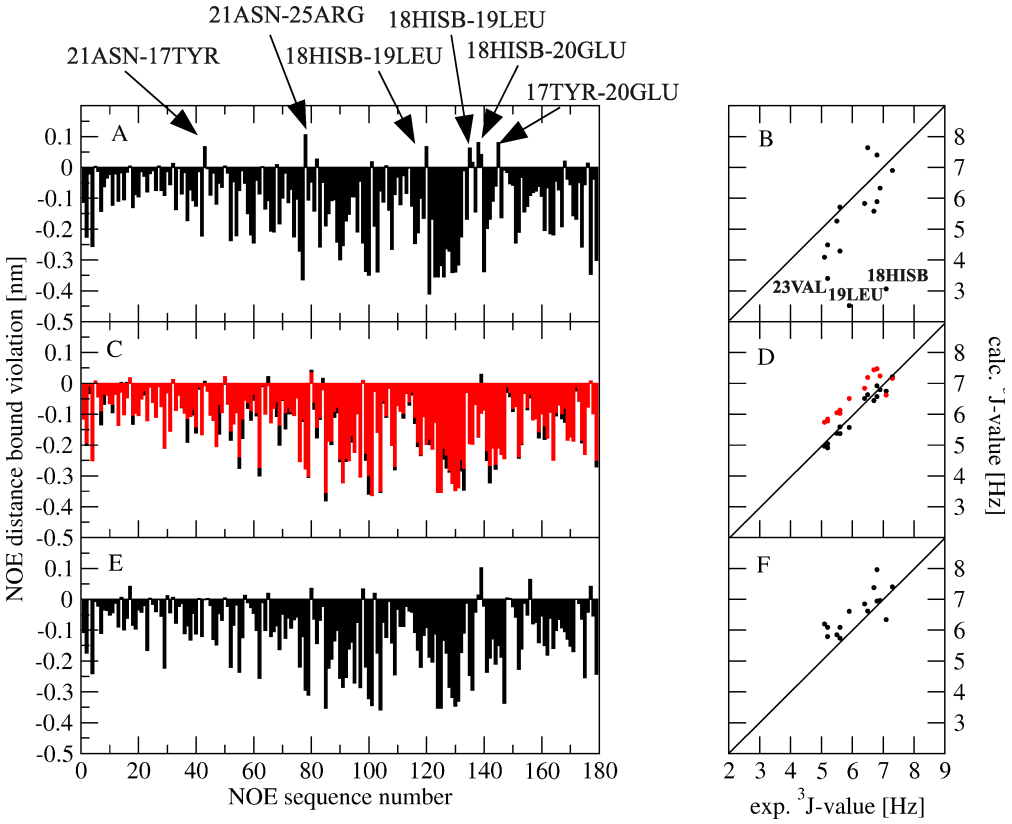


FIGURE 4.6. Deviations from the experimentally derived NOE upper distance bounds as a function of the NOE sequence number (left-hand panel, A) and comparison of the experimental and calculated ^{3}J(HN-HC$_\alpha$)-coupling constants (right-hand panel, B) for the 20 NMR model structures. The corresponding deviations (left-hand panel, C) and comparisons (right-hand panel, D) for the simulations in which time-averaged NOE distances (NOE_TAR) were restrained and either the ^3J-couplings were instantaneously restrained (3J_TAR) or a local-elevation biased sampling of torsional angles corresponding to ^3J-couplings were applied (3J_LE): NOE_TAR+3J_IR (black) and NOE_TAR+3J_LE (red). The corresponding deviations (left-hand panel, E) and comparisons (right-hand panel, F) for the weighted averages of the central members of the first ten conformational clusters of the NOE_TAR+3J_LE simulation. The 179 NOEs and 15 ^3J-couplings are defined in[48] .

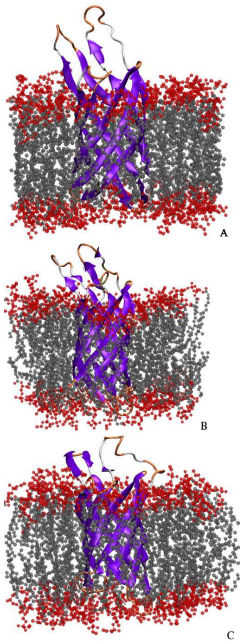


FIGURE 4.7. OmpX protein inserted in a lipid bilayer. A snapshot is shown at the beginning (a) and at the end of the simulations OmpX-DMPC-1 (b) and OmpX-DMPC-2 (c). The different colors indicate the secondary structure assignment. Lipid head groups are represented in red and lipid side chains in grey space-filling models.[50]

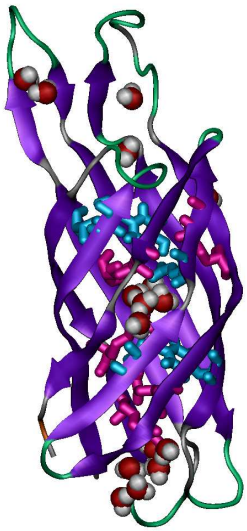


FIGURE 4.8. Water molecules trapped by hydrogen bonds inside the β-barrel for the OmpX-DHPC simulation[50] are shown as red and grey space-filling models. Residues for which intra-molecular hydrogen bonds are present for more than 90% or between 20 and 90% of the simulation time are drawn in pink or blue stick diagrams, respectively.

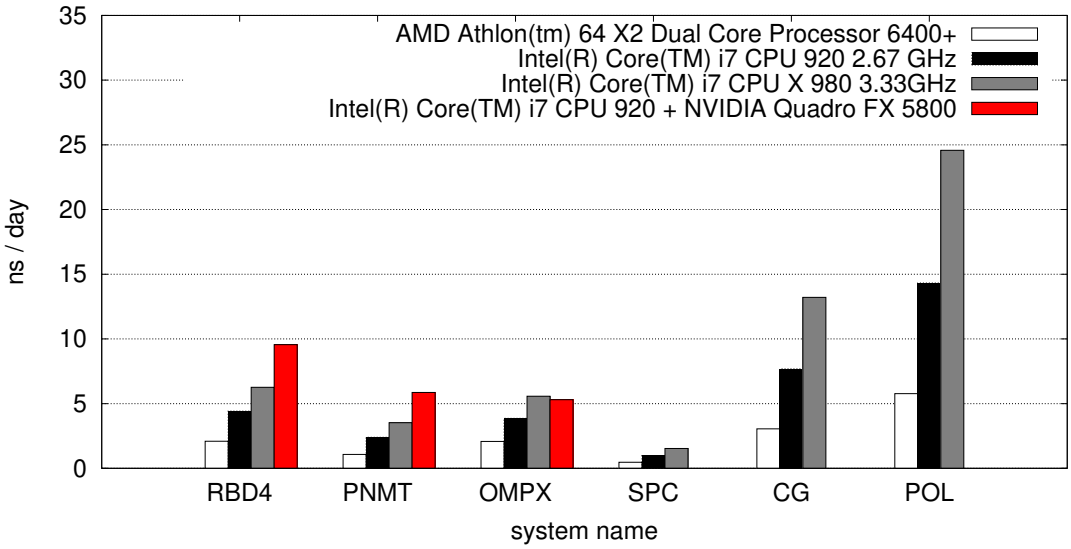


FIGURE 4.9. Benchmark results in nanoseconds per day for various biomolecular systems obtained on for different workstation class computers. Simulation settings: 2 fs integration time-step, twin-range cutoff scheme with cutoff values of 0.8 nm (short range) and 1.4 nm (long range). The pair list and long-range interactions were calculated every 5 steps. Systems: *RBD4*: polypyrimidine tract binding protein - RNA binding domain 4 bound to RNA CUCUCU, 1028 solute atoms, 5411 SPC solvent molecules. *PNMT*: phenylethanolamine N-methyltransferase (PNMT) protein with bound inhibitor, 2677 solute atoms, 9909 SPC solvent molecules. *OMPX*: Outer membrane protein X in a DPPC bilayer, 6322 solute atoms, 6559 SPC solvent molecules. *SPC*: 12800 SPC molecules treated as solute, 38400 atoms. *CG*: 2560 Coarse-grained water molecules, corresponding to 12800 fine-grained water molecules, 5120 solute atoms. *POL*: A beta heptapeptide in polarisable methanol, 63 solute atoms, 1095 polarisable MeOH molecules.

CHAPTER 5

Limitations of GROMOS

When applying MD to simulate a particular system, a number of preliminary questions have to be answered and choices related to the level of accuracy must be made. The *assumptions* and *approximations* that are made with respect to the molecular model and the computational procedures will determine the accuracy of the results obtained. Clearly there are *limitations* to the usefulness of application of simulation techniques. These will be briefly discussed below.

1. Since Newton's equations of motion are solved in an MD simulation, a *classical description* must be appropriate for the phenomena to be studied. Generally, when considering a molecular system at room temperature, quantum effects will not play a significant role as long as no covalent bonds are broken, etc.
2. With the use of modern computers the length of a MD simulation extends from a few tens of picoseconds up till hundreds of nanoseconds, depending on the size of the system. This means that the *time scale of a process* that can be simulated at the atomic level, is limited. For the simulation of activated processes special techniques are available, which require the pathway of the process to be known.
3. Only a limited *number of atoms* can be simulated, typically up till 10^5 atoms. The question is how many atoms are essentially involved in the phenomena to be studied. Atomic degrees of freedom that are not essential for an adequate description of the phenomenon being studied, may be removed by applying constraints, or stochastic techniques in combination with potentials of mean force, or the extended wall region boundary condition.
4. Last but not least, the *interaction function* or *force field* that is used will determine the accuracy of the obtained simulation results. A great variety of molecular models and force fields for molecular systems under various conditions is available. The choice of a particular force field should depend on the system properties one is interested in. Some applications require more refined force fields than others. Moreover, there should be a balance between the level of accuracy or refinement of different parts of a molecular model. Otherwise the computer effort put into a very detailed and accurate part of the calculation may easily be wasted due to the distorting effects of the crude parts of the model.

Although computer simulation is a very powerful technique to study the properties of molecular systems at the atomic level, one should bear in mind the various assumptions and approximations that are made and be aware of the limitations of the method.

Bibliography

[1] J. Hermans, H.J.C. Berendsen, W.F. van Gunsteren, and J.P.M. Postma. A Consistent Empirical Potential for Water-Protein Interactions. *Biopolymers*, 23:1513–1518, 1984.

[2] W.F. van Gunsteren and M. Karplus. Effect of Constraints on the Dynamics of Macromolecules. *Macromolecules*, 15:1528–1544, 1982.

[3] L.J. Smith, A.E. Mark, C.M. Dobson, and W.F. van Gunsteren. Comparison of MD simulations and NMR experiments for hen lysozyme: Analysis of local fluctuations, cooperative motions and global changes. *Biochemistry*, 34:10918–10931, 1995.

[4] X. Daura, A.E. Mark, and W.F. van Gunsteren. Parametrization of Aliphatic CHn United Atoms of GROMOS96 Force Field. *J. Comput. Chem.*, 19:535–547, 1998.

[5] L.D. Schuler and W.F. van Gunsteren. On the Choice of Dihedral Angle Potential Energy Functions for n-Alkanes. *Mol. Simul.*, 25:301–319, 2000.

[6] L.D. Schuler, X. Daura, and W.F. van Gunsteren. An Improved GROMOS96 Force Field for Aliphatic Hydrocarbons in the Condensed Phase. *J. Comput. Chem.*, 22:1205–1218, 2001.

[7] I. Chandrasekhar, M. Kastenholz, R.D. Lins, C. Oostenbrink, L.D. Schuler, D.P. Tieleman, and W.F. van Gunsteren. A consistent potential energy parameter set for lipids: Dipalmitoylphosphatidylcholine as a benchmark of the GROMOS96 45A3 force field. *Eur. Biophys. J.*, 32:67–77, 2003.

[8] C. Oostenbrink, A. Villa, A.E. Mark, and W.F. van Gunsteren. A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. *J. Comput. Chem.*, 25:1656, 2004.

[9] T.A. Soares, P.H. Hünenberger, M.A. Kastenholz, V. Kräutler, T. Lenz, R.D. Lins, C. Oostenbrink, and W.F. van Gunsteren. An Improved Nucleic-Acid Parameter Set for the GROMOS Force Field. *J. Comput. Chem.*, 26:725–737, 2005.

[10] R.D. Lins and P.H. Hünenberger. A new GROMOS force field for hexopyranose-based cardohydrates. *J. Comput. Chem.*, 26:1400–1412, 2005.

[11] M. Winger, A.H. de Vries, and W.F. van Gunsteren. Force-field dependence of the conformational properties of alpha,omega-dimethoxypolyethylene glycol. *Mol. Phys.*, 107:1313–1321, 2009.

[12] D. Poger, W.F. van Gunsteren, and A.E. Mark. A new force field for simulating phosphatidylcholine bilayers. *J. Comput. Chem.*, 31:1117–1125, 2010.

[13] H. Hansen and P.H. Hünenberger. A reoptimized GROMOS force field for hexopyranose-based carbohydrates accounting for the relative free energies of ring conformers, anomers, epimers, hydroxymethyl rotamers and glycosidic linkage conformers. *J. Comput. Chem.*, 32:998–1032, 2011.

[14] N. Schmid, A. P. Eichenberger, A. Choutko, S. Riniker, M. Winger, A.E.. Mark, and W.F. van Gunsteren. Definition and testing of the GROMOS force-field versions: 54A7 and 54B7. *Eur. Biophys. J.*, 40:843–856, 2011.

[15] M.M. Reif, P.H. Hünenberger, and C. Oostenbrink. New interaction parameters for charged amino acid side chains in the GROMOS force field. *J. Chem. Theory Comput.*, 8:3705–3723, 2012.

[16] W.F. van Gunsteren, S.R. Billeter, A.A. Eising, P.H. Hünenberger, P. Krüger, A.E. Mark, W.R.P. Scott, and I.G. Tironi. *Biomolecular Simulation: The GROMOS96 Manual and User Guide*. Vdf Hochschulverlag AG an der ETH Zürich, Zürich, Switzerland, 1996.

[17] D.P. Geerke and W.F. van Gunsteren. Force Field Evaluation for Biomolecular Simulation: Free Enthalpies of Solvation of Polar and Apolar Compounds in Various Solvents. *ChemPhysChem*, 7:671–678, 2006.

[18] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, and J. Hermans. Interaction models for water in relation to protein hydration. In B. Pullman, editor, *Intermolecular Forces*, pages 331–342. Reidel, Dordrecht, 1981.

[19] A. Glättli, X. Daura, and W.F. van Gunsteren. Derivation of an improved SPC model for liquid water: SPC/A and SPC/L. *J. Chem. Phys.*, 116:9811–9828, 2002.

[20] R. Walser, A.E. Mark, W.F. van Gunsteren, M. Lauterbach, and G. Wipff. The effect of force-field parameters on properties of liquids: Parametrization of a simple three-site model for methanol. *J. Chem. Phys.*, 112:10450–10459, 2000.

[21] D.P. Geerke, C. Oostenbrink, N.F.A. van der Vegt, and W.F. van Gunsteren. An Effective Force Field for Molecular Dynamics Simulations of Dimethyl Sulfoxide and Dimethyl Sulfoxide-Water Mixtures. *J. Phys. Chem. B*, 108:1436, 2004.

[22] I.G. Tironi and W.F. van Gunsteren. A molecular dynamics simulation study of chloroform. *Mol. Phys.*, 83:381–403, 1994.

[23] I.G. Tironi, P. Fontana, and W.F. van Gunsteren. A molecular dynamics simulation study of liquid carbon tetrachloride. *Mol. Simul.*, 18:1–11, 1996.

[24] L.J. Smith, H.J.C. Berendsen, and W.F. van Gunsteren. Computer Simulation of Urea-Water Mixtures: A Test of Force Field Parameters for Use in Biomolecular Simulations. *J. Phys. Chem. B*, 108:1065–1071, 2004.

[25] P.J. Gee and W.F. van Gunsteren. Acetonitrile revisited: a molecular dynamics study of the liquid phase. *Mol. Phys.*, 104:477–483, 2006.

[26] H. Yu, D.P. Geerke, H. Liu, and W.F. van Gunsteren. Molecular dynamics simulations of liquid methanol and methanol-water mixtures with polarizable models. *J. Comput. Chem.*, 27:1494–1504, 2006.

[27] D.P. Geerke and W.F. van Gunsteren. The performance of non-polarizable and polarizable force-field parameter sets for ethylene glycol in molecular dynamics simulation of the pure liquid and its aqueous mixtures. *Mol. Phys.*, 105:1861–1881, 2007.
[28] D.P. Geerke and W.F. van Gunsteren. On the calculation of atomic forces in classical simulation using the charge-on-spring method to explicitly treat electronic polarisation. *J. Chem. Theory Comput.*, 3:2128–2137, 2007.
[29] D.P. Geerke and W.F. van Gunsteren. Calculation of the free energy of polarization: quantifying the effect of explicitly treating electronic polarization on the transferability of force-field parameters. *J. Phys. Chem. B*, 111:6425–6436, 2007.
[30] A.P. Kunz and W.F. van Gunsteren. Development of a non-linear classical polarisation model for liquid water and aqueous solutions: COS/D. *J. Phys. Chem. A*, 113:11570–11579, 2009.
[31] Z. Lin, A.P. Kunz, and W.F. van Gunsteren. A one-site polarizable model for liquid chloroform: COS/C. *Mol. Phys.*, 108:1749–1757, 2010.
[32] A.P.E. Kunz, A.P. Eichenberger, and W.F. van Gunsteren. A simple, efficient polarisable molecular model for liquid carbon tetrachloride. *Mol. Phys.*, 109:365–372, 2011.
[33] O.M. Szklarczyk, S.J. Bachmann, and W.F. van Gunsteren. A polarisable empirical force field for molecular dynamics simulation of liquid hydrocarbons. *J. Comput. Chem.*, 35:789–801, 2014.
[34] S.J. Bachmann and W.F. van Gunsteren. Polarisable model for DMSO and DMSO-water mixtures. *J. Phys. Chem. B*, 118:10175–10186, 2014.
[35] S.J. Bachmann and W.F. van Gunsteren. On the compatibility of polarisable and non-polarisable models for liquid water. *Mol. Phys.*, 112:2761–2780, 2014.
[36] S.J. Bachmann and W.F. van Gunsteren. An improved polarisable water model for use in biomolecular simulation. *J. Chem. Phys.*, 141:22D515, 2014.
[37] Z. Lin, S.J. Bachmann, and W.F. van Gunsteren. GROMOS polarisable charge-on-spring models for liquid urea: COS/U and COS/U2. *J. Chem. Phys.*, 142:094117, 2015.
[38] S. Riniker and W.F. van Gunsteren. A simple, efficient polarisable coarse-grained water model for molecular dynamics simulations. *J. Chem. Phys.*, 134:084110, 2011.
[39] O. Szklarczyk, E. Arvaniti, and W.F. van Gunsteren. Polarisable coarse-grained models for molecular dynamics simulation of liquid cyclohexane. *J. Comput. Chem.*, 36:1311–1321, 2015.
[40] A.P. Eichenberger, W. Huang, S. Riniker, and W.F. van Gunsteren. A supra-atomic coarse-grained GROMOS force field for aliphatic hydrocarbons in the liquid phase. *J. Chem. Theory Comput.*, 11:2925–2937, 2015.
[41] J.R. Allison, S. Riniker, and W.F. van Gunsteren. Coarse-grained models for the solvents dimethyl sulfoxide, chloroform and methanol. *J. Chem. Phys.*, 136:054505, 2012.
[42] W. Huang, S. Riniker, and W.F. van Gunsteren. Rapid sampling of folding equilibria of β-peptides in methanol using a supramolecular solvent model. *J. Chem. Theory Comput.*, 10:2213–2223, 2014.
[43] W. Huang, N. Hansen, and W.F. van Gunsteren. On the use of a supramolecular coarse-grained model for the solvent in simulations of the folding equilibrium of an octa-β-peptide in MeOH and H2O. *Helv. Chim. Acta*, 97:1591–1605, 2014.
[44] S. Riniker, A.P.E. Kunz, and W.F. van Gunsteren. On the calculation of the dielectric permittivity of molecular models in the liquid phase. *J. Chem. Theory Comput.*, 7:1469–1475, 2011.
[45] Z. Lin, N. Schmid, and W.F. van Gunsteren. The effect of using a polarizable solvent model upon the folding equilibrium of different beta-peptides. *Mol. Phys.*, 109:493–506, 2011.
[46] A.P.E. Kunz and W.F. van Gunsteren. Enhancing the configurational sampling of ions in aqueous solution using adiabatic decoupling with translational temperature scaling. *J. Phys. Chem. B*, 115:2931–2936, 2011.
[47] S. Riniker, C.D. Christ, N. Hansen, A.E. Mark, P.C. Nair, and W.F. van Gunsteren. Comparison of enveloping distribution sampling and thermodynamic integration to calculate binding free energies of phenylethanolamine N-methyltransferase inhibitors. *J. Chem. Phys.*, 135:024105, 2011.
[48] J. Dolenc, J.H. Missimer, M.O. Steinmetz, and W.F. van Gunsteren. Methods of NMR structure refinement: molecular dynamics simulations improve the agreement with measured NMR data of a C-terminal peptide of GCN4-p1. *J. Biomol. NMR*, 47:221–235, 2010.
[49] J.H. Missimer, J. Dolenc, M.O Steinmetz, and W.F. van Gunsteren. Exploring the trigger sequence of the GCN4 coiled-coil: biased molecular dynamics resolves apparent inconsistencies in NMR measurements. *Protein Sci.*, 19:2462–2474, 2010.
[50] A. Choutko, A. Glättli, C. Fernández, C. Hilty, K. Wüthrich, and W.F. van Gunsteren. Membrane protein dynamics in different environments: simulation study of the outer membrane protein X in a lipid bilayer and in a micelle. *Eur. Biophys. J.*, 40:39–58, 2010.

The GROMOS Software for (Bio)Molecular Simulation

Volume 2: Algorithms and Formulae for Modelling of Molecular Systems

January 9, 2021

Contents

CHAPTER 1

Introduction

In this volume, the molecular model, the molecular interactions, and the computational procedures used in GROMOS are described. Chap. 2 discusses the basic choices in the definition of a molecular model in a general way. The application of these choices within GROMOS are outlined in Chap. 3. Chap. 4 discusses various aspects of the spatial boundary conditions implemented in GROMOS. Chaps. 5, 6 and 7 deal with the description of molecular interactions divided into covalent interactions (Chap. 5), nonbonded van der Waals interactions (Chap. 6) and nonbonded electrostatic interactions (Chap. 7). Chap. 8 describes coarse-grained interactions between supra-molecular particles representing sets of identical molecules. Chaps. 9 and 10 deal with geometrical boundary conditions, which may be either imposed on the system in a soft way, using restraints (Chap. 9) or enforced in a hard way, defined as constraints (Chap. 10). The next three chapters involve searching and sampling of conformational space using energy minimization (Chap. 11), molecular dynamics (Chap. 12), or stochastic dynamics (Chap. 13). Chap. 14 deals with the calculation of free-energy differences and QM/MM simulations are described in Chap. 15. In Chap. 16 the implementation of replica exchange methods in GROMOS is discussed. In Chap. 17 all contributions to the molecular forces and the virial due to the interactions described in Chaps. 5 - 8 are collected and Chap. 18 forms an appendix with mathematical details on the conversion between different types of force constants.

CHAPTER 2

Basic choices in the definition of a molecular model

2.1. Introduction

The four *basic choices*[1–3] involved in the definition of a molecular model are (Fig. 2.1) :

A. Choice of the *degrees of freedom*[3,4] (Sec. 2.2) :
What are the variables of the model, *i.e.* those uniquely defining a system configuration in this model. Most commonly, this choice is equivalent to that of the elementary particles, *i.e.* smallest entities, considered by the model, *i.e.* the model resolution. The selected degrees of freedom are treated explicitly by the model. In contrast, degrees of freedom internal to the selected elementary particles are handled implicitly.

B. Choice of the description of the *interaction*[4] (Sec. 2.3) :
What is the Hamiltonian operator for quantum-mechanical degrees of freedom or the Hamiltonian function for classical degrees of freedom describing the interaction and kinetic energy associated with the selected degrees of freedom. This interaction must incorporate the mean effect of the implicit degrees of freedom on the explicit ones.

C. Choice of method for the *configuration generation*[5,6] (Sec. 2.4) :
What method will be used to generate configurations of the system *i.e.* values for the variables defining the degrees of freedom of the model, generally based on information from the corresponding interaction. The selection of a method, e.g., MC, MD or SD, affects the properties of the resulting set of configurations in terms of structural, thermodynamic and dynamic properties.

D. Choice of the *boundary conditions*[7] (Sec. 2.5) :
What are the global restrictions imposed on the system during the configuration generation. These restrictions may represent the interface of the internal degrees of freedom of the system to the outside world and are typically of a spatial/geometric, thermodynamic or alchemical nature.

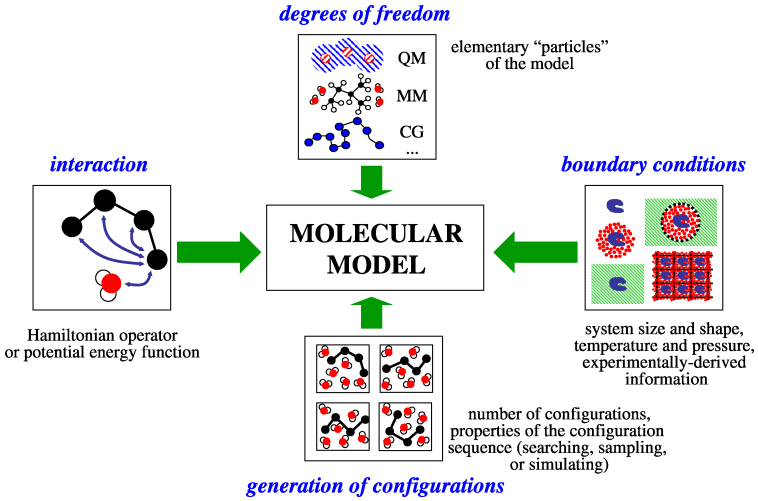


FIGURE 2.1. Four basic choices involved in the definition of a molecular model

Two important considerations typically precede the definition of a molecular model. First, most molecular models result in equations that are too complex or too numerous for an analytical solution. They must thus be solved numerically on a computer. Because the computing power is limited, one is always compelled to strike a balance between the required accuracy of the process or property of interest, the quality of the

models and the size of the relevant, accessible configurational space of the system that is to be probed in a tractable amount of computer time.

2.2. Choice of degrees of freedom

The possible physical choices of degrees of freedom in the context of the modelling of molecular systems, from the highest to the lowest resolution, can be broadly classified as[4] (Fig. 2.2) :

A. Nuclei and electrons of atoms
B. Nuclei and valence electrons of atoms
C. Nuclei and (valence) electrons of solute atoms
D. Atoms
E. United-atoms
F. Solute atoms
G. Beads representing atom groups
H. Rigid bodies representing molecules
I. Local properties of finite volume elements

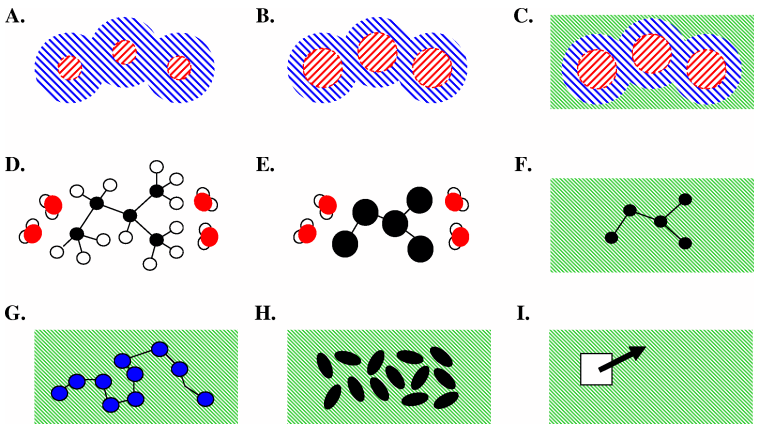


FIGURE 2.2. Broad classification of the possible levels of resolution or choice of the degrees of freedom in the modelling of molecular systems.

Each lower level of resolution turns a set of degrees of freedom that were handled explicitly at the previous level into implicit degrees of freedom at the next level. Models at the levels of resolution of Pts. A-C are commonly referred to as *Quantum Mechanical* (QM) models. Models at the levels of resolution of Pts. D-F are commonly referred to as *Atomistic* (AA for All Atoms) models. Models at the level of resolution of Pt. G are commonly referred to as *Coarse Grained* (CG) models. A number of *hybrid methods* attempt to merge different levels of resolution for different parts of a molecular system.

The selected level of resolution determines what should be referred to as the *configuration* or microstate of a corresponding system of $\mathcal{N}_d$ degrees of freedom. At the levels of resolution of Pts. A-C, a *quantum-mechanical* description is the most appropriate. In this case, a configuration of a system of particles is defined in the position representation by a wavefunction $\Psi \doteq \Psi(\boldsymbol{r})$, where $\boldsymbol{r}$ is the $\mathcal{N}_d$-dimensional vector characterizing the $\mathcal{N}_d = 3\mathcal{N}_a$ positional degrees of freedom associated with the $\mathcal{N}_a$ quantum-mechanical particles considered. At the levels of resolution of Pts. D-G, a *classical* description is the most appropriate. In this case, a configuration of a system of particles is defined by two $\mathcal{N}_d$-dimensional vectors $\boldsymbol{q}$ and $\boldsymbol{p_q}$, containing $\mathcal{N}_d = 3\mathcal{N}_a$ generalized coordinates and momenta associated with the $\mathcal{N}_a$ classical particles considered, respectively. In the special case where a Cartesian coordinate system is adopted, $\boldsymbol{q}$ is written as $\boldsymbol{r}$ and contains the components of the Cartesian particle coordinates as triplets, while $\boldsymbol{p_q}$ is written $\boldsymbol{p}$ and contains the components of the Cartesian particle momenta as triplets. This situation may be noted

$$\boldsymbol{r}^N \doteq \{\mathbf{r}_i \mid i = 1..\mathcal{N}_a\} \quad \text{and} \quad \boldsymbol{p}^N \doteq \{\mathbf{p}_i \mid i = 1..\mathcal{N}_a\} \ , \tag{2.1}$$

where $\mathbf{r}_i$ and $\mathbf{p}_i$ are the three-dimensional Cartesian coordinate and momentum vectors of a particle i. For simplicity and unless otherwise specified, a Cartesian coordinate description will be used systematically throughout this manual. At the level of resolution of Pt. H, there are twelve degrees of freedom per rigid

molecule (position, orientation, translational velocity, rotational velocity). Finally, at the level of resolution of Pt. I, a *continuous-material* description is adopted and the number of degrees of freedom per volume element depends on the (scalar or vector) properties that are considered.

In many cases, the choice of the degrees of freedom included in a model is not physical, *i.e.* matching exactly the real degrees of freedom of a molecular system. Important examples include :

J. The *Car Parrinello* (CP) approach[8] :
Here the electronic degrees of freedom in the form of the coefficients of the electronic wavefunction are treated as classical degrees of freedom and evolved in time simultaneously with the classically treated nuclei.

K. The *Charge On Spring* (COS) approach[9] :
Here, the instantaneous dipoles associated with electronic polarization effects are accounted for by means of artificial point charges connected to the atoms by means of harmonic springs.

L. The *Path Integral* (PI) approach[10] :
Here, a system of particles treated at the classical level is used to emulate the statistical-mechanical properties of a corresponding quantum-mechanical system a single quantum-mechanical particle being assigned to a given number of the classical particles, connected by harmonic springs in a ring topology.

M. Various *extended-system* methods :
Here, one or a few additional degrees of freedom are added to those of the physical system and evolved simultaneously, *e.g.* thermostat or barostat variables,[7,11] force-field parameters to be refined,[12] atomic charges to account for polarization effects[9] as in the fluctuating charge model, Hamiltonian coupling parameters.

N. Ensemble propagation approaches[5,13] :
Here the degrees of freedom do not correspond to those of a single molecular system, but to the coefficients of an ensemble probability distribution of such systems (as projected in a given basis set).[13]

The choice of the degrees of freedom or particles that are handled explicitly in the model automatically implies a definition of the implicit degrees of freedom, the internal degrees of freedom of the selected particles that are not considered explicitly in the model.

The possibilities in the domain of application of the GROMOS package in terms of degrees of freedom are discussed in Sec. 3.2.

2.3. Choice of the description of the interaction

When the degrees of freedom of the model are to be treated according to the laws of quantum mechanics, the interaction associated with the selected degrees of freedom is described by a *Hamiltonian operator* $\hat{\mathcal{H}}$. Assuming that this operator is constant *i.e.* not explicitly dependent on any external parameter or on time, and that all elementary, subatomic, particles are explicitly included in the model, the expectation value $\langle\hat{\mathcal{H}}\rangle$ of the Hamiltonian operator in terms of the wavefunction $\Psi \doteq \Psi(\boldsymbol{r})$ representing a system configuration represents the total, kinetic plus potential, energy of the system in this configuration, *i.e.*

$$\langle\hat{\mathcal{H}}\rangle[\Psi] \doteq \langle\Psi \mid \hat{\mathcal{H}} \mid \Psi\rangle \ , \tag{2.2}$$

where the square brackets indicate a functional dependence and it is assumed that Ψ is normalized. The Hamiltonian operator is itself the sum of a *kinetic energy operator* $\hat{\mathcal{K}}$ and a *potential energy operator* $\hat{\mathcal{V}}$, *i.e.*

$$\hat{\mathcal{H}} \doteq \hat{\mathcal{K}} + \hat{\mathcal{V}} \tag{2.3}$$

with

$$\hat{\mathcal{K}} \doteq -\sum_{i=1}^{\mathcal{N}_a} \frac{\hbar^2}{2m_i}\nabla_i^2 \ , \tag{2.4}$$

where m_i is the mass of particle i and $\hbar \doteq (2\pi)^{-1}h$, h being Planck's constant. The potential energy operator accounts for the proper interaction between the different quantum-mechanical particles. When the quantum-mechanical degrees of freedom included in the model do not encompass all particles in the system, the interaction may still tentatively be formulated using an effective Hamiltonian operator that encompasses the mean effect of the implicit degrees of freedom on the explicit ones, e.g. as in the use of pseudo-potentials

accounting for the implicit core electrons or of effective solvation terms for the implicit solvent in the potential energy operator.

When the degrees of freedom of the model are to be treated according to the laws of classical mechanics, the interaction associated with the selected degrees of freedom is described by a *Hamiltonian function* $\mathcal{H}$. Assuming that this function is constant, i.e. not explicitly dependent on any external parameter or on time, that all elementary particles or atoms are explicitly included in the model, and using a Cartesian coordinate system, the Hamiltonian function can be written $\mathcal{H}(\boldsymbol{r}^N, \boldsymbol{p}^N)$, and its value is equal to the total, kinetic plus potential, energy of the system in the configuration $(\boldsymbol{r}^N, \boldsymbol{p}^N)$. The Hamiltonian function is the sum of a *kinetic energy function* $\mathcal{K}$ that solely depends on the Cartesian momenta of the particles and a *potential energy function* $\mathcal{V}$ that solely depends on the Cartesian coordinates of the particles, *i.e.*

$$\mathcal{H}(\boldsymbol{r}^N, \boldsymbol{p}^N) \doteq \mathcal{K}(\boldsymbol{p}^N) + \mathcal{V}(\boldsymbol{r}^N) \tag{2.5}$$

with

$$\mathcal{K}(\boldsymbol{p}^N) \doteq \sum_{i=1}^{\mathcal{N}_a} \frac{1}{2m_i} \mathbf{p}_i^2 \ . \tag{2.6}$$

When a generalized coordinate system is employed, the kinetic energy is in the general case a function of both coordinates and momenta. The potential energy function accounts for the proper interaction between the different classical particles. When the classical degrees of freedom included in the model do not encompass all particles in the system, the interaction may still tentatively be formulated using an effective Hamiltonian function that encompasses the mean effect of the implicit degrees of freedom on the explicit ones. According to classical statistical mechanics, the ensemble properties of the reduced system will be identical to those of a corresponding all-particle system if the potential energy function involved in the effective Hamiltonian is defined as a *potential of mean force*, *i.e.* a potential energy term or function, the derivative of which is equal to the force on the explicit degrees of freedom ensemble-averaged over the compatible values of the implicit degrees of freedom. In this situation, the notation $\overline{\mathcal{V}}$ (potential of mean force) instead of $\mathcal{V}$ (all-particle potential energy term or function) will sometimes be used when the distinction is important.

Finally, when the degrees of freedom of the model are to be treated according to a continuous-material description, the interaction is typically described in terms of *gradients* of specific intensive properties and *conservation equations* for specific extensive properties.

When modelling aims at emulating the physical behaviour of a molecular system as closely as possible, the approximations employed to define the interaction should be as representative as possible for the real physical situation. In some cases, however, alterations are performed, typically in the form of extra unphysical and possibly time-dependent terms included into the potential energy function to serve specific purposes, *e.g.* enhanced probing of the configurational space, inclusion of experimental data as a boundary condition. The possibilities in the domain of application of the GROMOS package in terms of interactions are discussed in Sec. 3.3.

2.4. Choice of method for configuration generation

The possible choices of methods to generate configurations of a molecular system can be broadly classified as:[6]

A. *Search methods*:
   Generation of a set of relevant configurations within the configurational space of the system, typically configurations representing local energy minima, without any further requirement on the distribution or time sequence of these configurations.
B. *Sampling methods*:
   Generation of a set of system configurations obeying a well-defined probability distribution in terms of energy, a configurational ensemble.
C. *Simulation methods*:
   Generation of a time sequence of system configurations obeying a particular probability distribution both governed by the chosen equation of motion.

Both sampling and simulation methods permit the evaluation of *thermodynamic properties* from the generated configurational ensemble. Only simulation methods permit the evaluation of *dynamical properties* from the generated trajectory.

At the quantum-mechanical level assuming that the Hamiltonian operator is constant, the simulation of a molecular system involves the integration of the *time-dependent Schrödinger equation*

$$\hat{\mathcal{H}}\Psi(\boldsymbol{r},t) = i\hbar\frac{\partial}{\partial t}\Psi(\boldsymbol{r},t) \ , \tag{2.7}$$

where $\Psi(\boldsymbol{r},t)$ represents the instantaneous state of the system at time t. Integrating this equation numerically forward in time on a computer to simulate quantum-dynamical behaviour of a molecular system is called *quantum molecular dynamics* (QMD) simulation. Two simplifying approximations are often made in QMD : (i) A molecule is built up from just two types of particles, nuclei with negligible size and irrelevant internal structure and electrons, the motion of which can be separated using the Born-Oppenheimer approximation. Thus, a complex molecular system can be described as a system of point masses moving in an effective potential field; (ii) Some particles, e.g. the nuclei, obey the laws of classical mechanics. This is a reasonable assumption at room temperature and for all but the lightest atoms. For the latter the path-integral formalism can be used to obtain the quantum equilibrium distribution using classical equations of motion.

At the classical level, using a Cartesian coordinate system, the simulation of a molecular system involves the integration of the *Newtonian equations of motion*

$$\dot{\boldsymbol{p}}^N(t) = -\nabla\mathcal{V}(\boldsymbol{r}^N(t)) \quad \text{and} \quad \dot{\boldsymbol{r}}^N(t) = \underline{\mathbf{m}}^{-1}\boldsymbol{p}^N(t) \tag{2.8}$$

where $\underline{\mathbf{m}}$ is a $\mathcal{N}_d$-dimentional (diagonal) mass matrix of a system of $\mathcal{N}_a$ particles containing the mass of the atoms by triplets along its diagonal and $(\boldsymbol{r}^N(t),\boldsymbol{p}^N(t))$ represents the instantaneous configuration of the system at time t. These equations represent a particular case of the more general Hamiltonian equations of motion in the special case of a Cartesian coordinate system. Two relevant quantities entering in Eq. 2.8 are the $\mathcal{N}_d$-dimensional force and velocity vectors, *i.e.*

$$\boldsymbol{v}^N(t) \doteq \underline{\mathbf{m}}^{-1}\boldsymbol{p}^N(t) = \dot{\boldsymbol{r}}^N(t) \tag{2.9}$$

and

$$\boldsymbol{f}^N(t) \doteq -\nabla\mathcal{V}(\boldsymbol{r}^N(t)) = \dot{\boldsymbol{p}}^N(t) \ , \tag{2.10}$$

with the notation

$$\boldsymbol{v}^N \doteq \{\mathbf{v}_i \mid i = 1..\mathcal{N}_a\} \quad \text{and} \quad \boldsymbol{f}^N \doteq \{\mathbf{f}_i \mid i = 1..\mathcal{N}_a\} \ . \tag{2.11}$$

where $\mathbf{v}_i$ and $\mathbf{f}_i$ are the three-dimensional Cartesian velocity and force vectors of a particle i. In terms of these quantities, the first equation in Eq. 2.8 may equivalently be written

$$\underline{\mathbf{m}}\dot{\boldsymbol{v}}^N(t) = \boldsymbol{f}^N(t) \ . \tag{2.12}$$

Integrating Eq. 2.8 numerically forward in time on a computer to simulate classical-dynamical behaviour of a molecular system is called classical *molecular dynamics* (MD) simulation.

An alternative to *Newtonian equations of motion* are the *Langevin equations of motion*

$$\underline{\mathbf{m}}\dot{\boldsymbol{v}}^N(t) = \overline{\boldsymbol{f}}^N(t) + \boldsymbol{f}^{st,N}(t) - \underline{\mathbf{m}}\underline{\gamma}\boldsymbol{v}^N(t) \ , \tag{2.13}$$

where

$$\overline{\boldsymbol{f}}(t) \doteq -\nabla\overline{\mathcal{V}}(\boldsymbol{r}(t)) \ . \tag{2.14}$$

The mean force $\overline{\boldsymbol{f}}$ is the negative gradient of the potential of mean force $\overline{\mathcal{V}}$, *i.e.* it includes the forces between the explicit particles but also accounts in an effective fashion for the average forces exerted by the implicit particles.

The *stochastic force* is denoted by $\boldsymbol{f}^{st}(t)$ and the *frictional force* is proportional to the velocities $\mathbf{v}_i$ with the proportionality factor $\underline{\mathbf{m}}\underline{\gamma}$, in which $\underline{\gamma}$ is a diagonal matrix containing the atomic friction coefficients γ_i. A stochastic force introduces energy into the system, and a frictional force removes energy from it. The condition of zero energy loss on average will relate the two forces. If the stochastic force $\mathbf{f}_i^{st}$ obeys a Gaussian probability distribution with zero mean, if it is not correlated with prior velocities or forces, and if the friction coefficient γ_i is independent of time, this condition reads

$$< (\mathbf{f}_i^{st})^2 > \ = \ 6m_i\gamma_i \ k_B \ T_{ref} \tag{2.15}$$

where a time average is denoted by $<...>$, k_B is Boltzmann's constant, and T_{ref} is the reference temperature of the system. Numerical integration of the stochastic equations of motion is called *stochastic dynamics (*SD*) simulation*. Different approximations to this purpose are *Stochastic Dynamics* (SD), *high viscosity* SD *or Brownian Dynamics* (BD) or *Diffusive Particle Dynamics* (DPD).

Finally, when the degrees of freedom of the model are to be treated according to a continuous-material description, the equations of motion are typically formulated in terms of *transport equations* that relate the flow of specific extensive properties to the gradients of specific intensive properties.

A molecular system can be coupled to external quantities in different ways (see also Sec. 2.5):

1. Inclusion of an extra term $\mathcal{V}^{(res)}(\boldsymbol{r})$, a *penalty function*, in the interaction function of the system that restrains the motion of the particles such that the simulated value approaches the prescribed value of the given external quantity.
2. The prescribed value of the external quantity can be imposed as a *constraint* on to the system, such that it is satisfied by every configuration.
3. A first-order equation can be added to the particle equations of motion that drives the simulated value of the external quantity towards the prescribed value: the so-called *weak-coupling method*.

The latter two methods imply modification of the classical equations of motion of the molecular system.

We note that the total energy of the system will only be conserved in an MD simulation if the potential energy $\mathcal{V}$ is only a function of the current configuration. When applying weak coupling or a penalty function with time-averaged parameter values, this condition is not fulfilled.

The possibilities in the domain of application of the GROMOS package in terms of configuration generation are discussed in Sec. 3.4.

When performing *energy minimization*, the potential energy $\mathcal{V}$ molecular system is minimized using the negative gradient of this function.

2.5. Choice of the boundary conditions

The term *boundary condition* refers to global restrictions imposed on the system during the configuration generation. A restriction may be *hard*, a constraint affecting all individual configurations, or *soft*, a constraint on the average value of an observable over the generated configurations or a restraint biasing this average value towards a specified target value. One may distinguish the following types of boundary conditions:

A. *Spatial* boundary conditions
   These concern the nature of the confinement or periodicity applied to the molecular system. Typical examples are: *Vacuum Boundary Conditions* (VBC), a set of molecules in vacuum, *Fixed Boundary Conditions* (FBC), a system confined to a fixed region of space by means of a wall, and *Periodic Boundary Conditions* (PBC), a periodic system defined by the replication of a computational box of space-filling shape.

B. *Geometric* boundary conditions
   These concern the possible presence of constraints or restraints on specific internal coordinates within the system. Typical examples are the use of rigid bond lengths, the use of experiment-based restraints affecting specific internal coordinates or averages thereof, the use of artificial constraints or restraints on a specific internal coordinate, *e.g.* to avoid structural distortions during the equilibration phase of a simulation, or to bias the generation of configurations towards specific regions of the configurational space.

C. *Thermodynamic* boundary conditions
   These concern the thermodynamic state point associated with the generated configurational ensemble. Typical examples are a constant *number of particles vs* a constant *chemical potential*, a constant *volume vs* a constant *pressure*, a constant *energy* or enthalpy *vs* a constant *temperature*. In specific cases, other variables may be required for the definition of the thermodynamic state point.

The possibilities in the domain of application of the GROMOS package in terms of boundary conditions are discussed in Sec. 3.5.

CHAPTER 3

Scope of the GROMOS package

3.1. Introduction

The GROMOS package for molecular simulation currently consists of two subpackages that are tightly linked together but can largely be compiled and used as separate entities.

a. GROMOS++ is a library of supporting programs for pre- and post-MD tasks. It mainly consists of programs to facilitate the setup of simulations and of programs to analyze trajectories that are the results of such simulations.
b. MD++ is an energy minimizer and simulator, written in *C++* in an object oriented manner.

This chapter discusses the options offered by the two subpackages regarding the four basic aspects of molecular simulation.

3.2. Choice of the degrees of freedom

GROMOS can only consider degrees of freedom behaving according to the laws of classical mechanics. These are usually atoms or united atoms with two exceptions : (*i*) beads assumed to behave classically in a coarse-grained representation of molecular systems; (*ii*) beads connected by harmonic springs in a ring topology as a path-integral representation of a quantum-mechanical system. GROMOS is primarily intended for simulations of condensed-phase systems, solutions, pure liquids and crystals, with an explicit representation of the solvent molecules in the case of solutions. However, the simulation of systems in the gas phase and of solutions with an implicit-solvent representation are also possible. Since GROMOS was originally developed for atomic degrees of freedom, particles are generally named atoms, although they actually may be groups of atoms or path-integral beads.

3.3. Choice of the description of the interaction

In classical simulations, the Hamiltonian of a molecular system has the form

$$\mathcal{H}(\boldsymbol{p}^N, \boldsymbol{r}^N) \quad = \quad \mathcal{K}(\boldsymbol{p}^N) + \mathcal{V}(\boldsymbol{r}^N) \tag{3.1}$$

The first term is the *kinetic energy* term

$$\mathcal{K}(\boldsymbol{p}^N) = \sum_{i=1}^{N} \frac{\mathbf{p}_i{}^2}{2m_i} = \sum_{i=1}^{N} \tfrac{1}{2} m_i \mathbf{v}_i{}^2 \ , \tag{3.2}$$

which is independent of the particle coordinates $\boldsymbol{r}^N$ in the absence of geometric positional constraints. If constraints are imposed, the components of the momenta $\mathbf{p}_i$ or the velocities $\mathbf{v}_i$ along the constrained degrees of freedom must be zero. The second term is the *potential energy* term or interaction function, which describes the interaction energy in terms of particle coordinates $\mathbf{r}$

$$\mathcal{V}(\boldsymbol{r};\mathbf{s}) \quad \equiv \quad \mathcal{V}(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N; \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_M). \tag{3.3}$$

Here $\boldsymbol{r}^N$ denotes the $3\mathcal{N}_a$-dimensional Cartesian coordinate vector of the system (periodic copies of the atoms within the reference box if PBC is applied). Generally, such a potential energy function $\mathcal{V}(\boldsymbol{r}^N;\mathbf{s})$ depends on a number (M) of parameters, also called force-field parameters, here indicated by $\mathbf{s} \equiv (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_M)$. In practice, the interaction function $\mathcal{V}(\boldsymbol{r}^N)$ consists of a sum of terms which represent different types of interactions. We distinguish two types of interactions or forces:

1. The standard *physical* atomic *interaction* function $\mathcal{V}^{(phys)}(\boldsymbol{r}^N, \mathbf{s})$, e.g. the GROMOS force field.

2. The *non-physical* atomic *interaction* function terms $\mathcal{V}^{(spec)}(\boldsymbol{r}^N)$, which are included to serve a special purpose, e.g. atomic position restraining, distance restraining, dihedral-angle restraining, etc.

So, we have

$$\mathcal{V}(\boldsymbol{r};\mathbf{s}) \quad = \quad \mathcal{V}^{(phys)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(spec)}(\boldsymbol{r}^N) \tag{3.4}$$

The physical potential energy term $\mathcal{V}^{(phys)}$ is further divided into a term $\mathcal{V}^{(cov)}$ corresponding to *covalent interactions* and a term $\mathcal{V}^{(nbd)}$ corresponding to *non-bonded interactions*. This results in

$$\mathcal{V}^{(phys)}(\boldsymbol{r}^N;\mathbf{s}) \doteq \mathcal{V}^{(cov)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(nbd)}(\boldsymbol{r}^N;\mathbf{s}) \ . \tag{3.5}$$

The covalent term is further partitioned as a sum of contributions from *bond stretching*, *bond-angle bending*, *improper dihedral-angle bending* and *proper dihedral-angle torsion* interactions, namely

$$\mathcal{V}^{(cov)}(\boldsymbol{r}^N;\mathbf{s}) \doteq \mathcal{V}^{(b)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(\theta)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(\xi)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(\varphi)}(\boldsymbol{r}^N;\mathbf{s}) \ . \tag{3.6}$$

The non-bonded term is further partitioned as a sum of contributions from *van der Waals* and *electrostatic* interactions, namely

$$\mathcal{V}^{(nbd)}(\boldsymbol{r}^N;\mathbf{s}) \doteq \mathcal{V}^{(vdw)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(ele)}(\boldsymbol{r}^N;\mathbf{s}) \ . \tag{3.7}$$

The different terms involved in Eqs. 3.5-3.7 are described in Chaps. 5, 6 and 7. The force $\mathbf{f}_i$ on particle i due to a particular interaction term is given by the relation

$$\mathbf{f}_i \quad = \quad -\frac{\partial}{\partial \mathbf{r}_i}\, \mathcal{V}(\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_N), \tag{3.8}$$

which can also be used to obtain the interaction energy difference that corresponds to a given force $\mathbf{f}_i$. We note that a MD trajectory only depends on the forces on the atoms, not on the energies.

The derivatives of the potential energy terms with respect to atomic coordinates, box or coupling parameters are provided in Chap. 17, while the parameters involved in these terms are included as part of the GROMOS force-field description in Vol. 3.

3.3.1. Charge groups, searching neighbours. The bulk of the computer time required by a simulation time step is used for calculating the non-bonded interactions, that is, for finding the nearest neighbour atoms and subsequently evaluating the van der Waals and electrostatic interaction terms for the obtained atom pairs. Therefore, various schemes for performing this task as efficiently as possible are available.[14]

Since the non-bonded interaction between atoms decreases with the distance between them, only interactions between atoms closer to each other than a certain cut-off distance R_c are generally taken into account in simulations.

When the (partial) atomic charges of a group of atoms add up to exactly zero, the leading term of the electrostatic interaction between two such groups of atoms is of dipolar ($1/r^3$) character. The sum of the 1/r monopole contributions of the various atom pairs to the group-group interaction will be zero. Therefore, the range of the electrostatic interaction can be considerably reduced when atoms are assembled in so-called charge groups, which have a zero net charge, and for which the electrostatic interaction with other (groups of) atoms is either calculated for all atoms of the charge group or for none.

The GROMOS force fields make use of this *concept of charge groups*. The atoms that belong to a charge group are chosen such that their partial atomic charges add up to zero. For groups of atoms with a total charge of $+e$ or $-e$, like the sidechain atoms of Arg or Asp, the partial atomic charges of the charge group may add up to $+e$ or $-e$. When a cut-off radius is used, one can choose to base this cutoff on the atomic positions (AT) or to use a charge group based cutoff. The *position of a charge group* is defined differently for a charge group belonging to the "solute" part of the molecular topology and one in the "solvent" part of the molecular topology.

- The position of a "solute" charge group is taken to be its centre of geometry:

$$R_{cg} = \sum_{i=1}^{N_{cg}} \mathbf{r}_i / N_{cg} \tag{3.9}$$

  where the number of atoms belonging to the charge group is denoted by N_{cg}.
- The position of a "solvent" charge group is taken to be the position of the first atom of a solvent molecule. A "solvent" molecule may only contain one charge group.

Therefore, in the GROMOS non-bonded interaction subroutines the cut-off radius R_c , denoted by RCUTP, is used to select nearest-neighbour charge groups.

The simplest way to find the neighbouring charge groups of a charge group, that is, the charge groups that lie within R_c , is to scan all possible charge group pairs in the system. For a system consisting of N_{CG} charge groups, the number of pairs amounts to $1/2{N_{CG}}^2$, which makes the computer time required for finding the neighbours in this way proportional to ${N_{CG}}^2$. Faster neighbour-search algorithms do exist, see Sec. 3.3.3. Once the neighbours have been found, the time required for calculating the non-bonded interaction is proportional to N_{CG} . We note that non-bonded interactions within a charge group may need to be calculated, when the charge group contains many atoms.

3.3.2. Twin-range method for long-range interactions. In order to evaluate the non-bonded interaction (Eq. 3.7) with sufficient accuracy, a long cut-off radius R_{cl} has to be used; for molecular systems a value of at least 1.4 nm seems necessary. But such a range is very expensive if pair interactions are evaluated; the number of neighbour atoms within 1.4 nm will exceed 300. Therefore, in GROMOS the non-bonded interaction can be evaluated using a *twin-range method.*[15] Secondly, the electrostatic interactions beyond the long-range cutoff R_{cl} can be approximated by a Poisson-Boltzmann generalized *reaction field* term.

The non-bonded interactions in Eq. 3.7 are evaluated at every simulation step using the charge group pair list that is generated with a short range cut-off radius R_{cp} (=RCUTP). The longer range non-bonded interactions, that is, those between charge groups at a distance longer than R_{cp} and smaller than R_{cl} (=RCUTL), are evaluated less frequently, viz. only at every n-th (=NSNB) simulation step when also the pair list is updated. They are kept unchanged between these updates. In this way the long-range non-bonded forces can be approximately taken into account, without increasing the computing effort significantly, at the expense of neglecting the fluctuation of the forces beyond R_{cp} during n simulation steps.

The long-range interaction, which is calculated for charge group pairs at distances between R_{cp} and R_{cl}, is evaluated by using the electrostatic term in Eq. 3.7 and the normal van der Waals parameters in the Lennard-Jones term. It is assumed that no excluded neighbours, no third-neighbour or 1,4-interactions and no intra-charge-group interactions exist at these distances. So R_{cp} must not be chosen too small.

The evaluation of the nonbonded interactions in GROMOS relies on the application of the twin-range method.[1,16–18] The GROMOS implementation of this approach relies on the definition of: (1) a short-range pairlist and cutoff distance R_{cp}; (2) an intermediate-range pairlist and cutoff distance R_{cl}; (3) anupdate frequency N_s for the short-range pairlist and for the intermediate range interactions; Short-range interactions are computed every time step based on a short-range pairlist containing pairs in the distance range [0; R_{cp}].

The short-range pairlist is reevaluated every N_s time steps. It can be generated either on the basis of distances between charge groups (groups of covalently linked atoms defined in the system topology) or of distances between individual atoms. Intermediate range interactions are computed every N_s time steps based on all pairs in the distance range [R_{cp}; R_{cl}] at the time of the evaluation of these interactions. The energy, forces, and virial contributions associated with intermediate-range interactions are assumed constant between two updates (i.e., during N_s steps).

The evaluated interaction includes Lennard-Jones and electrostatic components. The latter component may include a reaction-field contribution or the real-space contribution to a lattice-sum method. Note that the real-space contribution to a lattice-sum method may only be computed within the short-range contribution to the interaction.

3.3.3. Pairlist construction. Pairlist construction may be performed in three different ways:

1. using the standard double-loop algorithm;
2. using a grid-based pairlist algorithm[19];
3. using a slight variation of the grid-based algorithm, which permits easier parallelization and avoids periodicity corrections during the interaction evaluation.

The standard double-loop algorithm is selected by setting the `algorithm` flag in the `PAIRLIST` block to 0. A grid-based pairlist algorithm is implemented to allow for a fast construction of cutoff-based nonbonded pairlists in molecular simulations under periodic boundary conditions based on an arbitrary box shape (rectangular, truncated-octahedral, or triclinic).[19] The key features of this algorithm are: (1) the use of a one-dimensional mask array (to determine which grid cells contain interacting atoms) that incorporates the effect of periodicity, and (2) the grouping of adjacent interacting cells of the mask array into stripes, which

permits the handling of empty cells with a very low computational overhead. Testing of the algorithm on water systems of different sizes (containing about 2000 to 11,000 molecules) has shown that the method: (1) is about an order of magnitude more efficient compared to a standard (double-loop) algorithm, (2) achieves quasi-linear scaling in the number of atoms, (3) is weakly sensitive in terms of efficiency to the chosen number of grid cells. This grid-based algorithm is set by setting the `algorithm` parameter in the `PAIRLIST` block to 2.

Furthermore, MD++ includes a slightly modified version of this grid-based pairlist algorithm following ideas similar to those of a published pairlist algorithm.[20] In an effort to reduce the number of nearest image determinations during the pairlist generation and the nonbonded force calculation, the system gets extended on all sides before the pairlist construction. The additional atom or charge-group positions are obtained by simple shifts of the original positions by the box vectors. To allow for more efficient (distributed- memory) parallelization and to save time, the central computational box is divided into N layers. Each of the P parallel processes only has to extend over N/P layers. After every extension, the atom pairs consisting of one atom within the layer and a second atom from one of the above (not extended) layers are added to the respective pairlist (using a one-dimensional mask array). Filtering or energy and force evaluation can then be carried out right away (without nearest image determinations owing to the preshifted atomic positions), or at a later stage with the information on the shift vectors encoded into the pairlist, thus enabling a fast reconstruction of the shifted positions. This version of the grid-based algorithm is selected by setting the `algorithm` flag of the `PAIRLIST` block to 1.

3.4. Choice of the method for the configuration generation

The GROMOS program MD++ may be used to perform energy minimizations, molecular dynamics simulations or stochastic dynamics simulations. Details of the algorithms that are implemented and considerations to be kept in mind when setting up the calculations are given in Chaps. 11, 12 and 13 respectively.

3.5. Choice of the boundary conditions

GROMOS offers a wide range of algorithms to apply boundary conditions to an energy minimization, molecular or stochastic dynamics simulation. Spatial boundary conditions are limited to vacuum boundary conditions, fixed boundary conditions or periodic boundary conditions in various shapes. Details for spatial boundary conditions are described in Chap. 4. Geometric boundary conditions may be applied in the form of restraints through the use of special force-field terms (Chap. 9) or as constraints through a direct adaptation of the equations of motion (Chap. 10). GROMOS currently cannot perform simulations in a grand-canonical ensemble, *i.e.* with constant chemical potential rather than constant number of particles.

CHAPTER 4

Spatial boundary conditions

4.1. Introduction

When simulating a system of finite size, some thoughts must be given to the way how the boundary of the system will be treated. Spatial boundary conditions define the shape, size and confinement of the simulated system. GROMOS can handle the following types of spatial boundary conditions:

A. *Vacuum boundary conditions* (VBC) describe a system of particles surrounded by vacuum.
B. *Fixed boundary conditions* (FBC) describe a system of particles confined within a finite volume.
C. *Periodic boundary conditions* (PBC) describe a system of particles within a reference computational box of space-filling shape surrounded by an infinite lattice of periodic replicas of itself.

The simplest choice is the *vacuum* boundary condition, which is discussed in Sec. 4.2. The *fixed* boundary conditions (Sec. 4.3) are not implemented separately in GROMOS and are treated like a special case of a vacuum simulation. Periodic boundary conditions are useful in simulations of solutions, to remove surface effects when dealing with microscopic samples, and in simulations of crystals based on the periodicity determined by the crystallographic unit cell. When periodic boundary conditions are applied, the shape, size and orientation of the computational box must be defined. Sec. 4.4 discusses the use of periodic boundary conditions in GROMOS.

4.2. Vacuum boundary conditions (VBC)

Simulating a molecular system in vacuo, that is, without any wall or boundary, corresponds to the gas phase at pressure zero. When the vacuum boundary is used for a molecule in solution, properties of atoms near or at the surface of the system will be distorted[21, 22]. The vacuum boundary condition may also distort the shape of a non-spherical molecule, since it generally tends to minimize the surface area. Moreover, the shielding effect of a solvent with high dielectric permittivity, such as water, on the electric interaction between charges or dipoles in a molecule is missed in vacuo. Therefore, simulation of a charged extended molecule *e.g.* DNA in vacuo is a precareous undertaking. The best results in vacuo are obtained for relatively large globular macromolecules.

The vacuum boundary condition is selected using the switch NTB = 0 in the input block BOUNDCOND. When the molecular system contains groups of atoms with a total net charge not equal to zero, it is advised to use the *GROMOS 45B4 or 54B7 force field* in which charged charge groups are neutralized and some van der Waals parameters are modified in order to maintain the hydrogen binding capacity of the charge group that is neutralized.

When simulating a system in vacuo, the translational momentum and the angular momentum are conserved quantities. Therefore, it is common practice to *stop* the *translational* motion of the centre of mass and the *rotational motion* around the centre of mass at the start of such a simulation. This is done by using NTICOM=2 in the input block INITIALISE at the start of a simulation. In vacuo these motions will remain absent (zero), except in the case of long simulations, when the numerical noise may build up a sizeable centre of mass translation or rotational motion, especially when the system is coupled to a temperature bath. Therefore, it is advisable to stop the centre of mass motion regularly, e.g. by setting the number of steps after which centre of mass motion is stopped, NSCM = 10000 in the input block COMTRANSROT in MD++. By specifying a negative number for NSCM, the centre of mass rotation is stopped additionally.

The temperature T of a system is calculated using the relation

$$\sum_{i=1}^{\mathcal{N}_a} 1/2 m_i \mathbf{v}_i^2 \quad = \quad \tfrac{1}{2}\mathcal{N}_d k_B T \tag{4.1}$$

where k_B is Boltzmann's constant and the number of degrees of freedom in the system is denoted by

$$\mathcal{N}_d \quad = \quad 3\mathcal{N}_a - N_c - NDFMIN \tag{4.2}$$

Here the number of atoms is $\mathcal{N}_a$ and the number of geometric constraints is N_c. The choice of the value for NDFMIN in input block BOUNDCOND should depend on the boundary condition chosen and on whether the centre of mass motion is regularly stopped.

When the overall translational and rotational motion has been stopped and the system is in vacuo, six degrees of freedom have to be subtracted from the total in Eq. 4.2, so NDFMIN = 6, in order to obtain the correct kinetic energy per degree of freedom.

4.3. Fixed boundary conditions (FBC)

When simulating crystals or solutions of large molecules, the application of periodic boundary conditions may require many atoms to be included in the computational box and so may become very expensive. In that case the number of atoms in the simulation can be limited by simulating only part of the molecular system. For example, only the atoms within a spherical region around a specific atom or point in the system are retained, while all atoms lying beyond a radius R_2 are removed from the system. Edge effects, due to the presence of vacuum beyond R_2, can be minimized by restraining the motion of the atoms in the outer shell, viz. between radii R_1 and R_2. This shell is called the extended wall region (see Fig. 4.1). The atoms in this *wall region* can be kept fixed or near given stationary reference positions by the technique of *position restraining*, which is discussed in Sec. 9.2. Atoms in the inner region, within R_1, are simulated without any restraints.

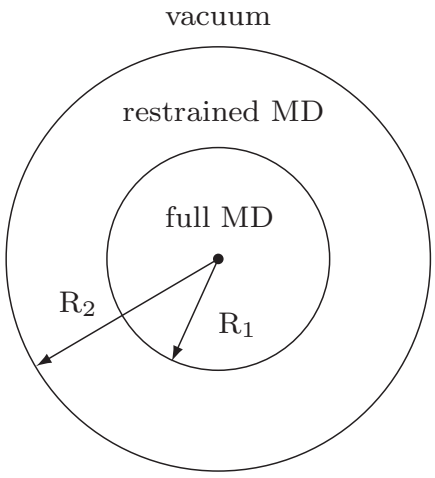


FIGURE 4.1. Spherical wall region with restrained atomic motion

When applying the extended wall region boundary condition, the *molecular topology* (Vol. 4) can be *reduced* to one containing only atoms, *i.e.* charge groups, that lie within a distance R_2 from a given point. The subset of atoms within the simulation sphere and in the extended wall region may be obtained from a complete system using the GROMOS++ program *pairlist* (Sec. 5-5.5), the molecular topology may subsequently be reduced using the program *red_top* (Sec. 5-2.27). The coordinate file can be modified using the GROMOS++ program filter (Sec. 5-4.29). The reduced system is not restricted to be of spherical shape. During a simulation the vacuum boundary condition should be selected (NTB = 0) and position

restraining applied to the atoms forming the wall region. Again, the restrained region is not restricted to being a spherical shell.

In order to avoid distorting effects of the vacuum beyond R_2 on the atomic motion within R_1 , one should choose these radii such that

$$R_c \quad < \quad R_2 - R_1 \tag{4.3}$$

that is, the thickness of the shell of restrained atoms should be larger than the nonbonded *cut-off radius* R_c .

4.4. Periodic boundary conditions (PBC)

The classical way to minimize edge effects in a finite system is to use periodic boundary conditions. The atoms of the system that is to be simulated are put into a periodic, space filling box, cubic, rectangular, triclinic, truncated octahedral, which is surrounded by identical translated images of itself (Fig. 4.2).

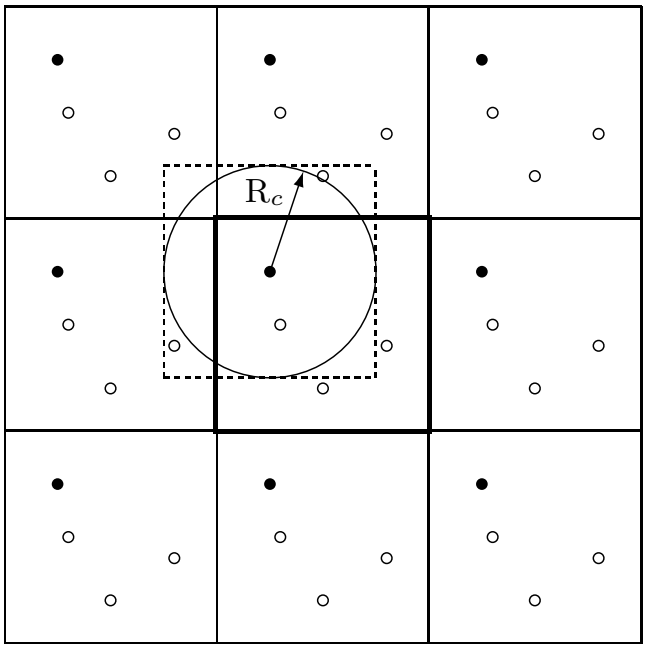


FIGURE 4.2. Periodic boundary conditions.

When an atom leaves the central box at one side, it enters it with identical velocity at the opposite side at the translated image position. Application of periodic boundary conditions means that in fact a crystal is simulated. For a molecule in solution the periodicity is an artifact of the computation, so the effects of periodicity on the forces on the atoms should not be significant. Possible distorting effects of the periodic boundary condition may be traced by simulating systems of different sizes.

When simulating a system using periodic boundary conditions, the translational momentum is a conserved quantity, but the *angular momentum* of the system is *not conserved*. As in vacuo, it is common practice to stop the translational motion of the centre of mass of the system and also the rotational motion around the centre of mass at the start of a simulation. However, the rotational degrees of freedom will pick up energy when the system evolves, and the translational energy should remain zero. Therefore, when the temperature of the system is calculated using Eqs. 4.1-4.2 three degrees of freedom have to be subtracted from the total in Eq. 4.2, so NDFMIN = 3, in order to obtain the correct kinetic energy per degree of freedom. As in the case of vacuum boundary condition, the centre of mass motion should be stopped now and then to avoid build-up of translational motion due to numerical noise in long simulations.

As all space filling boxes, *i.e.* cube, rectangular box, triclinic box, truncated octahedron, used in GROMOS can be expressed as triclinic boxes, this case is discussed in more detail in the following section. Periodic boundary conditions are switched on when NTB $\neq 0$. If NTB=1 rectangular periodic boundary conditions will be applyied, triclinic periodic boundary conditions for NTB=2, and truncated-octahedral periodic boundary condition for NTB=-1.

4.4.1. Triclinic computational box under PBC**.** The spatial properties, including position, shape, size and orientation, of an arbitrary triclinic box relative to a *reference coordinate system* $\{O, \hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z\}$, assumed Cartesian and right-handed, can be specified by 12 real numbers.

In the *matrix* representation, the spatial properties are specified by the position vector $\mathbf{T}$ of one reference corner of the box (relative to the origin O), along with the three edge vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ of the box, chosen so as to define a right-handed set, *i.e.* $\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) > 0$. The elements of these vectors will be assumed to represent the corresponding components in the reference coordinate system $\{\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z\}$.

In the *angular* representation, the spatial properties are specified by the position vector $\mathbf{T}$, along with the box edge lengths a, b and c, the edge angles α between $\mathbf{b}$ and $\mathbf{c}$, β between $\mathbf{a}$ and $\mathbf{c}$ and γ between $\mathbf{a}$ and $\mathbf{b}$, and the Euler angles ϕ, θ and ψ defining the orientation of the box relative to the reference coordinate system. In the definition of the Euler angles, the three edge vectors are used to define a box-linked right-handed Cartesian coordinate system $(\hat{\mathbf{e}}_{x'}, \hat{\mathbf{e}}_{y'}, \hat{\mathbf{e}}_{z'})$ in the following way : (i) the x'-axis is chosen along and in the direction of $\mathbf{a}$; (ii) the y'-axis is chosen orthogonal to $\mathbf{a}$ in the plane defined by $\mathbf{a}$ and $\mathbf{b}$, and oriented in the direction of $\mathbf{b}$; (iii) the z'-axis is chosen orthogonal to both $\mathbf{a}$ and $\mathbf{b}$, and oriented in the direction of $\mathbf{c}$. The reference coordinate system can be rotated onto the box-linked coordinate system by the following series of rotations : (i) a rotation by an angle ϕ around the z-axis ; (ii) a rotation by an angle θ around the new y-axis ; (iii) a rotation by an angle ψ around the new x-axis. The angles ϕ, θ and ψ thus represent the three Euler rotation angles in the zyx or yaw-pitch-roll convention. At any time during a simulation, the box angles α, β and γ are restricted to the range $]0; \pi[$, the Euler angles ϕ and ψ to the range $]-\pi; \pi]$, and the Euler angle θ to the range $[-\pi/2; \pi/2]$. The condition on θ follows from the observation that the changes $\theta \rightarrow \pi - \theta$, $\phi \rightarrow \pi + \phi$ and $\psi \rightarrow \pi + \psi$ always lead to an equivalent description of the box orientation. When $\theta = \pm\pi/2$, the angles ϕ and ψ are individually undefined only $\phi - \psi$ or $\phi + \psi$ are defined when $\theta = \pi/2$ and $-\pi/2$, respectively. In this situation, ϕ is arbitrarily set to zero. For both rectangular and truncated-octahedral boxes, the conditions $\alpha = \beta = \gamma = \pi/2$ and $\phi = \theta = \psi = 0$ are also enforced at any time during the simulation. For a truncated-octahedral box, the condition $a = b = c$ is enforced in addition.

4.4.1.1. *Coordinate transformations: triclinic box.* Based on a general triclinic box in an arbitrary orientation, as introduced in Sec. 4.4.1 the coordinates of an atom can be specified in four distinct ways : (i) through oblique fractional coordinates $\check{\mathbf{r}} = (q, s, t)$ with reference to the box-edge vectors ; (ii) through oblique coordinates $\check{\mathbf{r}} = (u, v, w)$ with reference to the box-edge vectors ; (iii) through coordinates $\mathbf{r}' = (x', y', z')$ within the box-linked Cartesian coordinate system ; (iv) through coordinates $\mathbf{r} = (x, y, z)$ within the reference Cartesian coordinate system. The different sets of coordinates are related through

$$u = qa \ , v = sb \ , w = tc \ , \tag{4.4}$$

$$\mathbf{r}' = ua^{-1}\mathbf{a}' + vb^{-1}\mathbf{b}' + wc^{-1}\mathbf{c}' \ , \tag{4.5}$$

$$\mathbf{r} = x'\hat{\mathbf{e}}_{x'} + y'\hat{\mathbf{e}}_{y'} + z'\hat{\mathbf{e}}_{z'} \tag{4.6}$$

and

$$\mathbf{r} = ua^{-1}\mathbf{a} + vb^{-1}\mathbf{b} + wc^{-1}\mathbf{c} \ , \tag{4.7}$$

where $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ are the components of the edge vectors in the reference Cartesian coordinate system, and $\mathbf{a}'$, $\mathbf{b}'$ and $\mathbf{c}'$ the corresponding components in the box-linked Cartesian coordinate system.

Defining the matrix $\underline{\mathbf{B}}$ as the diagonal matrix containing the box-edge lengths a, b and c as its elements, *i.e.*

$$\underline{\mathbf{B}} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \ , \tag{4.8}$$

Eq. 4.4 may be rewritten

$$\check{\mathbf{r}} = \underline{\mathbf{B}}\check{\mathbf{r}} \ . \tag{4.9}$$

Defining the transformation matrix $\underline{\mathbf{S}}$ as the matrix containing $a^{-1}\mathbf{a}'$, $b^{-1}\mathbf{b}'$ and $c^{-1}\mathbf{c}'$ in its columns, *i.e.*

$$\underline{\mathbf{S}} = \begin{pmatrix} a^{-1}a_x{}' & b^{-1}b_x{}' & c^{-1}c_x{}' \\ a^{-1}a_y{}' & b^{-1}b_y{}' & c^{-1}c_y{}' \\ a^{-1}a_z{}' & b^{-1}b_z{}' & c^{-1}c_z{}' \end{pmatrix} = \begin{pmatrix} 1 & \cos\gamma & \cos\beta \\ 0 & \sin\gamma & \sin\beta\cos\delta \\ 0 & 0 & \sin\beta\sin\delta \end{pmatrix} , \tag{4.10}$$

where

$$\cos\delta = \frac{\cos\alpha - \cos\beta\cos\gamma}{\sin\beta\sin\gamma} \quad \text{with} \quad \delta \in]0;\pi[, \tag{4.11}$$

Eq. 4.5 may be rewritten as

$$\mathbf{r}' = \underline{\mathbf{S}}\check{\mathbf{r}} . \tag{4.12}$$

The inverse transformation is performed through the matrix

$$\underline{\mathbf{S}}^{-1} = \begin{pmatrix} 1 & -\cot\gamma & \cot\gamma\cot\delta - \cot\beta\sin^{-1}\delta \\ 0 & \sin^{-1}\gamma & -\cot\delta\sin^{-1}\gamma \\ 0 & 0 & \sin^{-1}\beta\sin^{-1}\delta \end{pmatrix} . \tag{4.13}$$

In MD++ the matrix $\underline{\mathbf{S}}$ is defined as `math::Matrixl math::smat`.

Defining the transformation matrix $\underline{\mathbf{R}}$ as the matrix containing $\hat{\mathbf{e}}_{x'}$, $\hat{\mathbf{e}}_{y'}$ and $\hat{\mathbf{e}}_{z'}$ in its columns, *i.e.*

$$\underline{\mathbf{R}} = \begin{pmatrix} \cos\theta\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \cos\theta\sin\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi \\ -\sin\theta & \sin\psi\cos\theta & \cos\psi\cos\theta \end{pmatrix} , \tag{4.14}$$

Eq. 4.6 may be rewritten as

$$\mathbf{r} = \underline{\mathbf{R}}\mathbf{r}' . \tag{4.15}$$

The inverse transformation is performed through the matrix $\underline{\mathbf{R}}^{-1} = {}^t\underline{\mathbf{R}}$. In MD++ the matrix $\underline{\mathbf{R}}$ is defined by `math::Matrixl math::rmat` which can be inverted by transposition.

Finally, Eq. 4.7 corresponds to the combined transformation

$$\mathbf{r} = \underline{\mathbf{T}}\check{\mathbf{r}} \tag{4.16}$$

with

$$\underline{\mathbf{T}} = \underline{\mathbf{R}}\,\underline{\mathbf{S}} . \tag{4.17}$$

The inverse transformation is obtained through $\underline{\mathbf{T}}^{-1} = \underline{\mathbf{S}}^{-1}\underline{\mathbf{R}}^{-1} = \underline{\mathbf{S}}^{-1}\,{}^t\underline{\mathbf{R}}$. In MD++ the matrices `math::Matrixl math::mmat` and `math::Matrixl math::minvmat` can be used.

It is also convenient to define the matrix $\underline{\mathbf{L}}'$ as the matrix containing $\mathbf{a}'$, $\mathbf{b}'$ and $\mathbf{c}'$ in its columns, *i.e.*

$$\underline{\mathbf{L}}' = \begin{pmatrix} a_x{}' & b_x{}' & c_x{}' \\ a_y{}' & b_y{}' & c_y{}' \\ a_z{}' & b_z{}' & c_z{}' \end{pmatrix} = \underline{\mathbf{S}}\,\underline{\mathbf{B}} , \tag{4.18}$$

and the matrix $\underline{\mathbf{L}}$ as the matrix containing containing $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ in its columns, *i.e.*

$$\underline{\mathbf{L}} = \begin{pmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{pmatrix} = \underline{\mathbf{R}}\,\underline{\mathbf{L}}' = \underline{\mathbf{R}}\,\underline{\mathbf{S}}\,\underline{\mathbf{B}} = \underline{\mathbf{T}}\,\underline{\mathbf{B}} . \tag{4.19}$$

Obviously,

$$\mathbf{r}' = \underline{\mathbf{L}}'\check{\mathbf{r}} \quad \text{and} \quad \mathbf{r} = \underline{\mathbf{L}}\check{\mathbf{r}} . \tag{4.20}$$

In some cases, it may be necessary to calculate the Euler angles ϕ, θ and ψ from the components of the $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ vectors. This may be done by constructing the unit vectors of the box-linked Cartesian coordinate system as

$$\begin{aligned} \hat{\mathbf{e}}_{x'} &= a^{-1}\,\mathbf{a} , \\ \hat{\mathbf{e}}_{y'} &= \parallel \mathbf{b} - (a\,b)^{-1}(\mathbf{a}\cdot\mathbf{b})\mathbf{a} \parallel^{-1} \; [\mathbf{b} - (a\,b)^{-1}(\mathbf{a}\cdot\mathbf{b})\mathbf{a}] , \\ \hat{\mathbf{e}}_{z'} &= \hat{\mathbf{e}}_{x'} \times \hat{\mathbf{e}}_{y'} . \end{aligned} \tag{4.21}$$

Identifying the matrix containing $\hat{\mathbf{e}}_{x'}$, $\hat{\mathbf{e}}_{y'}$ and $\hat{\mathbf{e}}_{z'}$ in its columns with the matrix $\underline{\mathbf{R}}$ of Eq. 4.14, one has

$$\begin{aligned} \theta &= -\text{sign}(\underline{\mathbf{R}}_{31})\text{acos}[(\underline{\mathbf{R}}_{11}^2 + \underline{\mathbf{R}}_{21}^2)^{1/2}] && (4.22) \\ \psi &= \text{sign}(\underline{\mathbf{R}}_{32}\cos^{-1}\theta)\text{acos}(\underline{\mathbf{R}}_{33}\cos^{-1}\theta) \\ \phi &= \text{sign}(\underline{\mathbf{R}}_{21}\cos^{-1}\theta)\text{acos}(\underline{\mathbf{R}}_{11}\cos^{-1}\theta) && (4.23) \end{aligned}$$

if $R_{11}^2 + R_{21}^2 \neq 0$, or

$$\begin{aligned} \theta &= -\text{sign}(\underline{\mathbf{R}}_{31})\pi/2 && (4.24) \\ \psi &= 0 \\ \phi &= -\text{sign}(\underline{\mathbf{R}}_{12})\text{acos}(\underline{\mathbf{R}}_{22}) \ , && (4.25) \end{aligned}$$

otherwise. The function $sign(x)$ delivers the sign of x.

4.4.1.2. *Nearest image and gathering.* When calculating the forces on the black atom in the central box in Fig. 4.2, all interactions with atoms in the central box or images in the surrounding boxes that lie within the spherical cut-off radius R_c are taken into account. To avoid artificial periodicity effects, an atom should not simultaneously interact with another atom and its image. Therefore, in GROMOS only the *interaction between nearest images* is evaluated. Consequently, the smallest wall-to-wall distance R_{box} *must exceed twice the cut-off radius* R_c:

$$R_{box} > 2R_c \qquad (4.26)$$

that is, the *cut-off radius* (RCUTP or RCUTL) must be smaller than half the smallest edge of the box. Since the cut-off radii RCUTP and RCUTL are applied to charge groups one should choose

$$RCUTP \leqslant RCUTL \leqslant R_c - R^{cg} \qquad (4.27)$$

with R_c satisfying Eq. 4.26 and where R^{cg} is the charge group radius.

This condition can be met by choosing the system large enough, e.g. in a crystal the computational box may contain more than one unit cell. The application of non-bonded neighbour search techniques in periodic systems is discussed in[14]. Possible distorting effects of the periodic boundary condition may be traced by simulating systems of different size.

Applying periodic boundary conditions implies that when an atom leaves the central box through one of its walls, it enters at the opposite image position with the same velocity. However, in GROMOS this *periodic translation* is not performed for single atoms, but for all *atoms of a charge group*. Solute charge group atoms and solvent molecules are translated, applying periodic boundary conditions such that the first atom of a solute charge group or of a solvent molecule lies within the central periodic box.

For an arbitrary molecular configuration the atoms of a charge group may lie far apart in the central box, close to opposite walls, while their nearest images are close to each other. In that case the atoms of a charge group must first be gathered by applying periodic boundary conditions, such that the atoms of a charge group lie within $R_{box}/2$ of each other. It is always assumed that the atoms of a solvent molecule lie within $R_{box}/2$ of each other, viz. that the *solvent atom coordinates are generated without mixing different periodic images in one solvent molecule*.

When a solute molecule consists of a chain of covalently bound atoms, this chain may be cut into different segments by the periodic boundaries. Following the chain one may leave the central box through one wall and enter it at the opposite wall at the image position. This means that when the various contributions to the potential energy in Eq. 3.5 are computed, nearest images of atoms involved in bonds, bond angles, dihedral angles, etc. have to be used.

In a triclinic box, the direction of the edges are denoted by **a**, **b** and **c** and lengths a, b, and c. The atom or charge group i with position $\mathbf{r}_i$ can be *kept in the computational box* that lies in the positive quadrant with respect to an origin at $\mathbf{r}_0$, by applying the translation

$$\mathbf{r}_i' = \mathbf{r}_i - NINT((z_i - z_0 - c/2)/c)\mathbf{c}$$

$$\begin{aligned}
\mathbf{r}_i'' &= \mathbf{r}_i' - NINT((y_i' - y_0 - b/2)/b)\mathbf{b} \\
\mathbf{r}_i''' &= \mathbf{r}_i'' - NINT((x_i'' - x_0 - a/2)/a)\mathbf{a} \\
\mathbf{r}_i &= \mathbf{r}_i'''. \qquad (4.28)
\end{aligned}$$

where the function NINT(x) delivers the integer number that is nearest to x. When calculating in oblique coordinates, Eq. 4.53 can be used. For two atoms or charge groups i and j the vector

$$\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j \qquad (4.29)$$

can be transformed to the vector $\mathbf{r}_{ij}^{NI}$ *connecting nearest images* by the transformation

$$\begin{aligned}
\mathbf{r}_{ij}' &= \mathbf{r}_i - NINT((z_{ij}/c)\mathbf{c} \\
\mathbf{r}_{ij}'' &= \mathbf{r}_i' - NINT((y_{ij}'/b)\mathbf{b} \\
\mathbf{r}_{ij}''' &= \mathbf{r}_i'' - NINT((x_{ij}''a)\mathbf{a} \\
\mathbf{r}_{ij}^{NI} &= \mathbf{r}_i'''. \qquad (4.30)
\end{aligned}$$

4.4.1.3. *Geometric properties.* In a general triclinic box, the square length of a vector is given in terms of the corresponding oblique coordinates by

$$\begin{aligned}
r^2 &= x^2 + y^2 + z^2 = (x')^2 + (y')^2 + (z')^2 \qquad (4.31) \\
&= u^2 + v^2 + w^2 + 2uv\cos\gamma + 2uw\cos\beta + 2vw\cos\alpha \ .
\end{aligned}$$

The volume of a triclinic box is given by

$$\mathcal{V} = abc[1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\cos\beta\cos\gamma]^{1/2} \ . \qquad (4.32)$$

The acceptable cutoff value within a triclinic box is restricted to half the minimum distance between any two opposite walls of the cell, *i.e.* the restriction Eq. 4.26 becomes

$$R_c \leq \frac{1}{2}\,\mathrm{MIN}((ab\sin\gamma)^{-1}\mathcal{V}; (ac\sin\beta)^{-1}\mathcal{V}; (bc\sin\alpha)^{-1}\mathcal{V}), \qquad (4.33)$$

The function MIN(x,y,z) delivers the smallest of its arguments x, y, z. In principle, the condition

$$R_c \leq \frac{1}{2}\mathrm{MIN}\{\parallel n_1\mathbf{a} + n_2\mathbf{b} + n_3\mathbf{c} \parallel \mid \ n_1, n_2, n_3 = 0 \text{ or } 1 \ , \ \mathbf{n} \neq \mathbf{0}\} \qquad (4.34)$$

would be sufficient to ensure that at most one periodic copy of each particle is within the cutoff distance of any other. However, this closest periodic copy is not necessarily the minimum image as determined by Eq. 4.52. To avoid this complication, the more restrictive condition of Eq. 4.33 is required.

The checking whether the box parameters are in allowed ranges (Sec. 4.4.1) and that the cutoff values are compatible with these is performed by `math::boundary_check_cutoff` in MD++.

4.4.1.4. *Reciprocal-lattice.* In the triclinic case, the reciprocal-lattice vectors $\tilde{\mathbf{a}}$, $\tilde{\mathbf{b}}$ and $\tilde{\mathbf{c}}$ associated with the edge vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ are defined by

$$\tilde{\mathbf{a}} = \mathcal{V}^{-1}\mathbf{b}\times\mathbf{c} \quad , \quad \tilde{\mathbf{b}} = \mathcal{V}^{-1}\mathbf{c}\times\mathbf{a} \quad \text{and} \quad \tilde{\mathbf{c}} = \mathcal{V}^{-1}\mathbf{a}\times\mathbf{b} \ . \qquad (4.35)$$

The matrix containing in its columns the components of $\tilde{\mathbf{a}}'$, $\tilde{\mathbf{b}}'$ and $\tilde{\mathbf{c}}'$ in the box-linked Cartesian coordinate system is easily shown to be

$$\begin{pmatrix} \tilde{a}_{x'} & \tilde{b}_{x'} & \tilde{c}_{x'} \\ \tilde{a}_{y'} & \tilde{b}_{y'} & \tilde{c}_{y'} \\ \tilde{a}_{z'} & \tilde{b}_{z'} & \tilde{c}_{z'} \end{pmatrix} = ({}^t\underline{\mathbf{L}}')^{-1} = {}^t\underline{\mathbf{S}}^{-1}\,\underline{\mathbf{B}}^{-1} \ . \qquad (4.36)$$

Similarly, the matrix containing in its columns the components of $\tilde{\mathbf{a}}$, $\tilde{\mathbf{b}}$ and $\tilde{\mathbf{c}}$ in the reference Cartesian coordinate system is given by

$$\begin{pmatrix} \tilde{a}_x & \tilde{b}_x & \tilde{c}_x \\ \tilde{a}_y & \tilde{b}_y & \tilde{c}_y \\ \tilde{a}_z & \tilde{b}_z & \tilde{c}_z \end{pmatrix} = {}^t\underline{\mathbf{L}}^{-1} = \underline{\mathbf{R}}\,({}^t\underline{\mathbf{L}}')^{-1} = \underline{\mathbf{R}}\,{}^t\underline{\mathbf{S}}^{-1}\,\underline{\mathbf{B}}^{-1} = {}^t\underline{\mathbf{T}}^{-1}\,\underline{\mathbf{B}}^{-1} \ . \qquad (4.37)$$

A reciprocal-space vector $\mathbf{k}$ is defined by

$$\mathbf{k} = 2\pi(l_a\tilde{\mathbf{a}} + l_b\tilde{\mathbf{b}} + l_c\tilde{\mathbf{c}}) \ , \qquad (4.38)$$

where $\mathbf{l} = (l_a, l_b, l_c)$ is a vector with integer, positive or negative, components. A reciprocal-space vector can be specified in five distinct ways : (*i*) through the integer vector $\mathbf{l}$; (*ii*) through oblique fractional

reciprocal space vectors $\breve{\mathbf{k}} = (\chi_q, \chi_s, \chi_t)$ with reference to the reciprocal-lattice vectors ; (iii) through oblique reciprocal space vectors $\breve{\mathbf{k}} = (\kappa_u, \kappa_v, \kappa_w)$ with reference to the reciprocal-lattice vectors ; (iv) through reciprocal space vectors $\mathbf{k}' = (rlk_{x'}', rlk_{y'}', rlk_{z'}')$ within the box-linked Cartesian coordinate system ; (v) through reciprocal space vectors $\mathbf{k} = (rlk_x, rlk_y, rlk_z)$ within the reference Cartesian coordinate system. The different coordinates are related through

$$\breve{\mathbf{k}} = 2\pi \mathbf{l} \; , \tag{4.39}$$

$$\breve{\mathbf{k}} = \underline{\mathbf{B}}^{-1} \breve{\mathbf{k}} = 2\pi \underline{\mathbf{B}}^{-1} \mathbf{l} \; , \tag{4.40}$$

$$\mathbf{k}' = {}^t\underline{\mathbf{S}}^{-1} \breve{\mathbf{k}} = ({}^t\underline{\mathbf{L}}')^{-1} \breve{\mathbf{k}} = 2\pi ({}^t\underline{\mathbf{L}}')^{-1} \mathbf{l} \; , \tag{4.41}$$

and

$$\mathbf{k} = \underline{\mathbf{R}} \mathbf{k}' = {}^t\underline{\mathbf{T}}^{-1} \breve{\mathbf{k}} = {}^t\underline{\mathbf{L}}^{-1} \breve{\mathbf{k}} = 2\pi {}^t\underline{\mathbf{L}}^{-1} \mathbf{l} \; . \tag{4.42}$$

At this point, it is also useful to state a number of important relationships. First, scalar products between real- and reciprocal-space vectors can be formulated similarly in the different coordinate representations, *i.e.*

$$\mathbf{k} \cdot \mathbf{r} = \mathbf{k}' \cdot \mathbf{r}' = \breve{\mathbf{k}} \cdot \breve{\mathbf{r}} = \breve{\mathbf{k}} \cdot \breve{\mathbf{r}} \; , \tag{4.43}$$

which follow immediately from the coordinate transformations given above. Second, a few useful differential relationships can be stated. For differentiating a reciprocal-space vector with respect to the box parameters, given in the form of the matrix $\underline{\mathbf{L}}$, one has

$$\frac{\partial \mathbf{k}}{\partial \underline{\mathbf{L}}_{\mu\nu}} = -\mathbf{k}_\mu \, {}^t\underline{\mathbf{L}}^{-1} \hat{\mathbf{e}}_\nu \; . \tag{4.44}$$

Following from this result, the differentiation of a scalar product of two reciprocal-space vectors with respect to the box parameters leads to

$$\frac{\partial (\mathbf{k}_1 \cdot \mathbf{k}_2)}{\partial \underline{\mathbf{L}}_{\mu\nu}} = -\mathbf{k}_{1,\mu} \, [\underline{\mathbf{L}}^{-1} \mathbf{k}_2]_\nu - \mathbf{k}_{2,\mu} \, [\underline{\mathbf{L}}^{-1} \mathbf{k}_1]_\nu \; . \tag{4.45}$$

Introducing the differentiation with respect to a matrix as a differentiation on a component-by-component basis, this may be rewritten as

$$\frac{\partial (\mathbf{k}_1 \cdot \mathbf{k}_2)}{\partial \underline{\mathbf{L}}} = -(\mathbf{k}_1 \otimes \mathbf{k}_2 + \mathbf{k}_2 \otimes \mathbf{k}_1) {}^t\underline{\mathbf{L}}^{-1} \; . \tag{4.46}$$

For differentiating the box volume $\mathcal{V} = | \, \underline{\mathbf{L}} \, |$ with respect to the box parameters, one has

$$\frac{\partial \mathcal{V}}{\partial \underline{\mathbf{L}}} = V \, {}^t\underline{\mathbf{L}}^{-1} \; . \tag{4.47}$$

4.4.1.5. *Tensor transformations.* It may be necessary to transform rank-two tensors, 3×3 matrices, among the various coordinate representations. If a tensor is written as $\underline{\mathbf{\Omega}}$ in terms of real-space oblique coordinates, $\underline{\mathbf{W}}'$ in the box-linked Cartesian coordinate system and $\underline{\mathbf{W}}$ in the reference Cartesian coordinate system, the conversion between the different representations is given by

$$\underline{\mathbf{W}}' = \underline{\mathbf{S}} \, \underline{\mathbf{\Omega}} \, {}^t\underline{\mathbf{S}} \tag{4.48}$$

and

$$\underline{\mathbf{W}} = \underline{\mathbf{R}} \, \underline{\mathbf{W}}' \, {}^t\underline{\mathbf{R}} = \underline{\mathbf{T}} \, \underline{\mathbf{\Omega}} \, {}^t\underline{\mathbf{T}} \; . \tag{4.49}$$

These three types of transformations can be easily checked by considering the simple tensors $\underline{\mathbf{W}} = \mathbf{r} \otimes \mathbf{r}$, $\underline{\mathbf{W}}' = \mathbf{r}' \otimes \mathbf{r}'$ and $\underline{\mathbf{\Omega}} = \breve{\mathbf{r}} \otimes \breve{\mathbf{r}}$ with $\mathbf{r}' = \underline{\mathbf{S}} \breve{\mathbf{r}}$ and $\mathbf{r} = \underline{\mathbf{R}} \mathbf{r}'$.

If the tensor is written $\underline{\tilde{\mathbf{\Omega}}}$ in terms of reciprocal-space oblique coordinates, the conversion between the different representations becomes

$$\underline{\mathbf{W}}' = {}^t\underline{\mathbf{S}}^{-1} \, \underline{\tilde{\mathbf{\Omega}}} \, \underline{\mathbf{S}}^{-1} \tag{4.50}$$

and

$$\underline{\mathbf{W}} = \underline{\mathbf{R}} \, \underline{\mathbf{W}}' \, {}^t\underline{\mathbf{R}} = {}^t\underline{\mathbf{T}}^{-1} \, \underline{\tilde{\mathbf{\Omega}}} \, \underline{\mathbf{T}}^{-1} \tag{4.51}$$

The transformation corresponding to the last equality can be checked by considering the simple tensors $\underline{\mathbf{W}} = \mathbf{k} \otimes \mathbf{k}$, $\underline{\mathbf{W}}' = \mathbf{k}' \otimes \mathbf{k}'$ and $\underline{\tilde{\mathbf{\Omega}}} = \breve{\mathbf{k}} \otimes \breve{\mathbf{k}}$ with $\mathbf{k}' = {}^t\underline{\mathbf{S}}^{-1} \breve{\mathbf{k}}$ and $\mathbf{k} = \underline{\mathbf{R}} \mathbf{k}'$.

In practice, these transformations are used to interconvert the various representations of the virial tensor.

4.4.1.6. *Application of periodicity.* In the triclinic case, the periodicity requirements apply to the oblique coordinates $\check{\mathbf{r}}$. The Cartesian components of the minimum-image vector $\overline{\mathbf{r}} = (\overline{x}, \overline{y}, \overline{z})$ associated with a vector $\mathbf{r} = (x, y, z)$ are obtained by

$$\begin{aligned} \overline{u} &= u - a\,\mathrm{NINT}\left(\frac{x-a}{a}\right) \\ \overline{v} &= v - b\,\mathrm{NINT}\left(\frac{y-b}{b}\right) \\ \overline{w} &= w - c\,\mathrm{NINT}\left(\frac{z-c}{c}\right) , \end{aligned} \tag{4.52}$$

where (u, v, w) are obtained from (x, y, z) through the inverse of Eq. 4.16 and $(\overline{x}, \overline{y}, \overline{z})$ from $(\overline{u}, \overline{v}, \overline{w})$ through Eq. 4.16. For the calculation of $\overline{r}^2$, the second operation may be replaced by Eq. 4.31.

4.4.2. Special periodic boundary conditions.

4.4.2.1. *Periodic rectangular box.* The simplest case of a periodic box is a rectangular box (NTB = 1). The lengths of the edges in the $x-$, $y-$ and $z-$directions are denoted by a, b and c and $\alpha = \beta = \gamma = \pi/2$. The atom or charge group i can be *kept in the computational box* that lies in the positive quadrant with respect to an origin at $\mathbf{r}_0$, by applying the translation

$$\begin{aligned} x_i &= x_i - NINT((x_i - x_0 - a/2)/a)a \\ y_i &= y_i - NINT((y_i - y_0 - b/2)/b)b \\ z_i &= z_i - NINT((z_i - z_0 - c/2)/c)c \end{aligned} \tag{4.53}$$

For two atoms or charge groups i and j the vector

$$\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j \tag{4.54}$$

can be transformed to the vector $\mathbf{r}_{ij}^{NI}$ *connecting nearest images* by the transformation

$$\begin{aligned} x_{ij}^{NI} &= x_{ij} - NINT(x_{ij}/a)a \\ y_{ij}^{NI} &= y_{ij} - NINT(y_{ij}/b)b \\ z_{ij}^{NI} &= z_{ij} - NINT(z_{ij}/c)c \end{aligned}$$

For a reactangular computational box the *requirement* Eq. 4.26 becomes

$$R_c < 1/2MIN(a, b, c) \tag{4.55}$$

4.4.2.2. *Truncated-octahedral computational box under* PBC. When simulating a spherical solute, use of a more spherically shaped computational box instead of a rectangular one may considerably reduce the number of solvent molecules that is needed to fill the remaining empty space in the box. A more spherically shaped space filling periodic box is a truncated octahedron, shown in Fig. 4.3. For a truncated-octahedral box, the vectors **a**, **b**, **c** correspond to the edges of the cube based on which the truncated octahedron is constructed with $a = b = c$ and $\alpha = \beta = \gamma = \pi/2$. The distance between the square planes is a, and between the six-sided planes it is $\frac{\sqrt{3}}{2}a$. The volume of the truncated octahedron is $\frac{1}{2}a^3$.

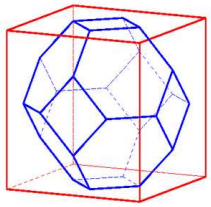


FIGURE 4.3. Truncated octahedron

The atom or charge group i can be *kept in the computational box* that lies in the positive quadrant with respect to an origin at $\mathbf{r}_0$, by applying the rectangular periodicity check Eq. 4.53 with $a = b = c$ followed by the translation

$$\text{if} \qquad |x_i - x_0 - a/2| + |y_i - y_0 - a/2| + |z_i - z_0 - a/2| > 3a/4$$

$$\text{then} \qquad \left\{ \begin{array}{rcl} x_i &=& x_i - \text{sign}(x_i - x_0 - a/2)a/2 \\ y_i &=& y_i - \text{sign}(y_i - y_0 - a/2)a/2 \\ z_i &=& z_i - \text{sign}(z_i - z_0 - a/2)a/2 \end{array} \right\} \tag{4.56}$$

The vector $\mathbf{r}_{ij}^{NIto}$ connecting nearest images of atoms or charge groups i and j is obtained by applying the rectangular periodicity check Eq. 4.55 with $a = b = c$ followed by the transformation

$$\text{if} \qquad |x_{ij}^{NI}| + |y_{ij}^{NI}| + |z_{ij}^{NI}| > 3a/4$$

$$\text{then} \quad \left\{ \begin{array}{rcl} x_{ij}^{NIto} &=& x_{ij}^{NI} - \text{sign}(x_{ij}^{NI})a/2 \\ y_{ij}^{NIto} &=& y_{ij}^{NI} - \text{sign}(y_{ij}^{NI})a/2 \\ z_{ij}^{NIto} &=& z_{ij}^{NI} - \text{sign}(z_{ij}^{NI})a/2 \end{array} \right\} \tag{4.57}$$

If *only* $(\mathbf{r}_{ij}^{NIto})^2$ *is required*, the last step, after applying Eq. 4.53 is replaced by calculating

$$(\mathbf{r}_{ij}^{NIto})^2 = (x_{ij}^{NI})^2 + (y_{ij}^{NI})^2 + (z_{ij}^{NI})^2 + a\ MIN(0,\ 3a/4 - |x_{ij}^{NI}| - |y_{ij}^{NI}| - |z_{ij}^{NI}|) \tag{4.58}$$

For a truncated octahedron the *requirement* Eq. 4.26 becomes

$$R_C \leq \frac{\sqrt{3}}{4}\, a \approx 0.433a\ . \tag{4.59}$$

that is, the *cut-off radius* must be smaller than half the distance between opposite planes that are defining the truncated octahedron. However, if the truncated-octahedron is mapped to an equivalent triclinic box (Sec. 4.4.2.3), Eq. 4.33 leads to the more restrictive condition

$$R_c \leq \frac{1}{2\sqrt{2}}\, a \approx 0.354a\ . \tag{4.60}$$

4.4.2.3. *Coordinate transformations: truncated-octahedral to triclinic box.* A simulation performed in a truncated-octahedral box can equivalently be performed in a special type of triclinic box, by applying an appropriate coordinate transformation.[23] If the edge vectors of the cube based on which the truncated-octahedron is constructed are $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ (recall that $a = b = c$, $\alpha = \beta = \gamma = \pi/2$ and $\phi = \theta = \psi = 0$ in this case), a possible choice for the edges $\mathbf{a}_t$, $\mathbf{b}_t$ and $\mathbf{c}_t$ of the transformed triclinic box is

$$\mathbf{a}_t = \mathbf{a}\ , \quad \mathbf{b}_t = (1/2)(\mathbf{a} + \mathbf{b} + \mathbf{c}) \quad \text{and} \quad \mathbf{c}_t = (1/2)(-\mathbf{a} - \mathbf{b} + \mathbf{c})\ . \tag{4.61}$$

The corresponding box-edge lengths, box angles and Euler angles are $a_t = a$, $b_t = c_t = (\sqrt{3}/2)a$, $\alpha_t = \text{acos}(-1/3) \approx 109.5^o$, $\beta_t = \text{acos}(-1/\sqrt{3}) \approx 125.3^o$, $\gamma_t = \text{acos}(1/\sqrt{3}) \approx 54.8^o$, $\phi_t = \theta_t = 0$, and $\psi_t = 45°$. The mapping of atomic coordinates within a truncated-octahedral box to atomic coordinates within the transformed triclinic box is performed by applying shifts along the $\mathbf{a}_t$, $\mathbf{b}_t$ and $\mathbf{c}_t$ vectors. This transformation is performed by `math::truncoct_triclinic` in MD++. This formalism is applied for the generalization of grid-based pairlist algorithms (Sec. 3.3.3) and lattice-sum electrostatics (Sec. 7.4) to truncated-octahedral boxes.

4.4.3. Multiple unit-cell simulations under PBC**.** It is possible to simulate a periodic computational box consisting of multiple identical periodic copies, M_a, M_b and M_c along the three edge vectors, of a smaller unit cell. This option may be useful when trying to simulate a single unit cell of a crystal that is too small to allow for the application of a reasonably large cutoff value. The application of such multiple-unit-cell simulations is restricted to rectangular or triclinic periodic boundary conditions.

In MD++ the topology is constructed using `multicell_topo` where the normal topology is multiplied by M. Every timestep, before interactions are calculated, the configuration, that is coordinates and velocities, is prepared using `expand_configuration` where copies j of the original configuration $\boldsymbol{r}_i$ are added to the

original coordinate $\boldsymbol{r}_i$ using $\boldsymbol{r}_j = \boldsymbol{r}_i + (M_a M_b m_c \mathbf{a} + M_a m_b \mathbf{b} + m_c \mathbf{c})$ with m_a ranging from 1 to M_a, m_b from 1 to M_b, and m_c from 1 to M_c. The simulation time step is then performed. After that only the configuration of the original box is kept.

Note that the removal of the center of mass motion whenever required, is applied to charge groups and solvent molecules gathered in the individual subcells. Note also that the application of particle-mesh methods to evaluate electrostatic interactions (Sec. 7.4.4) will only give rise to exactly periodic forces if the number of P^3M grid subdivisions along each axis is an integer multiple of the corresponding number of subcell boundaries.

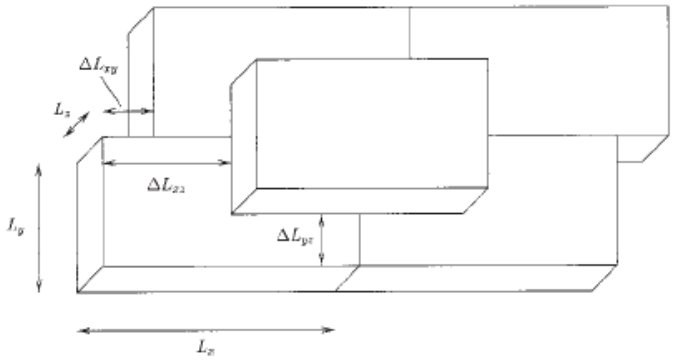


FIGURE 4.4. Rectangular-brickwall

4.4.4. Rectangular-brickwall box. A simulation performed in a triclinic box under triclinic periodic boundary conditions can equivalently be performed in a rectangular box using brickwall boundary conditions.[23] Under brickwall boundary conditions, the periodic copies of the reference box no longer systematically share common faces, but are staggered as depicted in Fig. 4.4. This transformation is applied to the triclinic box in the box-linked coordinate system, *i.e.* after rotation of the box and coordinates. The box edges of the rectangular-brickwall box will be noted $L_{x'}$, $L_{y'}$, $L_{z'}$, and the three corresponding offsets as $\Delta L_{x'y'}$, $\Delta L_{x'z'}$ and $\Delta L_{y'z'}$, primed because they refer to the Cartesian axes of the box-linked coordinate system. Out of three possible definitions, it will be further specified here that the periodic boxes only share, in the general case, common y' z' -faces and common x' -directed edges of their x' z' -faces. The edges $L_{x'}$, $L_{y'}$, and $L_{z'}$ of the rectangular box, together with the offsets $\Delta L_{x'y'}$, $\Delta L_{x'z'}$ and $\Delta L_{y'z'}$ determining the relative positions of neighboring boxes, are dictated by the lattice parameters, *i.e.* three edges and three angles, of the original triclinic box, so that the center of each rectangular box is located at a point of this triclinic lattice. To define these offsets in a unique fashion, it is further imposed that

$$-\frac{L_{x'}}{2} < \Delta L_{x'y'},\ \frac{L_{x'}}{2} \geq \Delta L_{x'z'}\ and\ -\frac{L_{y'}}{2} < \Delta L_{y'z'} \leq \frac{L_{y'}}{2} \tag{4.62}$$

Under brick-wall boundary conditions, the minimum-image function is not separable into Cartesian components, and must operate directly on a vector, that is, one has

$$\begin{aligned} z''_{ij} &= \tilde{z}'_{ij} - NINT((\tilde{z}'_{ij}/L_{z'})L_{z'}\ with\ \tilde{z}'_{ij} = z'_{ij} \\ y''_{ij} &= \tilde{y}'_{ij} - NINT((\tilde{y}'_{ij}/L_{y'})L_{y'}\ with\ \tilde{y}'_{ij} = y'_{ij} + (z''_{ij} - z''_{ij})/L_{z'} \\ x''_{ij} &= \tilde{x}'_{ij} - NINT((\tilde{x}'_{ij}/L_{x'})L_{x'} \\ with\ & \tilde{x}'_{ij} = x'_{ij} + (z''_{ij} - z''_{ij})/L_{z'} + (y''_{ij} - y''_{ij})/L_{y'} \end{aligned} \tag{4.63}$$

This formalism is applied for the generalization of grid-based pairlist algorithms (Sec. 3.3.3) to triclinic boxes and truncated-octahedral boxes, after transformation to the equivalent triclinic box, see Sec. 4.4.2.2. In the special case of a simulation performed in a regular rectangular box, the rectangular-brickwall box is identical to the original computational box and one has $L_{x'} = a$, $L_{y'} = b$, $L_{y'} = c$, $\Delta L_{x'y'} = 0$, $\Delta L_{x'z'} = 0$ and $\Delta L_{y'z'} = 0$.

The volume of the rectangular-brickwall box is equal to that of the triclinic box (Sec. 4.4.2.2) and is also identical to that of the original box.

Finally, if a triclinic box is mapped to an equivalent rectangular-brickwall box, the condition for the cut-off radius (Eq. 4.26) becomes

$$R_c \leq \frac{1}{2} \min(L_{x'}, L_{y'}, L_{z'}) \ . \tag{4.64}$$

When using a grid-based pairlist algorithm (Sec. 3.3.3), the condition is slightly more restrictive, namely

$$R_c \leq \frac{1}{2} \min(L_{x'}(1 - {G_{x'}}^{-1}), L_{y'}(1 - {G_{y'}}^{-1}), L_{z'}(1 - G_{z'})^{-1}) \ , \tag{4.65}$$

where $G_{x'}$, $G_{y'}$ and $G_{z'}$ represent the number of grid-cell subdivisions along the three axes of the box-linked coordinate system. These restrictions only apply to the specific, short- or long-range, cutoff values involved in the grid-based algorithm.

CHAPTER 5

Bonded interaction force-field terms

The potential energy or force-field term associated with *covalent interactions* is the term $\mathcal{V}^{(cov)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.5. The four contributions to this term as defined by Eq. 3.6 are described in the following sections.

5.1. Bond stretching force-field term

The potential energy term associated with *bond-stretching interactions* is the term $\mathcal{V}^{(b)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.6. It is given by

$$\mathcal{V}^{(b)}(\boldsymbol{r}^N;\mathbf{s}) = \sum_{n=1}^{N^{(b)}} V^{(b)}(b_n;k_n^{(b)},b_n^0) \ , \tag{5.1}$$

where $N^{(b)}$ is generally equal to the total number of all covalent bonds present in the system *i.e.* each covalent bond is associated with one and only one stretching term in the GROMOS force field, and $V^{(b)}$ is the function describing the potential energy associated with the stretching of a single bond. The quantity $b_n \doteq b_n(\boldsymbol{r}^N)$ represents the length of bond n in the given system configuration, *i.e.* the distance $\mathbf{r}_{ij}$ between the two atoms $i \doteq i(n)$ and $j \doteq j(n)$ connected by the covalent bond n, *i.e.* the minimum-image distance if PBC is applied, namely

$$b_n \doteq r_{ij} \tag{5.2}$$

with

$$\mathbf{r}_{ij} \doteq \mathbf{r}_i - \mathbf{r}_j \tag{5.3}$$

and

$$r_{ij} \doteq (\mathbf{r}_{ij}\cdot\mathbf{r}_{ij})^{1/2} \ . \tag{5.4}$$

The quantities $k_n^{(b)}$ and b_n^0 represent force-field parameters, force constant and reference length, respectively, characteristic for the specific bond n, as encoded by a corresponding *bond type code* $M_n^{(b)}$, *i.e.* one may write $k_n^{(b)} \doteq k^{(b)}(M_n^{(b)})$ and $b_n^0 \doteq b^0(M_n^{(b)})$.

Two different expressions can be used for the function $V^{(b)}$ in GROMOS. The *quartic* bond stretching interaction form $V^{(b)} = V^{(b,q)}$, the default, as used in GROMOS96, is defined as[24]

$$V^{(b,q)}(b;k^{(b,q)},b^0) \doteq \tfrac{1}{4}k^{(b,q)}(b^2-(b^0)^2)^2 \ . \tag{5.5}$$

The *harmonic* bond stretching interaction form $V^{(b)} = V^{(b,h)}$ is defined as

$$V^{(b,h)}(b;k^{(b,h)},b^0) \doteq \tfrac{1}{2}k^{(b,h)}(b-b^0)^2 \ . \tag{5.6}$$

The quartic form is computationally less expensive, since it avoids the square-root operation in the calculation of b from b^2. However, the efficiency gain is moderate for most systems, where computational costs are dominated by the calculation of the non-bonded interactions, and the harmonic form is more common and conceptually simpler. The procedure for interconverting the quartic and harmonic force constants is described in Sec. 18.1. We note that this procedure is not of immediate relevance to the users of GROMOS, since both force constants are explicitly included in the force-field files. The selection of one or the other form for bond stretching interactions can be made using the COVALENTFORM block of the MD++ input (Vol. 4).

The derivatives of the bond stretching term are described in Sec. 17.1 and the GROMOS parameters involved in this term are provided in Sec. 3-2.2.

5.2. Bond-angle bending force-field term

The potential energy term associated with *bond angle bending interactions* is the term $\mathcal{V}^{(\theta)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.6. It is given by

$$\mathcal{V}^{(\theta)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s}) = \sum_{n=1}^{N^{(\theta)}} V^{(\theta)}(\theta_n; k_n^{(\theta)}, \theta_n^0) \ , \tag{5.7}$$

where $N^{(\theta)}$ is generally equal to the total number of all covalent bond angles present in the system *i.e.* each definable covalent bond angle is associated with one and only one bending term in the GROMOS force field, and $V^{(\theta)}$ is the function describing the potential energy associated with the bending of a single bond angle. The quantity $\theta_n \doteq \theta_n(\boldsymbol{r}^N)$ represents the value of bond angle n in the given system configuration, *i.e.* the angle formed by the three atoms $i \doteq i(n)$, $j \doteq j(n)$ and $k \doteq k(n)$ defining the covalent bond angle n, *i.e.* the minimum-image triplet if PBC is applied, namely

$$\theta_n \doteq \arccos\left(\frac{\mathbf{r}_{ij}\cdot\mathbf{r}_{kj}}{r_{ij}r_{kj}}\right) \quad \text{with} \quad 0 \le \theta_n \le \pi \tag{5.8}$$

where

$$\mathbf{r}_{ij} \doteq \mathbf{r}_i - \mathbf{r}_j \quad \text{and} \quad \mathbf{r}_{kj} \doteq \mathbf{r}_k - \mathbf{r}_j \ , \tag{5.9}$$

and

$$r_{ij} \doteq (\mathbf{r}_{ij}\cdot\mathbf{r}_{ij})^{1/2} \quad \text{and} \quad r_{kj} \doteq (\mathbf{r}_{kj}\cdot\mathbf{r}_{kj})^{1/2} \ . \tag{5.10}$$

The quantities $k_n^{(\theta)}$ and θ_n^0 represent force-field parameters, force constant and reference bond angle, respectively, characteristic for the specific bond angle n, as encoded by a corresponding *bond-angle type code* $M_n^{(\theta)}$, *i.e.* one may write $k_n^{(\theta)} \doteq k^{(\theta)}(M_n^{(\theta)})$ and $\theta_n^0 \doteq \theta^0(M_n^{(\theta)})$.

Two different expression can be used for the function $V^{(\theta)}$ in GROMOS. The *cosine-harmonic* bond angle bending interaction form $V^{(\theta)} = V^{(\theta,c)}$, the default, as used in GROMOS96, is defined as

$$V^{(\theta,c)}(\theta; k^{(\theta,c)}, \theta^0) \doteq \tfrac{1}{2}k^{(\theta,c)}(\cos\theta - \cos\theta^0)^2 \ . \tag{5.11}$$

The *angle-harmonic* bond-angle bending interaction form $V^{(\theta)} = V^{(\theta,h)}$ is defined as

$$V^{(\theta,h)}(\theta; k^{(\theta,h)}, \theta^0) \doteq \tfrac{1}{2}k^{(\theta,h)}(\theta - \theta^0)^2 \ . \tag{5.12}$$

The cosine-harmonic form is computationally less expensive, since it avoids the arc-cosine operation in the calculation of θ from $\cos\theta$. However, the efficiency gain is moderate for most systems, where computational costs are dominated by the calculation of the non-bonded interactions, and the harmonic form is more common and conceptually simpler. Furthermore, the cosine-harmonic form may pose problems in cases where the reference bond angle is close to 0 or π, *e.g.* in linear molecules or functional groups, since the curve described by $V^{(\theta,h)}$ then becomes flat around $\theta \approx \theta^0$, *i.e.* very high force constants are required to maintain the linear bond angle geometry.

The procedure for interconverting the cosine-harmonic and harmonic force constants is described in Sec. 18.1. We note that this procedure is not of immediate relevance to the users of GROMOS, since both force constants are explicitly included in the force-field files. The selection of one or the other form for bond-angle bending interactions can be made using the COVALENTFORM block of the MD++ input (Vol. 4).

The derivatives of the bond-angle bending term are described in Sec. 17.2 and the GROMOS parameters involved in this term are provided in Sec. 3-2.3.

5.3. Improper dihedral-angle bending force-field term

The potential energy term associated with *improper dihedral-angle bending interactions*, typically controlling out-of-plane or out-of-tetrahedron distortions, is the term $\mathcal{V}^{(\xi)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.6. It is given by

$$\mathcal{V}^{(\xi)}(\boldsymbol{r}^N;\mathbf{s}) = \sum_{n=1}^{N^{(\xi)}} V^{(\xi)}(\xi_n; k_n^{(\xi)}, \xi_n^0) \ , \tag{5.13}$$

where $N^{(\xi)}$ generally corresponds to a subset of all possibly definable improper dihedral angles in the system, and $V^{(\xi)}$ is the function describing the potential energy associated with the bending of a single improper

dihedral angle. The quantity $\xi_n \doteq \xi_n(\boldsymbol{r}^N)$ represents the value of improper dihedral angle n in the given system configuration, *i.e.* the dihedral angle formed by the four atoms $i \doteq i(n)$, $j \doteq j(n)$, $k \doteq k(n)$ and $l \doteq l(n)$ defining the covalent improper dihedral angle n, *i.e.* the minimum-image quadruplet if PBC is applied, namely

$$\xi_n \doteq \text{sign}(\mathbf{r}_{ij} \cdot \mathbf{r}_{nk}) \arccos\left(\frac{\mathbf{r}_{mj} \cdot \mathbf{r}_{nk}}{r_{mj} r_{nk}}\right) \quad \text{with} \quad -\pi < \mid \xi_n \mid \leq \pi \tag{5.14}$$

where

$$\mathbf{r}_{mj} \doteq \mathbf{r}_{ij} \times \mathbf{r}_{kj} \quad \text{and} \quad \mathbf{r}_{nk} \doteq \mathbf{r}_{kj} \times \mathbf{r}_{kl} \tag{5.15}$$

and

$$r_{mj} \doteq (\mathbf{r}_{mj} \cdot \mathbf{r}_{mj})^{1/2} \quad \text{and} \quad r_{nk} \doteq (\mathbf{r}_{nk} \cdot \mathbf{r}_{nk})^{1/2} \ . \tag{5.16}$$

Note that the sign of the dihedral angle as defined by Eq. 5.14 follows the IUPAC-IUB convention,[25] and that the improper dihedral angle is undefined if either $r_{mj} = 0$ or $r_{nk} = 0$. The quantities $k_n^{(\xi)}$ and ξ_n^0 represent force-field parameters, force constant and reference improper dihedral angle, respectively, characteristic for the specific improper dihedral angle n, as encoded by a corresponding *improper dihedral angle type code* $M_n^{(\xi)}$, *i.e.* one may write $k_n^{(\xi)} \doteq k^{(\xi)}(M_n^{(\xi)})$ and $\xi_n^0 \doteq \xi^0(M_n^{(\xi)})$.

The function $V^{(\xi)}$ is always a *harmonic* function in GROMOS, *i.e.*

$$V^{(\xi)}(\xi; k^{(\xi)}, \xi^0) \doteq {}^1\!/_2 k^{(\xi)}(\xi - \xi^0)^2 \quad \text{with} \quad -\pi < \xi - \xi^0 \leq \pi \ . \tag{5.17}$$

Note that since improper dihedral angles are periodic variables, of period 2π, the interval selected for evaluating the difference $\xi - \xi^0$ must be specified, *i.e.* the interval $]-\pi; \pi]$ as indicated.

Unlike for bond-stretching and bond-angle bending terms, the summation in Eq. 5.13 only involves a subset of $N^{(\xi)}$ terms selected by considering all possibly definable improper dihedral angles in the system, *i.e.* only some among all definable improper dihedral angles are associated with a single improper dihedral-angle term in GROMOS. These improper dihedral angles are selected to keep groups of atoms close to a specified spatial configuration, typically planar or tetrahedral. For example, in an amino acid residue, the atoms CA, C, O and N are kept near a planar configuration by defining an improper dihedral C-CA-N-O with $\xi^0 = 0°$. As another example, if the CA atom of an amino acid residue carries no explicit hydrogen, *i.e.* it is a united-atom of type CH1, the atoms CA, N, C and CB are kept near a tetrahedral configuration by defining an improper dihedral angle CA-N-C-CB (L-amino acid) or CA-C-N-CB (D-amino acid) with $\xi^0 = 35.26°$. A third example is that of an aromatic ring, like in the phenylalanine amino acid residue, which is kept close to planarity by defining 6 improper dihedral angles (CG-CD1-CE1-CZ, CD1-CE1-CZ-CE2, CE1-CZ-CE2-CD2, CZ-CE2-CD2-CG, CE2-CD2-CG-CD1 and CD2-CG-CD1-CE1), all with $\xi^0 = 0°$.

The derivatives of the improper dihedral-angle bending term are described in Sec. 17.3 and the GROMOS parameters involved in this term are provided in Sec. 3-2.4.

5.4. Proper dihedral-angle torsion force-field term

The potential energy term associated with *torsional dihedral-angle bending interactions*, typically controlling, together with non-bonded interactions, the rotational barriers around covalent bonds, is the term $\mathcal{V}^{(\varphi)}(\boldsymbol{r}^N; \mathbf{s})$ in Eq. 3.6. It is given by

$$\mathcal{V}^{(\varphi)}(\boldsymbol{r}^N; \mathbf{s}) = \sum_{n=1}^{N^{(\varphi)}} V^{(\varphi)}(\varphi_n; k_n^{(\varphi)}, \varphi_n^0, m_n^{(\varphi)}) \ , \tag{5.18}$$

where $N^{(\varphi)}$ generally corresponds to a subset of all possibly definable proper, torsional dihedral angles in the system. We note that multiple terms may be associated to the covalent proper dihedral angle in the GROMOS force field. $V^{(\varphi)}$ is the function describing the potential energy contribution of the term to the torsion of the corresponding proper dihedral angle. The quantity $\varphi_n \doteq \varphi_n(\boldsymbol{r}^N)$ represents the value of proper dihedral angle n in the given system configuration, *i.e.* the dihedral angle formed by the four atoms $i \doteq i(n)$, $j \doteq j(n)$, $k \doteq k(n)$ and $l \doteq l(n)$ defining the covalent proper dihedral angle n, *i.e.* the minimum-image quadruplet if PBC is applied, namely

$$\varphi_n \doteq \text{sign}(\mathbf{r}_{ij} \cdot (\mathbf{r}_{kj} \times \mathbf{r}_{kl})) \arccos\left(\frac{\mathbf{r}_{im'} \cdot \mathbf{r}_{ln'}}{r_{im'} r_{ln'}}\right) \quad \text{with} \quad -\pi < \varphi_n \leq \pi \tag{5.19}$$

where

$$\mathbf{r}_{im'} \doteq \mathbf{r}_{ij} - \frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{kj}}{r_{kj}^2} \cdot \mathbf{r}_{kj} \quad \text{and} \quad \mathbf{r}_{in'} \doteq -\mathbf{r}_{kl} + \frac{\mathbf{r}_{kl} \cdot \mathbf{r}_{kj}}{r_{kj}^2} \cdot \mathbf{r}_{kj} \tag{5.20}$$

and

$$r_{im'} \doteq (\mathbf{r}_{im'} \cdot \mathbf{r}_{im'})^{1/2} \quad \text{and} \quad r_{in'} \doteq (\mathbf{r}_{in'} \cdot \mathbf{r}_{in'})^{1/2} \; . \tag{5.21}$$

Note that Eq. 5.19 is readily shown to be equivalent to Eq. 5.14. The sign of the dihedral angle as defined by Eq. 5.19 follows the IUPAC-IUB convention,[25] and the proper dihedral angle is undefined if either $r_{im'} = 0$ or $r_{in'} = 0$. The quantities $k_n^{(\varphi)}$, φ_n^0 and $m_n^{(\varphi)}$ represent force-field parameters, force constant, reference dihedral angle, and multiplicity, respectively. The reference dihedral angle is also called the phase shift. The multiplicity is a positive non-zero integer. These parameters are characteristic for the specific proper dihedral angle term n, as encoded by a corresponding *proper dihedral-angle type code* $M_n^{(\varphi)}$, *i.e.* one may write $k_n^{(\varphi)} \doteq k^{(\varphi)}(M_n^{(\varphi)})$, $\varphi_n^0 \doteq \varphi^0(M_n^{(\varphi)})$ and $m_n^{(\varphi)} \doteq m^{(\varphi)}(M_n^{(\varphi)})$

Two different expression can be used for the function $V^{(\varphi)}$ in GROMOS. These expressions do not differ in the resulting value of the interaction, but only in their range of application. The *symmetric* proper dihedral-angle torsion interaction form, the default, as used in GROMOS96, is defined as

$$V^{(\varphi)}(\varphi; k^{(\varphi)}, \varphi^0, m^{(\varphi)}) \doteq k^{(\varphi)}(1 + \cos\varphi^0 \cos m^{(\varphi)}\varphi) \quad \text{with} \quad \varphi^0 = 0, \pi \; . \tag{5.22}$$

In this form, the value of the phase shift is restricted to 0 or π, which leads to a potential that is symmetric with respect to the eclipsed conformation, either corresponding to a minimum or a maximum, depending on the sign of $\cos\varphi^0$. In addition, considering practical usefulness and computational efficiency, the multiplicity $m^{(\varphi)}$ may not exceed a value of six here. The *generalized* proper dihedral-angle torsion interaction form is defined as

$$V^{(\varphi)}(\varphi; k^{(\varphi)}, \varphi^0, m^{(\varphi)}) \doteq k^{(\varphi)} \left[1 + \cos(m^{(\varphi)}\varphi - \varphi^0)\right] \quad \text{with} \quad \varphi^0 \in [0, 2\pi[\; . \tag{5.23}$$

Here, no restrictions are made on the values of the phase shift and multiplicity. It is easily verified that Eqs. 5.22 and 5.23 are equivalent, within the domain of validity of the former expression. The symmetric form is computationally less expensive, since it avoids the arc-cosine operation in the calculation of φ from $\cos\varphi$, owing to the expansions

$$\begin{aligned}
\cos(1\varphi) &= \cos\varphi \\
\cos(2\varphi) &= 2\cos^2\varphi - 1 \\
\cos(3\varphi) &= 4\cos^3\varphi - 3\cos\varphi \\
\cos(4\varphi) &= 8\cos^4\varphi - 8\cos^2\varphi + 1 \\
\cos(5\varphi) &= 16\cos^5\varphi - 20\cos^3\varphi + 5\cos\varphi \\
\cos(6\varphi) &= 32\cos^6\varphi - 48\cos^4\varphi + 18\cos^2\varphi - 1 \; .
\end{aligned} \tag{5.24}$$

However, the efficiency gain is moderate for most systems, where computational costs are dominated by the calculation of the non-bonded interactions, and the generalized form offers the advantage of being more flexible. In particular, it may be useful in cases where a potential that is not symmetric with respect to the eclipsed conformation may permit an improved fitting of rotational profiles against experimental or theoretical data.

Note that since proper dihedral angles are periodic variables, of period 2π, the interval selected for evaluating the φ should in principle be specified. However, this selection has no influence of the result of Eqs. 5.22 and 5.23.

Unlike for bond-stretching and bond-angle bending terms, the summation in Eq. 5.18 only involves a subset of $N^{(\varphi)}$ terms selected by considering all possibly definable proper dihedral angles in the system, *i.e.* only some among all definable proper dihedral angles are associated with a single, or few proper dihedral-angle terms in GROMOS. These proper dihedral-angles are selected to control, in combination with non-bonded interactions, the rotational barriers around covalent bonds for this selection. The following guidelines can be given:

1. In general, for any bond between atoms j and k, only one set of atoms i, j, k and l is chosen to define a proper dihedral angle.
2. For bonds between atoms j and k in rigid, planar rings (aromatics), no proper torsional dihedral angle is defined, but rather improper dihedrals are used to maintain the planarity of the ring.

3. To obtain correct torsional-angle energy profiles, several torsional dihedral angles with different parameters can be defined on the same set of atoms i, j, k and l. This is for instance done for the protein backbone ϕ and ψ angles (in the GROMOS parameter set 54A7), in sugar rings or along the backbone of a nucleotide sequence.

The selection of one or the other form for proper dihedral torsion interactions can be made using the COVALENTFORM block of the MD++ input (Vol. 4).

The derivatives of the proper dihedral-angle torsion term are described in Sec. 17.4 and the GROMOS parameters involved in this term are provided in Sec. 3-2.5.

CHAPTER 6

van der Waals interactions

6.1. Introduction

In GROMOS, the van der Waals (vdW) interaction term, $\mathcal{V}^{(vdw)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.7, is represented by a Lennard-Jones function, and the term is partitioned as a sum of two contributions.

$$\mathcal{V}^{(vdw)}(\boldsymbol{r}^N;\mathbf{s}) = \sum_{i}^{\mathcal{N}_a-1} \sum_{j=i+1}^{\mathcal{N}_a} \left[\frac{C_{12}(i,j)}{\mathbf{r}_{ij}^6} - C_6(i,j) \right] \cdot \frac{1}{\mathbf{r}_{ij}^6} \tag{6.1}$$

where $C_{12}(i,j)$ is the van der Waals repulsion coefficient for the interaction between atoms or sites i and j, $C_6(i,j)$ is the dispersion coefficient, and $\mathbf{r}_{ij}$ the distance between the two sites.

$$C_{12}(i,j) = 4\epsilon\sigma^{12} \tag{6.2}$$

and

$$C_6(i,j) = 4\epsilon\sigma^{6} \tag{6.3}$$

in which ϵ is the depth of the potential well, σ is the corresponding distance between the two sites where $\mathcal{V}^{(vdw)} = 0$. The minimum-energy distance is $2^{1/6}\sigma$.

The total van der Waals interaction of all particles is in principle obtained from a summation of all pairs of atoms within the pairlist (Sec. 3.3). However, in practice a number of pairs are excluded from the summation. These pairs are defined as excluded neighbour atom pairs. This is discussed in Sec. 6.2.

The derivatives of the van der Waals interaction term are described in Sec. 17.5.

6.2. Excluded neighbours

Atoms i and j that are covalently bound are called first neighbour atoms and if they are each covalently bound to one common neighbour atom, they are called second neighbours. Due to the short distance $\mathbf{r}_{ij}$ between first or second neighbours, the non-bonded interaction (Eq. 6.1) between such neighbour atoms i and j will be very large. In addition, first and second neighbour interactions are represented by bond-stretching and bond-angle bending interaction terms. Therefore, *first and second neighbours are excluded* from the summation. *Third neighbours* are only in special cases excluded, e.g. between atoms in or attached to aromatic rings, or between specific atoms in certain carbohydrate building blocks.

Lists of excluded atoms are kept in the molecular topology file, see Vol. 4. The atom sequence numbers j of the excluded neighbours of a solute atom i are listed in ascending order and i<j. In the solvent part of the molecular topology all atoms that form a solvent molecule are excluded neighbours of each other.

For the GROMOS force fields the excluded atom information can be found in the molecular topology building block files *.mtb.

6.3. Normal van der Waals interactions

The non-bonded interaction van der Waals parameters C_{12}(i,j) and C_6(i,j) in formula (Eq. 6.1) depend on the atom type or more specifically the integer atom codes I = IAC[i] and J = IAC[j] of the atoms with atom sequence numbers i and j. The integer atom codes of the various types of atoms are listed in Chap. 3-3. Lists of integer atom codes are kept in the molecular topology file Vol. 4. The molecular topology file contains the full matrix of interaction parameters for all combinations of integer atom types. In this way it is possible to change the van der Waals interaction between each pair of atom types independently. In practice, the

GROMOS van der Waals parameters for an atom pair with integer atom codes I and J are derived from single atom van der Waals parameters using the relations

$$C_6(I,J) = {C_6}^{1/2}(I,I){C_6}^{1/2}(J,J) \tag{6.4}$$

and

$$C_{12}(I,J) = {C_{12}}^{1/2}(I,I){C_{12}}^{1/2}(J,J) \tag{6.5}$$

For the GROMOS force fields, the single atom van der Waals parameters $C_6(I,I)^{1/2}$ and $C_{12}(I,I)^{1/2}$ are given in the third and fourth columns of Tabs. 3-3.7 (45A4 and 45B4) and 3-3.22 (54A7 and 54B7) as a function of integer atom code or non-bonded atom type.

The GROMOS force fields do not contain a special term in the interaction function (Eq. 3.4) that mimics hydrogen bonding. The hydrogen bonding capacity of molecules is the result of a balance between Coulomb and van der Waals attraction and repulsion. In order to mimic correctly the hydrogen bonding properties of polar atoms, their van der Waals repulsion has been increased over the value resulting from the use of ${C_{12}}^{1/2}$ from the fourth column in Tabs. 3-3.7 and 3-3.22. The fifth column of these tables contains the ${C_{12}}^{1/2}$(I,I) values to be used between polar atoms. For correct modelling of hydrogen bonds between atoms that are part of a charged moiety, like the OM atom in a COO^- group and a NL atom in a NH_3^+ group, the repulsive part of the van der Waals interaction has been increased even more. The sixth column in Tabs. 3-3.7 and 3-3.22 contains the ${C_{12}}^{1/2}$(I,I) values to be used between oppositely charged atoms. In Tabs. 3-3.10 and 3-3.24 it is denoted which values for ${C_{12}}^{1/2}$(I,I) and ${C_{12}}^{1/2}$(J,J) are to be used in formula (Eq. 6.5) for obtaining C_{12}(I,J).

If another combination rule than formulae Eqs. 6.4 and 6.5 is to be used to obtain C_6(I,J) and C_{12}(I,J) or if these mixed atom type pair parameters are not related to the single atom type van der Waals parameters ${C_6}^{1/2}$(I,I) and ${C_6}^{1/2}$(J,J) or ${C_{12}}^{1/2}$(I,I) and ${C_{12}}^{1/2}$(J,J), the interaction parameters for the pair of atom types (I,J) must be explicitly given as C_6(I,J) and C_{12}(I,J). In the GROMOS force field this is the case for the van der Waals parameters for the solvent chloroform. The non-standard mixed atom type pair van der Waals parameters are listed in Tab. 3-3.9. Furthermore, exceptions for individual pairs of atoms i,j may be defined at the building block level, using the LJEXCEPTION listings (see Vol. 4).

For the GROMOS force fields the non-bonded atom type information and the normal van der Waals interaction parameters can be found in the interaction function parameter files *.ifp.

The van der Waals parameters of the 45B4 force field which is to be used for in vacuo calculations, are identical to those of the 45A4 force field which is to be used for calculations including solvent, except for the repulsive parameters of the OM and NL atoms of which the charge has been reduced when deriving the 45B4 force field from the 45A4 one.

For an atom pair with integer atom codes I and J, the ${C_{12}}^{1/2}$(I,I) value in formula (Eq. 6.5) is taken from the fourth column in Tabs. 3-3.7 and 3-3.22 if the matrix element (I,J) equals 1; it is taken from the fifth column if the matrix element is equal to 2 and from the sixth element if it is equal to 3. Similarly, the ${C_{12}}^{1/2}$(J,J) value in formula (Eq. 6.5) is selected using the matrix element (J,I).

6.4. Third-neighbour van der Waals interaction

When the van der Waals parameters for the united atoms (CH1, CH2, CH3, CR1) are applied to atoms that are separated by three covalent bonds, so-called third neighbours, they induce a too large repulsion in gauche conformations.

In order to avoid this effect, the smaller van der Waals parameters that are given in Tab. 3-3.11 (45A4, 45B4) and Tab. 3-3.26 (53A5, 53B5, 54A7, 54B7) are used for united atoms when obtaining C_6(I,J) and C_{12}(I,J) from (Eq. 6.4-Eq. 6.5) for third-neighbour atoms. The van der Waals parameters for *third-neighbour or 1-4 interactions* are kept in the molecular topology file (Vol. 4).

Lists of third-neighbour atoms are kept in the molecular topology file (Vol. 4). The sequence numbers j of an atom that is a third neighbour of the atom with sequence number i, are listed in ascending order and $i < j$. In the solvent part of the molecular topology all atoms that form a solvent molecule are excluded neighbour atoms of each other, so will have neither normal nor third-neighbour van der Waals interaction.

The list of third neighbours can be derived from the list of covalent bonds occurring in the "solute". This is done in program *make_top* upon construction of a molecular topology file from the molecular building blocks.

For the GROMOS force fields the third-neighbour van der Waals interaction parameters can be found in the interaction function parameter files (*.ifp). GROMOS offers a possibility to specify the van der Waals parameters for a specific atom pair, thereby overruling the interaction parameters as derived from the normal (or third-neighbour) interaction parameters. This can be done by introducing a LJEXCEPTIONS block in the molecular topology file (see Sec. 4-3.2).

6.5. Soft-core interactions

The van der Waals interaction in Eq. 6.1 and the electrostatic interaction between charges of equal sign in Eq. 7.2 become infinitely large for $\mathbf{r}_{ij}$ approaching zero. Using interactions Eqs. 6.1 and 7.2 the atoms have a so-called hard core, which restricts the sampling of configurational space. By smoothening the potential energy surface $\mathcal{V}^{(vdw)}(\boldsymbol{r}^N;\mathbf{s})$ the sampling can be considerably enhanced. Using a so-called *soft-core interaction* function for the interactions between particles i and j of the form[26]

$$\mathcal{V}_{ij}^{(vdw),sc}(\boldsymbol{r}^N;\mathbf{s}) = \left[\frac{C_{12}(i,j)}{\alpha_{LJ}(i,j)\lambda^2 C_{126}(i,j)+(r_{ij})^6} - C_6(i,j)\right]\cdot\frac{1}{\alpha_{LJ}(i,j)\lambda^2 C_{126}(i,j)+(r_{ij})^6} \tag{6.6}$$

and

$$\mathcal{V}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s}) = \frac{q_i q_j}{4\pi\epsilon_0\epsilon_1}\left[\frac{1}{[\alpha_C(i,j)\lambda^2+(\mathbf{r}_{ij})^2]^{1/2}} - \frac{\frac{1}{2}C_{rf}(\mathbf{r}_{i,j})^2}{\left[\alpha_C(i,j)\lambda^2+R_{rf}^2\right]^{3/2}} - \frac{(1-\frac{1}{2}C_{rf})}{R_{rf}}\right] \tag{6.7}$$

The singularity at $r_{ij}=0$ is removed and the interaction function is smoothened for $\lambda\neq 0$. α_{LJ} (i,j) is a softness parameter and C_{126} is defined in Eq. 14.79.

In GROMOS, the soft-core interaction Eq. 6.6 can only be selected in the framework of a so-called perturbation molecular topology required for free energy calculations using the λ-coupling parameter approach. Therefore, the soft-core interaction is described further in Chap. 14.

CHAPTER 7

Electrostatic interactions

7.1. Introduction

In this chapter, we provide the possible forms employed in GROMOS for the term $\mathcal{V}^{(ele)}(\boldsymbol{r}^N;\mathbf{s})$ in Eq. 3.7. In all cases, this quantity can be further partitioned as a sum of contributions from a *pairwise*, a *self* and a *surface* term, namely

$$\mathcal{V}^{(ele)}(\boldsymbol{r}^N;\mathbf{s}) \doteq \mathcal{V}^{(ele,pws)}(\boldsymbol{r}^N;\mathbf{s}) + \mathcal{V}^{(ele,slf)}(\mathbf{s}) + \mathcal{V}^{(ele,srf)}(\boldsymbol{r}^N;\mathbf{s}) \; . \tag{7.1}$$

The expressions defining these three contributions differ whether one decides to employ a *reaction-field* (RF) or a *lattice-sum* (LS) scheme to evaluate the long-distance electrostatic interactions. The RF scheme is applicable under either FBC or PBC, while the LS scheme is only applicable under PBC. The features common to the two schemes are summarized in Sec. 7.2. The specifics of the RF and LS schemes are then provided in Secs. 7.3 and 7.4, respectively.

7.2. Common features

The pairwise contribution $\mathcal{V}^{(ele,pws)}$ in Eq. 7.1 can be written in the form

$$\mathcal{V}^{(ele,pws)}(\boldsymbol{r}^N;\mathbf{s}) \doteq (4\pi\epsilon_0\epsilon_{cs})^{-1} \sum_{i=1}^{\mathcal{N}_a-1} \sum_{j=i+1}^{\mathcal{N}_a} q_i q_j [\Psi_{ij}^{(ele)}(\boldsymbol{r}^N;\mathbf{s}) - \delta_{ij}^{(exc)}(\mathbf{s})\overline{r}_{ij}{}^{-1}] \tag{7.2}$$

where ϵ_0 is the permittivity of vacuum, and ϵ_{cs} the relative permittivity of the medium in which the simulation is performed. In simulations with explicit solvent we generally set $\epsilon_{cs} = 1$. q_i and q_j are the charges of particles i and j, respectively, $\overline{\mathbf{r}}_{ij}$ the (minimum-image, under PBC) vector connecting j to i (norm $\overline{r}_{ij}$), $\Psi_{ij}^{(ele)}(\boldsymbol{r}^N;\mathbf{s})$ is the *electrostatic influence function* associated with the particle pair $i-j$, $\delta_{ij}^{(exc)}$ is the *non-bonded exclusion indicator* for the particle pair $i-j$, 1 if the atoms are excluded, 0 otherwise. In the GROMOS force field, first- and second covalent neighbours are normally excluded from electrostatic interactions, as well as some specific pairs of atoms that are separated by three or more covalent bonds, as described in Sec. 6.2. The definition of excluded pairs is assumed to be encompassed in the force-field parameter vector $\mathbf{s}$. Although Coulomb's law would suggest that $\Psi_{ij}^{(ele)}(\boldsymbol{r}) = r_{ij}^{-1}$, this choice is almost never directly applicable in practice (except for systems under FBC and in the absence of exclusions; see below).

The self contribution $\mathcal{V}^{(ele,slf)}$ in Eq. 7.1 can be written in the form

$$\mathcal{V}^{(ele,slf)}(\boldsymbol{r}^N;\mathbf{s}) \doteq (8\pi\epsilon_0\epsilon_{cs})^{-1} \sum_{i=1}^{\mathcal{N}_a} q_i^2 \Psi^{(ele,slf)} \tag{7.3}$$

where $\Psi^{(ele,slf)}$ is the *electrostatic self influence function*, defined by

$$\Psi^{(ele,slf)} \doteq \lim_{\mathbf{r}_i \to \mathbf{r}_o} \Psi_{oi}^{(ele)}(\boldsymbol{r};\mathbf{s}) \; . \tag{7.4}$$

The fact that $\Psi^{(ele,slf)}$ solely depends on the boundary conditions and associated parameters will be explicited in the following sections.

Finally, the surface contribution $\mathcal{V}^{(ele,srf)}$ in Eq. 7.1 arises from the definition of the medium surrounding an infinite periodic system (under PBC). It can be written in the form

$$\mathcal{V}^{(ele,srf)}(\boldsymbol{r}^N;\mathbf{s}) \doteq [2\pi\epsilon_0(2\epsilon_{ls}+1)\mathcal{V}]^{-1}\mathbf{M}^2 \tag{7.5}$$

where ϵ_{ls} is the relative permittivity of the medium surrounding the infinite periodic system, $\mathcal{V}$ is the volume of the computational box, and $\mathbf{M}$ the box dipole moment.

The exact forms of taken by the functions $\Psi_{ij}^{(ele)}$ and $\Psi^{(ele,slf)}$ in Eqs. 7.2 and 7.4, respectively, will be detailed in Secs. 7.3 and 7.4, respectively.

7.3. Reaction-field (RF) interactions

The RF scheme is applicable under either FBC or PBC. The electrostatic influence function $\Psi_{ij}^{(ele)}$ (Eq. 7.2) in the RF case, which is a cutoff-based scheme, may take two different forms depending on whether a *particle (atom)* (AT) or a *charge-group* (CHG) cutoff truncation is applied.

In the AT case, this function reads

$$\Psi_{ij}^{(ele)}(\boldsymbol{r}) \doteq H(R_C - \overline{r}_{ij})\psi^{(RF)}(\overline{r}_{ij}) \ , \tag{7.6}$$

(independent of $\mathbf{s}$ here) while in the CHG case, it reads

$$\Psi_{ij}^{(ele)}(\boldsymbol{r};\mathbf{s}) \doteq H(R_C - \overline{R}_{ij}(\boldsymbol{r}))\psi^{(RF)}(\overline{r}_{ij}) \ , \tag{7.7}$$

where $\overline{\mathbf{r}}_{ij}$ is the (minimum-image, under PBC) vector vector connecting particle j to particle i (norm $\overline{r}_{ij}$), $\overline{\mathbf{R}}_{ij}$ the (minimum-image, under PBC) vector connecting the centers of the CHG to which the two particles belong (norm $\overline{R}_{ij}$), R_C is the cutoff distance and H the Heaviside step function (1 if its argument is positive, 0 otherwise). In the two above equations, the function $\psi^{(RF)}$ is the same and reads

$$\psi^{(RF)}(r) \doteq \frac{1}{r} - \frac{C_{RF}}{2{R_{RF}}^3}r^2 - \frac{1-(1/2)C_{RF}}{R_{RF}} \tag{7.8}$$

with

$$C_{RF} \doteq \frac{(2\epsilon_{cs}-2\epsilon_{RF})(1+\kappa_{RF}R_{RF}) - \epsilon_{RF} * (\kappa_{RF}R_{RF})^2}{(\epsilon_{cs}+2\epsilon_{RF})(1+\kappa_{RF}R_{RF}) + \epsilon_{RF} * (\kappa_{RF} * R_{RF})^2} \ , \tag{7.9}$$

where ϵ_{RF} is the RF permittivity, κ_{RF} the inverse Debye screening length and R_{RF} the RF cutoff.

The three terms resulting from the insertion of Eqs. 7.6 or 7.7 into Eq. 7.2 are termed *coulombic*, *distance-dependent* and *distance-independent*, namely

$$\mathcal{V}^{(ele,pws,RF-CB)} \doteq (4\pi\epsilon_0)^{-1}\sum_i\sum_j \frac{q_iq_j}{\overline{r}_{ij}} \tag{7.10}$$

$$\mathcal{V}^{(ele,pws,RF-RF)} \doteq (4\pi\epsilon_0)^{-1}\sum_i\sum_j -\frac{q_iq_jC_{RF}{\overline{r}_{ij}}^2}{2{R_{RF}}^3} \tag{7.11}$$

and

$$\mathcal{V}^{(ele,pws,RF-RC)} \doteq (4\pi\epsilon_0)^{-1}\sum_i\sum_j -\frac{q_iq_j(1-(1/2)C_{RF})}{R_{RF}} \tag{7.12}$$

In GROMOS, Eq. 7.10 is not evaluated for excluded atoms (Sec. 6.2), while Eqs. 7.11 and 7.12 are evaluated for these atoms as well, unless the simulation is performed in the GROMOS96 compatibility mode, see Vol. 4.

Inserting $\Psi_{ij}^{(ele)}$ from Eqs. 7.6 or 7.7 into Eq. 7.4, one finds that the self influence function is in both cases given by

$$\Psi^{(ele,slf)} = -\frac{1-(1/2)C_{RF}}{R_{RF}} \tag{7.13}$$

The fact that $\Psi^{(ele,slf)}$ solely depends on the boundary conditions and associated parameters will be explicited in the following sections.

The surface term is as in Eq. 7.5 - if we decide to add it.

7.4. Lattice-sum (LS) interactions

7.4.1. Introduction. Lattice-sum (LS) methods rely on two key principles : (*i*) the treatment of electrostatic interactions as exactly periodic within the simulated system ; (*ii*) the splitting of the interaction into a short-range component, evaluated by direct summation over the pairs of atoms, and a long-range component, evaluated by Fourier series.

To ensure overall neutrality of the system, each "charge" is actually represented by a charge density defined by a periodic point charge plus a homogeneous neutralizing background charge density filling the infinite periodic system. This background charge has absolutely no influence for overall neutral systems. For charged systems, it permits the calculation of a finite electrostatic energy, containing a self-interaction (Wigner) term, but has no influence on the forces.

The box dipole moment comes into the definition of the surface term (Eq. 7.30). In the general case, this quantity is not translationally invariant. Different definitions of the reference computational box (*i.e.* of $\mathbf{r}_c$) may lead to different values of $\mathbf{M}$.

Lattice sum methods rely on the use of a charge-shaping function with a width a to split the electrostatic potential into a real-space contribution and a reciprocal-space contribution, plus a constant. The width parameter a should be smaller than the short-range cutoff R_{cp} (input parameter RCUTP). Otherwise, an error will be issued. When the shaping function is a TP function (N_γ=1...10 in Tab. 7.1), the real-space interaction may be computed exactly provided that all atom pairs within a distance a or smaller are at any time included in the pairlist. When all atom pairs within a distance R_{cp} or smaller are at any time included in the pairlist, the optimal value of a is R_{cp}. When the shaping function is a Gaussian (N_γ=-1 in Tab. 7.1), the real-space interaction is not computed exactly, because the Gaussian is infinite-ranged. In this case, the real-space interaction is computed for all atom pairs within the pairlist, and will be accurate when a/R_{cp} is small enough. In many cases, $a \approx R_{cp}/3$ is a reasonable choice.

The two lattice-sum methods available differ in the way they evaluate the reciprocal-space contribution to the electrostatic energy and forces (Secs. 7.4.2, 7.4.3 and 7.4.4) : the Ewald method is based on direct summation over reciprocal-space lattice vectors while the particle-particle-particle-mesh (PPPM) method makes use of a fast Fourier transform (FFT) algorithm.

It will be useful to make the following definitions.

We consider a periodic system of Nq charges q_i at positions $\mathbf{r}_i$ within a general triclinic computational box with arbitrary orientation (Sec. 4.4.1). We further define the box dipole moment

$$\mathbf{M} = \sum_{i=1}^{Nq} q_i \left(\mathbf{r}_i - \mathbf{r}_c\right) , \tag{7.14}$$

where $\mathbf{r}_c$ is the center of the computational box, the box overall charge

$$S = \sum_{i=1}^{Nq} q_i , \tag{7.15}$$

and the box overall square charge

$$\tilde{S}^2 = \sum_{i=1}^{Nq} q_i^2 . \tag{7.16}$$

The electrostatic influence function $\Psi_{ij}^{(ele)}$ (Eq. 7.2) in the LS case, which is a periodic scheme (no cutoff), may formally be written

$$\Psi_{ij}^{(ele)}(\boldsymbol{r}) \doteq \psi^{(LS)}(\mathbf{r}_{ij}) = 4\pi \mid \underline{\mathbf{L}} \mid^{-1} \sum_{\mathbf{l},l\neq 0} k^{-2} \exp[i\mathbf{k}\cdot\mathbf{r}_{ij}] , \tag{7.17}$$

where $\underline{\mathbf{L}}$ is a matrix containing the Cartesian component of the box edge vectors in its columns (triclinic computational box), $\mathbf{l}$ a *lattice vector* with (positive or negative) integer components, $\mathbf{k} = 2\pi\underline{\mathbf{L}}^{-1}\mathbf{l}$ the associated *reciprocal-lattice vector*, and $\mathbf{r}_{ij} \doteq \mathbf{r}_i - \mathbf{r}_j$ the vector connecting sites i and j (in the computational box). The influence function $\psi^{(LS)}(\mathbf{x})$ describes the (periodic) electrostatic influence at position $\mathbf{x}$ relative to a point charge screened by a homogeneous neutralizing background charge density of opposite magnitude. Here, the electrostatic influence is just the electrostatic potential divided by $(4\pi\epsilon_0)^{-1}q$.

Because the summation over reciprocal-lattice vectors involved in this equation converges very slowly, in practice, $\Psi_{ij}^{(ele)}$ is partitioned into two contributions, termed the *real-space* and the *reciprocal-space* contributions (Ewald splitting).

7.4.1.1. *Charge-shaping function.* Lattice-sum methods rely on the use of a charge-shaping function $a^{-3}\gamma(a^{-1}r)$ of width a to split the electrostatic potential into a real-space contribution and a reciprocal-space contribution, plus a constant. In practice, it is assumed that the charge-shaping function satisfies the condition

$$\gamma(a^{-1}r) = 0 \quad \text{for} \quad r \geq R_{cp} , \quad \text{with} \quad R_{cp} \leq (1/2)\min\{L_x, L_y, L_z\} , \tag{7.18}$$

where R_{cp} is the real-space cutoff distance (input parameter RCUTP), which implies that the switch function $\eta(a^{-1}r)$ also vanishes beyond R_{cp}. This condition can be enforced in a strict fashion for all truncated-polynomial charge-shaping functions (Tab. 7.1; $N_\gamma = 0...10$) by setting a (input parameter ASHAPE) equal to R_{cp}. It can be enforced in an approximate manner for the Gaussian charge-shaping function (Tab. 7.1;

$N_\gamma = -1$) by setting $a \ll R_{cp}$. In many cases, $a \approx R_{cp}/3$ is a reasonable choice. If $a > R_{cp}$, an error will be issued in the FORCE routine. If $R_{cp} > (1/2)\min\{L_x, L_y, L_z\}$, an error will be issued.

The charge-shaping function is normalized to satisfy the condition

$$4\pi\, a^{-3} \int_0^\infty dr\, r^2\, \gamma(a^{-1}r) = 1 \ . \tag{7.19}$$

The following definitions are related to the charge-shaping function. The Fourier coefficients of (a lattice sum of) the charge-shaping function are given by

$$\hat{\gamma}(ak) = \begin{cases} 4\pi k^{-1}\, a^{-3} \int_0^\infty dr\, r\, \sin(kr)\, \gamma(a^{-1}r) & \text{for } \ k \neq 0 \\ 1 & \text{for } \ k = 0 \end{cases} \ , \tag{7.20}$$

where reciprocal-lattice vectors are defined as $\mathbf{k} = 2\pi \underline{\mathbf{L}}^{-1}\mathbf{l}$ with $\mathbf{l} \in \mathbb{Z}^3$.

The switch function $\eta(a^{-1}r)$ associated with the charge-shaping function is defined by

$$\eta(a^{-1}r) = 4\pi\, a^{-3} \int_r^\infty d\rho\, \rho\, (\rho - r)\, \gamma(a^{-1}\rho) \ . \tag{7.21}$$

Finally, the constants A_1, A_2 and A_3 are defined as

$$A_1 = -4\pi\, \mathcal{V}^{-1} \int_0^\infty dr\, r\, \eta(a^{-1}r) \ , \tag{7.22}$$

$$A_2 = 4\pi\, \mathcal{V}^{-1} \sum_{\mathbf{l}\in\mathbb{Z}^3\,,\, l\neq 0} k^{-2}\, \hat{\gamma}(ak) \ , \tag{7.23}$$

and

$$A_3 = \lim_{r\to 0} [\, \sum_{\mathbf{n}\in\mathbb{Z}^3} \parallel \mathbf{r} + \underline{\mathbf{L}}\,\mathbf{n} \parallel^{-1} \eta(a^{-1} \parallel \mathbf{r} + \underline{\mathbf{L}}\,\mathbf{n} \parallel) \quad - \quad r^{-1}\,] \ . \tag{7.24}$$

This limit becomes independent of the direction of $\mathbf{r}$ when $\mathbf{r} \to \mathbf{0}$.

The shaping functions ccurrently implemented and indexed by N_γ, and the related Fourier coefficients, switch functions and A-constants are listed in Tabs. 7.1 - 7.6. The charge-shaping function is selected through the value of NSHAPE in the LONGRANGE block.

The complementary error function required for the Gaussian charge-shaping function is calculated via the Chebyshev approximation (see ref.[27]). The evaluation of the exponential function for both $\eta(\xi)$ and $\hat{\gamma}(\kappa)$ might lead to computational underflows when their argument is negative and large. We did not encounter such cases for now, but if underflows are flagged at run time, they are likely to come from there.

7.4.1.2. *Electrostatic energy.* Due to self-interactions (Wigner term) and interactions with the dielectric continuum outside the infinite periodic system (surface term; this term vanishes only in the limit $\epsilon_{LS} \to \infty$, referred to as conducting or tinfoil boundary conditions), $\mathcal{V}^{(ele)}$ is actually a free energy. However, within the force field, it plays the role of a normal energy term. Note also that both E_γ and E_A also contain self energies. However, these cancel out and $E_\gamma + E_\eta + E_A$ is truly a pairwise energy (*i.e.* this quantity vanishes for a system consisting of a single charge). This is not the case of $\mathcal{V}^{(ele,slf)}$ and $\mathcal{V}^{(ele,srf)}$.

The electrostatic (reversible-charging) energy $\mathcal{V}^{(ele)}$ of the periodic system of charges can now be written as[28]

$$\mathcal{V}^{(ele)} = \mathcal{V}^{(ele,pws,LS-KS)} + \mathcal{V}^{(ele,pws,LS-RS)} + \mathcal{V}^{(ele,srf)} + \tag{7.25}$$

with

$$E_\gamma = (2\epsilon_0\epsilon_{ls}\, V)^{-1} \sum_{i=1}^{N_q} \sum_{j=1}^{N_q} q_i q_j \sum_{\mathbf{l}\in\mathbb{Z}^3\,,\, l\neq 0} k^{-2} \hat{\gamma}(ak)\, \cos(\mathbf{k}\cdot\mathbf{r}_{ij}) \ , \tag{7.26}$$

$$E_\eta = (4\pi\epsilon_0\epsilon_{ls})^{-1} \sum_{i=1}^{N_q} \sum_{j=1\,,\, j>i}^{N_q} q_i\, q_j \sum_{\mathbf{n}\in\mathbb{Z}^3} \parallel \mathbf{r}_{ij} + \underline{\mathbf{L}}\,\mathbf{n} \parallel^{-1} \eta(a^{-1} \parallel \mathbf{r}_{ij} + \underline{\mathbf{L}}\,\mathbf{n} \parallel) \ , \tag{7.27}$$

$$E_A = (8\pi\epsilon_0\epsilon_{ls})^{-1} \left[A_1\, S^2 - (A_1 + A_2)\, \tilde{S}^2\right] \ , \tag{7.28}$$

$$\mathcal{V}^{(ele,slf)} = (8\pi\epsilon_0\epsilon_{ls})^{-1}\, (A_1 + A_2 + A_3)\, \tilde{S}^2 \ , \tag{7.29}$$

N_γ	m	$\pi\gamma(\xi)$
-1	∞	$\pi^{-1/2}\, e^{-\xi^2}$
0	0	$(3/4)\, H(1-\xi)$
1	1	$3(1-\xi)\, H(1-\xi)$
2	2	$(15/2)\, (1-\xi)^2\, H(1-\xi)$
3	2	$(15/4)\, (1-\xi)^2(1+2\xi)\, H(1-\xi)$
4	3	$(105/16)\, (1-\xi)^3(3\xi+1)\, H(1-\xi)$
5	4	$(21/2)\, (1-\xi)^4(4\xi+1)\, H(1-\xi)$
6	4	$(63/8)\, (1-\xi)^4(5\xi^2+4\xi+1)\, H(1-\xi)$
7	5	$(45/4)\, (1-\xi)^5(8\xi^2+5\xi+1)\, H(1-\xi)$
8	6	$(165/32)\, (1-\xi)^6(35\xi^2+18\xi+3)\, H(1-\xi)$
9	6	$(165/64)\, (1-\xi)^6(64\xi^3+69\xi^2+30\xi+5)\, H(1-\xi)$
10	7	$(2145/128)\, (1-\xi)^7(21\xi^3+19\xi^2+7\xi+1)\, H(1-\xi)$

TABLE 7.1. Charge-shaping functions currently implemented. Related quantities are listed in Tabs. 7.2-7.6. N_γ : code of the function (input switch NSHAPE ; $N_\gamma = 0...10$: optimal TP-function of order N_γ; $N_\gamma = -1$: Gaussian) ; m : convergence rate of $\hat{\gamma}(\kappa)$ towards zero (convergence is as $\kappa^{-(m+2)}$ when $\kappa \to \infty$) ; $\pi\gamma(\xi)$: charge-shaping function amplified by π (the actual shaping function is $a^{-3}\gamma(a^{-1}r)$) ; $H(\xi)$: Heaviside function ($H(\xi) = 1$ when $\xi \geq 0$, zero otherwise). Note that the Gaussian function is infinite-ranged.

N_γ	$\eta(\xi)$
-1	erfc(ξ)
0	$(1/2)\, (1-\xi)^2(\xi+2)\, H(1-\xi)$
1	$(1-\xi)^3(\xi+1)\, H(1-\xi)$
2	$(1/2)\, (1-\xi)^4(3\xi+2)\, H(1-\xi)$
3	$(1/4)\, (1-\xi)^4(4\xi^2+7\xi+4)\, H(1-\xi)$
4	$(1/8)(1-\xi)^5(15\xi^2+19\xi+8)\, H(1-\xi)$
5	$(1-\xi)^6(3\xi^2+3\xi+1)\, H(1-\xi)$
6	$(1/16)\, (1-\xi)^6(35\xi^3+66\xi^2+51\xi+16)\, H(1-\xi)$
7	$(1/8)(1-\xi)^7(32\xi^3+49\xi^2+31\xi+8)\, H(1-\xi)$
8	$(1/16)\, (1-\xi)^8(105\xi^3+136\xi^2+73\xi+16)\, H(1-\xi)$
9	$(1/32)\, (1-\xi)^8(160\xi^4+335\xi^3+312\xi^2+151\xi+32)\, H(1-\xi)$
10	$(1/128)(1-\xi)^9(1155\xi^4+2075\xi^3+1665\xi^2+697\xi+128)\, H(1-\xi)$

TABLE 7.2. Switch functions corresponding to the charge-shaping functions currently implemented (see Tab. 7.1). N_γ : code of the function ; $\eta(\xi)$: switch function (the real-space interaction function is $r^{-1}\eta(a^{-1}r)$) ; erfc : complementary error function.

and

$$\mathcal{V}^{(ele,srf)} = [2\epsilon_0\epsilon_{ls}\,(2\epsilon_{LS}+1)\,\mathcal{V}]^{-1}\,\mathbf{M}^2\ , \tag{7.30}$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$, ϵ_0 is the dielectric permittivity of vacuum, and ϵ_{LS} (input parameter EPSLS) the relative permittivity of the medium "surrounding" the infinite periodic system. If the switch EPSLS is set to 0.0, conducting boundary conditions will be used, *i.e.* $\epsilon_{LS} \to \infty$. The expression for the surface term is obtained by assuming that the infinite periodic system is built by assembling a (roughly) spherical assembly of N periodic cells centered at $\mathbf{r}_c$, the center of the reference computational box, with $N \to \infty$. In this case, if the medium outside the assembly is a continuum of permittivity ϵ_{LS}, the Onsager self energy reads (for N

N_γ	$-\eta'(\xi)$
-1	$2\pi^{-1/2}e^{-\xi^2}$
0	$(3/2)\,(1-\xi)(\xi+1)\,H(1-\xi)$
1	$2(1-\xi)^2(2\xi+1)\,H(1-\xi)$
2	$(5/2)\,(1-\xi)^3(3\xi+1)\,H(1-\xi)$
3	$(3/4)\,(1-\xi)^3(8\xi^2+9\xi+3)\,H(1-\xi)$
4	$(21/8)(1-\xi)^4(5\xi^2+4\xi+1)\,H(1-\xi)$
5	$3(1-\xi)^5(8\xi^2+5\xi+1)\,H(1-\xi)$
6	$(9/16)\,(1-\xi)^5(35\xi^3+47\xi^2+25\xi+5)\,H(1-\xi)$
7	$(5/8)(1-\xi)^6(64\xi^3+69\xi^2+30\xi+5)\,H(1-\xi)$
8	$(55/16)\,(1-\xi)^7(21\xi^3+19\xi^2+7\xi+1)\,H(1-\xi)$
9	$(15/32)\,(1-\xi)^7(128\xi^4+203\xi^3+141\xi^2+49\xi+7)\,H(1-\xi)$
10	$(65/128)(1-\xi)^8(231\xi^4+312\xi^3+186\xi^2+56\xi+7)\,H(1-\xi)$

TABLE 7.3. Derivative of the switch functions corresponding to the charge-shaping functions currently implemented (see Tabs. 7.1 and 7.2). N_γ : code of the function ; $-\eta'(\xi)$: derivative switch function amplified by -1 ($\eta'(\xi) = d\eta(\xi)/d\xi$).

N_γ	$\kappa^{N_\gamma+3}\hat{\gamma}(\kappa)$
-1	$\kappa^2 e^{-\kappa^2/4}$
0	$3[-\kappa C + S]$
1	$12[2 - 2C - \kappa S]$
2	$60[2\kappa + \kappa C - 3S]$
3	$90[8 + (\kappa^2-8)C - 5\kappa S]$
4	$630[8\kappa + 7\kappa C + (\kappa^2-15)S]$
5	$5040[4(\kappa^2-6) - (\kappa^2-24)C + 9\kappa S]$
6	$7560[48\kappa - (\kappa^2-57)\kappa C + 3(4\kappa^2-35)S]$
7	$75600[24(\kappa^2-8) - 3(5\kappa^2-64)C - (\kappa^2-87)\kappa S]$
8	$831600[8\kappa(\kappa^2-24) + (\kappa^2-123)\kappa C - (18\kappa^2-315)S]$
9	$1247400[192(\kappa^2-10) + (\kappa^4-207\kappa^2+1920)C - (22\kappa^2-975)\kappa S]$
10	$16216200[64\kappa(\kappa^2-30) + (26\kappa^2-1545)\kappa C + (\kappa^4-285\kappa^2+3465)S]$

TABLE 7.4. Fourier coefficients corresponding to the shaping functions currently implemented (see Tab. 7.1). N_γ : code of the function ; $\kappa^{N_\gamma+3}\hat{\gamma}(\kappa)$: Fourier coefficient amplified by $\kappa^{N_\gamma+3}$ (the actual Fourier coefficient is $\hat{\gamma}(ak)$) ; C : short notation for $\cos(\kappa)$; S : short notation for $\sin(\kappa)$.

large)

$$\Delta G_o(\epsilon_{LS}) = -(8\pi\epsilon_0)^{-1}\frac{2(\epsilon_{LS}-1)}{2\epsilon_{LS}+1}\frac{N\mathbf{M}^2}{[(3/4)\pi^{-1}NV]^{1/3}}$$

The quantity $\mathcal{V}^{(ele,srf)}$ being a correction from conducting (tinfoil) boundary conditions to finite-permittivity is then given by $\mathcal{V}^{(ele,srf)} = \Delta G_o(\epsilon_{LS}) - \Delta G_o(\infty)$.

The interpretation of the different contributions to $\mathcal{V}^{(ele)}$ in Eq. 7.25 is given below, with reference to the following terminology for charge densities (which can be added to or subtracted from one another): point charge (p), γ-shaped charge (γ), and homogeneous background charge (b). The term E_γ represents the electrostatic energy (including self interaction) of a set of $(p-b)$-charges of magnitude $\{q_i\}$ at positions $\{\mathbf{r}_i\}$ in the potential generated by the corresponding periodic system of $(\gamma-b)$-charges. Since the potential is a non-singular and generally smooth function of position, E_γ is conveniently evaluated in reciprocal

N_γ	$\kappa^{N_\gamma+4}\hat{\gamma}'(\kappa)$
-1	$-(1/2)\kappa^4 e^{-\kappa^2/4}$
0	$3[3\kappa C + (\kappa^2-3)S]$
1	$12[-8-(\kappa^2-8)C + 5\kappa S]$
2	$60[-8\kappa - 7\kappa C - (\kappa^2-15)S]$
3	$90[-48-(9\kappa^2-48)C - (\kappa^2-33)\kappa S]$
4	$630[-48\kappa + (\kappa^2-57)\kappa C - 3(4\kappa^2-35)S]$
5	$5040[-24(\kappa^2-8) + 3(5\kappa^2-64)C + (\kappa^2-87)\kappa S]$
6	$7560[-384\kappa + 3(6\kappa^2-187)\kappa C + (\kappa^4-141\kappa^2+945)S]$
7	$75600[-192(\kappa^2-10) - (\kappa^4-207\kappa^2+1920)C + (22\kappa^2-975)\kappa S]$
8	$831600[-64\kappa(\kappa^2-30) - (26\kappa^2-1545)\kappa C - (\kappa^4-285\kappa^2+3465)S]$
9	$1247400[-1920(\kappa^2-12) - 15(2\kappa^4-203\kappa^2+1536)C - (\kappa^4-405\kappa^2+12645)\kappa S]$
10	$16216200[-640\kappa(\kappa^2-36) + (\kappa^4-545\kappa^2+22005)\kappa C - 5(7\kappa^4-936\kappa^2+9009)S]$

TABLE 7.5. derivative of the Fourier coefficients corresponding to the shaping functions currently implemented (see Tabs. 7.1 and 7.4). N_γ : code of the function ; $\kappa^{N_\gamma+4}\hat{\gamma}'(\kappa)$: derivative Fourier coefficient amplified by $\kappa^{N_\gamma+4}$ (the actual derivative of the Fourier coefficient is $\hat{\gamma}'(ak)$).

-1	$-(1/2)\kappa^4 e^{-\kappa^2/4}$	
0	$3[3\kappa C + (\kappa^2-3)S]$	
N_γ	$-V\pi^{-1}a^{-2}A_1$	$-aA_3$
-1	1	$2\pi^{-1/2}$
0	2/5	3/2
1	4/15	2
2	4/21	5/2
3	3/14	9/4
4	1/6	21/8
5	2/15	3
6	8/55	45/16
7	4/33	25/8
8	4/39	55/16
9	10/91	105/32
10	2/21	455/128

TABLE 7.6. A-constants corresponding to the shaping functions currently implemented (see Tab. 7.1). N_γ : code of the function ; $-V\pi^{-1}a^{-2}A_1$: constant A_1 amplified by $-V\pi^{-1}a^{-2}$; $-aA_3$: constant A_3 amplified by $-a$. These results are derived using Eq. 7.31, and thus valid exactly when $a \leq \min\{L_x, L_y, L_z\}$ (TP functions) or as an approximation when $a \ll \min\{L_x, L_y, L_z\}$ (Gaussian).

space, using the Ewald method[29] or the PPPM method[30] (see Sec. 7.4.3 and Sec. 7.4.4). The term E_η represents the electrostatic energy (excluding self interaction) of a set of $(p-b)$-charges of magnitude $\{q_i\}$ at positions $\{\mathbf{r}_i\}$ in the potential generated by the corresponding periodic system of $(p-\gamma)$-charges. With an appropriate choice of charge-shaping function, the function $\eta(a^{-1}r)$ can be made a quickly decreasing function of distance, in which case E_η is conveniently evaluated by direct (real-space) summation over the charge pairs (Sec. 7.4.1). The terms E_A and $\mathcal{V}^{(ele,slf)}$ (Sec. 7.4.1.3) are configuration-independent. The term E_A eliminates the self-energies present in E_γ and contains a small correction due to the constraint of zero average potential within the periodic system. The term $\mathcal{V}^{(ele,slf)}$ accounts for the self-energy of set

of $(p-b)$-charges of magnitude $\{q_i\}$ in the potential generated by the corresponding periodic system of $(p-b)$-charges (Wigner term[31–33]). Finally, the term $\mathcal{V}^{(ele,srf)}$ accounts for the interaction of the infinite periodic system with a dielectric continuum of relative permittivity ϵ_{LS} surrounding it. In the absence of the surface term, lattice-sum methods lead to conducting or tinfoil boundary conditions, which corresponds to $\epsilon_{LS} \to \infty$. The surface term is thus a correction to account for a medium of finite permittivity instead.

Only three energy contributions are reported by GROMOS : the pairwise energy $\mathcal{V}^{(ele,pws)} = E_\gamma + E_\eta + E_A$, the self-energy term $\mathcal{V}^{(ele,slf)}$, and the surface term $\mathcal{V}^{(ele,srf)}$. These quantities are stored in the arrays normally for the Coulomb, distance-dependent reaction field and distance-independent reaction field contributions. The splitting between $\mathcal{V}^{(ele,pws)}$ and $\mathcal{V}^{(ele,slf)}$ is only meaningful when the constant A_2 is calculated (switch NA2CLC≠0). Otherwise, it is arbitrary and only the sum is correct.

7.4.1.3. *Constant and self-energy terms.* The constant term E_A and the self-energy term $\mathcal{V}^{(ele,slf)}$ are given by Eqs. 7.28 and 7.29, respectively, where the A-constants are defined by Eqs. 7.22, 7.23 and 7.24. When Eq. 7.18 is satisfied the $\mathbf{n}$-sum in Eq. 7.24 can be restricted to the $\mathbf{n} = \mathbf{0}$ term, leading to

$$A_3 = \lim_{r \to 0} r^{-1} \left[\eta(a^{-1}r) \quad - \quad 1 \right] . \tag{7.31}$$

Because the Gaussian charge-shaping function is infinite-ranged, Eq. 7.18 is never exactly satisfied, and Eq. 7.31 only applies here as an approximation. In this case, A_1 and A_3 have analytical expressions for a given charge-shaping function (Tab. 7.6). In the general case, the constant A_2 must be computed numerically by direct summation. In practice, this is done by evaluating

$$A_2(l_{max}) = 4\pi\, V^{-1} \sum_{\mathbf{l} \in \mathbb{Z}^3\,,\, l \neq 0\,,\, l_x, l_y, l_z \leq l_{max}} k^{-2}\, \hat{\gamma}(ak) \tag{7.32}$$

for increasing values of l_{max}, until a user-specified relative tolerance (input parameter TOLA2) is reached. In the specific case of a cubic unit cell ($L_x = L_y = L_z = L$), this evaluation is replaced by[28,33] $A_2 \approx \xi_{EW} L^{-1} - A_1 - A_3$ with $\xi_{EW} = -2.83792748$ (macro WIGNER_CUBE in define.inc). The quantity A_2 is calculated (input switch NA2CLC≥2) either (*i*) once at the beginning of the simulation (constant-volume simulation), (*ii*) whenever an energy output (input switches NTPR and NTWE) is required (constant-pressure simulation and NA2CLC=2) or (*iii*) every step (constant-pressure simulation and NA2CLC=3 or 4). When NA2CLC=0, the value of A_2 is set to zero. When NA2CLC=2, only the energies are affected by the calculation, not the virial. Thus, the estimation of A_2 is not required unless an energy output is needed. The value of A_2 is assumed constant between updates. When NA2CLC≥ 3, both the energies and the virial are affected by the calculation. Thus, the estimation of A_2 is required at every step in a constant-pressure simulation.

The quantity A_2 calculated through Eq. 7.32 represents the exact (in the limit of large l_{max}) value of A_2 entering into $\mathcal{V}^{(ele,slf)}$ (Eq. 7.29). However, because the reciprocal-space interaction energy is evaluated with a finite precision (*i.e.* through the Ewald or PPPM method), this value of A_2 will only approximately remove the reciprocal-space self-energy when included in E_A (Eqs. 7.28). In particular, using this A_2 value in the expression for E_A will lead to a pairwise energy $E_\eta + E_\gamma + E_A$ that will not exactly vanish for a system consisting of a single charge. Although for most practical purposes, this approximation (corresponding to NA2CLC=2) is sufficient, it is possible to compute a more accurate method-dependent value $\tilde{A}_2$ to be used in the evaluation of E_A, *i.e.*

$$E_A = (8\pi\epsilon_0)^{-1} \left[A_1\, S^2 - (A_1 + \tilde{A}_2)\, \tilde{S}^2 \right] , \tag{7.33}$$

The calculation of this quantity is feasible for the the Ewald (input switch NA2CLC=1 or 3) or PPPM (input switch NA2CLC=3 or 4).

The E_A and $\mathcal{V}^{(ele,slf)}$ terms give rise to no force contribution. The corresponding virial contributions are given in Sec. 17.7. Although the A_2 term in E_A is configuration-dependent, the corresponding force on the atoms is neglected

The calculation of the A_2 term may be entirely skipped (input switch NA2CLC=0). In this case, A_2 is set to zero and the A_2 contributions are omitted in the evaluation of both E_A and $\mathcal{V}^{(ele,slf)}$ (Eqs. 7.28 and 7.29). As a consequence, the splitting between pairwise and self contributions becomes arbitrary, but the sum of the two quantities (and thus the overall electrostatic energy) remains correct within the approximation $A_2 \approx \tilde{A}_2$. Skipping the calculation of A_2 (and/or $\tilde{A}_2$) may be advantageous in constant-pressure simulations, where these quantities must be recalculated at every step.

The surface term $\mathcal{V}^{(ele,srf)}$ is given by Eq. 7.30 where $\mathbf{M}$ is defined by Eq. 7.14 and ϵ_{LS} is the relative permittivity of the medium surrounding the infinite periodic system (input parameter EPSLS; a value of 0.0 selects an infinite permittivity, *i.e.* conducting or tinfoil boundary conditions). The corresponding force on atom i is given by

$$\mathbf{f}_{srf,i} = -[\epsilon_0 (2\epsilon_{LS} + 1) V]^{-1} q_i \mathbf{M} . \tag{7.34}$$

The corresponding virial contribution is discussed in Sec. 17.7.

Energy partitioning. The partitioning of the E_γ (Eq. 7.26; for PPPM: Eq. 7.72) term into energy groups requires multiple reciprocal-space evaluations. For N_e (N_e>1) energy groups, the $N_e(N_e+1)/2-1$ quantities

$$(E_\gamma)_{IJ}(2\epsilon_0 \mathcal{V})^{-1} \sum_{\mathbf{l}\in G\,,\, l\neq 0} \hat{G}_g(\mathbf{k_l}) \mid (\hat{s}_g(\mathbf{k_l}))_{IJ} \mid^2 \tag{7.35}$$

with $I = 1..N_e - 1$ and $J = I..N_e$ are evaluated. $(E)_{N_eN_e}$ is obtained by subtracting the sum of the other terms from E_γ.

The splitting of the E_A and $\mathcal{V}^{(ele,slf)}$ into energy groups is based on the splitting of the quantities S^2 and $\tilde{S}^2$ (Eqs. 7.15 and 7.16) as

$$\tilde{S}^2_{IJ} = \delta_{IJ} \sum_{i\in I} q_i^2 \tag{7.36}$$

and

$$S^2_{IJ} = (2 - \delta_{IJ})(\sum_{i\in I} q_i)(\sum_{j\in J} q_j) \tag{7.37}$$

where I and J denote two energy groups with $I < J$. These definitions ensure

$$\sum_I \sum_{J\geq I} \tilde{S}^2_{IJ} = \tilde{S}^2 \quad \text{and} \quad \sum_I \sum_{J\geq I} S^2_{IJ} = S^2 .$$

In this case, the pairwise group contributions to E_A and $\mathcal{V}^{(ele,slf)}$ (Eqs. 7.28 and 7.29) read

$$(E_A)_{IJ} = (8\pi\epsilon_0)^{-1} [A_1 S^2_{IJ} - (A_1 + A_2) \tilde{S}^2_{IJ}] , \tag{7.38}$$

$$(\mathcal{V}^{(ele,slf)})_{IJ} = (8\pi\epsilon_0)^{-1} (A_1 + A_2 + A_3) \tilde{S}^2_{IJ} , \tag{7.39}$$

If Eq. 7.33 is used instead for E_A together with an exact evaluation of $\tilde{A}_2$ (input switch NA2CLC=3), the splitting must be done by evaluating separately $(\tilde{A}_2)_{II}$ for each group based on Eq. 7.104 (with $\tilde{S}^2_{II}$ instead of $\tilde{S}^2$). In this case, one has

$$(E_A)_{IJ} = (8\pi\epsilon_0)^{-1} \{A_1 [(S^2)_{IJ} - (\tilde{S}^2)_{IJ} - \delta_{IJ}(\tilde{A}_2)_{II}(\tilde{S}^2)_{II}\} . \tag{7.40}$$

The splitting of the $\mathcal{V}^{(ele,srf)}$ term into energy groups is based on the splitting of the quantity $\mathbf{M}^2$ (Eqs. 7.14) as

$$(\mathbf{M}^2)_{IJ} = (2 - \delta_{IJ})[\sum_{i\in I} q_i (\mathbf{r}_i - \mathbf{r}_c)][\sum_{j\in J} q_j (\mathbf{r}_j - \mathbf{r}_c)] , \tag{7.41}$$

In this case, the pairwise group contributions to $\mathcal{V}^{(ele,srf)}$ (Eq. 7.30) reads

$$(\mathcal{V}^{(ele,srf)})_{IJ} = [2\epsilon_0 (2\epsilon_{LS} + 1) V]^{-1} (\mathbf{M}^2)_{IJ} . \tag{7.42}$$

When perturbation is applied, the soft core is only applied to the 1/r component of the real-space interaction, but not to the complement $\eta - r^{-1}$.

7.4.2. Real-space interactions in LS electrostatics. In the most general form, the real-space contribution to the electrostatic interactions is given by Eq. 7.27. The switch functions η corresponding to the charge-shaping functions implemented are listed in Tab. 7.2. In practice, it is assumed that the charge-shaping function satisfies the condition

$$\gamma(a^{-1}r) = 0 \quad \text{for} \quad r \geq R_{cp} \ , \quad \text{with} \quad R_{cp} \leq (1/2)\min\{L_x, L_y, L_z\} \ , \tag{7.43}$$

where R_{cp} is the real-space cutoff distance (input parameter RCUTP), which implies that the switch function $\eta(a^{-1}r)$ also vanishes beyond R_{cp}. This condition can be enforced in a strict fashion for all truncated-polynomial charge-shaping functions (Tab. 7.1; $N_\gamma = 0...10$) by setting a (input parameter ASHAPE) equal to R_{cp}. It can be enforced in an approximate manner for the Gaussian charge-shaping function (Tab. 7.1; $N_\gamma = -1$) by setting $a \ll R_{cp}$. In many cases, $a \approx R_{cp}/3$ is a reasonable choice. If $a > R_{cp}$, an error will be issued in the FORCE routine. If $R_{cp} > (1/2)\min\{L_x, L_y, L_z\}$, an error will be issued. In this case, and taking into account that Coulombic interaction between excluded covalent neighbours should be removed, one may rewrite Eq. 7.27 as

$$\begin{aligned} E_\eta \quad = \quad & (4\pi\epsilon_0)^{-1} \sum_{i=1}^{N_q} \sum_{j=1\,,\,j>i\,,\,\overline{r}_{ij}<R_{cp}\,,\,j\notin Exc(i)}^{N_q} q_i\, q_j\, \overline{r}_{ij}^{-1}\, \eta(a^{-1}\overline{r}_{ij}) \\ & +(4\pi\epsilon_0)^{-1} \sum_{i=1}^{N_q} \sum_{j=1\,,\,j>i\,,\,j\in Exc(i)}^{N_q} q_i\, q_j\, \overline{r}_{ij}^{-1}\, [\eta(a^{-1}\overline{r}_{ij}) - 1] \ , \end{aligned} \tag{7.44}$$

where $\overline{\mathbf{r}}_{ij}$ is the minimum-image vector corresponding to $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and the notation $j \in Exc(i)$ indicates that atom j belongs to the exclusion list of atom i. First and second covalent neighbours are excluded from electrostatic interactions. Only the interaction between atom i and the nearest periodic copy of atom j must be removed, not the interaction of i with the other periodic copies of j. Therefore, a contribution in $\overline{r}_{ij}^{-1}$ must be subtracted. It is also assumed that all pairs of excluded neighbours are within a minimum-image distance smaller than R_{cp}.

When using truncated-polynomial charge-shaping functions, the evaluation of E_η will only be exact provided that all atom pairs within a distance smaller than a are included into the short-range (charge-group or atomic) pairlist. This may not be the case because (i) charge-group pairs which are too far away to be in a molecular pairlist may still contain atom pairs at a distance smaller than R_{cp}, and (ii) the charge-group or molecular pairlist may not be updated every step. To alleviate the first source of inaccuracy if using a molecular pairlist, it is recommended to define each solute atom as its own charge group in the topology and to set $a = R_{cp} - d$ where d is the maximal distance between the first atom and any other atom within a solvent molecule. To alleviate the second source of inaccuracy, it is recommended either to update the pairlist very frequently or to use a slightly extended pairlist with R_n (input parameter RCUTN) larger than R_{cp}.

When using a Gaussian, the evaluation of E_η is always approximate. If care is taken that the pairlist contains all atom pairs within a minimum-image distance smaller than R_{cp} (previous footnote), it is likely that it will also contain pairs with distances slightly above R_{cp}. To avoid the inclusion of such arbitrariness in the calculation, these additional pairs are ignored and the interaction is restricted solely to pairs within R_{cp}. This restriction can be removed by setting the macro GAUSSIAN_LOOSE_CUTOFF in define.inc. The same restriction to pairs within R_{cp} is applied to Lennard-Jones interactions and can be removed by setting the macro VDW_LOOSE_CUTOFF in define.inc.

Note that no real-space interaction is calculated in intermediate range of charge-group or atom pairs with distances between R_{cp} and R_{cl} (input parameter RCUTL).

The corresponding atomic forces (Eq. 17.32) and contributions to the virial are discussed in Sec. 17.7.

7.4.3. Ewald reciprocal-space interactions in LS **electrostatics.** In the Ewald method, the reciprocal-space energy E_γ defined by Eq. 7.26, as well as the corresponding forces and virial, are evaluated by direct summation over reciprocal-lattice vectors.

For improved computational speed the triple-sum (over $\mathbf{l}$, i and j) in Eq. 7.26 is rewritten as a double-sum (over $\mathbf{l}$ and i)

$$E_\gamma = (2\epsilon_0 V)^{-1} \sum_{\mathbf{l}\in\mathbb{W}\,,\,l\neq 0} k^{-2}\, \hat{\gamma}(ak)\, [\, C^2(\mathbf{k}) + S^2(\mathbf{k})\,] \ , \tag{7.45}$$

with the definitions

$$C(\mathbf{k}) = \sum_{i=1}^{N_q} q_i \cos \mathbf{k} \cdot \mathbf{r}_i \qquad \text{and} \qquad S(\mathbf{k}) = \sum_{i=1}^{N_q} q_i \sin \mathbf{k} \cdot \mathbf{r}_i \ . \tag{7.46}$$

The equivalence between the two equations follows from writing

$$\sum_{i=1}^{N_q} \sum_{j=1}^{N_q} q_i q_j \sum_{\mathbf{l} \in \mathbb{W}, l \neq 0} k^{-2} \hat{\gamma}(ak) \cos(\mathbf{k} \cdot \mathbf{r}_{ij}) = \mathrm{Re} \{ \sum_{\mathbf{l} \in \mathbb{W}, l \neq 0} k^{-2} \hat{\gamma}(ak) \mid \sum_{i=1}^{N_q} q_i e^{i \mathbf{k} \cdot \mathbf{r}_i} \mid^2 \} \ ,$$

and noting that the square norm of a real number is always real. Note that the use of Eq. 7.45 rather than Eq. 7.26 is considerably more efficient, but does no longer allow for a pairwise decomposition of the contributions to E_γ (and to the corresponding forces and virial).

The force on atom i corresponding to Eq. 7.45 can be found in Eq. 17.33, and the corresponding virial contribution is given in Sec. 17.7.

The reciprocal-lattice vectors are defined as $\mathbf{k} = 2\pi^t \underline{\mathbf{L}}^{-1} \mathbf{l}$, where $\underline{\mathbf{L}}$ is the matrix $\underline{\mathbf{L}}$ containing as columns the Cartesian components of the box vectors (Sec. 4.4.1) and $\mathbf{l}$ is a vector with integer components. In the general case of a triclinic box, it is conveninent to introduce the oblique fractional coordinates $\check{\mathbf{r}}_i = \underline{\mathbf{L}}^{-1} \mathbf{r}_i$, and the oblique reciprocal-lattice vector $\check{\mathbf{k}} = 2\pi \mathbf{l}$. In this case, it is easily seen that any occurrence of the scalar product $\mathbf{k} \cdot \mathbf{r}_i$ can be replaced by $\check{\mathbf{k}} \cdot \check{\mathbf{r}}_i$.

The summation in Eq. 7.45 is restricted to the subset $\mathbb{W}$ of integer vectors $\mathbf{l}$ defined by components l_x, l_y, and l_z in the ranges $[-l_{x,max}; l_{x,max}]$, $[-l_{y,max}; l_{y,max}]$, and $[-l_{z,max}; l_{z,max}]$, and by a norm of k smaller or equal to a cutoff value k_c.

7.4.3.1. *Implementation.* The trigonometric functions involved in the two latter quantities in Eq. 7.46 can be written as

$$\begin{aligned} \cos \mathbf{k} \cdot \mathbf{r}_i \quad = \quad & c_{x,i,l_x} c_{y,i,l_y} c_{z,i,l_z} - c_{x,i,l_x} s_{y,i,l_y} s_{z,i,l_z} \\ & - s_{x,i,l_x} c_{y,i,l_y} s_{z,i,l_z} - s_{x,i,l_x} s_{y,i,l_y} c_{z,i,l_z} \end{aligned} \tag{7.47}$$

and

$$\begin{aligned} \sin \mathbf{k} \cdot \mathbf{r}_i \quad = \quad & - s_{x,i,l_x} s_{y,i,l_y} s_{z,i,l_z} + s_{x,i,l_x} c_{y,i,l_y} c_{z,i,l_z} \\ & + c_{x,i,l_x} s_{y,i,l_y} c_{z,i,l_z} + c_{x,i,l_x} c_{y,i,l_y} s_{z,i,l_z} \ , \end{aligned} \tag{7.48}$$

with

$$c_{\mu,i,l} = \cos 2\pi L_\mu^{-1} l r_{i,\mu} \quad \text{and} \quad s_{\mu,i,l} = \sin 2\pi L_\mu^{-1} l r_{i,\mu} \ . \tag{7.49}$$

In practice, the quantities defined by Eq. 7.49 are precomputed by recursion and stored into an array for all the allowed positive values of l_x, l_y, and l_z. The corresponding recursion equations are

$$c_{\mu,i,l} = \begin{cases} 1 & \text{if} \ \ l = 0 \\ \cos 2\pi L_\mu^{-1} r_{i,\mu} & \text{if} \ \ l = 1 \\ c_{\mu,i,l-1} c_{\mu,i,1} - s_{\mu,i,l-1} s_{\mu,i,1} & \text{otherwise} \end{cases} \tag{7.50}$$

and

$$s_{\mu,i,l} = \begin{cases} 0 & \text{if} \ \ l = 0 \\ \sin 2\pi L_\mu^{-1} r_{i,\mu} & \text{if} \ \ l = 1 \\ c_{\mu,i,l-1} s_{\mu,i,1} + s_{\mu,i,l-1} c_{\mu,i,1} & \text{otherwise} \end{cases} \ . \tag{7.51}$$

The results for $l > 1$ follow from expansion into complex exponentials. The corresponding quantities for negative l values are not stored into the array, since they are simply given by

$$c_{\mu,i,-l} = c_{\mu,i,l} \quad \text{and} \quad s_{\mu,i,-l} = -s_{\mu,i,l} \ . \tag{7.52}$$

The maximal values for l_x, l_y, and l_z corresponding to wavectors of $\mathbb{W}$ are $\max\{l_{x,max}; k_d L_x\}$, $\max\{l_{y,max}; k_d L_y\}$, and $\max\{l_{z,max}; k_d L_z\}$.

In a rectangular box, an increase in computational efficiency can be obtained by noting that the terms in the $\mathbf{l}-$sum involved in Eq. 7.45 are invariant upon changing $\mathbf{k}$ to $-\mathbf{k}$. Thus, the summation can be restricted to the half-space with $l_x \geq 0$, and the resulting energies and forces (and virial contribution) multiplied by two. More precisely, to avoid double counting, the half-space should include the (i) vectors with $l_x > 0$, (ii) the vectors with $l_x = 0$ and $l_y > 0$, and (iii) the vectors with $l_x = 0, l_y = 0$ and $l_z > 0$. The zero wavevector is in any case excluded from the summations.

An alternative way is to observe that, due the symmetry properties of the trigonometric functions, the quantities $C^2(\mathbf{k})$ and $S^2(\mathbf{k})$ in Eq. 7.46 do not contain any cross-terms when expanded through Eqs. 7.47 or 7.48, *i.e.*

$$C^2(\mathbf{k}) = C_o^2(\mathbf{k}) + C_x^2(\mathbf{k}) + C_y^2(\mathbf{k}) + C_z^2(\mathbf{k}) \tag{7.53}$$

and

$$S^2(\mathbf{k}) = S_o^2(\mathbf{k}) + S_x^2(\mathbf{k}) + S_y^2(\mathbf{k}) + S_z^2(\mathbf{k}) \ , \tag{7.54}$$

with

$$\begin{array}{lll} C_o(\mathbf{k}) = \sum_{i=1}^{N_q} q_i c_{x,i,l_x} c_{y,i,l_y} c_{z,i,l_z} & , & C_x(\mathbf{k}) = \sum_{i=1}^{N_q} q_i c_{x,i,l_x} s_{y,i,l_y} s_{z,i,l_z} \\ C_y(\mathbf{k}) = \sum_{i=1}^{N_q} q_i s_{x,i,l_x} c_{y,i,l_y} s_{z,i,l_z} & , & C_z(\mathbf{k}) = \sum_{i=1}^{N_q} q_i s_{x,i,l_x} s_{y,i,l_y} c_{z,i,l_z} \end{array} \tag{7.55}$$

and

$$\begin{array}{lll} S_o(\mathbf{k}) = \sum_{i=1}^{N_q} q_i s_{x,i,l_x} s_{y,i,l_y} s_{z,i,l_z} & , & S_x(\mathbf{k}) = \sum_{i=1}^{N_q} q_i s_{x,i,l_x} c_{y,i,l_y} c_{z,i,l_z} \\ S_y(\mathbf{k}) = \sum_{i=1}^{N_q} q_i c_{x,i,l_x} s_{y,i,l_y} c_{z,i,l_z} & , & S_z(\mathbf{k}) = \sum_{i=1}^{N_q} q_i c_{x,i,l_x} c_{y,i,l_y} s_{z,i,l_z} \end{array} \ . \tag{7.56}$$

Due to symmetry Eq. 7.45 is then easily rewritten as a sum over the positive octant

$$\begin{aligned} E_\gamma \ = \ & 4(\epsilon_0 V)^{-1} \sum_{\mathbf{l}\in\mathbb{W}',\, l\neq 0} k^{-2}\, \hat{\gamma}(ak)\, \sigma_{\mathbf{k}} [\, C_o^2(\mathbf{k}) + C_x^2(\mathbf{k}) + C_y^2(\mathbf{k}) + C_z^2(\mathbf{k}) \\ & + S_o^2(\mathbf{k}) + S_x^2(\mathbf{k}) + S_y^2(\mathbf{k}) + S_z^2(\mathbf{k}) \,] \ , \end{aligned} \tag{7.57}$$

where $\mathbb{W}'$ denotes the restriction of $\mathbb{W}$ to wavevectors with positive or zero components, and $\sigma_{\mathbf{k}}$ is a symmetry factor equal to 2^{-n}, where n is the number of vanishing wavevector components. The force is given in Eq. 17.34.

The reciprocal-space Ewald contribution to the atomic virial W_μ (corresponding to the energy term E_γ defined by Eq. 7.26 and evaluated as Eq. 7.45) is given in Eq. 17.38.

7.4.3.2. *Calculation of the $\tilde{A}_2$ self term.* In the Ewald case, the quantity $\tilde{A}_2$ is evaluated as

$$\tilde{A}_2 = (\epsilon_0 \epsilon_{ls} V)^{-1} \sum_{\mathbf{l}\in\mathbb{W},\, l\neq 0} k^{-2}\, \hat{\gamma}(ak) \ . \tag{7.58}$$

As discussed above, an increase in computational efficiency can be obtained by restricting the summation to a half-space and doubling the result, or even, summing over an octant and multiplying the result by eight. Because the calculation of $\tilde{A}_2$ in the Ewald case is inexpensive, while this quantity generally represents a reasonably-accurate estimate for A_2, it is possible to suppress the calculation of A_2 by Eq. 7.32 and set $A_2 = \tilde{A}_2$ in this case (input switch NA2CLC=1, only allowed for Ewald).

7.4.4. PPPM **reciprocal-space interactions in** LS **electrostatics.** The reciprocal-space contribution E_γ to the total electrostatic energy evaluated through the LS method (Eq. 7.26) may be calculated via the particle-particle particle-mesh (PPPM) algorithm of Hockney and Eastwood.[30] Alternatively, this algorithm may be applied to the calculation of the reciprocal-space contribution E_ω to the total electrostatic energy evaluated through the LSERF method.

For a general triclinic computational box, the PPPM algorithm relies on the discretization of the box by means of a grid (mesh). The number of grid subdivisions along each of the box axes must be even. The algorithm consists of six steps : (*i*) assignment of the charge density associated with the atomic partial charges to the grid points by means of an assignment function ; (*ii*) conversion of the charge density grid to reciprocal space by means of a three-dimensional fast Fourier transform (3D-FFT) ; (*iii*) solution for the reciprocal-space electrostatic potential via multiplication by an optimized influence function ; (*iv*) conversion of the reciprocal-space potential grid to real space by means of a 3D-FFT ; (*v*) evaluation of the electrostatic field on the grid by means of a finite-difference operator ; (*vi*) interpolation of the field at the location of the atomic partial charges by means of the same assignment function used in the first step. In the *ik*-differentiation variant, the fifth and sixth steps are replaced by : (*v*) evaluation of the reciprocal-space field through multiplication by $i\mathbf{k}$; (*iv*) conversion of the reciprocal-space field grid to real space by means of three 3D-FFTs, one for each field component.

The influence function describes the electrostatic potential generated at the different grid points by a unit $(\gamma - b)$-charge at the origin (Sec. 7.4). It is stored in reciprocal-space as its corresponding value at each of the reciprocal-space grid points. A key to the accuracy of the algorithm is to preoptimize this function so that it compensates for errors inherent to the discretization process, the use of an approximate assignment function and the use of an approximate finite-difference operator. When the virial is to be calculated or

when the box dimensions may vary in the course of a simulation, six grids are computed simultaneously, containing the relevant derivatives of the optimal influence function with respect to the box parameters. The optimization of the influence function (and the evaluation of its derivatives when required) is computationally expensive. In simulations without variations of the box parameters, however, this function is constant (as well as its derivatives) and the calculation needs to be performed only once at the beginning of a simulation. In simulations involving a variation of the box parameters, the accuracy of the influence function may progressively deteriorate with time as the box changes shape and size. Two mechanisms are then used to improve the accuracy of the current influence function at reasonable computational costs. First, the derivative information computed together with the optimized influence function is used to apply a first-order correction to the current influence function upon variation of the box parameters. Second, the accuracy of the algorithm may be reevaluated at periodic intervals, and a reoptimization of the influence function (and recalculation of its derivatives) undertaken when this accuracy falls below a user-specified threshold. If desired, the optimal influence function (and its derivatives, whenever required) may be read from file in the first step of a simulation, and may be written to file in the last step of a simulation.

The 3D-FFTs are performed using the FFTW libraries.

The use of the PPPM method is restricted to systems under periodic boundary conditions and in three dimensions only. The implementation for truncated-octahedral boxes results from the coordinate transformation described in Sec. 4.4.2.3, so that only the general triclinic case will be discussed in detail. The special case of a rectangular box results in minor simplifications at the level of the calculation of the influence function, which will be mentioned in this context.

7.4.4.1. *Discretization of the computational box.* A general triclinic box may be discretized by means of a grid G, defined by the number of subdivisions N_a, N_b and N_c along the $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ box-edge vectors. The three numbers must be even. It will be convenient to introduce the diagonal matrix $\underline{\mathbf{N}}$ with elements N_a, N_b and N_c, *i.e.*

$$\underline{\mathbf{N}} = \begin{pmatrix} N_a & 0 & 0 \\ 0 & N_b & 0 \\ 0 & 0 & N_c \end{pmatrix} . \tag{7.59}$$

The matrices $\underline{\mathbf{H}}'$ and $\underline{\mathbf{H}}$ are then defined as

$$\underline{\mathbf{H}}' = \begin{pmatrix} N_a^{-1}a'_x & N_b^{-1}b'_x & N_c^{-1}c'_x \\ N_a^{-1}a'_y & N_b^{-1}b'_y & N_c^{-1}c'_y \\ N_a^{-1}a'_z & N_b^{-1}b'_z & N_c^{-1}c'_z \end{pmatrix} = \underline{\mathbf{L}}'\,\underline{\mathbf{N}}^{-1} = \underline{\mathbf{S}}\,\underline{\mathbf{B}}\,\underline{\mathbf{N}}^{-1} , \tag{7.60}$$

and

$$\underline{\mathbf{H}} = \begin{pmatrix} N_a^{-1}a_x & N_b^{-1}b_x & N_c^{-1}c_x \\ N_a^{-1}a_y & N_b^{-1}b_y & N_c^{-1}c_y \\ N_a^{-1}a_z & N_b^{-1}b_z & N_c^{-1}c_z \end{pmatrix} = \underline{\mathbf{L}}\,\underline{\mathbf{N}}^{-1} = \underline{\mathbf{R}}\,\underline{\mathbf{H}}' = \underline{\mathbf{T}}\,\underline{\mathbf{B}}\,\underline{\mathbf{N}}^{-1} . \tag{7.61}$$

The volume of a grid cell is noted $V_G = \mid \underline{\mathbf{H}}' \mid = \mid \underline{\mathbf{H}} \mid$.

Each point of the real-space grid G is associated with an index $\mathbf{n} = (n_a, n_b, n_c)$, with $n_a \in [0; N_a - 1]$, $n_b \in [0; N_b - 1]$ and $n_c \in [0; N_c - 1]$. Points outside this range are periodic copies of points within the range. The corresponding real-space vector may be written in the different representations (Sec. 4.4.1.1)

$$\check{\mathbf{r}}_\mathbf{n} = \underline{\mathbf{N}}^{-1}\mathbf{n} \; , \breve{\mathbf{r}}_\mathbf{n} = \underline{\mathbf{B}}\,\underline{\mathbf{N}}^{-1}\mathbf{n} \; , \mathbf{r}'_\mathbf{n} = \underline{\mathbf{H}}'\mathbf{n} \; , \mathbf{r}_\mathbf{n} = \underline{\mathbf{H}}\mathbf{n} . \tag{7.62}$$

Similarly, each point of the reciprocal-space grid G is associated with an index $\mathbf{l} = (l_a, l_b, l_c)$, with $l_a \in [-N_a/2 + 1; N_a/2]$, $l_b \in [-N_b/2 + 1; N_b/2]$ and $l_c \in [-N_c/2 + 1; N_c/2]$. The corresponding reciprocal-space vector may be written in the different representations (Sec. 4.4.1.4)

$$\check{\mathbf{k}}_\mathbf{l} = 2\pi\mathbf{l} \; , \breve{\mathbf{k}}_\mathbf{l} = 2\pi\underline{\mathbf{B}}^{-1}\mathbf{l} \; , \mathbf{k}'_\mathbf{l} = 2\pi({}^t\underline{\mathbf{L}}')^{-1} \; , \mathbf{k}_\mathbf{l} = 2\pi\,{}^t\underline{\mathbf{L}}^{-1}\mathbf{l} . \tag{7.63}$$

All gridded functions, *i.e.* real- or reciprocal-space functions that only take a value at a grid point of G, will be indicated with a g subscript.

The forward 3D-FFT operation converts a gridded function $f_g(\mathbf{r}_\mathbf{n})$ on the grid G into its finite Fourier coefficients $\hat{f}_g(\mathbf{k}_\mathbf{l})$ on the same grid

$$\hat{f}_g(\mathbf{k}_\mathbf{l}) = V_G \sum_{\mathbf{n}\in G} f_g(\mathbf{r}_\mathbf{n}) \; e^{-i\mathbf{k}_\mathbf{l}\cdot\mathbf{r}_\mathbf{n}} \; , \tag{7.64}$$

where $\mathbf{k_l} = 2\pi {}^t\underline{\mathbf{L}}^{-1}\mathbf{l}$ with $\mathbf{l} \in G$. The backward 3D-FFT performs the reverse operation, namely

$$f_g(\mathbf{r_n}) = V^{-1} \sum_{\mathbf{l}\in G} \hat{f}_g(\mathbf{k_l})\, e^{i\mathbf{k_l}\cdot\mathbf{r_n}} \ . \tag{7.65}$$

7.4.4.2. PPPM *potential.* The PPPM algorithm starts by distributing the N_q atomic partial charges q_i at locations $\mathbf{r}_i$ within the computational box onto the neighbouring grid points (taking periodicity into account), so as to generate the charge-density grid s_g. This assignment is performed as

$$s_g(\mathbf{r_n}) = \sum_{i=1}^{N_q} q_i\, \sigma_g(\mathbf{r_n}; \mathbf{r}_i) \tag{7.66}$$

with

$$\sigma_g(\mathbf{r_n}; \mathbf{r}) = P(\mathbf{r_n} - \mathbf{r}) \ , \tag{7.67}$$

where P is a so-called assignment function discussed in Sec. 7.4.4.4. Note that s_g is a real quantity. The charge density grid is converted to its (complex) reciprocal-space representation $\hat{s}_g(\mathbf{k_l})$ by applying a forward 3D-FFT to $s_g(\mathbf{r_n})$.

The reciprocal-space potential, *i.e.* the potential generated by the corresponding gridded (γ-b)-charges (Sec. 7.4) is then computed in reciprocal space as

$$\hat{\Phi}_{\gamma,g}(\mathbf{k_l}) = (\epsilon_0 \epsilon_{ls})^{-1}\, \hat{G}_g^{\dagger}(\mathbf{k_l})\, \hat{s}_g(\mathbf{k_l}) \ , \tag{7.68}$$

where $\hat{G}_g^{\dagger}$ represents the Fourier coefficients of the influence function. If all charges were located exactly at grid points or if the grid spacing was infinitesimal, the quantity $\hat{G}_g$ would be given by $k_{\mathbf{l}}^{-2}\, \hat{\gamma}(ak_{\mathbf{l}})$. However, since this is generally not the case, a significant gain in accuracy is reached by optimizing $\hat{G}_g^{\dagger}$ to compensate for errors linked with the discretization procedure, taking into account possible variations in the shape and size of the computational box. To this purpose, the influence function $\hat{G}_g^{\dagger}$ is defined as

$$\hat{G}_g^{\dagger}(\mathbf{k_l}) = \hat{G}_g^{o}(\mathbf{k_l}) - \mathrm{Tr}[\widehat{\underline{\mathbf{\Gamma}}}_g^{o}(\mathbf{k_l})({}^t\underline{\mathbf{L}}^{o})^{-1}({}^t\underline{\mathbf{L}} - {}^t\underline{\mathbf{L}}^{o})] \tag{7.69}$$

where $\hat{G}_g^{o}$ is the influence function optimized for a given set $\underline{\mathbf{L}}^{o}$ of box parameters and $\widehat{\underline{\mathbf{\Gamma}}}_g^{o}$ contains the corresponding first-order derivative information in the form

$$\widehat{\underline{\mathbf{\Gamma}}}_g^{o}(\mathbf{k_l}) = -\frac{\partial \hat{G}_g^{o}(\mathbf{k_l})}{\partial \underline{\mathbf{L}}^{o}}\, {}^t\underline{\mathbf{L}}^{o} \ . \tag{7.70}$$

The calculation of the quantities $\hat{G}_g^{o}$ and $\widehat{\underline{\mathbf{\Gamma}}}_g^{o}$ is described below. The optimal influence function $\hat{G}_g^{o}$ is only optimal for a specific set $\underline{\mathbf{L}}^{o}$ of box parameters $\underline{\mathbf{L}}$. When the shape and size of the computational box may vary, the second term in Eq. 7.69 includes a first-order correction to the influence function optimized at $\underline{\mathbf{L}}^{o}$, based on the derivative information calculated simultaneously. In practice, because $\widehat{\underline{\mathbf{\Gamma}}}_g^{o}$ is symmetric, the storage of the two quantities requires seven grids of reals. This evaluation (but not the storage requirement) is omitted for simulations not involving the calculation of the virial or the variation of the box parameters.

7.4.4.3. PPPM *energy and forces.* The reciprocal-space contribution E_γ to the total electrostatic energy (Eq. 7.26) is given by

$$E_\gamma = (2\epsilon_0 \epsilon_{ls}\, \mathcal{V})^{-1} \sum_{i=1}^{N_q} \sum_{j=1}^{N_q} \sum_{\mathbf{l}\in G\,,\, l\neq 0} q_i\, \hat{\sigma}_g(\mathbf{k_l}; \mathbf{r}_i)\, q_j\, \hat{\sigma}_g^{*}(\mathbf{k_l}; \mathbf{r}_j)\, \hat{G}_g^{\dagger}(\mathbf{k_l}) \ . \tag{7.71}$$

For computational efficiency, this pairwise sum is evaluated as a single sum, through

$$E_\gamma = (2\epsilon_0 \epsilon_{ls}\, \mathcal{V})^{-1} \sum_{\mathbf{l}\in G\,,\, l\neq 0} \hat{G}_g^{\dagger}(\mathbf{k_l}) \mid \hat{s}_g(\mathbf{k_l}) \mid^2 \ . \tag{7.72}$$

The (approximate) forces associated with the energy contribution E_γ (Eq. 7.72) are obtained through the evaluation of the gridded field $\mathbf{E}_g$ and its interpolation at the location of the charges. The reciprocal-space force on atom i is then written

$$\mathbf{F}_{\gamma,i} = q_i\, \mathbf{E}(\mathbf{r}_i) \tag{7.73}$$

with

$$\mathbf{E}(\mathbf{r}) = V_G \sum_{\mathbf{n}\in G} P(\mathbf{r} - \mathbf{r_n})\, \mathbf{E}_g(\mathbf{r_n}) \ . \tag{7.74}$$

The same assignment function P should be used here and for the charge assignment (Eq. 7.67), to ensure conservation of the total linear momentum during the dynamics. The gridded field $\mathbf{E}_g$ to be used in Eq. 7.74 can be obtained in either of two ways.

The first method (finite-difference) relies on performing a backward 3D-FFT of the potential to obtain the (real) real-space potential, and using a finite-difference approximation to compute the gridded field as

$$\mathbf{E}_g(\mathbf{r_n}) = -\sum_{\mathbf{n}'\in G} i\,V_G\,\mathbf{D}_g(\mathbf{r_n}-\mathbf{r_{n'}})\,\Phi_{\gamma,g}(\mathbf{r_{n'}}) \tag{7.75}$$

where $i\,V_G\,\mathbf{D}_g$ is is a so-called finite-difference operator, discussed below.

The second method (ik-differentiation) relies on computing the exact gridded field in reciprocal-space

$$\widehat{\mathbf{E}}_g(\mathbf{k_l}) = -i\mathbf{k_l}\,\hat{\Phi}_{\gamma,g}(\mathbf{k_l})\ , \tag{7.76}$$

One then performs three backward 3D-FFTs (one for each Cartesian component) to obtain the corresponding (real) quantity $\mathbf{E}_g$ in real-space. Note that the choice of $\mathbf{k_l}$ in Eq. 7.76 leaves room for some ambiguity. This is because the gridded potential can be mapped to a continuous function (of which the gradient is to be taken) in an infinite number of ways. Accordingly, $\mathbf{k_l}$ in Eq. 7.76 can be any alias vector $\mathbf{k_{l+m\underline{N}}}$ of $\mathbf{k_l}$ with $\mathbf{m}\in\mathbb{Z}^3$. The most realistic mapping of the potential is the one avoiding the introduction of spurious oscillations, which corresponds to the $\mathbf{k_l}$ vectors with the smallest norms. The convention adopted here is to select the alias vectors with $\mathbf{l}$ components are in the ranges $[-N_x/2+1; N_x/2]$, $[-N_y/2+1; N_y/2]$, and $[-N_z/2+1; N_z/2]$.

In principle, the use of $ik-$differentiation requires the storage of three complex grids. In the program, this storage is reduced to two by recycling the charge density / potential grid for the field component along the z-axis.

The virial associated with the energy contribution E_γ (Eq. 7.72) is given in Eqs. 17.40 and 17.41.

7.4.4.4. *Assignment function.* The assignment function P of order p (Eqs. 7.67 and 7.74) performs the distribution (interpolation) of a continuous function at an arbitrary location onto (from) values at the neighbouring p^3 grid points. This function is defined as[28]

$$P(\mathbf{r}) \;=\; V_G^{-1}\sum_{\mathbf{n}\in\mathbb{Z}^3}\tilde{P}(\mathbf{r}+\underline{\mathbf{L}}\mathbf{n}) \tag{7.77}$$

with

$$\tilde{P}(\mathbf{r}) = w_p([\underline{\mathbf{H}}^{-1}\mathbf{r}]_a)w_p([\underline{\mathbf{H}}^{-1}\mathbf{r}]_b)w_p([\underline{\mathbf{H}}^{-1}\mathbf{r}]_c)\ . \tag{7.78}$$

Here, $w_p(\xi)$ is a normalized one-dimensional function vanishing for $\mid\xi\mid\geq p/2$. These functions are listed in Tab. 7.7. The assignment scheme is formulated in terms of oblique coordinates. Thus, in the general case, the distribution (interpolation) of the function from (at) an arbitrary location onto (from) the neighbouring grid points is not necessarily correlated with the Cartesian distance between the points (this is only the case for a rectangular computational box). Note that P is a real quantity.

p	$w_p(\xi)$	range
1	1	$\mid\xi\mid< 1/2$
2	$1-\mid\xi\mid$	$\mid\xi\mid< 1$
3	$3/4-\xi^2$	$\mid\xi\mid< 1/2$
	$(1/8)(2\mid\xi\mid-3)^2$	$1/2<\mid\xi\mid< 3/2$
4	$(1/6)(3\mid\xi\mid^3-6\xi^2+4)$	$\mid\xi\mid< 1$
	$-(1/6)(\mid\xi\mid-2)^3$	$1<\mid\xi\mid< 2$
5	$(1/192)(48\xi^4-120\xi^2+115)$	$\mid\xi\mid< 1/2$
	$-(1/96)(16\xi^4-80\mid\xi\mid^3+120\xi^2-20\mid\xi\mid-55)$	$1/2<\mid\xi\mid< 3/2$
	$(1/384)(2\mid\xi\mid-5)^4$	$3/2<\mid\xi\mid< 5/2$

TABLE 7.7. One-dimensional functions $w_p(\xi)$ used to define the charge-assignment functions (Eq. 7.78). The functions $w_p(\xi)$ vanish for $\mid\xi\mid\geq p/2$.

In practice, the assignment is performed as follows. For p odd, one finds the grid point $\mathbf{n}_i$ closest to the location of charge q_i. For p even, one finds the grid-cell center closest to the location of charge q_i. In this

case, $\mathbf{n}_i$ has half-integer components. With the definitions $\boldsymbol{\zeta}_i = \mathbf{n}_i - \underline{\mathbf{H}}^{-1}\mathbf{r}_i$ and $\Delta\mathbf{n}_i = \mathbf{n} - \mathbf{n}_i$, the fraction of charge allocated to the periodic copy of the grid point $\mathbf{n}$ inside the computational box is given by

$$P(\mathbf{r_n} - \mathbf{r}_i) = V_G^{-1}\, w_p(\zeta_{i,a} + \Delta n_{i,a}) w_p(\zeta_{i,b} + \Delta n_{i,b}) w_p(\zeta_{i,c} + \Delta n_{i,c}) \ . \tag{7.79}$$

Because, by construction, the components of $\boldsymbol{\zeta}_i$ are smaller than 1/2 in absolute value while $w_p(\xi) = 0$ for $\mid \xi \mid \geq p/2$, $w_p(\zeta + \Delta n)$ vanishes for $\mid \Delta n \mid \geq p/2$. The $w_p(\xi)$ functions reexpressed as $w_p(\zeta + \Delta n)$ are listed in Tab. 7.8.

p	Δn	$w_p(\zeta + \Delta n)$
1	0	1
2	$\pm 1/2$	$-(1/2)(\pm 2\zeta - 1)$
3	0	$-(1/4)(4\zeta^2 - 3)$
	± 1	$(1/8)(\pm 2\zeta - 1)^2$
4	$\pm 1/2$	$(1/48)[32 - 12(\pm 2\zeta + 1)^2 + 3(\pm 2\zeta + 1)^3]$
	$\pm 3/2$	$-(1/48)(\pm 2\zeta - 1)^3$
5	0	$(1/192)(48\zeta^4 - 120\zeta^2 + 115)$
	± 1	$(1/96)[-16\zeta^4 \pm 16\zeta^3 + 24\zeta^2 - (\pm 44\zeta) + 19]$
	± 2	$(1/384)(\pm 2\zeta - 1)^4$

TABLE 7.8. One-dimensional functions $w_p(\xi)$ (Tab. 7.7) represented as $w_p(\zeta + \Delta n)$ with Δn integer (p odd) or half-integer (p even), and $\mid \zeta \mid \leq 1/2$. The functions $w_p(\zeta + \Delta n)$ vanish for $\mid \Delta n \mid \geq p/2$.

The Fourier coefficient $\hat{P}$ of the assignment function P are given by

$$\hat{P}(\mathbf{k}) = \hat{w}_p([^t\underline{\mathbf{H}}\mathbf{k}]_a)\hat{w}_p([^t\underline{\mathbf{H}}\mathbf{k}]_b)\hat{w}_p([^t\underline{\mathbf{H}}\mathbf{k}]_c) \ , \tag{7.80}$$

with ${}^t\underline{\mathbf{H}}\mathbf{k} = 2\pi\underline{\mathbf{N}}^{-1}\mathbf{l}$, where $\hat{w}_p(\kappa)$ is the continuous Fourier transform of $w_p(\xi)$, which evaluate to

$$\hat{w}_p(\kappa) = [2\kappa^{-1}\sin(\kappa/2)]^p(1 - \delta_\kappa) \quad + \quad \delta_\kappa \ . \tag{7.81}$$

Note that $\hat{P}$ is a real quantity, that is non-periodic, and odd (p odd) or even (p even) with respect to a change in the sign of any component of $\mathbf{l}$

7.4.4.5. *Finite- and exact-difference operators.* The finite-difference operator $iV_G\mathbf{D}_g$ of order q (Eq. 7.75) estimates the gradient of a given function at a grid point based on the value of the function at the $6q$ neighbouring grid points along the three box axes. The operator $\mathbf{D}_g$ is defined as[28]

$$\mathbf{D}_g(\mathbf{r_n}) = \sum_{j=1}^{q} c_j\, d_{vg,j}(\mathbf{r_n}) \ . \tag{7.82}$$

In this expression, the centered-difference operator $iV_G d_{vg,j}$ generates the gradient contribution evaluated from the function at the 6 neighbouring points distant from the grid point by $\pm jN_a^{-1}a$, $\pm jN_b^{-1}b$ and $\pm jN_c^{-1}c$ along the the three box axes. The operator $d_{vg,j}$ is given by

$$d_{vg,j}(\mathbf{r_n}) = \underline{\mathbf{R}}\,\underline{\mathbf{S}}\mathbf{e}_{g,j}(\mathbf{r_n}) \ , \tag{7.83}$$

where $\mathbf{e}_{g,j}$ is the corresponding operator in terms of oblique coordinates, *i.e.*

$$e_{g,j,a}(\mathbf{r_n}) = i\, V_G^{-1}\, (2jN_a^{-1}a)^{-1}\, \delta_{n_b}\delta_{n_c} \sum_{m\in\mathbb{Z}} (\delta_{n_a - j + N_a m} - \delta_{n_a + j + N_a m}) \ , \tag{7.84}$$

with similar expressions for the b and c components. Note that $iV_G\mathbf{D}_g$ is a real quantity.

The Fourier coefficients $\widehat{\mathbf{D}}_g$ of the operator $\mathbf{D}_g$ are given by

$$\widehat{\mathbf{D}}_g(\mathbf{k_l}) = \sum_{j=1}^{q} c_j\, \widehat{\mathbf{d}}_{g,j}(\mathbf{k_l}) \ , \tag{7.85}$$

where

$$\widehat{\mathbf{d}}_{g,j}(\mathbf{k_l}) = \underline{\mathbf{R}}\,{}^t\underline{\mathbf{S}}^{-1}\,\hat{\mathbf{e}}_{g,j}(\mathbf{k_l}) \tag{7.86}$$

and

$$\hat{e}_{g,j,a}(\mathbf{k_l}) = (jN_a^{-1}a)^{-1}\sin(j[{}^t\underline{\mathbf{H}}\mathbf{k_l}]_a) \ , \tag{7.87}$$

with similar expressions for the b and c components. Taken together, Eqs. 7.85, 7.86 and 7.87 may be written

$$\widehat{\mathbf{D}}_g(\mathbf{k_l}) = {}^t\underline{\mathbf{H}}^{-1}\sum_{j=1}^{q} c_j\, j^{-1} \begin{pmatrix} \sin(j[{}^t\underline{\mathbf{H}}\mathbf{k_l}]_a) \\ \sin(j[{}^t\underline{\mathbf{H}}\mathbf{k_l}]_b) \\ \sin(j[{}^t\underline{\mathbf{H}}\mathbf{k_l}]_c) \end{pmatrix} \tag{7.88}$$

with ${}^t\underline{\mathbf{H}}\mathbf{k_l} = 2\pi\underline{\mathbf{N}}^{-1}\mathbf{l}$. Note that $\widehat{\mathbf{D}}_g$ is a real quantity, periodic in the components of $\mathbf{l}$ (with periods N_a, N_b and N_c), and characterized by the same symmetry properties as $\mathbf{k_l}$ with respect to $\mathbf{l}$ (central antisymmetry in the general case, antisymmetry with respect to changing the sign of any component for a rectangular box). In addition, any component of ${}^t\underline{\mathbf{H}}\widehat{\mathbf{D}}_g(\mathbf{k_l})$ vanishes when the corresponding l component is a half-integer multiple of the corresponding number of grid subdivisions. This implies in particular that $\widehat{\mathbf{D}}_g(\mathbf{k_l}) = \mathbf{0}$ for the vectors $\mathbf{l}$ with components equal to $n_aN_a/2$, n_bN_b and n_cN_c with $\mathbf{n} \in \mathbb{Z}^3$

In reciprocal space, Eq. 7.75 can be written

$$\widehat{\mathbf{E}}_g(\mathbf{k_l}) = -i\widehat{\mathbf{D}}_g(\mathbf{k_l})\,\hat{\Phi}_{\gamma,g}(\mathbf{k_l}) \ . \tag{7.89}$$

Comparing this expression with Eq. 7.76 shows that the exact difference operator corresponds to

$$\widehat{\mathbf{D}}_g(\mathbf{k_l}) = \mathbf{k_l} \ . \tag{7.90}$$

In this specific case, $\widehat{\mathbf{D}}_g$ retains the same symmetry properties as the finite-difference operators with respect to the components of $\mathbf{l}$, but is no longer periodic.

For a given finite-difference operator of order q, the coefficients c_j in Eq. 7.82 may be selected so as to minimize the difference between the corresponding finite-difference operator and the exact-difference operator[28,34]. In reciprocal space, this leads to the requirement

$${}^t\underline{\mathbf{H}}^{-1}\sum_{j=1}^{q} c_j\, j^{-1} \begin{pmatrix} \sin(2\pi jN_a^{-1}l_a) \\ \sin(2\pi jN_b^{-1}l_b) \\ \sin(2\pi jN_c^{-1}l_c) \end{pmatrix} \approx \mathbf{k_l} \tag{7.91}$$

for any $\mathbf{l}$ or, equivalently,

$$\sum_{j=1}^{q} c_j\, j^{-1}\sin(2\pi jx) \approx 2\pi x \tag{7.92}$$

for any x. Differentiating with respect to x and dividing by 2π gives

$$\sum_{j=1}^{q} c_j\cos(2\pi jx) \approx 1 \ . \tag{7.93}$$

Expanding the left-hand side in Taylor series around $x = 0$ up to order $2q - 2$, equating the coefficients of power m to δ_m, and solving the resulting system of equations results in the optimal coefficients listed in Tab. 7.9.

q	c_1	c_2	c_3	c_4	c_5
1	1				
2	4/3	-1/3			
3	3/2	-3/5	1/10		
4	8/5	-4/5	8/35	-1/35	
5	5/3	-20/21	5/14	-5/63	1/12

TABLE 7.9. Optimal weighting coefficients c_j ($j \leq q$) for the finite-difference operator (Eq. 7.82).

7.4.4.6. *Optimal influence function and derivatives.* For a given set of box parameters characterized by the matrix $\underline{\mathbf{L}}^o$, the influence function that optimally compensates for discretization errors can be computed as

$$\hat{G}_g^o(\mathbf{k_l}) = \frac{\widehat{\mathbf{D}}_g(\mathbf{k_l}) \cdot [\sum_{\mathbf{m}\in\mathbb{Z}^3} \mathbf{k_{l,m}} k_{\mathbf{l,m}}^{-2} \hat{\gamma}(a k_{\mathbf{l,m}}) \hat{P}^2(\mathbf{k_{l,m}})]}{\widehat{\mathbf{D}}_g^2(\mathbf{k_l}) [\sum_{\mathbf{m}\in\mathbb{Z}^3} \hat{P}^2(\mathbf{k_{l,m}})]^2} \ , \tag{7.94}$$

where

$$\mathbf{k_{l,m}} = \mathbf{k_{l+\underline{N}\,m}} = \mathbf{k_l} + 2\pi^t \underline{\mathbf{H}}^{-1} \mathbf{m} \tag{7.95}$$

with $\mathbf{m} \in \mathbb{Z}^3$ is an alias vector of $\mathbf{k}$. This is valid for $\mathbf{l} \in G$ and $\mathbf{l} \neq \mathbf{0}$, together with $\hat{G}_g(\mathbf{0}) = 0$. In practice, the summation over alias vectors is restricted to $\mathbf{m}$ vectors with integer components in the range $[-m_{max}...m_{max}]$. A value of 2 or 3 for m_{max} is usually sufficient to reach convergence. The quantity $\hat{P}$ is given in by Eq. 7.80, the quantity $\widehat{\mathbf{D}}_g$ by Eqs. 7.88 (finite-difference) or 7.90 (ik-differentiation), and the quantity $\hat{\gamma}$ by Eq. 7.20. Note that $\hat{G}_g^o$ is a real quantity.

Due to the summation over alias vectors, the second terms in both the numerator and denominator of Eq. 7.94 are periodic in the components of $\mathbf{l}$ (with periods N_a, N_b and N_c, respectively). The second term in the numerator is also characterized by the same symmetry properties as $\mathbf{k_l}$ with respect to these components (central antisymmetry in the general case, antisymmetry with respect to changing the sign of any component for a rectangular box). Due to the properties of $\widehat{\mathbf{D}}_g$ (Sec. 7.4.4.5) one observes that : (i) $\hat{G}_g^o$ is periodic when using a finite-difference operator (but not when using ik-differentiation) ; (ii) in the general triclinic case, $\hat{G}_g^o$ is even with respect to a change in the sign of $\mathbf{l}$; (iii) in the rectangular case, $\hat{G}_g^o$ is even with respect to a change in the sign of any component of $\mathbf{l}$. Thus, for a triclinic box, the calculation of the influence function may be simplified by evaluating it over a half-space ($l_c \in [0; N_c/2]$) and symmetrizing the function afterwards. When using a finite-difference operator and a triclinic box, some care must be taken in regard to the boundary points. For example, if the points $(N_a/2, l_b, l_c)$ and $(-N_a/2, -l_b, -l_c)$ share the same value of $\hat{G}_g^o$, the points $(-N_a/2, -l_b, -l_c)$ and $(N_a/2, -l_b, -l_c)$ do not. Therefore, symmetry cannot be applied to the boundary point, which still must be calculated only over the entire space to enforce the convention for the definition of $\mathbf{k_l}$ in Eq. 7.76. This problem does not arise when using a finite-difference operator ($\hat{D}_g$ is then periodic), or using a rectangular computational box (where the points $(-N_a/2, -l_b, -l_c)$ and $(N_a/2, -l_b, -l_c)$ share the same value of $\hat{G}_g^o$ due to symmetry). For a rectangular box, the calculation may be further limited to an single octant ($l_a \in [0; N_a/2]$, $l_b \in [0; N_b/2]$ and $l_c \in [0; N_c/2]$) and symmetrizing the function afterwards. The quantity $k_{\mathbf{l,m}}$ can never be zero (and is thus not invertible) because $\hat{G}_g^o$ is not needed for $l = 0$ (Eq. 7.72). However, when using a finite-difference operator, the numerator and denominator vanish for the vectors $\mathbf{l}$ with components equal to $n_a N_a/2$, $n_b N_b$ and $n_c N_c$ with $\mathbf{n} \in \mathbb{Z}^3$, in which case $\hat{G}_g^o$ should be separately set to zero.

In practice, the sum in the denominator of Eq. 7.94 is evaluated analytically as

$$\sum_{\mathbf{m}\in\mathbb{Z}^3} \hat{P}^2(\mathbf{k_{l,m}}) = \prod_{\mu=x,y,z} \omega_p(H_{\mu k_\mu}) \tag{7.96}$$

with

$$\omega_p(\kappa) = -[\sin(\kappa/2)]^{2p} \frac{1}{(2m-1)!} \frac{d^{2p-1}}{dx^{2p-1}} \cot\theta \mid_{\theta=\kappa/2} \ . \tag{7.97}$$

The values of the $\omega_p(\kappa)$ functions are reported in Tab. 7.10

p	$\omega_p(\kappa)$
1	1
2	$1 - (2/3)s^2$
3	$1 - s^2 + (2/15)s^4$
4	$1 - (4/3)s^2 + (2/5)s^4 - (4/315)s^6$
5	$1 - (5/3)s^2 + (7/9)s^4 - (17/189)s^6 + (2/2835)s^8$

TABLE 7.10. Functions used in the analytical evaluation of the denominator in Eq. 7.94 (Eq. 7.97), with $s = \sin(\kappa/2)$.

The negative derivative of the optimal influence function with respect to the box parameters $\underline{\mathbf{L}}^o$, amplified on the right by ${}^t\underline{\mathbf{L}}^o$ (Eq. 7.70) may be calculated simultaneously with the influence function using Eq. 4.46 and noting that $\hat{P}$ is independent of $\underline{\mathbf{L}}^o$

$$\begin{aligned}
\widehat{\underline{\mathbf{\Gamma}}}^o_g(\mathbf{k_l}) \;=\; & \frac{1}{\widehat{\mathbf{D}}^2_g(\mathbf{k_l})\,[\sum_{\mathbf{m}\in\mathbb{Z}^3}\hat{P}^2(\mathbf{k_{l,m}})]^2}\sum_{\mathbf{m}\in\mathbb{Z}^3}\frac{\hat{P}^2(\mathbf{k_{l,m}})}{k^2_{\mathbf{l,m}}}\Big\{ \\
& \big\{k_{\mathbf{l,m}}\otimes\widehat{\mathbf{D}}_g(\mathbf{k_l})+\widehat{\mathbf{D}}_g(\mathbf{k_l})\otimes k_{\mathbf{l,m}} \quad -\frac{2[\mathbf{k_{l,m}}\cdot\widehat{\mathbf{D}}_g(\mathbf{k_l})]\mathbf{k_{l,m}}\otimes\mathbf{k_{l,m}}}{k^2_{\mathbf{l,m}}} \\
& -\frac{2[\mathbf{k_{l,m}}\cdot\widehat{\mathbf{D}}_g(\mathbf{k_l})]\widehat{\mathbf{D}}_g(\mathbf{k_l})\otimes\widehat{\mathbf{D}}_g(\mathbf{k_l})}{\widehat{\mathbf{D}}^2_g(\mathbf{k_l})}\big\}\hat{\gamma}(ak_{\mathbf{m}}) \\
& +\frac{[\mathbf{k_{l,m}}\cdot\widehat{\mathbf{D}}_g(\mathbf{k_l})]k_{\mathbf{l,m}}\otimes k_{\mathbf{l,m}}}{k^2_{\mathbf{l,m}}}\;ak_{\mathbf{l,m}}\,\hat{\gamma}'(ak_{\mathbf{l,m}})\Big\}
\end{aligned} \tag{7.98}$$

with $\hat{\gamma}'(\kappa)=d\hat{\gamma}(\kappa)/d\kappa$ (Tab. 7.5), valid for $\mathbf{l}\in G$ and $\mathbf{l}\neq\mathbf{0}$, together with $\widehat{\underline{\mathbf{\Gamma}}}^o_g(\mathbf{0})=\underline{\mathbf{0}}$. Note that $\widehat{\underline{\mathbf{\Gamma}}}^o_g$ is a real quantities. One further observes that : (i) $\widehat{\underline{\mathbf{\Gamma}}}^o_g$ is periodic when using a finite-difference operator (but not when using ik-differentiation) ; (ii) in the general triclinic case, $\widehat{\underline{\mathbf{\Gamma}}}^o_g$ is even with respect to a change in the sign of $\mathbf{l}$; (iii) in the rectangular case, the components of $\widehat{\underline{\mathbf{\Gamma}}}^o_g$ are odd with respect to a change in the sign of any component of $\mathbf{l}$ involved in their definition. Thus, as for the influence function, the calculation of $\widehat{\underline{\mathbf{\Gamma}}}^o_g$ can be restricted to a half-space (triclinic box) or to an octant (rectangular box), followed by symmetrization.

7.4.4.7. *Accuracy estimate.* When the charge-shaping function and the real-space cutoff are chosen so that Eq. 7.43 is satisfied, which implies that the real-space force evaluation is exact, and under the assumptions that (i) the force error is a sum of pairwise contributions and (ii) the system is homogeneous and disordered, the PPPM root-mean-square force error ΔF can be estimated as[35],

$$\Delta F=\big[\,\frac{1}{N_q}\sum_{i=1}^{N_q}(\mathbf{F}_i-\mathbf{F}_i^{exact})^2\,\big]^{1/2}\approx\frac{1}{4\pi\epsilon_0}\tilde{S}^2\,(\frac{Q}{N_qV})^{1/2}\;, \tag{7.99}$$

where $\mathbf{F}_i^{exact}$ is the the reference (exact) force, with

$$\begin{aligned}
Q \;=\; & 16\pi^2V^{-1}\sum_{\mathbf{l}\in G\,,\,\mathbf{l}\neq\mathbf{0}}\Big\{\sum_{\mathbf{m}\in\mathbb{Z}^3}k^{-2}_{\mathbf{l,m}}\,\hat{\gamma}^2(ak_{\mathbf{l,m}}) \\
& +[\hat{G}^\dagger_g]^2(\mathbf{k_l})\,\widehat{\mathbf{D}}^2_g(\mathbf{k_l})\,[\sum_{\mathbf{m}\in\mathbb{Z}^3}\hat{P}^2(\mathbf{k_{l,m}})]^2 \\
& -2\,\hat{G}^\dagger_g(\mathbf{k_l})\,\widehat{\mathbf{D}}_g(\mathbf{k_l})\cdot\sum_{\mathbf{m}\in\mathbb{Z}^3}\mathbf{k_{l,m}}\,k^{-2}_{\mathbf{l,m}}\,\hat{\gamma}(ak_{\mathbf{l,m}})\,\hat{P}^2(\mathbf{k_{l,m}})\Big\}\;.
\end{aligned} \tag{7.100}$$

Note that Eq. 7.100 is valid for any influence function (not necessary the optimal one defined in Eq. 7.94) so that it can be used to estimate accuracy losses upon using the first-order corrected influence function of Eq. 7.69 upon variations of the box dimensions. As for the influence function and its derivatives (Sec. 7.4.4.6) the calculation can be restricted to a half-space (triclinic box) or to an octant (rectangular box), and amplified by a symmetry weighting factor. Note that the vectors $\mathbf{l}$ with components equal to $n_aN_a/2$, n_bN_b and n_cN_c with $\mathbf{n}\in\mathbb{Z}^3$, for which $\hat{G}^o_g$ was set to zero (Sec. 7.4.4.6) must be included here.

This estimate is generally an upper bound to the error for realistic molecular systems. It is excellent for random distribution of charges, but generally provides an upper bound in realistic molecular systems with charge compensation (dipolar molecules, configurations favouring charge-charge, charge-dipole and dipole-dipole cancellations).

In the special case where $\hat{G}^\dagger_g=\hat{G}^o_g$, inserting Eq. 7.94 into Eq. 7.100 gives

$$\begin{aligned}
Q=16\pi^2V^{-1}\sum_{\mathbf{l}\in G\,,\,l\neq0}\Big\{\sum_{\mathbf{m}\in\mathbb{Z}^3}k^{-2}_{\mathbf{l,m}}\,\hat{\gamma}^2(k_{\mathbf{l,m}}) \\
-\frac{\big[\widehat{\mathbf{D}}_g(\mathbf{k_l})\;\cdot\;[\sum_{\mathbf{m}\in\mathbb{Z}^3}\mathbf{k_{l,m}}k^{-2}_{\mathbf{l,m}}\hat{\gamma}(k_{\mathbf{l,m}})\hat{P}^2(\mathbf{k_{l,m}})]\big]^2}{\widehat{\mathbf{D}}^2_g(\mathbf{k_l})\,[\sum_{\mathbf{m}\in\mathbb{Z}^3}\hat{P}^2(\mathbf{k_{l,m}})]^2}\Big\}\;.
\end{aligned} \tag{7.101}$$

An accuracy reevaluation through Eqs. 7.100 occurs every N_{ev} steps in MD simulations (including step zero if the influence function is read from file). If the estimated r.m.s. force error is larger than the threshold ΔF_{ev}, the influence function is reoptimized (together with its derivatives if required). In addition, any calculation

of the influence function (in the first step unless the function was read from file, or subsequent during reoptimizations after accuracy reevaluation) is accompanied by an accuracy evaluation through Eq. 7.101. In this case, if the estimated r.m.s. force error is larger than ΔF_{ev}, the program stops with an error message.

7.4.4.8. *Calculation of the $\tilde{A}_2$ self term in* PPPM. In the PPPM case, the quantity $\tilde{A}_2$ is evaluated as follows. In a first step, one constructs the grid $R_g(\mathbf{r_n})$ using the assignment-like equation

$$R_g(\mathbf{r_n}) = V_G^{-1} \sum_{i=1}^{N_q} q_i^2 \sum_{\mathbf{m}\in\mathbb{Z}^3} \prod_{\mu=x,y,z} f_p(L_\mu m_\mu + n_\mu; s_{i,\mu}) \tag{7.102}$$

where

$$f_p(n;s) = \sum_{n'=-(p-1)/2}^{(p-1)/2} w_p(s+n'-n) w_p(s+n'), \tag{7.103}$$

the n'-sum running over integers (p odd) or half integers (p even). For each charge, $\mathbf{s}_i$ is defined as follows. For p odd, one finds the grid point $\mathbf{n}_i$ closest to the location of charge q_i. For p even, one finds the grid-cell center closest to the location of charge q_i. In this case, $\mathbf{n}_i$ has half-integer components. Introducing the definition $\mathbf{s}_i = \mathbf{n}_i - \underline{\mathbf{H}}^{-1}\mathbf{r}_i$, one easily sees that the components of $\mathbf{s}_i$ satisfy the condition $\mid s_{i,\mu} \mid \leq 1/2$. The functions w_p in Eq. 7.103 are those used to construct the assignment function of order p (Tab. 7.8). Because $w_p(s+n)$ with $\mid s \mid \leq 1/2$ vanishes for $\mid n \mid \geq p/2$, $f_p(n;s)$ vanishes for $\mid n \mid \geq p$. Thus, the assignment of Eq. 7.102 distributes the square charge onto the $(2p-1)^3$ grid points neighbouring the origin of the grid (taking periodicity into account). The functions f_p are listed in Tab. 7.11. In a second step, one performs a fast Fourier transform of $R_g(\mathbf{r_n})$ to obtain $\hat{R}_g(\mathbf{k_l})$. The time required for this FFT is included in the timing information for the A2 calculation, not for the PPPM FFT. Note also that due to the symmetry of $f_p(n;s)$, a cosine transform would be sufficient here.

In a third step, one computes A_2 as

$$\tilde{A}_2 = \frac{4\pi}{V\tilde{S}^2} \sum_{\mathbf{l}\in G} \hat{G}_g^*(\mathbf{k_l})\, \hat{R}_g(\mathbf{k_l}) \ , \tag{7.104}$$

where $\hat{G}_g^*(\mathbf{k_l})$ is the first-order corrected influence function. Due to the properties of $\mathbf{s}_i$, the reciprocal-space self-energy of a charge is a function of its relative position to the nearest grid point (or grid-cell center). For large enough systems, these relative positions will be randomly distributed, and an estimate of the average self-energy can be obtained by replacing $f_p(n;s)$ by

$$\overline{f}_p(n) = \int_{-1/2}^{1/2} ds\, f_p(n;s) \tag{7.105}$$

in Eq. 7.102. The functions $\overline{f}_p$ are listed in Tab. 7.11. Using f_p (input switch NA2CLC=3), the self-energy term is configuration-dependent and needs to be recalculated every step of the simulation, except for $p=1$ ($f_1(n;s) = \overline{f}_1$) where the two choices lead to the same (configuration-independent) result. Using $\overline{f}_p$ (input switch NA2CLC=4), this calculation needs only be done once (if the volume is constant). In practice, the quantity $\tilde{A}_2$ is calculated (input switch NA2CLC=3 or 4) either (*i*) once at the beginning of the simulation (constant-volume simulation and NA2CLC=4), (*ii*) whenever an energy output (input switches NTPR and NTWE) is required (constant-volume simulation and NA2CLC=3), or (*iii*) every step (constant-pressure simulation). The value of $\tilde{A}_2$ is assumed constant between updates. These choices, together with the updating mode of A_2 (see above), ensure that the constant-pressure dynamics is always correct, and the energies and virial correct whenever printed into an energy output (they are only approximate in-between).

7.4.4.9. *Reading and writing of the influence function and derivatives.* If desired, the optimal influence function $\hat{G}_g^o$ (and its derivatives $\hat{\Gamma}_g^o$ whenever required), together with the corresponding reference edge lengths $\underline{\mathbf{L}}^o$, can be read from file at the beginning of a simulation. They can also be written to file at the end of a simulation. The file is unformatted and contains : 3 integers N_a, N_b and N_c ; 6 reals for the box parameters $\underline{\mathbf{L}}^o$; 1 logical indicating whether the six derivative grids are in the file ; 1 grid $\hat{G}_g^o$; 6 grids $\hat{\Gamma}_g^o$ in the order *aa*, *ab*, *ac*, *bb*, *bc* and *cc* (if the corresponding logical is set). Upon reading the file, the values of N_a, N_b and N_c are checked against the corresponding current parameters, the value $\underline{\mathbf{L}}^o$ against the current $\underline{\mathbf{L}}$ (within an acceptably small tolerance, $\underline{\mathbf{L}}^o$ is then exactly equalized to $\underline{\mathbf{L}}$), and the logical against the current need for derivatives. Any incompatibility results in an error message.

p	$\mid n \mid$	$f_p(n;s)$	$\overline{f}_p(n)$
1	0	1	1
2	0	$1/2(1+4s^2)$	$2/3$
	1	$1/4(1-4s^2)$	$1/6$
3	0	$1/32(19-24s^2+48s^4)$	$11/20$
	1	$1/16(3+8s^2-16s^4)$	$13/60$
	2	$1/64(1-8s^2+16s^4) = 1/64(1-4s^2)^2$	$1/120$
4	0	$1/576(265+204s^2-528s^4+320s^6)$	$151/315$
	1	$1/2304(575-564s^2+1488s^4-960s^6)$	$397/1680$
	2	$1/1152(23+84s^2-240s^4+192s^6)$	$1/42$
	3	$1/2304(1-12s^2+48s^4-64s^6) = 1/2304(1-4s^2)^3$	$1/5040$
5	0	$1/73728(32227-9520s^2+28960s^4-29440s^6+8960s^8)$	$15619/36288$
	1	$1/18432(4389+1792s^2-5472s^4+5632s^6-1792s^8)$	$44117/181440$
	2	$1/36864(1559-1456s^2+4512s^4-4864s^6+1792s^8)$	$913/22680$
	3	$1/18432(19+128s^2-416s^4+512s^6-256s^8)$	$251/181440$
	4	$1/147456(1-16s^2+96s^4-256s^6+256s^8) = 1/147456(1-4s^2)^4$	$1/362880$

TABLE 7.11. The functions $f_p(n;s)$ and $\overline{f}_p(n)$ required for the evaluation of the exact PPPM self energy term $\tilde{A}_2$ (Eqs. 7.103 and 7.105). The variable s must satisfy $\mid s \mid \leq 1/2$, and both functions vanish for $\mid n \mid \geq p/2$.

7.5. Polarization

7.5.1. Introduction. Polarisability allows for a more accurate description of the non-bonded interactions in classical atomistic simulations and must be implemented in the software when using polarisable force fields[9].

Several methods have been described in the literature that explicitly treat electronic polarization. GROMOS uses the charge on spring (COS) model[36] (also called Drude oscillator[37] or shell model[38]) where an inducible dipole $\boldsymbol{\mu}_i$ is modeled by attaching a massless, virtual site with a point-charge q_i^v to the polarisable center i, via a spring with harmonic force constant $k^{ho}{}_i$,

$$k^{ho}{}_i = \frac{(q_i^v)^2}{\alpha_i}. \tag{7.106}$$

The inducible dipole $\boldsymbol{\mu}_i$ adapts its size and direction according to its polarisability α_i and the electric field $\mathbf{E}'_i$ at the COS of the polarisable center i (assuming isotropic α_i and linear dependence of $\boldsymbol{\mu}_i$ on $\mathbf{E}'_i$) according to

$$\boldsymbol{\mu}_i = \alpha_i \mathbf{E}'_i. \tag{7.107}$$

The charge at the polarisable center with position $\mathbf{r}_i$, which may have a permanent charge q_i, is then $(q_i - q_i^v)$. Thus, the induced dipoles $\boldsymbol{\mu}_i$ are represented by

$$\boldsymbol{\mu}_i = q_i^v(\mathbf{r}'_i - \mathbf{r}_i) \tag{7.108}$$

where $\mathbf{r}'_i$ is the position of the charge-on-spring. There is no electrostatic interaction between the virtual charge q_i^v at $\mathbf{r}'_i$ and the charge $(q_i - q_i^v)$ at $\mathbf{r}_i$. In COS based schemes in which the charges-on-spring are not explicitly treated as additional degrees of freedom, the sum of the forces acting on any charge-on-spring should be zero, and the virtual charge q_i^v must be positioned such that

$$\mathbf{f}'^{ho}_i + \mathbf{f}'^{coul}_i = 0 \tag{7.109}$$

with the force $\mathbf{f}'^{ho}_i$ due to the spring given by

$$\mathbf{f}'^{ho}_i = -k^{ho}{}_i(\mathbf{r}'_i - \mathbf{r}_i) = -\frac{(q_i^v)^2}{\alpha_i}(\mathbf{r}'_i - \mathbf{r}_i) \quad , \tag{7.110}$$

and $\mathbf{f}_i'^{coul}$ due to the (Coulombic) electric field at the charge-on-spring ($\mathbf{E}_i'$) given by

$$\mathbf{f}_i'^{coul} = q_i^v \mathbf{E}_i' \quad . \tag{7.111}$$

To satisfy eq Eq. 7.109, the $\boldsymbol{\mu}_i$, i.e., the $\mathbf{r}_i'$, must be determined from the $\mathbf{E}_i'$s where the field $\mathbf{E}_i'$ does not contain a contribution from the charge $(q_i - q_i^v)$ at $\mathbf{r}_i$. However, since the displacement $|\mathbf{r}_i' - \mathbf{r}_i|$ of the charge-on-spring from the polarisable center is nonzero upon polarisation, an approximation of the ideal inducible dipole $\boldsymbol{\mu}_i$ at site i would require to determine $\mathbf{r}_i'$ from the electric field $\mathbf{E}_i$ at the polarisable center itself[39],

$$\mathbf{r}_i' = \mathbf{r}_i + \frac{\alpha_i \mathbf{E}_i}{q_i^v} \tag{7.112}$$

Using this approximation, the total force acting on the charge-on-spring is only zero if

$$\mathbf{E}_i = \mathbf{E}_i' \tag{7.113}$$

which is usually not the case for the induced dipole due to the nonzero values for $|\mathbf{r}_i' - \mathbf{r}_i|$. By choosing q_i^v large enough, $|\mathbf{r}_i' - \mathbf{r}_i|$ adopts relatively small values, resulting in small differences between $\mathbf{E}_i$ and $\mathbf{E}_i'$. On the other hand, the size of q_i^v is limited to values for which $|\mathbf{r}_i' - \mathbf{r}_i|$ is significant enough with respect to interatomic distances such that numerical precision is ensured when calculating, e.g. interaction energies involving induced dipoles.[36,39] The COS method has been employed in combination with iterative procedures to minimize the energy of the COS with respect to its position $\mathbf{r}_i'$, *i.e.* solving Eq. 7.109. In GROMOS q_i^v is generally set to $-8e$.

7.5.2. Theory. The COS method treats the induced dipole moments via additional point charges only, which allows for an easy introduction of polarisability into schemes to compute long-range electrostatic forces, such as the reaction-field, Ewald-summation, Particle-Particle- Particle-Mesh (PPPM) and Particle-Mesh-Ewald (PME) techniques or into a quantum-mechanical Hamiltonian for the electronic degrees of freedom of a (solute) molecule. The electrostatic potential ϕ_i at the polarisable centers i due to the charges in the system can be expressed using Coulombic terms

$$\phi_i(\mathbf{r}, \mathbf{r}') = \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{j \neq i \\ j \text{ inside cut-off } i \\ (i,j) \text{ not excluded}}}^{N} \left[\frac{(q_j - q_j^v)}{|\mathbf{r}_i - \mathbf{r}_j|} + \frac{q_j^v}{|\mathbf{r}_i - \mathbf{r}_j'|} \right] \tag{7.114}$$

where the positions of the N atoms and corresponding virtual charges denoted by $\boldsymbol{r}^N = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N)$ and $\boldsymbol{r}'^N = (\mathbf{r}_1'\mathbf{r}_2', \ldots, \mathbf{r}_N')$, and ϵ_{cs} is the relative dielectric permittivity used for the model interactions. Because of the dependence of the $\mathbf{r}_i'$ on the $\mathbf{r}_j$s (and $\mathbf{r}_i'$s) via $\mathbf{E}_i$ in eq Eq. 7.112, the relation between ϕ_i and the electric field $\mathbf{E}_i$ is given by

$$\mathbf{E}_i = -\nabla_i \phi_i(\mathbf{r}, \mathbf{r}') = -\left(\frac{\partial \phi_i}{\partial \mathbf{r}_i} + \sum_{k \neq i}^{N} \frac{\partial \phi_i}{\partial \mathbf{r}_k'} \cdot \frac{\partial \mathbf{r}_k'}{\partial \mathbf{r}_i} \right) \tag{7.115}$$

When applying a Born-Oppenheimer-like iterative self-consistent field (SCF) procedure, however, the $\mathbf{r}_i$s are at every iteration step determined in the fixed electric field due to the other q_js and q_j^vs. When using a convergence criterion which minimizes the ϕ_is with respect to the positions $\mathbf{r}_i'$, the second term in eq Eq. 7.115 is zero at convergence because $\partial\phi_i/\partial\mathbf{r}_k' = 0$. Thus

$$\mathbf{E}_i = -\frac{\partial \phi_i}{\partial \mathbf{r}_i} = \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{j \neq i \\ j \text{ inside cut-off } i \\ (i,j) \text{ not excluded}}}^{N} \left[\frac{(q_j - q_j^v)(\mathbf{r}_i - \mathbf{r}_j)}{|\mathbf{r}_i - \mathbf{r}_j|^3} + \frac{q_j^v(\mathbf{r}_i - \mathbf{r}_j')}{|\mathbf{r}_i - \mathbf{r}_j'|^3} \right] \tag{7.116}$$

The electrostatic part $\mathcal{V}^{(ele)}$ of the potential energy can also be expressed in terms of Coulomb interactions. The only non-Coulombic term to be added to $\mathcal{V}^{(ele)}$ is the selfpolarization energy V^{self}, which in the COS model can be expressed in terms involving point charges as well

$$\mathcal{V}^{(ele)} = V^{coul} + V^{self} \tag{7.117}$$

with

$$V^{coul}(\mathbf{r},\mathbf{r}') = \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{i=1}}^{N-1}\sum_{\substack{j>i \\ j \text{ inside cut-off } i \\ (i,j) \text{ not excluded}}}^{N} \left[\frac{(q_i-q_i^v)(q_j-q_j^v)}{|\mathbf{r}_i-\mathbf{r}_j|} + \frac{(q_i-q_i^v)q_j^v}{|\mathbf{r}_i-\mathbf{r}'_j|} + \frac{(q_j-q_j^v)q_i^v}{|\mathbf{r}'_i-\mathbf{r}_j|} + \frac{q_i^v q_j^v}{|\mathbf{r}'_i-\mathbf{r}'_j|}\right] \tag{7.118}$$

and

$$V^{self}(\mathbf{r},\mathbf{r}') = \frac{1}{2}\sum_{i=1}^{N}\frac{(q_i^v)^2}{\alpha_i}|\mathbf{r}'_i-\mathbf{r}_i|^2 \tag{7.119}$$

Next we consider the expression for the forces $\mathbf{f}_i$ that act on (polarisable) atomic centers i

$$\mathbf{f}_i = -\nabla_i V_i^{ele}(\mathbf{r},\mathbf{r}') = -\left(\frac{\partial\mathcal{V}^{(ele)}}{\partial\mathbf{r}_i} + \sum_{k\neq i}^{N}\frac{\partial\mathcal{V}^{(ele)}}{\partial\mathbf{r}'_k}\cdot\frac{\partial\mathbf{r}'_k}{\partial\mathbf{r}_i}\right). \tag{7.120}$$

Note again the dependence of the $\mathbf{r}'_k$s on the $\mathbf{r}_i$s that appears in the second term on the right in eq Eq. 7.120, which might adopt nonzero values because $\mathcal{V}^{(ele)}$ not only contains terms due to the ϕ_is (first two terms on the right in Eq. 7.118) but also due to the ϕ_is (last two terms on the right in Eq. 7.118) and V^{self}, whereas when using eq Eq. 7.112 only the ϕ_is have been minimized with respect to the $\mathbf{r}'_i$s. When nevertheless using assumptions Eqs. 7.109 and 7.113, Eq. 7.120 reduces to

$$\mathbf{f}_i^{red} = -\frac{\partial\mathcal{V}^{(ele)}}{\partial\mathbf{r}_i} = -\left(\frac{\partial V^{coul}}{\partial\mathbf{r}_i} + \frac{\partial V^{self}}{\partial\mathbf{r}_i}\right) \quad . \tag{7.121}$$

From the assumptions in Eqs. 7.109 and 7.113 we have

$$-\frac{\partial V^{self}}{\partial\mathbf{r}_i} = \mathbf{f}_i^{ho} = -\mathbf{f}_i'^{ho} = \mathbf{f}_i'^{coul} = -\frac{\partial V^{coul}}{\partial\mathbf{r}_i} \tag{7.122}$$

and the reduced expression $\mathbf{f}_i^{red}$ for the atomic forces

$$\begin{aligned}
\mathbf{f}_i^{red} &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{i=1}}^{N-1}\sum_{\substack{j>i \\ j \text{ inside cut-off } i \\ (i,j) \text{ not excluded}}}^{N} \left[\frac{(q_i-q_i^v)(q_j-q_j^v)(\mathbf{r}_i-\mathbf{r}_j)}{|\mathbf{r}_i-\mathbf{r}_j|^3}\right. \\
&+ \frac{(q_i-q_i^v)q_j^v(\mathbf{r}_i-\mathbf{r}'_j)}{|\mathbf{r}_i-\mathbf{r}'_j|^3} \\
&+ \frac{(q_j-q_j^v)q_i^v(\mathbf{r}'_i-\mathbf{r}_j)}{|\mathbf{r}'_i-\mathbf{r}_j|^3} \\
&+ \left.\frac{q_i^v q_j^v(\mathbf{r}'_i-\mathbf{r}'_j)}{|\mathbf{r}'_i-\mathbf{r}'_j|^3}\right].
\end{aligned} \tag{7.123}$$

If the COS model is used with reaction field, a reaction-field term is to be added to $\mathcal{V}^{(ele)}$

$$\begin{aligned}
V^{RF}(\boldsymbol{r}^N) &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{i=1}}^{N-1}\sum_{\substack{j>i \\ j \text{ inside cut-off } i}}^{N} (q_i-q_i^v)(q_j-q_j^v)\left(-\frac{\frac{1}{2}C_{rf}|\mathbf{r}_i-\mathbf{r}_j|^2}{{R_{rf}}^3} - \frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right) \\
&+ (q_i-q_i^v)q_j\left(-\frac{\frac{1}{2}C_{rf}|\mathbf{r}_i-\mathbf{r}'_j|^2}{{R_{rf}}^3} - \frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right) \\
&+ q_i^v(q_j-q_j^v)\left(-\frac{\frac{1}{2}C_{rf}|\mathbf{r}'_i-\mathbf{r}_j|^2}{{R_{rf}}^3} - \frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right) \\
&+ q_i^v q_j^v\left(-\frac{\frac{1}{2}C_{rf}|\mathbf{r}'_i-\mathbf{r}'_j|^2}{{R_{rf}}^3} - \frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right) \qquad (7.124) \\
&- \frac{1}{4\pi\epsilon_0\epsilon_{cs}}\sum_{i=1}^{N}\frac{q_i^2}{2}\frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \qquad (7.125)
\end{aligned}$$

where R_{rf} is the reaction-field cutoff and C_{rf} is the reaction-field constant defined as

$$C_{rf} = \frac{2(\epsilon_{cs} - \epsilon_{rf})(1 + \kappa_{rf}R_{rf}) - \epsilon_{rf}(\kappa_{rf}R_{rf})^2}{(\epsilon_{cs} + 2\epsilon_{rf})(1 + \kappa_{rf}R_{rf}) + \epsilon_{rf}(\kappa_{rf}R_{rf})^2} \tag{7.126}$$

where κ_{rf} is the inverse Debye screening length and ϵ_{rf} is the reaction-field dielectric permittivity outside the reaction-field cutoff radius R_{rf}. The reaction field term makes the electrostatic force and interaction zero at the reaction-field cut-off distance. Using a cut-off radius plus reaction field the summation over j in Eq. 7.123 is over sites j inside the cut-off of site i, and omits the covalently bound nearest neighbours j of atom i that are excluded from the non-bonded interaction. These neighbours j of i must not be excluded in the summation over j in Eq. 7.124 because they contribute to the reaction field[40]. The last term in Eq. 7.124 is a constant that is added to represent the self-interaction of the permanent, i.e. non-COS, charges.

The Coulomb contribution $\mathbf{E}_i^{coul}$ to the electrc field $\mathbf{E}_i$ is given by Eq. 7.116. The reaction field contribution $\mathbf{E}_i^{RF}$ is

$$\mathbf{E}_i^{RF} = \frac{C_{rf}}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{j \neq i \\ j \text{ inside cut-off } i}}^{N} \left[\frac{(q_j - q_j^v)(\mathbf{r}_i - \mathbf{r}_j)}{{R_{rf}}^3} + \frac{q_j^v(\mathbf{r}_i - \mathbf{r}'_j)}{{R_{rf}}^3} \right] . \tag{7.127}$$

Since iteratively solving Eqs. 7.112, 7.116 and 7.127 is costly, the number of iterations to be carried through till

$$\mathbf{E}_i = \mathbf{E}_i^{coul} + \mathbf{E}_i^{RF} \tag{7.128}$$

does not change significantly anymore should be kept low *i.e.* lower than a few. Since the molecular configuration will not change much between subsequent MD time steps Δt, one may use previous consistent values for the displacement of the virtual charge q_i^v

$$\Delta\mathbf{r}_i^v = \mathbf{r}'_i - \mathbf{r}_i, \tag{7.129}$$

i.e. iterated till $\Delta\mathbf{r}_i^v$ and $\mathbf{E}_i$ reach consistency, to obtain a good prediction of the next $\Delta\mathbf{r}_i^v$ to be used to start the iteration at $t + \Delta t$,

$$\Delta\mathbf{r}_i^v(predicted) = 2\Delta\mathbf{r}_i^v(t) - \Delta\mathbf{r}_i^v(t - \Delta t). \tag{7.130}$$

On average, two to three iterations are required at every time step to calculate the consistent fields with the convergence criterion of[41,42]

$$max_{i,x,y,z}(|\Delta E_{i,x}|, |\Delta E_{i,y}|, |\Delta E_{i,z}|)|q_O|d_{OH} < \Delta U \tag{7.131}$$

with $\Delta U = 2.5$ kJ mol^{-1}, and where $\Delta E_{i,x}$, $\Delta E_{i,y}$ and $\Delta E_{i,z}$ are the changes between consecutive iteration steps in the electric field component at the COS site i along the x-, y- and z- axes, q_O denotes the non-polarisable part of the charge of an oxygen atom (typically of a water molecule) and d_{OH} is the length of the OH bond of water molecule.

7.5.3. Off-atom sites. For certain polarisable models[43] it is necessary to add an additional virtual atomic center M. The position of this virtual site M is defined in terms of the positions of three non-virtual atoms i, j and k,

$$\mathbf{r}_M = \mathbf{r}_i + \frac{\gamma^{pol}}{2}(\mathbf{r}_{ji} + \mathbf{r}_{ki}) \tag{7.132}$$

where γ^{pol} is a constant, which determines the distance $d_{iM} = r_{iM}$ as a function of the distances $d_{ij} = r_{ij}$ and $d_{ik} = r_{ik}$. For example, in a model for H_2O, i would denote the oxygen atom and j and k the hydrogen atoms. The addition of the massless site M defined by Eq. 7.132 does not introduce any extra degrees of freedom into the molecule in the calculation of the kinetic energy of the system. The " pseudo-force" $\mathbf{f}_M$ that acts on the virtual-atom site M is redistributed to the non-virtual atoms i, j and k according to

$$\begin{aligned} &\mathbf{f}_i + (1 - \gamma^{pol})\mathbf{f}_M \\ &\mathbf{f}_j + \frac{\gamma^{pol}}{2}\mathbf{f}_M \\ &\mathbf{f}_k + \frac{\gamma^{pol}}{2}\mathbf{f}_M \end{aligned} \tag{7.133}$$

7.5.4. Non-linear polarizability. A problematic feature of many polarisable models, apart from their larger demand for computing power than nonpolarisable ones, is their tendency to allow for overpolarisation in the presence of strong local electric fields, leading to the polarisation catastrophe and a too large static dielectric permittivity. There are several approaches to resolve this problem. Using the GROMOS force fields the polarisation catastrophe is avoided by a big enough repulsive Lennard-Jones interaction between non-hydrogen atoms leading to dipole-dipole distances larger than $(4\alpha_i^2)^{1/6}$. Additionally the GROMOS simulation software allows the linear dependence of the induced dipole $\boldsymbol{\mu}_{ind,i}$ on the electric field $\mathbf{E}_i$ to be substituted by a sublinear dependence for large field strengths[44,45] which can be achieved by making the polarisability α_i electric field dependent. For example, α_i is replaced by $\alpha_{D,i} = \alpha_{D,i}(E)$ for large $\mathbf{E}_i$ using

$$\alpha_{D,i} = \begin{cases} \alpha_i & for\ E_i \le E_{0,i} \\ \frac{\alpha_i E_{0,i}}{p_i E}\left[p_i + 1 - \left(\frac{E_{0,i}}{E_i}\right)^{p_i}\right] & for\ E_i > E_{0,i}. \end{cases} \tag{7.134}$$

where p_i is a polarisation damping parameter. The induced dipole is then defined as

$$\boldsymbol{\mu}_i^{ind} = \begin{cases} \alpha_i \mathbf{E}_i & for\ E_i \le E_{0,i} \\ \frac{\alpha_i E_{0,i}}{p_i}\left[p_i + 1 - \left(\frac{E_{0,i}}{E_i}\right)^{p_i}\right]\frac{\mathbf{E}_i}{E_i} & for\ E_i > E_{0,i}. \end{cases} \tag{7.135}$$

This change also influences the self-polarisation contribution to the potential energy

$$V^{self,i} = \begin{cases} \frac{1}{2}\alpha_i E_i^2 & for\ E_i \le E_{0,i} \\ \frac{1}{2}\alpha_i E_{0,i}^2 & \\ \quad + \frac{\alpha_i E_{0,i}^2}{p_i(p_i-1)}\left[-p_i^2 + (p_i^2-1)\left(\frac{E_i}{E_{0,i}}\right) + \left(\frac{E_{0,i}}{E_i}\right)^{p_i-1}\right] & for\ E_i > E_{0,i} \end{cases} \tag{7.136}$$

with $V^{self} = \sum_i V^{self,i}$ where i runs over all polarisable centers.

CHAPTER 8

Coarse-grained models and multi-resolution simulation

8.1. Introduction

In chemistry, different levels of modelling, i.e. involving different types of degrees of freedom, can be chosen (Tab. 8.1). At the second most fine-grained level, one considers nuclei and electrons, as done in quantum chemistry. If one is not interested in breaking or forming chemical bonds or excited states of molecules, for example, one may eliminate the electronic degrees of freedom from the model and only consider atoms. In other words, the fine-grained model is coarse-grained by elimination of electronic degrees of freedom. This coarse-graining procedure can be applied between any two levels of modelling and thus any model in chemistry can be viewed as a coarse-grained model with respect to the eliminated degrees of freedom. However, the term coarse-grained modelling has predominately been used to indicate models in which the particles that constitute the degrees of freedom of the model represent more than one non-hydrogen atom.

Level	Particles	Size of bead /nm	Scaling effort	CG reduction N_{df}	CG reduction comput. effort
I	Nucleons + electrons	10^{-6}	$N_n^{\geq 3}$		
				10-100	$> 10^3$
II	Nuclei + electrons	$10^{-6} - 10^{-5}$	$N_e^{\geq 3}$		
				10-100	$> 10^3$
III	Atoms	0.03-0.3	N_a^{1-2}		
				2-5	2-25
IV	Supra-atomic beads	0.5-1.0	N_b^{1-2}		
				2-10	2-100
V	Supra-molecular beads	0.5-1.0	N_b^{1-2}		

TABLE 8.1. Characteristic sizes of particles at different levels of resolution of modelling, scaling of the computational effort as a function of the number of nucleons (N_n), electrons (N_e), atoms (N_a) or beads (N_b), and the reduction of the number of degrees of freedom or interactions N_{df} and the reduction of computational effort that can be achieved by coarse-graining to the next level of modelling

If these atoms belong to one molecule, such a model is a supra-atomic, or molecular, coarse-grained model. If the particles that constitute the degrees of freedom represent more than one molecule, such a model is a supra-molecular coarse-grained model.

The interactions governing the motion of the particles of the different levels are: (I) strong interaction, Coulomb and Pauli principle (II) Coulomb and Pauli principle (III) Coulomb, van der Waals, repulsive and bonded terms (IV) Coulomb, van der Waals, repulsive and bonded terms (V) Coulomb, van der Waals, repulsive terms The interactions of levels I and II are governed by quantum mechanics, while the interactions

of levels III-V are governed by classical statistical mechanics. The number of degrees of freedom, particles or interaction sites will determine, together with the applicable equations of motion (quantum- or classical-mechanical), the computational effort required, and thus the reduction of the latter that can be reached by coarse-graining (Tab. 8.1). Coarse-graining from level II to level III has different characteristics and problematic issues than coarse-graining from level III to level IV or V, because of the limited compatibility of quantum and classical mechanical concepts. Therefore, here we only consider coarse-graining in the realm of classical mechanics, i.e. between levels III, IV and V. For coarse-graining from level II to level III we refer to Ref.[3] and Chap. 15.

Coarse-graining implies eliminating degrees of freedom. This leads inevitably to a decrease of the applicability of the model. For example, when coarse-graining from level II (nuclei and electrons) to level III (atoms), relaxation of electronically excited states of molecules is not covered by the model any more. Generally, coarse-graining leads to a loss of accuracy of the model, although for particular properties and types of models this need not be the case. For example, the properties of liquid water at ambient temperature are more accurately described by the SPC model,[46] a level III model, than by level II ab initio models based on density-functional theory, due to the limited accuracy of the functionals used. In general, the choice of degrees of freedom to be eliminated depends on the property and phase of the substance of interest.

The conditions that must be fulfilled by degrees of freedom in order that they may be eliminated in a physically correct manner in the coarse-graining process, such that a computationally efficient and yet accurate coarse-grained model is obtained, are:

1. they must be non-essential for the process or property of interest.
2. they must be large in number or computationally intensive, so that the computational gain is substantial enough to offset the loss in accuracy.
3. the interactions governing these degrees of freedom to be eliminated should be largely decoupled from the interactions governing the other degrees of freedom of the system which are to be maintained. This means that the frequency components of the motion along the degrees of freedom to be eliminated must be well separated from the other frequencies occurring in the system, and that the coupling between the two types of motion is weak.[47]
4. their elimination should allow a simple, efficient representation of the interaction governing the other, remaining degrees of freedom.

Two examples of coarse-graining between levels III and IV are discussed: the use of so-called united atoms and of bond-length constraints. By treating the aliphatic CH, CH2 and CH3 groups as united atoms, the number of atomic interaction sites is substantially reduced, up to almost a factor of 10 fewer pairwise non-bonded interactions for lipids, at the cost of losing the dipolar interactions of the CH moieties and the van der Waals interactions of the H atoms. The intra-moiety motions of these CHn groups are largely decoupled from the motions of the other atoms and the torsional interactions involving these H atoms can be incorporated into the corresponding interactions for the torsional angle that does not involve an aliphatic H atom. If the positions of these H atoms are needed, i.e. when calculating quantities such as nuclear Overhauser effects (NOE s), residual dipolar couplings (RDCs) or S^2 order parameters measurable by NMR, the H atom positions can be easily recovered based on the positions of the carbon atom and its non-hydrogen covalently bound neighbours. Thus, all four conditions for appropriate coarse-graining are largely met in this case.

The other example of coarse-graining is the use of geometric constraints for small molecules without intra-molecular torsional-angle degrees of freedom, such as the solvents water, methanol or chloroform, or bond-length constraints in macromolecules.[47] The latter are standardly used in biomolecular simulations, because they satisfy conditions 1 to 4 and allow, through the use of SHAKE or other similar techniques to maintain such constraints, a gain of about a factor of four in computational efficiency.

An example of coarse-graining that does not satisfy conditions 3 and 4 is the use of an implicit solvent model, *i.e.* the attempt to mimic the effect of the solvent by a function that is only dependent on the solute coordinates. If the solvent is water, this leads to severe distortions of the energy surface of the solute. Although the motions of a large solute may cover time scales ranging from femtoseconds to milliseconds and the relaxation times of water molecules are of the order of picoseconds, their motions on picosecond to nanosecond time scales are not decoupled, and thus condition 3 is not satisfied for particular processes.

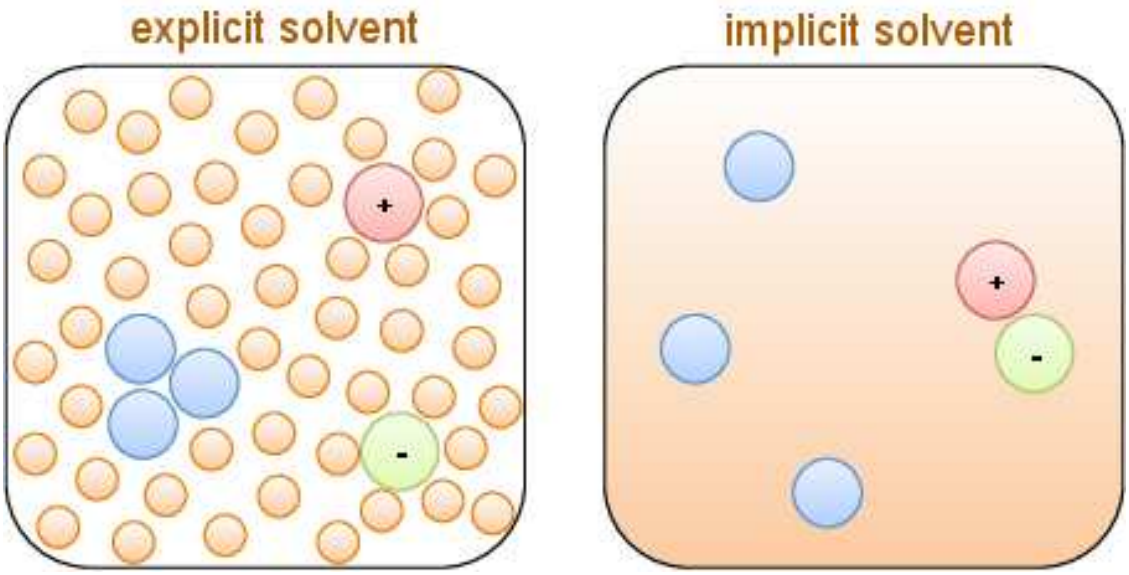


FIGURE 8.1. Illustration of the hydrophobic effect and the adverse effect of eliminating solvent degrees of freedom in the process of coarse-graining. The solvent is shown in orange, the hydrophobes in blue, and the ions in red (positively charged) and green (negatively charged).

In explicit solvents (orange particles in the left panel of Fig. 8.1), the non-polar particles (blue) aggregate, and the electrostatic interaction between ions (red and green) is reduced, leading to dissolution. So-called hydrophobic or non-polar particles do like water, but their interaction with water is less strong than the interaction of water with itself, leading to water excluding the hydrophobes and their subsequent aggregation. Ions with opposite charges do like water more than each other, which leads to water surrounding the ions and dissolution of ion pairs. The "hydrophobic effect", the apparent attraction between non-polar molecules or repulsion between ions in aqueous solution due to the stronger interaction between the water molecules or between water molecules and ions, cannot be properly modelled in terms of solute and ion coordinates only (right panel Fig. 8.1), because the effective interaction between solute atoms and their entropy is a complex function of the distribution of solvent coordinates. Thus, condition 4 is difficult to meet.

Coarse-graining from level III to IV for biomolecules is a challenge because of the heterogeneity of biomolecules. The scale invariance that lies at the heart of the renormalisation group approach to coarse-graining of largely homogeneous polymers is not observed for biopolymers, which are composed of many different, complex structural units that are connected through different types of interactions. In the coarse-graining process, the basic geometry and the balance between the various interactions must be preserved in order to avoid losing characteristic features of these molecules. In addition, entropy plays a non-negligible role in biomolecular processes, which means that the loss of entropy in the process of coarse-graining must be compensated for by a loss in energy in order to maintain the relevant free energy differences. Finally, the reduction of the computational effort between levels III and IV is rather modest compared to that between other levels (Tab. 8.1). These considerations lead us to the conclusion that coarse-graining from level III to level IV does not pay off for biomolecules such as proteins, DNA, RNA and sugars. Only a limited decrease in the number of interaction sites is reached at the cost of losing the essential characteristics of such molecules in terms of intra-molecular interactions, interactions with the solvent and entropy. Only lipids, which have relatively long homogeneous aliphatic tails, may be able to retain the dominant characteristics of an amphiphilic molecule with a particular geometry and flexibility upon coarse-graining from level III to level IV. Due to the abundance of lipids in membranes, the reduction in computational cost may off-set the loss of accuracy.

Since the inclusion of solvent degrees of freedom is essential to properly represent the hydrophobic effect and because the calculation of the solvent-solvent interactions in a simulation of a biomolecule such as a

protein or a fragment of DNA in aqueous solution dominates the computational effort, coarse-graining of the solvent degrees of freedom holds much promise to reduce the computational costs, in particular when more than one solvent molecule is subsumed into a supra-molecular bead. In the case of water, such coarse-graining from level III to level V should retain the thermodynamic and dielectric screening properties and hydrogen-bonding capacity of water as much as possible, and a proper ratio between entropy and energy.[48,49] This is not the case if a water bead is modelled as a Lennard-Jones particle without charge. Coarse-graining of solvent degrees of freedom in a biomolecular simulation has a good chance of meeting conditions 1 to 4, depending on how the coarse-grained interaction is modelled.

When coarse-graining from level III to level V, a few technical issues emerge that are generally not present in atomic-level models.

1. Atomic biomolecular force fields generally use a relative dielectric constant ϵ_{cs} of 1 in the Coulomb interaction, because there is vacuum between the atoms and the polarisability of atoms is neglected. The supra-molecular beads should represent the polarisability or dielectric screening capability of Nmol molecules. This is accounted for by using values of $\epsilon_{cs} > 1$ in the direct Coulomb interaction.
2. When comparing the pressure calculated for the supra-molecular beads with the desired experimental value, one should account for the fact that this pressure will be N_{mol} times smaller than in an atomic-level simulation by using a scaling factor $S_{SM} = N_{mol}$.
3. For thermodynamic quantities such as the heat of vaporisation, the excess free energy of a liquid or the free energy of solvation that are defined by a difference of an energy or free energy between the gas phase and the liquid phase, a meaningful comparison of values calculated with a supra-molecular model ($N_{mol} > 1$) and experimental ones is not possible, because it would require a reliable calculation of the (free) energy of cluster decomposition in the gas phase.

8.2. Multi-resolution simulation

Generally, a model developed for a particular level of modelling is only used at the same level of modelling. For example, models for small molecular compounds in the liquid phase are used to study the properties of mixtures of such compounds. However, this may limit the accessible time and length scales in biomolecular simulations. Because of the heterogeneity of biomolecular systems in terms of their relaxation time scales and the different types of interactions present it is of interest to combine different levels of modelling in one simulation or system.
The combination of different levels of modelling or resolution, *i.e.* multi-graining, can take different forms.[50]

1. The simulation switches between the two levels of modelling in time: multi-graining in time. This can be done in two ways:
   a. the simulation is performed at the coarse-grained level and particular configurations are later mapped back to the fine-grained level;
   b. a coupling parameter λ is introduced that defines a path between the fine-grained (*e.g.* $\lambda = 0$) and the coarse-grained (*e.g.* $\lambda = 1$) representation of the particles, which allows a smooth switching between different levels of modelling in *e.g.* a Hamiltonian replica-exchange simulation.[51]
2. The system contains a mixture of fine-grained and coarse-grained particles: multi-graining in space. This can be done in two ways:
   a. the space occupied by the system is divided into a fine-grained and a coarse-grained region with a small buffer region in which the particles change from one resolution to the other. The resolution of the particles thus depends on their position in space;
   b. the particles of the system are either fine-grained or coarse-grained and can freely mix. The resolution of the particles is thus fixed.[45,52,53]

Multi-graining of type 2(b) is for example also applied between level II and level III in so-called hybrid QM/MM simulations in which the electrons are treated quantum-mechanically (level II) and the nuclei and surrounding atoms classically (level III). See Chap. 15.

CHAPTER 9

Special force-field terms

9.1. Introduction

The interaction function $\mathcal{V}(\boldsymbol{r};\mathbf{s})$ as given in Eq. 3.4 may contain special terms indicated as $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$, which are only used under special circumstances. In GROMOS, the following special interaction terms have been implemented.

A. *atom position restraining* (Sec. 9.2)
B. *atom-atom distance restraining* (Sec. 9.3)
C. *bond-angle restraining* (Sec. 9.5)
D. *dihedral-angle restraining* (Sec. 9.6)
E. ^3J*-coupling constant restraining* (Sec. 9.7)
F. S^2*-order parameter restraining* (Sec. 9.8)
G. *crystallographic structure-factor amplitude restraining* (Sec. 9.9)
H. *crystallographic electron-density restraining* (Sec. 9.10)
I. *crystallographic symmetry restraining* (Sec. 9.11)
J. *distance-field restraining* (Sec. 9.12)
K. *local-elevation biasing* (Sec. 9.13)

In cases Pts. A-F, the motion along a particular type of degree of freedom is restrained. Therefore, these special terms are often called *restraining interaction terms*, $\mathcal{V}^{(res)}(\boldsymbol{r};\mathbf{s})$. They are commonly used in structure refinement based on NMR data and can also be employed as so-called umbrella functions to focus the sampling of configuration space when determining a free energy profile along a given coordinate (Sec. 14.8). Case K is an example of the use of a repulsive restraint, which diverts the system away from the parts of configuration space it has already visited. This technique is useful when searching conformational space and can also be used as an alternative to restraining. One may also choose to remove all motion for specific atoms or along certain degrees of freedom. Such hard boundary conditions are referred to constraints and will be the topic of Chap. 10.

When the interaction function $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ refers to non-atomic sites as centres of interaction, it is still possible to decompose the force on a non-atomic site into forces on those atoms, of which the atomic positions are used to define the position of the non-atomic site. The use of such non-atomic interaction sites, also called *virtual* or *pseudo atoms*, is discussed in Sec. 9.4. It has only been implemented in GROMOS in the context of atom-atom distance restraining special interaction terms.

When restraining an atom-atom distance or a ^3J-coupling constant or an S^2 order parameter to experimental values obtained from NMR experiments, which generally represent an average over time and space, one should *restrain* only the *time-average* of the function r^{-3} of the distance r or the *time average* of the ^3J-coupling constant or S^2 order parameter. This is discussed in Secs. 9.3, 9.7 and 9.8, respectively.

9.2. Atom-position restraining or fixed atoms

When simulating a molecular system, it may be desirable to fix specific atoms or parts of the system. In MD simulation it is in general not advisable to immobilize atoms completely, because this may reduce the flexibility of the system such that transitions and motions that are normally occurring in the system, are completely inhibited by the rigidity of the atoms. Therefore, a better way to keep specific atoms approximately at given reference positions is to restrain the motion of those atoms around these positions by applying a harmonic restraining force, which still leaves room for flexibility and mobility. The application of position constraints will be discussed in Sec. 10.3.

The special interaction term in $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4 that performs *atom position restraining* reads

$$\begin{aligned}\mathcal{V}^{(pr)}(\boldsymbol{r};\mathbf{s}) &= \sum_{n=1}^{\mathcal{N}^{(pr)}} \mathcal{V}^{(pr)}{}_n(\mathbf{r}_n; k^{(pr)}{}_n; \mathbf{r}_n^0) \\ &= \sum_{n=1}^{\mathcal{N}^{(pr)}} \tfrac{1}{2} k^{(pr)}{}_n [\mathbf{r}_n - \mathbf{r}_n^0]^2 \end{aligned} \tag{9.1}$$

The summation runs over a set of $\mathcal{N}^{(pr)}$ selected atoms. The fixed reference positions are denoted by $\mathbf{r}_n^0$. The harmonic oscillator force constant $k^{(pr)}{}_n$ can be chosen

- to be equal for all selected atoms, $k^{(pr)}{}_n = CPOR$, $(NTPOR = 1)$, or
- to be inversely proportional to the atomic B-factors of the selected atoms, $k^{(pr)}{}_n = CPOR/BFAC[n]$, $(NTPOR = 2)$,

where CPOR, NTPOR and BFAC[n] are user specified (see Vol. 4).

The actual position of the n-th restrained atom is denoted by $\mathbf{r}_n$.

The force on atom n due to the n-th term in Eq. 9.1 is

$$\mathbf{f}_n = -k^{(pr)}{}_n [\mathbf{r}_n - \mathbf{r}_n^0] \tag{9.2}$$

The atom sequence numbers of the restrained atoms are stored in JRC[1..NRC], $NRC = \mathcal{N}^{(pr)}$. These are used to select atom reference position coordinates from $XC[1..3 * \mathcal{N}_a]$, where $\mathcal{N}_a$ denotes the total number of atoms in the system. Reading of the reference positions in program MD++ is controlled by the switch NTPORB. The reference positions can either be read from a startup file $(NTPORB = 0)$ or from a special file $(NTPORB = 1)$. The specification of atoms and reference positions for atom position restraining is discussed in Secs. 4-3.11 and 4-4.2. Position restraining can be applied to any atom of solute or solvent. The switch NTPOR controls the atom position restraining or fixed atom option:

NTPOR	$= 0$,	no atom position restraining
	$= 1$,	harmonic atom position restraining, $k^{(pr)}{}_n = CPOR$
	$= 2$,	harmonic atom position restraining, $k^{(pr)}{}_n = CPOR/BFAC[n]$
	$= 3$,	position constraining

The reference positions can be scaled upon pressure scaling by using the switch NTPORS.

Applications of atom position restraining are the following.

- When a solute is placed in a box with solvent molecules, the solute-solvent atomic contacts may be very unfavourable. When performing energy minimization or molecular dynamics starting from such a configuration the solute conformation may be distorted by the bad non-bonded contacts with the solvent molecules. By applying atom position restraining to the solute atoms the unfavourable contacts can be relaxed without changing the solute conformation.
- When only a part of a molecular system is simulated under fixed boundary conditions (FBC), it is necessary to restrain the atoms in the wall region in order to avoid distortion of the system due to the vacuo boundary condition, see Sec. 4.3.

9.3. Distance restraining

When simulating a molecular system it may be desirable to restrain the distance between selected atoms to a given value or to only a minimum value or maximum value.[54] This can be performed by using as a special term $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in the interaction function (Eq. 3.4) a harmonic oscillator or one half of a harmonic oscillator for lower or upper bound. However, when the actual distance $r_{nn'}$ between atoms n and n' is very different from the reference distance $r^0_{nn'}$ the energy and force due to a harmonic function may become very large. Therefore, the special interaction term for atom-atom distance restraining is chosen to become linear beyond a specified deviation $\Delta r^h = r^1 - r^0$ of $r_{nn'}$ from $r^0_{nn'}$[55] (see Fig. 9.1).

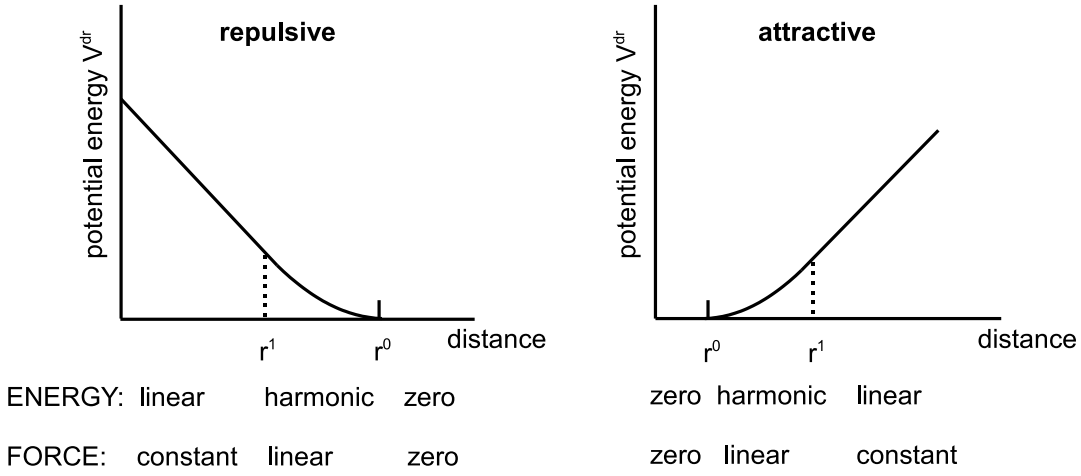


FIGURE 9.1. Potential energy term for atom-atom distance restraining. The function and its derivative are continuous at $r = r^0$ and $r = r^1$.

The special interaction term in $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4 that performs *atom-atom distance restraining* reads

$$\mathcal{V}^{(dr)}(\boldsymbol{r};\mathbf{s}) \quad = \quad \sum_{m=1}^{N^{(dir)}} \mathcal{V}^{(dr)}{}_m(r_{nn'};k^{(dr)}{}_m,r^0_m,\Delta r^h) \tag{9.3}$$

where the m-th atom-atom distance restraint involves atoms denoted by n and n'. The summation runs over a set of $N^{(dir)}$ distance restraints. An *attractive distance restraint* m with length r^0_m between atoms n and n' is represented by[56]

$$\begin{aligned}
&\mathcal{V}^{(dr)}{}_m(r_{nn'};k^{(dr)}{}_m,r^0_m,\Delta r^h) \\
&\quad = 0 && 0 < r_{nn'} < r^0_m \\
&\quad = \tfrac{1}{2}k^{(dr)}{}_m[r_{nn'}-r^0_m]^2 && r^0_m < r_{nn'} < r^0_m + \Delta r^h \\
&\quad = +k^{(dr)}{}_m[r_{nn'}-r^0_m-\tfrac{1}{2}\Delta r^h]\Delta r^h && r^0_m + \Delta r^h < r_{nn'}
\end{aligned} \tag{9.4}$$

The actual distance between atoms n and n' is denoted by $r_{nn'}$ and $r^1_m = r^0_m + \Delta r^h$ is the distance at which $\mathcal{V}^{(dr)}{}_m$ changes from a quadratic (harmonic) to a linear function of $r_{nn'}$

The forces on atoms n and n' due to $\mathcal{V}^{(dr)}{}_m$ in Eq. 9.4 are

$$\begin{aligned}
\mathbf{f}_n \quad &= \quad -\frac{\partial \mathcal{V}^{(dr)}{}_m}{\partial r_{nn'}}\frac{\partial r_{nn'}}{\partial \mathbf{r}_n} \\
&= \quad 0 && 0 < r_{nn'} < r^0_m \\
&= \quad -k^{(dr)}{}_m[r_{nn'}-r^0_m]\cdot[\mathbf{r}_{nn'}/r_{nn'}] && r^0_m < r_{nn'} < r^0_m + \Delta r^h \\
&= \quad -k^{(dr)}{}_m\cdot\Delta r^h\cdot[\mathbf{r}_{nn'}/r_{nn'}] && r^0_m + \Delta r^h < r_{nn'}
\end{aligned} \tag{9.5}$$

and

$$\mathbf{f}_{n'} \quad = \quad -\mathbf{f}_n \tag{9.6}$$

A *repulsive distance restraint* m with length r^0_m between atoms n and n' is represented by

$$\begin{aligned}
&\mathcal{V}^{(dr)}{}_m(r_{nn'}, k^{(dr)}{}_m, r^0_m, \Delta r^h) \\
&= -k^{(dr)}{}_m[r_{nn'} - r^0_m + \tfrac{1}{2}\Delta r^h]\Delta r^h \qquad 0 < r_{nn'} < r^0_m - \Delta r^h \\
&= \tfrac{1}{2}k^{(dr)}{}_m[r_{nn'} - r^0_m]^2 \qquad r^0_m - \Delta r^h < r_{nn'} < r^0_m \\
&= 0 \qquad r^0_m < r_{nn'}
\end{aligned} \tag{9.7}$$

where$r^1_m = r^0_m - \Delta r^h$ is the distance at which $\mathcal{V}^{(dr)}{}_m$ changes from a linear to a quadratic (harmonic) function of $r_{nn'}$.

The forces on atoms n and n' due to $\mathcal{V}^{(dr)}{}_m$ in Eq. 9.7 are

$$\begin{aligned}
\mathbf{f}_n &= -\frac{\partial \mathcal{V}^{(dr)}{}_m}{\partial r_{nn'}}\frac{\partial r_{nn'}}{\partial \mathbf{r}_n} \\
&= +k^{(dr)}{}_n \cdot \Delta r^h \cdot [\mathbf{r}_{nn'}/r_{nn'}] \qquad 0 < r_{nn'} < r^0_m - \Delta r^h \\
&= -k^{(dr)}{}_m[r_{nn'} - r^0_m] \cdot [\mathbf{r}_{nn'}/r_{nn'}] \qquad r^0_m - \Delta r^h < r_{nn'} < r^0_m \\
&= 0 \qquad r^0_m < r_{nn'}
\end{aligned} \tag{9.8}$$

and

$$\mathbf{f}_{n'} = -\mathbf{f}_n \tag{9.9}$$

The specification of the atoms n and n' is given in a distance restraints file as discussed in Sec. 9.4, of this volume and in Sec. 4-3.4 and Sec. 4-4.10. The force constants $k^{(dr)}{}_n$ can be chosen

- to be equal for all specified distance restraints, $k^{(dr)}{}_m = CDIR$, ($|NTDIR| = 1$), or
- to be proportional to a distance restraint weight factor W0[1..NB], $k^{(dr)}{}_m = CDIR * W0[m]$, ($|NTDIR| = 2$).

The factor CDIR is read from the input by program MD++. The reference distances r^0_m are denoted by $B0[1..NB] = R0[1..NDR]$, where $NB = NDR = N^{(dir)}$. Attractive restraining is selected when $B0 = R0 > 0$, whereas repulsive restraining is selected by changing the sign of B0 = R0 (< 0). Of course, $r^0_m = |R0|$ is used in the formulae. The distance Δr^h is denoted by $DB0 = DIR0$.

When the given atom-atom distance restraints r^0_m in Eq. 9.3 have been derived from NOE cross-peak intensities originating from nuclei n and n', they represent an average over space and time,

$$< r^{-p}_{nn'} >^{-\frac{1}{p}} . \tag{9.10}$$

This means that in the restraining interaction $\mathcal{V}^{(dr)}{}_m(r_{nn'}; k^{(dr)}{}_m, r^0_m, \Delta r^h)$, the instantaneous distance $r_{nn'}$ should be replaced by the average Eq. 9.10, so that only this average is restrained. The ensemble average Eq. 9.10 can be taken as a time (trajectory) average

$$< r^{-p}_{nn'} >= \overline{r^{-p}_{nn'}(t)} \equiv t^{-1}\int_0^t r^{-p}(t')dt' \tag{9.11}$$

or as an average over space, that is, different molecules[57]. In MD or SD simulations, the use of the time average Eq. 9.11 is the natural choice. When averaging over relatively short times, the angular correlation in the vector $\mathbf{r}_{nn'}$ should be neglected, which gives $p = 3$. The true average Eq. 9.11 is not suitable for use in Eq. 9.3, from which the restraining force is obtained during a simulation: the rate of change of $\overline{r^{-3}_{nn'}(t)}$

depends on the length of the averaging period, t. This problem is avoided by building a decay into the summation over time with a characteristic decay time or *memory relaxation time* τ_{dr}, so that[58]

$$< r_{nn'}^{-3} >= \overline{r_{nn'}^{-3}(t;\tau_{dr})} \equiv \frac{\int_0^t \exp(-(t-t')/\tau_{dr}) r_{nn'}^{-3}(t')dt'}{\tau_{dr}[1-exp(-t/\tau_{dr})]} \tag{9.12}$$

This time average is easily obtained in a simulation using discrete time steps Δt. When

$$t \gg \tau_{dr} \tag{9.13}$$

we have

$$\begin{aligned} \overline{r_{nn'}^{-3}(t;\tau_{dr})} &= [1-\exp(-\Delta t/\tau_{dr})] r_{nn'}^{-3}(t) \\ &+ \exp(-\Delta t/\tau_{dr}) \overline{r_{nn'}^{-3}(t-\Delta t;\tau_{dr})}. \end{aligned} \tag{9.14}$$

Upon insertion of

$$\overline{r}_{nn'} \equiv \left[\overline{r_{nn'}^{-3}(t;\tau_{dr})}\right]^{-\frac{1}{3}} \tag{9.15}$$

into the distance restraining interaction Eq. 9.3 we obtain for the force on atom n due to ${\mathcal{V}^{(dr)}}_m$ at time t

$$\mathbf{f}_n = -\frac{\partial {\mathcal{V}^{(dr)}}_m(\overline{r}_{nn'}; {k^{(dr)}}_m, r_m^0, \Delta r^h)}{\partial \overline{r}_{nn'}} \frac{\partial \overline{r}_{nn'}}{\partial r_{nn'}} \frac{\mathbf{r}_{nn'}}{r_{nn'}} \tag{9.16}$$

The second factor in Eq. 9.16 is

$$\frac{\partial \overline{r}_{nn'}}{\partial r_{nn'}} = [1-\exp(-\Delta t/\tau_{dr})] \left(\frac{\overline{r}_{nn'}}{r_{nn'}}\right)^4 \tag{9.17}$$

and causes large fluctuations in the force. The switch FORCESCALE in the DISTANCERES block of MD++ allows the selection of approximations. Setting FORCESCALE = 0, leads to the approximation

$$\frac{\partial \overline{r}_{nn'}}{\partial r_{nn'}} = 1 \tag{9.18}$$

and setting FORCESCALE = 1, leads to the approximation

$$\frac{\partial \overline{r}_{nn'}}{\partial r_{nn'}} = [1-\exp(-\Delta t/\tau_{dr})] \tag{9.19}$$

while when using FORCESCALE = 2, equation Eq. 9.17 is used in formula Eq. 9.16. We note that the total energy will not be conserved when applying time averaging due to the approximation Eq. 9.18 and the dependence of the restraining interaction Eq. 9.3 on time points before time t[59].

At the start of a simulation, the value of the average Eq. 9.15 is set such as to satisfy the restraint distance,

$$\left[\overline{\mathbf{r}_{nn'}^{-3}(t=0;\tau_{dr})}\right]^{-\frac{1}{3}} = r_m^0 \tag{9.20}$$

At the end of a simulation the value of the average Eq. 9.15 is stored with the final configuration for use in a continuation simulation (Sec. 4-4.10)

The choice of the values for the parameters ${k^{(dr)}}_m$ and τ_{dr} = TAUDIR is discussed in[60]. The latter should satisfy the condition

$$\tau_{dr} \ll t_{MD} \tag{9.21}$$

where t_{MD} is the length of the simulation.

The switch NTDIR in program MD++ controls the atom-atom distance restraining options:

NTDIR	=	0,	no atom-atom distance restraining
	>	0,	atom-atom distance restraining, no time averaging
	<	0,	atom-atom distance restraining with time averaging
|NTDIR|	=	1,	force constants equal, ${k^{(dr)}}_m = CDIR$
	=	2,	force constants proportional to the individual distance restraint weight factors, ${k^{(dr)}}_m = CDIR * W0[m]$

Applications of atom-atom distance restraining are the following.

- When a molecular structure is to be obtained that satisfies a set of given interatomic distances, atom-atom distance restraining can be used during EM or MD simulation to force the molecular conformation in the desired direction.
- When a part of a molecule is required to keep its form during a simulation, atom-atom distance restraining can be used to fix relative atom positions of a group of atoms without restraining the mobility of the group.
- When a free energy (profile) as a function of the distance between two atoms is to be determined by MD or SD simulation, the atom-atom distance restraining term can serve as umbrella function to focus the sampling (Sec. 14.8).
- In mixed fine-grained / coarse-grained simulations, a layer of fine-grained solvent molecules may be kept around the solute by applying the appropriate atom-atom distance restraints. Note that for this application, a contribution to the virial due to the special interaction energy terms is appropriate, which can be selected using option VDIR = 1 in the MD++ input file.

9.4. Virtual and pseudo atoms

In the GROMOS force fields most aliphatic hydrogen atoms are not explicitly treated, but are incorporated in the carbon atom to which they are attached forming united atoms (Vol. 3). However, an atom-atom distance restraint which is derived from proton NMR experiments, may refer to such a hydrogen atom, which is called here a *virtual atom.* In that case the distance restraint interaction (Eq. 9.3) refers to a *non-atomic site as a centre of interaction.* A proton-proton distance restraint may also refer to non-atomic sites when a stereospecific assignment of a resonance to a proton cannot be obtained, e.g. C_β protons in proteins or methyl groups in the amino acid residues Leu and Val, or when dynamic effects such as rotation of methyl group hydrogens and flipping of aromatic rings influence the NMR signal. In these cases the distance restraint must refer to a *pseudo atom* and a *correction term* Δr_n^{ps} must be added to the restraint distance $r_{nn'}^0$

$$r_{nn'}^0(pseudo) = r_{nn'}^0(real) + \Delta r_n^{ps} + \Delta r_{n'}^{ps} \tag{9.22}$$

The value of Δr_n^{ps}depends on the geometry of construction of the pseudo atom site and is given in Tab. 9.1. So, virtual and pseudo sites or atoms are massless points, whose positions are rigorously related to the positions of the masses in the molecule.

When using pseudo or virtual sites, two additional steps are added to the calculation of the forces on the real atoms.

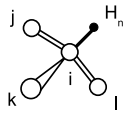
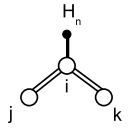
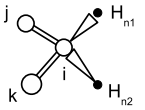
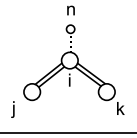
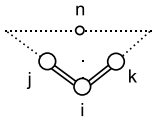
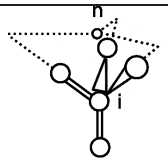
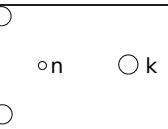
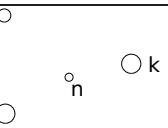
group	configuration	atom type	correction term Δr^{ps} on distance restraint r^0 *(nm)*	geometric code TYPE1 or TYPE2 *)
CH1 (aliphatic)		virtual	.00	1
CH1 (aromatic)		virtual	.00	2
CH2 (stereospecific)		virtual	.00	4
CH2 (non-stereospecific)		pseudo	.09	3
CH3		pseudo	.10	5
CH3 (non-stereospecific, Val, Leu)		pseudo	.22	6
CH3 (non-stereospecific, t-butyl)		pseudo	.23	7
COG (center of geometry)		pseudo	.00	-1
COM (center of mass)		pseudo	.00	-2

TABLE 9.1. Virtual and pseudo hydrogen atoms and distance restraint correction terms.
*) see Sec. 4-3.4

1. Before the interaction terms Eqs. 9.3, 9.4, 9.7 involving virtual or pseudo sites can be evaluated, the virtual or pseudo site n must be constructed using the positions of those atoms (i,j,k,l) that define the virtual or pseudo atom n.
2. The force $\mathbf{f}_n$ acting on the virtual or pseudo site n must be redistributed over the atoms (i,j,k,l) that define the virtual or pseudo atom n. The contribution of $\mathbf{f}_n$ to the force on atom i is *e.g.*

$$\mathbf{f}_i = \mathbf{f}_{nx}\frac{\partial x_n}{\partial \mathbf{r}_i} + \mathbf{f}_{ny}\frac{\partial y_n}{\partial \mathbf{r}_i} + \mathbf{f}_{nz}\frac{\partial z_n}{\partial \mathbf{r}_i} \tag{9.23}$$

or in matrix notation

$$\begin{pmatrix} \mathbf{f}_{ix} \\ \mathbf{f}_{iy} \\ \mathbf{f}_{iz} \end{pmatrix} = \begin{pmatrix} \partial x_n/\partial x_i & \partial y_n/\partial x_i & \partial z_n/\partial x_i \\ \partial x_n/\partial y_i & \partial y_n/\partial y_i & \partial z_n/\partial y_i \\ \partial x_n/\partial z_i & \partial y_n/\partial z_i & \partial z_n/\partial z_i \end{pmatrix} \begin{pmatrix} \mathbf{f}_{nx} \\ \mathbf{f}_{ny} \\ \mathbf{f}_{nz} \end{pmatrix} \tag{9.24}$$

Corresponding formulae hold for atoms j, k and l. The matrices of partial derivatives in Eq. 9.24 can easily be derived from the definition of $\mathbf{r}_n$ in terms of $\mathbf{r}_i$, $\mathbf{r}_j$, $\mathbf{r}_k$ and $\mathbf{r}_l$. These definitions have been kept as simple as possible in order to avoid too complex derivatives.

The virtual and pseudo atoms that can be used when the distance restraining potential energy term (Eqs. 9.3, 9.4, 9.7) is applied, are displayed in Tab. 9.1 (see also[55]). Three types of virtual atoms and six types of pseudo atoms are distinguished:

9.4.1. CH1-group (aliphatic). The position vector of the hydrogen H_n is given by

$$\mathbf{r}_n = \mathbf{r}_i + d_{CH}\mathbf{s}/s \tag{9.25}$$

with

$$\mathbf{s} = 3\mathbf{r}_i - \mathbf{r}_j - \mathbf{r}_k - \mathbf{r}_l \tag{9.26}$$

and

$$\begin{aligned} s &= (\mathbf{s}\cdot\mathbf{s})^{\frac{1}{2}} \\ &= [{s_x}^2 + {s_y}^2 + {s_z}^2]^{\frac{1}{2}} \end{aligned} \tag{9.27}$$

The carbon-hydrogen distance d_{CH} is stored in DISH. The partial derivatives are

$$\frac{\partial \mathbf{r}_n}{\partial \mathbf{r}_i} = \begin{pmatrix} [1+3d_{CH}(s^2-{s_x}^2)/s^3] & -3d_{CH}s_ys_x/s^3 & -3d_{CH}s_zs_x/s^3 \\ -3d_{CH}s_xs_y/s^3 & [1+3d_{CH}(s^2-{s_y}^2)/s^3] & -3d_{CH}s_zs_y/s^3 \\ -3d_{CH}s_xs_z/s^3 & -3d_{CH}s_ys_z/s^3 & [1+3d_{CH}(s^2-{s_z}^2)/s^3] \end{pmatrix} \tag{9.28}$$

and

$$\frac{\partial \mathbf{r}_n}{\partial \mathbf{r}_j} = \begin{pmatrix} -d_{CH}(s^2-{s_x}^2)/s^3 & d_{CH}s_ys_x/s^3 & d_{CH}s_zs_x/s^3 \\ d_{CH}s_xs_y/s^3 & -d_{CH}(s^2-{s_y}^2)/s^3 & d_{CH}s_zs_y/s^3 \\ d_{CH}s_xs_z/s^3 & d_{CH}s_ys_z/s^3 & -d_{CH}(s^2-{s_z}^2)/s^3 \end{pmatrix} \tag{9.29}$$

and $\partial\mathbf{r}_n/\partial\mathbf{r}_k$ and $\partial\mathbf{r}_n/\partial\mathbf{r}_l$ are identical to Eq. 9.29.

9.4.2. CH1-group (aromatic). The position vector of the hydrogen H_n is given by

$$\mathbf{r}_n = \mathbf{r}_i + d_{CH}\mathbf{s}/s \tag{9.30}$$

with

$$\mathbf{s} = 2\mathbf{r}_i - \mathbf{r}_j - \mathbf{r}_k \tag{9.31}$$

and s defined by Eq. 9.27. Again, the carbon-hydrogen distance is denoted by d_{CH}. The partial derivatives are

$$\partial\mathbf{r}_n/\partial\mathbf{r}_i = \begin{pmatrix} [1+2d_{CH}(s^2-{s_x}^2)/s^3] & -2d_{CH}s_ys_x/s^3 & -2d_{CH}s_zs_x/s^3 \\ -2d_{CH}s_xs_y/s^3 & [1+2d_{CH}(s^2-{s_y}^2)/s^3] & -2d_{CH}s_zs_y/s^3 \\ -2d_{CH}s_xs_z/s^3 & -2d_{CH}s_ys_z/s^3 & [1+2d_{CH}(s^2-{s_z}^2)/s^3] \end{pmatrix} \tag{9.32}$$

and $\partial\mathbf{r}_n/\partial\mathbf{r}_j$ and $\partial\mathbf{r}_n/\partial\mathbf{r}_k$ are given by Eq. 9.29.

9.4.3. CH2-group (two virtual protons). The position vector of the hydrogen H_{n1} is given by

$$\mathbf{r}_{n_1} = \mathbf{r}_i + \alpha\mathbf{s}/s + \beta\mathbf{q}/q \tag{9.33}$$

with $\mathbf{s}$ defined by Eq. 9.31, s by Eq. 9.27 and $\mathbf{q}$ by

$$\mathbf{q} = (\mathbf{r}_i - \mathbf{r}_j) \times (\mathbf{r}_i - \mathbf{r}_k) \tag{9.34}$$

with

$$q = ({q_x}^2 + {q_y}^2 + {q_z}^2)^{\frac{1}{2}} \tag{9.35}$$

The values of α and β are derived from

$$\begin{aligned} \alpha &= d_{CH}\cos(\theta/2) \\ \beta &= d_{CH}\sin(\theta/2) \end{aligned} \tag{9.36}$$

where θ is the tetrahedral angle and the carbon-hydrogen distance is d_{CH}. The partial derivatives are (for $n = n_1$):

$$\partial\mathbf{r}_n/\partial\mathbf{r}_i = \mathbf{A} + \mathbf{B} + \mathbf{C} \tag{9.37}$$

with

$$\mathbf{A} = \begin{pmatrix} [1 + 2\alpha(s^2 - {s_x}^2)/s^3] & -2\alpha s_y s_x/s^3 & -2\alpha s_z s_x/s^3 \\ -2\alpha s_x s_y/s^3 & [1 + 2\alpha(s^2 - {s_y}^2)/s^3] & -2\alpha s_z s_y/s^3 \\ -2\alpha s_x s_z/s^3 & -2\alpha s_y s_z/s^3 & [1 + 2\alpha(s^2 - {s_z}^2)/s^3] \end{pmatrix} \tag{9.38}$$

and

$$\mathbf{B} = \begin{pmatrix} -\beta q_x a_x/q^3 & -\beta q_y a_x/q^3 & -\beta q_z a_x/q^3 \\ -\beta q_x a_y/q^3 & -\beta q_y a_y/q^3 & -\beta q_z a_y/q^3 \\ -\beta q_x a_z/q^3 & -\beta q_y a_z/q^3 & -\beta q_z a_z/q^3 \end{pmatrix} \tag{9.39}$$

and

$$\mathbf{C} = \begin{pmatrix} 0 & +\beta b_z/q & -\beta b_y/q \\ -\beta b_z/q & 0 & +\beta b_x/q \\ +\beta b_y/q & -\beta b_x/q & 0 \end{pmatrix} \tag{9.40}$$

where $\boldsymbol{a} = \mathbf{q} \times \mathbf{r}_{kj} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ and $\boldsymbol{b} = \mathbf{r}_{kj}$, and

$$\partial\mathbf{r}_n/\partial\mathbf{r}_j = \mathbf{D} + \mathbf{E} + \mathbf{F} \tag{9.41}$$

where

$$\mathbf{D} = \begin{pmatrix} -\alpha(s^2 - {s_x}^2)/s^3 & \alpha s_y s_x/s^3 & \alpha s_z s_x/s^3 \\ \alpha s_x s_y/s^3 & -\alpha(s^2 - {s_y}^2)/s^3 & \alpha s_z s_y/s^3 \\ \alpha s_x s_z/s^3 & \alpha s_y s_z/s^3 & -\alpha(s^2 - {s_z}^2)/s^3 \end{pmatrix} \tag{9.42}$$

and $\mathbf{E}$ is identical to $\mathbf{B}$ in Eq. 9.39 but with $\boldsymbol{a} = \mathbf{q} \times \mathbf{r}_{ik}$ and $\mathbf{F}$ is identical to $\mathbf{C}$ in Eq. 9.40 but with $\boldsymbol{b} = \mathbf{r}_{ik}$, and

$$\partial\mathbf{r}_n/\partial\mathbf{r}_k = \mathbf{D} + \mathbf{G} + \mathbf{H} \tag{9.43}$$

where $\mathbf{G}$ is identical to $\mathbf{B}$ in Eq. 9.39 but with $\boldsymbol{a} = \mathbf{q} \times \mathbf{r}_{ji}$ and $\mathbf{H}$ is identical to $\mathbf{C}$ in Eq. 9.40 but with $\boldsymbol{b} = \mathbf{r}_{ji}$.

The position vector $\mathbf{r}_{n2}$ of the second virtual proton H_{n2} can be obtained from the same formulae by interchanging j and k.

9.4.4. CH2-groups (one pseudo site). When a distance restraint refers to one proton of a CH2-group of which no stereospecific assignment is known, the restraint is referred to a pseudo site between the two protons. The position vector of the pseudo atom n is defined by

$$\mathbf{r}_n = \mathbf{r}_i + \alpha\mathbf{s}/s \tag{9.44}$$

where α is given by Eq. 9.36 and $\mathbf{s}$ is defined by Eq. 9.31 and s by Eq. 9.27. The partial derivatives are

$$\partial\mathbf{r}_n/\partial\mathbf{r}_i = \mathbf{A} \tag{9.45}$$

and

$$\partial\mathbf{r}_n/\partial\mathbf{r}_j = \partial\mathbf{r}_n/\partial\mathbf{r}_k = \mathbf{D} \tag{9.46}$$

9.4.5. CH3-group (one pseudo site). For a single methyl group and for a methyl group of a diastereotopic pair for which the stereospecific assignment is known, the pseudo atom n is defined to be in the middle of the three hydrogens:

$$\mathbf{r}_n = \mathbf{r}_i + \gamma\mathbf{s}/s \tag{9.47}$$

with

$$\gamma = d_{CH}\cos(\pi - \theta) \tag{9.48}$$

and

$$\mathbf{s} = \mathbf{r}_i - \mathbf{r}_j \tag{9.49}$$

Again, θ is the tetrahedral angle and the carbon-hydrogen distance is d_{CH}.

The partial derivatives are

$$\frac{\partial\mathbf{r}_n}{\partial\mathbf{r}_i} = \begin{pmatrix} [1+\gamma(s^2-{s_x}^2)/s^3 & -\gamma s_y s_x/s^3 & -\gamma s_z s_x/s^3 \\ -\gamma s_x s_y/s^3 & [1+\gamma(s^2-{s_y}^2)/s^3] & -\gamma s_z s_y/s^3 \\ -\gamma s_x s_z/s^3 & -\gamma s_y s_z/s^3 & [1+\gamma(s^2-{s_z}^2)/s^3] \end{pmatrix} \tag{9.50}$$

and $\partial\mathbf{r}_n/\partial\mathbf{r}_j$ is identical to $\mathbf{D}$ in Eq. 9.42 but with α replaced by γ.

9.4.6. Two CH3-groups (one pseudo site). When a restraint involves a methyl group of a stereotopic pair for which no stereospecific assignments are known, the restraint is referred to a pseudo atom n at the geometric mean position of the 6 hydrogens of the pair:

$$\mathbf{r}_n = \mathbf{r}_i + \delta\mathbf{s}/s \tag{9.51}$$

with

$$\delta = -\cos(\theta/2)[d_{CC} + \gamma] \tag{9.52}$$

where $\mathbf{s}$ is defined by Eq. 9.31, the carbon-carbon distance d_{CC} is stored in DISC, θ is tetrahedral and γ is given by Eq. 9.48. The partial derivative $\partial\mathbf{r}_n/\partial\mathbf{r}_i$ is identical to $\mathbf{A}$ in Eq. 9.38, but with α replaced by δ, $\partial\mathbf{r}_n/\partial\mathbf{r}_j$ and $\partial\mathbf{r}_n/\partial\mathbf{r}_k$ are identical to $\mathbf{D}$ in Eq. 9.42 but with α replaced by δ.

9.4.7. Three CH3-groups (one pseudo site). When a distance restraint refers to a t-butyl for which no stereospecific assignments are known, the restraint is referred to a pseudoatom n at the geometric mean position of the 9 hydrogens of the three CH3 groups:

$$\mathbf{r}_n = \mathbf{r}_i + \varepsilon\mathbf{s}/s \tag{9.53}$$

with

$$\varepsilon = (d_{CC} + \gamma)\cos(\pi - \theta) \tag{9.54}$$

where $\mathbf{s}$ is defined by Eq. 9.49, the carbon-carbon distance d_{CC} is stored in DISC, γ is given by Eq. 9.48, and θ is the tetrahedral angle. The partial derivative $\partial\mathbf{r}_n/\partial\mathbf{r}_i$ is identical to Eq. 9.50, but with γ replaced by ε, and $\partial\mathbf{r}_n/\partial\mathbf{r}_j$ is identical to $\mathbf{D}$ in Eq. 9.42 but with α replaced by ε.

9.4.8. Center of geometry (one pseudo site). For the δ and ε protons of tyrosine and phenylalanine rings that are displaying fast 180^o ring flips, or for the protons i and j of a planar NH2 group, a pseudo atom can be constructed at the center of geometry of these atoms.

$$\mathbf{r}_n = N_{ps}^{-1} \sum_{i=1}^{N_{ps}} \mathbf{r}_i \tag{9.55}$$

where N_{ps} denotes the number of non-virtual atoms with position vector $\mathbf{r}_i$ the centre of geometry of which is to serve as pseudo atom n. The partial derivative $\partial \mathbf{r}_n / \partial \mathbf{r}_i$ is

$$\partial \mathbf{r}_n / \partial \mathbf{r}_i \quad = \quad \begin{pmatrix} N_{ps}^{-1} & 0 & 0 \\ 0 & N_{ps}^{-1} & 0 \\ 0 & 0 & N_{ps}^{-1} \end{pmatrix} \tag{9.56}$$

9.4.9. Center of mass (one pseudo site). Alternatively, one may choose to define a pseudo atom at the centre of mass of given atoms using

$$\mathbf{r}_n = M_{ps}^{-1} \sum_{i=1}^{N_{ps}} m_i \mathbf{r}_i \tag{9.57}$$

with

$$M_{ps} = \sum_{i=1}^{N_{ps}} m_i \tag{9.58}$$

where N_{ps} denotes the number of non-virtual atoms with position vector $\mathbf{r}_i$ and mass m_i the centre of mass of which is to serve as pseudo atom n. The partial derivative $\partial \mathbf{r}_n / \partial \mathbf{r}_i$ is

$$\partial \mathbf{r}_n / \partial \mathbf{r}_i \quad = \quad \begin{pmatrix} m_i/M_{ps} & 0 & 0 \\ 0 & m_i/M_{ps} & 0 \\ 0 & 0 & m_i/M_{ps} \end{pmatrix} \tag{9.59}$$

The various atom types are selected using the code shown in Tab. 9.1. For the first atom n_1 of the pairs these codes are listed in TYPE1[1..NB] = TYPE1[1..NDR] and the atom sequence numbers of the atoms i, j, k and l that define atom n_1 are given in I1, J1, K1, L1[1..NB] see Sec. 4-3.4. For the second atom n_2 of the pairs the corresponding arrays are denoted by TYPE2, etc. The carbon-hydrogen distance is denoted by DISH and the carbon-carbon distance by DISC.

We note that the correction terms Δr_n^{ps} which are listed in Tab. 9.1 are *not* automatically incorporated into the specified restraint distance r_m^0 by applying Eq. 9.22. The user must incorporate the correction terms Δr_n^{ps} into B0[1..NB] = R0[1..NDR]. The GROMOS++ program *prep_noe* may help to prepare the correct distance restraints for virtual and pseudo atoms commonly encountered in biomolecular systems.

9.5. Bond-angle restraining

One may wish to restrain a bond angle to a given value. For this case of restraining no special subroutines have been made in GROMOS. The *bond angle* $\theta(ijk)$ between atoms i, j and k can be restrained by applying distance restraints (Sec. 9.3) to the three distances ij, jk and ik. An alternative is to add to the molecular topology of a molecular system the bond angle $\theta(ijk)$ and to choose appropriate values for $K_{\theta n}$ and θ_{0n} (Sec. 5.2).

9.6. Dihedral-angle restraining

The special interaction term $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4 that performs *dihedral-angle restraining* reads

$$\mathcal{V}^{(tr)}(\boldsymbol{r};\mathbf{s}) \quad = \quad \sum_{n=1}^{\mathcal{N}^{(tr)}} {\mathcal{V}^{(tr)}}_n(\varphi_n; k^{(tr)}, \varphi_n^0, \Delta\varphi^h) \tag{9.60}$$

The summation runs over a set of $\mathcal{N}^{(tr)}$ selected dihedral angles φ_n , which are defined by specifying the atom sequence numbers i, j, k and l of the atoms forming dihedral angle $\varphi_n(i-j-k-l)$. The ${\mathcal{V}^{(tr)}}_n$ reads

$${\mathcal{V}^{(tr)}}_n(\varphi_n; k^{(tr)}, \varphi_n^0, \Delta\varphi^h)$$
$$= \quad -{k^{(tr)}}_n(\Delta\varphi_n + \tfrac{1}{2}\Delta\varphi^h)\Delta\varphi^h \qquad\qquad \Delta\varphi_n < \Delta\varphi^h$$

$$
\begin{aligned}
&= \tfrac{1}{2}k^{(tr)}{}_n(\Delta\varphi_n)^2 && -\Delta\varphi^h < \Delta\varphi_n < \Delta\varphi^h \\
&= k^{(tr)}{}_n(\Delta\varphi_n - \tfrac{1}{2}\Delta\varphi^h)\Delta\varphi^h && \Delta\varphi_n > \Delta\varphi^h
\end{aligned}
\tag{9.61}
$$

where $\Delta\varphi_n = \varphi_n - \varphi_n^0 + 2m\pi$, where m is chosen such that φ_n is within the range $[\varphi_n^0 + \delta_n - 2\pi, \varphi_n^0 + \delta_n]$. Using this dihedral angle restraint formulation δ_n determines at which position the direction of the rotation around the dihedral angle inverts. Typically, δ_n is set to 180^o. The interaction $\mathcal{V}^{(tr)}(\boldsymbol{r};\mathbf{s})$ has the same form as the improper dihedral angle interaction $V^{(\xi)}(\boldsymbol{r};\mathbf{s})$ described in Sec. 5.3. So, the formulae Eq. 17.11-Eq. 17.14 give the forces on atoms i, j, k and l due to the interaction (Eq. 9.60), if ξ_n is replaced by φ_n, $k_n^{(\xi)}$ by $k^{(tr)}{}_n$ and ξ_n^0 by φ_n^0.

The atom sequence numbers of the atoms i, j, k and l that define φ_n and the force constant $k^{(tr)}{}_n$ and reference dihedral angle φ_n^0 are to be specified in a dihedral angle restraints file and stored in the arrays IPLR, JPLR, KPLR, LPLR, WPLR, PDLR, DELTA[1..NDLR], with NDLR $= \mathcal{N}^{(tr)}$, as described in Sec. 4-3.5. The dihedral angle restraining interaction (Eq. 9.60) can be multiplied by an overall weight factor, CDLR, which is read from the input by program MD++. The switch NTDLR controls the dihedral angle restraining option:

NTDLR	$= 0$,	no dihedral-angle restraining
	$= 1$,	dihedral-angle restraining with equal force constants, $k^{(tr)}{}_n = CDLR$
	$= 2$,	dihedral angle restraining with force constants proportional to the individual dihedral angle weight factors, $k^{(tr)}{}_n = CDLR * WDLR[n]$
	$= 3$,	dihedral angle constraining

Dihedral angle restraining may be *applied* as umbrella function when a free energy (profile) as a function of a dihedral angle is to be determined by MD or SD simulation.[61,62] The implementation of dihedral angle constraints will be discussed in Sec. 10.6.

9.7. ^3J-coupling constant restraining

When simulating a molecular system, it may be desirable to restrain the spin-spin ^3J-coupling constant, $^3J_{mm'}$, between two nuclei m and m', to a given value $^3J^0$. If $^3J^0$ is an experimental value, measured as an average over time and space in an NMR experiment, then the time-average should be restrained.

The ^3J-coupling constant $^3J_{mm'}$ depends on the value of the dihedral angle $\zeta_n(m-j-k-m')$ involving the three covalent bonds connecting the atoms m and m' through atoms j and k according to the Karplus relation (see, e.g. Fig. 9.2)

$$
\begin{aligned}
{}^3J_{mm'} &= a\cos^2(\zeta_n(m-j-k-m')) \\
&\quad + b\cos(\zeta_n(m-j-k-m')) + c.
\end{aligned}
\tag{9.62}
$$

The coefficients a, b and c will depend on the types of the atoms m, j, k and m' (Tab. 9.2).

The special interaction term $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4 that performs ^3J*-coupling constant restraining* reads

$$
\mathcal{V}^{(Jr)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N_{Jr}} \mathcal{V}^{(Jr)}{}_n(J(\zeta_n); k_n^{(Jr)}, J^0{}_n)
\tag{9.63}
$$

where the dihedral angle $\zeta(m-j-k-m')$ is denoted by ζ_n, the reference $^3J^0{}_{mm'}$ value by $J^0{}_n$ and the superscript 3 has been dropped from the J.

In the case of experimentally measured ^3J-couplings, which are averages over space and time, it is preferable to apply the restraint to the time-averaged ^3J-couplings. The special interaction term then becomes

$$
\mathcal{V}^{(Jr)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N_{Jr}} \mathcal{V}^{(Jr)}{}_n(\overline{J(\zeta_n)}; k_n^{(Jr)}, J^0{}_n)
\tag{9.64}
$$

where the horizontal bar denotes a time-average.

The functional form of ${\mathcal{V}^{(Jr)}}_n$ may be harmonic, half-harmonic attractive, half-harmonic repulsive, biquadratic or periodically scaled. Whilst using a single half-harmonic potential energy term makes little sense in the case of a periodic structural feature such as a dihedral angle, two half-harmonic potential energy terms may be combined to form a full harmonic potential energy term that is asymmetric with respect to ${J^0}_n$ (in contrast to standard flat-bottomed restraining, see Eqs. 9.75 and 9.76).

For instantaneous restraining, the harmonic form is defined as

$$ {\mathcal{V}^{(Jr)}}_n(J(\zeta_n);k_n^{(Jr)},{J^0}_n) = \frac{1}{2}k_n^{(Jr)}\left[J(\zeta_n)-{J^0}_n\right]^2, \tag{9.65} $$

the half-harmonic attractive form as

$$ \begin{aligned} {\mathcal{V}^{(Jr)}}_n(J(\zeta_n);k_n^{(Jr)},{J^0}_n) &= 0 & J(\zeta_n)<{J^0}_n \\ &= \frac{1}{2}k_n^{(Jr)}\left[J(\zeta_n)-{J^0}_n\right]^2 & J(\zeta_n)>{J^0}_n, \end{aligned} \tag{9.66} $$

and the half-harmonic repulsive form as

$$ \begin{aligned} {\mathcal{V}^{(Jr)}}_n(J(\zeta_n);k_n^{(Jr)},{J^0}_n) &= \frac{1}{2}k_n^{(Jr)}\left[J(\zeta_n)-{J^0}_n\right]^2 & J(\zeta_n)<{J^0}_n \\ &= 0 & J(\zeta_n)>{J^0}_n. \end{aligned} \tag{9.67} $$

For standard time-averaging, the harmonic form is

$$ {\mathcal{V}^{(Jr)}}_n(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n) = \frac{1}{2}k_n^{(Jr)}\left[\,\overline{J(\zeta_n)}-{J^0}_n\right]^2 \tag{9.68} $$

and the half-harmonic forms are easily derived from Eqs. 9.66 and 9.67.

A further option with time-averaging is to use a biquadratic penalty function[63]

$$ {\mathcal{V}^{(Jr)}}_n(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n) = \frac{1}{2}k_n^{(Jr)}\left[\,J(\zeta_n)-{J^0}_n\right]^2\cdot\left[\,\overline{J(\zeta_n)}-{J^0}_n\right]^2. \tag{9.69} $$

This functional form avoids the large structural fluctuations that occur when standard time-averaging is used with ${}^3J$-value restraints.[64,65]

When using time-averaging, it is possible to periodically scale the potential energy function for the dihedral angles related to the ${}^3J$-couplings and the ${}^3J$-coupling restraint itself by the introduction of an oscillating factor $\cos^2(\omega^{Jr}t)$. During the scaling period, $\tau_{Jr}^s=\frac{180^\circ}{\omega^{Jr}}$, the restraining function is given by[66]

$$ \begin{aligned} {\mathcal{V}^{(Jr)}}_n(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n) &= \cos^2(\omega^{Jr}t)\cdot \\ &\quad \left[V^{(\varphi)}(\varphi_n;k^{(\varphi)},\varphi_n^0)+{\mathcal{V}^{(Jr)}}_n(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n)\right]. \end{aligned} \tag{9.70} $$

and $V^{(\varphi)}\ (\varphi_n\ ;\ k^{(\varphi)}\ ,\ \varphi_n^0\ )$ is taken out of the sum in Eq. 5.18.

The oscillating factor is switched on as soon as the average ${}^3J$-value deviates more than a certain threshold (ΔJ^0, see below) from the target value. After one oscillating period, τ_{Jr}^s, is completed, the scaling is suspended for a time period Δt_ω. This allows the system to deviate from the reference value for some time, which may be useful to overcome the degeneracy of the Karplus curve, as in the following example. Consider the Karplus curve in Fig. 9.2 and assume that a reference ${J^0}_n$ value of 9 Hz is used. For dihedral angle values around $\phi=60^\circ$, the restraint may get stuck in the local minimum with maximum ${}^3J$ values of 7 Hz.

A further possibility is to use local-elevation (LE) biasing rather than restraining the ${}^3J$-couplings. In this case, ${\mathcal{V}^{(Jr)}}_n$ is a sum of N_{le} LE terms

$$ {\mathcal{V}^{(Jr)}}_n(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n) = \sum_{i=1}^{N_{le}}{\mathcal{V}^{(Jr)}}_{ni}(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n). \tag{9.71} $$

Only the Gaussian functional form is currently implemented, giving

$$ {\mathcal{V}^{(Jr)}}_{ni}(\overline{J(\zeta_n)};k_n^{(Jr)},{J^0}_n) = k_n^{(Jr)}w_{\zeta ni}\exp\left(-(\zeta_n-{\zeta_{ni}}^0)^2/2(\Delta\zeta^0)^2\right). \tag{9.72} $$

In Eq. 9.72, $w_{\zeta ni}$ is the weight of the ith penalty term, the centres ${\zeta_{ni}}^0$ of the Gaussian functions ${\mathcal{V}^{(Jr)}}_{ni}$ are equally distributed over the range of possible values of ζ_n (${\zeta_{ni}}^0 = 360°i/N_{le}$ with $i = 1, \ldots, N_{le}$) and the width is given by $\Delta\zeta^0 = 360°/N_{le}$.

The weight, $w_{\zeta ni}$, is calculated according to

$$w_{\zeta ni} = t^{-1} \int_0^t \delta_{\zeta_n {\zeta_n}^0} \Big[\overline{J(\zeta_n)} - {J^0}_n \Big]^2 \tag{9.73}$$

for time-averaging and

$$w_{\zeta ni} = t^{-1} \int_0^t \delta_{\zeta_n {\zeta_n}^0} \big[J(\zeta_n) - {J^0}_n \big]^2 \cdot \Big[\overline{J(\zeta_n)} - {J^0}_n \Big]^2 \tag{9.74}$$

for biquadratic time-averaging. The Kronecker delta, $\delta_{\zeta_n {\zeta_n}^0}$, is defined using finite differences

$$\delta_{\zeta_n {\zeta_n}^0} = \begin{array}{ll} 1 & \text{if } {\zeta_{ni}}^0 - \Delta\zeta^0/2 \le \zeta_n < {\zeta_{ni}}^0 + \Delta\zeta^0/2 \\ 0 & \text{otherwise.} \end{array}$$

It is often desirable to allow for some uncertainty in the reference ${}^3J$-values, ${J^0}_n$. This can be done by using flat-bottomed restraining, where the restraint penalty function is only applied if the calculated ${}^3J$-values deviate by more than a given value ΔJ^0 from ${J^0}_n$. The instantaneous contribution to the special interaction function (Eqs. 9.63, 9.65, 9.66 and 9.67) then depends on

$$\begin{array}{ll} [J(\zeta_n) - {J^0}_n - \Delta J^0]^2 & \text{for } J(\zeta_n) > {J^0}_n + \Delta J^0 \\ [J(\zeta_n) - {J^0}_n + \Delta J^0]^2 & \text{for } J(\zeta_n) < {J^0}_n - \Delta J^0 \\ 0 & \text{otherwise} \end{array} \tag{9.75}$$

and, likewise, the time-averaged factor (Eqs. 9.64 and 9.67) depends on

$$\begin{array}{ll} [\overline{J(\zeta_n)} - {J^0}_n - \Delta J^0]^2 & \text{for } \overline{J(\zeta_n)} > {J^0}_n + \Delta J^0 \\ [\overline{J(\zeta_n)} - {J^0}_n + \Delta J^0]^2 & \text{for } \overline{J(\zeta_n)} < {J^0}_n - \Delta J^0 \\ 0 & \text{otherwise.} \end{array} \tag{9.76}$$

A slight complication arises when one or both of the atoms m and m' defining the angle $\zeta_n(m-j-k-m')$ are not explicitly treated in the simulation. This is, for example, the case for the GROMOS force fields, in which most hydrogen atoms that are attached to carbon atoms are not explicitly treated, but are instead incorporated into the carbons, forming united atoms (Sec. 3.2). In such a case, when atom m is incorporated into atom j or atom m' is incorporated into atom k, the dihedral angle $\zeta_n(m-j-k-m')$ is not defined in terms of atomic coordinates of real atoms. However, if another (real, non-H) atom i is bound to atom j, or another (real, non-H) atom l is bound to atom k, the dihedral angle $\zeta_n(m-j-k-m')$ can be related to the dihedral angle $\eta_n(i-j-k-l)$ by the relation

$$\zeta_n(m-j-k-m') = \eta_n(i-j-k-l) + \delta_n \tag{9.77}$$

where δ_n is a phase shift. Examples of the relation Eq. 9.77 are given in Tab. 9.2 and a Karplus curve is shown in Fig. 9.2.

The true time-averaged ${}^3J$-coupling constant is defined as a trajectory average

$$\begin{aligned} \overline{J_n(t)} &\equiv t^{-1} \int_0^t J_n(t')dt' \\ &= a\ \overline{\cos^2(\eta_n(t) + \delta_n)} + b\ \overline{\cos(\eta_n(t) + \delta_n)} + c. \end{aligned} \tag{9.78}$$

As for distance restraining, the true average (Eq. 9.78) is not suitable for use in Eq. 9.64 to derive the forces on atoms i, j, k and l. A characteristic decay time or memory relaxation time τ_{Jr} is therefore introduced, so that[64]

$$\overline{J_n(t;\tau_{Jr})} \equiv \frac{\int_0^t \exp(-(t-t')/\tau_{Jr}) J_n(t')dt'}{[\tau_{Jr}[1 - \exp(-t/\tau_{Jr})]]}$$

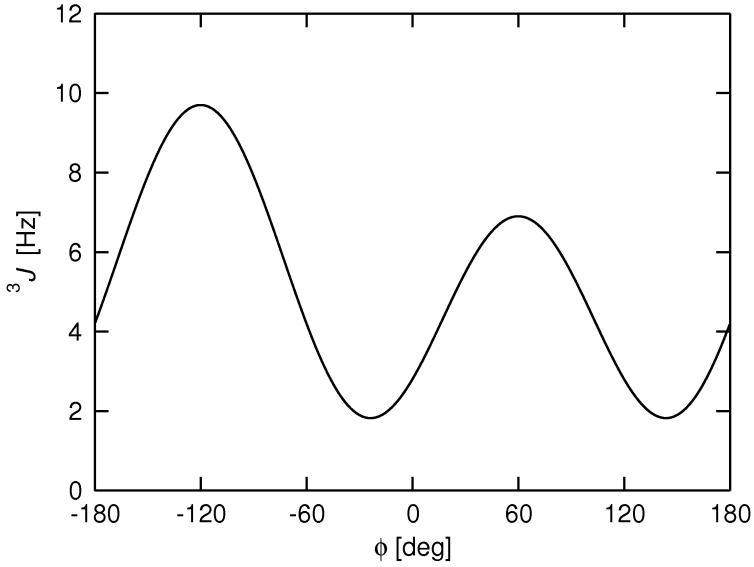


FIGURE 9.2. The Karplus curve (Eq. 9.62) for the angle ζ_n (H-N-C_α-H_α) given as a function of the angle $\eta_n = \phi$(C-N-C_α-C) (Eq. 9.77). The phase shift $\delta_n = \zeta_n - \eta_n = -60°$ for an L-amino acid has been applied and the calibration of Pardi *et al.*[67], with $a = 6.4$ Hz, $b = -1.4$ Hz and $c = 1.9$ Hz, was used.

ζ_n	η_n	δ_n (degrees)	a (Hz)	b (Hz)	c (Hz)
H - N - C_α - H_α[67]	C - N - C_α - C	- 60 (L) +60 (D)	6.4	-1.4	1.9
H_α - C_α - C_β - $H_{\beta 2}$[68]	N- C_α - C_β - C_γ	-120 (L) 0 (D)	9.5	-1.6	1.8
H_α - C_α - C_β - $H_{\beta 3}$[68]	N - C_α - C_β - C_γ	0 (L) +120 (D)	9.5	-1.6	1.8
N - C_α - C_β - $H_{\beta 2}$[69]	N - C_α - C_β - C_γ	120 (L) -120 (D)	4.4	-1.2	-0.1
N - C_α - C_β - $H_{\beta 3}$[69]	N - C_α - C_β - C_γ	-120 (L) 0 (D)	4.4	-1.2	-0.1
H - N - C_α- C_β[70]	C - N - C_α - C	+60 (L) - 60 (D)	4.7	-1.5	-0.2
H - N - C_α- C[70]	C - N - C_α- C	-180 (L,D)	5.7	-2.7	0.1

TABLE 9.2. Relations between ^3J-coupling constants and dihedral angles $\zeta_n = \eta_n + \delta_n$ occurring in polypeptides. The dihedral angles and atom names are defined according to the IUPAC-IUB convention.[25] L- and D-amino acid residues are indicated by L and D respectively.

$$= \quad a\ \overline{\cos^2(\eta_n + \delta_n)(t;\tau_{Jr})} + b\ \overline{\cos(\eta_n + \delta_n)(t;\tau_{Jr})} + c. \tag{9.79}$$

The time averages of $\cos^m(\eta_n(t) + \delta_n)$ in Eq. 9.79 are easily obtained in a simulation using discrete time steps Δt. When

$$t \quad \ll \quad \tau_{Jr} \tag{9.80}$$

we have

$$\begin{aligned} \overline{\cos^m(\eta_n + \delta_n)(t;\tau_{Jr})} \quad &= \quad [1 - \exp(-\Delta t/\tau_{Jr})]\cos^m(\eta_n(t) + \delta_n) \\ &\quad + \exp(-\Delta t/\tau_{Jr})\ \overline{\cos^m(\eta_n + \delta_n)(t - \Delta t;\tau_{Jr})}. \end{aligned} \tag{9.81}$$

The force on an atom i due to an instantaneous ^3J-value restraint in Eq. 9.63 can be written as

$$\mathbf{f}^{inst} \quad = \quad -\frac{\partial \mathcal{V}^{(Jr)}{}_n(J(\zeta_n); k_n^{(Jr)}, J^0{}_n)}{\partial J} \cdot \frac{\partial J_n}{\partial \eta_n} \cdot \frac{\partial \eta_n}{\partial \mathbf{r}_i}. \tag{9.82}$$

where the first factor in Eq. 9.82 is

$$\frac{\partial \mathcal{V}^{(Jr)}{}_n}{\partial J_n} \quad = \quad k_n^{(Jr)} \; [J_n \; - \; J^0{}_n]. \tag{9.83}$$

for the harmonic function (Eq. 9.65) and corresponding expressions are obtained for the half-harmonic cases (Eq. 9.66 and Eq. 9.67).

Using the relation

$$cos(\eta_n + \delta_n) \quad = \quad cos\eta_n cos\delta_n \; - \; sin\eta_n sin\delta_n \tag{9.84}$$

we find for the second factor in Eq. 9.82

$$\begin{aligned} \frac{\partial J_n}{\partial \eta_n} \quad &= \quad (2a \; cos(\eta_n + \delta_n) + b) \cdot (-sin(\eta_n + \delta_n)) \\ &= \quad (2a \; (cos\eta_n cos\delta_n \; - \; sin\eta_n sin\delta_n) + b) \cdot (-sin\eta_n cos\delta_n - cos\eta_n sin\delta_n). \end{aligned} \tag{9.85}$$

The third factor in Eq. 9.82, the derivative of the angle η_n (i-j-k-l) with respect to the positions $\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k$ and $\mathbf{r}_l$ can be obtained from the expressions and definitions in Sec. 17.4 with the angle ξ_n replaced by η_n.

$$\frac{\partial \eta_n}{\partial \mathbf{r}_i} \quad = \quad \frac{r_{kj}}{r_{mj}^2} \mathbf{r}_{mj} \quad , \tag{9.86}$$

$$\frac{\partial \eta_n}{\partial \mathbf{r}_l} \quad = \quad -\frac{r_{kj}}{r_{nk}^2} \mathbf{r}_{nk} \quad , \tag{9.87}$$

$$\frac{\partial \eta_n}{\partial \mathbf{r}_k} \quad = \quad -\frac{\partial \eta_n}{\partial r_i} - \frac{\partial \eta_n}{\partial \mathbf{r}_j} - \frac{\partial \eta_n}{\partial \mathbf{r}_l} \quad , \tag{9.88}$$

$$\frac{\partial \eta_n}{\partial \mathbf{r}_j} \quad = \quad \left[\; \frac{(\mathbf{r}_{ij} \cdot \mathbf{r}_{kj})}{\mathbf{r}_{kj}^2} - 1 \right] \frac{\partial \eta_n}{\partial \mathbf{r}_i} - \frac{(\mathbf{r}_{kl} \cdot \mathbf{r}_{kj})}{r_{kj}^2} \frac{\partial \eta_n}{\partial \mathbf{r}_l} \quad . \tag{9.89}$$

For time averaged ${}^3J$-value restraints we use the short-hand notation

$$\overline{J_n} \quad = \quad \overline{J_n(t; \tau_{Jr})}, \tag{9.90}$$

and find the force on atom i due to $\mathcal{V}^{(Jr)}{}_n$ in Eq. 9.64 at time t to be

$$\mathbf{f}_i \quad = \quad -\frac{\partial \mathcal{V}^{(Jr)}{}_n(\overline{J_n}; k_n^{(Jr)}, J^0{}_n)}{\partial \overline{J_n}} \cdot \frac{\partial \overline{J_n}}{\partial \eta_n} \cdot \frac{\partial \eta_n}{\partial \mathbf{r}_i}. \tag{9.91}$$

The first factor in Eq. 9.91 is

$$\frac{\partial \mathcal{V}^{(Jr)}{}_n}{\partial \overline{J_n}} \quad = \quad k_n^{(Jr)} \; \left[\; \overline{J_n} - J^0{}_n \right] \tag{9.92}$$

for the harmonic function (Eq. 9.65) and corresponding expressions are obtained for the half-harmonic cases (Eqs. 9.66 and 9.67).

Using the relation Eq. 9.84

$$\cos(\eta_n + \delta_n) \quad = \quad \cos\eta_n \cos\delta_n - \sin\eta_n \sin\delta_n \tag{9.93}$$

we find for the second factor in Eq. 9.91

$$\begin{aligned} \frac{\partial \overline{J_n}}{\partial \eta_n} \quad &= \quad [1 - \exp(-\Delta t/\tau_{Jr})] \, [2a \cos(\eta_n + \delta_n) + b] \, [-\sin(\eta_n + \delta_n)] \\ &= \quad [1 - \exp(-\Delta t/\tau_{Jr})] \cdot [2a[\cos\eta_n \cos\delta_n - \sin\eta_n \sin\delta_n] + b] \cdot \end{aligned}$$

$$[-\sin\eta_n\cos\delta_n - \cos\eta_n\sin\delta_n]. \tag{9.94}$$

As for distance restraining, the factor $[1-\exp(\Delta t/\tau_{Jr})]$ may be set to one when inserting Eq. 9.94 into Eq. 9.91. This may be done for practical reasons: to avoid having to choose the values for $k_n^{(Jr)}$ proportional to $[1-\exp(\Delta t/\tau_{Jr})]^{-1}$.

When using the biquadratic functional form the force on atom i becomes the sum of equations Eq. 9.82 and Eq. 9.91, where

$$\mathbf{f}_i^{bi} = \mathbf{f}_i^{tav} + \mathbf{f}_i^{inst} \tag{9.95}$$

And

$$\frac{\partial \mathcal{V}^{(Jr)}{}_n}{\partial J_n} = k_n^{(Jr)}\,(\overline{J_n} - J^0{}_n)^2(J_n - J^0{}_n). \tag{9.96}$$

and

$$\frac{\partial \mathcal{V}^{(Jr)}{}_n}{\partial \overline{J_n}} = k_n^{(Jr)}\left[\overline{J_n} - J^0{}_n\right](J_n - J^0{}_n)^2. \tag{9.97}$$

As for normal time averaging the term on the right hand side of (Eq. 9.97) is multiplied with the factor $[1-\exp(\Delta t/\tau_{Jr})]$. Alternatively this factor can be set to one or to zero, respectively. The latter option means that there will be no contribution of (Eq. 9.97) to the restraining force.

We note that the total energy will not be conserved when applying time-averaging due to the dependence of the restraining interaction (Eq. 9.64) on time points before time t.[64]

At the start of a simulation the value of the average (Eq. 9.79) is set equal to the ^3J-value calculated from the starting configuration

$$\begin{aligned}\overline{J_n(t=0;\tau_{Jr})} &= J_n(t=0)\\ &= a\cos^2(\eta_n(t=0)+\delta_n) + b\cos(\eta_n(t=0)+\delta_n) + c.\end{aligned} \tag{9.98}$$

At the end of a simulation the values of the averages (Eq. 9.79) are stored with the final configuration for use in a continuation simulation (Sec. 4-4.11).

The type of restraining applied to the ^3J-couplings (NTJVR: instantaneous, time-averaged or biquadratic time-averaged or with local-elevation (LE) (only with time-averaging)), whether or not this is a continuation run (NTJVRA), the value of the overall force constant, $k_n^{(Jr)}$ = CJVR, the coupling time, τ_{Jr} = TAUJVR, the option to omit the factor $[1-\exp(\Delta t/\tau_{Jr})]$, i.e. set it to one, in case of normal time-averaging (NJVRTARS), the option to weight the contribution of (Eq. 9.97) to the force in case of biquadratic restraining (NJVRBIQW), the number of grid points for the local-elevation potential (NGRID), the tolerance to deviation from the experimental value, ΔJ^0 (DELTA) and whether to write the ^3J-value data to a special trajectory are specified in the JVALUERES block of the MD input file (see Chap. 4-8).

The use of periodic scaling is controlled by setting RESTYPE=jrest in the PERSCALE block of the MD input file (see Chap. 4-8). Within this block, the maximum scaling factor for the dihedral angle potential energy term (KDIH) and for the ^3J-value restraint potential energy term (KJ), the period τ_{Jr}^s of the cosine scaling function (T), the minimum deviation ΔJ^0 from the target value to start a scaling period (DIFF), the minimum fraction of τ_{Jr}^s that needs to be passed before starting a new scaling period (RATIO) and the reading of the scaling parameters (READ) can be specified. An example of ^3J-coupling constant restraining using this method is given in[66].

An application of ^3J-coupling constant restraining using time-averaging, including a discussion of the choice of the values for CJVR and TAUJVR, is given in[64]. The latter should satisfy the condition

$$\tau_{Jr} \ll t_{\mathrm{MD}} \tag{9.99}$$

where t_{MD} is the length of the simulation. The overall force constant CJVR can be multiplied by individual weight factors WJVR specific to each ^3J-coupling constant (see below).

The choices for NGRID, DELTA and TAUJVR when local-elevation is used are discussed in[71], which contains an example of ^3J-value restraining using time-averaging and local-elevation.

The atom sequence numbers of the atoms i, j, k and l that define η_n (IPJV, JPJV, KPJV, LPJV [1..NDJV]), the individual restraint weights (WJVR[1..NDJV]), the reference ^3J-values $J^0{}_n$ (PJR0[1..NDJV]), the phase shifts $\delta_n = \zeta_n - \eta_n$ (PSJR[1..NDJV]) the Karplus parameters a, b and c (AJV, BJV, CJV[1..NDJV]) belonging

to the angles ζ_n and the type of restraining potential energy term (NHJV: harmonic, half-harmonic attractive or half-harmonic repulsive) are specified in the JVALRESSPEC block of a ^3J-coupling constant restraints file (see Sec. 4-3.6).

9.8. S^2-order parameter restraining

The special interaction term $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4 that performs S^2-*order parameter restraining* reads[72]

$$\mathcal{V}^{(Sr)}(\boldsymbol{r};\mathbf{s}) \;=\; \sum_{n=1}^{N_{Sr}} \mathcal{V}^{(Sr)}{}_n(\overline{S(\mathbf{r}_{XY})};k_n^{(Sr)},S_n^0,\Delta S_n^0) \tag{9.100}$$

where the horizontal bar denotes a time-average and the n-th S^2-order parameter restraint restrains the motion of atoms X and Y. The summation runs over a set of N_{Sr} S^2-order parameter restraints. The functional form of $\mathcal{V}^{(Sr)}$ reads

$$\begin{aligned}
&\mathcal{V}^{(Sr)}{}_n(\overline{S(\mathbf{r}_{XY})};k_n^{(Sr)},S_n^0,\Delta S_n^0)\\
&= \tfrac{1}{2}k_n^{(Sr)}[\overline{S(\mathbf{r}_{XY})}-S_n^0-\Delta S_n^0]^2 \quad \text{for} \quad \overline{S(\mathbf{r}_{XY})} > S_n^0+\Delta S_n^0\\
&= \tfrac{1}{2}k_n^{(Sr)}[\overline{S(\mathbf{r}_{XY})}-S_n^0+\Delta S_n^0]^2 \quad \text{for} \quad \overline{S(\mathbf{r}_{XY})} < S_n^0-\Delta S_n^0\\
&= 0 \quad \text{otherwise.}
\end{aligned} \tag{9.101}$$

where the value ΔS_n^0 allows for some uncertainty in the reference S_n^0 -values. The time-averaged order parameter is calculated from[72,73]

$$\overline{S(\mathbf{r}_{XY}(t))} = \frac{1}{2}\left\{3\sum_{\alpha=1}^{3}\sum_{\beta=1}^{3}\left[\overline{Q_{\alpha\beta}(t)}\right]^2 - \left[\overline{D(t)}\right]^2\right\}\cdot(r_{\mathrm{XY}}^{\mathrm{eff}})^6 \tag{9.102}$$

where $r_{\mathrm{XY}}^{\mathrm{eff}}$ is an effective internuclear distance between atoms X and Y. The time averaged quantities $\overline{Q_{\alpha\beta}(t)}$ and $\overline{D(t)}$ are calculated in the usual damped memory manner[58] with the memory relaxation time τ_{sr},

$$\overline{Q_{\alpha\beta}(t)} = \frac{1}{\tau_{\mathrm{sr}}[1-e^{-t/\tau_{\mathrm{sr}}}]}\int_0^t e^{-(t-t')/\tau_{\mathrm{sr}}}Q_{\alpha\beta}(t')dt' \tag{9.103}$$

and

$$\overline{D(t)} = \frac{1}{\tau_{\mathrm{sr}}[1-e^{-t/\tau_{\mathrm{sr}}}]}\int_0^t e^{-(t-t')/\tau_{\mathrm{sr}}}D(t')dt' \tag{9.104}$$

and

$$Q_{\alpha\beta}(t') = \frac{(r_{\mathrm{X}\alpha}(t')-r_{\mathrm{Y}\alpha}(t'))(r_{\mathrm{X}\beta}(t')-r_{\mathrm{Y}\beta}(t'))}{(r_{\mathrm{XY}}(t'))^5} \tag{9.105}$$

and

$$D(t') = \frac{1}{r_{\mathrm{XY}}(t')^3} \tag{9.106}$$

with

$$\mathbf{r}_{XY} = \mathbf{r}_X - \mathbf{r}_Y \quad \text{and} \quad r_{\mathrm{XY}} = [(\mathbf{r}_X-\mathbf{r}_Y)\cdot(\mathbf{r}_X-\mathbf{r}_Y)]^{1/2} \tag{9.107}$$

and

$$\begin{aligned}
r_{\mathrm{X1}} &= \mathrm{x-component\ of\ vector}\ \mathbf{r}_X\\
r_{\mathrm{X2}} &= \mathrm{y-component\ of\ vector}\ \mathbf{r}_X\\
r_{\mathrm{X3}} &= \mathrm{z-component\ of\ vector}\ \mathbf{r}_X
\end{aligned} \tag{9.108}$$

and likewise for $\mathbf{r}_Y$. The discretized form, applicable to atomic trajectories, in which configurations are separated by a time interval Δt, is

$$Q_{\alpha\beta}(m\Delta t) = \frac{(r_{\mathrm{X}\alpha}(m\Delta t) - r_{\mathrm{Y}\alpha}(m\Delta t))(r_{\mathrm{X}\beta}(m\Delta t) - r_{\mathrm{Y}\beta}(m\Delta t))}{(r_{\mathrm{XY}}(m\Delta t))^5} \tag{9.109}$$

and

$$\overline{Q_{\alpha\beta}(n\Delta t)} = Q_{\alpha\beta}(n\Delta t)\left\{1 - e^{-\Delta t/\tau_{\mathrm{sr}}}\right\} + e^{-\Delta t/\tau_{\mathrm{sr}}}\overline{Q_{\alpha\beta}((n-1)\Delta t)} \tag{9.110}$$

and

$$\overline{S^2_{\mathrm{XY}}(n\Delta t)} = \frac{1}{2}\left\{3\sum_{\alpha=1}^{3}\sum_{\beta=1}^{3}\left[\overline{Q_{\alpha\beta}(n\Delta t)}\right]^2 - \left[\overline{D(n\Delta t)}\right]^2\right\}\cdot (r^{\mathrm{eff}}_{\mathrm{XY}})^6 \tag{9.111}$$

with

$$D(m\Delta t) = \frac{1}{r_{\mathrm{XY}}(m\Delta t)^3} \tag{9.112}$$

and

$$\overline{D(n\Delta t)} = D(n\Delta t)\left\{1 - e^{-\Delta t/\tau_{\mathrm{sr}}}\right\} + e^{-\Delta t/\tau_{\mathrm{sr}}}\overline{D((n-1)\Delta t)}. \tag{9.113}$$

The restraining force on atom X then becomes

$$\begin{aligned}\mathbf{f}_X(t) &= -\frac{\partial V^{\mathrm{restr}}(\vec{r}^N(t))}{\partial \vec{r}_{\mathrm{X}}(t)} \\ &= -K^{\mathrm{sr}}\left[\overline{S^2_{\mathrm{XY}}(\vec{r}^N(t))} - S^2_{\mathrm{XY}}(\mathrm{exp})\right] \\ &\quad \cdot\frac{1}{2}\left\{3\sum_{\alpha=1}^{3}\sum_{\beta=1}^{3}2\overline{Q_{\alpha\beta}(t)}\frac{\partial\overline{Q_{\alpha\beta}(t)}}{\partial Q_{\alpha\beta}(t)}\frac{\partial Q_{\alpha\beta}(t)}{\partial \vec{r}_{\mathrm{X}}(t)} - 2\overline{D(t)}\frac{\partial\overline{D(t)}}{\partial D(t)}\frac{\partial D(t)}{\partial \vec{r}_{\mathrm{X}}(t)}\right\}\cdot (r^{\mathrm{eff}}_{\mathrm{XY}})^6\end{aligned} \tag{9.114}$$

and the restraining force on atom Y becomes

$$\begin{aligned}\mathbf{f}_Y(t) &= -\frac{\partial V^{\mathrm{restr}}(\vec{r}^N(t))}{\partial \vec{r}_{\mathrm{Y}}(t)} \\ &= -K^{\mathrm{sr}}\left[\overline{S^2_{\mathrm{XY}}(\vec{r}^N(t))} - S^2_{\mathrm{XY}}(\mathrm{exp})\right] \\ &\quad \cdot\frac{1}{2}\left\{3\sum_{\alpha=1}^{3}\sum_{\beta=1}^{3}2\overline{Q_{\alpha\beta}(t)}\frac{\partial\overline{Q_{\alpha\beta}(t)}}{\partial Q_{\alpha\beta}(t)}\frac{\partial Q_{\alpha\beta}(t)}{\partial \vec{r}_{\mathrm{Y}}(t)} - 2\overline{D(t)}\frac{\partial\overline{D(t)}}{\partial D(t)}\frac{\partial D(t)}{\partial \vec{r}_{\mathrm{Y}}(t)}\right\}\cdot (r^{\mathrm{eff}}_{\mathrm{XY}})^6\end{aligned} \tag{9.115}$$

although using 9.110 we have

$$\frac{\partial\overline{Q_{\alpha\beta}(t)}}{\partial Q_{\alpha\beta}(t)} = \frac{\partial\overline{Q_{\alpha\beta}(n\Delta t)}}{\partial Q_{\alpha\beta}(n\Delta t)} = \left[1 - e^{-\Delta t/\tau_{\mathrm{sr}}}\right] \tag{9.116}$$

the approximation[63]

$$\frac{\partial\overline{Q_{\alpha\beta}(t)}}{\partial Q_{\alpha\beta}(t)} = 1 \tag{9.117}$$

is often used, which only leads to a rescaling of K^{sr} in practice, and likewise

$$\frac{\partial\overline{D(t)}}{\partial D(t)} = 1. \tag{9.118}$$

For the derivatives $\frac{\partial Q_{\alpha\beta}(t)}{\partial \vec{r}_{\mathrm{X}}(t)}$ we find using 9.105, where we omit the variable t and denote the three components of the position vector $\vec{r}_{\mathrm{X}}$ of atom X by $r_{\mathrm{X}\gamma}$ with $\gamma = 1, 2, 3$,

$$\begin{aligned}\frac{\partial Q_{\alpha\beta}}{\partial r_{\mathrm{X}_\gamma}} &= \frac{(r_{\mathrm{XY}})^5\left\{\delta_{\gamma\alpha}(r_{\mathrm{X}\beta} - r_{\mathrm{Y}\beta}) + \delta_{\gamma\beta}(r_{\mathrm{X}\alpha} - r_{\mathrm{Y}\alpha})\right\}}{(r_{\mathrm{XY}})^{10}} \\ &\quad -\frac{(r_{\mathrm{X}\alpha} - r_{\mathrm{Y}\alpha})(r_{\mathrm{X}\beta} - r_{\mathrm{Y}\beta})\cdot 5(r_{\mathrm{XY}})^4(r_{\mathrm{X}\gamma} - r_{\mathrm{Y}\gamma})(r_{\mathrm{XY}})^{-1}}{(r_{\mathrm{XY}})^{10}} \\ &= \frac{(r_{\mathrm{XY}})^2\left\{\delta_{\gamma\alpha}(r_{\mathrm{X}\beta} - r_{\mathrm{Y}\beta}) + \delta_{\gamma\beta}(r_{\mathrm{X}\alpha} - r_{\mathrm{Y}\alpha})\right\} - 5(r_{\mathrm{X}\alpha} - r_{\mathrm{Y}\alpha})(r_{\mathrm{X}\beta} - r_{\mathrm{Y}\beta})(r_{\mathrm{X}\gamma} - r_{\mathrm{Y}\gamma})}{(r_{\mathrm{XY}})^7}\end{aligned} \tag{9.119}$$

where δ_{ij} is the Kronecker delta. For the derivatives $\frac{\partial Q_{\alpha\beta}(t)}{\partial \vec{r}_{\rm Y}(t)}$ we find likewise

$$\frac{\partial Q_{\alpha\beta}}{\partial r_{{\rm Y}\gamma}} = -\frac{\partial Q_{\alpha\beta}}{\partial r_{{\rm X}\gamma}}. \tag{9.120}$$

Thus the restraining force on atom Y is the negative of the restraining force on atom X for this restraining function. For the derivatives $\frac{\partial D(t)}{\partial \vec{r}_{\rm X}(t)}$ we find using 9.106 likewise

$$\frac{\partial D}{\partial r_{{\rm X}\gamma}} = \frac{-3(r_{{\rm X}\gamma} - r_{{\rm Y}\gamma})}{(r_{\rm XY})^5} \tag{9.121}$$

and

$$\frac{\partial D}{\partial r_{{\rm X}\gamma}} = -\frac{\partial D}{\partial r_{{\rm Y}\gamma}}. \tag{9.122}$$

Note that the use of order parameters to bias MD simulations is conceptually different from the use of other properties such as nuclear Overhauser enhancement (NOE) atom-atom distance bounds or ^3J-coupling constants, because order parameters are not instantaneous observables, i.e. the order parameter for a single configuration is by definition equal to unity. Therefore, $\tau_{\rm sr}$ represents not only the memory relaxation but also the experimentally determined averaging period. Thus it should be chosen larger than the decay time of the internal autocorrelation function of the vector connecting the two atoms, but not larger than the sensitivity time window of the NMR experiment.

At the start of a simulation the values of the averages 9.103 and 9.104 are set to their instantaneous values calculated from the starting configuration. At the end of a simulation the values of the averages (9.103 and 9.104) are stored with the final configuration for use in a continuation simulation.

9.9. X-ray structure factor amplitude restraining

In GROMOS a five Gaussian parametrisation[74] of the atomic scattering factor f_i is used,

$$f_i(\mathbf{k}) = O_i \exp\left(-\frac{B_i}{4k^2}\right)\left[\sum_{j=1}^{5}\left[a_{ij}\exp\left(-\frac{b_{ij}}{4k^2}\right)\right] + c_i\right]. \tag{9.123}$$

Here, O_i and B_i are the atomic occupancy and the atomic B-factor, a, b and c are coefficients which depend on the type and charge of the atom i[74]. The atomic electron density ρ_i is given by the analytical Fourier backwards transform of the scattering factor f_i. The global electron density is then computed as the sum of the atomic densities

$$\rho(\mathbf{r}; \boldsymbol{r}\mathcal{N}_a) = \sum_{i=1}^{\mathcal{N}_a} \rho_i(\mathbf{r}; \boldsymbol{r}^{\mathcal{N}_a}) \tag{9.124}$$

on a grid whose resolution is an adjustable parameter. The structure factor is obtained by Fourier transform of the electron density of the system

$$F(\mathbf{k}; \boldsymbol{r}^{\mathcal{N}_a}) = \mathcal{F}\{\rho(\mathbf{r}; \boldsymbol{r}^{\mathcal{N}_a})\}. \tag{9.125}$$

The structure factors obtained by the procedure described above are denoted as the *calculated* structure factors. Their amplitudes $\mid F \mid$ can be compared to the observed structure-factor amplitudes $\mid F^0 \mid$, derived from the scattering intensities, through the R-factor which is given by

$$R = \frac{\sum_{i=1}^{N_F} \mid \mid F_i^0 \mid - s \mid F_i \mid \mid}{\sum_{i=1}^{N_F} \mid F_i^0 \mid}, \tag{9.126}$$

where N_F is the number of observed structure-factor amplitudes. As $\mid F^0 \mid$ is generally on an arbitrary scale, the calculated amplitudes are fitted to the observed ones by means of a weighted linear regression, leading to the scaling factor

$$s = \frac{\sum_{i=1}^{N_F} w_i \mid F_i^0 \mid\mid F_i \mid}{\sum_{i=1}^{N_F} w_i \mid F_i \mid^2}. \tag{9.127}$$

The weight w_i of an individual reflection is generally taken as the inverse of the variance of the structure-factor amplitude.

As for ^3J-couplings, a harmonic potential energy term for the structure-factor amplitude can be used to restrain the computed structure-factor amplitudes to the observed ones

$$\mathcal{V}^{(Fxr)}(\boldsymbol{r}^{N_F};\mid F^0 \mid) = \frac{1}{2}\frac{k^{xr}}{\sum_{i=1}^{N_F} w_i \mid F_i^0 \mid^2}\sum_{i=1}^{N_F} w_i \left(\mid F_i^0 \mid -s \mid F_i(\boldsymbol{r}^{N_F}) \mid\right)^2. \tag{9.128}$$

The weight factor or force constant k^{xr} is made resolution independent by the factor $\frac{1}{\sum_{i=1}^{N_F} w_i |F_i^0|^2}$. Time-averaging is introduced by substitution of $\mid F_i \mid$ by $\langle \mid F_i \mid \rangle_t$ in 9.128[75]. In addition to the instantaneous and time-averaging restraining term, a biquadratic term is available

$$\begin{aligned}\mathcal{V}^{(Fxr)}(\boldsymbol{r}^{N_F};\mid F^0 \mid) &= \frac{1}{2}\frac{k^{xr}}{\sum_{i=1}^{N_F} w_i \mid F_i^0 \mid^4} \\ &\sum_{i=1}^{N_F} w_i \left(\mid F_i^0 \mid -s_{\text{inst}} \mid F_i(\boldsymbol{r}^{N_F}) \mid\right)^2 \left(\mid F_i^0 \mid -s_{\text{avg}} \left\langle \mid F_i(\boldsymbol{r}^{N_F}) \mid \right\rangle_t\right)^2.\end{aligned} \tag{9.129}$$

We note that two individual scaling constants s_{inst} and s_{avg} are used. s_{inst} is computed using 9.127, while s_{avg} is calculated using the same relation, where the calculated amplitudes $\mid F_i \mid$ were replaced by the time-averaged ones $\langle \mid F_i \mid \rangle_t$.

9.10. X-ray electron density restraining

In X-ray crystallography, the electron density ρ^0 is not an experimentally observable quantity. As the phase of F^0 is generally not observable, it is taken from the structure factor F computed from an atomic model. The electron density ρ^0 is then computed as

$$\rho^0(\mathbf{r}) = \mathcal{F}^{-1}\{\left(2 \mid F^0(\mathbf{k}) \mid -s \mid F(\mathbf{k}) \mid\right) \exp\left[i \arg\left(F(\mathbf{k})\right)\right]\}, \tag{9.130}$$

where $\arg(F)$ is the phase computed from the model. In analogy to 9.126, a real-space R-factor[76] can be defined

$$R = \frac{\sum_{\mathbf{r}} \mid \beta\rho^0(\mathbf{r}) + \alpha - \rho(\mathbf{r}) \mid}{\sum_{\mathbf{r}} \mid \beta\rho^0(\mathbf{r}) + \alpha + \rho(\mathbf{r}) \mid}. \tag{9.131}$$

Summation is only carried out over the extent of the atoms of interest. α and β are scaling constants computed using a linear regression of

$$\rho(\mathbf{r}) = \beta\rho^0(\mathbf{r}) + \alpha. \tag{9.132}$$

These values can be calculated for an arbitrary set of atoms. Fitting is carried out using only the grid points lying within spheres with a radius R_{cut} centered at the positions of these atoms.

The electron density computed using 9.130 can be used for restraining. The penalty function[77] is defined as

$$\mathcal{V}^{(exr)}(\boldsymbol{r}^{\mathcal{N}_a};\rho^0) = \frac{1}{2}k^{xr}\sum_{\mathbf{r}}\left(\beta\rho^0 + \alpha - \rho(\boldsymbol{r}^{\mathcal{N}_a})\right)^2. \tag{9.133}$$

The summation is carried out over all grid points of the unit cell. Time-averaging is not available for electron density restraints, neither is a biquadratic penalty function.

9.11. X-ray crystallographic symmetry restraining

In a crystal, having a space group different from P1, multiple identical asymmetric units (ASUs) are assembled to construct the unit cell. The relationship between the ASUs is fully defined by the space group and the size of the unit cell. A space group S contains N_{sym} number of symmetry operations $\mathbb{S}_i$. Every atom position in the first asymmetric unit $\mathbf{r}_i$ has $N_{\text{sym}} - 1$ images $\mathbf{r}_i^{(j)}$ defined as

$$\mathbf{r}_i^{(j)} = \mathbb{S}_j\mathbf{r}_i. \tag{9.134}$$

Note that $\mathbb{S}_1$ is the identity symmetry operation and thus $\mathbf{r}_i^{(1)}$ and $\mathbf{r}_i$ are identical. Symmetry operations can be described in terms of a rotation followed by a translation,

$$\mathbb{S}_j\mathbf{r}_i = \mathbf{R}_j\mathbf{r}_i + \mathbf{t}_j. \tag{9.135}$$

Here, the rotation matrix $\mathbf{R}_j$ of the symmetry operation j is, in contrast to the translation vector $\mathbf{t}_j$, independent of the size of the unit cell. The inverse symmetry operation $symop^{-1}$ can be formulated as

$$\mathbb{S}_j{}^{-1}\mathbf{r}_i = \mathbf{R}_j{}^{-1}\left(\mathbf{r}_i - \mathbf{t}_j\right). \tag{9.136}$$

When simulating a unit cell, one may want to impose restraints on the symmetry of the system as, in many cases, insufficient experimental data for refinement of the whole unit cell is available. An easy way of achieving this is by adding harmonic potential energy terms for every symmetry-related pair. For every atom i in the first asymmetric unit, the term

$$\mathcal{V}^{(sxr)}\left(\mathbf{r}_i; k^{sym}\right) = \frac{1}{2}k^{sym}\sum_{i'=1}^{N_{\mathrm{sym}}-1}\sum_{j'=i'+1}^{N_{\mathrm{sym}}}\left[\mathbb{S}_{j'}{}^{-1}\mathbf{r}_i^{(j')} - \mathbb{S}_{i'}{}^{-1}\mathbf{r}_i^{(i')}\right]^2 \tag{9.137}$$

is added. The term can be interpreted as follows: all possible symmetry pairs are transformed to their position in the first asymmetric unit and a set of springs between corresponding atoms is introduced for each pair of molecules.

9.12. Distance-field distance restraining

The calculation of protein-ligand binding free energy energies is an important goal in the field of computational chemistry. Applying path-sampling methods for this purpose involves calculating the associated potential of mean force (PMF) and gives insight into the binding free energy along the binding process. Without a priori knowledge about the binding path, sampling reversible binding can be difficult to achieve. To alleviate this problem, the distance field (DF) has been introduced as reaction coordinate for such calculations.[78] DF is a grid-based method in which the shortest distance between the binding site and a ligand is determined avoiding routes that pass through the protein. Combining this reaction coordinate with Hamiltonian replica-exchange molecular dynamics (REMD) allows for the reversible binding of the ligand to the protein.

In the case of protein ligand binding, a DF restraint can be applied between (virtual) atom i and (virtual) atom j, representing the active site and the ligand, respectively. At the start of a simulation employing a DF restraint, a 3-dimensional grid is created with the grid spacing, g_s, as set by the user (input parameter GRID in the DISTANCEFIELD block). The minimal DF distance from a (virtual) atom i to each of the grid points is then assigned by applying Dijkstra's algorithm.[79]

1. The DF distance at every grid point is initialized to a large value that cannot be reached during the simulation. Here, we have chosen this value to be $4 \times a \times b \times c$ where a, b and c are the box edge lengths. The grid points that are positioned within a user-specified cutoff distance (PROTEINCUTOFF in block DISTANCEFIELD) of any protein atom are flagged by a large value (PROTEINOFFSET in block DISTANCEFIELD).
2. All grid points are marked as unvisited. The initial node is the grid point closest to (virtual) atom i. This grid point is assigned as the current node and a DF distance of 0 is assigned to it.
3. The unvisited nodes that are neighboring the current node are subsequently considered.
   - A new DF distance for the neighboring grid point is initially calculated as the DF distance assigned to the current node plus the distance between the two grid points (g_s). If a neighboring node is flagged as protein, an additional user-specified protein penalty is added to the distance.
   - When the new DF distance is less than the previously assigned preliminary distance, the latter is overwritten. Otherwise the distance does not change. In both cases, the neighbors are still marked 'unvisited' and the assigned DF distances are still preliminary.
4. When all neighbors are considered, the current node is marked as 'visited'. With this marking, the preliminary distance becomes final and this node will not be considered anymore.
5. The unvisited grid point with the lowest preliminary distance becomes the current node. Go back to step 3 and continue until all grid points are marked 'visited'.

Using this approach, periodic boundary conditions are taken fully into account and grid points are not revisited. Once the updating step is completed, all grid points are assigned the shortest DF distance from atom i. The DF distances are updated (by applying steps 1-5) every UPDATE steps (in block DISTANCEFIELD). Not updating every time step speeds up the simulation significantly and is allowed as long as the protein does not change its conformation too much between the updates, even though this will come at the expense of a slight loss of energy conservation. Using an UPDATE value of 100, the simulations are slowed down by 20 percent with respect to regular distance restraint simulations.

To avoid simulation artifacts, several extra features are implemented.

1. If (virtual) atom i is flagged as being within the protein, the DF distances of the grid points are not updated. It sometimes happens that this virtual atom temporarily gets buried in the active site due to the flexibility of the protein. If an update step would be performed, all DF distances are very large, because each of them has the additional protein penalty which was added to the first point. The forces induced by a DF restraint will thus get very large and may disrupt the structure. Also, because each grid point has the additional penalty, the DF distance converges to a normal radial distance with an initial offset.
2. The user can specify the number of smoothening rounds that are applied after each updating step (input parameter SMOOTH in the DISTANCEFIELD block). In a smoothening round, we loop over all non-protein grid points and check if one of its neighbors is flagged as protein. If this is the case, we are dealing with a grid point that is at the edge of the protein. In order to avoid large forces pointing away from the protein, the DF distance on the flagged protein grid point is determined again based on its direct neighbors, but now without the protein penalty. In this way, the large forces arising from the protein penalty on the grid are buried within the protein, and the regular van der Waals repulsion will ensure that the ligand never reaches these grid points. The DF distances of other grid points and the optimal route for atom j are not affected as this smoothening step is performed after the normal updating steps (1-5) have finished.

Once the DF distances are defined for all grid points, we can impose a DF restraint on the distance between atoms i and j. The forces and energies due to the DF restraint are calculated at each time step. We start this process by determining the 8 grid points that are closest to atom j, to which DF distances have been determined in the updating step. In order to be able to calculate the forces, the derivatives of the DF distance, l in the x, y and z directions have to be determined for each of the 8 neighboring grid points. This is done using a finite differences approach as illustrated in Eq. 9.138 for grid point k, in the x direction.

$$\frac{dl}{dx_k} = \frac{l(x_k - 1, y_k, z_k) - l(x_k + 1, y_k, z_k)}{2g_s} \tag{9.138}$$

Here, $l(x_k - 1, y_k, z_k)$ is the DF distance assigned to the neighbor of grid point k with the smaller x coordinate and g_s is the grid spacing. In order to interpolate the DF distance and its derivatives from the neighboring points to atom j, an assignment function of order 2 is used, similar to the one used for charges in the PPPM method (see Sec. 7.4.4.4).

The potential energy associated with the DF restraint, $\mathcal{V}^{(df)}$, can now be calculated using

$$\begin{aligned}
&\mathcal{V}^{(df)}(l_{ij}, k^{(df)}, l^0, \Delta l^h) \\
&= -k^{(df)}[l_{ij} - l^0 + \tfrac{1}{2}\Delta l^h]\Delta l^h && l_{ij} \leq l^0 - \Delta l^h \\
&= \tfrac{1}{2}k^{(df)}[l_{ij} - l^0]^2 && l^0 - \Delta l^h < l_{ij} \leq l^0 + \Delta l^h \\
&= k^{(df)}[l_{ij} - l^0 - \tfrac{1}{2}\Delta l^h]\Delta l^h && l_{ij} > l^0 + \Delta l^h
\end{aligned} \tag{9.139}$$

$k^{(df)}$ is the force constant of the harmonic DF restraint, l^0 is the reference DF distance and l_{ij} is the current DF distance between atoms i and j. The interaction term is linearized after a certain deviation Δl^h to prevent too large energy and forces for larger deviations (input parameter RL in the DISTANCEFIELD block). The forces on atom i and j are calculated with

$$\begin{aligned}
\mathbf{f}_j &= -\frac{\partial \mathcal{V}^{(df)}}{\partial l_{ij}} \frac{\partial l_{ij}}{\partial \mathbf{r}_j} \\
&= k^{(df)} \Delta l^h \frac{\partial l_{ij}}{\partial \mathbf{r}_j} && l_{ij} \leq l^0 - \Delta l^h
\end{aligned}$$

$$
\begin{aligned}
&= -k^{(df)}(l_{ij}-l^0)\frac{\partial l_{ij}}{\partial \mathbf{r}_j} \qquad l^0-\Delta l^h < l_{ij} \le l^0+\Delta l^h \\
&= -k^{(df)}\Delta l^h \frac{\partial l_{ij}}{\partial \mathbf{r}_j} \qquad\qquad\qquad\qquad l_{ij} > l^0+\Delta l^h
\end{aligned} \tag{9.140}
$$

and

$$
\mathbf{f}_i = -\mathbf{f}_j \tag{9.141}
$$

9.13. Biasing energy functions

The force field may be combined with a bias energy, so as to penalise or favour specified parts of conformational space. Such a bias energy may be time-dependent, such as in the local elevation method (Sec. 9.13.1) or time-independent, such as in the umbrella sampling method (Sec. 9.13.2) or may be a combination of these, such as in the local elevation umbrella sampling methods (Sec. 9.13.3 and Sec. 9.13.4). GROMOS allows a specification of a bias energy based on the following steps:

1. definition of a set of N_{LE} coordinates, collectively noted by the vector $\boldsymbol{Q} = \{\mathcal{Q}_n, n = 1..N_{LE}\}$, of which the bias energy will depend. The chosen set of coordinates will be named local elevation- or LE-coordinates. These coordinates are assumed to be well defined and differentiable functions of the vector $\boldsymbol{r}$ encompassing the Cartesian coordinates of all particles in the system, *i.e.* the vector function $\boldsymbol{Q} = \boldsymbol{Q}(\boldsymbol{r})$ must be defined for any $\boldsymbol{r}$ and its derivative must be non-singular. Examples of possible internal coordinates include *e.g.* distances between atom pairs, angles between atom triples, dihedral angles between atom quadruples, root-mean-square atomic positional deviations from given reference structures, extended-system variables (*e.g.* λ-variables in λ-dynamics[80–82]), or any (differentiable) mathematical combination of these. Currently implemented is dihedral angle [VARTYPE=1], distance [VARTYPE=2] and distance-field distance [VARTYPE=6].
2. a specification of one (or more) grid(s) within the space of the LE-coordinates consisting of $\Gamma+1$ grid points and a memory force-constant vector $\boldsymbol{M}_k = \{M_{k,i}, i = 0..\Gamma_k\}$ giving the force constants at each grid point. Three types of grid may be specified, (*i*) a full-dimensional grid, used in the LE and LEUS methods; (*ii*) a spherical grid and (*iii*) a line grid, the latter two used in the B&S − LEUS[83] method.
3. a set of basis functions γ to give a continuous interpolation between grid points. Various LE basis functions are available for the use in GROMOS, and a discussion of the different functions can be found in[84]. The LE basis function may be of Gaussian type

   $$
   \gamma^{Gauss}(x) = H(r_{cut}-|x|)\exp\Big[-\frac{(x)^2}{2\sigma^2}\Big] \tag{9.142}
   $$

   or of polynomial form with continuous 1st derivative ($NTLEFU = 0$)

   $$
   \gamma^{Poly_1}(x) = H(r_{cut}-|x|)[1-3\frac{|x|^2}{\sigma^2}+2\frac{|x|^3}{\sigma^3}] \tag{9.143}
   $$

   or with continuous second derivative

   $$
   \gamma^{Poly_2}(x) = H(r_{cut}-|x|)[1-10\frac{|x|^3}{\sigma^3}+15\frac{|x|^4}{\sigma^4}-6\frac{|x|^5}{\sigma^5}]\ , \tag{9.144}
   $$

   where σ describes the width of the basis functions $WLES$. For the polynomial forms σ equals the grid spacing of the defined grid.

9.13.1. Local elevation biasing. MD or SD simulation can also be used as a method to search the conformational space of a molecule for conformations of low energy. Due to the presence of relatively high barriers on the energy surface, MD and SD simulations have the tendency to repeatedly visit a small set of local energy minima. To avoid resampling of conformations, we may add a penalty potential energy to energetically penalise conformations already sampled. The local elevation method[85] offers a way to produce such a penalty potential energy by gradually adding small, local, repulsive potential energy terms (LE basis function γ) during the simulation. In practice, this is performed by increasing the values of the memory force-constant vector $\boldsymbol{M}$ according to the grid point visited at the current time, giving a time-dependent bias energy. The local elevation method can also be used in combination with time-averaging to bias a simulation towards conformations that, on average, satisfy a set of experimental data such as ^3J-couplings (Sec. 9.7).

If the number of LE coordinates is low ($N_{le} \approx\leq 5$), one may enhance the sampling in the complete LE-subspace. This can be done by using a full-dimensional grid (NTLES), where the special interaction term ($\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ in Eq. 3.4) defining the *local elevation bias* reads

$$\begin{aligned}\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s}) &= \mathcal{V}^{(le)}(\boldsymbol{Q};\boldsymbol{M}(t)) \\ &= \sum_{k=0}^{\Gamma} M_k(t) \prod_{i=1}^{N_{le}} \gamma_i(\boldsymbol{Q}_i - \boldsymbol{Q}^0_{k,i}) \end{aligned} \tag{9.145}$$

where the sum k goes over the $\Gamma + 1$ grid points (in practice over visited grid points), the product i runs over the N_{le} local elevation (LE) coordinates $\boldsymbol{Q}_i$, γ_i is the LE basis function and $\boldsymbol{M}(t)$ describes a (time-dependent) memory force-constant vector. In the LE method, the time dependence of the memory force-constant vector is simply given as

$$M_k(t) = M_k(t - \Delta t) + k^{(le)} \tag{9.146}$$

where $k^{(le)}$ is the LE force constant $CLES$ and $\boldsymbol{Q}_i(t)$ is within the range of grid point k. We note that the total energy will not be conserved when applying the LE interaction due to the change of the memory force-constant vector $\boldsymbol{M}(t)$ as a function of time[85]. Under specific conditions of the memory force-constant vector $\boldsymbol{M}(t)$, the bias potential energy generated by the LE procedure will, together with the physical potential energy, create an approximately flat free energy surface[86,87], thus we can in principle approximate the free energy surface as the negative of the LE bias potential energy. However, to obtain accurate free energies (and other ensemble properties), it is highly recommended to freeze the build up and use the LE bias potential energy as an (time-independent) umbrella potential energy term, as done in the LEUS method.

At the end of a simulation, the information concerning the definition of LE conformations, which LE conformations have been visited and how many times (ILEPOT), are stored with the final configuration for use in a continuation simulation.

Whether or not to apply LE (NTLES) and the build-up or freezing of each potential energy term (NTLEFR) are read from the LOCALELEV block of the MD input file. Parameters to describe the form of the potential energy term may be read from the LOCALELEV block (NTLESA=0, starting with $\mathcal{V}^{(le)} = 0$), or together with the potential energy from the coordinate file (NTLESA=1) or a LEUS database file (NTLESA=2). The atom sequence numbers defining the LE coordinate and the identifiers of the potential energy are given in the LOCALELEVSPEC block of the LE input file. An application of LE search is given in[85].

9.13.2. Umbrella sampling. According to the umbrella sampling method[88], one can recover the ensemble properties of an unbiased ensemble by means of reweighting as

$$A = \frac{< A(\boldsymbol{q},\boldsymbol{p_q}) \exp[\frac{\mathcal{V}^{(spec)}}{k_B T}] >_B}{< \exp[\frac{\mathcal{V}^{(spec)}}{k_B T}] >_B} \tag{9.147}$$

A suitable bias energy may therefore be specified, so as to calculate ensemble properties such as free energy, ^3J-couplings and occurrences of hydrogen bonds at a higher efficiency than a normal MD simulation. One example may be the application of an energy bias that cancels out the potential of mean force, giving uniform sampling in the space of the selected LE-coordinates. The potential of mean force is however in most cases unknown and is often the sought result of the simulation, and a manual specification of the bias potential energy term is therefore in most cases not possible.

9.13.3. Local elevation umbrella sampling (LEUS**).** As described in Sec. 9.13.1, an energy bias as built by the LE method will approximate the negative of the free energy surface. In practice however, the quality of the approximation is depending on the chosen build-up parameters and includes a certain amount of statistical uncertainty. To get accurate free energies and other ensemble properties one may either attempt to optimise the build-up procedure itself, or correct for the uncertainties by freezing the LE-build up and run an umbrella sampling procedure where the bias energy is time-independent. In practice, the second choice is simpler to accomplish, as uncertainties in the bias energy up to approximately $k_B T$ will be corrected for. This is the principle of the LEUS method: first a relatively short LE run is performed to create a biasing term that allows an approximate uniform sampling in the space of LE-coordinates and secondly a US run is performed with the LE bias energy term frozen to obtain accurate simulation results.

The build-up or freezing of the bias potential energy terms (NTLEFR) are read from the LOCALELEV block of the MD input file.

An application of LEUS is given in Ref.[89].

9.13.4. Ball and stick LEUS**.** In the case where the LE-space is of a high dimension, the application of a full-dimensional grid becomes unpractical. A solution to this problem for systems with well-defined conformational states is presented in the *ball-and-stick* local elevation umbrella sampling method (B&S − LEUS)[83], where sampling is restricted to and enhanced within a prespecified subspace of the chosen LE coordinates.

The B&S − LEUS method consists of seven steps.

1. Choice of a LE subspace permitting the definition of the relevant conformational states. Note that periodic internal coordinates (*e.g.* dihedral angles) should not be $''refolded''$ to a reference period, *i.e.* their time evolution must be continuous. Furthermore it is assumed that the definition of any internal coordinate (with a specified unit) is associated with the selection of a corresponding reference value σ_n (with the same unit), and that $\mathcal{Q}_n$ is defined by the unitless ratio of the two quantities. It is important to stress that the results of a B&S − LEUS simulation depend on a given choice of the σ_n factors, so that these factors must be clearly specified as an integral part of the definition of the conformational subspace.
2. Representation of the relevant conformational states by means of K centered volumes within the reduced conformational subspace. One possible type of centered volume is the sphere. The biasing potential energy function $\mathcal{B}_k(\boldsymbol{Q})$ corresponding to a sphere $\mathcal{S}_k$ associated with a state k is defined by the following parameters: a sphere center $\boldsymbol{Q}_k$, a radius R_k, a restraining force constant c_k, a number of radial grid points $\Gamma_k + 1$, and a memory force-constant vector $\boldsymbol{M}_k = \{M_{k,i}, i = 0..\Gamma_k\}$. The corresponding expression is (for $k = 1..K$)

$$\mathcal{B}_k(\boldsymbol{Q}) = \begin{cases} M_{k,\Gamma_k} + \frac{1}{2}c_k(r_k - R)^2 & \text{if } r_k \geq R_k \\ \sum_{i=0}^{\Gamma_k} M_{k,i}\gamma(d_{k,i}) & \text{if } r_k < R_k \end{cases} , \tag{9.148}$$

where γ is the LE basis function, and the quantities r_k and $d_{k,i}$ depend on $\boldsymbol{Q}$ as

$$r_k = ||\boldsymbol{Q} - \boldsymbol{Q}_k|| \tag{9.149}$$

and

$$d_{k,i} = \Gamma_k R_k^{-1} r_k - i \ . \tag{9.150}$$

Within the sphere ($r_k < R_k$), the memory vector $\boldsymbol{M}_k$ permits to enforce a radially-dependent potential energy term of arbitrary form (with an approximate resolution $\Gamma_k^{-1}R_k$), expressed as a weighted sum of $\Gamma_k + 1$ repulsive local functions γ. Outside the sphere ($r_k \geq R_k$), the potential energy term is changed to an attractive half-harmonic restraint. It is easily verified that the biasing potential energy term $\mathcal{B}_k$ defined by Equation Eq. 9.148 is continuous and differentiable[84].

A carefully adjusted one-dimensional memory may be used to obtain a biasing potential energy term enforcing a homogeneous radial sampling of the centered volume, but this potential energy term will generally not lead to a homogeneous sampling of the multi-dimensional volume itself, the directional (non-radial) dimensions remaining unbiased. Note also that such a potential energy term will guarantee that the center of the volume is sampled.

3. Definition of a set of conformational paths connecting the centers of the volumes representing the K states. Only the simplest possible type of path will be considered here, namely the line or, more precisely, the line segment. The numbering of the corresponding biasing potential energy terms will start at $K + 1$. The biasing potential energy term $\mathcal{B}_l(\boldsymbol{Q})$ corresponding to a line $\mathcal{L}_l$ is defined by the following parameters: a starting point $\boldsymbol{Q}_l$, an ending point $\boldsymbol{Q}'_l$, a width W_l, a restraining force constant c_l, a number of longitudinal grid points $\Gamma_l + 1$, and a memory force-constant vector $\boldsymbol{M}_l = \{M_{l,i}, i = 0..\Gamma_l\}$. The corresponding expression is (for $k = K + 1..K + L$)

$$\mathcal{B}_l(\boldsymbol{Q}) = \begin{cases} M_{l,0} + \frac{1}{2}c_l H(r_l - W_l)(r_l - W_l)^2 & \text{if } u_l \leq 0 \\ M_{l,\Gamma_l} + \frac{1}{2}c_l H(r'_l - W_l)(r'_l - W_l)^2 & \text{if } u_l \geq U_l \\ \sum_{i=0}^{\Gamma_k}[M_{l,i} + \frac{1}{2}c_l H(p_l - W_l)(p_l - W_l)^2]\gamma(d_{l,i}) & \text{if } 0 < u_l < U_l \end{cases} , \tag{9.151}$$

where the where γ is the LE basis function, U_l is the line length

$$U_l = ||\boldsymbol{Q}'_l - \boldsymbol{Q}_l|| \ , \tag{9.152}$$

and the quantities u_l, p_l, r_l, r'_l and $d_{l,i}$ depend on $\boldsymbol{Q}$ as

$$u_l = U_l^{-1}(\boldsymbol{Q}'_l - \boldsymbol{Q}_l)^T(\boldsymbol{Q} - \boldsymbol{Q}_l) \ , \tag{9.153}$$
$$p_l = ||(\boldsymbol{Q} - \boldsymbol{Q}_l) - U_l^{-1}u_l(\boldsymbol{Q}'_l - \boldsymbol{Q}_l)|| \ , \tag{9.154}$$
$$r_l = ||\boldsymbol{Q} - \boldsymbol{Q}_l|| \ , \tag{9.155}$$
$$r'_l = ||\boldsymbol{Q} - \boldsymbol{Q}'_l|| \ , \tag{9.156}$$

$\boldsymbol{v}^T$ indicating the transpose of a vector $\boldsymbol{v}$, and

$$d_{l,i} = \Gamma_l U_l^{-1} u_l - i \ . \tag{9.157}$$

Note that Eq. 9.151 is formulated so as to allow for displaced lines and lines with longitudinally-dependent widths or force constants. The quantity u_l represents the longitudinal distance between the starting point of the line and the current point $\boldsymbol{Q}$, while the parameter p_l represents the corresponding transverse (perpendicular) distance. Within the line ($0 < u_l < U_l$), the memory vector $\boldsymbol{M}_l$ permits to enforce a longitudinally-dependent potential energy term of arbitrary form (with an approximate resolution $\Gamma_l^{-1}U_l$), expressed as a weighted sum of $\Gamma_l + 1$ repulsive local functions γ, and applied together with a transverse attractive flat-bottom (width W_l) half-harmonic restraining potential energy term. Outside the line, *i.e.* when going past its two terminal points in terms of longitudinal distance ($u_l \leq 0$ or $u_l \geq U_l$), the potential energy term is changed to an attractive flat-bottom (width W_l) half-harmonic restraint depending on the distance to the corresponding end point. It is easily verified that the biasing potential energy term $\mathcal{B}_l$ defined by Equation Eq. 9.151 is continuous and differentiable.

GROMOS also allows a non-linear variant of the line, the displaced line, where an offset coordinate $\Delta\boldsymbol{Q}_{l,i}$ perpendicular to the line (*i.e.* with $\Delta\boldsymbol{Q}_{l,i}^T(\boldsymbol{Q}'_l - \boldsymbol{Q}_l) = 0$ and $\Delta\boldsymbol{Q}_{l,0} = \Delta\boldsymbol{Q}_{l,\Gamma_l} = 0$), and replacing p_l in Equation Eq. 9.151 by

$$p_{l,i} = ||(\boldsymbol{Q} - \boldsymbol{Q}_l) - U_l^{-1}u_l(\boldsymbol{Q}'_l - \boldsymbol{Q}_l) - \Delta\boldsymbol{Q}_{l,i}||. \tag{9.158}$$

This requires the specification of a set of orthonormal vectors, orthogonal to the defined line together with the specification of the displacement along each of these vectors for each grid point. Another possible variant involves the use of longitudinally-dependent line widths or/and restraining force constants, *i.e.* the replacement of c_l and W_l in Equation Eq. 9.151 by corresponding grid-point dependent quantities $c_{l,i}$ and $W_{l,i}$. In principle, the L paths will be chosen to connect pairs among the K centered volumes defining the states. This must be done in such a way that all states are connected to each other *via* at least one path or succession thereof. The minimum number of paths is thus $L = K - 1$ (maximum-spanning tree), but it may be advantageous in terms of convergence properties to include additional (redundant) paths. Note that the end points of the paths must be identical to the centers of the states (*e.g.* they should not connect to the periphery of the centered volumes). This is essential because independent biasing potential energy terms leading to a homogeneous radial sampling of the centered volumes and longitudinal sampling of the paths can only guarantee that these specific points are sampled (for lines, assuming sufficiently small line widths at the end points).

4. Unification of the biasing potential energy terms associated with the $M = K + L$ centered volumes and paths into a single biasing potential energy term according to the enveloping distribution sampling procedure[90]. For the ease of notation, these objects have been given the generic notation $\mathcal{B}_m$, where the index m ranges from 1 to $M = K + L$ ($1..K$ for the centered volumes, $K + 1..M$ for the paths). The various LEUS potential energy terms are combined following the EDS principle as

$$\mathcal{V}^{(bias)}(\boldsymbol{r}; \boldsymbol{M}) = -\frac{1}{\beta s} \ln \left(\sum_{m=1}^{M} \exp[-\beta s \mathcal{B}_m(\boldsymbol{Q}(\boldsymbol{r}))] \right) \ , \tag{9.159}$$

where $\beta = (k_B T)^{-1}$, k_B being Boltzmann's constant and T the absolute temperature, $\boldsymbol{r}$ represents the system configuration (Cartesian coordinates of all particles), $\boldsymbol{Q}(\boldsymbol{r})$ the corresponding representative point in the reduced subspace, s a (positive) smoothing parameter, and $\boldsymbol{M}$ the joint memories of the M objects, *i.e.* a vector containing $N_M = \sum_{m=1}^{M} \Gamma_m$ elements.

Qualitatively speaking, the exponential weighting in Equation Eq. 9.159 ensures that the combined biasing potential energy term $\mathcal{V}^{(bias)}$ is low in the regions of the conformational subspace where *any* of the $\mathcal{B}_m$ is low, and high in the regions where *all* the $\mathcal{B}_m$ are high. For the ease of

reference, the subvolume of the reduced conformational subspace where any of the $\mathcal{B}_m$ is low, *i.e.* the union of all centered volumes and paths, will be referred to as the active subspace.

The forces derived from $\mathcal{V}^{(bias)}$ in Eq. 9.159 are given by

$$\boldsymbol{f}_{bias}(\boldsymbol{r}) = -\frac{\partial \mathcal{V}^{(bias)}(\boldsymbol{r};\boldsymbol{M})}{\partial \boldsymbol{r}} = -\sum_{m=1}^{M} w_m(\boldsymbol{Q}) \frac{\mathrm{d}\mathcal{B}_m(\boldsymbol{Q})}{\mathrm{d}\boldsymbol{Q}} \frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{r}} \tag{9.160}$$

where

$$w_m(\boldsymbol{Q}) = \frac{\exp[-\beta s \mathcal{B}_m(\boldsymbol{Q})]}{\sum_{m=1}^{M} \exp[-\beta s \mathcal{B}_m(\boldsymbol{Q})]} \tag{9.161}$$

can be interpreted as measuring the relative influence (weight) of a single-object biasing potential energy term m on the dynamics of the system in a conformation $\boldsymbol{Q}$. Note that the forces defined by Equation Eq. 9.160 are non-singular, because $\mathcal{B}_m(\boldsymbol{Q})$ and $\boldsymbol{Q}(\boldsymbol{r})$ are both differentiable functions of their arguments.

5. LE build-up phase to optimize the memory, leading to a biasing potential energy term enabling nearly uniform sampling (radially within the centered volumes, longitudinally within the paths) of the active subspace. The updating scheme for the memory $\boldsymbol{M}(t)$ relies on the equation

$$M_{m,i}(t+\Delta t) = M_{m,i}(t) + k^{(le)} f_{LE}^{I_{\mathcal{R}}(t;\gamma_{LE},n_{LE})} j_{m,i}(\boldsymbol{Q}) h_{m,i}(\boldsymbol{Q}) w_m(\boldsymbol{Q}) \ , \tag{9.162}$$

where $M_{m,i}$ is the memory associated with grid point i of object m, $\boldsymbol{Q} = \boldsymbol{Q}(\boldsymbol{r}(t))$, Δt is the simulation timestep, $k^{(le)}$ the basis force-constant increment, f_{LE} a force-constant reduction factor, $I_{\mathcal{R}}$ a force-constant reduction counter, associated with a defined conformational region $\mathcal{R}$, γ_{LE} a local visiting cutoff (real), n_{LE} a global visiting cutoff (integer), $j_{m,i}$ a distribution-alteration function, $h_{m,i}$ a grid-assignment function, and w_m the weight defined by Eq. 9.161. In the absence of prior knowledge concerning the form on the free-energy hypersurface, the memory will typically be initiated to $\boldsymbol{M}(0) = \boldsymbol{0}$. The different factors involved in Eq. 9.162 are explained below.

The grid-assignment function evaluates to one for a single grid point in each of the single-object biasing potential energy terms and to zero for all other grid points, namely the grid point i in object m that is (radially for centered volumes, longitudinally for paths) closest to $\boldsymbol{Q}$. For a sphere k, one has ($k = 1..K$)

$$h_{k,i}(\boldsymbol{Q}) = \begin{cases} \delta_{i,\Gamma_k} & \text{if } r_k \geq R_k \\ \delta_{i,\mathrm{NINT}(\Gamma_k R_k^{-1} r_k)} & \text{if } r_k < R_k \end{cases} \ , \tag{9.163}$$

where δ is the Kronecker symbol and the function NINT returns the nearest integer to a real number. For a line l, one has ($l = K+1..M$)

$$h_{k,i}(\boldsymbol{Q}) = \begin{cases} \delta_{i,0} & \text{if } u_l \leq 0 \\ \delta_{i,\Gamma_l} & \text{if } u_l \geq U_l \\ \delta_{i,\mathrm{NINT}(\Gamma_l U_l^{-1} u_l)} & \text{if } 0 < u_l < U_l \end{cases} \ . \tag{9.164}$$

As a result, the build-up always affects one and only one grid point in each of the M single-object memories. However, the presence of the weight factor w_m in Equation Eq. 9.162 ensures that the build-up is only significant within the objects encompassing or closest to point $\boldsymbol{Q}$ (note that the sum of w_m over all objects is one).

The distribution-alteration function is generally set to

$$j_{m,i}(\boldsymbol{Q}) = 1 \ , \tag{9.165}$$

leading to a nearly homogenous sampling (radially within the centered volumes, longitudinally within the paths) of the active subspace. However, this function may be used to enforce deviations from this homogenous sampling. As a simple example, one may observe that the volume of relevant conformational subspace accounted for by a radial grid point i within a sphere k (distance $\Gamma_k^{-1} i R_k$ from the center) increases with $(i_k+\frac{1}{2})^{N-1}$ (Jacobian factor), where N is the subspace dimensionality. One may then decide to bias the sampling of the sphere towards its periphery, which can be achieved by setting for all spheres k ($k = 1..K$)

$$j_{k,i}(\boldsymbol{Q}) = \frac{(i+\frac{1}{2})^{1-N}}{\sum_{j=0}^{\Gamma_k} (j+\frac{1}{2})^{1-N}} \ . \tag{9.166}$$

The force-constant reduction factor can be used in the context of an iterative procedure to progressively decrease the build-up rate during the searching phase. As noted previously by other authors[86,91], a high build-up rate is desired in the early stage of the searching, where the deep free-energy basins have to be "filled up" coarsely (*i.e.* without wasting computer time), while a low build-up rate (near-equilibrium situation) is preferable in the later stage, where the remaining shallower free-energy wiggles have to be "levelled off" (so as to produce a close-to-optimal biasing potential energy term). This can be achieved by a progressive reduction of the build-up rate, enforced in Equation Eq. 9.162 by using $f_{LE} < 1$ along with a force-constant reduction counter $I_{\mathcal{R}}$ progressively increasing with time (the choice $f_{LE} = 1$ switches off the force-reduction procedure). In the B&S − LEUS algorithm, the force-reduction procedure is associated with a region $\mathcal{R}$ within the active subspace, defined by a specific collection of grid points. The reduction counter $I_{\mathcal{R}}$ is propagated in time according to the following procedure. $I_{\mathcal{R}}(t)$ as well as an auxiliary counter $N_c(t)$ are set to 0 at $t = 0$. An auxiliary memory $\boldsymbol{A}(t)$ is also set to $\mathbf{0}$ at $t = 0$ and propagated in time according to the equation

$$A_{m,i}(t + \Delta t) = A_{m,i}(t) + w_m(\boldsymbol{Q})h_{m,i}(\boldsymbol{Q}) \ . \tag{9.167}$$

When $A_{m,i}(t)$ exceeds a specified local visiting cutoff γ_{LE} for all grid points $(m, i) \in \mathcal{R}$, the auxiliary counter is increased by one and the auxiliary memory reset to zero. When the auxiliary counter exceeds a global visiting cutoff n_{LE}, $I_{\mathcal{R}}$ is increased by one and the auxiliary counter reset to zero. Two possible (reasonable) choices for $\mathcal{R}$ are either: (i) the $i = 0$ (central) grid points of all centered volumes k ($k = 1..K$), a choice that will be noted $\mathcal{R} = \mathcal{C}$; (ii) all grid points i of all objects m ($m = 1..M$), a choice that will be referred to as $\mathcal{R} = \mathcal{A}$. Possible (reasonable) choices for the parameters γ_{LE} and n_{LE} are 1.0 and 2, respectively. The reasoning behind the present force-constant reduction scheme (assuming $\gamma_{LE} = 1.0$ and $n_{LE} = 2$) is that when all grid points of $\mathcal{R}$ have undergone an "effective" number of visits (auxiliary memory, *i.e.* based on the w_m weights) of one, it is still possible that the "flattened" free-energy hypersurface retains an overall "slope". However, when all these points have undergone an "effective" number of visits of one for the second time, even the points that were "uphill" have been revisited. When this condition is met, it becomes advantageous to reduce the build-up rate by incrementing $I_{\mathcal{R}}$, which in effect scales this rate by a factor f_{LE}.

Finally, the constant $k^{(le)}$ in Equation Eq. 9.162 represents the basic force-constant increment (units of energy) and determines the initial rate of the build-up. Note that the above force-reduction procedure also presents the advantage of permitting a convergence assessment of the build-up phase, by monitoring the time evolution of $I_{\mathcal{R}}$. The build-up phase can, for example, be terminated whenever $I_{\mathcal{R}}$ reaches a threshold value $I_{\mathcal{R}}^{max}$. In this case, the procedure guarantees that all grid points of $I_{\mathcal{R}}$ have undergone an "effective" number of visits of at least $n_{LE}\gamma_{LE}I_{\mathcal{R}}^{max}$, while the energetic resolution of the biasing potential energy term is of the order of $f_{LE}^{I_{\mathcal{R}}^{max}} k^{(le)}$. Alternatively, the termination may be based on the time interval separating successive incrementations of $I_{\mathcal{R}}$. In the initial stage of the build-up, the diffusion of the system within the active subspace will be accelerated (hill surfing). However, as the free-energy hypersurface becomes increasingly "flat" and the build-up rate is decreased, this diffusion will progressively slow down towards a "natural" regime (as determined by the physical system after removal of the free-energy bias). Thus, the force-reduction procedure could also be terminated when the interval separating successive increments of $I_{\mathcal{R}}$ has increased and leveled off to an approximately constant time.

6. US sampling phase to generate of a biased ensemble of configurations, using the biasing potential energy term pre-optimized during the LE build-up phase. Owing to Eq. 9.159 the biased sampling during this phase should be approximately homogeneous (radially within the centered volumes, longitudinally along the paths) within the active subspace.
7. Reweighting and state assignment so as to calculate the relative free energies of the states in the physical ensemble. For each state k, the free energy can be written (for $k = 1...K$)

$$G_k = -\beta^{-1} \ln \left\langle \exp[\beta \mathcal{V}^{(bias)}(\boldsymbol{r}; \boldsymbol{M})] \right\rangle_{\boldsymbol{Q}(\boldsymbol{r}) \in \mathcal{S}'_k} + C_G \tag{9.168}$$

where C_G is an offset constant and $\langle .. \rangle_{\boldsymbol{Q}(\boldsymbol{r}) \in \mathcal{S}'_k}$ denotes ensemble (trajectory) averaging over the biased ensemble (sampling phase), restricted to conformations belonging to state k (see also Sec. 14.8 and Eq. 14.162). The symbol $\mathcal{S}'_k$ has been used rather than $\mathcal{S}_k$ to underline the fact that the regions

used to assign the states need not necessarily be exactly identical to the centered volumes involved in the construction of the biasing potential energy term.

CHAPTER 10

Constraints

10.1. Introduction

This chapter will discuss various kinds of hard geometric boundary conditions that are implemented in GROMOS. These may concern the position of atoms, the distance between pairs of atoms, the dihedral angle defined by four atoms or the overall translation and rotation of a set of atoms. The *application* of holonomic geometric *constraints* in molecular simulation may have the following *advantages*.

1. When constructing a molecular model in which certain degrees of freedom are ill-defined, one may optimize or sample these degrees of freedom under the constraint that the other degrees of freedom do not change.

2. If a degree of freedom is characterized by vibrational frequencies ν, for which

   $$h\nu \gg k_B T \tag{10.1}$$

   the motion will be of quantum-mechanical nature. In equilibrium at room temperature, condition Eq. 10.1 is fulfilled for e.g. the bond-stretching vibrations. Treating the bonds as constraints is probably a better approximation of their quantum behaviour than treating them as classical harmonic oscillators[92].

3. When the range of frequencies in a molecular simulation is very broad, this may cause the energy relaxation between high-frequency and low-frequency modes to be slow. If the forces for the different modes are computed with different accuracies, the different modes will suffer from different heating rates. When the latter are faster than the energy relaxation rate of the system, the system will not reach equilibrium, but remain in a stationary state. This problem can be solved by coupling the degrees of freedom of different frequency and heating rate to separate temperature baths (Sec. 12.2). However, the application of constraints generally reduces the range of frequencies in the system and thus alleviates this problem.

4. The application of constraints will generally save computing effort. The length of the time step Δt in a MD or SD simulation is limited by the highest frequency ν_{max} occurring in the molecular system of interest,

   $$\Delta t \gg \nu_{\mathrm{max}}^{-1}. \tag{10.2}$$

   By constraining the degrees of freedom with the highest frequencies the time step Δt can generally be lengthened, which reduces the computer time required for a simulation of a given length. For molecular systems one may think of reducing computational effort by constraining bond lengths or additionally bond angles, the latter only for molecules without internal torsional degrees of freedom.

5. A potential of mean force along a degree of freedom can be obtained by applying a constraint along the degree of freedom and performing a thermodynamic integration in which the derivative of the free energy with respect to a modification of the constraint is calculated from the constraint forces.

6. In order to minimize the amount of solvent needed to solvate a large molecular system, one may constrain the overall translation and rotation of a given set, allowing for a smaller computational box, but ensuring that the cutoff criterion (Eq. 4.26) will not be violated due to a rotation of the system within the box.

The application of *constrained dynamics* makes physical sense only when[47]

1. the frequencies of the frozen degrees of freedom are (considerably) higher than those of the remaining ones, thereby allowing a (considerable) increase of Δt, and when
2. the frozen degrees of freedom are only weakly coupled to the remaining ones, viz. when the motion of the molecules is not affected by application of the constraints[93], and when
3. metric tensor effects due to constraining the molecules to a hypersurface in configuration space, play a minor role.[94]

For biomolecules the effect of constraining bond lengths and bond angles has been evaluated[93]. It turns out that the application of *bond-length constraints* saves about a factor of 2 in computer time when hydrogen atoms are explicitly treated and a factor of 3 when the united-atom model is used[47]. Metric tensor corrections play no role when only bond-length constraints are applied[94]. The use of *bond-angle constraints* is not allowed in flexible (viz. with rotational internal degrees of freedom) molecules, since it affects the dynamics of such molecules considerably[93]. Moreover, metric tensor effects are non-negligible in this case[94]. The bond-angle degrees of freedom appear to be coupled to the other molecular degrees of freedom, such as the torsional-angle ones. However, for completely rigid molecules, that is, without internal degrees of freedom, metric tensor effects play no role, and the application of bond-length and bond-angle constraints is common practice.

The application of position constraints is described in Sec. 10.2. For application of distance constraints GROMOS uses the *SHAKE-method*[95] and its derivatives (viz. SETTLE[96], M-SHAKE[97], and LINCS[98], and FLEXSHAKE[99]), which will be described in Sec. 10.3. The application of bond-length constraints to a solute molecule is described in Sec. 10.4, and the use of constraints in rigid solvent molecules is discussed in Sec. 10.5. Dihedral-angle constraints are described in Sec. 10.6, and overall translational and rotational constraints in Sec. 10.7.

10.2. Position Constraints

In certain cases, it may be advantageous to prevent certain atoms from moving completely. GROMOS allows the user to apply position constraints to selected atoms. The selection of the atoms which are to be kept fixed at given reference positions during a simulation or energy minimization is done as for harmonic position restraining (Sec. 9.2), except that the switch $NTPOR = 3$. Solvent atom positions cannot be kept fixed.

Atoms can be kept at their initial positions $\mathbf{r}_i = \mathbf{r}_i^0$ by setting their velocities $\mathbf{v}_i$ and forces $\mathbf{f}_i$ equal to zero before every MD or EM integration (time) step. When applying constraints using the SHAKE method, see Sec. 10.3.1, the atomic position of an atom i involved in a constraint is changed with m_i^{-1} as weight factor. If such an atom is selected for position fixing, its mass is increased such that the atom position is effectively not changed when applying SHAKE. If two atoms i and j are involved in a distance (bond-length) constraint $r_{ij} = d_{ij}^0$ to be imposed by SHAKE ($NTC > 1$) and they are *both* selected as atoms to be kept fixed, the constraint d_{ij}^0 will be skipped when the constraint list is handled by SHAKE.

If one or more atoms of the molecular system are kept fixed, removal of centre of mass motion is disabled (even though $NSCM > 0$).

Fixing of atom positions should not be used in a free energy perturbation calculation.

When calculating a temperature, the total kinetic energy of the degrees of freedom for which the temperature is to be determined, is divided by the number of these degrees of freedom multiplied by $\frac{1}{2}k_B$(k_B = Boltzmann's constant). When $\mathcal{N}^{(pr)}$ atoms are kept fixed, the system contains $3\mathcal{N}^{(pr)}$ less degrees of freedom. If atoms are kept fixed, the choice of temperature bath coupling is restricted: i) solute internal/rotational degrees of freedom should be coupled to the same bath as solute translational degrees of freedom, ii) if the whole molecule is positionally constrained, temperature coupling makes no sense.

10.3. Constraints using the SHAKE method and its derivatives

10.3.1. Constraints using the SHAKE method. The SHAKE method[95] can be used to impose distance constraints onto the molecular system. Bond-length and bond-angle constraints can be put in the form of *distance constraints* between atoms k_1 and k_2,

$$\sigma_k(\mathbf{r}^N) \equiv r_{k_1k_2}^2 - \left(d_{k_1k_2}^0\right)^2 = 0 \quad k = 1, 2, ..., N^{(c)} \tag{10.3}$$

where $k \equiv (k_1,k_2)$, the constraint distance is given by $d^0_{k_1k_2}$ and the actual distance between atoms k_1 and k_2 by $r_{k_1k_2}$. The equations of motion Eqs. 2.8, 2.9, 2.10 and 2.12 (Newtonian equations of motion) or 2.13 (Langevin equations of motion) have to be integrated while satisfying conditions Eq. 10.3. This can be accomplished by applying *Lagrange's* method of undetermined *multipliers*. A zero term, expression Eq. 10.3, is added to the potential energy function $\mathcal{V}(\mathbf{r})$ in Eq. 2.10, which then yields the equations of motion (Newton)

$$m_i \frac{d^2\mathbf{r}_i(t)}{dt^2} = -\frac{\partial}{\partial \mathbf{r}_i}\left[\mathcal{V}(\mathbf{r}) + \sum_{k=1}^{N^{(c)}} l_k(t)\sigma_k(\mathbf{r})\right]. \tag{10.4}$$

The time-dependent multipliers $l_k(t)$ are to be determined such that the conditions Eq. 10.3 are satisfied. The first term on the right hand side of Eq. 10.4 represents the *unconstrained force* $\mathbf{f}^{uc}_i(t)$ derived from the interaction function, and the second term represents the yet unknown *constraint force*

$$\mathbf{f}^{(c)}_i(t) = -\sum_{k=1}^{N^{(c)}} l_k(t)\frac{\partial \sigma_k(\mathbf{r}(t))}{\partial \mathbf{r}_i(t)} \tag{10.5}$$

which compensates the components of $\mathbf{f}^{uc}_i(t)$ that act along the directions of the constraints.

In the leap-frog MD or SD algorithm, which is used in GROMOS, solving for $l_k(t)$ is done as follows. The leap-frog equation (see Sec. 12.1) for the atomic positions with the unconstrained forces gives the unconstrained positions at time $t + \Delta t$,

$$\mathbf{r}^{uc}_i(t+\Delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t-\Delta t/2)\Delta t + m_i^{-1}\mathbf{f}^{uc}_i(t)(\Delta t)^2 \tag{10.6}$$

Since the constrained positions at $t + \Delta t$,

$$\mathbf{r}_i(t+\Delta t) = \mathbf{r}^{uc}_i(t+\Delta t) + m_i^{-1}\mathbf{f}^{(c)}{}_i(t)(\Delta t)^2 \tag{10.7}$$

must satisfy the constraint equation Eq. 10.3, we find

$$\left(\left[\, \mathbf{r}^{uc}_{k_1}(t+\Delta t) + m_{k_1}^{-1}\mathbf{f}^{(c)}_{k_1}(t)(\Delta t)^2 - \mathbf{r}^{uc}_{k_2}(t+\Delta t) - m_{k_2}^{-1}\mathbf{f}^{(c)}_{k_2}(t)(\Delta t)^2\right]\right)^2$$

$$-\left(d^0_{k_1k_2}\right)^2 = 0. \qquad with\ k = 1,2,...,N^{(c)} \tag{10.8}$$

Substituting

$$\mathbf{f}^{(c)}_i(t) = -2\sum_{k=1}^{N^{(c)}} l_k(t)(\delta_{ik_1} - \delta_{ik_2})\mathbf{r}_{k_1k_2}(t) \tag{10.9}$$

into Eq. 10.8 we find

$$\left[\, \mathbf{r}^{uc}_{k_1k_2}(t+\Delta t) - 2l_k(t)(m_{k_1}^{-1} + m_{k_2}^{-1})\mathbf{r}_{k_1k_2}(t)(\Delta t)^2\right]^2 - \left(d^0_{k_1k_2}\right)^2 = 0 \qquad k = 1,2,...,N^{(c)} \tag{10.10}$$

which is a set of $N^{(c)}$ quadratic equations from which the $N^{(c)}$ multipliers $l_k(t)$ are to be solved and used to obtain the constrained positions $\mathbf{r}$ $(t + \Delta t)$ through Eqs. 10.7 and 10.9. In the SHAKE method, the equations Eq. 10.10 are linearized by neglecting the terms quadratic in $l_k(t)$, which yields

$$l_k(t) = \frac{\left(d^0_{k_1k_2}\right)^2 - (\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t))^2}{-4(\Delta t)^2(m_{k_1}^{-1} + m_{k_2}^{-1})(\mathbf{r}_{k_1k_2}(t)\cdot\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t))} \tag{10.11}$$

Since the atoms may be involved in more than one constraint, the set of equations Eq. 10.10 is solved iteratively, where the corrections to the unconstrained positions are given by

$$\Delta\mathbf{r}^{uc}_{k_1}(t+\Delta t) = -2(\Delta t)^2 m_{k_1}^{-1} l_k(t)\mathbf{r}_{k_1k_2}(t)$$

$$= m_{k_1}^{-1} g_k(t)\mathbf{r}_{k_1k_2}(t) \tag{10.12}$$

and

$$\Delta\mathbf{r}^{uc}_{k_2}(t+\Delta t) = +2(\Delta t)^2 m_{k_2}^{-1} l_k(t)\mathbf{r}_{k_1k_2}(t)$$

$$= -m_{k_2}^{-1} g_k(t)\mathbf{r}_{k_1k_2}(t) \tag{10.13}$$

This implies that corrections due to the distance constraint $d^0_{k_1k_2}$ to the positions of atoms k_1 and k_2 are applied in the direction of the vector $\mathbf{r}_{k_1k_2}$, and are in opposite direction, weighted by the inverse mass of atoms k_1 and k_2. This is illustrated in Fig. 10.1. The iterations over all $N^{(c)}$ constraints in SHAKE are terminated,

1. if all $N^{(c)}$ distance constraints are satisfied within a given relative tolerance *tol* (TOL), or
2. if an excessive (1000) number of iterations is required, or
3. if the positional shift in the unconstrained step, $\mathbf{r}^{uc}_{k1}$ - $\mathbf{r}_{k_1}$ or $\mathbf{r}^{uc}_{k_2}$ - $\mathbf{r}_{k_2}$ is so big that no corrections (Eq. 10.12, Eq. 10.13) along $\mathbf{r}_{k_1k_2}$ can be found that will produce a distance $d^0_{k_1k_2}$ between the corrected positions of atoms k_1 and k_2 , see Fig. 10.2.

The *application* of the procedure *SHAKE* will be denoted by

$$SHAKE\ (\mathbf{r}(t); \mathbf{r^{uc}}(t+\Delta t); \mathbf{r}(t+\Delta t)). \tag{10.14}$$

This means that the positions $\mathbf{r}^{uc}(t+\Delta t)$ that result from the non-constrained time step, will be reset to give the constrained positions $\mathbf{r}(t+\Delta t)$. The direction of the correction vectors is determined by the *reference positions* $\mathbf{r}(t)$, that is, for each individual distance constraint involving atom k_1 and k_2 , the correction vector is parallel to the vector $\mathbf{r}_{k_1k_2}(t)$ of the reference configuration.

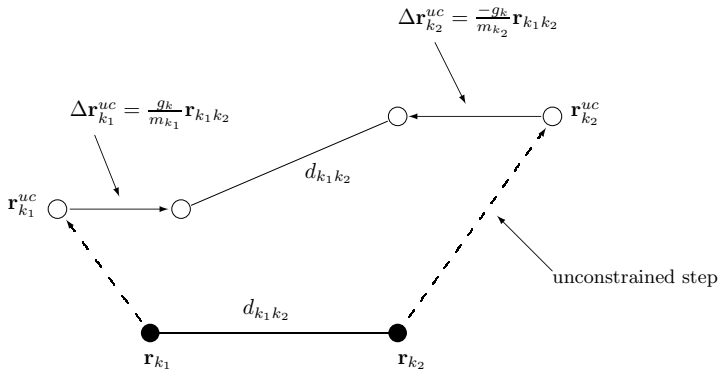


FIGURE 10.1. Positional corrections $\Delta\mathbf{r^{uc}}$ induced by SHAKE.

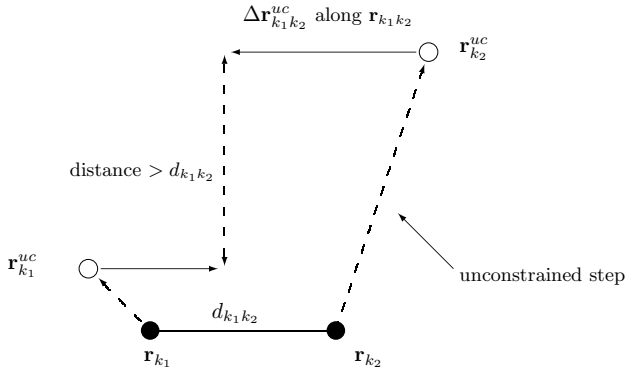


FIGURE 10.2. Situation in which SHAKE fails due to a too large unconstrained step.

When using SHAKE in combination with the GROMOS force field the *geometric tolerance* (TOL) should be chosen such that the noise in the simulation due to SHAKE is much smaller than that due to other sources, such as the application of a nonbonded interaction cutoff radius, etc. When applying energy minimization

(EM), we choose TOL = 10^{-3}, whereas in MD a value of at least TOL = 10^{-4} is used, based on the observation that MD is more sensitive to the accumulation of errors when moving through configuration space.

10.3.2. Constraints using the SETTLE method. The time derivatives of Eq. 10.3 give constraints on the velocities,

$$\mathbf{r}_{ij} \cdot \mathbf{v}_{ij} \quad = \quad 0 \tag{10.15}$$

This suggests that the component of the relative velocity along the bond should be zero. This constraint can be applied to both the position constraint, similar to the SHAKE method, and the velocity constraint, similar to the RATTLE method.

In the SETTLE method, the determination of $\mathbf{r}_i(t_0 + \delta t)$ is through the use of quasi-Euler angles thus the explicit calculation of constraint forces is avoided.

Taking a rigid water molecule as an example, a triangle ABC is defined with the oxygen and the two hydrogen atoms corresponding to A, B, and C respectively. $\Delta A_0 B_0 C_0$, $\Delta A_1 B_1 C_1$, and $\Delta A_3 B_3 C_3$ are the triangles at time t_0, $t_0 + \delta t$ in the absence of constraints, and $t_0 + \delta t$ with constraints. Ideally, triangles $\Delta A_0 B_0 C_0$ and $\Delta A_3 B_3 C_3$ should overlap each other after proper rotation (about the y' axis by ψ ($-\pi/2 \leq \psi \leq \pi/2$, about the x' axis by ϕ ($-\pi < \phi \leq \pi$) and about the z' axis by θ ($-\pi < \phi \leq \pi$)), assuming zero centre of mass motion.

By introducing an orthogonal coordinate system $x'y'z'$ which the origin coincides with the center of mass d_0 and the $x'y'$ plane is parallel to the plane of $\Delta A_0 B_0 C_0$, the angles ψ, ϕ are determined uniquely from $\Delta A_1 B_1 C_1$. The angle θ can then be calculated analytically by using the condition that constraint forces directed along the bond at time t_0 are of equal magnitudes and opposite orientations.

In the case of velocity constraints, the solution of the constrained velocity is

$$\mathbf{v}_i(t_0 + \delta t) \quad = \quad \mathbf{v}_i^{uc}(t_0 + \delta t) + \delta \mathbf{v}_i^0(t_0 + \delta t) \tag{10.16}$$

where $\mathbf{v}_i^{uc}(t_0 + \delta t)$ is the velocity after an unconstraint step (δt) and $\delta \mathbf{v}_i(t_0 + \delta t)$ is the correction velocity necessary to satisfy the constraints,

$$\begin{aligned} \delta \mathbf{v}_i(t_0 + \delta t) &= 1/2 \cdot \delta t / m_i \cdot \sum \mathbf{g}_{ij}(t_0 + \delta t) \\ &= 1/2 \cdot \delta t / m_i \cdot \sum l_{ij}(t_0 + \delta t) \mathbf{r}_{ij}(t_0 + \delta t). \end{aligned} \tag{10.17}$$

where the Lagrangian multipliers, $l_{ij}(t_0)$, ($l_{ij} = l_{ji}$) are chosen so that the constraint Eq. 10.3 are satisfied at time $t_0 + \delta t$.

A physical picture of this is that constraint forces $[\mathbf{f}^{(c)}{}_{ij}(t_0)]$ of equal magnitudes and opposite orientations are applied to the atoms i and j and are directed along the bond vectors $[\mathbf{r}_{ij}(t_0)]$ at time t_0. In the conventional method a set of quadratic equations for l_{ij} is obtained by substitution of Eq. 10.16 and Eq. 10.17 into Eq. 10.15. The solution to the quadratic equations is given by first solving them in their linear form and subsequently iterating them until all the constraints are fulfilled to within an acceptable tolerance. The linear equations can be solved by either matrix inversion or by the SHAKE method.

The constrained velocities at time $t_0 + \delta t$ of the three vertices are:

$$\begin{aligned} \mathbf{v}_A(t_0 + \delta t) \quad = \quad & \mathbf{v}_A^{uc}(t_0 + \delta t) + \delta t / 2 m_a \cdot (\tau_{AB}(t_0 + \delta t) \\ & - \tau_{CA}(t_0 + \delta t)) \end{aligned} \tag{10.18}$$

where τ_{AB} and τ_{CA} are the Lagrangian multipliers.

According to the constraint in Eq. 10.15,

$$\mathbf{r}_{AB} \cdot \mathbf{v}_{AB} = \mathbf{r}_{AB} \cdot (\mathbf{v}_B - \mathbf{v}_A) \quad = \quad 0 \tag{10.19}$$

the rearrangement of Eq. 10.19 gives

$$\begin{aligned} & \delta t (m_a + m_b) \tau_{AB} + \delta t \cdot m_a \tau_{BC} \cdot cos\mathbf{B} + \delta t \cdot m_b \tau_{CA} \cdot cos\mathbf{A} \\ & \qquad = 2 m_a m_b \hat{\mathbf{e}}_{AB} \cdot \mathbf{v}_{AB}^{uc} \end{aligned} \tag{10.20}$$

where $cosA$ and $cosB$ are cosines of the apex angles of A and B, and $\mathbf{e}_{AB}$ is the unit vector of $\mathbf{r}_{AB}$.

Similarly,

$$\delta t(m_b + m_c)\tau_{BC} + \delta t \cdot m_b\tau_{CA}cos\mathbf{C} + \delta t \cdot m_c\tau_{AB}cos\mathbf{B} = 2m_b m_c \mathbf{e}_{BC} \cdot \mathbf{v}^{uc}_{BC} \tag{10.21}$$

$$\delta t(m_c + m_a)\tau_{CA} + \delta t \cdot m_c\tau_{AB}cos\mathbf{A} + \delta t \cdot m_a\tau_{BC}cos\mathbf{C} = 2m_c m_a \mathbf{e}_{CA} \cdot \mathbf{v}^{uc}_{CA} \tag{10.22}$$

These equations are simultaneous linear equations with respect to the variables $\tau_{AB}, \tau_{BC}, \tau_{CA}$, and they can be solved by use of Cramer's rule. The $\tau_{AB}, \tau_{BC}, \tau_{CA}$ can then be used to calculate the constrained velocities.

10.3.3. Constraints using the LINCS method. The LINear Constraint Solver (LINCS) method was developed for the treatment of bond constraints by means of two steps of projections:

1. project the new bonds onto the old directions of the bonds;
2. correction for rotational lengthening.

The equation of motion with constraints Eq. 10.4 can be rewritten as:

$$\text{-}\underline{\mathbf{m}}\frac{d^2\mathbf{r}}{dt^2} + \mathbf{B}^T\mathbf{l} + \mathbf{f}^{uc} = 0 \tag{10.23}$$

where $\mathbf{B}$ is a $K \times 3N$ matrix containing the directions of the constraints, and defined by

$$\mathbf{B}_{ki} = \frac{\partial \sigma_k}{\partial r_i} \tag{10.24}$$

Since

$$\frac{d^2\sigma_k}{dt^2} = \mathbf{B}\frac{d^2\mathbf{r}}{dt^2} + \frac{d\mathbf{B}}{dt}\frac{d\mathbf{r}}{dt} = 0 \tag{10.25}$$

The equation of motion Eq. 10.23 can be rewritten as

$$\frac{d^2\mathbf{r}}{dt^2} = (\mathbf{I} \text{ - } \mathbf{TB})\underline{\mathbf{m}}^{-1}\mathbf{f}^{uc} \text{- } \mathbf{T}\frac{d\mathbf{B}}{dt}\frac{d\mathbf{r}}{dt} \tag{10.26}$$

where $\mathbf{T} = \underline{\mathbf{m}}^{-1}B^T(B\underline{\mathbf{m}}{-}1B^T)^{-1}$ is a $3N \times K$ matrix that transforms motions in Cartesian coordinates, without changing the equations of motion of the unconstrained coordinates. **I - TB** is a projection matrix that sets the constrained coordinates to zero, $\mathbf{B}\underline{\mathbf{m}}^{-1}\mathbf{f}^{uc}$ is a $\mathbf{K}$ vector of second derivatives of the bond lengths in the direction of the bonds. The last term represents centripetal forces caused by rotating bonds.

In a leap-frog algorithm, the constrained positions at time step $n+1$ (in other words, at time $(n+1)\Delta t$),

$$\begin{aligned}
\mathbf{r}_{n+1} &= (\mathbf{I} - \mathbf{T_n}\mathbf{B_n})(\mathbf{r}_n + \Delta t\mathbf{v}_{n-\frac{1}{2}} + \Delta t^2\underline{\mathbf{m}}^{-1}\mathbf{f}^{uc}_n) + \mathbf{T}_n\mathbf{d^0} \\
&= (\mathbf{I} - \mathbf{T_n}\mathbf{B_n})\mathbf{r}^{uc}_{n+1} + \mathbf{T}_n\mathbf{d^0} \\
&= \mathbf{r}^{uc}_{n+1} - \underline{\mathbf{m}}^{-1}\mathbf{B}_n(\mathbf{B}_n\underline{\mathbf{m}}^{-1}\mathbf{B}_n^T)^{-1}(\mathbf{B}\cdot\mathbf{r}^{uc}_{n+1} - \mathbf{d^0})
\end{aligned} \tag{10.27}$$

The matrix $(\mathbf{B}_n\ \underline{\mathbf{m}}^{-1}\ \mathbf{B}_n^T)$ has diagonal elements of $1/m_{k_1} + 1/m_{k_2}$, and non-zero off-diagonal elements of $cos\phi/m_c$ when two bonds are connected by the atom with the mass of m_c and a bond angle of ϕ. Its inversion can be transformed to the following form:

$$\begin{aligned}
(\mathbf{B}_n\underline{\mathbf{m}}^{-1}\mathbf{B}_n^T)^{-1} &= \mathbf{S}\mathbf{S}^{-1}(\mathbf{B}_n\underline{\mathbf{m}}^{-1}\mathbf{B}_n^T)^{-1}\mathbf{S}^{-1}\mathbf{S} \\
&= \mathbf{S}(\mathbf{I} - \mathbf{A_n})^{-1}\mathbf{S}
\end{aligned} \tag{10.28}$$

where $\mathbf{S}$ is a $K \times K$ matrix,

$$\mathbf{S} = Diag(\sqrt{\frac{1}{m_{1_1}} + \frac{1}{m_{1_2}}}, \ldots, \sqrt{\frac{1}{m_{k_1}} + \frac{1}{m_{k_2}}}) \tag{10.29}$$

Since $\mathbf{A}_n$ is symmetric and sparse and has zeros on the diagonal, the inversion of the matrix ($\mathbf{I}$ - $\mathbf{A}_n$) can be solved efficiently through an expansion,

$$(\mathbf{I} - \mathbf{A_n})^{-1} \quad = \quad \mathbf{I} + \mathbf{A}_n + \mathbf{A}_n^2 + \mathbf{A}_n^3 + \ldots \tag{10.30}$$

10.3.4. Constraints using the M-SHAKE method. In the SHAKE procedure, two approximations are made to solve a system of $N^{(c)}$ quadratic equations for the Lagrange multipliers:

1. the system of equations is linearized by neglecting any term quadratic in the multipliers,
2. the multipliers are determined independently in sequence by omitting the coupling between distance constraints involving a common atom.

As a consequence of these approximations, the procedure must be performed iteratively until satisfaction of all constraints within a specified tolerance. Note, however, that the second approximation can easily be removed by a M-SHAKE procedure (Matrix-inversion SHAKE). In this procedure, the linearized system of coupled equations is solved exactly through the inversion of an $N^{(c)} \times N^{(c)}$ matrix.

According to Eq. 10.10, a set of $N^{(c)}$ quadratic equations is obtained and to be solved for the Lagrange multipliers $l_k(t)$

$$\begin{aligned} &\{\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t) - 2(\Delta t)^2 \sum_{k'=1}^{N^{(c)}} l_{k'}(t)\mathbf{r}_{k'_1k'_2}(t) \\ &\times[m_{k_1}^{-1}(\delta_{k_1,k'_1} - \delta_{k_1,k'_2}) + m_{k_2}^{-1}(\delta_{k_2,k'_2} - \delta_{k_2,k'_1})]\}^2 \\ &\quad -(d^0_{k_1k_2})^2 = 0 \qquad k = 1, 2, ..., N^{(c)} \end{aligned} \tag{10.31}$$

After linearization, Eq. 10.31 is rewritten in the matrix form

$$\mathbf{Al} \quad = \quad \mathbf{c} \tag{10.32}$$

where l is an $N^{(c)}$-dimensional vector containing the Lagrange multipliers, and the elements of the vector $\mathbf{c}$ are given by:

$$c_k \quad = \quad \frac{[\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t)]^2 - (d^0_{k_1k_2})^2}{4(\Delta t)^2} \tag{10.33}$$

and the elements of the matrix $\mathbf{A}$

$$\begin{aligned} A_{kk'} = &[m_{k_1}^{-1}(\delta_{k_1,k'_1} - \delta_{k_1,k'_2}) + m_{k_2}^{-1}(\delta_{k_2,k'_2} - \delta_{k_2,k'_1})] \\ &\times \mathbf{r}_{k'_1k'_2}(t)\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t). \end{aligned} \tag{10.34}$$

It is easily seen that the diagonal elements A_{kk} characterize the force directly due to constraint k, whereas the off-diagonal elements $A_{kk'}(k' \neq k)$ account for the effect along a constraint k of the forces due to a constraint k'. These off-diagonal elements are neglected in SHAKE.

In M-SHAKE, Eq. 10.32 is solved directly by matrix inversion with the following methods:

1. Explicit (hard-coded) calculation of $\mathbf{A}^{-1}$. This was only done for water ($N^{(c)} = 3$), because the expressions quickly become very complex for larger matrices.
2. Use of Cramer's rule. In this method, each component of the vector l is calculated as $l_k = |A_k|/|A_j|$, where $\mathbf{A}_k$ is the determinant obtained by replacing the kth column of $\mathbf{A}$ by the vector $\mathbf{c}$. Here again, the expressions quickly become untractable for larger matrices and the method was only tested for water.
3. Use of the *LU*-factorization method. In this method, a square, non-singular matrix $\mathbf{A}$ is factorized into a lower triangular matrix $\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$, such that $\mathbf{LU} = \mathbf{A}$. The computational cost of the factorization scales as $N^{(c)3}$, whereas the inversion of the triangular matrices only scales as $N^{(c)2}$.
4. Use of the $\mathbf{LDL}^t$-factorization method. In this case, the matrix $\mathbf{A}$ is approximated by a symmetric matrix **A'** with identical diagonal elements, but in which any occurrence of $\mathbf{r}^{uc}_{k_1k_2}(t+\Delta t)$ in the off-diagonal elements is replaced by $\mathbf{r}_{k_1k_2}(t)$. The symmetric matrix **A'** is then factorized as $\mathbf{LDL}^t = \mathbf{A}$, where $\mathbf{D}$ is a diagonal matrix, $\mathbf{L}$ a lower triangular matrix, and the t superscript denotes the

transpose. This factorization method also scales as $N^{(c)3}$, but is about twice less expensive in terms of floating-point operations compared to LU-factorization.

10.3.5. Constraints using the FLEXSHAKE method. In the FLEXSHAKE method, the ideal constrained bond lengths for every constraint are recalculated at each time step, and the total energy U^{uc} and a hypothetic harmonic bond-stretching energy $U^c = \sum_{k=1}^{N^{(c)}} U_k^c$ is minimized with respect to the bond or constraint lengths d_k^0 at each time step[99].

The Hamiltonian for a flexible constrained system is:

$$\frac{d}{dt}\boldsymbol{q} = \underline{\mathbf{m}}^{-1}\boldsymbol{p} \tag{10.35}$$

$$\frac{d}{dt}\boldsymbol{q} = -\nabla_{\boldsymbol{q}} U^c(\boldsymbol{q}) - \nabla_{\boldsymbol{q}} \sum_{k=1}^{N^{(c)}} l_k(t)\sigma_k'(\boldsymbol{q},t) \tag{10.36}$$

$$\sigma_k'(\boldsymbol{q},t) = |\mathbf{r}_k|^2 - \left(d_k^0\right)^2(t), k = 1, 2, \cdots, N^{(c)} \tag{10.37}$$

The constrained forces are given by

$$\begin{aligned}\mathbf{f}_i^{(c)}(t) &= -\nabla_{\boldsymbol{q}} \sum_{k=1}^{N^{(c)}} l_k(t)\sigma_k'(\boldsymbol{q},t) \\ &= -2\sum_{k=1}^{N^{(c)}} l_k[\mathbf{r}_k(t)(\delta_{ik_1} - \delta_{ik_2}) - d_k^0(t+\Delta t)\nabla_{\boldsymbol{q}_i} d_k^0(t+\Delta t)]\end{aligned} \tag{10.38}$$

The new constrained positions are given by

$$\begin{aligned}\mathbf{r}_i(t+\Delta t) &= \mathbf{r}_i^{uc}(t+\Delta t) - \frac{2(\Delta t)^2}{m_i}\sum_{k=1}^{N^{(c)}} l_k[(\delta_{ik_1} - \delta_{ik_2})\mathbf{r}_k(t) \\ &\quad -d_k^0(t+\Delta t)\nabla_{\boldsymbol{q}_i} d_k^0(t+\Delta t)].\end{aligned} \tag{10.39}$$

Inserting Eq. 10.39 into Eq. 10.37 and neglecting the second order terms of l_k gives the Lagrange multipliers l_k

$$\begin{aligned}l_k(t) &= \frac{d_k^0(t+\Delta t)^2 - (\mathbf{r}_k^{uc}(t+\Delta t))^2}{-4(\Delta t)^2 \cdot \mathbf{r}_k^{uc}(t+\Delta t)} \\ &\quad \cdot \frac{1}{(m_{k_1}^{-1} + m_{k_2}^{-1})\mathbf{r}_k(t) - d_k^0(t+\Delta t)[m_{k_1}^{-1}\nabla_{\boldsymbol{q}_{k_1}} d_k^0(t+\Delta t) - m_{k_2}^{-1}\nabla_{\boldsymbol{q}_{k_2}} d_k^0(t+\Delta t)]}\end{aligned} \tag{10.40}$$

The $N^{(c)}$ equations can be solved iteratively using the SHAKE method, and the l_k can be then used to calculate the new positions and constrained forces.

10.3.6. Constrained positions. When a given molecular configuration $\mathbf{r}^0$ does not satisfy a set of constraints, SHAKE or its derived methods can be used to obtain a *constrained configuration* $\mathbf{r}$, viz. that satisfies the constraints. This is done by using $\mathbf{r}^0$ as reference positions when using, e.g. SHAKE, with $\mathbf{r}^{uc} = \mathbf{r}^0$ as initial configuration:

$$SHAKE(\mathbf{r}^0; \mathbf{r^{uc}}; \mathbf{r}) \tag{10.41}$$

10.3.7. Constrained velocities. Since the unconstrained forces $\mathbf{f}^{uc}$ in Eq. 10.4 will generally contain components along the constraint directions, this will also be true for the velocities obtained from the leap-frog equation.

$$\mathbf{v}_i(t+\Delta t/2) = \mathbf{v}_i(t-\Delta t/2) + m_i^{-1}\mathbf{f}_i^{uc}(t)\Delta t, \tag{10.42}$$

When applying constraints these components have to be removed, that is, the velocities must be shaken. In the leap-frog algorithm the *constrained velocities* are simply determined by inverting the leap-frog equation for the positions

$$\mathbf{r}_i(t+\Delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t+\Delta t/2)\Delta t \tag{10.43}$$

and using constrained positions at times t and $t + \Delta t$,

$$\mathbf{v}_i(t + \Delta t/2) = [\mathbf{r}_i(t + \Delta t) - \mathbf{r}_i(t)]/\Delta t \tag{10.44}$$

If unconstrained positions $\mathbf{r}(t)$ and velocities $\mathbf{v}(t)$ are given at the same time t, the positions are made to satisfy the constraints by applying SHAKE as specified in Sec. 10.3.1. Using the constrained positions $\mathbf{r}(t)$ the velocities are constrained in three steps:

1. Compute
$$\mathbf{r}_i^{uc}(t + \Delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t + \frac{\Delta t}{2})\Delta t \tag{10.45}$$
2. Perform
$$SHAKE(\mathbf{r}(t); \mathbf{r^{uc}}(t + \Delta t); \mathbf{r}(t + \Delta t)) \tag{10.46}$$
3. Obtain the *shaken or constrained velocities* from
$$\mathbf{v}_i(t + \frac{\Delta t}{2}) = [\mathbf{r}_i(t + \Delta t) - \mathbf{r}_i(t)]/\Delta t \tag{10.47}$$

10.3.8. Constrained forces. From Eq. 10.7 it is clear that the *constraint forces* $\mathbf{f}_i^{(c)}(t)$ can be derived by saving the unconstrained configuration $\mathbf{r}^{uc}(t)$ before application of SHAKE and using

$$\mathbf{f}_i^{(c)}(t) \quad = \quad [\mathbf{r}_i(t + \Delta t) - \mathbf{r}_i^{uc}(t + \Delta t)]m_i/(\Delta t)^2 \tag{10.48}$$

We note that the *constrained forces*, that is, the forces that contain no components along the constraints, are given by $\mathbf{f}_i^{uc}$+$\mathbf{f}_i^{(c)}$, since the procedure SHAKE yields a $\mathbf{f}_i^{(c)}$ which compensates the components of $\mathbf{f}_i^{uc}$ along the directions of the constraints.

10.4. Bond-length constraints in the solute

The "solute" part of a molecular topology contains two lists of covalent bonds, one of bonds involving hydrogen atoms, and one involving the other bonds. The two lists of covalent bonds can be used to specify the constraints that are imposed on the molecular system by SHAKE or its derived methods. In addition MD++ allows the user to specify a third list in the topology with selected bonds to be considered. The *switch NTC* in MD++ controls which lists of atom pairs are to be used for constraining. The switches NTF[1] and NTF[2] in the input block FORCE are used to skip the bond interaction terms in the interaction function Eq. 3.6 when the bond lengths are constrained. A sensible choice of the longest time step is:

1. 0.0005 ps when no constraints are used;
2. 0.001 ps when only bonds with hydrogen atoms are constrained;
3. 0.002 ps when all bonds are constrained.

The usage of the switches that control the constraint lists and methods are shown below:
MD++: blockname CONSTRAINT

- NTC: controls application of constraints to bonds
  - 0: no constraints are applied
  - 1: constraints are applied to solvent only
  - 2: constraints are applied to solvent and solute bonds involving hydrogen atoms
  - 3: constraints are applied to solvent and solute bonds
  - 4: constraints are applied to bonds specified in the CONSTRAINT block in topology only
- NTCP: controls the algorithms to apply solute constraints
  - shake(1): apply SHAKE for solute
  - lincs(2): apply LINCS for solute
  - flexshake(3): apply flexible SHAKE for solute
- NTCS: controls the algorithm to apply solvent constraints
  - shake(1): apply SHAKE for solvent
  - lincs(2): apply LINCS for solvent
  - flexshake(3): apply flexible SHAKE for solvent
  - settle(4): apply SETTLE for solvent
  - m_shake(5): apply M_SHAKE for solvent
  - gpu_shake(6): apply M_SHAKE for solvent using GPU

In an MD simulation the solute temperature T is determined by the total kinetic energy of the solute molecules, $\mathcal{K}$ (solutes), and the number of degrees of freedom of the solutes,

$$\mathcal{K}(solutes) = \sum_{\substack{i=1 \\ solutes}}^{\mathcal{N}_a} \tfrac{1}{2} m_i \mathbf{v}_i^2 = \tfrac{1}{2} \mathcal{N}_d(solutes) k_B T \tag{10.49}$$

where k_B is Boltzmann's constant ($k_B = 8.31441\ 10^{-3}$ kJ mol^{-1}K^{-1}) and the number of degrees of freedom of the solutes is denoted by

$$\mathcal{N}_d(solutes) \quad = \quad 3\mathcal{N}_a(solutes) - N^{(c)}(solutes) \tag{10.50}$$

Here the number of solute atoms is $\mathcal{N}_a$(solutes) and the number of solute constraints is $N^{(c)}$ (solutes). The latter is dependent on the value of switch NTC. In an MD simulation, the size of the solute kinetic energy $\mathcal{K}$ (*solute*) and the size and direction of the atomic velocities $\mathbf{v}_i$ will depend on the value of NTC, that is, whether solute constraints are applied or not. This means that if the value of NTC is changed in the course of a MD simulation the velocities have to be rescaled in order to let them correspond to the unchanged temperature T.

For example, an MD job with NTC=3 has produced a final configuration with velocities $\mathbf{v}_i$(NTC=3). If one would continue the simulation with NTC=1 so, if one would assume $\mathbf{v}_i$(NTC=1) = $\mathbf{v}_i$(NTC=3), the temperature T (NTC=1) would become approximately 2/3 T (NTC=3). This can be understood by taking Eq. 10.49 for NTC=1 and NTC=3 and using the equality of the velocities,

$$T(NTC = 1) \quad = \quad \frac{\mathcal{N}_d(solutes; NTC = 3)}{\mathcal{N}_d(solutes; NTC = 1)} \ \cdot \ T(NTC = 3) \tag{10.51}$$

Since in a linear molecule the number of covalent bonds is approximately equal to the number of atoms, one has T (NTC=1) = 2/3 T (NTC=3).

In order to avoid this effect, the *solute velocities should be rescaled upon changing NTC* from NTC to NTC' by a factor $\frac{\mathcal{N}_d(solutes;NTC')}{\mathcal{N}_d(solutes;NTC)}$.

If two atoms i and j are involved in a bond-length constraint $\mathbf{r}_{ij} = d_{ij}^0$ to be imposed by SHAKE (NTC>1) and they are both selected as atoms to be kept at fixed positions, the constraint d_{ij}^0 will be skipped when handling the set of constraints in SHAKE.

10.5. Bond-length and bond-angle constraints in solvent

In Chap. 4-3 it is discussed that solvent molecules of which the topological properties are contained in the "solvent" part of a molecular topology file, are subject to a number of restrictions. One of these is that *a solvent molecule is assumed to be rigid*. Its internal structure is maintained by the application of distance constraint forces between its atoms using SHAKE or its derived methods. It is called with the constraint lengths taken from the SOLVENTCONSTRAINT block in the molecular topology. For examples, see Chap. 3-4.

The solvent temperature is calculated from

$$\mathcal{K}(solvent) = \sum_{\substack{i=1 \\ solvent}}^{\mathcal{N}_a} \tfrac{1}{2} m_i \mathbf{v}_i^2 = \tfrac{1}{2} \mathcal{N}_d(solvent) k_B T \tag{10.52}$$

with

$$\mathcal{N}_d(solvent) = 3\mathcal{N}_a(solvent) - N^{(c)}(solvent) = 6 \tag{10.53}$$

10.6. Dihedral-angle constraints

GROMOS also allows for dihedral-angle constraints which may be used to evaluate a potential of mean force.[100] For dihedral-angle constraints, the derivation of the expressions for the constraint forces $\mathbf{f}^{(c)}$ follows the same lines as that for the distance constraints. However, due to the not very simple dependence of a dihedral angle $\varphi_n(\mathbf{r})$ upon the positions $\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k$ and $\mathbf{r}_l$ of its four constituting atoms i, j, k and l (*i.e.* $i - j - k - l$), the formulae become much more complicated.

Following the definition in Eq. 5.19 we have

$$\varphi_n \doteq \text{sign}(\varphi_n) \ \arccos\left(\frac{\mathbf{r}_{m'j} \cdot \mathbf{r}_{n'k}}{r_{m'j} r_{n'k}}\right) \quad \text{with} \quad -\pi < \varphi_n \leq \pi \tag{10.54}$$

where

$$\mathbf{r}_{m'j} \equiv \mathbf{r}_{ij} \times \mathbf{r}_{kj}, \tag{10.55}$$

$$\mathbf{r}_{n'k} \equiv \mathbf{r}_{kj} \times \mathbf{r}_{kl}, \tag{10.56}$$

and

$$sign(\varphi_n) = sign(\mathbf{r}_{ij} \cdot \mathbf{r}_{n'k}), \tag{10.57}$$

following the IUPAC-IUB convention.[25]

Because of the occurrence of the arccos function in the definition of the dihedral angle φ_n the constraints are to be formulated in terms of $cos(\varphi_n)$. The occurrence of the square root functions in the distances $|\mathbf{r}_{m'j}|$ and $|\mathbf{r}_{n'k}|$ in the denominator of Eq. 10.54 suggests that the use of $cos^2(\varphi_n)$ will simplify the expressions. Thus, we consider a set of N_c dihedral-angle constraints

$$\sigma_n(\varphi_n(\mathbf{r}); \varphi_n^0(\lambda)) \equiv cos^2(\varphi_n(\mathbf{r})) - cos^2(\varphi_n^0(\lambda)) = 0, \quad n = 1, 2, .., N_c \tag{10.58}$$

where the angle $\varphi_n(\mathbf{r})$ is constrained to the λ-dependent value (see Chap. 14)

$$\varphi_n^0(\lambda) = (1-\lambda)\varphi_n^{0\,A} + \lambda\varphi_n^{0\,B}, \tag{10.59}$$

in which $\varphi_n^{0\,A}$ is the φ_n-value in state A and $\varphi_n^{0\,B}$ that in state B.

Newton's equations of motion for $\mathcal{N}_a$ atoms become

$$m_m \frac{d^2\mathbf{r}_i(t)}{\Delta t^2} = -\frac{\partial}{\partial \mathbf{r}_m}\left(\mathcal{V}^{(phys)}(\mathbf{r}) + \sum_{n=1}^{N_c} l_n(t)\sigma_n(\varphi_n(\mathbf{r}); \varphi_n^0(\lambda))\right), \quad m = 1, 2, .., N \tag{10.60}$$

where the Lagrange multipliers $l_k(t)$ are to be determined such that the condition given in Eq. 10.58 is satisfied. The second term on the right in Eq. 10.60 represents the (yet unknown) constraint forces,

$$\begin{aligned} \mathbf{f}_m^{(c)}(t) &= -\sum_{n=1}^{N_c} l_n(t)\frac{\partial\sigma_n(\varphi_n(\mathbf{r}); \varphi_n^0(\lambda))}{\partial \mathbf{r}_m} \\ &= +\sum_{n=1}^{N_c} l_n(t) 2cos(\varphi_n)sin(\varphi_n)\frac{\partial\varphi_n(\mathbf{r})}{\partial \mathbf{r}_m}. \end{aligned} \tag{10.61}$$

where $\frac{\partial\varphi_n}{\partial \mathbf{r}_m}$ is given by[101–103]

$$\begin{aligned} \frac{\partial\varphi_n(\mathbf{r})}{\partial \mathbf{r}_m} &= \delta_{mi}\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{m'j}|^2}\mathbf{r}_{m'j} \\ &+\delta_{mj}\left[\left(\frac{\mathbf{r}_{ij}\cdot\mathbf{r}_{kj}}{|\mathbf{r}_{kj}|^2} - 1\right)\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{m'j}|^2}\mathbf{r}_{m'j} + \frac{\mathbf{r}_{kl}\cdot\mathbf{r}_{kj}}{|\mathbf{r}_{kj}|^2}\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{n'k}|^2}\mathbf{r}_{n'k}\right] \\ &-\delta_{mk}\left[\left(\frac{\mathbf{r}_{kl}\cdot\mathbf{r}_{kj}}{|\mathbf{r}_{kj}|^2} - 1\right)\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{n'k}|^2}\mathbf{r}_{n'k} + \frac{\mathbf{r}_{ij}\cdot\mathbf{r}_{kj}}{|\mathbf{r}_{kj}|^2}\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{m'j}|^2}\mathbf{r}_{m'j}\right] \\ &-\delta_{ml}\frac{|\mathbf{r}_{kj}|}{|\mathbf{r}_{n'k}|^2}\mathbf{r}_{n'k}. \end{aligned} \tag{10.62}$$

To shorten the expressions we denote the four terms in Eq. 10.62, apart from the Kronecker delta's, by $\mathbf{a}_i, \mathbf{a}_j, \mathbf{a}_k$, and $\mathbf{a}_l$, respectively. Then we have

$$\mathbf{f}^{(c)}{}_m(t) = \sum_{k=1}^{N_c} l_k(t) sin(2\varphi_n(t)\left[\delta_{mi}\mathbf{a}_i(t) + \delta_{mj}\mathbf{a}_j(t) + \delta_{mk}\mathbf{a}_k(t) + \delta_{ml}\mathbf{a}_l(t)\right]. \tag{10.63}$$

The leap-frog scheme yields the unconstrained positions $\mathbf{r}_m^{uc}(t_n+\delta t)$ from Eq. 10.6. The constrained positions $\mathbf{r}_m(t_n+\delta t)$ are related to the constraint forces (Eq. 10.63) through Eq. 10.7 and should satisfy the constraint Eq. 10.58,

$$cos^2(\varphi_n(\mathbf{r}(t_n+\delta t)) - cos^2(\varphi_n^0(\lambda)) = 0, \quad k = 1, 2, .., N_c, \tag{10.64}$$

or using Eq. 10.54,

$$\left[\frac{\mathbf{r}_{m'j}(t_n+\delta t)\cdot\mathbf{r}_{n'k}(t_n+\delta t)}{|\mathbf{r}_{m'j}(t_n+\delta t)||\mathbf{r}_{n'k}(t_n+\delta t)|}\right]^2 - cos^2(\varphi_n^0(\lambda)) = 0. \tag{10.65}$$

Since $\mathbf{r}_{m'j}(t_n+\delta t)$ and $\mathbf{r}_{n'k}(t_n+\delta t)$ are each quadratic in the Lagrange multipliers $l_k(t_n)$, both the numerator and the denominator of the left term in Eq. 10.65 contain powers of up to eight of the $l_k(t_n)$. Thus a set of

N_c equations consisting of terms containing up to powers of eight of the unknowns $l_k(t_n)$ is to be solved. As for the case of distance constraints this is achieved by linearizing the equations for each constraint, omitting the coupling between the different constraints (equations), and iterating through all *ncon* equations until the $l_k(t_n)$ converge to a consistent value.

Using Eqs. 10.7 and 10.63 we find for the k-th constraint

$$\begin{aligned}\mathbf{r}_{ij}(t_n+\delta t) &= \mathbf{r}^{uc}_{ij}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\left(m_i^{-1}\mathbf{a}_i(t_n)-m_j^{-1}\mathbf{a}_j(t_n)\right)\end{aligned} \tag{10.66}$$

and likewise for $\mathbf{r}_{kj}(t_n+\delta t)$ and $\mathbf{r}_{kl}(t_n+\delta t)$. Building the cross products in Eqs. 10.55 and 10.56 and linearizing the resulting expressions yields

$$\begin{aligned}\mathbf{r}_{m'j}(t_n+\delta t) &= \mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\\ &\quad \Big[\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\left(m_k^{-1}\mathbf{a}_k(t_n)-m_j^{-1}\mathbf{a}_j(t_n)\right)-\\ &\quad \mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\left(m_i^{-1}\mathbf{a}_i(t_n)-m_j^{-1}\mathbf{a}_j(t_n)\right)\Big]\end{aligned} \tag{10.67}$$

or using a shorter notation $\mathbf{b}_{ijk}(t_n+\delta t)$ for the last factor

$$\begin{aligned}\mathbf{r}_{m'j}(t_n+\delta t) &= \mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\mathbf{b}_{ijk}(t_n,t_n+\delta t),\end{aligned} \tag{10.68}$$

and

$$\begin{aligned}\mathbf{r}_{n'k}(t_n+\delta t) &= \mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\\ &\quad \Big[\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\left(m_k^{-1}\mathbf{a}_k(t_n)-m_l^{-1}\mathbf{a}_l(t_n)\right)-\\ &\quad \mathbf{r}^{uc}_{kl}(t_n+\delta t)\times\left(m_k^{-1}\mathbf{a}_k(t_n)-m_j^{-1}\mathbf{a}_j(t_n)\right)\Big]\end{aligned} \tag{10.69}$$

or using the shorter notation

$$\begin{aligned}\mathbf{r}_{n'k}(t_n+\delta t) &= \mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\mathbf{b}_{jkl}(t_n,t_n+\delta t).\end{aligned} \tag{10.70}$$

The scalar product in the numerator of the first term in Eq. 10.65 becomes after linearization

$$\begin{aligned}\mathbf{r}_{m'j}(t_n+\delta t)\ \cdot\ &\mathbf{r}_{n'k}(t_n+\delta t)=\\ &\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)\cdot\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)\\ &+l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\\ &\Big[\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)\cdot\mathbf{b}_{jkl}(t_n,t_n+\delta t)+\\ &\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)\cdot\mathbf{b}_{ijk}(t_n,t_n+\delta t)\Big]\end{aligned} \tag{10.71}$$

or in a shorter notation

$$\begin{aligned}&\mathbf{r}_{m'j}(t_n+\delta t)\cdot\mathbf{r}_{n'k}(t_n+\delta t)=c_{ijkl}(t_n+\delta t)\\ &\quad +l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2d_{ijkl}(t_n,t_n+\delta t).\end{aligned} \tag{10.72}$$

The square becomes after linearization

$$\begin{aligned}&\Big(\mathbf{r}_{m'j}(t_n+\delta t)\cdot\mathbf{r}_{n'k}(t_n+\delta t)\Big)^2=\left(c_{ijkl}(t_n+\delta t)\right)^2\\ &+2l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2c_{ijkl}(t_n+\delta t)d_{ijkl}(t_n,t_n+\delta t).\end{aligned} \tag{10.73}$$

The factors in the denominator of the first term in Eq. 10.65 become

$$\begin{aligned}|\mathbf{r}_{m'j}(t_n+\delta t)|^2 &= \left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)^2\\ &\quad +2l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\\ &\quad \left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)\cdot\mathbf{b}_{ijk}(t_n+\delta t)\end{aligned} \tag{10.74}$$

and

$$\begin{aligned}|\mathbf{r}_{n'k}(t_n+\delta t)|^2 &= \left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)^2\\ &\quad +2l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2\\ &\quad \left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)\cdot\mathbf{b}_{jkl}(t_n+\delta t).\end{aligned} \tag{10.75}$$

The linearized denominator of the first term in Eq. 10.65 is then

$$
\begin{aligned}
&|\mathbf{r}_{m'j}(t_n+\delta t)|^2|\mathbf{r}_{n'k}(t_n+\delta t)|^2 = \\
&\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)^2\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)^2 \\
&+2l_k(t_n)sin(2\varphi_n(t_n))(\delta t)^2 \\
&\Bigg[\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)^2 \\
&\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)\cdot\mathbf{b}_{jkl}(t_n,t_n+\delta t) \\
&+\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)^2 \\
&\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)\cdot\mathbf{b}_{ijk}(t_n,t_n+\delta t)\Bigg]
\end{aligned}
\tag{10.76}
$$

Finally, the equation for the Lagrange multiplier of the k-th constraint becomes

$$
\begin{aligned}
l_k(t_n) \quad = \quad &\Bigg[cos^2(\varphi^0_n(\lambda))\left(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t)\right)^2 \\
&\left(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t)\right)^2-\left(c_{ijkl}(t_n+\delta t)\right)^2\Bigg] \\
&\Bigg[2sin(2\varphi_n(t_n))(\delta t)^2\Big[c_{ijkl}(t_n+\delta t)d_{ijkl}(t_n+\delta t) \\
&-cos^2(\varphi^0_n(\lambda))\Big((\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t))^2 \\
&(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t))\cdot\mathbf{b}_{jkl}(t_n,t_n+\delta t) \\
&+(\mathbf{r}^{uc}_{kj}(t_n+\delta t)\times\mathbf{r}^{uc}_{kl}(t_n+\delta t))^2 \\
&(\mathbf{r}^{uc}_{ij}(t_n+\delta t)\times\mathbf{r}^{uc}_{kj}(t_n+\delta t))\cdot\mathbf{b}_{ijk}(t_n,t_n+\delta t)\Big)\Big]\Bigg]^{-1}
\end{aligned}
\tag{10.77}
$$

The derivative of the contribution of the constraint forces to the free energy for the k-th constraint becomes

$$
\frac{d\mathbf{f}^{(c)}_k(\lambda)}{d\lambda} = <l_k>_\lambda\, sin(2\varphi^0_n(\lambda))\left(\varphi^{0\,B}_n-\varphi^{0\,A}_n\right). \tag{10.78}
$$

We note that the expressions given here for the application of dihedral-angle constraints are different from the formalism presented in Ref.[104], which is based on matrix inversion.

10.7. Translational and rotational constraints

In MD++ it is possible to apply rotational and translational constraints. These constraints, if jointly applied, are called *roto-translational constraints*. It may be advantageous to perform simulations with translational and/or rotational constraints, because: (*i*) the simulation of non-spherical solutes can be done in computational boxes whose shape is optimally adjusted to the shape of the solute, thus reducing the required amount of solvent molecules; (*ii*) it is possible to prevent the coupling of internal degrees of freedom with the roto-translational ones, which may speed up equilibration of a system; (*iii*) many interesting simulations can be performed, such as *e.g.* the modelling of solvent molecules as Langevin dipoles at fixed grid points, or ligand-binding studies involving ligands exempt of rotational degrees of freedom; (*iv*) simulations *in vacuo* may benefit from the application of statistically-mechanically rigorous roto-translational constraints rather than merely setting the angular momentum to zero.

A *translational constraint* $\mathcal{C}_t$ on a set of atoms $\mathcal{G}_{Tj}$ consisting of atoms $i=1,...,N_{\mathcal{G}_{Tj}}$ is defined as

$$
\mathcal{C}_t(\mathcal{G}_{Tj}): \quad \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i\mathbf{r}_i(t) = const \quad , \tag{10.79}
$$

where m_i denotes the atomic masses and $\mathbf{r}_i(t)$ the atomic Cartesian coordinates at time t in the fixed orthonormal laboratory frame. From Eq. 10.79 it is obvious that $\mathcal{C}_t(\mathcal{G}_{Tj})$ fixes the position (Cartesian

coordinates in the fixed orthonormal laboratory frame) of the centre-of-mass (COM) of $\mathcal{G}_{Tj}$,

$$\mathbf{r}_{COM}(t;\mathcal{G}_{Tj}) = \frac{1}{\sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i} \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i \mathbf{r}_i(t) \quad , \tag{10.80}$$

during the simulation. In a simulation involving constant pressure (*i.e.*, involving a variation of the volume of the computational box), it is the oblique Cartesian coordinates of the COM, $\tilde{\mathbf{r}}_{COM}(t;\mathcal{G}_{Tj})$ which are fixed, *i.e.*

$$\mathcal{C}_t(\mathcal{G}_{Tj}) : \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i \mathbf{L}^{-1}(t)\mathbf{r}_i(t) = const \quad , \tag{10.81}$$

where $\mathbf{L}(t)$ is the box matrix at time t, and

$$\tilde{\mathbf{r}}_{COM}(t;\mathcal{G}_{Tj}) = \frac{1}{\sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i} \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i \mathbf{L}^{-1}(t)\mathbf{r}_i(t) \quad . \tag{10.82}$$

Due to its more general nature, Eq. 10.81, rather than Eq. 10.79 is used in the following to describe translational constraints.
A *rotational constraint* $\mathcal{C}_r$ on a set of atoms $\mathcal{G}_{Tj}$ consisting of atoms $i = 1, ..., N_{\mathcal{G}_{Tj}}$ is defined as

$$\mathcal{C}_r(\mathcal{G}_{Tj}) : \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i[\ \mathbf{q}^o{}_i \times \mathbf{q}_i(t)] = \mathbf{0} \quad , \tag{10.83}$$

where $\mathbf{q}^o{}_i$ and $\mathbf{q}_i(t)$ are reference and instantaneous atomic Cartesian coordinates, respectively, in a a local orthonormal frame whose origin lies at the COM of $\mathcal{G}_{Tj}$. From Eq. 10.83 it is obvious that $\mathcal{C}_r(\mathcal{G}_{Tj})$ fixes the rotational orientation of $\mathcal{G}_{Tj}$ with respect to the reference orientation given by the set of $\mathbf{q}^o{}_i$ during the simulation.

The implementation of Eq. 10.81 and Eq. 10.83 is done in a Lagrange-multiplier formalism relying on six linear holonomic constraints captured by the corresponding $\mathcal{C}_t(\mathcal{G}_{Tj})$ and $\mathcal{C}_r(\mathcal{G}_{Tj})$, as described in Ref.[105]. The gradients of these constraint equations, $\frac{\partial \mathcal{C}_t{}^{(k)}(\mathcal{G}_{Tj})}{\partial q_{i,\alpha}}(t)$ and $\frac{\partial \mathcal{C}_r{}^{(k)}(\mathcal{G}_{Tj})}{\partial q_{i,\alpha}}(t)$, where $k = 1, 2, 3$, i denotes an atom of $\mathcal{G}_{Tj}$ and α denotes a Cartesian vector component (x, y or z), give the constraint forces as linear combinations thereof (in conjunction with the corresponding Lagrange multipliers as the coefficients of the gradients in this linear combination).
More precisely,

$$\mathcal{C}_t{}^{(1)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i q_{i,x}(t) = 0 \quad , \tag{10.84}$$

$$\mathcal{C}_t{}^{(2)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i q_{i,y}(t) = 0 \quad , \tag{10.85}$$

$$\mathcal{C}_t{}^{(3)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i q_{i,z}(t) = 0 \quad , \tag{10.86}$$

$$\mathcal{C}_r{}^{(1)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i [q^o_{i,y} q_{i,z}(t) - q^o_{i,z} q_{i,y}(t)] = 0 \quad , \tag{10.87}$$

$$\mathcal{C}_r{}^{(2)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i [q^o_{i,z} q_{i,x}(t) - q^o_{i,x} q_{i,z}(t)] = 0 \quad , \tag{10.88}$$

$$\mathcal{C}_r{}^{(3)}(t;\mathcal{G}_{Tj}) \doteq \sum_{i=1}^{N_{\mathcal{G}_{Tj}}} m_i [q^o_{i,x} q_{i,y}(t) - q^o_{i,y} q_{i,x}(t)] = 0 \quad , \tag{10.89}$$

and the resulting constraint forces acting on atom i are ${\mathbf{F}^{(\mathcal{C}_t)}}_i[t;\mathcal{C}_t(\mathcal{G}_{T\,j})]$ and ${\mathbf{F}^{(\mathcal{C}_r)}}_i[t;\mathcal{C}_r(\mathcal{G}_{T\,j})]$, with components $f_{i,\alpha}^{(\mathcal{C}_t)}[t;\mathcal{C}_t(\mathcal{G}_{T\,j})]$, $f_{i,\alpha}^{(\mathcal{C}_r)}[t;\mathcal{C}_r(\mathcal{G}_{T\,j})]$ given by

$$f_{i,\alpha}^{(\mathcal{C}_t)}[t;\mathcal{C}_t(\mathcal{G}_{T\,j})] = \sum_{k=1}^{3} \lambda_{t,k}(t) \frac{\partial {\mathcal{C}_t}^{(k)}(t;\mathcal{G}_{T\,j})}{\partial q_{i,\alpha}}(t) \tag{10.90}$$

and

$$f_{i,\alpha}^{(\mathcal{C}_r)}[t;\mathcal{C}_r(\mathcal{G}_{T\,j})] = \sum_{k=1}^{3} \lambda_{r,k}(t) \frac{\partial {\mathcal{C}_r}^{(k)}(t;\mathcal{G}_{T\,j})}{\partial q_{i,\alpha}}(t) \quad , \tag{10.91}$$

where the Lagrange multipliers $\boldsymbol{\lambda}_t = (\lambda_{t,1}\ \lambda_{t,2}\ \lambda_{t,3})^T$ and $\boldsymbol{\lambda}_r = (\lambda_{r,1}\ \lambda_{r,2}\ \lambda_{r,3})^T$ (with T denoting the transpose) are given as

$$\boldsymbol{\lambda}_t = -\boldsymbol{\Theta}_t^{-1}\mathbf{c}_t \tag{10.92}$$

and

$$\boldsymbol{\lambda}_r = -\boldsymbol{\Theta}_r^{-1}\mathbf{c}_r \quad , \tag{10.93}$$

with

$$\boldsymbol{\Theta}_t = \begin{pmatrix} \theta_{t,11} & 0 & 0 \\ 0 & \theta_{t,22} & 0 \\ 0 & 0 & \theta_{t,33} \end{pmatrix} \tag{10.94}$$

and

$$\boldsymbol{\Theta}_r = \begin{pmatrix} \theta_{r,11} & \theta_{r,12} & \theta_{r,13} \\ \theta_{r,21} & \theta_{r,22} & \theta_{r,23} \\ \theta_{r,31} & \theta_{r,32} & \theta_{r,33} \end{pmatrix} \quad . \tag{10.95}$$

The matrix elements $\theta_{t,ii}$ and $\theta_{r,ij}$ can be shown to be

$$\theta_{t,11} = \theta_{t,22} = \theta_{t,33} = (\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i \quad , \tag{10.96}$$

$$\theta_{r,11} = (\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i ({q_{i,y}^o}^2 + {q_{i,z}^o}^2) \quad , \tag{10.97}$$

$$\theta_{r,12} = \theta_{r,21} = -(\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i q_{i,x}^o q_{i,y}^o \quad , \tag{10.98}$$

$$\theta_{r,13} = \theta_{r,31} = -(\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i q_{i,x}^o q_{i,z}^o \quad , \tag{10.99}$$

$$\theta_{r,22} = (\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i ({q_{i,x}^o}^2 + {q_{i,z}^o}^2) \quad , \tag{10.100}$$

$$\theta_{r,23} = \theta_{r,32} = -(\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i q_{i,y}^o q_{i,z}^o \quad , \tag{10.101}$$

$$\theta_{r,33} = (\Delta t)^2 \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} m_i ({q_{i,x}^o}^2 + {q_{i,y}^o}^2) \quad , \tag{10.102}$$

where Δt is the integration time step. Note that $\theta_{r,ij}$ is dependent on the set of reference coordinates ${\mathbf{q}^o}_i$. If simulations are performed at constant pressure, these reference coordinates are scaled along with all other box-dependent quantities (box matrix, atomic Cartesian coordinates) by the appropriate scaling factors.

That is, $\mathbf{\Theta}_r = \mathbf{\Theta}_r(t)$, *i.e.* will change during the simulation.
The vectors $\mathbf{c}_t = (c_{t,1}\ c_{t,2}\ c_{t,3})^T$ and $\mathbf{c}_r = (c_{r,1}\ c_{r,2}\ c_{r,3})^T$ can be shown to be

$$c_{t,k} = \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} \sum_{\alpha=x,y,z} \frac{\partial \mathcal{C}_t{}^{(k)}(t;\mathcal{G}_{T\,j})}{\partial q_{i,\alpha}} \Delta r_{i,\alpha} \tag{10.103}$$

and

$$c_{r,k} = \sum_{i=1}^{N_{\mathcal{G}_{T\,j}}} \sum_{\alpha=x,y,z} \frac{\partial \mathcal{C}_r{}^{(k)}(t;\mathcal{G}_{T\,j})}{\partial q_{i,\alpha}} \Delta r_{i,\alpha} \quad , \tag{10.104}$$

where $k = 1, 2, 3$ and $\Delta r_{i,\alpha}$ is the α-component of the displacement vector $\Delta\mathbf{r}_i$ of atom i, with the displacement being that coordinate change since the last time step which the corresponding constraint force is correcting for, *e.g.*: (*i*) the displacement due to the conservative force in the current time step, if no other constraint forces are applied; (*ii*) the displacement due to the conservative force and a subsequent SHAKE force in the current time step, if bond constraints are enforced with the SHAKE algorithm; (*iii*) the displacement due to the conservative force and any sum of subsequently, iteratively-applied other constraint forces in the current time step.

Any input settings concerning roto-translational constraints are specified in the ROTTRANS and INITIALISE blocks of the MD++ imd file.
Roto-translational constraints are always applied on the first specified number of atoms. It is not possible to separately apply a translational or rotational constraint only. Thus, for instance, one can apply a roto-translational constraint to a solvated biomolecule (one solute molecule).
Note that, upon starting a new MD simulation, a rotational constraint on a certain temperature group can be initialised in either of two ways: The reference coordinates $\mathbf{q}_i^o$ may be read from the ROTOTRANSREFPOS block in the configuration file, or may be computed from scratch based on the set of initial coordinates given in the configuration file (POSITION block). This decision is specified in the INITIALISE block of the MD++ input file.

CHAPTER 11

Energy Minimization

11.1. Introduction

Energy minimization (EM) with an empirical energy function such as Eq. 3.4 is a widely used tool for obtaining low-energy configurations of a molecular system. Various function minimization methods can be used, which can be classified as follows:

1. *Direct search methods*, requiring only function values. They converge slowly and are therefore not considered here.
2. *Gradient methods*, requiring function and derivative values. These methods fall into three subclasses:
   a. The *steepest descent method* (SDEM) performs well far from a minimum, but converges slowly near a minimum, or when searching in a long, thin, curving valley. It is a robust method, which is easy to implement.
   b. The *conjugate gradient method*[106] (CGEM) which searches along directions corresponding to the local quadratic approximation to the function, usually converges superlinearly. Because it is the most rapidly converging minimizer that does not require manipulation and storage of matrices of dimension equal to the number of degrees of freedom, it appears to be most appropriate for very large systems, like macromolecules.
   c. The *variable metric or quasi-Newton methods*, which use various approximations to the inverse of the Hessian matrix (matrix of second partial derivatives), are also quadratically convergent, but they require storage space for the inverse Hessian and time for its manipulation. Hence, they are less suited for application to large systems.
3. *Second-order methods*, requiring function, derivative and Hessian matrix. These methods are not well suited for application to large systems for the same reason as mentioned under 2c.

For references to the different methods we refer to ref.[107]. The method of steepest descents is discussed in Sec. 11.2, the conjugate gradient technique in Sec. 11.3. In Sec. 11.4 and 11.5 it is described how the SHAKE method for constraining bond lengths and/or bond angles can be incorporated in the steepest descent and conjugate gradient energy minimization algorithm.

When applying EM algorithms one searches for a minimum energy configuration of a system by moving (approximately) along the gradient of the potential energy through configuration space,

$$\Delta \mathbf{r}_i \sim -\frac{\partial}{\partial r_i} V(\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_N), \tag{11.1}$$

where $\Delta \mathbf{r}_i$ denotes the shift in position of atom i. Since in this way one basically moves only downhill over the energy hypersurface, EM yields only a *local* minimum energy configuration, which is generally not far from the initial one. Using formulae like Eq. 11.1 crossing of energy barriers is impossible. A more efficient way to find low energy configurations is to apply molecular dynamics (MD). The available kinetic energy may be used to pass over energy barriers which are not much higher than $k_B T$ (k_B = Boltzmann's constant and T = absolute temperature). It has been shown[108] that MD at elevated temperatures can be used to generate a variety of configurations. Therefore, MD searches a larger part of configuration space for an energy minimum and generally ends up in a lower energy minimum than an ordinary energy minimizer does[109]. However, the application of MD starting from a highly strained, very high potential energy configuration is not recommended, since the immediate conversion of potential energy into kinetic energy will raise the temperature to unacceptably high values. In that case, one should first perform a number of EM steps in order to reduce the high potential energy of the system. When the high potential energy is due to close non-bonded contacts or stretched bond lengths or bent bond angles in a molecular system, ten to fifty EM steps generally suffice to reduce the potential energy to values which are normal at room temperature.

The configuration at the n-th minimization step is denoted by $\boldsymbol{r}(t_n) \equiv (\mathbf{r}_1(t_n), \mathbf{r}_2(t_n), \ldots, \mathbf{r}_N(t_n))$, where t_n is used in analogy with MD simulation and the configuration consists of N atoms. The *scalar product of two configurations* $\boldsymbol{r}$ and $\boldsymbol{r}'$ is denoted by

$$< \boldsymbol{r} | \boldsymbol{r}' > \quad \equiv \quad \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{r}'_i \tag{11.2}$$

When a set of atoms is to be kept fixed, their positions $\mathbf{r}_i$ are not changed and the forces (negative gradients) f_i on them are kept equal to zero at each minimization step. Their inverse masses are set to zero in order to immobilize these atoms when their position might be up for resetting by the procedure SHAKE in case a fixed atom is involved in a constraint to a non-fixed atom.

11.2. Steepest-descent minimization

Energy minimization by the *steepest-descent (*SDEM*) algorithm* is simple. The *computational scheme* for the (n+1)-th minimization step reads,

1. Calculate the forces $\boldsymbol{f}(t_n) = \boldsymbol{f}(\boldsymbol{r}(t_n))$ from the interaction function $V(\boldsymbol{r})$ (see Eq. 3.4) using expression Eq. 2.10 and the configuration $\boldsymbol{r}(t_n)$.
2. Compute the next configuration from

   $$\boldsymbol{r}(t_{n+1}) = \boldsymbol{r}(t_n) + \Delta x < \boldsymbol{f}(t_n) | \boldsymbol{f}(t_n) >^{-\frac{1}{2}} \boldsymbol{f}(t_n) \tag{11.3}$$

   where the step size is denoted by Δx.

SDEM is selected by setting switch NTEM=1 in the input of MD++. The initial step size Δx is to be specified in DX0. As long as the potential energy decreases, the step size Δx is increased by 20% per step. If the potential energy increases, Δx is halved. The growth of the step size can be limited by specifying a maximum value, DXM. The energy minimization is terminated when the number of EM steps reaches the value NSTLIM or when the potential energy change between two subsequent steps is less than the value DELE. However, one may specify a minimum number of steps to take using parameter NMIN. The final configuration is saved.

At every NTPR-th minimization step a number of *quantities* (step number (TIMESTEP), various energies (ENERGY block), etc. see Sec. 12.7) are *printed*. In addition, coordinate or energy trajectories may be written (see Vol. 4).

11.3. Conjugate-gradient minimization

The *conjugate gradient (*CGEM*) algorithm* can be summarized as follows. It is started by calculating the forces or negative gradients, $\boldsymbol{f}(t_0) = \boldsymbol{f}(\boldsymbol{r}(t_0))$, from the interaction function $V(\boldsymbol{r})$ (see Eq. 3.4) using expression Eq. 2.10 and the initial configuration $\boldsymbol{r}(t_0)$, and by taking the first search directions $\boldsymbol{p}(t_0) \equiv (\mathbf{p}_1(t_0), \mathbf{p}_2(t_0), \ldots, \mathbf{p}_N(t_0))$ along the negative gradients, that is,

$$\boldsymbol{p}(t_0) = \boldsymbol{f}(t_0) \tag{11.4}$$

The *computational scheme* for the (n+1)-th minimization step reads,

1. Find the minimum of the potential energy $\mathcal{V}(\boldsymbol{r})$ on the line through $\boldsymbol{r}(t_n)$ in the direction $\boldsymbol{p}(t_n)$, that is, determine s_{min} such that

   $$\mathcal{V}(\boldsymbol{r}(t_n) + s_{min}\boldsymbol{p}(t_n)) \tag{11.5}$$

   finds its minimum, or equivalently, such that

   $$g(t_n; s_{min}) \equiv < \boldsymbol{p}(t_n) | \boldsymbol{f}(\boldsymbol{r}(t_n) + s_{min}\boldsymbol{p}(t_n)) >= 0 \tag{11.6}$$

   The determination s_{min} is done in two stages: the first establishes bounds a and b on $s_{min'}(a < s_{min} \leqslant b)$, and the second interpolates its value in the interval (a, b).

a. As a first guess for the bounds, take $a = 0$, and $b = \Delta x < \boldsymbol{p}(t_n)|\boldsymbol{p}(t_n) >^{-\frac{1}{2}}$. The step size Δx should be chosen such that no iterations are required when establishing the bound b. We already know the energy $\mathcal{V}(\boldsymbol{r}(t_n)+a\boldsymbol{p}(t_n)) = V(\boldsymbol{r}(t_n))$ and also the forces $\boldsymbol{f}(\boldsymbol{r}(t_n)+a\boldsymbol{p}(t_n)) = \boldsymbol{f}(\boldsymbol{r}(t_n))$, so $g(t_n; a)$ is easily obtained. The same quantities have to be examined at b, so we compute $\boldsymbol{r}(t_n) + b\boldsymbol{p}(t_n)$ and evaluate the energy $\mathcal{V}(\boldsymbol{r}(t_n) + b\boldsymbol{p}(t_n))$ and the forces $\boldsymbol{f}(\boldsymbol{r}(t_n) + b\boldsymbol{p}(t_n))$, and compute $g(t_n; b)$. The value b is accepted as upper bound on s_{min} if

$$g(t_n; b) \quad < \quad 0 \tag{11.7}$$

or

$$\mathcal{V}(\boldsymbol{r}(t_n) + b\boldsymbol{p}(t_n)) \geq \mathcal{V}(\boldsymbol{r}(t_n) + a\boldsymbol{p}(t_n)) \tag{11.8}$$

Otherwise s_{min} lies beyond b, so b is too small and so is Δx and the process of looking for bounds on s_{min} is to be repeated with doubled b.

b. Use a cubic interpolation formula in order to estimate s_{min} in the interval $(a,\ b)$,

$$s_{min} \quad = \quad b - \frac{[W - Z - g(t_n; b)][b - a]}{[g(t_n; a) - g(t_n; b) + 2W]} \tag{11.9}$$

in which

$$W \quad \equiv \quad \left[\ Z^2 - g(t_n; a)g(t_n; b)\right]^{1/2} \tag{11.10}$$

and

$$Z \quad \equiv \quad \frac{3[\mathcal{V}(\boldsymbol{r}(t_n) + a\boldsymbol{p}(t_n)) - \mathcal{V}(\boldsymbol{r}(t_n) + b\boldsymbol{p}(t_n))]}{[b - a]} - g(t_n; a) - g(t_n; b). \tag{11.11}$$

c. In order to improve the estimate of $s_{min,0}$ and allow for tighter convergence, it is accepted only if RMSD between the new and the previous estimate is smaller than the threshold Δr_{thres}, that is if

$$|s_{min,i+1} - s_{min,i}|\sqrt{\frac{\langle \boldsymbol{p}(t_n)|\boldsymbol{p}(t_n)\rangle}{N}} \quad < \quad \Delta r_{thres} \tag{11.12}$$

where for first iteration $s_{min,0} = a$. If RMSD is higher than Δr_{thres}, the energy $\mathcal{V}(\boldsymbol{r}(t_n) + s_{min,i+1}\boldsymbol{p}(t_n))$, forces $\boldsymbol{f}(\boldsymbol{r}(t_n)+s_{min,i+1}\boldsymbol{p}(t_n))$ and $g(t_n; s_{min,i+1})$ are calculated. If $g(t_n; s_{min,i+1})$ is less than 0 and the energy is lower than $\mathcal{V}(\boldsymbol{r}(t_n) + a\boldsymbol{p}(t_n))$, the upper bound b is moved to $s_{min,i+1}$, that is $b = s_{min,i+1}$, otherwise $a = s_{min,i+1}$. Steps **1b** and **1c** are repeated until RMSD is below Δr_{thres} or maximum number of steps i_{max} is reached. Δr_{thres} and i_{max} can be set using parameters CGIC and CGIM, respectively. By setting $i_{max} = 1$ the first estimate of s_{min} is always accepted. $i_{max} > 1$ is useful for tight convergence. With the accepted s_{min} the new configuration is

$$\boldsymbol{r}(t_{n+1}) \quad = \quad \boldsymbol{r}(t_n) + s_{\min}\boldsymbol{p}(t_n) \tag{11.13}$$

2. Calculate the new energy $\mathcal{V}(\boldsymbol{r}(t_{n+1}))$ and the forces $\boldsymbol{f}(t_{n+1}) = \boldsymbol{f}(\boldsymbol{r}(t_{n+1}))$ from the interaction function $\mathcal{V}(\boldsymbol{r})$.

3. Determine the new search directions $\boldsymbol{p}(t_{n+1})$ from

$$\boldsymbol{p}(t_{n+1}) \quad = \quad \boldsymbol{f}(t_{n+1}) + \beta_n\boldsymbol{p}(t_n) \tag{11.14}$$

where in Fletcher-Reeves method (FRCG)[106]

$$\beta_n \quad = \quad \frac{< \boldsymbol{f}(t_{n+1})|\boldsymbol{f}(t_{n+1}) >}{< \boldsymbol{f}(t_n)|\boldsymbol{f}(t_n) >}. \tag{11.15}$$

or in Polak-Ribiére method (PRCG)[110]

$$\beta_n = \frac{< \boldsymbol{f}(t_{n+1}) - \boldsymbol{f}(t_n) | \boldsymbol{f}(t_{n+1}) >}{< \boldsymbol{f}(t_n) | \boldsymbol{f}(t_n) >}. \tag{11.16}$$

Each new direction of search is partly determined by previous search directions. The weight depends on the relative size of the forces at $\boldsymbol{r}(t_{n+1})$ and $\boldsymbol{r}(t_n)$. Although both CGEM algorithms perform at least two function evaluations of $\mathcal{V}(\boldsymbol{r})$ per minimization step, they are generally more efficient than the SDEM algorithm, which performs one function evaluation per step.[107] However, the latter algorithm is more robust.

CGEM energy minimization is selected by setting the switch NTEM = 2 for FRCG or NTEM = 3 for PRCG in the input of program MD++. The CGEM step size Δx is to be specified in DX0 and should be carefully chosen. It should be just large enough to avoid interval doubling iterations when establishing bounds a and b on s_{min} (step **1a**), but small enough to allow for a good estimate of s_{min} by interpolation (step **1b**). If the estimated s_{min} was found within first 10% of the search interval, Δx is decreased by 10% in the next step. If s_{min} was found beyond the initial b, next Δx is increased by 10%. The growth of Δx can be limited by specifying a maximum value, DXM. If the energy decreases slowly near a local minimum, one may try to reach it faster by restarting the CGEM algorithm with a smaller Δx value. It may be sometimes useful to limit the number of CGEM steps taken with a given series of search direction vectors. This can be achieved by setting the variable NCYC a finite (<NSTLIM) value: after every NCYC CGEM minimization steps the algorithm will use

$$\beta_n = 0 \tag{11.17}$$

instead of β_n given by 11.15 or 11.16, thereby discarding the contribution of previous search directions to $\boldsymbol{p}(t_{n+1})$ in Eq. 11.13.

So, for NCYC = 1 the CGEM algorithm reduces to a SDEM algorithm with cubic interpolation. With NCYC = 0 the search direction vectors are reset only if the energy grows in the search direction, that is if

$$g(t_n; a) < 0 \tag{11.18}$$

The energy minimization is terminated when the number of EM steps reaches the value NSTLIM or when the RMS force is less than the value DELE

$$f_{RMS}(t_n) = \sqrt{\frac{\langle \boldsymbol{f}(t_n) | \boldsymbol{f}(t_n) \rangle}{N}} < \text{DELE} \tag{11.19}$$

However, one may also specify a minimum number of steps to take using parameter NMIN. The final configuration is saved. In addition, coordinate or energy trajectories may be written (see Vol. 4).

11.4. Steepest-descent minimization with constraints (SHAKE)

The essential feature of the SHAKE method for conserving constraints is that after each minimization step, the constraints are satisfied by adding displacement vectors to the position vectors of the atoms that result from an unconstrained minimization step. The added displacement vectors are determined such that the constraints are satisfied at the final positions (see Sec. 10.3).

The procedure *SHAKE* can be directly incorporated into the *steepest-descent* energy minimization *algorithm* in the following way. The initial configuration $\boldsymbol{r}(t_0)$ must be made satisfy the constraints as discussed in Sec. 10.3.1. Shaking of the initial configuration is selected by taking NTISHK = 1 in the input file. Then, the *computational scheme* for the (n+1)-th constrained minimization step reads,

1. Calculate the unconstrained forces $\mathbf{f}^{uc}(t_n) = \mathbf{f}^{uc}(\boldsymbol{r}(t_n))$ from the interaction function $\mathcal{V}(\boldsymbol{r})$ (see Eq. 3.4), from which the terms acting *only* along the constrained degrees of freedom are excluded (e.g. the bond length interactions), using Eq. 2.8 and the (shaken) configuration $\boldsymbol{r}(t_n)$.

2. Determine the unconstrained positions at step t_{n+1},

$$\mathbf{r}^{uc}(t_{n+1}) \quad = \quad \boldsymbol{r}(t_n) + \Delta x < \mathbf{f}^{uc}(t_n)|\mathbf{f}(t_n) >^{-1/2} \mathbf{f}^{uc}(t_n) \tag{11.20}$$

where the step size is denoted by Δx. The positions $\mathbf{r}^{uc}(t_{n+1})$ do not, in general, satisfy the constraints, as the forces normally contain components in the constrained directions.

3. The positions are made satisfy the constraints by performing

$$SHAKE(\boldsymbol{r}(t_n); \mathbf{r}^{uc}(t_{n+1}); \boldsymbol{r}(t_{n+1})) \tag{11.21}$$

Most of the parameters of the SDEM algorithm with SHAKE are identical to those discussed in Sec. 11.2 for the unconstrained SDEM algorithm. An additional feature is that the step size Δx is halved when the number of iterations in SHAKE in Eq. 11.21 exceeds 100. The constrained forces $\boldsymbol{f}(t_n) = \mathbf{f}^{uc}(t_n) + \mathbf{f}^{(c)}(t_n)$ can be obtained from solving Eq. 11.20 for $\mathbf{f}^{uc}(t_n)$ and using $\boldsymbol{r}(t_{n+1})$ instead of $\mathbf{r}^{uc}(t_{n+1})$,

$$\boldsymbol{f}(t_n) \quad = \quad \frac{\boldsymbol{r}(t_{n+1}) - \boldsymbol{r}(t_n)}{\Delta x < \mathbf{f}^{uc}(t_n)|\mathbf{f}^{uc}(t_n) >^{-1/2}}. \tag{11.22}$$

11.5. Conjugate-gradients minimization with constraints (SHAKE)

Incorporation of *SHAKE* into the *conjugate-gradients algorithm* is more complex than in the case of steepest descent, since there only the positions $\mathbf{r}^{uc}(t_{n+1})$ had to be shaken. Here, the search direction, which is composed of the force direction and previous search directions, also must be chosen such that it does not contain components along the constraints. Hence, $\mathbf{f}^{uc}(t_{n+1})$ and $\mathbf{p}^{uc}(t_{n+1})$ have to be shaken too.

The initial configuration $\boldsymbol{r}(t_0)$ must be made satisfy the constraints as discussed in Sec. 10.3.1. Shaking of the initial configuration is selected by taking NTISHK = 1 in the input block INITIALIZE. The initial unconstrained forces $\mathbf{f}^{uc}(t_0)$ can be shaken by the procedure, which was used to remove components along the constraint directions from the velocities in Sec. 10.3.7,

A. Determine

$$\mathbf{r}^{uc}(t_1) \quad = \quad \mathbf{r}(t_0) + \Delta x^{uc} \mathbf{f}^{uc}(t_0) \tag{11.23}$$

where $\Delta x^{uc} = \Delta x \, \langle \mathbf{f}^{uc}(t_0)|\mathbf{f}^{uc}(t_0)\rangle^{-\frac{1}{2}}$ and Δx is the conjugate-gradients step size.

B. Perform

$$SHAKE(\mathbf{r}(t_0); \mathbf{r}^{uc}(t_1); \mathbf{r}(t_1)). \tag{11.24}$$

C. Obtain the constrained or shaken forces $\mathbf{f}(t_0)$ from

$$\mathbf{f}(t_0) \quad = \quad \frac{\mathbf{r}(t_1) - \mathbf{r}(t_0)}{\Delta x^{uc}}. \tag{11.25}$$

The initial search direction is taken along $\boldsymbol{f}(t_0)$, that is, $\mathbf{p}(t_0) = \boldsymbol{f}(t_0)$. Then, the *computational scheme* for the (n+1)-th minimization step reads,

1. Find the minimum of the potential energy $\mathcal{V}(\boldsymbol{r})$ on the line through $\boldsymbol{r}(t_n)$ in the direction $\mathbf{p}(t_n)$, that is, determine s_{min} such that

$$\mathcal{V}(\boldsymbol{r}(t_n) + s_{\min}\mathbf{p}(t_n)) \tag{11.26}$$

finds its minimum, or such that

$$g(t_n; s_{\min}) \equiv \langle \mathbf{p}(t_n)|\mathbf{f}(r(t_n) + s_{\min}\mathbf{p}(t_n))\rangle = 0 \quad . \tag{11.27}$$

When the components along the constraints have been eliminated from $\mathbf{p}(t_n)$ and $\boldsymbol{f}(\boldsymbol{r}(t_n) + s_{min}\mathbf{p}(t_n)$, the conditions Eq. 11.26 and Eq. 11.27 are *not* equivalent and $\mathcal{V}(\boldsymbol{r}(t_n) + s_{min}\mathbf{p}(t_n))$ may have a minimum with $g(t_n; s_{min}) \neq 0$.

The determination of s_{min} is done in two stages: the first establishes bounds a and b on $s_{\min'}(a < s_{\min} \leqslant b)$, and the second interpolates its value in the interval $(a,\ b)$.

a. As a first guess for the bounds, take $a = 0$ and $b = \Delta x \langle \mathbf{p}(t_n)|\mathbf{p}(t_n)\rangle^{\frac{1}{2}}$. We already know the energy $\mathcal{V}(\boldsymbol{r}(t_n) + a\mathbf{p}(t_n)) = \mathcal{V}(\boldsymbol{r}(t_n))$ and the constrained (shaken) forces $\mathbf{f}(\boldsymbol{r}(t_n) + a\mathbf{p}(t_n)) = \mathbf{f}(\mathbf{r}(t_n))$, so $g(t_n; a)$ is easily obtained. The same quantities have to be examined at b, so we compute

$$\mathbf{r}(t_n; b) \quad \equiv \quad \mathbf{r}(t_n) + b\mathbf{p}(t_n) \tag{11.28}$$

and evaluate the energy $\mathcal{V}(\boldsymbol{r}(t_n) + b\mathbf{p}(t_n))$ and the forces

$$\mathbf{f}^{uc}(t_n; b) \quad \equiv \quad \mathbf{f}^{uc}(\mathbf{r}(t_n) + b\mathbf{p}(t_n)) \tag{11.29}$$

from the interaction function, from which the terms acting *only* along the constrained degrees of freedom are excluded. The components along the directions of the constraints are removed by the procedure Eq. 11.23-Eq. 11.25, that is, by performing

$$\mathbf{r}^{uc}(t_{n+1}; b) \quad \equiv \quad \mathbf{r}(t_n; b) + \Delta x^{uc}\mathbf{f}^{uc}(t_n; b) \tag{11.30}$$

where

$$\Delta x^{uc} \quad = \quad \frac{\mathbf{b}\,\langle \mathbf{p}(t_n)|\mathbf{p}(t_n)\rangle}{\langle \mathbf{f}^{uc}(t_n; b)|\mathbf{f}^{uc}(t_n; b)\rangle} \tag{11.31}$$

and

$$SHAKE(\mathbf{r}(t_n; b); \mathbf{r}^{uc}(t_{n+1}; b); \mathbf{r}(t_{n+1}; b)) \tag{11.32}$$

and

$$\mathbf{f}(t_n; b) \quad \equiv \quad \frac{\mathbf{r}(t_{n+1}; b) - \mathbf{r}(t_n; b)}{\Delta x^{uc}}. \tag{11.33}$$

With the shaken forces (Eq. 11.33) *g(t$_n$;b)* can be computed. The values a and b are accepted as bounds on s_{min} if

$$g(t_n; b) \quad < \quad 0 \tag{11.34}$$

or

$$\mathcal{V}(\boldsymbol{r}(t_n) + b\mathbf{p}(t_n)) \quad \geqslant \quad \mathcal{V}(\mathbf{r}(t_n) + a\mathbf{p}(t_n)) \tag{11.35}$$

Otherwise s_{min} lies beyond b, so b is too small (and so is Δx) and the process of looking for bounds on s_{min} is to be repeated with b doubled.

a. Use the cubic interpolation formulae Eqs. 11.9-11.13 in order to find s_{min} in the interval $(a,\ b)$ and determine the new unconstrained configuration

$$\mathbf{r}^{uc}(t_{n+1}) \quad = \quad \boldsymbol{r}(t_n) + s_{\min}\mathbf{p}(t_n) \tag{11.36}$$

which has to be shaken,

$$SHAKE(\boldsymbol{r}(t_n); \mathbf{r}^{uc}(t_{n+1}); \boldsymbol{r}(t_{n+1})) \tag{11.37}$$

2. Calculate the new energy $\mathcal{V}(\boldsymbol{r}(t_{n+1}))$ and the unconstrained forces $\mathbf{f}^{uc}(t_{n+1}) = \mathbf{f}^{uc}(\boldsymbol{r}(t_{n+1}))$ from the interaction function excluding the interaction terms that act *only* along the constraints. The components of the forces along the constraints are removed as above by performing

$$\mathbf{r}^{uc}(t_{n+2}) \quad = \quad \mathbf{r}(t_{n+1}) + \Delta x^{uc}\mathbf{f}^{uc}(t_{n+1}) \tag{11.38}$$

where

$$\Delta x^{uc} = \frac{s_{\min} \langle \mathbf{p}(t_n) | \mathbf{p}(t_n) \rangle^{1/2}}{< \mathbf{f}^{uc}(t_{n+1}) | \mathbf{f}^{uc}(t_{n+1}) >^{1/2}} \tag{11.39}$$

and

$$SHAKE(\mathbf{r}(t_{n+1}); \mathbf{r}^{uc}(t_{n+2}); \mathbf{r}(t_{n+2})) \tag{11.40}$$

and

$$\mathbf{f}(t_{n+1}) \equiv \frac{\mathbf{r}(t_{n+2}) - \mathbf{r}(t_{n+1})}{\Delta x^{uc}}. \tag{11.41}$$

3. The new search direction $\mathbf{p}(t_{n+1})$ could be directly obtained from Eq. 11.13. It is a linear combination of (n+2) shaken vectors. As the number of minimization steps n increases, the components of $\mathbf{p}(t_{n+1})$ along the constraint directions will grow, unless the search direction is also shaken at every minimization step. This can be achieved by solving $\mathbf{p}(t_n)$ from Eq. 11.36 with $\boldsymbol{r}(t_{n+1})$ instead of $\mathbf{r}^{uc}(t_{n+1})$,

   $$\mathbf{p}(t_n) \equiv \frac{\boldsymbol{r}(t_{n+1}) - \boldsymbol{r}(t_n)}{s_{\min}}. \tag{11.42}$$

   Then, the new search direction becomes

   $$\mathbf{p}(t_{n+1}) = \boldsymbol{f}(t_{n+1}) + \beta_n \mathbf{p}(t_n), \tag{11.43}$$

   with β_n given by Eq. 11.14).

Most of the parameters of the CGEM algorithm with SHAKE are identical to those discussed in Sec. 11.3 for the unconstrained CGEM algorithm. When applying constraints, conditions Eqs. 11.26 and 11.27 are not equivalent. This implies that it may occur that

$$Z^2 - g(t_n; a) g(t_n; b) < 0 \tag{11.44}$$

in (Eq. 11.9), in which case the minimization is terminated.

At every NTPR-th minimization step the same *quantities* as described in Chapters Secs. 11.2-11.3 for unconstrained SDEM and CGEM minimization are *printed*.

CHAPTER 12

Molecular Dynamics

12.1. Introduction

Molecular dynamics (MD) simulation with an empirical energy function such as Eq. 3.4 is a widely used tool to study the equilibrium and non-equilibrium structural, dynamical and thermodynamic properties of molecular systems. The equations of motion of classical mechanics for the atoms are integrated forward in time. When using Cartesian coordinates, these equations are Newton's equations of motion Eq. 2.8, with the forces $\boldsymbol{f}$ calculated as the negative gradient of the energy function $\mathcal{V}$, Eq. 2.10.

A simple but efficient algorithm for integration of Newton's equations of motion is the leap-frog scheme, which is obtained as follows. A Taylor expansion of the velocity $\mathbf{v}_i(t_n - \Delta t/2)$ at time point $t = t_n$ is subtracted from a Taylor expansion of $\mathbf{v}_i(t_n + \Delta t/2)$ at $t = t_n$.

Neglecting terms of third and higher order in the time step Δt and using Eq. 2.12 we obtain

$$\mathbf{v}_i(t_n + \Delta t/2) \quad = \quad \mathbf{v}_i(t_n - \Delta t/2) + m_i^{-1}\mathbf{f}_i(t_n)\Delta t, \tag{12.1}$$

the leap-frog velocity formula. Subtracting a Taylor expansion of the position $\mathbf{r}_i(t_n)$ at time point $t = t_n + \Delta t/2$ from a Taylor expansion of $\mathbf{r}_i(t_n + \Delta t)$ at $t = t_n + \Delta t/2$, neglecting terms of third and higher order in Δt and using Eq. 2.9 we obtain

$$\mathbf{r}_i(t_n + \Delta t) \quad = \quad \mathbf{r}_i(t_n) + \mathbf{v}_i(t_n + \Delta t/2)\Delta t, \tag{12.2}$$

the leap-frog position formula. Eq. 12.1 and Eq. 12.2 form the *leap-frog scheme* (Fig. 12.1).

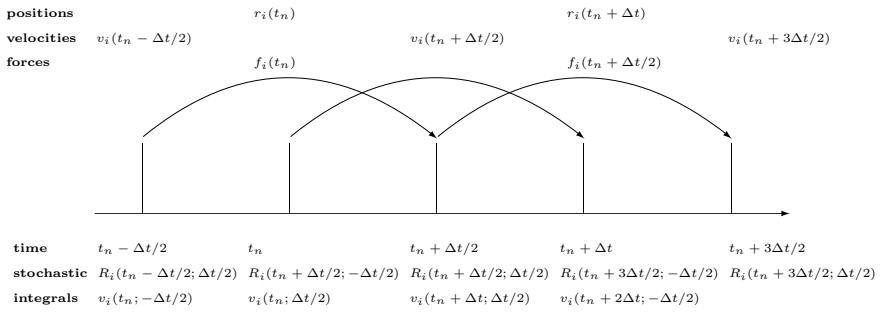


FIGURE 12.1. The leap-frog scheme.

In a standard MD simulation, the total energy E of the molecular system is a constant of motion. Since the number of atoms N and the volume of the computational (periodic) box V are standardly kept fixed, such an MD simulation is called an (N, V, E) simulation. In practice one would often prefer to keep the temperature T fixed instead of the energy, leading to an (N, V, T) simulation. In addition, one may want to keep the pressure P fixed instead of the volume, leading to an (N, P, T) simulation. Coupling of the molecular system to a temperature or pressure bath will imply a modification of the equations of motion and so of the simple leap-frog MD algorithm. In Sec. 12.2 *coupling to a temperature bath* is described and in Sec. 12.5 *coupling to a pressure bath*. The latter requires the *calculation of the virial*, which is discussed in Sec. 12.4. In Sec. 12.6 the leap-frog scheme including the modifications due to temperature scaling, pressure scaling and the presence of constraints, as it has been implemented in GROMOS, is presented. In Sec. 12.7 issues with respect to initialization, equilibration and sampling of an MD simulation are discussed.

12.2. Temperature scaling

MD simulations at constant temperature have several advantages: (*i*) the ensemble generated may be closer to the ensemble generated under experimental conditions; (*ii*) the sampling of conformations may be enhanced by performing simulations at high temperatures; (*iii*) energetic drifts caused by numerical inaccuracies or cutoff artifacts can be ameliorated; (*iv*) temperature dependent properties can be studied.

Thermostatting in MD++ relies on coupling a certain set of degrees of freedom $\{D\}_i$ to a certain temperature bath $\mathcal{B}_{Tj}(\{D\}_i)$ such that energy can "flow" between $\{D\}_i$ and $\mathcal{B}_{Tj}(\{D\}_i)$ and hence allow for the instantaneous temperature $\mathcal{T}(t;\{D\}_i)$ of the degrees of freedom set $\{D\}_i$ to be rigorously constant, or to fluctuate about the reference temperature $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$. (This flow of energy does, of course, not change the reference temperature $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$). The coupling is implemented by scaling the velocities of all atoms contained in $\{D\}_i$ with a (time-dependent) *scaling factor* $\lambda[t;\mathcal{B}_{Tj}(\{D\}_i)]$.

$\{D\}_i$ may contain translational, internal+rotational, or both types of degrees of freedom. The calculation of the corresponding instantaneous temperature $\mathcal{T}(t;\{D\}_i)$ is explained in Sec. 12.2.1. The variables characterizing $\mathcal{B}_{Tj}$, as well as the precise form of coupling between $\mathcal{B}_{Tj}$ and $\{D\}_i$ are dependent on the thermostat algorithm employed. These algorithms will be detailed in Sec. 12.2.2. The concept of temperature groups, sets of degrees of freedom and temperature baths will be discussed in more detail in Sec. 12.2.3. The precise calculation of the number of degrees of freedom is explained in Sec. 12.3.

12.2.1. Temperature calculation in MD++. Thermostatting involves the comparison of the instantaneous temperature $\mathcal{T}(t-\frac{\Delta t}{2};\{D\}_i)$ with the reference temperature $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$, where

$$\mathcal{T}\left(t-\frac{\Delta t}{2};\{D\}_i\right)=\frac{2\mathcal{K}(t-\frac{\Delta t}{2};\{D\}_i)}{N_{df}(\{D\}_i)k_B} \quad , \tag{12.3}$$

with $\mathcal{K}(t-\frac{\Delta t}{2};\{D\}_i)$ and $N_{df}(\{D\}_i)$ being the kinetic energy and number of degrees of freedom, respectively associated with $\{D\}_i$ and k_B being Boltzmann's constant. In the following, we use $t-\frac{\Delta t}{2}=t'$.
$\mathcal{K}(t';\{D\}_i)$ is computed as the sum of the kinetic energies pertaining to the individual temperature groups $\mathcal{G}_{Tj}$ contained in $\{D\}_i$,

$$\mathcal{K}(t';\{D\}_i)=\sum_{\mathcal{G}_{Tj}\in\{D\}_i}\mathcal{K}(t';\mathcal{G}_{Tj}) \quad . \tag{12.4}$$

The calculation of $\mathcal{K}(t';\mathcal{G}_{Tj})$, in turn, considers the type of degrees of freedom contributed by the different temperature groups:
Translational velocities $\mathbf{v}_{tr}(t';\mathcal{G}_{Tj})$ of a temperature group $\mathcal{G}_{Tj}$ are computed as

$$\mathbf{v}_{tr}(t';\mathcal{G}_{Tj})=M^{-1}(\mathcal{G}_{Tj})\sum_{k\in\mathcal{G}_{Tj}}m_k\mathbf{v}_k(t') \quad , \tag{12.5}$$

where $M(\mathcal{G}_{Tj})=\sum_{k\in\mathcal{G}_{Tj}}m_k$ is the total mass of the temperature group $\mathcal{G}_{Tj}$ and $\mathbf{v}_k(t')$ and m_k are the velocities and masses of atom k (belonging temperature group $\mathcal{G}_{Tj}$). The corresponding *translational kinetic energy* is

$$\mathcal{K}_{tr}(t';\mathcal{G}_{Tj})=\frac{1}{2}\sum_{k\in\mathcal{G}_{Tj}}\left\{m_k\mathbf{v}_k^2(t')-m_k\left[\mathbf{v}_k(t')-\mathbf{v}_{tr}(t';\mathcal{G}_{Tj})\right]^2\right\} \quad . \tag{12.6}$$

The corresponding internal+rotational kinetic energy is

$$\mathcal{K}_{ir}(t';\mathcal{G}_{Tj})=\frac{1}{2}\sum_{k\in\mathcal{G}_{Tj}}m_k\left[\mathbf{v}_k(t')-\mathbf{v}_{tr}(t';\mathcal{G}_{Tj})\right]^2 \quad . \tag{12.7}$$

Eq. 12.6 and Eq. 12.7 can be easily understood as the latter arising from atomic velocities measured with respect to the centre of mass velocity of the temperature group the atoms belong to, whereas the former merely constitutes the rest (difference of total kinetic energy and the internal+rotational contribution).

For the calculation of the instantaneous temperature according to Eq. 12.3, a set of degrees of freedom involving only translational degrees of freedom will contribute translational kinetic energies only, whereas a set of degrees of freedom involving only internal+rotational degrees of freedom will contribute internal+rotational kinetic energies only. A set of degrees of freedom involving both translational and internal+rotational degrees of freedom will contribute a sum of translational and internal+rotational kinetic

energies.
Finally, it should be emphasized that the velocities $\mathbf{v}_k(t')$ used in the temperature calculation are *constrained velocities*, that is, rather than being free-flight velocities deriving from the physical force field, they are corrected for the application of various constraints.

12.2.2. Thermostat algorithms in MD++. Algorithms that modify Newton's equation of motion in order to generate a constant-temperature (rather than a constant-energy) ensemble are called *thermostat algorithms*. Several methods for performing MD at constant temperature have been proposed in the literature.[7,111] Among non-stochastic thermostatting methods, one can distinguish between temperature *constraining* and temperature *relaxation* approaches. The latter can *e.g.* be implemented in a *weak coupling* or an *extended-system* scheme.
MD++ offers a weak coupling and an extended-sytem [Nosé-Hoover (chain)] thermostat. Here, we present the implementation of these algorithms in the context of a simple velocity-scaling scheme acting on *free-flight* velocities. Denoting the unscaled free-flight velocity of atom k pertaining to the degrees of freedom set $\{D\}_i$ with ${\mathbf{v}^{un}}_k\left(t+\frac{\Delta t}{2}\right)$, the velocity scaling obeys the following equations:

- in the case of only the translational degrees of freedom of atom k (belonging to temperature group $\mathcal{G}_{Tj}$) being coupled to a temperature bath $\mathcal{B}_{Tn}(\{D\}_i)$,

$$\begin{aligned}\mathbf{v}_k\left(t+\frac{\Delta t}{2}\right) &= \lambda[t;\mathcal{B}_{Tn}(\{D\}_i)]\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right) \\ &+ \left[{\mathbf{v}^{un}}_k\left(t+\frac{\Delta t}{2}\right)-\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right)\right] \quad ;\end{aligned} \tag{12.8}$$

- in the case of only the internal+rotational degrees of freedom of atom k (belonging to temperature group $\mathcal{G}_{Tj}$) being coupled to a temperature bath $\mathcal{B}_{Tm}(\{D\}_i)$,

$$\begin{aligned}\mathbf{v}_k\left(t+\frac{\Delta t}{2}\right) &= \mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right) \\ &+ \lambda[t;\mathcal{B}_{Tm}(\{D\}_i)]\left[{\mathbf{v}^{un}}_k\left(t+\frac{\Delta t}{2}\right)-\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right)\right] \quad ;\end{aligned} \tag{12.9}$$

- in the case of the translational degrees of freedom of atom k (belonging to temperature group $\mathcal{G}_{Tj}$) being coupled to a temperature bath $\mathcal{B}_{Tn}(\{D\}_i)$ and the internal+rotational degrees of freedom of atom k (belonging to temperature group $\mathcal{G}_{Tj}$) being coupled to a temperature bath $\mathcal{B}_{Tm}(\{D\}_i)$,

$$\begin{aligned}\mathbf{v}_k\left(t+\frac{\Delta t}{2}\right) &= \lambda[t;\mathcal{B}_{Tn}(\{D\}_i)]\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right) \\ &+ \lambda[t;\mathcal{B}_{Tm}(\{D\}_i)]\left[{\mathbf{v}^{un}}_k\left(t+\frac{\Delta t}{2}\right)-\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right)\right] \quad ,\end{aligned} \tag{12.10}$$

where, in analogy to Eq. 12.5, the *unscaled* translational velocity of temperature group $\mathcal{G}_{Tj}$ is computed as

$$\mathbf{v}^{un}_{tr}\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right)=M^{-1}(\mathcal{G}_{Tj})\sum_{k\in\mathcal{G}_{Tj}}m_k{\mathbf{v}^{un}}_k\left(t+\frac{\Delta t}{2};\mathcal{G}_{Tj}\right) \quad . \tag{12.11}$$

For details about the isothermal equations of motions resulting from the application of the different thermostat algorithms, the corresponding phase space metrics and conserved properties, the reader is referred to the original papers introducing these methods.

Woodcock thermostat

A temperature constraint algorithm does not allow any temperature fluctuations, *i.e.*, it enforces, at each time step, the instantaneous temperature $\mathcal{T}(t';\{D\}_i)$ to be equal to the corresponding reference temperature $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$. This can be done with *e.g.* the Woodcock thermostat[112], where the scaling factor at time t is computed as

$$\lambda[t;\mathcal{B}_{Tj}(\{D\}_i)]=\left[\frac{N_{df}(\{D\}_i)-1}{N_{df}(\{D\}_i)}\frac{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]}{\mathcal{T}(t';\{D\}_i)}\right]^{\frac{1}{2}} \quad . \tag{12.12}$$

The Woodcock thermostat generates a canonical distribution of coordinates, but (due to the absence of kinetic energy fluctuations) not momenta, at $\mathcal{T}_o$.

Berendsen thermostat

A weak-coupling algorithm[11] achieves a first-order relaxation of the instantaneous temperature $\mathcal{T}(t';\{D\}_i)$ to the corresponding reference temperature $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$) according to

$$\frac{d\mathcal{T}(t';\{D\}_i)}{dt} = \tau^{-1}[\mathcal{B}_{Tj}(\{D\}_i)]\left\{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)] - \mathcal{T}(t';\{D\}_i)\right\} \quad . \tag{12.13}$$

The scaling factor describing this relaxation in terms of a velocity-scaling according to Eq. 12.8-Eq. 12.10 is found by expressing the change of the kinetic energy in terms of the heat capacity (per degree of freedom) at constant volume c_v^{df} as

$$\Delta\mathcal{K}(t';\{D\}_i) = N_{df}(\{D\}_i)c_v^{df}k_B\Delta T(t') \quad , \tag{12.14}$$

so that, using[11] $c_v^{df} = \frac{1}{2}k_B$, the so-called Berendsen thermostat is recovered, with

$$\begin{aligned}\lambda[t;\mathcal{B}_{Tj}(\{D\}_i)] &= \left\{1 + 2c_v^{df}k_B^{-1}\tau^{-1}[\mathcal{B}_{Tj}(\{D\}_i)]\Delta t\left[\frac{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]}{\mathcal{T}(t';\{D\}_i)} - 1\right]\right\}^{\frac{1}{2}} \\ &= \left\{1 + \tau^{-1}[\mathcal{B}_{Tj}(\{D\}_i)]\Delta t\left[\frac{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]}{\mathcal{T}(t';\{D\}_i)} - 1\right]\right\}^{\frac{1}{2}} \quad .\end{aligned} \tag{12.15}$$

The temperature bath-specific parameter $\tau[\mathcal{B}_{Tj}(\{D\}_i)]$ is a purely empirical quantity adjusting the strength of the coupling of the degrees of freedom set $\{D\}_i$ to the heat bath $\mathcal{B}_{Tj}(\{D\}_i)$. In the limit $\tau[\mathcal{B}_{Tj}(\{D\}_i)] = \Delta t$, the Berendsen algorithm is equivalent to the Woodcock temperature constraining algorithm, except for the reduction of degrees of freedom by one. In the limit $\tau[\mathcal{B}_{Tj}(\{D\}_i)] \to \infty$, thermostatting is disabled, and Newton's equations of motion are recovered. Note that the isothermal equations of motion given by the Berendsen thermostat will not sample canonical coordinates *and* momenta. Only if the Berendsen thermostat reduces to the Woodcock one ($\tau[\mathcal{B}_{Tj}(\{D\}_i)] = \Delta t$) will the generated configurations be canonical. Neither the Woodcock nor the Berendsen thermostat render possible canonical sampling of the momenta. This deficiency has to be considered if fluctuation-dependent properties[113] are to be computed in the analysis of simulation data.

Nosé-Hoover thermostat

The modification of Newton's equations of motion to generate an isothermal trajectory can also be done in an extended-system approach. This thermostatting method was originally developed by Nosé,[114] and subsequently cast into the more practical framework nowadays-known as the Nosé-Hoover thermostat.[115,116] The corresponding equations of motion sample a microcanonical ensemble in the extended system, and a canonical emsemble in the real system. The extension of the system is brought about by introducing an additional artificial dynamical variable $\tilde{s}_1(t)$ [with mass $Q_1 > 0$ and velocity $\dot{\tilde{s}}_1(t)$]. The velocity $\dot{\tilde{s}}_1(t)$ of the extended-system variable is equal to the function $\gamma_1(t)$ controlling the heat exchange between the thermostatted (real) system and the heat bath, such that, for atom k,

$$\dot{\mathbf{p}}_k(t) = \mathbf{f}_k(t) - \gamma_1(t)\mathbf{p}_k(t) \quad , \tag{12.16}$$

with $\mathbf{f}_k(t)$ denoting the force acting on atom k at time t and $\mathbf{p}_k(t)$ denoting the momentum corresponding to the thermostatted velocity of atom k. The time-evolution of $\tilde{s}_1(t)$ is described by a second-order equation, which implies that heat fluxes between the system and the bath, and hence the temperature evolution, are oscillatory. These oscillations can be tuned by the mass Q_1 of $\tilde{s}_1(t)$. Considering the first-order evolution of $\gamma_1(t)$ [rather than the second-order evolution of $\tilde{s}_1(t)$] for a certain temperature bath $\mathcal{B}_{Tj}$,

$$\begin{aligned}\dot{\gamma}_1[t;\mathcal{B}_{Tj}(\{D\}_i)] &= -k_B N_{df}(\{D\}_i)Q_1^{-1}[\mathcal{B}_{Tj}(\{D\}_i)]\cdot \\ &\quad \left\{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)] - \mathcal{T}(t';\{D\}_i)\right\} \quad ,\end{aligned} \tag{12.17}$$

and using $Q_1[\mathcal{B}_{Tj}(\{D\}_i)] = \tau_1^2[\mathcal{B}_{Tj}(\{D\}_i)]N_{df}(\{D\}_i)k_B\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$, one finds

$$\dot{\gamma}_1[t;\mathcal{B}_{Tj}(\{D\}_i)] = -\tau_1^{-2}[\mathcal{B}_{Tj}(\{D\}_i)]\left\{1 - \frac{\mathcal{T}(t';\{D\}_i)}{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]}\right\} \quad , \tag{12.18}$$

thereby introducing an effective relaxation time

$$\tau_1[\mathcal{B}_{Tj}(\{D\}_i)] = \left\{\frac{Q_1[\mathcal{B}_{Tj}(\{D\}_i)]}{N_{df}(\{D\}_i)k_B\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]}\right\}^{\frac{1}{2}} \tag{12.19}$$

instead of the less intuitive effective mass $Q_1[\mathcal{B}_{Tj}(\{D\}_i)]$ to characterize the strength of the coupling to the heat bath.

MD++ implements the Nosé-Hoover thermostat by using

$$\lambda[t;\mathcal{B}_{Tj}(\{D\}_i)] = 1 - \Delta t \gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)] \quad , \tag{12.20}$$

where $\gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)]$ is determined by discrete integration,

$$\begin{aligned} \gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)] &= \gamma_1[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] \\ &+ \Delta t \tau_1^{-2}[\mathcal{B}_{Tj}(\{D\}_i)] \left\{ \frac{\mathcal{T}(t';\{D\}_i)}{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} - 1 \right\} \quad , \end{aligned} \tag{12.21}$$

which follows from Eq. 12.16 propagating the momentum variables at (due to use of the leap-frog scheme) half-integer time steps as

$$\mathbf{p}_k(t'+\Delta t) = \mathbf{p}_k(t') + \Delta t \left[\mathbf{f}_k(t) - \gamma_1(t)\mathbf{p}_k(t')\right] = m_k \mathbf{v}^{un}{}_k \left[1 - \gamma_1(t)\Delta t\right] \quad , \tag{12.22}$$

where, again, $\mathbf{v}^{un}{}_k$ denotes the unscaled free-flight velocity of atom k.
The temperature bath-specific parameter $\tau[\mathcal{B}_{Tj}(\{D\}_i)]$ is a purely empirical quantity adjusting the strength of the coupling of the degrees of freedom set $\{D\}_i$ to the heat bath $\mathcal{B}_{Tj}(\{D\}_i)$. Too large values of $\tau[\mathcal{B}_{Tj}(\{D\}_i)]$ (loose coupling; high mass of the extended-system variable) may cause a poor temperature control (the limiting case $\tau[\mathcal{B}_{Tj}(\{D\}_i)] \to \infty$ generating the microcanonical ensemble), whereas too small values (tight coupling; low mass of the extended-system variable) may cause high-frequency temperature oscillations.

Nosé-Hoover chain thermostat

The Nosé-Hoover chain thermostat[117] aims at relaxing the instantaneous temperature $\mathcal{T}(t';\{D\}_i)$ to the reference value $\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]$ based on a chain of successive thermostat variables. In this case, the single thermostat variable $\gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)]$ of the Nosé-Hoover scheme is enhanced by a chain of variables $\gamma_c[t;\mathcal{B}_{Tj}(\{D\}_i)]$, $c = 2, ..., N_c$, applying a thermostat to each other in sequence. This algorithm has been introduced[117] to alleviate the two main drawbacks of the Nosé-Hoover algorithm: (*i*) the presence of spurious temperature oscillations; (*ii*) the non-ergodicity of the sampling for small or stiff systems, or systems at low temperatures.
MD++ implements the Nosé-Hoover chain thermostat in analogy to Eq. 12.20 and Eq. 12.21 as

$$\lambda[t;\mathcal{B}_{Tj}(\{D\}_i)] = 1 - \Delta t \gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)] \quad , \tag{12.23}$$

where $\gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)]$ is determined by discrete integration,

$$\begin{aligned} \gamma_1[t;\mathcal{B}_{Tj}(\{D\}_i)] &= \gamma_1[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] + \Delta t \tau_1^{-2}[\mathcal{B}_{Tj}(\{D\}_i)] \left\{ \frac{\mathcal{T}(t';\{D\}_i)}{\mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} - 1 \right\} \\ &- \Delta t \gamma_1[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)]\gamma_2[t;\mathcal{B}_{Tj}(\{D\}_i)] \quad , \end{aligned} \tag{12.24}$$

each successive chain thermostat $\gamma_c[t;\mathcal{B}_{Tj}(\{D\}_i)]$, $c = 2, ..., N_c - 1$ is propagated according to

$$\begin{aligned} \gamma_c[t;\mathcal{B}_{Tj}(\{D\}_i)] &= \gamma_c[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] \\ &+ \Delta t \left\{ \tau_c^{-2}[\mathcal{B}_{Tj}(\{D\}_i)] \left[\tau_{c-1}^2[\mathcal{B}_{Tj}(\{D\}_i)]\gamma_{c-1}^2[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] \right.\right. \\ &\left.\left. - {N_{df}}^{-1}(\{D\}_i) \right] - \gamma_c[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)]\gamma_{c+1}[t;\mathcal{B}_{Tj}(\{D\}_i)] \right\} \quad , \end{aligned} \tag{12.25}$$

the last one being

$$\begin{aligned} \gamma_{N_c}[t;\mathcal{B}_{Tj}(\{D\}_i)] &= \gamma_{N_c}[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] \\ &+ \Delta t \tau_{N_c}^{-2}[\mathcal{B}_{Tj}(\{D\}_i)] \left[\tau_{N_c-1}^2[\mathcal{B}_{Tj}(\{D\}_i)]\gamma_{N_c-1}^2[t-\Delta t;\mathcal{B}_{Tj}(\{D\}_i)] \right. \\ &\left. - {N_{df}}^{-1}(\{D\}_i) \right] \quad . \end{aligned} \tag{12.26}$$

This follows from the chain analog of Eq. 12.22

$$\mathbf{p}_k(t'+\Delta t) = \mathbf{p}_k(t') + \Delta t \left[\mathbf{f}_k(t) - \gamma_1(t)\mathbf{p}_k(t')\right] = m_k \mathbf{v}^{un}{}_k \left[1 - \gamma_1(t)\Delta t\right] \quad , \tag{12.27}$$

with the corresponding equations for the chain thermostat variables,

$$\begin{aligned} \mathbf{p}_{\gamma_1}(t'+\Delta t) &= \mathbf{p}_{\gamma_1}(t') \\ &+ \Delta t \left\{ 2\Delta\mathcal{K}(t';\{D\}_i) - N_{df}(\{D\}_i) k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)] \right\} \end{aligned} \tag{12.28}$$

$$- \quad \mathbf{p}_{\gamma_1}(t') \frac{\mathbf{p}_{\gamma_2}(t'+\Delta t)}{\tau_2^2[\mathcal{B}_{Tj}(\{D\}_i)] N_{df}(\{D\}_i) k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} \quad ,$$

$$\begin{aligned} \mathbf{p}_{\gamma_c}(t'+\Delta t) &= \mathbf{p}_{\gamma_c}(t') \\ &+ \Delta t \left\{ \frac{\mathbf{p}^2_{\gamma_{c-1}}(t')}{\tau_{c-1}^2[\mathcal{B}_{Tj}(\{D\}_i)] N_{df}(\{D\}_i) k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} - k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)] \right\} \\ &- \mathbf{p}_{\gamma_c}(t') \frac{\mathbf{p}_{\gamma_{c+1}}(t'+\Delta t)}{\tau_{c+1}^2[\mathcal{B}_{Tj}(\{D\}_i)] N_{df}(\{D\}_i) k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} \quad , \end{aligned} \tag{12.29}$$

and

$$\begin{aligned} \mathbf{p}_{\gamma_{N_c}}(t'+\Delta t) &= \mathbf{p}_{\gamma_{N_c}}(t') \\ &+ \Delta t \left\{ \frac{\mathbf{p}^2_{\gamma_{N_c-1}}(t')}{\tau_{N_c-1}^2[\mathcal{B}_{Tj}(\{D\}_i)] N_{df}(\{D\}_i) k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)]} - k_B \mathcal{T}_o[\mathcal{B}_{Tj}(\{D\}_i)] \right\} \quad , \end{aligned} \tag{12.30}$$

where N_c chain thermostats are used.

12.2.3. Use of temperature groups, sets of degrees of freedom and thermostats. If different parts of the molecular system are subject to different heating rates due to, for example, the use of a non-bonded interaction cut-off radius, it may be necessary to *separately couple the different parts of the system to different temperature baths*.
In the following we will explain the characterisation and definition of temperature groups, degrees of freedom sets and temperature baths in MD++. Finally, we will summarize the flexibility of temperature coupling in MD++ and provide an example of a typical thermostat setting in a biomolecular simulation.
In the following, we will use the abbreviation tr for translational, i+r for internal+rotational degrees of freedom and dof for degrees of freedom.

- **Temperature groups in MD++**
  Temperature groups are molecular entities serving to dissect tr and i+r properties, *e.g.* in the computation of degrees of freedom, velocities or kinetic energies, or in the application of translational and/or rotational constraints. Usually, one molecule forms one temperature group, *e.g.* one protein, or one water molecule. But also just one particle, *e.g.* an ion, can be one temperature group.

  Temperature groups are defined in the molecular topology file (TEMPERATUREGROUPS block) for those molecular entities contained in the solute molecular topology (SOLUTEATOM block). If the system contains solvent molecules from the solvent molecular topology (SOLVENTATOM block), each solvent molecule inevitably (*i.e.* not changeable by the user) constitutes one separate temperature group.
- **Sets of degrees of freedom in MD++**
  A dof set is a collection of atoms (MD++) that are affected by the same thermostat settings.
  A dof set I is defined and characterized in the MULTIBATH block *via* the joint specification of: (*i*) the last atom in this dof set [LAST(I)]; (*ii*) the temperature bath the tr dof of this dof set are coupled to [COM-BATH(I)]; (*iii*) the temperature bath the i+r dof of this dof set are coupled to [IR-BATH(I)].

- **Temperature baths in MD++**
  A temperature bath is a heat reservoir of constant reference temperature thermostatting one or several dof sets.

  A temperature bath I is defined and characterized in the MULTIBATH block *via* the joint specification of: (*i*) the reference temperature of this bath [TEMP0(I)]; (*ii*) the coupling time of this bath [TAU(I)]. Note that the specification of a certain thermostat algorithm, as well as of the number of chains in the Nosé-Hoover chain thermostat, thus inevitably apply to all temperature baths. In adddition, in the case of thermostatting with the Nosé-Hoover chain algorithm, each of the chain thermostat variables will have the same coupling time.

Finally, we consider the example of two peptide copies in a methanol-water mixture. More specifically, we consider a topology whose solute block consists of two peptides (2x50 atoms) and 500 methanol molecules (1500 atoms). Furthermore, we assume the system to contain 4000 water molecules, *i.e.* the coordinate file will contain 100 peptide and 1500 methanol atoms as well as 12000 water atoms. We define the following

4502 temperature groups: Peptides 1 and 2 (temperature group defined in the molecular topology file), each of methanol molecules 1-500 (temperature groups defined in the molecular topology file), and each of the water molecules 1-4000 (temperature group definitions being an immediate consequence of the solvent nature).

In a typical MD simulation, one would here employ three different temperature baths. In a physically-realistic equilibrium situation, these three baths would have equal temperatures. One would couple the tr dof of the two peptides to bath 1, the i+r dof the two peptides to bath 2, and the tr and rotational dof of all (rigid) methanol and water molecules to bath 3. We assume thermostatting at 298.15 K by the weak coupling method with a coupling time of 0.1 ps.
The MD++ MULTIBATH block reads:

```
MULTIBATH
# general thermostatting settings
# ALGORITHM
0
# NBATHS
3
# temperature bath definition and characterisation
# TEMP0(1..NBATHS)
298.15 298.15 298.15
# TAU(1..NBATHS)
0.1 0.1 0.1
# DOFSET
4
# dof set definition and characterisation
# LAST(1..DOFSET) COM-BATH(1..DOFSET) IR-BATH(1..DOFSET)
# peptides
100 1 2
# 500 methanol molecules
1600 3 3
# solvent molecules
13600 3 3
END
```

Finally, an important practical issue arises in thermostatted MD simulations: When Newton's equations of motion Eq. 2.8 are integrated, the total energy and the total translational momentum of the system are conserved. This need no longer be true when a coupling to a temperature bath is applied. The centre of mass may slowly pick up motion during an MD simulation, even when initially at rest. This motion may be regularly removed, which is described in Sec. 4.4.

12.3. Number of degrees of freedom

It is important to know exactly how many and which degrees of freedom are being simulated. For a system with $\mathcal{N}_a$ particles moving in three dimensions, the total number of degrees of freedom $\mathcal{N}_d$ is essentially $3\mathcal{N}_a$. However, the application of various boundary conditions in the form of geometric constraints will reduce this number. Each temperature group (TG) $\mathcal{G}_{Ti}$ is assigned a certain number of translational $[N^t_{df}(\mathcal{G}_{Ti})]$ and internal+rotational $[N^{ir}_{df}(\mathcal{G}_{Ti})]$ degrees of freedom (DOF). In the absence of any constraints, the values of these variables for a given TG $\mathcal{G}_{Ti}$ consisting of $N_{\mathcal{G}_{Ti}}$ atoms are $N^t_{df}(\mathcal{G}_{Ti}) = 3$, and $N^{ir}_{df}(\mathcal{G}_{Ti}) = 3(N_{\mathcal{G}_{Ti}} - 1)$, respectively. However, simulations are usually performed in ensembles involving a variety of different constraints, *e.g.* position, distance, dihedral angle, rotational and/or translational constraints, or constraints of the linear and/or angular momentum or of the temperature of the system. Any application of constraints (not redundant to already existing constraints) to a given TG I will reduce the number of degree of freedom of this TG. The following section explains how the number of degree of freedom is reduced.

If TG $\mathcal{G}_{Ti}$ contains ${N_{\mathcal{G}_T}^{poscon}}_i > 0$ posititionally-constrained atoms, $N_{df}^t(\mathcal{G}_{Ti})$ is decreased by three and $N_{df}^{ir}(\mathcal{G}_{Ti})$ by $3({N_{\mathcal{G}_T}^{poscon}}_i - 1)$. Remember that a temperature group consists of covalently bound atoms, meaning that a single position restraint on an atom in the group removes the translational degrees of freedom of the group.

If TG $\mathcal{G}_{Ti}$ contains ${N_{\mathcal{G}_T}^{discon}}_i > 0$ distance-constraints involving at least one non-positionally-constrained atom, $N_{df}^{ir}(\mathcal{G}_{Ti})$ is decreased by ${N_{\mathcal{G}_T}^{discon}}_i$. Any dof reduction arising from distance constraints between two different TG is distributed in equal amounts (one half) between the two TG.

If TG $\mathcal{G}_{Ti}$ contains ${N_{\mathcal{G}_T}^{dihcon}}_i > 0$ dihedral-angle constraints involving at least one non-positionally-constrained atom, $N_{df}^{ir}(\mathcal{G}_{Ti})$ is decreased by ${N_{\mathcal{G}_T}^{dihcon}}_i$. Any dof reduction arising from dihedral-angle constraints between two different TG $\mathcal{G}_{Ti}$, $\mathcal{G}_{Tj}$ is distributed in a ratio of $\frac{N_{dih,a}(\mathcal{G}_{Ti})}{N_{dih,a}(\mathcal{G}_{Tj})}$ between the two TG, where $1 \leq N_{dih,a}(\mathcal{G}_{Ti}) \leq 3$, $1 \leq N_{dih,a}(\mathcal{G}_{Tj}) \leq 3$ (with $N_{dih,a}(\mathcal{G}_{Ti}) + N_{dih,a}(\mathcal{G}_{Tj}) = 4$) are the numbers of dihedral-angle atoms contained in TG $\mathcal{G}_{Ti}$ and $\mathcal{G}_{Tj}$, respectively.

If TG $\mathcal{G}_{Ti}$ is translationally-constrained, $N_{df}^t(\mathcal{G}_{Ti})$ will be decreased by three. If TG $\mathcal{G}_{Ti}$ is rotationally-constrained, $N_{df}^{ir}(\mathcal{G}_{Ti})$ will be decreased by three.

When the system linear and/or angular momentum is constrained, three dof must be removed for each constrained type of momentum. After application of the previous rules, there will be M TG $\mathcal{G}_{Ti}$ for which $N_{df}^t(\mathcal{G}_{Ti}) > 0$. For these groups, $N_{df}^t(\mathcal{G}_{Ti})$ is decreased by $3/M$ if the system linear momentum is constrained and by $3/M$ if the system angular momentum is constrained. When instead the centre-of-mass motion is removed every NSCM steps, NDFMIN degrees of freedom are removed in the same way at those time points.

When temperature constraining is applied, the number of dof of TG $\mathcal{G}_{Ti}$ coupled to a temperature bath involved in temperature constraining must be decreased by one (to ensure canonical sampling in the configurational space). In this case, and after application of the previous rules and for a given translational set of DOF $\{D^t\}_j$, there will be $M(\{D^t\}_j)$ TG $\mathcal{G}_{Ti} \in \{D^t\}_j$ for which $N_{df}^t(\mathcal{G}_{Ti}) > 0$. For these groups, $N_{df}^t(\mathcal{G}_{Ti}) > 0$ is decreased by $1/M(\{D^t\}_j)$. Similarly, for a given internal+rotational set of DOF $\{D^{ir}\}_j$, there will be $N(\{D^{ir}\}_j)$ TG $\mathcal{G}_{Ti} \in \{D^{ir}\}_j$ for which $N_{df}^{ir}(\mathcal{G}_{Ti}) > 0$. For these groups, $N_{df}^{ir}(\mathcal{G}_{Ti})$ is decreased by $1/N(\{D^{ir}\}_j)$.

12.4. Calculation of the virial

In the GROMOS implementation, the way the virial is calculated can be chosen. In the PRESSURESCALE block the switch VIRIAL can be set to none(0) (no virial calculated), atomic(1) (atomic virial) or group(2) (group-based virial), and these settings equally apply to solute and solvent.

The group based instantaneous pressure tensor $\underline{\mathcal{P}}$ is related to the group-based virial and group-based internal kinetic energy tensor of the system. The word *group-based* refers to a pressure definition excluding virial and kinetic-energy contributions within user-specified groups of (covalently-linked) atoms[118,119]. These groups will be referred to as pressure groups (see Vol. 3). Single atoms can be used as pressure groups, in which case an atom-based pressure definition is recovered. In the same way molecules can be defined as pressure groups to recover a molecular-based pressure definition. The average pressure is not affected by the specific choice of groups, but the pressure fluctuations are. In practice, atom grouping is used to reduce these fluctuations. The pressure is only calculated for systems under periodic boundary conditions. Note also that the contribution of special (nonphysical) forces (e.g., atom-position or atom-distance restraining) to the pressure is generally not included, except for atom-distance restraining. The instantaneous atom-based pressure tensor is computed as

$$\underline{\mathcal{P}}^* = \frac{2}{\mathcal{V}}(\underline{\mathcal{K}}^* - \underline{\mathcal{W}}^*) \tag{12.31}$$

where

$$\underline{\mathcal{K}}^* = \tfrac{1}{2}\sum_{i=1}^{N} m_i \dot{\mathbf{r}}_i \otimes \dot{\mathbf{r}}_i \tag{12.32}$$

and

$$\underline{\mathcal{W}}^*_{\mu\nu} = \tfrac{1}{2}\sum_{\lambda}\frac{\partial \mathcal{U}}{\partial \mathcal{B}_{\mu\lambda}}\mathcal{B}_{\nu\lambda} \tag{12.33}$$

are the instantaneous atom-based internal kinetic energy and virial tensors, and $\mathcal{V}$ and $\mathcal{U}$ being the instantaneous volume and total potential energy of the system, $\underline{\mathcal{B}}$ the box matrix introduced in Sec. 3.5 and $\dot{\mathbf{r}}_i$ the internal velocities. The corresponding isotropic (scalar) quantities are related to the tensor quantities through

$$\mathcal{K}^* = Tr[\underline{\mathcal{K}}^*], \mathcal{W}^* = Tr[\underline{\mathcal{W}}^*], \mathcal{P}^* = (1/3)Tr[\underline{\mathcal{P}}^*], \tag{12.34}$$

where Tr returns the trace of a matrix, $\mathcal{K}^*$ is equivalent to Eq. 2.6, and $\mathcal{W}$ is defined as

$$\mathcal{W}^* = \frac{3\mathcal{V}}{2}\frac{\partial \mathcal{U}}{\partial \mathcal{V}}. \tag{12.35}$$

One can show that:[102,120] (1) the contribution to the atom-based virial tensor of a potential energy term that solely depends on the scalar products or determinants defined by a set of interatomic vectors is symmetric; (2) the contribution to the atom-based virial tensor of a potential energy term that solely depends on the angles defined by a set of vectors is (in addition) traceless. The first observation implies that all covalent (bond stretching or constraint, bond-angle bending, proper and improper dihedral angle), and pairwise nonbonded force field terms lead to a symmetric contribution to the atom-based virial. The second observation implies that covalent bond-angle bending as well as proper and improper dihedral angle (but not bond stretching or constraint and pairwise nonbonded) terms lead to a traceless contribution to the atom-based virial (i.e., no contribution to the scalar atom-based pressure). However, these results are generally not valid for the corresponding group-based tensor (see below).

In the special case of a pairwise-additive interaction term $\mathcal{U}_p$ depending on minimum-image interatomic distances and without explicit dependence on the box dimensions (bond stretching or constraint and pairwise nonbonded terms; but not reciprocal-space lattice-sum interactions[118,119] see Sec. 7.4), Eq. 12.33 leads to a virial contribution

$$\mathcal{W}^*_p = -\tfrac{1}{2}\sum_{i=1}^{N}\sum_{j>i}^{N}\mathbf{f}_{p,ij}\otimes\overline{\mathbf{r}}_{ij}, \tag{12.36}$$

where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ is the vector connecting j to i, $\overline{\mathbf{r}}_{ij}$ the corresponding minimum-image vector, and $\mathbf{f}_{p,ij}$ the pairwise force exerted by atom j on atom i. This equation is easily generalized to interaction terms involving more than two atoms (bond-angle bending, proper and improper dihedral-angle terms). The atom based virial contribution of all covalent (including bond constraints) and nonbonded (excluding reciprocal-space lattice-sum interactions) terms is calculated using Eq. 12.36 or one of its generalizations. Note that MD++ internally stores the virial without the prefactor $-\frac{1}{2}$.

The GROMOS implementation includes the possibility of using a group-based pressure definition (corresponding to any arbitrary partitioning of atoms into virial groups), instead of the atom-based one. In this case, the intragroup contribution to the kinetic energy as well as the contribution of intragroup forces to the virial are removed from the pressure definition (which affects the fluctuations of this quantity, but not its average value). As shown elsewhere[118,119] (the equations reported therein should be altered by halving the virial and replacing $\mathbf{r}_{ij}$ by $-\mathbf{r}_{ij}$ to match the present conventions), the group-based virial tensor can be calculated from the corresponding atom-based tensor by adding a simple correction term, which depends on the overall atomic forces and on the pressure group definitions. More precisely, the group-based virial tensor is given by

$$\underline{\mathcal{W}} = \underline{\mathcal{W}}* + \tfrac{1}{2}\sum_{I\alpha}\mathbf{f}_{I\alpha}\otimes\mathbf{d}_{\mathbf{I}\alpha}, \tag{12.37}$$

where $I\alpha$ denotes atom α in pressure group I, $\mathbf{f}_{I\alpha}$ the overall force on atom $I\alpha$, and $\mathbf{d}_{\mathbf{I}\alpha}$ the coordinate vector of atom $I\alpha$ relative to the center of mass of the gathered pressure group I containing this atom. The *gathered* representation of the pressure group is generated by following the atoms as they drift throughout the periodic system. The group-based pressure tensor is then calculated as

$$\underline{\mathcal{P}} = \frac{2}{\mathcal{V}}(\underline{\mathcal{K}} - \underline{\mathcal{W}}) \tag{12.38}$$

where $\underline{\mathcal{K}}$ is the group-based internal kinetic energy tensor, defined as

$$\underline{\mathcal{K}} = \tfrac{1}{2}\sum_{I=1}^{N_s}\left(\sum_{\alpha=1}^{N_\alpha(I)} m_i\right)^{-1}\left(\sum_{\alpha=1}^{N_\alpha(I)} m_i\dot{\mathbf{r}}_i\right)\otimes\left(\sum_{\alpha=1}^{N_\alpha(I)} m_i\dot{\mathbf{r}}_i\right) \tag{12.39}$$

where N_s is the number of pressure groups and $N_\alpha(I)$ the number of atoms in pressure group I.

Although the atom-based pressure tensor $\underline{\mathcal{P}}*$ is typically symmetric, this is generally not the case for the group-based pressure tensor $\underline{\mathcal{P}}$ (although the antisymmetric contribution to this tensor should vanish upon time averaging). When applying a barostat algorithm, the antisymmetric component of $\underline{\mathcal{P}}$ should induce an overall rotation of the computational box (which would alter the box angular momentum), while the symmetric component results in a deformation of the box (which conserves the box angular momentum). In practice, the overall rotation of the box is rather a nuisance, and is avoided by symmetrizing the tensor ($\underline{\mathcal{P}} \rightarrow [\underline{\mathcal{P}} + {}^T\underline{\mathcal{P}}]$) prior to application of the barostat algorithm[121] where the T presuperscript indicates the transpose of the matrix.

12.5. Pressure scaling

For compatibility with experiment, it is often desirable to sample configurations from the isothermal-isobaric ensemble (constant temperature and pressure). Thermostat algorithms have been described in Sec. 12.2. A modification of the basic MD scheme with the purpose of maintaining the pressure constant (on average) is called a barostat algorithm. The various methods for carrying out MD at constant pressure are based on the same principles as the constant temperature schemes with the role of temperature played by the pressure and the role of the atomic velocities played by the atomic positions.

The use of a barostat is only applicable to simulations under periodic boundary conditions. In the GROMOS implementation, the various options for the variations for the box parameters (and the associated scaling of atomic coordinates) involved in the use of a barostat are: (1) no variations of the box parameters; (2) isotropic scaling, that is, identical relative variations of the box-edge lengths only; (3) semi-anisotropic scaling, that is, two box-edge lengths are scaled identically (or left constant) while the third box-edge length is scaled individually (or left constant); (4) partially anisotropic scaling, that is, independent relative variations of the box-edge lengths only; (5) fully anisotropic scaling, that is, independent variations of all box parameters (box-edge lengths, box angles, and Euler angles). For a truncated-octahedral box, only the first two options are allowed. For a rectangular box, only the first four options are allowed. For a triclinic box, all options are allowed. In the latter case, variations in the box shape are accompanied by variations in the box Euler angles, so as to guarantee that the barostat does not introduce a rigid-body rotational component to the box orientation. Note, however, that the location of the box center of mass is affected by any type of coordinate scaling. Like in the temperature scaling the *weak-coupling method* (Berendsen barostat[11]) is again very simple.

The atomic equations of motion are modified such that the net result on the system is a first-order relaxation of the pressure $\mathcal{P}$ towards the preset reference value P_0

$$\frac{d\mathcal{P}(t)}{dt} = \tau_P^{-1}[P_0 - \mathcal{P}(t)] \tag{12.40}$$

As described in Sec. 12.4 the hydrostatic pressure tensor can be calculated with Eq. 12.38 using the virial theorem,

$$\mathcal{P}_{\alpha\beta}(t) = 2[\mathcal{K}_{\alpha\beta}(t) - \mathcal{W}_{\alpha\beta}(t)]/\mathcal{V}(t) \tag{12.41}$$

where $\alpha = x, y$ or z and $\beta = x, y$ or z, $\underline{\mathcal{W}}(t)$ is the virial tensor (Sec. 12.4) and $\mathcal{V}$ (t) is the volume of the computational box. For a rectangular box we have (Sec. 4.4.2.1)

$$\mathcal{V}(t) = a(t)b(t)c(t) \tag{12.42}$$

with a, b and c the lengths of the x-, y- and z-axes of the computational box. For a truncated octahedron we have Sec. 4.4.2.2

$$\mathcal{V}(t) = \tfrac{1}{2}a^3(t). \tag{12.43}$$

and for a triclinic box (Sec. 4.4.1.3)

$$\begin{aligned}\mathcal{V}(t) &= a(t)b(t)c(t)[1 - \cos^2\alpha(t) - \cos^2\beta(t) - \cos^2\gamma(t) \\ &+ 2\cos\alpha(t)\cos\beta(t)\cos\gamma(t)]^{1/2}\ .\end{aligned} \tag{12.44}$$

The pressure components along the x-, y- and z-axes, that is, the diagonal elements $\mathcal{P}_{xx}$, $\mathcal{P}_{yy}$ and $\mathcal{P}_{zz}$ can be defined correspondingly,

$$\mathcal{P}_{\alpha\alpha}(t) = 2[\mathcal{K}_{\alpha\alpha}(t) - \mathcal{W}_{\alpha\alpha}(t)]/\mathcal{V}(t) \tag{12.45}$$

where $\alpha = x, y$ or z. The isotropic hydrostatic pressure $\mathcal{P}$ can then be calculated with

$$\mathcal{P}(t) = \frac{1}{3}(\mathcal{P}_{xx}(t) + \mathcal{P}_{yy}(t) + \mathcal{P}_{zz}(t)) \tag{12.46}$$

as described in Eq. 12.34. The isothermal compressibility κ_T will relate a change in pressure $\Delta\mathcal{P}$ at constant temperature to a change in volume $\Delta\mathcal{V}$,

$$\Delta\mathcal{P}(t) = \frac{-\Delta\mathcal{V}(t)}{\kappa_T \mathcal{V}(t)} \tag{12.47}$$

and a change in volume can be obtained by scaling the atomic coordinates and the edges of the computational box with a factor μ,

$$\Delta\mathcal{V}(t) = [(\mu(t))^3 - 1]\mathcal{V}(t). \tag{12.48}$$

Discretizing Eq. 12.40 using the MD time step Δt and solving the *pressure scaling factor* $\mu(t)$ from Eq. 12.40 and Eq. 12.46, Eq. 12.47 yields

$$\mu(t) = \left[1 - \kappa_{_T}\frac{\Delta t}{\tau_p}[P_0 - \mathcal{P}(t)]\right]^{\frac{1}{3}} \tag{12.49}$$

in the isotropic case,

$$\mu_\alpha(t) = \left[1 - \kappa_{_T}\frac{\Delta t}{\tau_p}[P_{0,\alpha\alpha} - \mathcal{P}_{\alpha\alpha}(t)]\right]^{\frac{1}{3}} \tag{12.50}$$

in case the x-, y- and z-dimensions are scaled separately and

$$\mu_{\alpha\beta}(t) = \left[\delta_{\alpha\beta} - \kappa_{_T}\frac{\Delta t}{\tau_p}[P_{0,\alpha\beta} - \mathcal{P}_{\alpha\beta}(t)]\right]^{\frac{1}{3}} \tag{12.51}$$

in case of fully anisotropic scaling. In the case of semi anisotropic scaling where two axes α and β are scaled with the same scaling factor

$$\mu_{\alpha,\beta}(t) = \left[1 - \kappa_{_T}\frac{\Delta t}{\tau_p}[(P_{0,\alpha\alpha} - \mathcal{P}_{\alpha\alpha}(t)) + (P_{0,\beta\beta} - \mathcal{P}_{\beta\beta}(t)))/2]\right]^{\frac{1}{3}}. \tag{12.52}$$

The elements of $P_{0,\alpha\beta}$ can be set separately. This is done with a reference pressure tensor PRES0(,) in the PRESSURESCALE block.

The scaling of the atomic coordinates $\mathbf{r}_i(t)$ and the box edges $\mathbf{a}(t)$, $\mathbf{b}(t)$ and $\mathbf{c}(t)$ with the factor $\mu(t)$ at each MD step will make the pressure $\mathcal{P}(t)$ or the components $\mathcal{P}_{\alpha\beta}(t)$ relax towards $\mathcal{P}_0$ (or $\mathcal{P}_{0,\alpha\beta}$), the relaxation rate being controlled by the ratio of the isothermal compressibility κ_T of the system and the chosen *pressure relaxation time* τ_P. The value of κ_T may not be accurately known, but this is no problem, since τ_P is an adjustable parameter. The values of the isothermal compressibility of water at a pressure of 1atm = 1.01325 Bar = 0.0610184 kJ mol^{-1}nm^{-3} and a temperature of 293K is for example κ_T = 45.91 10^{-6} Bar^{-1} = 76.24 10^{-5} (kJmol^{-1}nm^{-3})$^{-1}$**[122]**. The compressibility of proteins is about 10 to 20% of that of water**[123]**. So, for a system consisting of half protein half water we find an estimated value of $\kappa_{_T}$ = 45.75 10^{-5} (kJmol^{-1}nm^{-3})$^{-1}$. The switch to set the compressibility κ_T is called COMP. The value of τ_P, TAUP, should be chosen sufficiently small (strong coupling) to achieve the required average pressure, but on the other hand sufficiently large (weak coupling) to avoid disturbance of the properties of the system by the coupling to the pressure bath.**[11]** Since the definition of the pressure, Eqs. 12.41, 12.45), depends on the kinetic energy, the pressure coupling should not be stronger than the temperature coupling,

$$\tau_p > \tau_{_T}. \tag{12.53}$$

Typical values for τ_P are 0.4-0.5 ps.**[11]**

The following switches of the PRESSURESCALE block in MD++ have not been described yet (see Vol. 4):
The switch COUPLE is used to control the pressure scaling: COUPLE=off(0), no pressure calculation or scaling
COUPLE=calc(1), pressure calculation, but no scaling

COUPLE=scale(2), pressures calculation and scaling.
The switch SCALE chooses the scaling applied:
SCALE=off(0), no pressure scaling
SCALE=iso(1), isotropic pressure scaling
SCALE=aniso(2), anisotropic pressure scaling (x-, y-, z-axes, no angle deformation)
SCALE=full(3), fully anisotropic pressure scaling
SCALE=semianiso(4), semi-anisotropic pressure scaling
SEMI(1..3) gives the settings for the semi anisotropic pressure scaling. The 3 numbers can have values 0, 1 or 2 and represent the x-, y- and z-axes. If one of the numbers is 0, the according axis is not scaled. If two have the same number, they are scaled together.

12.6. MD algorithms

The *algorithm for* MD *simulation* based on the leap-frog integration scheme, which may include coupling to a temperature bath (indicated by the symbol T) and a pressure bath (indicated by the symbol P), and which may include the application of distance constraints (indicated by the symbol C) using one of the constraining methods described in Chap. 10 can be summarized by the following steps. The application of periodic boundary conditions is indicated by the symbol B. Writing data to file is indicated by the symbol W.

0. The positions $\mathbf{r}_i(t_n)$ and velocities $\mathbf{v}_i(t_n - \Delta t/2)$ for all atoms are given. Calculate the kinetic energies $\mathcal{K}(t_n - \Delta t/2)$, and the temperatures $T(t_n - \Delta t/2)$. The initial step number is zero, $n = 0$.

0P. The box lengths $\mathbf{a}(t_n),\ \mathbf{b}(t_n),\ \mathbf{c}(t_n)$ are given.

0C. The mentioned positions satisfy the constraints and the mentioned velocities, temperatures and kinetic energies do not contain components along the constraints.

0B. The solvent molecules may not be split by the periodic boundaries, they must be covalently connected. The same condition must hold for the atoms of a solute charge group.

1. Calculate the translational ($\mathcal{K}_{tr}(t_n - \Delta t/2)$) and rotational ($\mathcal{K}_{ir}(t_n - \Delta t/2)$) kinetic energy for the relevant temperature groups, using Eq. 12.6 and Eq. 12.7. Calculate the relative positions $d_{i\alpha}$, for solute and solvent as preparation for the virial calculation (Eq. 12.37).

1B. If required, apply the periodic boundary conditions to put the solute charge groups and solvent molecules in the central computational box (see Sec. 4.4.1.2).

2. Calculate the (unconstrained) forces from the potential energy function $\mathcal{V}(\boldsymbol{r}(t))$

$$\mathbf{f}_i(t_n) \quad = \quad -\frac{\partial \mathcal{V}(\boldsymbol{r}(t_n))}{\partial \mathbf{r}_i} \tag{12.54}$$

(Eq. 12.54)) using the nearest image convention in case of periodic boundary conditions, and at the same time

2P. calculate the atomic or group-based virial $\mathcal{W}^*(t_n)$ or $\mathcal{W}(t_n)$.

3C. If required, apply positions constraints (see Sec. 10.2), *i.e.* set forces and velocities to zero.

4. Determine the (unconstrained) velocities $\mathbf{v}_i(t_n + \Delta t/2)$ from Eq. 12.1.

5T. In case the group-based translational kinetic energy and the internal, rotational kinetic energy are jointly coupled to a temperature bath, scale the velocities $\mathbf{v}_i(t_n + \Delta t/2)$ of the atoms that are coupled to a temperature bath with the appropriate temperature scaling factor $\lambda(t_n - \Delta t/2)$ based on the relevant contributions to $\mathcal{K}(t_n - \Delta t/2)$. The calculation of the scaling factors is dependent on the thermostat chosen, see Sec. 12.2.

6. Determine the (unconstrained) positions $\mathbf{r}_i(t_n + \Delta t)$ from Eq. 12.2.

7C. Make the positions satisfy the constraints (see Sec. 10.3.1)

$$C(\boldsymbol{r}(t_n); \boldsymbol{r}(t_n + \Delta t); \boldsymbol{r}(t_n + \Delta t)) \tag{12.55}$$

8C. Calculate the constrained velocities

$$\mathbf{v}(t_n + \Delta t/2) \quad = \quad [\boldsymbol{r}(t_n + \Delta t) - \boldsymbol{r}(t_n)]/\Delta t \tag{12.56}$$

9. Calculate the kinetic energies $\mathcal{K}(t_n + \Delta t/2)$ to determine the scaling factors $\lambda(t_n + \Delta t/2)$ and the temperatures $T(t_n + \Delta t/2)$.

10P. Calculate the volume of the periodic box $\mathcal{V}(t_n)$, and calculate the atomic or group-based pressure tensor $\mathcal{P}(t_n)$ from Eq. 12.41 and Eq. 12.45 using the appropriate kinetic energy $\mathcal{K}_{tr}$ calculated in step 1 and the virial from step 2P.

11P. Scale the atomic positions $\mathbf{r}_i(t_n + \Delta t)$ and the box lengths $\mathbf{a}(t_n)$, $\mathbf{b}(t_n)$, $\mathbf{c}(t_n)$ to obtain $\mathbf{a}(t_n + \Delta t)$, $\mathbf{b}(t_n + \Delta t)$, $\mathbf{c}(t_n + \Delta t)$ with the appropriate pressure scaling factor $\mu(t_n)$, Eq. 12.48, using the pressure (components) from step 10P. If atoms are positionally restrained or kept fixed (Sec. 10.2), scale their reference positions also with the pressure scaling factor.

12. In case of perturbation (see Sec. 14.6), possibly change λ-values and update the masses and individual λ-values.

13. Calculate the total energies and update the energy averages. The kinetic energy at t_n is calculated as the average of the kinetic energies at $t_n - \frac{1}{2}\Delta t$ and $t_n + \frac{1}{2}\Delta t$[124].

14W. At the end of a MD step print the energies (ENERGY block at time t_n), pressure and volume (PRESSURE block) and scaling data (MULTIBATHCOUPLING block and PCOUPLE block), replica data (REMD block). Write the configuration $\boldsymbol{r}(t_n + \Delta t)$ to the trajectory file.

15W. If at the end of the run, write the final configuration $\boldsymbol{r}(t_n + \Delta t)$ and velocities $\mathbf{v}(t_n + \Delta t/2)$ to a single configuration file and write perturbation data: energy derivatives with respect to λ at time t_n, λ-values at time $t_n + \Delta t$, atom-atom distance restraints data at t_n, ^3J-coupling constant restraints data at t_n, S^2-order parameter restraints data at t_n, local-elevation data at t_n, replica data at t_n, X-ray scaling constants at t_n and R-values at t_n.

15. Increase the time to $t_{n+1} = t_n + \Delta t$ and the step number n to *n+1*.

When a set of atoms is to be kept fixed, their positions $\mathbf{r}_i$ are kept equal to their reference positions and the forces $\mathbf{f}_i$ on them and their velocities $\mathbf{v}_i$ are kept equal to zero at each MD step. Their inverse masses are set to zero in order to immobilize these atoms when their position might be up for resetting by the constraint procedure in case a fixed atom is involved in a constraint to a non-fixed atom.

12.7. Initialization, equilibration and sampling

An MD simulation starts with initial atomic positions and velocities. Although the results should be independent of these, it is good practice to choose the initial configuration as representative for the molecular system in equilibrium as possible. *Strain* in the configuration *should be removed* by a short (10-100 steps) energy minimization in order to avoid conversion of the strain energy into kinetic energy leading to a high temperature which will induce unsollicited barrier crossings, dihedral angle transitions, etc. The reading of the initial configuration in GROMOS is controlled in the block INITIALISE. If no *velocities* are available, they can be taken *from a Maxwell distribution* of a temperature T_i (TEMPI)

$$P(\mathbf{v}_j) \quad = \quad [2\pi k_B T_i/m_j]^{-\frac{3}{2}} exp[-m_j \mathbf{v}_j^2/(2k_B T_i] \tag{12.57}$$

where $P(\mathbf{v}_j)$ is the probability of occurrence of velocity $\mathbf{v}_j$ of atom j. Mathematically, this velocity distribution has the form of a product of 3 Gaussian distributions for the components v_{xj}, v_{yj} and v_{zj} with the standard deviation σ given by

$$\sigma \quad = \quad [k_B T_i/m_j]^{\frac{1}{2}} \tag{12.58}$$

Given the atomic masses, Boltzmann's constant, the initial temperature (TEMPI) and a random number generator seed (IG), Maxwellian initial velocities $\mathbf{v}_i(t_0 - \Delta t/2)$ can be generated if the variable NTIVEL = 1. If NTIVEL = 0 the initial velocities will be read from single-configuration file. This option is required to continue an MD simulation without a discontinuity in the trajectory.

The initial configuration and velocities may not satisfy the solute or solvent constraints. Besides, the atoms of charge groups of the solute may be split by the periodic box, Sec. 4.4.1.2. In the block INITIALISE shaking of the initial configuration is controlled with the switch NTISHK:

NTISHK	=	0:	initial coordinates $\boldsymbol{r}(t_0)$ and velocities $\boldsymbol{v}(t_0 - \Delta t/2)$ will not be shaken.
	=	1:	initial coordinates $\boldsymbol{r}(t_0)$ will be shaken.
	=	2:	initial velocities $\boldsymbol{v}(t_0 - \Delta t/2)$ will be shaken.
	=	3:	initial coordinates $\boldsymbol{r}(t_0)$ and velocities $\boldsymbol{v}(t_0 - \Delta t/2)$ will be shaken; the solute charge groups are reassembled if necessary.

By using the switch NTICOM the translational motion of and the rotational motion about the centre of mass can be removed from the initial velocities:

NTICOM	=	0:	initial centre of mass motion will not be removed.
	=	1:	initial translational motion of the centre of mass will be removed.
	=	2:	in addition, initial rotational motion about the centre of mass will be removed.

The *removal of centre of mass* motion during MD simulation is controlled within the COMTRANSROT block with variable NSCM:

NSCM	=	0:	centre of mass motion will not be removed.
	<	0:	centre of mass translation and rotation will be removed every NSCM steps.
	>	0:	centre of mass translation will be removed every NSCM steps.

For long MD simulations removal of the centre of mass motion should be regularly (every 10-100 ps) done. If atoms are kept fixed in a simulation, the centre of mass motion is not removed. For an in vacuo simulation without centre of mass translation or rotation, NDFMIN = 6 degrees of freedom should be subtracted from the total number of degrees of freedom when calculating the temperature from the kinetic energy. For a simulation using periodic boundary conditions without centre of mass translation, NDFMIN = 3 should be used. For a simulation using an extended wall region of positionally restrained or fixed atoms, NDFMIN = 0 should be used. Alternatively, the simulation can be performed under rototranslational constraints, as described in Sec. 10.7.

In order to *continue an* MD *simulation* the single-configuration file must contain the appropriate data for a continuation run:

- atomic coordinates at time t_0,
- atomic velocities at time $t_0 - \Delta t/2$,
- dimensions of the periodic box,
- setting for lattice shift vectors,
- perturbation data (cumulative energy derivatives at time $t_0 - \Delta t$, λ-values at time t_0),
- atom-atom distance restraint data (averages at time $t_0 - \Delta t$),
- ^3J-coupling constant restraint data (averages at time $t_0 - \Delta t$),
- S^2-order parameter restraint data (averages at time $t_0 - \Delta t$),
- local-elevation data (conformations visited so far, at time $t_0 - \Delta t$),
- setting of positions and orientations for roto-translational constraints,
- replica data,
- X-ray scaling constants and R-values (averages at time $t_0 - \Delta t$).

These data are written after the last (NSTLIM-th) step to a single-configuration file in order to allow for further continuation simulation.

The number of MD time steps of size Δt (DT) is to be specified in NSTLIM within the block STEP. The initial time point t_0 may be specified in the variable T for administrative purposes. It does not play a role in the algorithm, which only counts steps. Before the first step the non-bonded interaction charge group pair list must be generated. It will be regenerated every NSNB (5-10) steps (PAIRLIST block).

In order to *start the simulation* in a gentle way, the initial velocities can be taken from a *Maxwell distribution of low temperature* (say 50K) and subsequently the reference temperature can be raised after short

simulation periods (few ps) in a few steps (100K, 200K) to room temperature. At the same time *harmonic position restraining* can be applied to parts, e.g. the solute of the system in order to prevent possible distortions due to the release of strain present in the initial configuration, e.g. due to bad solute-solvent contacts. The *coupling to the* temperature and pressure *baths can be made strong* ($\tau_T \approx 0.01$ps, $\tau_\mathcal{P} \approx 0.05$ps) during the initial stage (few ps) of a simulation in order to allow for a rapid transfer of excess heat and pressure to the respective baths.

Ideally, the *equilibration period* of a simulation should be longer than the relaxation time of the system in order to secure equilibration. It will depend on the relaxation time of the property one is interested in. Some properties, such as the kinetic energy, require short (picoseconds) equilibration times, whereas others, such as dielectric properties, may require longer times, of the order of hundreds of picoseconds. During a simulation a number of *quantities*, such as the kinetic energy, the potential energy and the various terms contributing to it, the diffusion away from the initial structure, can be *monitored* to obtain a picture of the stability of a simulation.

At every NTPR-th time step a number of quantities are printed. In the TIMESTEP block MD time step number and time at the current MD step are given. In the ENERGIES block the following quantites are printed at time t:

`E_Total`	= total energy of the molecular system
`E_Kinetic`	= total kinetic energy of the molecular system
`E_Potential`	= total potential energy of the molecular system
`E_Covalent`	= total energy of covalent terms (solutes)
`E_Bonds`	= total energy of bond-stretching terms (solutes)
`E_Angles`	= total energy of bond-angle bending terms (solutes)
`E_Improper`	= total energy of improper (harmonic) dihedral angle terms (solutes)
`E_Dihedral`	= total energy of (trigonometric) dihedral angle terms (solutes)
`E_Crossdihedral`	= total energy of cross-dihedral angle terms (solutes)
`E_Non-bonded`	= total energy of nonbonded terms (solutes)
`E_Vdw`	= total energy of van der Waals interaction terms (at time t)
`E_El (RF)`	= total energy of the electrostatic interaction terms, $\mathcal{V}^{(ele)}$, in the *reaction-field* electrostatic scheme
`E_El (LS)`	= total energy of the electrostatic interaction terms, $\mathcal{V}^{(ele)}$, in the *lattice sum* electrostatic scheme
`E_El (pair)`	= pairwise potential energy contribution, $\mathcal{V}^{(ele,pws)}$, to the total electrostatic energy interaction term $\mathcal{V}^{(ele)}$
`E_El (real-space)`	= real-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ in the *lattice sum* electrostatic scheme
`E_El (k-space)`	= reciprocal-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ in the *lattice sum* electrostatic scheme
`E_El (A term)`	= A-term potential energy contribution to $\mathcal{V}^{(ele,pws)}$ in the *lattice sum* electrostatic scheme

E_El (lattice sum self)
= self potential energy contribution, $\mathcal{V}^{(ele,slf)}$, to the total electrostatic energy interaction term $\mathcal{V}^{(ele)}$

E_El (surface term)
= surface potential energy contribution, $\mathcal{V}^{(ele,srf)}$, to the total electrostatic energy interaction term $\mathcal{V}^{(ele)}$

E_Polarisation self
= total energy of polarisation self term

E_Special = total energy of special terms

E_SASA = total energy of SASA terms

E_Volume = total energy of SASA volume term

E_Constraints = total energy due to constraints in the molecular system

E_Distanceres = total energy of atom-atom distance restraint term

E_Disfieldres = total energy of distance-field restraint

E_Dihrest = total energy of dihedral restraining term

E_Posrest = total energy of atom position restraining term

E_EDS reference = total energy of EDS reference state

E_Jrest = total energy of ^3J-value restraining term

E_X-ray restraints = total energy of X-ray restraining term

E_Local elevation = total energy of local elevation term

E_Order-parameter rest.
= total energy of S^2-order parameter restraining term

E_RDCrest = total energy of RDC restraints

E_Symmetry restraints
= total energy of symmetry restraints

E_EDS reference = EDS reference energy

E_Entropy = total entropy term

E_QM = total energy of QM/MM interactions

When a free energy perturbation calculation is performed, the free energy perturbation parameter λ and the derivatives of the terms listed above with respect to λ are also written out.

In the TEMPERATURES block active temperature baths coupled to temperature groups in the system are given. For each temperature bath the total kinetic energy (EKIN), translational kinetic energy (EKIN-MOL-TR), sum of internal and rotational kinetic energy (EKIN-MOL-IR) and the corresponding temperatures (T, T-MOL-TR and T-MOL-IR) as well as the temperature scaling factors (SCALE) are printed for the degrees of freedom that are coupled to each bath.

In the PRESSURE block molecular kinetic energies, virial, pressure tensor, volume of the periodic box and total pressure are printed.

The centre of mass motion is regularly printed, and a least-squares fit of the total energy as function of the step number to a straight line is also regularly carried out and printed. This allows an examination of the

conservation of translational and rotational momentum and of the total energy. The latter should be conserved in MD simulations without coupling to temperature and pressure baths, without time-averaging and local-elevation terms, without fixed atoms, and in the absence of cut-off noise and noise due to constraints. If in addition no atoms are positionally restrained, the translational momentum should also be conserved. If in addition no periodic boundary conditions are used, the rotational momentum should also be conserved. The dihedral-angle transitions are not printed to the output file but to a special trajectory (see Vol. 5).

The switch array NTF[1..6] controls the presence (=1) or absence (=0) in Eq. 3.5 of the different terms of the GROMOS force field, $\mathcal{V}^{(phys)}$ in Eq. 3.4:

NTF[1]: bond-stretching interaction

NTF[2]: bond-angle bending interaction

NTF[3]: improper (harmonic) dihedral-angle bending interaction

NTF[4]: (trigonometric) dihedral-angle torsion interaction

NTF[5]: non-bonded interaction involving charges

NTF[6]: non-bonded interaction

The special terms $V^{special}$ in Eq. 3.4 are controlled by separate switches, which were discussed in Chap. 9.

NTPOR: atom position restraining or fixing

NTDIR: atom-atom distance restraining

NTDFR: distance-field restraining

NTDLR: dihedral-angle restraining

NTJVR: ^3J-coupling constant restraining

NTOPR: S^2-order parameter restraining

At the end of a simulation the averages and root-mean-square fluctuations of the quantities mentioned above are printed together with the average temperatures and pressure and their r.m.s. fluctuations. The final configuration and velocities and other quantities that are needed to continue the simulation are saved (see Vol. 4).

The *sampling period* or analysis period of a simulation should be long enough to allow for an appropriate sampling of the property of interest, that is, the sampling period should be longer than the relaxation time of the property of interest.[125] The results are generally analyzed by taking time averages over the simulation or parts of a simulation. By monitoring the time averages as a function of the length of the averaging period it can be tested whether the time averages are converging.[126]

In the WRITETRAJ block there are several switches which control writing of trajectories of data (coordinates, velocities, forces, energies and free energies). The switches NTWX and NTWSE control the writing of configurations. For NTWX $>$ 0 solute and solvent coordinates are printed every NTWX steps, beginning at step n = 0 and ending at step n = NSTLIM-|NTWX|. For NTWX $<$ 0 only solute coordinates are printed, which may reduce disk space requirements. When searching conformational space, one would rather save low energy configurations than configurations at regular intervals. For NTWSE = 0 normal coordinate trajectory is written out and for NTWSE $>$ 0 a minimum-energy coordinate and energy trajectory is written out. The switch NTWV controls the writing of velocities. If NTWV $>$ 0 solute and solvent velocities are written out, beginning at step n = 0, and ending at step n = NSTLIM - |NTWV|. If NTWV $<$ 0 only solute velocities are written out. The switch NTWF controls writing of the force trajectory. For NTWF $>$ 0 the FREEFORCERED and CONSFORCERED blocks (see Vol. 4) are written out every NTWF steps, for NTWF $<$ 0 only solute forces are written out. The switch NTWE controls the writing of energy, pressure and volume data. If NTWE $>$ 0, every NTWE-th time step the ENERGY03 and VOLUMEPRESSURE03 blocks (see Vol. 4) are saved, beginning at step n = 0, and ending at step n = NSTLIM - NTWE. The switch

NTWG controls the writing of free energy data. If NTWG > 0 every NTWG-th time step the FREEENERDERIVS03 block (see Vol. 4) is saved, beginning at step n = 0, and ending at step n = NSTLIM - NTWG. The switch NTWB controls writing of a block-averaged energy trajectory. If NTWB > 0 block-averaged energies and free energies (if NTWG > 0) are printed every NTWB steps.

CHAPTER 13

Stochastic Dynamics

13.1. Introduction

Stochastic dynamics (SD) simulation with an empirical energy function such as Eq. 3.4 is also much used to study the equilibrium structural, dynamic and thermodynamic properties of molecules. It is an extension of MD simulation, a frictional force and a randomly fluctuating force are added to the forces derived from $\mathcal{V}^{(phys)}(\mathbf{r})$ and $\mathcal{V}^{(spec)}(\mathbf{r})$, which leads to the Langevin equations of motion Eq. 2.13, as discussed in Sec. 2.4. These additional forces approximate partly the effect of degrees of freedom that are not explicitly treated in the simulation on the explicitly treated degrees of freedom. A major application is the replacement of explicit solvent molecules in an MD simulation of a (macro)molecule in solution by (implicit) stochastic and frictional forces on the (macro)molecule in a SD simulation. SD simulation is also a useful method for searching conformational space for low energy conformers. Due to the random components in the forces, it may search a wider part of space than a comparable MD simulation. A third type of application of SD simulation is in conjunction with the extended wall region boundary condition (Sec. 4.3). The motion of the positionally restrained atoms of the wall region may be treated by the Langevin equation, so that it gets randomized and the wall region can exchange heat with the hypothetical environment (vacuum) outside the extended wall region. SD simulation may also be used to control the temperature, *i.e.* as a Langevin thermostat.[127]

Here, we only consider the differences of SD simulation with respect to MD simulation. In the first three applications of SD simulation mentioned above the application of periodic boundary conditions (Sec. 4.4) would not make much sense. However, it is possible to use periodic boundary conditions within SD. In case of using periodic boundary conditions it has to be considered that the calculations of the virial and pressure are incomplete. Thus, SD simulation at constant pressure, means coupling to a pressure bath, is impossible. The possibility to use weak coupling to a temperature bath is maintained for the following reason. Application of SD simulation with non-zero atomic friction coefficients γ_i implies an atomic coupling to a temperature T_{ref} (TEMPSD) in Eq. 2.15. However, atoms for which $\gamma_i = 0$ feel no random and no frictional force, so are not coupled to the temperature bath at T_{ref}. The combination of weak temperature coupling to a temperature bath at T_0, as discussed in Sec. 12.2, with SD simulation is a, be it not very elegant, option to redress the temperature of atoms for which $\gamma_i = 0$ and which are subject to noise. Therefore, only a simplified version of weak temperature coupling can be combined with SD.

The leap-frog SD algorithm is given in Sec. 13.2. The choice of the atomic friction coefficients γ_i is discussed in Sec. 13.3.

13.2. Leap-frog SD algorithm

Apart from a mean-force term, Langevin's equation of motion Eq. 2.13 differs from Newton's equation Eq. 2.8 by the occurrence of a stochastic force $\mathbf{f}_i^{st}(t)$ and a frictional force $-m_i\gamma_i\mathbf{v}_i(t)$. These extra terms make the SD version of the leap-frog scheme more complicated than the MD one[128]. The solution of Eq. 2.13 in terms of $\mathbf{v}_i(t)$ around time $t = t_n$ is

$$\begin{aligned}\mathbf{v}_i(t) &= \mathbf{v}_i(t_n)exp[-\gamma_i(t-t_n)] \\ &+ m_i^{-1}exp[-\gamma_i(t-t_n)]\int_{t_n}^{t} exp[-\gamma_i(t_n-t')][\mathbf{f}_i(t')+\mathbf{f}_i^{st}(t')]dt' \end{aligned} \tag{13.1}$$

Since the stochastic properties of the random force $\mathbf{f}_i^{st}(t')$ are given (postulated), the stochastic properties of the integral over $\mathbf{f}_i^{st}(t')$ can be obtained directly. The integral over the force $\mathbf{f}_i(t')$ is, as in Sec. 12.1 for

MD, obtained by expanding $\mathbf{f}_i(t')$ in a Taylor series around $t = t_n$ and omitting all terms beyond third order in the time step Δt in the positions, beyond second order in the velocities, and beyond first order in the forces. The SD equivalent of the *leap-frog velocity formula* Eq. 12.1 then becomes[128]

$$\begin{aligned} \mathbf{v}_i(t_n + \Delta t/2) &= \mathbf{v}_i(t_n - \Delta t/2)exp(-\gamma_i \Delta t) \\ &+ m_i^{-1}\mathbf{f}_i(t_n)\Delta t[1 - exp(-\gamma_i \Delta t)]/(\gamma_i \Delta t) \\ &- exp[-\gamma_i \Delta t]\mathbf{V}_i(t_n; -\Delta t/2) + \mathbf{V}_i(t_n; \Delta t/2) \end{aligned} \tag{13.2}$$

where

$$\mathbf{V}_i(t_n; \Delta t/2) \equiv m_i^{-1} exp(-\gamma_i \Delta t/2) \int_{t_n}^{t_n + \Delta t/2} exp[-\gamma_i(t_n - t')]\mathbf{f}_i^{st}(t')dt' \tag{13.3}$$

The equivalent of the *leap-frog position formula* Eq. 12.2 is obtained by integrating Eq. 2.9 using Eq. 13.1 for the velocity[128]

$$\begin{aligned} \mathbf{r}_i(t_n + \Delta t) &= \mathbf{r}_i(t_n) \\ &+ \mathbf{v}_i(t_n + \Delta t/2)\Delta t[exp(\gamma_i \Delta t/2) - exp(-\gamma_i \Delta t/2)]/(\gamma_i \Delta t) \\ &- \mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2) + \mathbf{R}_i(t_n + \Delta t/2; \Delta t/2) \end{aligned} \tag{13.4}$$

where

$$\mathbf{R}_i(t_n; \Delta t/2) \equiv (m_i \gamma_i)^{-1} \int_{t_n}^{t_n + \Delta t/2} [1 - exp[\gamma_i(t_n + \Delta t/2 - t')]]\mathbf{f}_i^{st}(t')dt'. \tag{13.5}$$

When the friction coefficient γ_i tends to zero, the SD *leap-frog scheme* Eq. 13.2 and Eq. 13.4 reduces to the MD *leap-frog scheme* Eq. 12.1 and Eq. 12.2. Due to condition Eq. 2.15, which connects the stochastic force $\mathbf{f}_i^{st}$ with the friction coefficient γ_i, the *stochastic integrals* Eq. 13.3 and Eq. 13.5 tend to zero for γ_i tending to zero.

When using the SD leap-frog integration scheme Eq. 13.2 and Eq. 13.4 it must be noted that the stochastic variable $V_i(t_n; -\Delta t/2)$ is correlated with the stochastic variable $\mathbf{R}_i(t_n - \Delta t/2; \Delta t/2)$, since they are different integrals of the stochastic force $\mathbf{f}_i^{st}(t')$ over the same time interval $(t_n - \Delta t/2, t_n)$. The same observation holds for the stochastic variables $\mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2)$ and $\mathbf{V}_i(t_n; \Delta t/2)$, which are integrals over $\mathbf{f}_i^{st}(t')$ over the same time interval $(t_n, t_n + \Delta t/2)$. This means that these correlated stochastic variables must be sampled in a correlated manner[128]

The probability distribution for the x-, y-, z-components of $\mathbf{R}_i(t_n - \Delta t/2; \Delta t/2)$, irrespective the value of $\mathbf{V}_i(t_n; -\Delta t/2)$, is

$$P(R_{ix}) = [2\pi\sigma_1^2]^{-\frac{1}{2}} exp[-R_{ix}^2/(2\sigma_1^2)], \tag{13.6}$$

and the conditional probability distribution for the x-, y-, z-components of $\mathbf{V}_i(t_n; -\Delta t/2)$, given the specific value $\mathbf{R}_i(t_n - \Delta t/2; \Delta t/2)$ sampled from Eq. 13.6, is

$$P(V_{ix}|R_{ix}) = [2\pi\sigma_2^2]^{-\frac{1}{2}} exp[-(V_{ix} - \sigma_3 R_{ix})^2/(2\sigma_2^2)] \tag{13.7}$$

where

$$\sigma_1^2 \equiv k_B T_{ref} m_i^{-1} C(\gamma_i \Delta t/2)/\gamma_i^2, \tag{13.8}$$

$$\sigma_2^2 \equiv k_B T_{ref} m_i^{-1} B(\gamma_i \Delta t/2)/C(\gamma_i \Delta t/2), \tag{13.9}$$

$$\sigma_3 \equiv \gamma_i D(\gamma_i \Delta t/2)/C(\gamma_i \Delta t/2), \tag{13.10}$$

with

$$B(\gamma_i \Delta t/2) \equiv \gamma_i \Delta t[exp(+\gamma_i \Delta t) - 1] - 4[exp(+\gamma_i \Delta t/2) - 1]^2, \tag{13.11}$$

$$C(\gamma_i \Delta t/2) \equiv \gamma_i \Delta t - 3 + 4exp(-\gamma_i \Delta t/2) - exp(-\gamma_i \Delta t) \tag{13.12}$$

and

$$D(\gamma_i \Delta t/2) \equiv 2 - exp(+\gamma_i \Delta t/2) - exp(-\gamma_i \Delta t/2). \tag{13.13}$$

The probability distribution for the x-, y-, z-components of $\mathbf{V}_i(t_n; \Delta t/2)$, irrespective the value of $\mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2)$, is

$$P(V_{ix}) = [2\pi\rho_1^2]^{-\frac{1}{2}} exp[-V_{ix}^2/(2\rho_1^2)], \tag{13.14}$$

and the conditional probability distribution for the x-, y-, z-components of $\mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2)$, given the specific value $\mathbf{V}_i(t_n; \Delta t/2)$ sampled from Eq. 13.14, is

$$P(R_{ix}|V_{ix}) = [2\pi\rho_2^2]^{-\frac{1}{2}} exp[-(R_{ix} - \rho_3 V_{ix})^2/(2\rho_2^2)] \tag{13.15}$$

where

$$\rho_1^2 \equiv k_B T_{ref} m_i^{-1}[1 - exp(-\gamma_i \Delta t)], \tag{13.16}$$

$$\rho_2^2 \equiv k_B T_{ref} m_i^{-1} B(-\gamma_i \Delta t/2)/[\gamma_i^2[1 - exp(-\gamma_i \Delta t)]], \tag{13.17}$$

$$\rho_3 \equiv D(-\gamma_i \Delta t/2)/[-\gamma_i[1 - exp(-\gamma_i \Delta t)]]. \tag{13.18}$$

The *algorithm for* SD *simulation* based on the leap-frog integration scheme, which may include weak coupling to a temperature bath of temperature T_0 (indicated by the symbol T), and which may include the application of distance constraints (indicated by the symbol C) using one of the constraining methods described in Chap. 10 can be summarized in the following steps.

0. The positions $\mathbf{r}_i(t_n)$ and velocities $\mathbf{v}_i(t_n - \Delta t/2)$ for all atoms are given. Calculate the kinetic energies $\mathcal{K}(t_n - \Delta t/2)$, and the temperatures $T(t_n - \Delta t/2)$. The initial step number is zero, $n = 0$.

0C. The mentioned positions satisfy the constraints and the mentioned velocities, temperatures and kinetic energies do not contain components along the constraints.

1. Remove the centre of mass motion: Calculate the molecular centre of mass positions for the temperature groups and the translational ($\mathcal{K}^{tr}(t_n - \Delta t/2; \mathcal{G}_T)$) and rotational $\mathcal{K}^{ir}(t_n - \Delta t/2; \mathcal{G}_T)$) kinetic energy. Calculate the relative positions $d_{i\alpha}(t_n)$, for use in Eq. 12.37.

1B. If required, apply the periodic boundary conditions (not for vacuum simulation).

2. Calculate the (unconstrained) forces from the potential energy function $\mathcal{V}(\boldsymbol{r})$

$$\mathbf{f}_i(t_n) = -\frac{\partial \mathcal{V}(\boldsymbol{r}(t_n)}{\partial \mathbf{r}_i} \tag{13.19}$$

3C. If required, apply positions constraints (see Sec. 10.2).

4. Sample the x-, y-, z-components of a vector $\mathbf{V}_i$ from a Gaussian distribution with zero mean and width σ_2^2 and determine

$$\mathbf{V}_i(t_n; -\Delta t/2) = \sigma_3 \mathbf{R}_i(t_n - \Delta t/2; \Delta t/2) + \mathbf{V}_i^{'} \tag{13.20}$$

5. Sample the x-, y-, z-components of a vector $\mathbf{V}_i(t_n; \Delta t/2)$ from a Gaussian distribution with zero mean and width ρ_1^2.

6. Determine the (unconstrained) velocities $\mathbf{v}_i(t_n + \Delta t/2)$ from Eq. 13.2.

7. Calculate the new positions excluding the contributions of the stochastic integrals in Eq. 13.4

$$\mathbf{r}_i(t_n + \Delta t) = \mathbf{r}_i(t_n) + \mathbf{v}_i(t_n + \Delta t/2)\Delta t E(\gamma_i \Delta t/2) \tag{13.21}$$

where

$$E(\gamma_i \Delta t/2) \equiv [exp(\gamma_i \Delta t/2) - exp(-\gamma_i \Delta t/2)]/(\gamma_i \Delta t) \tag{13.22}$$

8T. Scale the velocities $\mathbf{v}_i(t_n + \Delta t/2)$ of the atoms that are coupled to a temperature bath with the appropriate temperature scaling factor $\lambda(t_n - \Delta t/2)$, belonging to the groups of degrees of freedom that are jointly coupled to the bath (Sec. 12.2). The scaling factors are based on $\mathcal{K}(t_n - \Delta t/2)$ and depend on the thermostat chosen.

9C. Make the velocities satisfy the constraints by the following two steps. Perform

$$C(\boldsymbol{r}(t_n); \boldsymbol{r}(t_n + \Delta t); \boldsymbol{r}(t_n + \Delta t)) \tag{13.23}$$

Calculate the constrained velocities

$$\mathbf{v}(t_n + \Delta t/2) = [\mathbf{r}_i(t_n + \Delta t) - \mathbf{r}_i(t_n)]/[\Delta t E(\gamma_i \Delta t/2)] \tag{13.24}$$

10. Calculate the kinetic energies $\mathcal{K}(t_n + \Delta t/2)$ to determine the scaling factors $\lambda(t_n + \Delta t/2)$ and the temperatures $T(t_n + \Delta t/2)$.

11. Sample the x-, y-, z-components of a vector $\mathbf{R}_i^{'}$ from a Gaussian distribution with zero mean and width ρ_2^2 and determine

$$\mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2) = \rho_3 \mathbf{V}_i(t_n; \Delta t/2) + \mathbf{R}_i^{'} \tag{13.25}$$

12. Sample the x-, y-, z-components of $\mathbf{R}_i(t_n + \Delta t/2;\ \Delta t/2)$ from a Gaussian distribution with zero mean and width σ_1^2.

13. Add the stochastic integrals from Eq. 13.4 to obtain the (unconstrained) positions

$$\begin{aligned} \mathbf{r}_i(t_n + \Delta t) &= \mathbf{r}_i(t_n + \Delta t) - \mathbf{R}_i(t_n + \Delta t/2; -\Delta t/2) \\ &+ \mathbf{R}_i(t_n + \Delta t/2; \Delta t/2) \end{aligned} \tag{13.26}$$

14C. Make the positions satisfy the constraints

$$C(\boldsymbol{r}(t_n); \boldsymbol{r}(t_n + \Delta t); \boldsymbol{r}(t_n + \Delta t)) \tag{13.27}$$

15. In case of perturbation (see Sec. 14.6), possibly change λ-values and update the masses and individual λ-values.

16. Calculate the total energies and update the energy averages. The kinetic energy at t_n is calculated as the average of the kinetic energies at $t_n - \Delta t/2$ and $t_n + \Delta t/2$[124].

17W. At the end of an MD step print the energies (ENERGY block at time t_n), stochastic integrals $\mathbf{R}_i(t_n + \Delta t/2; \Delta t/2)$, scaling data (MULTIBATHCOUPLING block), replica data (REMD block). Write the configuration $\mathbf{r}_i(t_n + \Delta t)$ to the trajectory file.

18W. If at the end of the run, write the final configuration $\mathbf{r}_i(t_n + \Delta t)$ and velocities $\mathbf{v}_i(t_n + \Delta t/2)$ to a single configuration file and write perturbation data (energy derivatives with respect to λ at time t_n, the λ-values at time $t_n + \Delta t$), atom-atom distance restraints data at $\mathbf{t}_n$, ^3J-coupling constant restraints data at t_n, S^2-order parameter restraints data at t_n, local-elevation data at t_n, replica data at t_n, X-ray scaling constants at t_n and R-values at t_n.

18. Increase the time to $t_{n+1} = t_n + \Delta t$ and the step number $\mathcal{N}_t$ to $\mathcal{N}_t + 1$.

When computing the coefficients involving exponents of $\gamma\Delta t$, for small values of $\gamma_i\Delta t$ the numerical accuracy of the expressions is not guaranteed. Using an 48-bit mantissa they are only accurate to better than $1{:}10^6$ when $\gamma_i\Delta t > 0.05$. For $\gamma_i\Delta t < 0.05$ series expansion expressions are used.[128]

When weak coupling to a temperature bath is used, its temperature T_0(TEMP0m) should be taken equal to the SD reference temperature T_{ref} (TEMPSD).

13.3. Choice of atomic friction coefficient

When performing a SD simulation, atomic friction coefficients γ_i must be given. In the case of a solute in a stochastic solvent, the γ_i of the solute atoms should be proportional to the fraction ω_i of the atomic surface that is accessible to solvent and to the friction coefficient γ_{solv} of the solvent molecules,

$$\gamma_i(t) = \gamma_{solv}\omega_i(t) \tag{13.28}$$

An approximate value for γ_{solv} can be obtained from the experimental solvent viscosity η_{solv} using Stokes' law

$$\gamma_{solv} = 6\pi R_{solv}\eta_{solv}/m_{solv} \tag{13.29}$$

where R_{solv} is the Stokes radius and m_{solv} the mass of a solvent molecule. In this way a friction coefficient $\gamma_{solv} = 91$ ps^{-1} for H_2O and $\gamma_{solv} = 24$ ps^{-1} for CCl_4 was derived (T = 300K)[129]. The calculation of an exact solvent accessible surface area for each atom is an expensive task. Since the stochastic model of the solvent effects is anyway not correct in atomic detail, it makes no sense to accurately determine the ω_i. Therefore, one may use the approximate formula

$$\omega_i(t) = maximum(0, 1 - N_i^{nb}(t)/N^{nbref}) \tag{13.30}$$

in which the number of neighbour atoms of atom i at time t within a 3D sphere of radius R^{nbref}(RCUTF) is denoted by $N_i^{nb}(t)$ and N^{nbref}(NBREF) should be chosen equal to the number of neighbours at which atom i looses its contact with the solvent. For example, for $R^{nbref} = 0.3$ nm we use $N^{nbref} = 6$[129]. The atomic solvent accessible area weight factors $\omega_i(t)$ need not be calculated at every time step, since they should be a slowly varying function of the solute conformation. They will be recalculated every NSFR (100-1000) steps (if NTFR = 3).

An alternative is to specify the atomic friction coefficients GAM[i] in an atomic friction coefficients file, see Vol. 4, or to take them all equal to one value, CFRIC. This input variable also serves as overall weight factor. The switch NTFR controls the application of stochastic dynamics:

NTFR = 0, no SD simulation
= 1, SD simulation with γ_i = CFRIC
= 2, SD simulation with γ_i = CFRIC * GAM[i] and GAM[1..NR] is read from an atomic friction coefficients file
= 3, SD simulation with γ_i = CFRIC * ω_i from Eq. 13.30

CHAPTER 14

Free Energy Determination

14.1. Introduction

Several methods exist for calculating the free energy difference between two states A and B of a molecular system or between two molecular systems A and B, which are based on statistical mechanics.[130] In the so-called coupling parameter approach, the Hamiltonian Eq. 3.1 is made an analytical function of the coupling parameter λ,

$$\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda) = \mathcal{K}(\boldsymbol{p};\lambda)+\mathcal{V}(\boldsymbol{r};\lambda) \tag{14.1}$$

such that

$$\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda_A) = \mathcal{H}_A(\boldsymbol{p},\boldsymbol{r}) \tag{14.2}$$
$$\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda_B) = \mathcal{H}_B(\boldsymbol{p},\boldsymbol{r}) \tag{14.3}$$

where the Hamiltonians $\mathcal{H}_A$ and $\mathcal{H}_B$ characterize states or systems A and B. In order to keep the notation concise, we omit in the formulae of this chapter the explicit dependence of the Hamiltonian (terms) on the force-field parameters $\mathbf{s}$, the masses m and the box matrix $\underline{\mathcal{B}}$. Using the $\lambda-$dependent Hamiltonian Eq. 14.1, the (Helmholtz) free energy $\mathcal{F}$ becomes a function of λ,

$$\mathcal{F}(\lambda) = -k_BT\ln\mathcal{Z}(\lambda), \tag{14.4}$$

where the (λ-dependent) canonical partition function of the system of $\mathcal{N}_a$ atoms is given by

$$\mathcal{Z}(\lambda) = [h^{3\mathcal{N}_a}\mathcal{N}_a!]^{-1}\int\int e^{-\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda)/k_BT}d\boldsymbol{p}d\boldsymbol{r}. \tag{14.5}$$

where h is Planck's constant, and the factor $(\mathcal{N}_a!)^{-1}$ should be omitted when the atoms of the system are distinguishable. The free energy difference $\Delta\mathcal{F}_{BA}$ then reads

$$\begin{aligned}
\Delta\mathcal{F}_{BA} &= \mathcal{F}(\lambda_B)-\mathcal{F}(\lambda_A)\\
&= -k_BT\ln\left(\frac{\mathcal{Z}(\lambda_B)}{\mathcal{Z}(\lambda_A)}\right)\\
&= -k_BT\ln\left(\int\int e^{-[\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda_B)-\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda_A)]/k_BT}P(\boldsymbol{p},\boldsymbol{r};\lambda_A)d\mathbf{p}d\mathbf{r}\right)\\
&= -k_BT\ln\left(\left\langle e^{-[\mathcal{H}(\lambda_B)-\mathcal{H}(\lambda_A)]/k_BT}\right\rangle_{\lambda_A}\right).
\end{aligned} \tag{14.6}$$

The probability of the configuration $\boldsymbol{r}$ with momenta $\boldsymbol{p}$ in the (canonical) ensemble is defined as

$$P(\boldsymbol{p},\boldsymbol{r};\lambda) \equiv \frac{e^{-\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda)/k_BT}}{\int\int e^{-\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda)/k_BT}d\boldsymbol{p}d\boldsymbol{r}} \tag{14.7}$$

and the brackets $\langle ...\rangle_\lambda$ denote an ensemble average over an ensemble generated using the Hamiltonian $\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda)$. Eq. 14.6 is called the *free energy perturbation formula*, since the average involved will converge slowly unless the difference between the states described by $\mathcal{H}(\lambda_A)$ and $\mathcal{H}(\lambda_B)$ is small. The free energy difference $\Delta\mathcal{F}_{BA}$ can also be expressed as an ensemble average at $\lambda=\lambda_B$,

$$\Delta\mathcal{F}_{BA} = +k_BT\ln\left(\left\langle e^{-[\mathcal{H}(\lambda_A)-\mathcal{H}(\lambda_B)]/k_BT}\right\rangle_{\lambda_B}\right). \tag{14.8}$$

Formulae Eq. 14.6 and Eq. 14.8 can also be written as a perturbation formula for a change of Hamiltonian characterized by $\pm\Delta\lambda$,

$$\begin{aligned} \Delta\mathcal{F}_{\lambda\pm\Delta\lambda} &\equiv \mathcal{F}(\lambda\pm\Delta\lambda) - \mathcal{F}(\lambda) \\ &= -k_B T \ln\left(\left\langle e^{-[\mathcal{H}(\lambda\pm\Delta\lambda)-\mathcal{H}(\lambda)]/k_B T}\right\rangle_\lambda\right). \end{aligned} \tag{14.9}$$

Taking the derivative of $\mathcal{F}(\lambda)$ with respect to λ one finds

$$\mathcal{F}'(\lambda) \equiv \frac{\partial\mathcal{F}(\lambda)}{\partial\lambda} = \left\langle \frac{\partial\mathcal{H}(\lambda)}{\partial\lambda}\right\rangle_\lambda \tag{14.10}$$

which leads to the *thermodynamic integration formula*

$$\Delta\mathcal{F}_{BA} = \int_{\lambda_A}^{\lambda_B} \mathcal{F}'(\lambda)d\lambda = \int_{\lambda_A}^{\lambda_B} \left\langle \frac{\partial\mathcal{H}(\lambda)}{\partial\lambda}\right\rangle_\lambda d\lambda. \tag{14.11}$$

The formulae given above concern the canonical ensemble, so are applicable to trajectories from (N,V,T) simulations and give (Helmholtz) free energy differences. The corresponding formulae for (N,P,T) simulations giving Gibbs free energy or free enthalpy differences can be found in ref.[131].

In standard free energy perturbation calculations the *atomic masses* $m(\lambda)$ and *force-field parameters* $\mathbf{s}(\lambda)$ are made a function of the *coupling parameter* λ. The dependence of the Hamiltonian $\mathcal{H}(\boldsymbol{p},\boldsymbol{r};\lambda)$ on λ is given in Sec. 14.2.

When constraints are applied (Chap. 10), these appear formally as parameters in the Hamiltonian.[130] So, changing a distance constraint $\sigma_k(\boldsymbol{r};\lambda)$, Sec. 10.3.1, as a function of λ may lead to a change in free energy. Formulae for obtaining free *energy differences* due to differences in *distance constraints* (bond lengths) are given in Sec. 14.3.

Technical and *practical issues* with respect to the choice of pathway and of states A and B are discussed in Sec. 14.4. The use of the thermodynamic integration formula is discussed in Sec. 14.6 and that of the perturbation formula in Sec. 14.7.

In Sec. 14.8 it is briefly indicated how the various restraining functions of $\mathcal{V}^{(spec)}(\boldsymbol{r};\mathbf{s})$ can be used to bias the sampling along a chosen reaction coordinate R to obtain the free energy $\mathcal{F}(R)$ as a function of R using *umbrella sampling techniques*.[130,132] Sec. 14.9, finally, discusses a special use of the perturbation formula, in terms of enveloping distribution sampling.

14.2. Parameterization of the Hamiltonian

In free energy perturbation, the Hamiltonian $\mathcal{H}$ depends on the coupling parameter λ

$$\mathcal{H}(\mathbf{p},\mathbf{r};\lambda) = \mathcal{K}(\mathbf{p},\mathbf{r};\lambda) + \mathcal{V}(\mathbf{r};\lambda). \tag{14.12}$$

The parameter λ controls the change from state A to state B,

$$\begin{aligned} &\lambda = \lambda_A = 0: && \texttt{state A} \\ &\lambda = \lambda_B = 1: && \texttt{state B} \end{aligned} \tag{14.13}$$

Under these conditions we may write

$$\begin{aligned} \mathcal{H}(\mathbf{p},\mathbf{r};\lambda) &= \mathcal{K}(\mathbf{p},\mathbf{r};\lambda) + \mathcal{V}^{(b)}(\mathbf{r};\lambda) + \mathcal{V}^{(\theta)}(\mathbf{r};\lambda) + \mathcal{V}^{(\xi)}(\mathbf{r};\lambda) \\ &+ \mathcal{V}^{(\varphi)}(\mathbf{r};\lambda) + \mathcal{V}^{(nbd)}(\mathbf{r};\lambda) \\ &+ \mathcal{V}^{(Jr)}(\mathbf{r}) + \mathcal{V}^{(le)}(\mathbf{r}) + \mathcal{V}^{(pr)}(\mathbf{r}) \\ &+ \mathcal{V}^{(dr)}(\mathbf{r};\lambda) + \mathcal{V}^{(tr)}(\mathbf{r};\lambda) \end{aligned} \tag{14.14}$$

Of the last five terms, which are the special interaction terms described in Chap. 9, only $\mathcal{V}^{(dr)}$ (Sec. 9.3) and $\mathcal{V}^{(tr)}$ (Sec. 9.6) can be made $\lambda-$dependent, see Secs. 14.2.10 and Sec. 14.2.11.

We note that atom position fixing (Sec. 10.2) should not be used in a free energy perturbation calculation.

When distance constraints are applied, the kinetic energy $\mathcal{K}(\mathbf{p},\mathbf{r};\lambda)$ depends on the atomic coordinates $\mathbf{r}$. Using SHAKE, the momenta $\mathbf{p}$ will depend on the coordinates $\mathbf{r}$, since the velocities are calculated from shaken positions (Eq. 12.56 and Eq. 13.24). The $\lambda-$dependent kinetic energy is

$$\mathcal{K}(\mathbf{p},\mathbf{r};\lambda) \quad = \quad \sum_{i=1}^{\mathcal{N}_a} \tfrac{(\mathbf{p}_i)^2}{2m_i(\lambda)} = \sum_{i=1}^{\mathcal{N}_a} \tfrac{1}{2} m_i(\lambda)(\mathbf{v}_i)^2 \tag{14.15}$$

with

$$m_i(\lambda) = (1-\lambda)m_i^A + \lambda m_i^B. \tag{14.16}$$

The mass of atom i in state A is m_i^A and in state B it is m_i^B. For the derivatives with respect to λ (using $\mathcal{K}$ as a function of $\mathbf{p}$, not of $\mathbf{v}$) one finds[133]

$$\frac{\partial \mathcal{K}(\mathbf{p},\mathbf{r};\lambda)}{\partial \lambda} = -\sum_{i=1}^{\mathcal{N}_a} \tfrac{1}{2}(m_i^B - m_i^A)(\mathbf{v}_i)^2 \tag{14.17}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is, assuming that $\mathbf{p}$ and $\mathbf{r}$ correspond to λ,

$$\mathcal{K}(\mathbf{p},\mathbf{r};\lambda \pm \Delta\lambda) - \mathcal{K}(\mathbf{p},\mathbf{r};\lambda) = \mp\Delta\lambda \sum_{i=1}^{\mathcal{N}_a} \tfrac{1}{2}(m_i^B - m_i^A)\tfrac{m_i(\lambda)}{m_i(\lambda\pm\Delta\lambda)}(\mathbf{v}_i)^2 \tag{14.18}$$

We note that the mass of a positionally fixed atom (Sec. 10.3.6) cannot be made $\lambda-$dependent.

14.2.1. Covalent bond forces. The $\lambda-$dependent version of the quartic potential-energy function term describing the covalent bond-stretching interaction (Sec. 5.1) is obtained by making the force constant $k_n^{(b,q)}$ and the ideal bond length b_n^0 linearly dependent on λ,

$$\begin{aligned}
&V^{(b,q)}(\mathbf{r};\lambda) = \sum_{n=1}^{N^{(b)}} {V^{(b,q)}}_n(b_n;\lambda) \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{4}\Big[(1-\lambda)k_n^{(b,q)\,A} + \lambda k_n^{(b,q)\,B}\Big]\Big[\ {b_n}^2 - \big[\ (1-\lambda)b_n^{0\,A} + \lambda b_n^{0\,B}\big]^2\Big]^2 \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{4}k_n^{(b,q)\,A}\big[\ {b_n}^2 - (b_n^{0\,A})^2\big]^2 \\
&\quad + \sum_{n=1}^{N^{(b)}} \tfrac{1}{4}\lambda\Big\{ -2k_n^{(b,q)\,A}\big[\ b_n^{0\,B} - b_n^{0\,A}\big]\big[\ 2b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})\big]\big[({b_n}^2 - (b_n^{0\,A})^2\big] \\
&\quad + \lambda k_n^{(b,q)\,A}\big[\ b_n^{0\,B} - b_n^{0\,A}\big]^2\big[\ 2b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})\big]^2 \\
&\quad + \Big[\ k_n^{(b,q)\,B} - k_n^{(b,q)\,A}\Big]\Big[\ {b_n}^2 - \big[\ b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})\big]^2\Big]^2\Big\} \\
&= V^{(b,q)}(\mathbf{r};\lambda=\lambda_A=0) + \Delta V^{(b,q)\,A}(\mathbf{r};\lambda)
\end{aligned} \tag{14.19}$$

The forces $\mathbf{f}^{(b,q)\,A}_{i}$ and $\mathbf{f}^{(b,q)\,A}_{j}$ in state A ($\lambda = \lambda_A = 0$) on atoms i and j due to the n-th term in Eq. 5.1 are given by expressions Eq. 17.1 and Eq. 17.2, using $k_n^{(b,q)} = k_n^{(b,q)\,A}$ and using $b_n^0 = b_n^{0\,A}$. The forces on atoms i and j due to the n-th term of the energy difference $\Delta V^{(b,q)\,A}(\mathbf{r};\lambda)$ are

$$\begin{aligned}
\mathbf{f}^{(b,q)\,\Delta A}_{i} &= -\frac{\partial \Delta V^{(b,q)\,A}_{n}}{\partial {b_n}^2}\frac{\partial {b_n}^2}{\partial \mathbf{r}_i} \\
&= \lambda\Big\{\, k_n^{(b,q)\,A}\,\big[\, b_n^{0\,B} - b_n^{0\,A}\big]\,\big[\, 2b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})\big] \\
&\quad - \Big[\, k_n^{(b,q)\,B} - k_n^{(b,q)\,A}\Big]\,\big[\, {b_n}^2 - [b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})]^2\big]\,\Big\}\mathbf{r}_{ij}
\end{aligned} \tag{14.20}$$

and

$$\mathbf{f}^{(b,q)\,\Delta A}_{j} = -\mathbf{f}^{(b,q)\,\Delta A}_{i}. \tag{14.21}$$

The derivative with respect to λ of the energy $V^{(b,q)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(b,q)\,A}(\mathbf{r};\lambda)$ with respect to state A is

$$\begin{aligned}
\frac{\partial V^{(b,q)}(\mathbf{r};\lambda)}{\partial \lambda} &= \frac{\partial \Delta V^{(b,q)\,A}(\mathbf{r};\lambda)}{\partial \lambda} \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{4}\Big\{ -4\Big[\, k_n^{(b,q)\,A} + \lambda(k_n^{(b,q)\,B} - k_n^{(b,q)\,A})\Big]\,\big[\, b_n^{0\,B} - b_n^{0\,A}\big] \\
&\quad \cdot \big[\, b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})\big]\,\big[\, {b_n}^2 - [b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})]^2\big] \\
&\quad + \Big[\, k_n^{(b,q)\,B} - k_n^{(b,q)\,A}\Big]\,\big[\, {b_n}^2 - [b_n^{0\,A} + \lambda(b_n^{0\,B} - b_n^{0\,A})]^2\big]^2\Big\}.
\end{aligned} \tag{14.22}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is

$$V^{(b,q)}(\mathbf{r};\lambda \pm \Delta\lambda) - V^{(b,q)}(\mathbf{r};\lambda) \tag{14.23}$$

which can easily be obtained from Eq. 14.19.

The $\lambda-$dependent version of the harmonic potential-energy function term describing the covalent bond-stretching (Sec. 5.1) is obtained by making the force constant $k_n^{(b,h)}$ and the ideal bond length b_n^0 linearly dependent on λ,

$$\begin{aligned}
V^{(b,h)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(b)}} V^{(b,h)}(b_n;\lambda) \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{2}\Big[(1-\lambda)k_n^{(b,h)\,A} + \lambda k_n^{(b,h)\,B}\Big]\,\big[b_n - [(1-\lambda)b_n^{0\,A} + \lambda b_n^{0\,B}]\big]^2 \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{2}k_n^{(b,h)\,A}[b_n - b_n^{0\,A}]^2 \\
&\quad + \sum_{n=1}^{N^{(b)}} \tfrac{1}{2}\lambda\Big\{ -k_n^{(b,h)\,A}\,\big[\, b_n^{0\,B} - b_n^{0\,A}\big]\,\big[\, 2[b_n - b_n^{0\,A}] - \lambda[b_n^{0\,B} - b_n^{0\,A}]\big] \\
&\quad + \Big[\, k_n^{(b,h)\,B} - k_n^{(b,h)\,A}\Big]\,\big[\, b_n - b_n^{0\,A} - \lambda b_n^{0\,B} - b_n^{0\,A})\big]^2\Big\} \\
&= V^{(b,h)}(\mathbf{r};\lambda = \lambda_A = 0) + \Delta V^{(b,h)\,A}(\mathbf{r};\lambda).
\end{aligned} \tag{14.24}$$

The forces $\mathbf{f}^{(b,h)\,A}_{i}$ and $\mathbf{f}^{(b,h)\,A}_{j}$ in state A ($\lambda = \lambda_A = 0$) on atoms i and j due to the n-th term in Eq. 5.6 are given by expressions Eq. 17.3 and Eq. 17.4, using $k_n^{(b,h)} = k_n^{(b,h)\,A}$ and $b_n^0 = b_n^{0\,A}$.

The forces on atoms i and j due to the n-th term in the energy difference $\Delta V^{(b,h)\,A}(\mathbf{r};\lambda)$ are

$$\begin{aligned}
\mathbf{f}^{(b,h)\,\Delta A}_{i} &= -\frac{\partial \Delta V^{(b,h)\,A}_{n}}{\partial b_n}\frac{\partial b_n}{\partial \mathbf{r}_i} \\
&= \lambda \Big\{ \; k_n^{(b,h)\,A}[b_n^{0\,B} - b_n^{0\,A}] \\
&\quad - \Big[k_n^{(b,h)\,B} - k_n^{(b,h)\,A}\Big] \; \big[\; b_n - b_n^{0\,A} - \lambda(b_n^{0\,B} - b_n^{0\,A})\big] \Big\} \; \frac{\partial b_n}{\partial \mathbf{r}_i}
\end{aligned} \tag{14.25}$$

and

$$\mathbf{f}^{(b,h)\,\Delta A}_{j} = -\mathbf{f}^{(b,h)\,\Delta A}_{i} \tag{14.26}$$

The derivative with respect to λ of the energy $V^{(b,h)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(b,h)\,A}$ with respect to the state A is:

$$\begin{aligned}
&\tfrac{\partial V^{(b,h)}(\mathbf{r};\lambda)}{\partial \lambda} = \tfrac{\partial \Delta V^{(b,h)\,A}(\mathbf{r};\lambda)}{\partial \lambda} \\
&= \sum_{n=1}^{N^{(b)}} \tfrac{1}{2} \Big\{ \; -2 \Big[\; k_n^{(b,h)\,A} + \lambda(k_n^{(b,h)\,B} - k_n^{(b,h)\,A}) \Big] \; \big[\; b_n^{0\,B} - b_n^{0\,A}\big] \\
&\quad \big[\; b_n - b_n^{0\,A} - \lambda(b_n^{0\,B} - b_n^{0\,A})\big] \\
&\quad + \Big[\; k_n^{(b,h)\,B} - k_n^{(b,h)\,A}\Big] \; \big[\; b_n - b_n^{0\,A} - \lambda(b_n^{0\,B} - b_n^{0\,A})\big]^2 \Big\} .
\end{aligned} \tag{14.27}$$

The expression for use in the perturbation formula Eq. 14.9 is

$$V^{(b,h)}(\mathbf{r};\lambda \pm \Delta\lambda) - V^{(b,h)}(\mathbf{r};\lambda) \tag{14.28}$$

which can easily be obtained from Eq. 14.24.

14.2.2. Covalent bond forces (soft potential energy function). When the perturbation involves the breaking of a bond numerical instabilities occur in $\frac{\partial V^{(b,h)}(\mathbf{r};\lambda)}{\partial \lambda}$ because the distance b between the formerly bonded atoms becomes large in the state where the bond stretching force constant is 0. This can be avoided by using a modified, "soft" bond stretch potential energy function, introducing a softness term $S(b,\lambda)$. Wang et al[134] formulate the contribution to the potential energy due to a soft harmonic bond b with force constant $k^{(b,h)}$ and target value b^0 that is broken at $\lambda = 0$ and has its full strength at $\lambda = 1$ as follows:

$$V(b,\lambda) = \frac{1}{2} K \lambda (b - b^0)^2 \frac{1}{S(b,(1-\lambda))} \tag{14.29}$$

where

$$S(b,\lambda) = 1 + \alpha_b \lambda (b - b^0)^2$$

and α_b is a positive number that can be adjusted to maximize the phase space overlap between neighbouring λ windows.[134]

In GROMOS a more general form of the soft harmonic bond stretch potential energy function is implemented, which allows the bond in either of the two states to be broken and the bond lengths in the two states to be different. In analogy to Eq. 14.24:

$$\begin{aligned} V^{(bs,h)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(bs)}} V^{(bs,h)}(b_n;\lambda) \\ &= \sum_{n=1}^{N^{(bs)}} \tfrac{1}{2}\left[(1-\lambda)\tfrac{k_n^{(b,h)\,A}}{S_n^A(b_n,\lambda)} + \lambda\tfrac{k_n^{(b,h)\,B}}{S_n^B(b_n,1-\lambda)}\right]\left[b_n - b_n^0(\lambda)\right]^2 \end{aligned} \tag{14.30}$$

where $N^{(bs)}$ is the number of soft bonds and $S_n^X(b_n,\lambda)$ the softness term for state X

$$S_n^X(b_n,\lambda) \;=\; 1+\alpha_b\lambda(b_n - b_n^{0\,X})^2 \tag{14.31}$$

and

$$b_n^0(\lambda) \;=\; (1-\lambda)b_n^{0\,A} + \lambda b_n^{0\,B}$$

The forces on atoms i and j due to the n-th term in the energy $V^{(bs,h)}(\mathbf{r};\lambda)$ are

$$\begin{aligned} \mathbf{f}^{(bs,h)}{}_i &= -\frac{\partial V^{(bs,h)}{}_n}{\partial b_n}\frac{\partial b_n}{\partial \mathbf{r}_i} \\ &= -\left[\frac{(1-\lambda)k_n^{(b,h)\,A}}{S_n^A(b_n,\lambda)^2} + \frac{\lambda k_n^{(b,h)\,B}}{S_n^B(b_n,1-\lambda)^2}\right]\left[b_n - b_n^0(\lambda)\right]\frac{\partial b_n}{\partial \mathbf{r}_i} \\ &= -\left[\frac{(1-\lambda)k_n^{(b,h)\,A}}{S_n^A(b_n,\lambda)^2} + \frac{\lambda k_n^{(b,h)\,B}}{S_n^B(b_n,1-\lambda)^2}\right]\left[b_n - b_n^0(\lambda)\right]\frac{\mathbf{r}_{ij}}{r_{ij}} \end{aligned} \tag{14.32}$$

and

$$\mathbf{f}^{(bs,h)}{}_j \;=\; -\mathbf{f}^{(bs,h)}{}_i \tag{14.33}$$

The derivative with respect to λ of the energy $V^{(bs,h)}(\mathbf{r};\lambda)$ is:

$$\begin{aligned} &\frac{\partial V^{(bs,h)}(\mathbf{r};\lambda)}{\partial\lambda} = \\ &= \sum_{n=1}^{N^{(bs)}} \frac{1}{2}(b_n - b_n^0(\lambda))^2\left[\frac{k_n^{(b,h)\,A}}{S_n^A(b_n,\lambda)^2}(-X1+X2) + \frac{k_n^{(b,h)\,B}}{S_n^B(b_n,1-\lambda)^2}(X1+X2)\right] \\ &-(b_n - b_n^0(\lambda))(b_n^{0\,B} - b_n^{0\,A})\left[\frac{(1-\lambda)k_n^{(b,h)\,A}}{S_n^A(b_n,\lambda)} + \frac{\lambda k_n^{(b,h)\,B}}{S_n^B(b_n,1-\lambda)}\right] \end{aligned} \tag{14.34}$$

with

$$\begin{aligned} X1 &= 1+\alpha(b_n - b_n^0(\lambda))^2 \\ X2 &= 2\alpha\lambda(1-\lambda)(b_n - b_n{}^0(\lambda))(b_n^{0\,B} - b_n^{0\,A}) \end{aligned} \tag{14.35}$$

14.2.3. Covalent bond-angle forces. The $\lambda-$dependent version of the potential-energy function describing the covalent bond-angle bending interaction, which is harmonic in the cosine of the bond angles (Sec. 5.2 , Eq. 5.7) is obtained by making the force constant $k_n^{(\theta,c)}$ and the cosine of the ideal bond angle θ_n linearly dependent on λ,

$$
\begin{aligned}
V^{(\theta,c)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(\theta)}} {V^{(\theta,c)}}_n(\theta_n;\lambda) \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\Big[(1-\lambda)k_n^{(\theta,c)A} + \lambda k_n^{(\theta,c)B}\Big]\big[\cos\theta_n - [(1-\lambda)\cos\theta_n^{0\,A} + \lambda\cos\theta_n^{0\,B}]\big]^2 \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}k_n^{(\theta,c)A}\big[\,\cos\theta_n - \cos\theta_n^{0\,A}\big]^2 \\
&+ \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\lambda\Big\{\,-k_n^{(\theta,c)A}[\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}]\,\big[\,2[\cos\theta_n - \cos\theta_n^{0\,A}] - \lambda[\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}]\big] \\
&+[k_n^{(\theta,c)B} - k_n^{(\theta,c)A}]\big[\,\cos\theta_n - \cos\theta_n^{0\,A} - \lambda[\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}]\big]^2\Big\} \\
&= V^{(\theta,c)}(\mathbf{r};\lambda=\lambda_A=0) + \Delta V^{(\theta,c)A}(\mathbf{r};\lambda).
\end{aligned}
\tag{14.36}
$$

The forces $\mathbf{f}_{\,i}^{(\theta,c)A}$, $\mathbf{f}_{\,j}^{(\theta,c)A}$ and $\mathbf{f}_{\,k}^{(\theta,c)A}$ in state A ($\lambda=\lambda_A=0$) on atoms i, j and k due to the n-th term in Eq. 5.7 are given by expressions Eq. 17.5, Eq. 17.6 and Eq. 17.7 using $k_n^{(\theta,c)} = k_n^{(\theta,c)A}$ and $\theta_n^0 = \theta_n^{0\,A}$.

The forces on atoms i, j and k due to the n-th term in the energy difference $\Delta V^{(\theta,c)A}(\boldsymbol{r};\lambda)$ are

$$
\begin{aligned}
\mathbf{f}_{\,i}^{(\theta,c)\Delta A} &= -\frac{\partial \Delta V^{(\theta,c)A}_{\;\;n}}{\partial\cos\theta_n}\frac{\partial\cos\theta_n}{\partial\mathbf{r}_i} \\
&= \lambda\Big\{\,k_n^{(\theta,c)A}[\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}] \\
&\quad -\Big[\,k_n^{(\theta,c)B} - k_n^{(\theta,c)A}\Big]\,\big[\,\cos\theta_n - \cos\theta_n^{0\,A} - \lambda[\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}]\big]\Big\}\frac{\partial\cos\theta_n}{\partial\mathbf{r}_i}
\end{aligned}
\tag{14.37}
$$

$$
\mathbf{f}_{\,k}^{(\theta,c)\Delta A} = -\frac{\partial \Delta V^{(\theta,c)A}_{\;\;n}}{\partial\cos\theta_n}\frac{\partial\cos\theta_n}{\partial\mathbf{r}_k} \tag{14.38}
$$

and

$$
\mathbf{f}_{\,j}^{(\theta,c)\Delta A} = -\mathbf{f}_{\,i}^{(\theta,c)\Delta A} - \mathbf{f}_{\,k}^{(\theta,c)\Delta A}. \tag{14.39}
$$

The derivative with respect to λ of the energy $V^{(\theta,c)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(\theta,c)A}(\mathbf{r};\lambda)$ with respect to state A is

$$
\begin{aligned}
\frac{\partial V^{(\theta,c)}(\mathbf{r};\lambda)}{\partial\lambda} &= \frac{\partial \Delta V^{(\theta,c)A}(\mathbf{r};\lambda)}{\partial\lambda} \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\Big\{\,-2\Big[k_n^{(\theta,c)A} + \lambda(k_n^{(\theta,c)B} - k_n^{(\theta,c)A})\Big]\,\big[\,\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A}\big] \\
&\quad \big[\,\cos\theta_n - \cos\theta_n^{0\,A} - \lambda(\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A})\big] \\
&\quad +\Big[\,k_n^{(\theta,c)B} - k_n^{(\theta,c)A}\Big]\,\big[\,\cos\theta_n - \cos\theta_n^{0\,A} - \lambda(\cos\theta_n^{0\,B} - \cos\theta_n^{0\,A})\big]^2\Big\}.
\end{aligned}
\tag{14.40}
$$

The corresponding expressions for use in the perturbation formula Eq. 14.9 are

$$V^{(\theta,c)}(\mathbf{r};\lambda \pm \Delta\lambda) - V^{(\theta,c)}(\mathbf{r};\lambda) \tag{14.41}$$

which can easily be obtained from Eq. 14.36.

The $\lambda-$dependent version of the harmonic potential-energy function describing the covalent bond-angle bending interaction (Eq. 5.11) is obtained by making the force constant $k_n^{(\theta,h)}$ and the ideal bond angle θ_n^0 linearly dependent on λ,

$$\begin{aligned}
V^{(\theta,h)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(\theta)}} {V^{(\theta,h)}}_n(\theta_n;\lambda) \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\Big[(1-\lambda)k_n^{(\theta,h)A} + \lambda k_n^{(\theta,h)B}\Big]\big[\theta_n - [(1-\lambda)\theta_n^{0\,A} + \lambda\theta_n^{0\,B}]\big]^2 \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}k_n^{(\theta,h)A}\big[\theta_n - \theta_n^{0\,A}\big]^2 \\
&\quad + \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\lambda\Big\{ -k_n^{(\theta,h)A}[\theta_n^{0\,B} - \theta_n^{0\,A}]\,\big[\,2[\theta_n - \theta_n^{0\,A}] - \lambda[\theta_n^{0\,B} - \theta_n^{0\,A}]\big] \\
&\quad + [k_n^{(\theta,h)B} - k_n^{(\theta,h)A}]\big[\,\theta_n - \theta_n^{0\,A} - \lambda[\theta_n^{0\,B} - \theta_n^{0\,A}]\big]^2\Big\} \\
&= V^{(\theta,h)}(\mathbf{r};\lambda = \lambda_A = 0) + \Delta V^{(\theta,h)A}(\mathbf{r};\lambda).
\end{aligned} \tag{14.42}$$

The forces $\mathbf{f}_i^{(\theta,h)A}$, $\mathbf{f}_j^{(\theta,h)A}$ and $\mathbf{f}_k^{(\theta,h)A}$ in state A ($\lambda = \lambda_A = 0$) on atoms i, j and k due to Eq. 5.11 are given by expressions Eq. 17.8, Eq. 17.9 and Eq. 17.10 using $k_n^{(\theta,h)} = k_n^{(\theta,h)A}$ and $\theta_n^0 = \theta_n^{0\,A}$.

The forces on atoms i, j and k due to the n-th term in the energy difference $\Delta V^{(\theta,h)A}(\mathbf{r};\lambda)$ are

$$\begin{aligned}
\mathbf{f}_i^{(\theta,h)\Delta A} &= -\frac{\partial \Delta {V^{(\theta,h)A}}_n}{\partial\theta_n}\frac{\partial\theta_n}{\partial\mathbf{r}_i} \\
&= \lambda\Big\{ k_n^{(\theta,h)A}[\theta_n^{0\,B} - \theta_n^{0\,A}] \\
&\quad - \Big[\,k_n^{(\theta,h)B} - k_n^{(\theta,h)A}\Big]\,\big[\,\theta_n - \theta_n^{0\,A} - \lambda(\theta_n^{0\,B} - \theta_n^{0\,A})\big]\Big\}\frac{\partial\theta_n}{\partial\mathbf{r}_i}
\end{aligned} \tag{14.43}$$

$$\mathbf{f}_k^{(\theta,h)\Delta A} = -\frac{\partial \Delta {V^{(\theta,h)A}}_n}{\partial\theta_n}\frac{\partial\theta_n}{\partial\mathbf{r}_k} \tag{14.44}$$

and

$$\mathbf{f}_j^{(\theta,h)\Delta A} = -\mathbf{f}_i^{(\theta,h)\Delta A} - \mathbf{f}_k^{(\theta,h)\Delta A}. \tag{14.45}$$

The derivative with respect to λ of the energy $V^{(\theta,h)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(\theta,h)A}(\mathbf{r};\lambda)$ with respect to state A is

$$\begin{aligned}
\frac{\partial V^{(\theta,h)}(\mathbf{r};\lambda)}{\partial\lambda} &= \frac{\partial \Delta V^{(\theta,h)A}(\mathbf{r};\lambda)}{\partial\lambda} \\
&= \sum_{n=1}^{N^{(\theta)}} \tfrac{1}{2}\Big\{ -2\Big[\,k_n^{(\theta,h)A} + \lambda(k_n^{(\theta,h)B} - k_n^{(\theta,h)A})\Big]\,\big[\,\theta_n^{0\,B} - \theta_n^{0\,A}\big] \\
&\quad \big[\,\theta_n - \theta_n^{0\,A} - \lambda(\theta_n^{0\,B} - \theta_n^{0\,A})\big] \\
&\quad + \Big[\,k_n^{(\theta,h)B} - k_n^{(\theta,h)A}\Big]\,\big[\,\theta_n - \theta_n^{0\,A} - \lambda(\theta_n^{0\,B} - \theta_n^{0\,A})\big]^2\Big\}.
\end{aligned} \tag{14.46}$$

The expression for use in the perturbation formula Eq. 14.9 is

$$V^{(\theta,h)}(\mathbf{r};\lambda \pm \Delta\lambda) - V^{(\theta,h)}(\mathbf{r};\lambda) \tag{14.47}$$

which can easily be obtained from Eq. 14.42.

14.2.4. Covalent bond-angle forces (soft potential energy function). When a bond is broken also the force constants of affected bond angles should go to zero and numerical instabilities are - even though weaker than in the bond-stretching term (see Sec. 14.2.2) - also apparent in the λ derivatives of the bond-angle bending term $\frac{\partial V^{(\theta,c)}(\mathbf{r};\lambda)}{\partial\lambda}$. A soft potential energy function $V^{(\theta s,c)}(\mathbf{r};\lambda)$ and the corresponding λ derivative $\frac{\partial V^{(\theta s,c)}(\mathbf{r};\lambda)}{\partial\lambda}$ can be used analogous to Eq. 14.30 and Eq. 14.34, simply substituting b with $\cos\theta$.

The forces on atoms i, j, and k of the n-th soft angle are then:

$$\begin{aligned}
\mathbf{f}^{(\theta s,c)}{}_i &= -\frac{\partial V_n^{(\theta s,c)}}{\partial\cos\theta_n}\frac{\partial\cos\theta_n}{\partial\mathbf{r}_i} \qquad (14.48)\\
&= -\left[\frac{(1-\lambda)k_n^{(\theta,c)A}}{S_n^A(\cos\theta_n,\lambda)^2} + \frac{\lambda k_n^{(\theta,c)B}}{S_n^B(\cos\theta_n,1-\lambda)^2}\right]\left[\cos\theta_n - \cos\theta_n^0(\lambda)\right]\frac{\partial\cos\theta_n}{\partial\mathbf{r}_i}\\
&= -\left[\frac{(1-\lambda)k_n^{(\theta,c)A}}{S_n^A(\cos\theta_n,\lambda)^2} + \frac{\lambda k_n^{(\theta,c)B}}{S_n^B(\cos\theta_n,1-\lambda)^2}\right]\left[\cos\theta_n - \cos\theta_n^0(\lambda)\right]\left[\frac{\mathbf{r}_{kj}}{r_{kj}} - \frac{\mathbf{r}_{ij}}{r_{ij}}\cos\theta_n\right]\frac{1}{r_{ij}}\\
\mathbf{f}^{(\theta s,c)}{}_k &= -\frac{\partial V_n^{(\theta s,c)}}{\partial\cos\theta_n}\frac{\partial\cos\theta_n}{\partial\mathbf{r}_k}\\
&= -\left[\frac{(1-\lambda)k_n^{(\theta,c)A}}{S_n^A(\cos\theta_n,\lambda)^2} + \frac{\lambda k_n^{(\theta,c)B}}{S_n^B(\cos\theta_n,1-\lambda)^2}\right]\left[\cos\theta_n - \cos\theta_n^0(\lambda)\right]\left[\frac{\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{kj}}{r_{kj}}\cos\theta_n\right]\frac{1}{r_{kj}}\\
\mathbf{f}^{(\theta s,c)}{}_j &= -\mathbf{f}^{(\theta s,c)}{}_i - \mathbf{f}^{(\theta s,c)}{}_k
\end{aligned}$$

with

$$S_n^X(\cos\theta_n,\lambda) = 1 + \alpha_\theta\lambda(\cos\theta_n - \cos\theta_n^{0\,X})^2 \tag{14.49}$$

and

$$\cos\theta_n^0(\lambda) = (1-\lambda)\cos\theta_n^{0\,A} + \lambda\cos\theta_n^{0\,B}. \tag{14.50}$$

14.2.5. Improper dihedral-angle forces. The $\lambda-$dependent version of the harmonic improper dihedral-angle bending interaction (Sec. 5.3) is obtained by making the force constant $k_n^{(\xi)}$ and the ideal improper

dihedral-angle ξ_n^0 linearly dependent on λ,

$$
\begin{aligned}
V^{(\xi)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(\xi)}} {V^{(\xi)}}_n(\xi_n;\lambda) \\
&= \sum_{n=1}^{N^{(\xi)}} \tfrac{1}{2}\left[(1-\lambda)k_n^{(\xi)\,A} + \lambda k_n^{(\xi)\,B}\right]\left[\ \xi_n - [(1-\lambda)\xi_n^{0\,A} + \lambda\xi_n^{0\,B}]\right]^2 \\
&= \sum_{n=1}^{N^{(\xi)}} \tfrac{1}{2}k_n^{(\xi)\,A}[\xi_n - \xi_n^{0\,A}]^2 \\
&\quad + \sum_{n=1}^{N^{(\xi)}} \tfrac{1}{2}\lambda\Big\{\ -k_n^{(\xi)\,A}\left[\ \xi_n^{0\,B} - \xi_n^{0\,A}\right]\left[\ 2[\xi_n - \xi_n^{0\,A}] - \lambda[\xi_n^{0\,B} - \xi_n^{0\,A}]\right] \\
&\quad + \left[\ k_n^{(\xi)\,B} - k_n^{(\xi)\,A}\right]\left[\ \xi_n - \xi_n^{0\,A} - \lambda(\xi_n^{0\,B} - \xi_n^{0\,A})\right]^2\Big\} \\
&= V^{(\xi)}(\mathbf{r};\lambda = \lambda_A = 0) + \Delta V^{(\xi)\,A}(\mathbf{r};\lambda).
\end{aligned}
\tag{14.51}
$$

The forces ${\mathbf{f}^{(\xi)}}_i^A$, ${\mathbf{f}^{(\xi)}}_j^A$, ${\mathbf{f}^{(\xi)}}_k^A$ and ${\mathbf{f}^{(\xi)}}_l^A$ in state A ($\lambda = \lambda_A = 0$) on atoms i, j, k and l due to the n-th term in Eq. 5.13 are given by expressions Eq. 17.11, Eq. 17.12, Eq. 17.13 and Eq. 17.14 using $k_n^{(\xi)} = k_n^{(\xi)\,A}$ and $\xi_n^0 = \xi_n^{0\,A}$.

The forces on atoms i, j, k and l due to the n-th term in the energy difference $\Delta V^{(\xi)\,A}(\mathbf{r};\lambda)$ are

$$
\begin{aligned}
{\mathbf{f}^{(\xi)}}_i^{\Delta A} &= -\frac{\partial \Delta {V^{(\xi)}}_n^A}{\partial \xi_n}\frac{\partial \xi_n}{\partial \mathbf{r}_i} \\
&= \lambda\Big\{\ k_n^{(\xi)\,A}[\xi_n^{0\,B} - \xi_n^{0\,A}] \\
&\quad - \left[\ k_n^{(\xi)\,B} - k_n^{(\xi)\,A}\right]\left[\ \xi_n - \xi_n^{0\,A} - \lambda(\xi_n^{0\,B} - \xi_n^{0\,A})\right]\Big\}\frac{\partial \xi_n}{\partial \mathbf{r}_i}
\end{aligned}
\tag{14.52}
$$

$$
{\mathbf{f}^{(\xi)}}_j^{\Delta A} = -\frac{\partial \Delta {V^{(\xi)}}_n^A}{\partial \xi_n}\frac{\partial \xi_n}{\partial \mathbf{r}_j} \tag{14.53}
$$

$$
{\mathbf{f}^{(\xi)}}_l^{\Delta A} = -\frac{\partial \Delta {V^{(\xi)}}_n^A}{\partial \xi_n}\frac{\partial \xi_n}{\partial \mathbf{r}_l} \tag{14.54}
$$

and

$$
{\mathbf{f}^{(\xi)}}_k^{\Delta A} = -{\mathbf{f}^{(\xi)}}_i^{\Delta A} - {\mathbf{f}^{(\xi)}}_j^{\Delta A} - {\mathbf{f}^{(\xi)}}_l^{\Delta A}. \tag{14.55}
$$

The derivative with respect to λ of the energy $V^{(\xi)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(\xi)}(\mathbf{r};\lambda)$ with respect to state A is

$$\begin{aligned}
&\frac{\partial V^{(\xi)}(\mathbf{r};\lambda)}{\partial\lambda} = \frac{\partial \Delta V^{(\xi)\,A}(\mathbf{r};\lambda)}{\partial\lambda} \\
&= \sum_{n=1}^{N^{(\xi)}} \frac{1}{2}\Big\{ -2\Big[\, k_n^{(\xi)\,A} + \lambda(k_n^{(\xi)\,B} - k_n^{(\xi)\,A})\Big]\,\big[\,\xi_n^{0\,B} - \xi_n^{0\,A}\big] \\
&\quad \big[\,\xi_n - \xi_n^{0\,A} - \lambda(\xi_n^{0\,B} - \xi_n^{0\,A})\big] \\
&\quad + \Big[\, k_n^{(\xi)\,B} - k_n^{(\xi)\,A}\Big]\,\big[\,\xi_n - \xi_n^{0\,A} - \lambda(\xi_n^{0\,B} - \xi_n^{0\,A})\big]^2\Big\}.
\end{aligned} \tag{14.56}$$

The expression for use in the perturbation formula Eq. 14.9 is

$$V^{(\xi)}(\mathbf{r};\lambda \pm \Delta\lambda) - V^{(\xi)}(\mathbf{r};\lambda) \tag{14.57}$$

which can easily be obtained from Eq. 14.51.

14.2.6. Improper dihedral-angle forces (soft potential energy function). A soft potential energy function, analogous to the ones for the bond stretching and bond-angle bending terms can also be applied to the improper dihedrals to reduce numerical instabilities when a bond is broken. A soft potential energy $V^{(\xi s)}(\mathbf{r};\lambda)$ and the corresponding λ derivative $\frac{\partial V^{(\xi s)}(\mathbf{r};\lambda)}{\partial\lambda}$ are defined as in Eq. 14.30 and Eq. 14.34, simply substituting b with ξ.

The forces on atoms i, j, k and l of the n-th soft improper dihedral angle are then:

$$\begin{aligned}
\mathbf{f}^{(\xi s)}{}_i &= -\frac{\partial V^{(\xi s)}{}_n}{\partial\xi_n}\frac{\partial\xi_n}{\partial\mathbf{r}_i} \\
&= -\left[\frac{(1-\lambda)k_n^{(\xi)\,A}}{S_n^A(\xi_n,\lambda)^2} + \frac{\lambda k_n^{(\xi)\,B}}{S_n^B(\xi_n,1-\lambda)^2}\right]\big[\xi_n - \xi_n^0(\lambda)\big]\,\frac{\partial\xi_n}{\partial\mathbf{r}_i} \\
&= -\left[\frac{(1-\lambda)k_n^{(\xi)\,A}}{S_n^A(\xi_n,\lambda)^2} + \frac{\lambda k_n^{(\xi)\,B}}{S_n^B(\xi_n,1-\lambda)^2}\right]\big[\xi_n - \xi_n^0(\lambda)\big]\,\frac{r_{kj}}{r_{mj}^2}\mathbf{r}_{mj} \\
\mathbf{f}^{(\xi s)}{}_l &= -\frac{\partial V^{(\xi s)}{}_n}{\partial\xi_n}\frac{\partial\xi_n}{\partial\mathbf{r}_l} \\
&= -\left[\frac{(1-\lambda)k_n^{(\xi)\,A}}{S_n^A(\xi_n,\lambda)^2} + \frac{\lambda k_n^{(\xi)\,B}}{S_n^B(\xi_n,1-\lambda)^2}\right]\big[\xi_n - \xi_n^0(\lambda)\big]\,\frac{r_{kj}}{r_{nk}^2}\mathbf{r}_{nk} \\
\mathbf{f}^{(\xi s)}{}_j &= -\frac{\partial V^{(\xi s)}{}_n}{\partial\xi_n}\frac{\partial\xi_n}{\partial\mathbf{r}_j} \\
&= \left[\frac{\mathbf{r}_{ij}\cdot\mathbf{r}_{kj}}{r_{kj}^2} - 1\right]\mathbf{f}^{(\xi s)}{}_i - \frac{\mathbf{r}_{kl}\cdot\mathbf{r}_{kj}}{r_{kj}^2}\mathbf{f}^{(\xi s)}{}_l \\
\mathbf{f}^{(\xi s)}{}_k &= -\mathbf{f}^{(\xi s)}{}_i - \mathbf{f}^{(\xi s)}{}_j - \mathbf{f}^{(\xi s)}{}_l
\end{aligned} \tag{14.58}$$

where

$$S_n^X(\xi_n,\lambda) = 1 + \alpha_\xi\lambda(\xi_n - \cos\xi_n^{0\,X})^2, \tag{14.59}$$

$$\xi_n^0(\lambda) = (1-\lambda)\xi_n^{0\,A} + \lambda\xi_n^{0\,B} \tag{14.60}$$

and

$$\begin{aligned}
\mathbf{r}_{mj} &= \mathbf{r}_{ij} \times \mathbf{r}_{kj} \\
\mathbf{r}_{nk} &= \mathbf{r}_{kj} \times \mathbf{r}_{kl} \\
r_{mj} &= (\mathbf{r}_{mj} \cdot \mathbf{r}_{mj})^{1/2} \\
r_{nk} &= (\mathbf{r}_{nk} \cdot \mathbf{r}_{nk})^{1/2}.
\end{aligned}$$

14.2.7. Dihedral-angle torsion forces. The $\lambda-$dependent version of the trigonometric dihedral-angle torsion interaction (Sec. 5.4) is obtained by applying a linear combination of the interaction function at states A and B,

$$\begin{aligned}
V^{(\varphi,s)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(\varphi)}} V^{(\varphi,s)}{}_n(\varphi_n;\lambda) \\
&= \sum_{n=1}^{N^{(\varphi)}} (1-\lambda) k_n^{(\varphi,s)\,A}[1+\cos(\varphi_n^{0\,A})\cos(m_n^{(\varphi)\,A}\varphi_n)] + \lambda k_n^{(\varphi,s)\,B}[1+\cos(\varphi_n^{0\,B})\cos(m_n^{(\varphi)\,B}\varphi_n)] \\
&= \sum_{n=1}^{N^{(\varphi)}} k_n^{(\varphi,s)\,A}[1+\cos(\varphi_n^{0\,A})\cos(m_n^{(\varphi)\,A}\varphi_n)] + \sum_{n=1}^{N^{(\varphi)}} \lambda \Big\{ k_n^{(\varphi,s)\,B}[1+\cos(\varphi_n^{0\,B})\cos(m_n^{(\varphi)\,B}\varphi_n)] \\
&\quad - k_n^{(\varphi,s)\,A}[1+\cos(\varphi_n^{0\,A})\cos(m_n^{(\varphi)\,A}\varphi_n)] \Big\} \\
&= V^{(\varphi,s)}(\mathbf{r};\lambda=\lambda_A=0) + \Delta V^{(\varphi,s)\,A}(\mathbf{r};\lambda).
\end{aligned} \tag{14.61}$$

The forces $\mathbf{f}^{(\varphi,s)\,A}_{\;\;i}$, $\mathbf{f}^{(\varphi,s)\,A}_{\;\;j}$, $\mathbf{f}^{(\varphi,s)\,A}_{\;\;k}$ and $\mathbf{f}^{(\varphi,s)\,A}_{\;\;l}$ in state A ($\lambda = \lambda_A = 0$) on atoms i, j, k and l due to the n-th term in Eq. 5.18 are given by expressions Eq. 17.16, Eq. 17.18, Eq. 17.19 and Eq. 17.20 using $k_n^{(\varphi,s)} = k_n^{(\varphi,s)\,A}$, $\varphi_n^0 = \varphi_n^{0\,A}$ and $m_n^{(\varphi)} = m_n^{(\varphi)\,A}$.

The forces on atoms i, j, k and l due to the n-th term in the energy difference $\Delta V^{(\varphi,s)\,A}(\mathbf{r};\lambda)$ are

$$\begin{aligned}
\mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;i} &= -\frac{\partial \Delta V^{(\varphi,s)\,A}_{\;\;\;\;n}}{\partial \cos\varphi_n} \frac{\partial \cos\varphi_n}{\partial \mathbf{r}_i} \\
&= -\lambda \left\{ k_n^{(\varphi,s)\,B} \cos(\varphi_n^{0\,B}) \frac{\partial \cos(m_n^{(\varphi)\,B}\varphi_n)}{\partial \cos\varphi_n} \right. \\
&\qquad \left. - k_n^{(\varphi,s)\,A} \cos(\varphi_n^{0\,A}) \frac{\partial \cos(m_n^{(\varphi)\,A}\varphi_n)}{\partial \cos\varphi_n} \right\} \frac{\partial \cos\varphi_n}{\partial \mathbf{r}_i}
\end{aligned} \tag{14.62}$$

$$\mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;j} = -\frac{\partial \Delta V^{(\varphi,s)\,A}_{\;\;\;\;n}}{\partial \cos\varphi_n} \frac{\partial \cos\varphi_n}{\partial \mathbf{r}_j} \tag{14.63}$$

$$\mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;l} = -\frac{\partial \Delta V^{(\varphi,s)\,A}_{\;\;\;\;n}}{\partial \cos\varphi_n} \frac{\partial \cos\varphi_n}{\partial \mathbf{r}_l} \tag{14.64}$$

and

$$\mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;k} = -\mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;i} - \mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;j} - \mathbf{f}^{(\varphi,s)\,\Delta A}_{\;\;\;\;l}. \tag{14.65}$$

The derivative with respect to λ of the energy $V^{(\varphi,s)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(\varphi,s)\,A}(\mathbf{r};\lambda)$ with respect to state A is

$$\begin{aligned}
\frac{\partial V^{(\varphi,s)}(\mathbf{r};\lambda)}{\partial\lambda} &= \frac{\partial \Delta V^{(\varphi,s)\,A}(\mathbf{r};\lambda)}{\partial\lambda} \\
&= \sum_{n=1}^{N^{(\varphi)}} \Big\{ k_n^{(\varphi,s)\,B} \Big[1+\cos(\varphi_n^{0\,B})\cos(m_n^{(\varphi)\,B}\varphi_n)\Big] \\
&\qquad -k_n^{(\varphi,s)\,A}\Big[1+\cos(\varphi_n^{0\,A})\cos(m_n^{(\varphi)\,A}\varphi_n)\Big]\Big\}.
\end{aligned} \tag{14.66}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is

$$V^{(\varphi,s)}(\mathbf{r};\lambda\pm\Delta\lambda) - V^{(\varphi,s)}(\mathbf{r};\lambda) \tag{14.67}$$

which can easily be obtained from Eq. 14.61.

The generalized $\lambda-$dependent version of the trigonometric dihedral-angle torsion interaction (Sec. 5.4) is obtained in a similar way,

$$\begin{aligned}
V^{(\varphi,g)}(\mathbf{r};\lambda) &= \sum_{n=1}^{N^{(\varphi)}} V^{(\varphi,g)}{}_n(\varphi_n;\lambda) \\
&= \sum_{n=1}^{N^{(\varphi)}} (1-\lambda)k_n^{(\varphi,g)\,A}[1+\cos(m_n^{(\varphi)\,A}\varphi_n-\varphi_n^{0\,A})] + \lambda k_n^{(\varphi,g)\,B}[1+\cos(m_n^{(\varphi)\,B}\varphi_n-\varphi_n^{0\,B})] \\
&= \sum_{n=1}^{N^{(\varphi)}} k_n^{(\varphi,g)\,A}[1+\cos(m_n^{(\varphi)\,A}\varphi_n-\varphi_n^{0\,A})] + \sum_{n=1}^{N^{(\varphi)}} \lambda\Big\{ k_n^{(\varphi,g)\,B}[1+\cos(m_n^{(\varphi)\,B}\varphi_n-\varphi_n^{0\,B})] \\
&\quad -k_n^{(\varphi,g)\,A}[1+\cos(m_n^{(\varphi)\,A}\varphi_n-\varphi_n^{0\,B})]\Big\} \\
&= V^{(\varphi,g)}(\mathbf{r};\lambda=\lambda_A=0) + \Delta V^{(\varphi,g)\,A}(\mathbf{r};\lambda).
\end{aligned} \tag{14.68}$$

The forces $\mathbf{f}^{(\varphi,g)\,A}_{i}$, $\mathbf{f}^{(\varphi,g)\,A}_{j}$, $\mathbf{f}^{(\varphi,g)\,A}_{k}$ and $\mathbf{f}^{(\varphi,g)\,A}_{l}$ in state A ($\lambda=\lambda_A=0$) on atoms i, j, k and l due to the n-th term in Eq. 5.18 are given by expressions Eq. 17.21, Eq. 17.23, Eq. 17.25 and Eq. 17.26 using $k_n^{(\varphi,g)}=k_n^{(\varphi,g)\,A}$, $\varphi_n^0=\varphi_n^{0\,A}$ and $m_n^{(\varphi)}=m_n^{(\varphi)\,A}$.

The forces on atoms i, j, k and l due to the n-th term in the energy difference $\Delta V^{(\varphi,g)\,A}(\mathbf{r};\lambda)$ are

$$\begin{aligned}
\mathbf{f}^{(\varphi,g)\,\Delta A}_{i} &= -\frac{\partial \Delta V^{(\varphi,g)\,A}_{n}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_i} \\
&= -\lambda\Big\{ k_n^{(\varphi,g)\,B} m_n^{(\varphi)\,B} sin(m_n^{(\varphi)\,B}\varphi_n-\varphi_n^{0\,B}) \\
&\qquad -k_n^{(\varphi,g)\,A} m_n^{(\varphi)\,A} sin(m_n^{(\varphi)\,A}\varphi_n-\varphi_n^{0\,A})\frac{\partial\varphi_n}{\partial\mathbf{r}_i}
\end{aligned} \tag{14.69}$$

$$\mathbf{f}^{(\varphi,g)\,\Delta A}_{j} = -\frac{\partial \Delta V^{(\varphi,g)\,A}_{n}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_j} \tag{14.70}$$

$$\mathbf{f}^{(\varphi,g)\,\Delta A}_{l} = -\frac{\partial \Delta V^{(\varphi,g)\,A}_{n}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_l} \tag{14.71}$$

and

$$\mathbf{f}^{(\varphi,g)\,\Delta A}_{k} = -\mathbf{f}^{(\varphi,g)\,\Delta A}_{i} - \mathbf{f}^{(\varphi,g)\,\Delta A}_{j} - \mathbf{f}^{(\varphi,g)\,\Delta A}_{l}. \tag{14.72}$$

The derivative with respect to λ of the energy $V^{(\varphi,g)}(\mathbf{r};\lambda)$ and of the energy difference $\Delta V^{(\varphi,g)A}(\mathbf{r};\lambda)$ with respect to state A is

$$\begin{aligned}\frac{\partial V^{(\varphi,g)}(\mathbf{r};\lambda)}{\partial\lambda} &= \frac{\partial \Delta V^{(\varphi,g)A}(\mathbf{r};\lambda)}{\partial\lambda} \\ &= \sum_{n=1}^{N^{(\varphi)}} \Big\{ k_n^{(\varphi,g)B}\Big[\,1+\cos(m_n^{(\varphi)B}\varphi_n-\varphi_n^{0\,B})\Big] \\ &\qquad -k_n^{(\varphi,g)A}\Big[\,1+\cos(m_n^{(\varphi)A}\varphi_n-\varphi_n^{0\,A})\Big]\Big\}. \end{aligned} \tag{14.73}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is

$$V^{(\varphi,g)}(\mathbf{r};\lambda\pm\Delta\lambda)-V^{(\varphi,g)}(\mathbf{r};\lambda) \tag{14.74}$$

which can easily be obtained from Eq. 14.68.

14.2.8. Non-bonded forces. The $\lambda-$dependent version of the non-bonded (van der Waals and electrostatic) interaction (Sec. 3.3, Eqs. 6.6 and 6.7) to be a non-linear function of λ. Both the soft-core radius and the strength of the interaction depend non-linearly on λ in order to allow for a smooth change of real atoms to dummy atoms and vice versa[26]. The $\lambda-$dependent non-bonded interaction is only implemented for reaction field electrostatics and reads

$$\begin{aligned}&\mathcal{V}^{(nbd)}(\mathbf{r};\lambda)= \\ &\sum_{\substack{nonbonded\\ perturbed\\ pairs(i,j)}} \Big\{ \Big[\,\lambda^n\mathcal{V}^{(nbd)}(\mathbf{r}_{ij};B;(1-\lambda)) \;+(1-\lambda)^n\mathcal{V}^{(nbd)}(\mathbf{r}_{ij};A;\lambda)\Big]\;\Big\}\end{aligned} \tag{14.75}$$

with $n=$ integer >0, and for reaction field electrostatics,

$$\begin{aligned}&\mathcal{V}^{(nbd)}(\mathbf{r}_{ij};X;\lambda)=\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};X;\lambda) \\ &\quad +\mathcal{V}^{(ele)}(\mathbf{r}_{ij};q_i^X,q_j^X,X;\lambda)\end{aligned} \tag{14.76}$$

$$\begin{aligned}&=\frac{1}{[\alpha_{LJ}(i,j)(\lambda^2)C_{126}^X(i,j)+(r_{ij})^6]}\cdot\Big[\,\frac{{C_{12}}^X(i,j)}{[\alpha_{LJ}(i,j)(\lambda^2)C_{126}^X(i,j)+(r_{ij})^6]}-{C_6}^X(i,j)\Big] \\ &\quad +\frac{q_i^Xq_j^X}{4\pi\varepsilon_0\varepsilon_1}\left[\frac{1}{[\alpha_c(i,j)(\lambda^2)+(r_{ij})^2]^{\frac{1}{2}}}-\frac{\frac{1}{2}C_{rf}(r_{ij})^2}{[\alpha_c(i,j)(\lambda^2)+{R_{rf}}^2]^{\frac{3}{2}}}-\frac{(1-\frac{1}{2}C_{rf})}{R_{rf}}\right]\end{aligned} \tag{14.77}$$

where the interaction parameters that are different in states A and B are indicated by the superscript X ($=A$ or B):

$$\begin{aligned} q_i^X &= \text{partial charge of atom i in state } X(A \text{ or } B) \\ {C_{12}}^X(i,j) &= r^{12} \text{ vdW parameter for atom pair } (i,j) \text{ in state } X(A \text{ or } B) \\ {C_6}^X(i,j) &= r^6 \text{ vdW parameter for atom pair } (i,j) \text{ in state } X(A \text{ or } B)\end{aligned} \tag{14.78}$$

and

$$\begin{aligned} C_{126}^X(i,j) &= \frac{{C_{12}}^X(i,j)}{{C_6}^X(i,j)} \qquad \text{if} \quad {C_6}^X(i,j)\neq 0 \\ &= 0 \qquad\qquad\qquad\quad \text{if} \quad {C_6}^X(i,j)=0\end{aligned} \tag{14.79}$$

The reaction field parameters R_{rf} and C_{rf} are defined in Sec. 2-7.3.

Softcore parameters $\alpha_{LJ}(i,j)$ and $\alpha_C(i,j)$ are determined based on input parameters ALPHLJ and ALPHC in the input block PERTURB and may be multiplied by atom specific weights specified in the perturbation topology ISCLJ[i] and ISCC[i]. We distinguish the following cases

- Both atoms i and j are listed in the perturbation topology
  $\alpha_{LJ}(i,j)$ = ALPHLJ * ([ISCLJ[i] + ISCLJ[j])/2
  $\alpha_C(i,j)$ = ALPHC * (ISCC[i] + ISCC[j])/2

- Only atom i is listed in the perturbation topology
  $\alpha_{LJ}(i,j)$ = ALPHLJ * ISCLJ[i]
  $\alpha_C(i,j)$ = ALPHC * ISCC[i]
- Only atom j is listed in the perturbation topology
  $\alpha_{LJ}(i,j)$ = ALPHLJ * ISCLJ[j]
  $\alpha_C(i,j)$ = ALPHC * ISCC[j]

The *summation* in Eq. 14.75 runs *over all pairs* of atoms with sequence numbers i and j and *which involve at least one perturbed atom*. Atoms are considered to be perturbed when they are occurring in the perturbation molecular topology file (see Sec. 4-3.3). As discussed in Sec. 2.3, a number of pairs is excluded from the summation in Eq. 14.75:

- excluded neighbour atom pairs (Sec. 2.3), for these pairs, however, the reaction-field and self-interaction terms corresponding to Eq. 7.11 and Eq. 7.12 are evaluated accordingly.
- atom pairs not included in the charge group pair list (cut-off R_{cp}) or in the long-range non-bonded interaction (cut-off R_{cl}).

The non-bonded interaction between *not-perturbed atoms* is evaluated using Eq. 6.1 and Eq. 7.10-Eq. 7.12. This implies that for such atom pairs no soft-core interaction can be invoked. In other words, for a conformational search simulation involving soft-core atoms, the soft-core atoms can only be selected using the perturbation topology file.

We note that for $n > 1$ in Eq. 14.75 or $\alpha_{LJ} \neq 0$ or $\alpha_C \neq 0$ in Eq. 14.76 the path connecting $\mathcal{V}^{(nbd)}(A)$ and $\mathcal{V}^{(nbd)}(B)$ is non-linear. This implies that even when the end states are chosen to be identical, $A = B$, the path connecting them is $\lambda-$dependent. For example, considering only a $\lambda-$dependence between state A ($\lambda = 0$) and state B ($\lambda = 1$) taken equal to A, we have $\mathcal{V}^{(nbd)}(\lambda = 0) = \mathcal{V}^{(nbd)}(\lambda = 1) \neq \mathcal{V}^{(nbd)}(0 < \lambda < 1)$.

The forces on atoms i and j due to the (i,j)-th term in formula Eq. 14.75 are

$$\begin{aligned} \mathbf{f}^{(nbd)}{}_i &= -\frac{\partial \mathcal{V}^{(nbd)}}{\partial \mathbf{r}_i} \\ &= -\lambda^n \frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};B;(1-\lambda))}{\partial \mathbf{r}_i} - (1-\lambda)^n \frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};A;\lambda)}{\partial \mathbf{r}_i} \end{aligned} \tag{14.80}$$

and

$$\mathbf{f}^{(nbd)}{}_j = -\mathbf{f}^{(nbd)}{}_i \tag{14.81}$$

with

$$\begin{aligned} &\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};X;\lambda)}{\partial \mathbf{r}_i} = \\ &\frac{-6(r_{ij})^4}{\left[\ \alpha_{LJ}(i,j)(\lambda^2) C^X_{126}(i,j) + (r_{ij})^6\right]^2} \cdot \left[\frac{2{C_{12}}^X(i,j)}{\left[\ \alpha_{LJ}(i,j)(\lambda^2) C^X_{126}(i,j) + (r_{ij})^6\right]} - {C_6}^X(i,j) \right] \cdot \mathbf{r}_{ij} \\ &-\frac{q_i^X q_j^X}{4\pi\varepsilon_0\varepsilon_1} \left[\frac{1}{\left[\ \alpha_c(i,j)(\lambda^2) + (r_{ij})^2\right]^{\frac{3}{2}}} + \frac{C_{rf}}{\left[\ \alpha_c(i,j)(\lambda^2) + {R_{rf}}^2\right]^{\frac{3}{2}}} \right] \cdot \mathbf{r}_{ij}. \end{aligned} \tag{14.82}$$

The derivative with respect to λ of the non-bonded interaction is

$$\begin{aligned} &\frac{\partial \mathcal{V}^{(nbd)}}{\partial \lambda} = \\ &\sum_{\substack{nonbonded \\ perturbed \\ pairs(i,j)}} \Big\{ n\lambda^{n-1} \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};B;(1-\lambda)) + \lambda^n \frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};B;(1-\lambda))}{\partial \lambda} \\ &-n(1-\lambda)^{n-1} \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};A;\lambda) + (1-\lambda)^n \frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};A;\lambda)}{\partial \lambda} \Big\} \end{aligned} \tag{14.83}$$

with

$$\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};X;\lambda)}{\partial\lambda} =$$

$$\frac{-2\lambda\alpha_{LJ}(i,j)C^X_{126}(i,j)}{\left[\ \alpha_{LJ}(i,j)(\lambda^2)C^X_{126}(i,j)+(r_{ij})^6\right]^2}\cdot\left[\ \frac{2C_{12}{}^X(i,j)}{\left[\ \alpha_{LJ}(i,j)(\lambda^2)C^X_{126}(i,j)+(r_{ij})^6\right]}-C_6{}^X(i,j)\right] \tag{14.84}$$

$$-\frac{q^X_i q^X_j}{4\pi\varepsilon_0\varepsilon_1}\cdot\lambda\alpha_c(i,j)\cdot\left[\ \frac{1}{\left[\ \alpha_c(i,j)(\lambda^2)+(r_{ij})^2\right]^{\frac{3}{2}}}-\frac{\frac{3}{2}C_{rf}(r_{ij})^2}{\left[\ \alpha_c(i,j)(\lambda^2)+R_{rf}{}^2\right]^{\frac{5}{2}}}\right]$$

and

$$\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};X;(1-\lambda))}{\partial\lambda} = -\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r}_{ij};X;\lambda)}{\partial\lambda} \tag{14.85}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is

$$\mathcal{V}^{(nbd)}(\mathbf{r}_{ij};\lambda\pm\Delta\lambda)-\mathcal{V}^{(nbd)}(\mathbf{r}_{ij};\lambda) \tag{14.86}$$

which can easily be obtained from Eq. 14.75 and Eq. 14.76.

14.2.9. Polarization. For free energy calculations using the coupling parameter approach (Sec. 14.1) the Hamiltonian of the system has to be made dependent on a coupling parameter λ. Free energy differences between two states A ($\lambda=0$) and B ($\lambda=1$) are then obtained from a thermodynamic integration over the averages of the derivative of the Hamiltonian with repect to λ. Several terms will change compared to the non-polarisable case[45].

The λ-dependent V^{coul} is

$$\begin{aligned}
\mathcal{V}^{coul}(\boldsymbol{r}^N,\boldsymbol{r}'^N;\lambda) &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}}\sum_{i=1}^{N-1}\sum_{\substack{j>i\\ j\ inside\ cut-off\ i\\ (i,j)\ not\ excluded}}^{N}\Bigg[\\
&\quad (1-\lambda)^n\Bigg\{(q^A_i-q^{A,v}_i)(q^A_j-q^{A,v}_j)\left(\frac{1}{(|\mathbf{r}_i-\mathbf{r}_j|^2+\alpha_C\lambda^2)^{1/2}}\right)\\
&+ (q^A_i-q^{A,v}_i)q^A_j\left(\frac{1}{(|\mathbf{r}_i-\mathbf{r}'_j|^2+\alpha_C\lambda^2)^{1/2}}\right)\\
&+ q^{A,v}_i(q^A_j-q^{A,v}_j)\left(\frac{1}{(|\mathbf{r}'_i-\mathbf{r}_j|^2+\alpha_C\lambda^2)^{1/2}}\right)\\
&+ q^{A,v}_i q^{A,v}_j\left(\frac{1}{(|\mathbf{r}'_i-\mathbf{r}'_j|^2+\alpha_C\lambda^2)^{1/2}}\right)\Bigg\}\\
&+ \lambda^n\Bigg\{(q^B_i-q^{B,v}_i)(q^B_j-q^{B,v}_j)\left(\frac{1}{(|\mathbf{r}_i-\mathbf{r}_j|^2+\alpha_C(1-\lambda)^2)^{1/2}}\right)\\
&+ (q^B_i-q^{B,v}_i)q^B_j\left(\frac{1}{(|\mathbf{r}_i-\mathbf{r}'_j|^2+\alpha_C(1-\lambda)^2)^{1/2}}\right)\\
&+ q^{B,v}_i(q^B_j-q^{B,v}_j)\left(\frac{1}{(|\mathbf{r}'_i-\mathbf{r}_j|^2+\alpha_C(1-\lambda)^2)^{1/2}}\right)\\
&+ q^{B,v}_i q^{B,v}_j\left(\frac{1}{(|\mathbf{r}'_i-\mathbf{r}'_j|^2+\alpha_C(1-\lambda)^2)^{1/2}}\right)\Bigg\}\Bigg]
\end{aligned} \tag{14.87}$$

and the λ-dependent V^{rf} is

$$\begin{aligned}
V^{rf}(\boldsymbol{r}^N,\boldsymbol{r}'^N;\lambda) &= -\frac{1}{4\pi\epsilon_0\epsilon_{cs}}\sum_{i=1}^{N-1}\sum_{\substack{j>i\\ j\ inside\ cut-off\ i}}^{N}\Bigg[\\
&\quad (1-\lambda)^n\Bigg\{(q^A_i-q^{A,v}_i)(q^A_j-q^{A,v}_j)\left(C_{rf,aux}(\lambda)|\mathbf{r}_i-\mathbf{r}_j|^2+\frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right)
\end{aligned}$$

$$
\begin{aligned}
&+ \quad (q_i^A - q_i^{A,v})q_j^A \left(C_{rf,aux}(\lambda)|\mathbf{r}_i - \mathbf{r}'_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \\
&+ \quad q_i^{A,v}(q_j^A - q_j^{A,v}) \left(C_{rf,aux}(\lambda)|\mathbf{r}'_i - \mathbf{r}_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \\
&+ \quad q_i^{A,v}q_j^{A,v} \left(C_{rf,aux}(\lambda)|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \Bigg\} \\
&+ \quad \lambda^n \Bigg\{ (q_i^B - q_i^{B,v})(q_j^B - q_j^{B,v}) \left(C_{rf,aux}(1-\lambda)|\mathbf{r}_i - \mathbf{r}_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \\
&+ \quad (q_i^B - q_i^{B,v})q_j^B \left(C_{rf,aux}(1-\lambda)|\mathbf{r}_i - \mathbf{r}'_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \\
&+ \quad q_i^{B,v}(q_j^B - q_j^{B,v}) \left(C_{rf,aux}(1-\lambda)|\mathbf{r}'_i - \mathbf{r}_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \\
&+ \quad q_i^{B,v}q_j^{B,v} \left(C_{rf,aux}(1-\lambda)|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \frac{1-\frac{1}{2}C_{rf}}{R_{rf}} \right) \Bigg\} \Bigg] \\
&- \quad \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{i=1}^{N} [(1-\lambda)^n (q_i^A)^2 + \lambda^n (q_i^B)^2] \frac{1}{2} \frac{(1-\frac{1}{2}C_{rf})}{R_{rf}}
\end{aligned}
\tag{14.88}
$$

where

$$
C_{rfaux}(\lambda) = \frac{\frac{1}{2}C_{rf}}{({R_{rf}}^2 + \alpha_C \lambda^2)^{3/2}} \tag{14.89}
$$

is an auxilary function to simplify the formula, $q_i^{A(B)}$ is the charge on atom i in state $A(B)$, α_C is the soft core parameter, and C_{rf} defined by Eq. 7.126. The derivatives with respect to λ are then

$$
\begin{aligned}
\frac{\partial V^{coul}}{\partial \lambda} &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{i=1}^{N-1} \sum_{\substack{j \ inside \ cut-off \ i \\ (i,j) \ not \ excluded}}^{N} \Bigg[-(1-\lambda)^n \lambda \alpha_C \\
&\quad \Bigg\{ (q_i^A - q_i^{A,v})(q_j^A - q_j^{A,v}) \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C\lambda^2)^{3/2}} \right) \\
&+ \quad (q_i^A - q_i^{A,v})q_j^A \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C\lambda^2)^{3/2}} \right) \\
&+ \quad q_i^{A,v}(q_j^A - q_j^{A,v}) \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}_j|^2 + \alpha_C\lambda^2)^{3/2}} \right) + q_i^{A,v}q_j^{A,v} \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \alpha_C\lambda^2)^{3/2}} \right) \Bigg\} \\
&+ \quad \lambda^n(1-\lambda)\alpha_C \\
&\quad \Bigg\{ (q_i^B - q_i^{B,v})(q_j^B - q_j^{B,v}) \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C(1-\lambda)^2)^{3/2}} \right) \\
&+ \quad (q_i^B - q_i^{B,v})q_j^B \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C(1-\lambda)^2)^{3/2}} \right) \\
&+ \quad q_i^{B,v}(q_j^B - q_j^{B,v}) \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}_j|^2 + \alpha_C(1-\lambda)^2)^{3/2}} \right) + q_i^{B,v}q_j^{B,v} \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \alpha_C(1-\lambda)^2)^{3/2}} \right) \Bigg\} \\
&- \quad n(1-\lambda)^{n-1} \\
&\quad \Bigg\{ (q_i^A - q_i^{A,v})(q_j^A - q_j^{A,v}) \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C\lambda^2)^{1/2}} \right) + (q_i^A - q_i^{A,v})q_j^A \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C\lambda^2)^{1/2}} \right) \\
&+ \quad q_i^{A,v}(q_j^A - q_j^{A,v}) \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}_j|^2 + \alpha_C\lambda^2)^{1/2}} \right) + q_i^{A,v}q_j^{A,v} \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \alpha_C\lambda^2)^{1/2}} \right) \Bigg\} \\
&+ \quad n\lambda^{n-1} \\
&\quad \Bigg\{ (q_i^B - q_i^{B,v})(q_j^B - q_j^{B,v}) \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C(1-\lambda)^2)^{1/2}} \right) \\
&+ \quad (q_i^B - q_i^{B,v})q_j^B \left(\frac{1}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C(1-\lambda)^2)^{1/2}} \right) \\
&+ \quad q_i^{B,v}(q_j^B - q_j^{B,v}) \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}_j|^2 + \alpha_C(1-\lambda)^2)^{1/2}} \right) \\
&+ \quad q_i^{B,v}q_j^{B,v} \left(\frac{1}{(|\mathbf{r}'_i - \mathbf{r}'_j|^2 + \alpha_C(1-\lambda)^2)^{1/2}} \right) \Bigg\} \Bigg]
\end{aligned}
\tag{14.90}
$$

and

$$
\begin{aligned}
\frac{\partial V^{rf}}{\partial \lambda} &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{i=1}^{N-1} \sum_{\substack{j>i \\ j\ inside\ cut-off\ i}}^{N} \Bigg[-(1-\lambda)^n \lambda \alpha_C \\
&\quad \Bigg\{ (q_i^A - q_i^{A,v})(q_j^A - q_j^{A,v}) \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}_i - \mathbf{r}_j|^2}{(R_{rf}^2 + \alpha_C \lambda^2)^{5/2}} \right) + (q_i^A - q_i^{A,v}) q_j^A \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}_i - \mathbf{r}'_j|^2}{(R_{rf}^2 + \alpha_C \lambda^2)^{5/2}} \right) \\
&+ q_i^{A,v}(q_j^A - q_j^{A,v}) \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}'_i - \mathbf{r}_j|^2}{(R_{rf}^2 + \alpha_C \lambda^2)^{5/2}} \right) + q_i^{A,v} q_j^{A,v} \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}'_i - \mathbf{r}'_j|^2}{(R_{rf}^2 + \alpha_C \lambda^2)^{5/2}} \right) \Bigg\} \\
&+ \lambda^n (1-\lambda) \alpha_C \\
&\quad \Bigg\{ (q_i^B - q_i^{B,v})(q_j^B - q_j^{B,v}) \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}_i - \mathbf{r}_j|^2}{(R_{rf}^2 + \alpha_C (1-\lambda)^2)^{5/2}} \right) + (q_i^B - q_i^{B,v}) q_j^B \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}_i - \mathbf{r}'_j|^2}{(R_{rf}^2 + \alpha_C (1-\lambda)^2)^{5/2}} \right) \\
&+ q_i^{B,v}(q_j^B - q_j^{B,v}) \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}'_i - \mathbf{r}_j|^2}{(R_{rf}^2 + \alpha_C (1-\lambda)^2)^{5/2}} \right) + q_i^{B,v} q_j^{B,v} \left(-\frac{\frac{3}{2} C_{rf} |\mathbf{r}'_i - \mathbf{r}'_j|^2}{(R_{rf}^2 + \alpha_C (1-\lambda)^2)^{5/2}} \right) \Bigg\} \\
&- n(1-\lambda)^{n-1} \\
&\quad \Bigg\{ (q_i^A - q_i^{A,v})(q_j^A - q_j^{A,v}) \left(-C_{rf,aux}(\lambda) |\mathbf{r}_i - \mathbf{r}_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ (q_i^A - q_i^{A,v}) q_j^A \left(-C_{rf,aux}(\lambda) |\mathbf{r}_i - \mathbf{r}'_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ q_i^{A,v}(q_j^A - q_j^{A,v}) \left(-C_{rf,aux}(\lambda) |\mathbf{r}'_i - \mathbf{r}_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ q_i^{A,v} q_j^{A,v} \left(-C_{rf,aux}(\lambda) |\mathbf{r}'_i - \mathbf{r}'_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \Bigg\} \\
&+ n\lambda^{n-1} \\
&\quad \Bigg\{ (q_i^B - q_i^{B,v})(q_j^B - q_j^{B,v}) \left(-C_{rf,aux}(1-\lambda) |\mathbf{r}_i - \mathbf{r}_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ (q_i^B - q_i^{B,v}) q_j^B \left(-C_{rf,aux}(1-\lambda) |\mathbf{r}_i - \mathbf{r}'_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ q_i^{B,v}(q_j^B - q_j^{B,v}) \left(-C_{rf,aux}(1-\lambda) |\mathbf{r}'_i - \mathbf{r}_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \\
&+ q_i^{B,v} q_j^{B,v} \left(-C_{rf,aux}(1-\lambda) |\mathbf{r}'_i - \mathbf{r}'_j|^2 - \frac{1 - \frac{1}{2} C_{rf}}{R_{rf}} \right) \Bigg\} \Bigg] \\
&- \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{i=1}^{N} ([-n(1-\lambda)^{n-1} (q_i^A)^2 + n\lambda^{n-1} (q_i^B)^2] \frac{1}{2} \frac{(1 - \frac{1}{2} C_{rf})}{R_{rf}}
\end{aligned}
\tag{14.91}
$$

Using for the λ-dependence of the polarisability $\alpha_i(\lambda)$

$$\alpha_i(\lambda) = (1-\lambda)^m \alpha_i^A + \lambda^m \alpha_i^B,$$

the λ-dependence of the electric field $\mathbf{E}_i(\lambda)$ is

$$
\begin{aligned}
\mathbf{E}_i^{coul}(\lambda) &= \frac{1}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{j>i \\ j\ inside\ cut-off\ i \\ (i,j)\ not\ excluded}}^{N} \Bigg\{ (1-\lambda)^n \\
&\quad \left[\frac{(q_j^A - q_j^{A,v})(\mathbf{r}_i - \mathbf{r}_j)}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C \lambda^2)^{3/2}} + \frac{q_j^{A,v}(\mathbf{r}_i - \mathbf{r}'_j)}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C \lambda^2)^{3/2}} \right] \\
&\quad + \lambda^n \left[\frac{(q_j^B - q_j^{B,v})(\mathbf{r}_i - \mathbf{r}_j)}{(|\mathbf{r}_i - \mathbf{r}_j|^2 + \alpha_C (1-\lambda)^2)^{3/2}} + \frac{q_j^{B,v}(\mathbf{r}_i - \mathbf{r}'_j)}{(|\mathbf{r}_i - \mathbf{r}'_j|^2 + \alpha_C (1-\lambda)^2)^{3/2}} \right] \Bigg\}
\end{aligned}
\tag{14.92}
$$

and

$$
\mathbf{E}_i^{rf}(\lambda) = \frac{C_{rf}}{4\pi\epsilon_0\epsilon_{cs}} \sum_{\substack{j>i \\ j\ inside\ cut-off\ i}}^{N} \Bigg\{ (1-\lambda)^n
$$

$$\left[\frac{(q_j^A - q_j^{A,v})(\mathbf{r}_i - \mathbf{r}_j)}{(R_{rf}^2 + \alpha_C\lambda^2)^{3/2}} + \frac{q_j^{A,v}(\mathbf{r}_i - \mathbf{r}'_j)}{(R_{rf}^2 + \alpha_C\lambda^2)^{3/2}}\right]$$
$$+\lambda^n\left[\frac{(q_j^B - q_j^{B,v})(\mathbf{r}_i - \mathbf{r}_j)}{(R_{rf}^2 + \alpha_C(1-\lambda)^2)^{3/2}} + \frac{q_j^{B,v}(\mathbf{r}_i - \mathbf{r}'_j)}{(R_{rf}^2 + \alpha_C(1-\lambda)^2)^{3/2}}\right]\Bigg\} \tag{14.93}$$

and one gets for the λ-dependent $V^{self,i}$

$$\begin{aligned} V^{self,i} & & (14.94)\\ &= \frac{1}{2}\alpha_i(\lambda)(E_i(\lambda))^2 && for\ E_i(\lambda) \le E_{0,i}\\ &= \frac{1}{2}\alpha_i(\lambda)(E_{0,i})^2 &&\\ &+ \frac{\alpha_i(\lambda)(E_{0,i})^2}{p_i(p_i-1)}\left[-p_i^2 + (p_i^2-1)\left(\frac{E_i(\lambda)}{E_{0,i}}\right) + \left(\frac{E_{0,i}}{E_i(\lambda)}\right)^{p_i-1}\right] && for\ E_i(\lambda) > E_{0,i}. \end{aligned}$$

The λ-derivative of this perturbed self polarisation potential energy is

$$\begin{aligned} \frac{\partial V^{self,i}}{\partial\lambda} & & (14.95)\\ &= \frac{1}{2}m(\lambda^{m-1}\alpha_i^B - (1-\lambda)^{m-1}\alpha_i^A)(\mathbf{E}_i(\lambda))^2 + \alpha_i(\lambda)\mathbf{E}_i(\lambda)\cdot\frac{\partial\mathbf{E}_i(\lambda)}{\partial\lambda} && for\ E_i(\lambda) \le E_{0,i}\\ &= m(\lambda^{m-1}\alpha_i^B - (1-\lambda)^{m-1}\alpha_i^A)(E_{0,i})^2 &&\\ &\left(\frac{1}{2} + \frac{1}{p_i(p_i-1)}\left[-p_i^2 + (p_i^2-1)\left(\frac{E_i(\lambda)}{E_{0,i}}\right) + \left(\frac{E_{0,i}}{E_i(\lambda)}\right)^{p_i-1}\right]\right) &&\\ &+\frac{\alpha_i(\lambda)E_{0,i}}{p_i}\left[(p_i+1) - \left(\frac{E_{0,i}}{E_i(\lambda)}\right)^{p_i}\right]\frac{\partial E_i(\lambda)}{\partial\lambda} && for\ E_i(\lambda) > E_{0,i}. \end{aligned}$$

with

$$\frac{\partial E_i(\lambda)}{\partial\lambda} = \frac{\mathbf{E}_i(\lambda)\cdot\frac{\partial\mathbf{E}_i(\lambda)}{\partial\lambda}}{(\mathbf{E}_i(\lambda)\cdot\mathbf{E}_i(\lambda))^{1/2}} \tag{14.96}$$

14.2.10. Perturbed atom-atom distance restraints. The $\lambda-$dependent version of the atom-atom distance restraining term (Sec. 9.3) is obtained by making the restraining force constant ${k^{(dr)}}_k$ and the (repulsive or attractive) distance restraint k with length ${\mathbf{r}^0}_k$ between atoms i and i' dependent on λ,

$$\mathcal{V}^{(dr)}(\mathbf{r};\lambda) = \sum_{k=1}^{N^{(dir)}} 2^{n+m}\lambda^n(1-\lambda)^m{\mathcal{V}^{(dr)}}_k^{AB}(\mathbf{r}_{ii'}; {k^{(dr)}}_k^A, {k^{(dr)}}_k^B; {\mathbf{r}^0}_k^A, {\mathbf{r}^0}_k^B, \Delta r^h) \tag{14.97}$$

The pre-factor $2^{n+m}\ \lambda^n\ (1-\lambda)^m$ with user-specified exponents n and m is added to make the use of perturbed distance restraints possible without restraining the system in the end states,[100] using the so-called hidden restraints.

For a perturbed *attractive* distance restraint, ${\mathcal{V}^{(dr)}}_k^{AB}$ reads

$${\mathcal{V}^{(dr)}}_k^{AB}(\mathbf{r}_{ii'}, {k^{(dr)}}_n^A, {k^{(dr)}}_k^B, {\mathbf{r}^0}_k^A, {\mathbf{r}^0}_k^B, \Delta r^h) = 0$$

$$for\ 0 < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A + \lambda\ {\mathbf{r}^0}_k^B$$

$$= \tfrac{1}{2}((1-\lambda){k^{(dr)}}_k^A + \lambda{k^{(dr)}}_k^B)[r_{ii'} - (1-\lambda){\mathbf{r}^0}_k^A - \lambda{\mathbf{r}^0}_k^B]^2$$

$$for\ (1-\lambda){\mathbf{r}^0}_k^A + \lambda{\mathbf{r}^0}_k^B < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A + \lambda{\mathbf{r}^0}_k^B + \Delta r^h$$

$$= ((1-\lambda){k^{(dr)}}_k^A + \lambda{k^{(dr)}}_k^B)[r_{ii'} - (1-\lambda){\mathbf{r}^0}_k^A - \lambda{\mathbf{r}^0}_k^B - \tfrac{1}{2}\Delta r^h]\Delta r^h$$

$$for\ (1-\lambda){\mathbf{r}^0}_k^A - \lambda{\mathbf{r}^0}_k^B + \Delta r^h < r_{ii'} \tag{14.98}$$

and the forces on atom i and i' due to ${\mathcal{V}^{(dr)}}_k^{AB}$ in Eq. 14.98 are

$$\mathbf{f}^{(dir)}{}_i = -2^{n+m}\lambda^n(1-\lambda)^m \frac{\partial {\mathcal{V}^{(dr)}}_k^{AB}}{\partial r_{ii'}} \frac{\partial r_{ii'}}{\partial \mathbf{r}_i} \tag{14.99}$$

with

$$\begin{aligned}
&\frac{\partial {\mathcal{V}^{(dr)}}_k^{AB}}{\partial r_{ii'}} \frac{\partial r_{ii'}}{\partial \mathbf{r}_i} = 0 \\
&\qquad\qquad for\ 0 < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B \\
&= ((1-\lambda){k^{(dr)}}_k^A + \lambda {k^{(dr)}}_k^B)[r_{ii'} - (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B] \cdot \frac{\mathbf{r}_i}{r_{ii'}} \\
&\qquad\qquad for\ (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B + \Delta r^h \\
&= ((1-\lambda){k^{(dr)}}_k^A + \lambda {k^{(dr)}}_k^B) \cdot \Delta \mathbf{r}^h \cdot \frac{\mathbf{r}_i}{r_{ii'}} \\
&\qquad\qquad for\ (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B + \Delta r^h < r_{ii'}
\end{aligned} \tag{14.100}$$

and

$$\mathbf{f}^{(dir)}{}_{i'} = -\mathbf{f}^{(dir)}{}_i \tag{14.101}$$

For a perturbed *repulsive* distance restraint, ${\mathcal{V}^{(dr)}}_k^{AB}$ reads

$$\begin{aligned}
&{\mathcal{V}^{(dr)}}_k^{AB}(r_{ii'}, {k^{(dr)}}_k^A, {k^{(dr)}}_k^B, {\mathbf{r}^0}_k^A, {\mathbf{r}^0}_k^B, \Delta r^h) \\
&= ((1-\lambda){k^{(dr)}}_k^A + \lambda {k^{(dr)}}_k^B)[r_{ii'} - (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B - \tfrac{1}{2}\Delta r^h]\Delta r^h \\
&\qquad\qquad for\ 0 < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B - \Delta r^h \\
&= \tfrac{1}{2}((1-\lambda){k^{(dr)}}_k^A + \lambda {k^{(dr)}}_k^B)[r_{ii'} - (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B]^2 \\
&\qquad\qquad for\ (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B - \Delta r^h < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B \\
&= 0 \\
&\qquad\qquad for\ (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^0}_k^B < r_{ii'}
\end{aligned} \tag{14.102}$$

and the forces on atom i and i' due to ${\mathcal{V}^{(dr)}}_k^{AB}$ in Eq. 14.102 are calculated from Eq. 14.99 with

$$\begin{aligned}
&\frac{\partial {\mathcal{V}^{(dr)}}_k^{AB}}{\partial r_{ii'}} \frac{\partial r_{ii'}}{\partial \mathbf{r}_i} = ((1-\lambda){k^{(dr)}}_k^A + \lambda {k^{(dr)}}_k^B) \cdot \Delta r^h \cdot \frac{\mathbf{r}_i}{r_{ii'}} \\
&\qquad\qquad for\ 0 < r_{ii'} < (1-\lambda){\mathbf{r}^0}_k^A - \lambda {\mathbf{r}^{0\,0,B}}_k - \Delta r^h
\end{aligned}$$

$$= ((1-\lambda)k^{(dr)}{}_k^A + \lambda k^{(dr)}{}_k^B) \cdot [r_{ii'} - (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B] \cdot \frac{\mathbf{r}_i}{r_{ii'}}$$

$$for\ (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B - \Delta r^h < r_{ii'} < (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B$$

$$= 0$$

$$for\ (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B < r_{ii'} \tag{14.103}$$

and

$$\mathbf{f}^{(dir)}{}_{i'} = -\mathbf{f}^{(dir)}{}_i \tag{14.104}$$

The $\lambda-$derivative of $\mathcal{V}^{(dr)}(\mathbf{r};\lambda)$ reads

$$\frac{\partial \mathcal{V}^{(dr)}(\mathbf{r};\lambda)}{\partial \lambda} = \sum_{k=1}^{N^{(dir)}} 2^{n+m}[(n\lambda^{n-1}(1-\lambda)^m - m\lambda^n(1-\lambda)^{m-1})$$

$$\cdot\ \mathcal{V}^{(dr)}{}_k^{AB}(r_{ii'}, k^{(dr)}{}_k^A, k^{(dr)}{}_k^B, \mathbf{r}^0{}_k^A, \mathbf{r}^0{}_k^B, \Delta r^h)$$

$$+\lambda^n(1-\lambda)^m \frac{\partial \mathcal{V}^{(dr)}{}_k^{AB}(r_{ii'}, k^{(dr)}{}_k^A, k^{(dr)}{}_k^B, \mathbf{r}^0{}_k^A, \mathbf{r}^0{}_k^B, \Delta r^h)}{\partial \lambda}] \tag{14.105}$$

In case of an *attractive* distance restraint,

$$\frac{\partial \mathcal{V}^{(dr)}{}_k^{AB}(r_{ii'}, k^{(dr)}{}_k^A, k^{(dr)}{}_k^B, \mathbf{r}^0{}_k^A, \mathbf{r}^0{}_k^B, \Delta r^h)}{\partial \lambda} = 0$$

$$for\ 0 < r_{ii'} < (1-\lambda)\mathbf{r}^0{}_k^A + \mathbf{r}^0{}_k^B$$

$$= \tfrac{1}{2}(k^{(dr)}{}_k^B - k^{(dr)}{}_k^A)[r_{ii'} - (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B]^2$$

$$+((1-\lambda)k^{(dr)}{}_k^A + \lambda k^{(dr)}{}_k^B)[r_{ii'} - (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B] \cdot (\mathbf{r}^0{}_k^A - \mathbf{r}^0{}_k^B)$$

$$for\ (1-\lambda)\mathbf{r}^0{}_k^A + \lambda \mathbf{r}^0{}_k^B < r_{ii'} < (1-\lambda)\mathbf{r}^0{}_k^A + \lambda \mathbf{r}^0{}_k^B + \Delta r^h$$

$$= (k^{(dr)}{}_k^B - k^{(dr)}{}_k^A)(r_{ii'} - (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B - \tfrac{1}{2}\Delta r^h)\Delta r^h$$

$$+((1-\lambda)k^{(dr)}{}_k^A + \lambda k^{(dr)}{}_k^B)\Delta r^h(\mathbf{r}^0{}_k^A - \mathbf{r}^0{}_k^B)$$

$$for\ (1-\lambda)\mathbf{r}^0{}_k^A + \lambda \mathbf{r}^0{}_k^B + \Delta r^h < r_{ii'} \tag{14.106}$$

In case of a *repulsive* distance restraint,

$$\frac{\partial \mathcal{V}^{(dr)}{}_k^{AB}(r_{ii'}, k^{(dr)}{}_k^A, k^{(dr)}{}_k^B, \mathbf{r}^0{}_k^A, \mathbf{r}^0{}_k^B, \Delta r^h)}{\partial \lambda}$$

$$= -[(k^{(dr)}{}_k^B - k^{(dr)}{}_k^A)(r_{ii'} - (1-\lambda)\mathbf{r}^0{}_k^A - \lambda \mathbf{r}^0{}_k^B - \tfrac{1}{2}\Delta r^h)\Delta r^h$$

$$+((1-\lambda)k^{(dr)\,A}_k + \lambda k^{(dr)\,B}_k)\Delta r^h(\mathbf{r}^{0\,A}_k - \mathbf{r}^{0\,B}_k)]$$

$$for\ 0 < r_{ii'} < (1-\lambda)\mathbf{r}^{0\,A}_k - \lambda\mathbf{r}^{0\,B}_k - \Delta r^h$$

$$= \tfrac{1}{2}(k^{(dr)\,B}_k - k^{(dr)\,A}_k)[r_{ii'} - (1-\lambda)\mathbf{r}^{0\,A}_k - \lambda\mathbf{r}^{0\,B}_k]^2$$

$$+((1-\lambda)k^{(dr)\,A}_k + \lambda k^{(dr)\,B}_k)[r_{ii'} - (1-\lambda)\mathbf{r}^{0\,A}_k - \lambda\mathbf{r}^{0\,B}_k]\cdot(\mathbf{r}^{0\,A}_k - \mathbf{r}^{0\,B}_k)$$

$$for\ 0 < r_{ii'} < (1-\lambda)\mathbf{r}^{0\,A}_k - \lambda\mathbf{r}^{0\,B}_k - \Delta r^h$$

$$= 0$$

$$for\ (1-\lambda)\mathbf{r}^{0\,A}_k + \lambda\mathbf{r}^{0\,B}_k < r_{ii'} \tag{14.107}$$

14.2.11. Perturbed dihedral angle restraints. Similar to distance restraining, a λ dependence can be introduced to the dihedral-angle restraints to enforce conformational sampling along a pathway from state A to state B

$$\mathcal{V}^{(tr)}(\boldsymbol{r},\lambda) = \sum_{k=1}^{\mathcal{N}^{(tr)}} 2^{n+m}\lambda^n\,(1-\lambda)^m\,\mathcal{V}^{(tr)\,AB}(\phi_k; k^{(tr)\,A}, k^{(tr)\,B}; \phi^{0A}_k, \phi^{0B}_k, \delta\phi^h_k,), \tag{14.108}$$

The pre-factor $2^{n+m}\lambda^n(1-\lambda)^m$ with user-specified exponents n and m is added to make use of perturbed dihedral restraints possible without restraining the system in the end states, using so-called hidden restraints.[100]

For a perturbed dihedral angle restraint, $\mathcal{V}^{(tr)\,AB}$ reads

$$\begin{array}{ll}
\mathcal{V}^{(tr)\,AB}(\phi; k^{(tr)\,A}, k^{(tr)\,B}; \phi^{0A}_k, \phi^{0B}_k, \delta\phi^h_k) = & \\
-[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}](\Delta\phi_k + 1/2\Delta\phi^h)\Delta\phi^h & \Delta\phi_k < -\Delta\phi^h \\
1/2[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}](\Delta\phi_k)^2 & -\Delta\phi^h \le \Delta\phi_k \le \Delta\phi^h \\
[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}](\Delta\phi_k - 1/2\Delta\phi^h)\Delta\phi^h & \Delta\phi_k > \Delta\phi^h
\end{array} \tag{14.109}$$

where

$$\Delta\phi_k = \phi_k - (1-\lambda)\phi^{0A}_k - \lambda\phi^{0B}_k + 2m\pi. \tag{14.110}$$

and m is chosen such that ϕ_k is within the range $[(1-\lambda)\phi^{0A}_k - \lambda\phi^{0B}_k + \delta_k - 2\pi, (1-\lambda)\phi^{0A}_k - \lambda\phi^{0B}_k + \delta_k]$. Using this dihedral angle restraint formulation δ_k determines at which position the direction of the rotation around the dihedral angle inverts. Typically, δ_k is set to 180^o. The force on atom i is

$$\mathbf{f}_i \;=\; -2^{n+m}\lambda^n\,(1-\lambda)^m\,\frac{\partial\mathcal{V}^{(tr)\,AB}}{\partial\Delta\phi_k}\frac{\partial\phi_k}{\partial\mathbf{r}_i}$$

where

$$\begin{array}{ll}
\dfrac{\partial\mathcal{V}^{(tr)\,AB}}{\partial\Delta\phi_k} = & \\
-[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}]\Delta\phi^h\dfrac{\partial\phi_k}{\partial\mathbf{r}_i} & \Delta\phi_k < -\Delta\phi^h \\
[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}]\Delta\phi_k\dfrac{\partial\phi_k}{\partial\mathbf{r}_i} & -\Delta\phi^h \le \Delta\phi_k \le \Delta\phi^h \\
[(1-\lambda)k^{(tr)\,A} + \lambda k^{(tr)\,B}]\Delta\phi^h\dfrac{\partial\phi_k}{\partial\mathbf{r}_i} & \Delta\phi_k > \Delta\phi^h
\end{array} \tag{14.111}$$

and $\frac{\partial\phi_k}{\partial\mathbf{r}_i}$ is equivalent to the expression used for the physical dihedral angle potential term.

The λ-derivative of $\mathcal{V}^{(tr)}(\boldsymbol{r},\lambda)$ reads

$$\begin{aligned}\frac{\partial \mathcal{V}^{(tr)}(\boldsymbol{r},\lambda)}{\partial \lambda} &= \sum_{k=1}^{\mathcal{N}^{(tr)}} 2^{n+m}[n\lambda^{n-1}(1-\lambda)^m - m\lambda^n(1-\lambda)^m)\mathcal{V}^{(tr)\,AB} \\ &\quad +\lambda^n(1-\lambda)^m \frac{\partial \mathcal{V}^{(tr)\,AB}}{\partial \lambda}\end{aligned} \tag{14.112}$$

where

$$\begin{aligned}\frac{\partial \mathcal{V}^{(tr)\,AB}}{\partial \lambda} &= \\ &\Delta\phi^h\Big((k^{(tr)}{}^B_k - k^{(tr)}{}^A_k)\left(\Delta\phi_{k\lambda} - 1/2\phi^0_{lin}\right) \\ &+ \Big((1-\lambda)k^{(tr)}{}^A_k + \lambda k^{(tr)}{}^B_k\Big)\Big(\phi^{0,A}_k - \phi^{0,B}_k\Big)\Big), \qquad \Delta\phi_k < -\Delta\phi^h \\ &1/2\Big(-(k^{(tr)}{}^B_k - k^{(tr)}{}^A_k)(\Delta\phi_{k\lambda})^2 \\ &+ 2\Big((1-\lambda)k^{(tr)}{}^A_k + \lambda k^{(tr)}{}^B_k\Big)\Delta\phi_k\Big(\phi^{0,A}_k - \phi^{0,B}_k\Big)\Big), \quad -\Delta\phi^h \le \Delta\phi_k \le \Delta\phi^h \\ &\Delta\phi^h\Big((k^{(tr)}{}^B_k - k^{(tr)}{}^A_k)\left(\Delta\phi_k - 1/2\Delta\phi^h\right) \\ &+ \Big((1-\lambda)k^{(tr)}{}^A_k + \lambda k^{(tr)}{}^B_k\Big)\Big(\phi^{0,A}_k - \phi^{0,B}_k\Big)\Big). \qquad \Delta\phi_k > \Delta\phi^h\end{aligned}$$

14.2.12. Perturbed distance-field distance restraints. The DF restraint can also be made dependent on a coupling parameter λ, making it more broadly applicable. In this case, the force constant and reference value of a DF restraint at a certain λ-value can be obtained with

$$l^0_{AB}(\lambda) = (1-\lambda)l^0_A + \lambda l^0_B \tag{14.113}$$

$$k^{(df)}_{AB}(\lambda) = (1-\lambda)k^{(df)}_A + \lambda k^{(df)}_B \tag{14.114}$$

Here, l^0_A and l^0_B are the reference values in two states A and B, respectively. Similarly, $k^{(df)}_A$ and $k^{(df)}_B$ represent the force constants for the DF restraint in states A and B, respectively. The potential energy can be calculated with

$$\mathcal{V}^{(df)}(\lambda) = 2^{n+m}\lambda^n(1-\lambda)^m\mathcal{V}^{(df)}(l_{ij}, k^{(df)}_{AB}(\lambda), l^0_{AB}(\lambda), \Delta l^h) \tag{14.115}$$

with $\mathcal{V}^{(df)}(l_{ij}, k^{(df)}_{AB}(\lambda), l^0_{AB}(\lambda), \Delta l^h)$ calculated according to Eq. 9.139. Here, we have introduced the prefactor term with variables m and n to be able to create so called hidden restraints.[100] Depending on the choice of these variables, the restraint is not present in one or both of the end states. The energy derivative can now be calculated using Eq. 14.116 and Eq. 14.117:

$$\begin{aligned}\frac{\partial \mathcal{V}^{(df)}(\lambda)}{\partial \lambda} &= 2^{n+m}[n\lambda^{n-1}(1-\lambda)^m - m\lambda^n(1-\lambda)^{m-1}]\mathcal{V}^{(df)}(l_{ij}, k^{(df)}_{AB}(\lambda), l^0_{AB}(\lambda), \Delta l^h) \\ &+ 2^{n+m}\lambda^n(1-\lambda)^m \frac{\partial \mathcal{V}^{(df)}(l_{ij}, k^{(df)}_{AB}(\lambda), l^0_{AB}(\lambda), \Delta l^h)}{\partial \lambda}\end{aligned} \tag{14.116}$$

with

$$\begin{aligned}&\frac{\partial \mathcal{V}^{(df)}(l_{ij}, k^{(df)}_{AB}(\lambda), l^0_{AB}(\lambda), \Delta l^h)}{\partial \lambda} \\ &= -(k^{(df)}_B - k^{(df)}_A)[l_{ij} - l^0_{AB}(\lambda) + \tfrac{1}{2}\Delta l^h]\tfrac{1}{2}\Delta l^h + k^{(df)}_{AB}(\lambda)(l^0_B - l^0_A)\Delta l^h && l_{ij} \le l^0 - \Delta l^h \\ &= \tfrac{1}{2}(k^{(df)}_B - k^{(df)}_A)[l_{ij} - l^0_{AB}(\lambda)]^2 - k^{(df)}_{AB}(\lambda)[l_{ij} - l^0_{AB}(\lambda)](l^0_B - l^0_A) && l^0 - \Delta l^h < l_{ij} \le l^0 + \Delta l^h \\ &= (k^{(df)}_B - k^{(df)}_A)[l_{ij} - l^0_{AB}(\lambda) - \tfrac{1}{2}\Delta l^h]\tfrac{1}{2}\Delta l^h - k^{(df)}_{AB}(\lambda)(l^0_B - l^0_A)\Delta l^h && l_{ij} > l^0 + \Delta l^h\end{aligned} \tag{14.117}$$

The forces are calculated similar as for the non-perturbed DF restraint:

$$\mathbf{f}_j(\lambda) = 2^{n+m}\lambda^n(1-\lambda)^m\mathbf{f}^{AB}_j(\lambda) \tag{14.118}$$

and $\mathbf{f}^{AB}_j(\lambda)$ is calculated using Eq. 9.140, replacing $k^{(df)}$ with $k^{(df)}_{AB}(\lambda)$ and l^0 with $l^0_{AB}(\lambda)$.

The current implementation allows for applications of DF in normal MD simulations, in free energy calculations or in combination with enhanced sampling methods like local elevation (LE, see Sec. 9.13.1) or Hamiltonian replica exchange (REMD, see Sec. 16.3).

14.3. Constraints

When constraints are applied in a simulation (Chap. 10), these appear formally as parameters in the Hamiltonian[130,133]. If the distance constraint Sec. 10.3 is dependent on the coupling parameter λ,

$$\sigma_k(\mathbf{r};\lambda) \equiv \mathbf{r}^2_{k_1k_2} - {d^0}^2_{k_1k_2}(\lambda) = 0 \qquad with\ k = 1,2,...,N^{(c)} \tag{14.119}$$

the constraint forces at time t (Sec. 10.3.8)

$$\begin{aligned} {\mathbf{f}^{(c)}}_i(t) &= -\sum_{k=1}^{N^{(c)}} l_k(\lambda;t)\frac{\partial\sigma_k(\mathbf{r}(t);\lambda)}{\partial\mathbf{r}_i(t)} \\ &= -2\sum_{k=1}^{N^{(c)}} l_k(\lambda;t)(\delta_{ik_1}-\delta_{ik_2})\mathbf{r}_{k_1k_2}(t) \end{aligned} \tag{14.120}$$

and the Lagrange multipliers $l_k(\lambda;t)$ will also depend on λ. The $\lambda-$dependent version of the distance constraints is taken linear in the distance constraint lengths ${d^0}_{k_1k_2}$,

$${d^0}_{k_1k_2}(\lambda) = (1-\lambda){d^0}^A_{k_1k_2} + \lambda {d^0}^B_{k_1k_2}. \tag{14.121}$$

Following the derivation in Sec. 10.3 the set of $N^{(c)}$ quadratic equations from which the $N^{(c)}$ multipliers $l_k(\lambda;t)$ are to be solved and used to obtain the constrained positions $\mathbf{r}(t+\Delta t)$ through Eq. 10.7 and Eq. 10.8 become also $\lambda-$dependent,

$$\begin{aligned} &\big[\ {\mathbf{r}^{uc}}_{k_1k_2}(t+\Delta t) - 2l_k(\lambda;t)(m_{k_1}^{-1}+m_{k_2}^{-1})\mathbf{r}_{k_1k_2}(t)\cdot(\Delta t)^2\big]^2 \\ &\quad -\big[\ (1-\lambda){d^0}^A_{k_1k_2} + \lambda{d^0}^B_{k_1k_2}\big]^2 = 0 \qquad with\ k = 1,2,...,N^{(c)} \end{aligned} \tag{14.122}$$

After linearization of Eq. 14.122 the $\lambda-$dependent expressions for the Lagrange multipliers and the corrections to the unconstrained positions become

$$l_k(\lambda;t) = \frac{((1-\lambda){d^0}^A_{k_1k_2} + \lambda{d^0}^B_{k_1k_2})^2 - ({\mathbf{r}^{uc}}_{k_1k_2}(t+\Delta t))^2}{-4(\Delta t)^2(m_{k_1}^{-1}+m_{k_2}^{-1})(\mathbf{r}_{k_1k_2}(t)\cdot{\mathbf{r}^{uc}}_{k_1k_2}(t+\Delta t))}. \tag{14.123}$$

$$\begin{aligned} \Delta{\mathbf{r}^{uc}}_{k_1}(t+\Delta t) &= -2(\Delta t)^2 m_{k_1}^{-1} l_k(\lambda;t)\mathbf{r}_{k_1k_2}(t) \\ &= m_{k_1}^{-1} g_k(\lambda;t)\mathbf{r}_{k_1k_2}(t) \end{aligned} \tag{14.124}$$

and

$$\begin{aligned} \Delta{\mathbf{r}^{uc}}_{k_2}(t+\Delta t) &= +2(\Delta t)^2 m_{k_2}^{-1} l_k(\lambda;t)\mathbf{r}_{k_1k_2}(t) \\ &= -m_{k_2}^{-1} g_k(\lambda;t)\mathbf{r}_{k_1k_2}(t). \end{aligned} \tag{14.125}$$

Since the atoms may be involved in more than one constraint, the set of equations Eq. 14.122 is solved in linearized form iteratively by the procedure SHAKE, that is, SHAKE as discussed in Sec. 10.3.1 changes the atomic positions $\mathbf{r}^{uc}(t+\Delta t)$ iteratively, such that at the end of the iterative process they satisfy the constraints, and can be returned as $\mathbf{r}(t+\Delta t)$. The number of iterations N_{sh} is determined as discussed in Sec. 10.3. At the n-th iteration step, for each constraint k = 1, 2, ..., $N^{(c)}$ the quantities

$$g_k(\lambda;t;n) = -2(\Delta t)^2 l_k(\lambda;t;n) \tag{14.126}$$

are calculated from Eq. 14.123 and the atomic positions $\mathbf{r}^{uc}(t+\Delta t)$ changed by applying the current (n-th) approximation of the constraint force Eq. 14.120

$${\mathbf{r}^{uc}}_{k_2}(t+\Delta t;n) = {\mathbf{r}^{uc}}_{k_2}(t+\Delta t;n-1) - m_{k_2}^{-1} g_k(\lambda;t;n)\mathbf{r}_{k_1k_2}(t). \tag{14.127}$$

$${\mathbf{r}^{uc}}_{k_1}(t+\Delta t;n) = {\mathbf{r}^{uc}}_{k_1}(t+\Delta t;n-1) + m_{k_1}^{-1} g_k(\lambda;t;n)\mathbf{r}_{k_1k_2}(t) \tag{14.128}$$

The Lagrange multipliers at time t become

$$l_k(\lambda;t) = \sum_{n=1}^{N_{sh}} l_k(\lambda;t;n) \tag{14.129}$$

or

$$g_k(\lambda;t) = \sum_{n=1}^{N_{sh}} g_k(\lambda;t;n). \tag{14.130}$$

The contribution of the constraints Eq. 14.119 to the derivative of the free energy $\mathcal{F}(\lambda)$ with respect to λ at time t is[130]

$$\begin{aligned} \frac{\partial \mathcal{F}^{(c)}(\lambda;t)}{\partial \lambda} &= \frac{\partial}{\partial \lambda} \sum_{k=1}^{N^{(c)}} l_k(\lambda;t)\sigma_k(\mathbf{r}(t);\lambda) \\ &= \sum_{k=1}^{N^{(c)}} l_k(\lambda;t) \frac{\partial \sigma_k(\mathbf{r}(t);\lambda)}{\partial \lambda} \\ &= \sum_{k=1}^{N^{(c)}} \frac{\partial \mathcal{F}_k^{(c)}(\lambda;t)}{\partial \lambda} \end{aligned} \tag{14.131}$$

in which the contribution of the k-th constraint is

$$\begin{aligned} \frac{\partial \mathcal{F}_k^{(c)}(\lambda;t)}{\partial \lambda} &= -2l_k(\lambda;t)d^0{}_{k_1k_2}(\lambda)(d^{0B}{}_{k_1k_2} - d^{0A}{}_{k_1k_2}) \\ &= (\Delta t)^{-2} g_k(\lambda;t)[(1-\lambda)d^{0A}{}_{k_1k_2} + \lambda d^{0B}{}_{k_1k_2}](d^{0B}{}_{k_1k_2} - d^{0A}{}_{k_1k_2}). \end{aligned} \tag{14.132}$$

The corresponding expression for use in the perturbation formula Eq. 14.9 is, assuming that $\mathbf{r}$ corresponds to λ,

$$\begin{aligned} &\sum_{k=1}^{N^{(c)}} l_k(\lambda)[\sigma_k(\mathbf{r};\lambda \pm \Delta\lambda) - \sigma_k(\mathbf{r};\lambda)] \\ &= -\sum_{k=1}^{N^{(c)}} l_k(\lambda)[d^{02}{}_{k_1k_2}(\lambda \pm \Delta\lambda) - d^{02}{}_{k_1k_2}(\lambda)] \\ &= -\sum_{k=1}^{N^{(c)}} l_k(\lambda)\Delta\lambda(d^{0B}{}_{k_1k_2} - d^{0A}{}_{k_1k_2}) \cdot \left[\ \Delta\lambda(d^{0B}{}_{k_1k_2} - d^{0A}{}_{k_1k_2}) \pm 2(d^{0A}{}_{k_1k_2} + \lambda(d^{0B}{}_{k_1k_2} - d^{0A}{}_{k_1k_2}))\right] \\ &= \sum_{k=1}^{N^{(c)}} \tfrac{1}{2}(\Delta t)^{-2} g_k(\lambda) \left[\ d^{02}{}_{k_1k_2}(\lambda \pm \Delta\lambda) - d^{02}{}_{k_1k_2}(\lambda)\right]. \end{aligned} \tag{14.133}$$

14.4. Assigning different $\lambda-$dependences for specific groups of atoms

The Hamiltonian, as described in Sec. 14.1 and Sec. 14.2 was made dependent on a global coupling parameter λ, for which the value is set via the RLAM variable in the PERTURBATION block of the MD input file (Chap. 4-8). MD++ additionally offers the possibility to introduce (via the LAMBDAS block in the MD input file, Chap. 4-8) for specific types of interactions that involve atoms of two energy groups i and j a coupling parameter Λ_{ij}, which is related to the global coupling parameter λ via the polynomial

$$\Lambda_{ij} = a_{ij}\lambda^4 + b_{ij}\lambda^3 + c_{ij}\lambda^2 + d_{ij}\lambda + e_{ij}, \tag{14.134}$$

where the polynomial coefficients $a_{ij}, b_{ij}, c_{ij}, d_{ij}$ and e_{ij} are constants and specified by the user in the LAMBDAS block as well. When specifying a coupling parameter Λ_{ij} involving covalent, special interactions or masses, i should be equal to j, and covalent or special interactions are only perturbed using Λ_{ii} when

the first atom given in the topology (or restraining) file to specify the atom-atom distance or covalent bond, angle or (im)proper dihedral, is part of energy group i. Λ_{ij} can only be specified for covalent interactions listed in the perturbation topology, or for masses and nonbonded interaction involving atoms listed in the perturbation topology.

For any perturbed interaction that is not specified the LAMBDAS block of the MD input file, a_{ij}, b_{ij}, c_{ij} and e_{ij} in Eq. 14.134 can be considered to have a zero value, and $d_{ij} = 1$ in Eq. 14.134.

The use of Λ_{ij} for perturbing specific interactions implies the following for the expressions for the perturbed Hamiltonian. λ in the expressions for the atomic masses, kinetic energy and covalent interactions and forces as given in Sec. 14.2 is replaced by $\Lambda_{ij}(=\Lambda_{ii})$ and the right-hand side of the equations Eq. 14.17, Eq. 14.22, Eq. 14.27, Eq. 14.40, Eq. 14.46, Eq. 14.56, Eq. 14.66 and Eq. 14.73 for the λ-derivatives of the kinetic and covalent interaction energy terms (in which λ is to be changed with $\Lambda_{ij}(=\Lambda_{ii})$ as well) have to be multiplied by $\frac{\partial \Lambda_{ij}}{\partial \lambda}$. From equation Eq. 14.134

$$\frac{\partial \Lambda_{ij}}{\partial \lambda} = 4a_{ij}\lambda^3 + 3b_{ij}\lambda^2 + 2c_{ij}\lambda + d_{ij} \tag{14.135}$$

Expressions Eq. 14.75 and Eq. 14.76 for the perturbed non-bonded interaction is to be replaced by

$$\begin{aligned}
&\mathcal{V}^{(nbd)}(\mathbf{r};\lambda) = \\
&\sum_{\substack{nonbonded\\ perturbed\\ pairs(i,j)}} \Big\{ \Big[(\Lambda_{ij}^{(vdw)})^n \mathcal{V}^{(vdw)}(\mathbf{r}_{ij}; B; (1-\Lambda_{ij}^{(sl)})) + (1-\Lambda_{ij}^{(vdw)})^n \mathcal{V}^{(vdw)}(\mathbf{r}_{ij}; A; \Lambda_{ij}^{(sl)}) \Big] \\
&+ [((\Lambda_{ij}^{(ele)})^n \mathcal{V}^{(ele)}(\mathbf{r}_{ij}; B; (1-\Lambda_{ij}^{(sc)}))) + (1-\Lambda_{ij}^{(ele)})^n \mathcal{V}^{(ele)}(\mathbf{r}_{ij}; A; (1-\Lambda_{ij}^{(sc)}))\Big] \Big\}
\end{aligned} \tag{14.136}$$

where four different interaction-specific coupling parameters Λ_{ij} can be specified:

- $\Lambda_{ij}^{(vdw)}$ scales the Lennard-Jones interactions
- $\Lambda_{ij}^{(sl)}$ determines the softness in the Lennard-Jones interactions
- $\Lambda_{ij}^{(ele)}$ scales the electrostatic interactions
- $\Lambda_{ij}^{(sc)}$ determines the softness in the electrostatic interactions.

In equation Eq. 14.136, $\mathcal{V}^{(vdw)}$ and $\mathcal{V}^{(ele)}$ for state X reads as:

$$\begin{aligned}
&\mathcal{V}^{(vdw)}(\mathbf{r}_{ij}; X; \Lambda_{ij}^{(sl)}) = \frac{1}{[\alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2 C_{126}^X(i,j) + (r_{ij})^6]} \\
&\cdot \left[\frac{{C_{12}}^X(i,j)}{[\alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2 C_{126}^X(i,j) + (r_{ij})^6]} - {C_6}^X(i,j) \right]
\end{aligned} \tag{14.137}$$

and

$$\mathcal{V}^{(ele)}(\mathbf{r}_{ij};X;\Lambda_{ij}^{(sc)}) =$$

$$\frac{q_i^X q_j^X}{4\pi\varepsilon_0\varepsilon_1}\left[\frac{1}{[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})^2+(r_{ij})^2]} - \frac{\frac{1}{2}C_{rf}(r_{ij})^2}{[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})+(R_{rf})^2]^{\frac{3}{2}}} - \frac{(1-\frac{1}{2}C_{rf})}{R_{rf}}\right] \tag{14.138}$$

The forces on atoms i and j due to the (i,j)-th term in Eq. 14.136 are

$$\begin{aligned}\mathbf{f}^{(nbd)}{}_i = (\Lambda_{ij}^{(vdw)})^n\ \frac{\partial\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sl)}))}{\partial\mathbf{r}_i} + (1-\Lambda_{ij}^{(vdw)})^n\frac{\partial\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};A;\Lambda_{ij}^{(sl)})}{\partial\mathbf{r}_i}\\ +(\Lambda_{ij}^{(ele)})^n\frac{\partial\mathcal{V}^{(ele)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sc)}))}{\partial\mathbf{r}_i} + (1-\Lambda_{ij}^{(ele)})^n\frac{\partial\mathcal{V}^{(ele)}(\mathbf{r}_{ij};A;(1-\Lambda_{ij}^{(sc)}))}{\partial\mathbf{r}_i}\end{aligned} \tag{14.139}$$

and

$$\mathbf{f}^{(nbd)}{}_j = -\mathbf{f}^{(nbd)}{}_i \tag{14.140}$$

with

$$\begin{aligned}\frac{\partial\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};X;\lambda)}{\partial\mathbf{r}_i} &= \frac{-6(r_{ij})^4}{[\alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2C_{126}^X(i,j)+(r_{ij})^6]^2}\\ &\cdot\left[\frac{C_{12}{}^X(i,j)}{[\alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2C_{126}^X(i,j)+(r_{ij})^6]} - C_6{}^X(i,j)\right]\cdot\mathbf{r}_{ij}\end{aligned} \tag{14.141}$$

and

$$\frac{\partial\mathcal{V}^{(ele)}(r_{ij};X;\lambda)}{\partial\mathbf{r}_i} = \frac{q_i^X q_j^X}{4\pi\varepsilon_0\varepsilon_1}\left[\frac{1}{[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})^2+(r_{ij})^2]^{\frac{3}{2}}} + \frac{C_{rf}}{[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})^2+(R_{rf})^2]^{\frac{3}{2}}}\right]\cdot\mathbf{r}_{ij} \tag{14.142}$$

The $\lambda-$derivative of the right-hand side of equation Eq. 14.136 reads

$$\begin{aligned}\frac{\partial\mathcal{V}^{(nbd)}(\mathbf{r};\lambda)}{\partial\lambda} = \{&n(\Lambda_{ij}^{(vdw)})^{n-1}\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sl)}))\frac{\partial\Lambda_{ij}^{(vdw)}}{\partial\lambda}\\ &+(\Lambda_{ij}^{(vdw)})^n\frac{\partial\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sl)}))}{\partial\Lambda_{ij}^{(sl)}}\frac{\partial\Lambda_{ij}^{(sl)}}{\partial\lambda} - n(1-\Lambda_{ij}^{(vdw)})^n\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};A;\Lambda_{ij}^{(sl)})\frac{\partial\Lambda_{ij}^{(vdw)}}{\partial\lambda}\\ &+(1-\Lambda_{ij}^{(vdw)})^n\frac{\partial\mathcal{V}^{(vdw)}(\mathbf{r}_{ij};A;\Lambda_{ij}^{(sl)})}{\partial\Lambda_{ij}^{(sl)}}\frac{\partial\Lambda_{ij}^{(sl)}}{\partial\lambda} + n(\Lambda_{ij}^{(ele)})^{n-1}\mathcal{V}^{(ele)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sc)}))\frac{\partial\Lambda_{ij}^{(ele)}}{\partial\lambda}\\ &+(\Lambda_{ij}^{(ele)})^n\frac{\partial\mathcal{V}^{(ele)}(\mathbf{r}_{ij};B;(1-\Lambda_{ij}^{(sc)}))}{\partial\Lambda_{ij}^{(sc)}}\frac{\partial\Lambda_{ij}^{(sc)}}{\partial\lambda} - n(1-\Lambda_{ij}^{(ele)})^{n-1}\mathcal{V}^{(ele)}(\mathbf{r}_{ij};A;\Lambda_{ij}^{(sc)})\frac{\partial\Lambda_{ij}^{(ele)}}{\partial\lambda}\\ &+(1-\Lambda_{ij}^{(ele)})^n\frac{\partial\mathcal{V}^{(ele)}(\mathbf{r}_{ij};A;\Lambda_{ij}^{(sc)})}{\partial\Lambda_{ij}^{(sc)}}\frac{\partial\Lambda_{ij}^{(sc)}}{\partial\lambda}\end{aligned} \tag{14.143}$$

In Eq. 14.143, the $\frac{\partial \Lambda_{ij}^{X}}{\partial \lambda}$ terms are evaluated according to equation Eq. 14.135, and

$$\frac{\partial \mathcal{V}^{(vdw)}(\mathbf{r}_{ij};X;\Lambda_{ij}^{(sl)})}{\partial \Lambda_{ij}^{(sl)}} = \frac{-2\Lambda_{ij}^{(sl)}\alpha_{LJ}(i,j)C_{126}^{X}(i,j)}{\left[\ \alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2 C_{126}^{X}(i,j)+(r_{ij})^6\right]^2}\left[\frac{2{C_{12}}^{X}(i,j)}{\left[\alpha_{LJ}(i,j)(\Lambda_{ij}^{(sl)})^2 C_{126}^{X}(i,j)+(r_{ij})^6\right]} - {C_6}^{X}(i,j)\right] \tag{14.144}$$

$$\frac{\partial \mathcal{V}^{(ele)}(\mathbf{r}_{ij};X;\Lambda_{ij}^{(sc)})}{\partial \Lambda_{ij}^{(sc)}} = -\frac{q_i^X q_j^X}{4\pi\varepsilon_0\varepsilon_1}\cdot\Lambda_{ij}^{(sc)}\alpha_c(i,j) \cdot\left[\frac{1}{\left[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})^2+(r_{ij})^2\right]^{\frac{3}{2}}} - \frac{\frac{3}{2}C_{rf}(r_{ij})^2}{\left[\alpha_c(i,j)(\Lambda_{ij}^{(sc)})^2+(R_{rf})^2\right]^{\frac{5}{2}}}\right] \tag{14.145}$$

and

$$\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r};X;(1-\Lambda_{ij}^{(sl/sc)}))}{\partial \Lambda_{ij}^{(sl/sc)}} = -\frac{\partial \mathcal{V}^{(nbd)}(\mathbf{r};X;\Lambda_{ij}^{(sl/sc)})}{\partial \Lambda_{ij}^{(sl/sc)}} \tag{14.146}$$

14.5. Choice of pathway and states A and B

Technical aspects of specifying the states A and B are discussed in Sec. 4-3.3. The *molecular topology*, as described in Sec. 4-3.2, the so-called unperturbed topology, corresponds to *state A*. Since atoms cannot be created or destroyed, only their interaction with other atoms can be modified or perturbed, the unperturbed topology corresponding to state A must contain all atoms involved in the perturbation as either real or dummy (i.e. non-interacting) atoms. So, state B has the same number of atoms as state A. The *difference between state B and A* is specified in a *perturbation molecular topology* (Sec. 4-3.3). In Sec. 4-3.3 *eight points of interest* are listed, which should be kept in mind when constructing both, molecular and perturbation, topology files for use in a free energy calculation.

General considerations and *indications* with respect to the *choice of pathways* in free energy calculations have been discussed in refs.[130,131,135]. Here, we shall only list the most important issues.

1. *Choice of thermodynamic cycle and its four end states.*
   In most free energy calculations the concept of a thermodynamic cycle is used. Since the free energy $\mathcal{F}$ is a state function, a change in free energy, $\Delta\mathcal{F}$, will be independent of the path connecting the end states, as long as the system is in equilibrium and is changed in a reversible way. In that case, along a closed pathway or cycle we have $\Delta\mathcal{F} = 0$. This result implies that there are many possibilities of obtaining $\Delta\mathcal{F}$ between two end states A and B. One may calculate $\Delta\mathcal{F}$ directly using a pathway connecting states A and B, or one may design a cycle of which a pathway from state A to state B is only one part, and calculate the $\Delta\mathcal{F}$ of the remaining part of the cycle. The power of the thermodynamic cycle technique resides in the fact that *pathways* and *cycles* may correspond to *non-chemical processes*, and so may be chosen such as to optimize the accuracy of the calculation. Generally, equilibration and sampling are optimized by *keeping the differences between states A and B minimal*. Secondly, the *end states A and B* should be *structurally well characterized* in order to avoid excessively long equilibration and sampling periods[130]. The end states and the connecting pathway should be chosen such that the relaxation time of the system with respect to the change in Hamiltonian and the time required to sample the ensemble are both minimized. This implies that the most direct or chemical path is not necessarily the most efficient[136,137]. As only terms in the Hamiltonian that are modified, *i.e.* depend on λ, contribute to the change in free energy, the number of λ dependences should be minimized. The introduction of additional or removal of degrees of freedom should be avoided: generally, when changing a real atom into a dummy atom or vice versa, only non-bonded interactions should be modified, bonded terms such as bond-length, bond-angle and dihedral-angle terms should remain unchanged. This avoids the additional work required to modify the associated force constants and the additional sampling required as the atom

gains degrees of freedom. Perturbations should be defined such that an atom does not become fully uncoupled from the rest of the system. If possible, the mass of an atom should remain unperturbed.

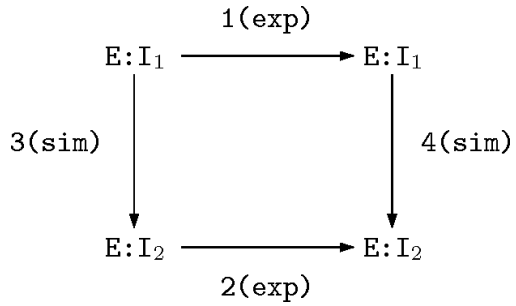


FIGURE 14.1. Thermodynamic cycle with respect to the relative binding of two inhibitors I_1 and I_2 to an enzyme E. The symbol ':' indicates complexation.[138]

2. *Choice of boundary conditions.*
   Thermodynamic cycles allow for the cancellation of systematic errors occurring in parallel legs of a cycle. For example, errors in processes 3 and 4 in Fig. 14.1 may cancel when $\Delta\Delta\mathcal{F}_{43} = \Delta\mathcal{F}_4 - \Delta\mathcal{F}_3$ is calculated. However, cancellation may only happen when the *simulation conditions* along the *parallel legs* are chosen *as similar as possible*: the simulations along legs 3 and 4 should not involve different box shapes, different spatial boundary conditions (e.g. periodic versus extended wall) or different thermodynamic boundary conditions (e.g. (N,V,T) versus (N,P,T)).

3. *Absence and presence of force field terms.*
   In a thermodynamic cycle the contribution of given force-field terms may be equal along parallel legs, and so might be ignored. Great care is required, however, when selecting force-field or Hamiltonian terms to be neglected. For example, the work done increasing or decreasing a charge during a simulation will depend on the dielectric permittivity of the medium. Long-range electrostatic contributions can only be ignored if the dielectric properties of the environments on both legs of the thermodynamic cycle are equivalent. This is not the case when one leg corresponds to the low dielectric interior of a protein or non-polar solvent and the other to a high dielectric medium such as water[139]. Whether bonded force terms can be ignored is also questionable: if the conformational freedom of (part of) the molecular system is different along parallel legs of a cycle, bonded force terms will make a different contribution along the different legs which must be explicitly calculated. Finally, we note that the *terms in the Hamiltonian* Eq. 14.14 *that do not depend on* λ *should be kept identical along (parallel) legs of a cycle*: position restraining, atom-atom distance restraining, dihedral-angle restraining, ^3J-coupling constant restraining interactions, etc. should not be different along different legs of a cycle. Switches for force field terms, such as NTF[1..10] (Sec. 12.7), should also not be different. Local-elevation interaction and time-averaging distance or ^3J-value restraining or S^2-order parameter restraining should not be used at all, since they involve changing the Hamiltonian as a function of time.

4. *Choice of atomic masses.*
   Due to the factor m_i^{-1} in the equations of motion Eq. 2.8 and Eq. 2.13, the mass of an atom should not be chosen equal to zero.
5. *Atoms without non-bonded interactions.*
   The van der Waals and Coulomb potential energy terms contain a singularity when the interatomic distance r becomes zero. For fully interacting atom pairs this singularity is never sampled in a simulation due to the repulsive term in the van der Waals interaction. This singularity in $\mathcal{V}^{(nbd)}$, $\frac{\partial\mathcal{V}^{(nbd)}}{\partial\mathbf{r}}$ or $\frac{\partial\mathcal{V}^{(nbd)}}{\partial\lambda}$ will be sampled if the non-bonded interaction is defined to be zero in state A or state B (presence of dummy atoms), leading to numerical instabilities in the simulation[26]. When the effective non-bonded interaction radius of an atom becomes smaller than the length of flight of the atom per (finite) MD time step, very high energy regions of the potential energy will be sampled. To avoid such problems, a soft-core interaction should be selected for atoms without non-bonded interaction (i.e. dummy atoms) in one of the end states B and A[26].

14.6. Thermodynamic integration

In the early applications of the thermodynamic integration (TI) Eq. 14.11 the coupling parameter λ was made a function of time t, $\lambda(t)$, and the integral was integrated in small time steps Δt during an MD simulation of $\mathcal{N}_t$ steps

$$\Delta\mathcal{F}_{BA} = \sum_{n=1}^{\mathcal{N}_t} \frac{\partial\mathcal{H}}{\partial\lambda}\frac{\partial\lambda}{\partial t}\Delta t. \tag{14.147}$$

This technique is called continuous change of λ, or *slow growth* or single-configuration TI. It suffers from a number of disadvantages[135]. The system is never truly in equilibrium, but lags behind the changing Hamiltonian. This results in excess work being done on the system and a systematic overestimation of the free energy change. Attempting to correct for this overestimation by averaging results for the forward and reverse change from short simulations is an unreliable procedure. A necessary but not sufficient condition to obtain a reliable free energy change is that the difference between forward and reverse processes or hysteresis is small. A small hysteresis, however, indicates only the degree of reversibility. It indicates neither that the system is in equilibrium nor that a representative ensemble has been sampled for each value of λ. If a change of $\mathcal{H}(\lambda)$ is carried out much faster than the system can respond, the system will remain trapped in a (non-representative) local state. If this state is adequately sampled during the simulation, the change will appear reversible and a small hysteresis will be observed. The calculated free energy change will, however, be dependent on the precise starting configuration and as the length of the simulation is increased, the apparent hysteresis will also increase[140]. In practice, slow growth procedures should only be used to bring the system gradually from one $\lambda-$value to another.

An alternative use of the thermodynamic integration formula Eq. 14.11 is to compute ensemble averages at a few, $\mathcal{N}_\lambda$, fixed $\lambda-$values and perform the integration numerically using interpolation formulae for the integrand,

$$\Delta\mathcal{F}_{BA} = \sum_{n=1}^{\mathcal{N}_\lambda} \left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n} w(\lambda_n), \tag{14.148}$$

where the weight factors of the numerical integration formula that is used are denoted by $w(\lambda_n)$. This technique is called TI *with numerical quadrature* or multi-configurational TI. It has a number of practical advantages. Effects due to the equilibration of the system with respect to the change in Hamiltonian and those depending on the extent of sampling can be largely separated. The convergence of the ensemble average $\langle ... \rangle_{\lambda_n}$ at fixed λ_n-value can be monitored as a function of the simulation time. If the integrand turns out to vary rapidly as a function of λ, more intermediate λ-values can be added into the numerical integration[140]. Simple numerical integration methods such as the trapezoidal rule, Simpson's rule or cubic spline integration may be used to integrate over λ.

To estimate the *precision of* $\langle \partial\mathcal{H}/\partial\lambda \rangle_{\lambda_n}$, the following formula can be used for the standard deviation on the mean[139],

$$\sigma\left(\left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n} \right) = \left(\frac{S_{\lambda_n}}{\mathcal{N}_{conf}} \right)^{\frac{1}{2}} \sigma\left(\frac{\partial\mathcal{H}}{\partial\lambda} \right), \tag{14.149}$$

where $\mathcal{N}_{conf}$ denotes the number of configurations over which the ensemble average at $\lambda = \lambda_n$ is taken, and $\sigma(\partial\mathcal{H}/\partial\lambda)$ is calculated as the square root of the variance over the ensemble

$$\sigma^2\left(\frac{\partial\mathcal{H}}{\partial\lambda} \right) = \frac{1}{\mathcal{N}_{conf}} \sum_{t=1}^{\mathcal{N}_{conf}} \left[\frac{\partial\mathcal{H}}{\partial\lambda} - \left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n,\mathcal{N}_{conf}} \right]^2. \tag{14.150}$$

To calculate the statistical inefficiency, S_{λ_n}, the total simulation time at $\lambda = \lambda_n$ is divided into M blocks of length b, and $\mathcal{N}_b$ configurations are taken from each block such that $M \cdot \mathcal{N}_b = \mathcal{N}_{conf}$.

The variance in the mean is then calculated for each possible block length b as

$$\sigma^2\left(\left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n,b} \right) = \frac{1}{M} \sum_{m=1}^{M} \left[\left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n,\mathcal{N}_b,m} - \left\langle \frac{\partial\mathcal{H}}{\partial\lambda} \right\rangle_{\lambda_n,\mathcal{N}_{conf}} \right]^2, \tag{14.151}$$

where $< ... >_{\lambda_n,\mathcal{N}_b,m}$ denotes the ensemble average at $\lambda = \lambda_n$ over $\mathcal{N}_b$ sample configurations of the m-th block. S_{λ_n} is then calculated as

$$S_{\lambda_n} = \lim_{\mathcal{N}_b \to \infty} \frac{\mathcal{N}_b \sigma^2 \left(\left\langle \frac{\partial \mathcal{H}}{\partial \lambda} \right\rangle_{\lambda_n, b} \right)}{\sigma^2 \left(\frac{\partial \mathcal{H}}{\partial \lambda} \right)}. \tag{14.152}$$

Effectively, only one configuration of every S_{λ_n} configurations used contributes new information to the average. The simulation periods needed to obtain a given degree of convergence of $\langle \partial \mathcal{H}/\partial \lambda \rangle_{\lambda_n}$ may be quite different for different λ_n-values[139, 141].

To estimate the *precision of the numerical quadrature*, one may assume that the integrand is Gaussian distributed with a mean $\langle \partial \mathcal{H}/\partial \lambda \rangle_{\lambda_n}$ and a width given by Eq. 14.149. Assuming no correlation between the distributions at different λ_n points, the standard deviation on $\Delta \mathcal{F}_{BA}$ becomes[139]

$$\sigma(\Delta \mathcal{F}_{BA}) = \left[\sum_{n=1}^{\mathcal{N}_\lambda} w(\lambda_n) \sigma^2 \left(\left\langle \frac{\partial \mathcal{H}}{\partial \lambda} \right\rangle_{\lambda_n} \right) \right]^{1/2}. \tag{14.153}$$

Additional tests of the reliability of the obtained free energy difference $\Delta \mathcal{F}_{BA}$ can and should be carried out[130].

1. The *addition of extra λ -values* in the numerical integration over λ in Eq. 14.147 should not dramatically change the $\Delta \mathcal{F}_{BA}$-value obtained so far[140].
2. When carrying out more than one change of a system, e.g. from state A to state B and from A to C, the quality of the equilibration, sampling and integration over λ can be tested by performing the change from state B to C, which *closes a cycle*[140]
$$\Delta \mathcal{F}_{BA} + \Delta \mathcal{F}_{CB} + \Delta \mathcal{F}_{AC} = 0 \tag{14.154}$$
3. Repetition of individual simulations with *different initial* equilibrium *configurations* or *velocities* should yield the same result.
4. *Small changes* in the *computational procedure* should not affect the obtained $\Delta \mathcal{F}_{BA}$-value[142].

14.7. Thermodynamic perturbation and extrapolation

The perturbation formulae Eq. 14.6, Eq. 14.8 and Eq. 14.9 can also be used to compute free energy differences. In the limit of infinite sampling or when the ensembles corresponding to $\mathcal{H}(\lambda)$ and $\mathcal{H}(\lambda \pm \Delta\lambda)$ overlap perfectly the perturbation formula Eq. 14.9 will yield the exact $\Delta \mathcal{F}_{\lambda \pm \Delta\lambda}$ value. In practice, these conditions are never fulfilled. If the ensembles corresponding to states A and B do not overlap closely, calculations based on the perturbation formula must be split into a number of steps between intermediate systems along the pathway connecting states A and B that are sufficiently similar to allow for the use of Eq. 14.9, and then $\Delta \mathcal{F}_{BA}$ is just the sum of the $\Delta \mathcal{F}$ values for all intermediate steps,

$$\begin{aligned} \Delta \mathcal{F}_{BA} &= \sum_{n=0}^{\mathcal{N}_\lambda - 1} -k_B T \ln \left\langle e^{-[\mathcal{H}(\lambda_{n+1}) - \mathcal{H}(\lambda_n)]/k_B T} \right\rangle_{\lambda_n} \\ &= \sum_{n=1}^{\mathcal{N}_\lambda} +k_B T \ln \left\langle e^{-[\mathcal{H}(\lambda_{n-1}) - \mathcal{H}(\lambda_n)]/k_B T} \right\rangle_{\lambda_n}. \end{aligned} \tag{14.155}$$

We note that the sampling and convergence properties of the thermodynamic integration formula Eq. 14.147 and the perturbation formula Eq. 14.155 are different.[143] The requirement of closely overlapping ensembles for neighbouring λ-values in Eq. 14.155 does not apply to the TI formula Eq. 14.147, since the latter is based on the assumption of the smoothness of $\mathcal{F}'(\lambda)$, which is different from the assumption of overlapping ensembles for λ and $\lambda \pm \Delta\lambda$[135]. For the TI formula the convergence of the ensemble average does not depend on the magnitude of the change $\Delta\lambda$ in λ, as it does for the perturbation formula. Therefore, the TI formula offers the better opportunity to reduce and monitor errors in practice.[139]

Free energy calculations based on Eq. 14.147 or Eq. 14.155 are computationally very expensive. If one would like to obtain the relative free energy differences of a number (M) of end states B_1, B_2, ..., B_M with respect to state A, $M \cdot \mathcal{N}_\lambda$ converged simulations, at the various $\lambda-$points along the different pathways from A to B_1, B_2, ..., B_M, are required. This number could be reduced to 1 if one could use the perturbation formula Eq. 14.6 directly, that is, without intermediate λ-values between states A and B_m. One would use state A as reference state, the ensemble of which is used to *extrapolate the behaviour of* $\Delta \mathcal{F}_{\lambda_A + \Delta\lambda}$ *to* $\Delta\lambda = \lambda_{B_m} - \lambda_A$, *i.e.* state B_m. This approach has two advantages.

1. Only a single reference state A ($\lambda = \lambda_A$) need be considered and simulated.
2. The fluctuations in this reference state A are only dependent on λ_A not on the λ_{B_m}, that is, not on the λ changes to be considered.

The question is, however, whether the changes in free energy, $\Delta\mathcal{F}_{B_m A}$ can be accurately estimated for physically relevant states A and B_m. This is the case when the following two concepts are used[143,144] (see Eq. 14.155).

1. Eq. 14.6 gives incorrect results, if the configurations sampled in the reference state A ($\lambda = \lambda_A$) do not correspond to low energy configurations in the end states B_m. This is especially the case when real atoms are changed into dummy atoms or vice versa. A remedy is *to introduce soft-core non-bonded interactions* such as Eq. 14.75 - Eq. 14.77 in state A at positions where real atoms are to be changed into dummy atoms or vice versa. In this way the sampling of this new *reference state* A' is biased such that it encompasses the parts of configuration space accessible to the system in state A *and* in the end states B_m. So, a *non-physical state* A' is simulated which is chosen such that the accuracy of the free energy differences $\Delta\mathcal{F}_{AA'}$ and $\Delta\mathcal{F}_{B_m A'}$ (m = 1,2, ..., M) calculated using Eq. 14.6 is optimized.
2. The *difference* in free energy between the various *physical states* A and B_m can then be determined as

$$\Delta\mathcal{F}_{B_m A} = \Delta\mathcal{F}_{B_m A'} - \Delta\mathcal{F}_{A' A} \qquad with\ m = 1, 2, ..., M. \tag{14.156}$$

Use of the perturbation formula in combination with soft-core non-bonded interaction sites and a non-physical reference state that is simulated makes estimation of a series of free energy differences based on a single simulation possible.

The *energetic contribution* $\Delta\mathcal{U}_{BA}$ and the *entropic contribution* $T\cdot\Delta\mathcal{S}_{BA}$ to the total free energy difference $\Delta\mathcal{F}_{BA}$ can be obtained using the formulae[143,145]

$$\begin{aligned} \Delta\mathcal{U}_{BA} &= \langle\mathcal{H}(\lambda_B)\rangle_{\lambda_B} - \langle\mathcal{H}(\lambda_A)\rangle_{\lambda_A} \\ &= \left\langle\mathcal{H}(\lambda_B) e^{+[\Delta\mathcal{F}_{BA} - [\mathcal{H}(\lambda_B) - \mathcal{H}(\lambda_A)]]/k_B T}\right\rangle_{\lambda_A} - \langle\mathcal{H}(\lambda_A)\rangle_{\lambda_A} \end{aligned} \tag{14.157}$$

and

$$T\Delta\mathcal{S}_{BA} = \Delta\mathcal{U}_{BA} - \Delta\mathcal{F}_{BA}. \tag{14.158}$$

Program dg_ener can be used to obtain free energy differences $\Delta\mathcal{F}_{\lambda_A+\Delta\lambda}$ from the time series of the Hamiltonian in a reference state A′ (molecular topology) and in states A, B_m. It calculates a perturbation formula free energy difference Eq. 14.9. Because only the difference of the Hamiltonian is included in this equation, only those terms in the Hamiltonian that are different in states A, B_m from state A' need to be re-evaluated from a molecular trajectory (e.g. using program ener, or by performing an analysis running over an existing stimulation (block READTRAJ, see Chap. 4-8).

14.8. Umbrella sampling

The TI and perturbation formulae are very powerful when a change in free energy is associated with a change of chemical composition of the molecular system. In some cases, one wishes to consider the free energy as a function of a given geometrical parameter $\mathcal{Q}$, the reaction coordinate, e.g. the distance r_{ij} between reactants i and j in a chemical reaction or a torsional angle φ_n in case of a conformational change. The free energy as a function of $\mathcal{Q}$, $\mathcal{F}$ ($\mathcal{Q}$), is called a potential of mean force or a free energy profile. It is defined as[130]

$$\mathcal{F}(\mathcal{Q}') = -k_B T \ln\left[\frac{\int\int \delta(\mathcal{Q}(\boldsymbol{r}) - \mathcal{Q}') e^{-\frac{\mathcal{H}}{k_B T}} d\boldsymbol{p} d\boldsymbol{r}}{\int\int e^{-\frac{\mathcal{H}}{k_B T}} d\boldsymbol{p} d\boldsymbol{r}}\right] \tag{14.159}$$

$$= -k_B T \ln P(\mathcal{Q}') \tag{14.160}$$

where the function $\mathcal{Q}(\boldsymbol{r})$ defines the reaction coordinate in terms of the atomic coordinates $\boldsymbol{r}$, δ is the Dirac delta function and the probability to find the system at $\mathcal{Q} = \mathcal{Q}'$ is $P(\mathcal{Q}')$. If during a single simulation the sampling along the whole range of $\mathcal{Q}'$-values has been sufficient, $\mathcal{F}$ ($\mathcal{Q}'$) can be directly calculated from the probability distribution $P(\mathcal{Q}')$ as obtained from the trajectory. However, if there are free energy barriers

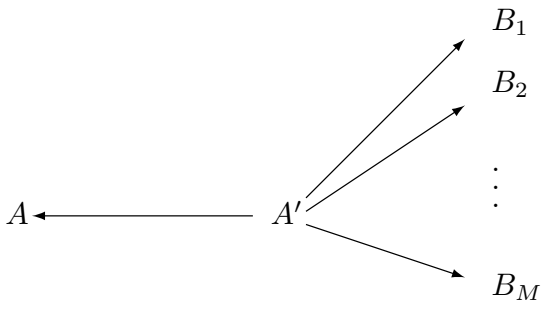


FIGURE 14.2. The free energy differences $\Delta\mathcal{F}_{B_m A}$ between the physical states A and B_1, B_2, ...B_m are obtained by using an ensemble for the non-physical state A′ ($\lambda = \lambda_{A'}$) which should contain soft-core non-bonded interaction sites for all atoms that have a different non-bonded interaction between states A and B_M ($m = 1, 2, ..., M$), and applying the perturbation formula between $\lambda = \lambda_{A'}$ and to $\lambda = \lambda_A$ and $\lambda = \lambda_{B_m}$ ($m = 1, 2, ..., M$), and Eq. 14.156.

much higher than k_BT along $\mathcal{Q}\,'$, these may inhibit a proper sampling along $\mathcal{Q}\,'$. In that case, the sampling may be improved by applying a biasing function.

In the so-called *umbrella-sampling technique* (US)[88] an auxiliary potential energy term $\mathcal{V}^{(spec)}(\mathcal{Q}(\boldsymbol{r}))$ is added to the Hamiltonian of the system, which *biases the sampling*. Simulation including the umbrella potential energy term then yields the biased probability distribution $P^{bias}(\mathcal{Q}')$, from which the unbiased probability distribution $P(\mathcal{Q}')$ can be obtained through the relation[130]

$$P(\mathcal{Q}') = P^{bias}(\mathcal{Q}')\frac{e^{+\frac{\mathcal{V}^{(spec)}(\mathcal{Q}')}{k_BT}}}{< e^{+\frac{\mathcal{V}^{(spec)}(\mathcal{Q}')}{k_BT}} >} \tag{14.161}$$

where $< ... >$ denotes an average over the simulations. The umbrella sampling technique can in principle be applied with any kind of biasing potential energy term $\mathcal{V}^{(spec)}(\mathcal{Q}'$, however mainly two types of umbrella potential energy functions are used.

1. Umbrella potential energy functions constructed to obtain uniform or quasi-uniform sampling. These functions typically will have to be created through an adaptive procedure, e.g. LE and LEUS. This is further explained in (Sec. 9.13.1).
2. Windowing umbrella potential energy functions, where the umbrella potential energy function is constructed to sample a specific range of $\mathcal{Q}$.

In the windowing umbrella sampling method a restraining potential energy function is added $\mathcal{V}^{(spec)} = \mathcal{V}^{(res)}(\mathcal{Q}_0)$ which *restrains the sampling* to within a given range of the reaction coordinate $\mathcal{Q}'$ around a given value $\mathcal{Q}' = \mathcal{Q}_0$. For the free energy profile $\mathcal{F}(\mathcal{Q}';\mathcal{Q}_0)$ around $\mathcal{Q}_0$ we then find

$$\begin{aligned}\mathcal{F}(\mathcal{Q}';\mathcal{Q}_0) &= -k_BT\ln P^{bias}(\mathcal{Q}';\mathcal{Q}_0) - \mathcal{V}^{(res)}(\mathcal{Q}';\mathcal{Q}_0) \\ &\quad +k_BT\ln < e^{+\frac{\mathcal{V}^{(res)}(\mathcal{Q};\mathcal{Q}_0)}{k_BT}} >_{\mathcal{Q}_0} -k_BT\ln\mathcal{Z} \\ &= -k_BT\ln P^{restr}(\mathcal{Q}';\mathcal{Q}_0) - \mathcal{V}^{(res)}(\mathcal{Q}';\mathcal{Q}_0) + C(\mathcal{Q}_0)\end{aligned} \tag{14.162}$$

where the (unknown) partition function of the molecular system is indicated by $\mathcal{Z}$. Since the last two terms in the first part of Eq. 14.162 do not depend on $\mathcal{Q}'$, they can be written as a constant depending only on $\mathcal{Q}_0$, $C(\mathcal{Q}_0)$. So using Eq. 14.162 the free energy profile within a given range around $\mathcal{Q}_0$ can be determined up to a constant. The overall free energy profile $\mathcal{F}(\mathcal{Q}')$ can then be obtained by a series of umbrella simulations restrained around the values $\mathcal{Q}_0, \mathcal{Q}_1, \mathcal{Q}_2, \ldots$ and by adding different constants $C(\mathcal{Q}_i)$ to the local free energy profiles $\mathcal{F}(\mathcal{Q}';\mathcal{Q}_i)$ so that the resulting free energy profile $\mathcal{F}(\mathcal{Q}')$ is a continuous function of the reaction coordinate $\mathcal{Q}'$[61,62,132,146].

The atom-atom *distance restraining* function Eq. 9.3, $\mathcal{V}^{(dr)}$, discussed in Sec. 9.3 or the *dihedral-angle restraining* function Eq. 9.60, $\mathcal{V}^{(tr)}$, discussed in Sec. 9.6 are suitable for use as umbrella potential energy terms. The force constants $k^{(pr)}$ in Eq. 9.4 or $k^{(tr)}$ in Eq. 9.60 should be chosen sufficiently large to focus the sampling around r_m or φ_n, but not too big in order to avoid too narrow probability distributions around the points $\mathcal{Q}_i = r_m$ or $\mathcal{Q}_i = \varphi_n$, in which case many simulations would be required to cover the whole range of $\mathcal{Q}$ -values.[61,132]

Alternatively, the free energy difference between two conformations, separated by high energy barriers may be elegantly calculated using hidden, perturbed distance restraints or hidden, perturbed dihedral angle restraints, as explained in Sec. 14.2.10 and Sec. 14.2.11.

14.9. Enveloping Distribution Sampling

Enveloping distribution sampling (EDS) is an alternative approach to calculating the free energy difference between various pairs of states A and B

$$\Delta\mathcal{F}_{BA} = \mathcal{F}_B - \mathcal{F}_A = -\beta^{-1} \ln\left(\frac{\mathcal{Z}_B}{\mathcal{Z}_A}\right). \tag{14.163}$$

This free energy difference can be calculated from the energy difference distributions

$$\begin{aligned} \rho_A\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right) &= \left\langle \delta\left[\Delta\mathcal{V} - \left(\mathcal{V}_B - \mathcal{V}_A\right)\right]\right\rangle_A \\ \rho_B\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right) &= \left\langle \delta\left[\Delta\mathcal{V} - \left(\mathcal{V}_B - \mathcal{V}_A\right)\right]\right\rangle_B, \end{aligned} \tag{14.164}$$

where $\langle ... \rangle_X$ indicates an average over an ensemble sampled at state X, $\mathcal{V}_X(\boldsymbol{r})$ is the potential energy part of the Hamiltonian $\mathcal{H}_X(\boldsymbol{p}, \boldsymbol{r}) = \mathcal{K}_X(\boldsymbol{p}) + \mathcal{V}_X(\boldsymbol{r})$, and $\Delta\mathcal{V}_{BA}(\boldsymbol{r}) = \mathcal{V}_B(\boldsymbol{r}) - \mathcal{V}_A(\boldsymbol{r})$. In the following the kinetic part of the Hamiltonian $\mathcal{K}_X(\boldsymbol{p})$ is omitted for simplicity. The free energy difference can be expressed as[147–149]

$$\rho_B\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right) e^{-\beta\Delta\mathcal{F}_{BA}} = \rho_A\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right) e^{-\beta\Delta\mathcal{V}}, \tag{14.165}$$

indicating that the free energy difference $\Delta\mathcal{F}_{BA}$ is the energy difference $\Delta\mathcal{V}$ where the two energy difference distributions intersect. Sampling of both $\rho_A\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right)$ and $\rho_B\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right)$ in a single simulation can be achieved by construction of a reference state R that envelopes the two states of interest. The free energy difference is then estimated as

$$\Delta\mathcal{F}_{BA} = -\beta^{-1} \ln \frac{\left\langle e^{-\beta\left(\mathcal{V}_B - \mathcal{V}_R\right)}\right\rangle_R}{\left\langle e^{-\beta\left(\mathcal{V}_A - \mathcal{V}_R\right)}\right\rangle_R}. \tag{14.166}$$

A reference state Hamiltonian which allows sampling of both $\rho_A\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right)$ and $\rho_B\left(\Delta\mathcal{V}; \Delta\mathcal{V}_{BA}\right)$ reads[150–152]

$$\mathcal{V}_R(\boldsymbol{r}) = -\beta^{-1} \ln\left\{e^{-\beta\left(\mathcal{V}_A(\mathbf{r}) - E_A^R\right)} + e^{-\beta\left(\mathcal{V}_B(\mathbf{r}) - E_B^R\right)}\right\}, \tag{14.167}$$

where E_A^R and E_B^R are energy offset parameters. If the important parts of configuration space of A and B lie far apart, regions of phase space important to A and to B will be separated by a high barrier on the potential energy surface of the reference state.

14.9.1. EDS with smoothness parameter s. To lower the energy barrier between the states, a dimensionless smoothness parameter $s > 0$ can be introduced:[153]

$$\mathcal{V}_R(\boldsymbol{r}) = -\left(\beta s\right)^{-1} \ln\left\{e^{-\beta s\left(\mathcal{V}_A(\mathbf{r}) - E_A^R\right)} + e^{-\beta s\left(\mathcal{V}_B(\mathbf{r}) - E_B^R\right)}\right\}, \tag{14.168}$$

In practice, the statistical efficiency[154] strongly depends on the chosen smoothness parameter s.[155] In order to ensure efficient sampling of the regions of phase space important to state A and those important to state B, the barrier can be decreased by lowering s ($s > 0$).

In order to estimate multiple free energy differences from a single simulation the reference state Hamiltonian has to be generalized to multiple EDS states. Currently three different ways of doing this are implemented in MD++. All presented Hamiltonians reduce to Eq. 14.167 for two EDS states. A generalized reference state that uses a single smoothness parameter reads[150–153]

$$\mathcal{V}_R(\boldsymbol{r}) = -\left(\beta s\right)^{-1} \ln\left\{\sum_{i=1}^{\mathcal{N}^{(s)}} e^{-\beta s\left(\mathcal{V}_i(\mathbf{r}) - E_i^R\right)}\right\}, \tag{14.169}$$

where $\mathcal{N}^{(s)}$ is the number of EDS states. The corresponding equations of motion read

$$\dot{\boldsymbol{r}}_k(t) = m^{-1}\boldsymbol{p}_k(t) \tag{14.170}$$

$$\begin{aligned}
\dot{\boldsymbol{p}}_k(t) &= \boldsymbol{f}_k(t) = \left(-\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \\
&= \sum_{i=1}^{\mathcal{N}^{(s)}} \left\{ \frac{e^{-\beta s\left(\mathcal{V}_i(\boldsymbol{r})-E_i^R\right)}}{\sum_{j=1}^{\mathcal{N}^{(s)}} e^{-\beta s\left(\mathcal{V}_j(\boldsymbol{r})-E_j^R\right)}} \left(-\frac{\partial \mathcal{V}_i(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \right\} \\
&= \sum_{i=1}^{\mathcal{N}^{(s)}} \left\{ \left[\sum_{j=1, j\neq i}^{\mathcal{N}^{(s)}} e^{-\beta s\left(\Delta\mathcal{V}_{ji}(\boldsymbol{r})-\Delta E_{ji}^R\right)} + 1 \right]^{-1} \left(-\frac{\partial \mathcal{V}_i(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \right\},
\end{aligned} \tag{14.171}$$

with $\Delta\mathcal{V}_{ji}(\boldsymbol{r}) = \mathcal{V}_j(\boldsymbol{r}) - \mathcal{V}_i(\boldsymbol{r})$ and $\Delta E_{ji}^R = E_j^R - E_i^R$. Employing a single smoothness parameter s can be problematic if the important parts of phase space of some states lie far apart (requiring a low s parameter) and that of others are close (allowing a higher s parameter). Therefore, a generalized reference state that employs $\mathcal{N}^{(s)}(\mathcal{N}^{(s)}-1)/2$ pairwise smoothness parameters s_{ij} can be defined[156]

$$\mathcal{V}_R(\boldsymbol{r}) = -\frac{1}{\beta}\ln\left\{ \left[\sum_{i=1}^{\mathcal{N}^{(s)}-1} \sum_{j>i}^{\mathcal{N}^{(s)}} \left(e^{-\beta s_{ij}\left(\mathcal{V}_i(\boldsymbol{r})-E_i^R\right)} + e^{-\beta s_{ij}\left(\mathcal{V}_j(\boldsymbol{r})-E_j^R\right)} \right)^{\frac{1}{s_{ij}}} \right] \frac{1}{\mathcal{N}^{(s)}-1} \right\}, \tag{14.172}$$

with the corresponding equations of motion

$$\dot{\boldsymbol{r}}_k(t) = m^{-1}\boldsymbol{p}_k(t) \tag{14.173}$$

$$\begin{aligned}
&\dot{\boldsymbol{p}}_k(t) = \boldsymbol{f}_k(t) = \left(-\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \\
&= \sum_{i=1}^{\mathcal{N}^{(s)}-1} \sum_{j>i}^{\mathcal{N}^{(s)}} \left\{ \frac{\left(e^{-\beta s_{ij}\left(\mathcal{V}_i(\boldsymbol{r})-E_i^R\right)} + e^{-\beta s_{ij}\left(\mathcal{V}_j(\boldsymbol{r})-E_j^R\right)} \right)^{\frac{1}{s_{ij}}}}{\sum_{l=1}^{\mathcal{N}^{(s)}-1} \sum_{m>l}^{\mathcal{N}^{(s)}} \left(e^{-\beta s_{lm}\left(\mathcal{V}_l(\boldsymbol{r})-E_l^R\right)} + e^{-\beta s_{lm}\left(\mathcal{V}_m(\boldsymbol{r})-E_m^R\right)} \right)^{\frac{1}{s_{lm}}}} \right.
\end{aligned}$$

$$\left. \cdot \left[\frac{\left(-\partial \mathcal{V}_i(\boldsymbol{r})/\partial \boldsymbol{r}_k\right)}{1 + e^{-\beta s_{ij}\left(\Delta\mathcal{V}_{ji}(\boldsymbol{r})-\Delta E_{ji}^R\right)}} + \frac{\left(-\partial \mathcal{V}_j(\boldsymbol{r})/\partial \boldsymbol{r}_k\right)}{1 + e^{+\beta s_{ij}\left(\Delta\mathcal{V}_{ji}(\boldsymbol{r})-\Delta E_{ji}^R\right)}} \right] \right\}. \tag{14.174}$$

However, only $(\mathcal{N}^{(s)}-1)$ s_{ij} parameters are necessary to connect all states with each other. Imagine that the "closest path" from A to C is via B. Then it would suffice to adapt s_{AB} and s_{BC} instead of introducing a (possibly very low) s_{AC}. This idea has been pursued in the third reference state Hamiltonian[156]

$$\mathcal{V}_R(\mathbf{r}) = -\frac{1}{\beta}\ln\left\{ \left[\sum_{\substack{(\mathcal{N}^{(s)}-1) \\ i,j\ pairs}} \left(e^{-\beta s_{ij}\left(\mathcal{V}_i(\boldsymbol{r})-E_i^R\right)} + \quad e^{-\beta s_{ij}\left(\mathcal{V}_j(\boldsymbol{r})-E_j^R\right)} \right)^{\frac{1}{s_{ij}}} \right] \frac{\mathcal{N}^{(s)}}{2(\mathcal{N}^{(s)}-1)} \right\}, \tag{14.175}$$

with the corresponding equations of motion

$$\dot{\boldsymbol{r}}_k(t) = m^{-1}\boldsymbol{p}_k(t) \tag{14.176}$$

$$\begin{aligned}
&\dot{\boldsymbol{p}}_k(t) = \boldsymbol{f}_k(t) = \left(-\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \\
&= \sum_{\substack{(\mathcal{N}^{(s)}-1) \\ i,j\ pairs}} \left\{ \frac{\left(e^{-\beta s_{ij}\left(\mathcal{V}_i(\boldsymbol{r})-E_i^R\right)} + e^{-\beta s_{ij}\left(\mathcal{V}_j(\boldsymbol{r})-E_i^R\right)} \right)^{\frac{1}{s_{ij}}}}{\sum_{\substack{(\mathcal{N}^{(s)}-1) \\ l,m\ pairs}} \left(e^{-\beta s_{lm}\left(\mathcal{V}_l(\boldsymbol{r})-E_l^R\right)} + e^{-\beta s_{lm}\left(\mathcal{V}_m(\boldsymbol{r})-E_m^R\right)} \right)^{\frac{1}{s_{lm}}}} \right.
\end{aligned}$$

$$\cdot \left[\frac{(-\partial \mathcal{V}_i(\boldsymbol{r})/\partial \boldsymbol{r}_k)}{1+e^{-\beta s_{ij}\left(\Delta\mathcal{V}_{ji}(\boldsymbol{r})-\Delta E^R_{ji}\right)}} + \frac{(-\partial \mathcal{V}_j(\boldsymbol{r})/\partial \boldsymbol{r}_k)}{1+e^{+\beta s_{ij}\left(\Delta\mathcal{V}_{ji}(\boldsymbol{r})-\Delta E^R_{ji}\right)}} \right] \Bigg\} . \tag{14.177}$$

Here, the sum is only performed over $(\mathcal{N}^{(s)}-1)$ pairs. The $(\mathcal{N}^{(s)}-1)$ pairs are chosen from all $\mathcal{N}^{(s)}(\mathcal{N}^{(s)}-1)/2$ pairs such that a maximum spanning tree of s_{ij} parameters is obtained. That is, only those EDS end states that show the closest "distance" in phase space (i.e. allow for the largest s_{ij}) are directly connected.

Although the three reference state Hamiltonians (Eq. 14.169, Eq. 14.172, Eq. 14.175) are of increasing complexity, the computational effort in an MD simulation employing these Hamiltonians is comparable. This is due to the fact that the bottleneck of the computation is not the combination of the end state energies and forces to obtain the reference state potential energy (Eq. 14.169, Eq. 14.172, Eq. 14.175) and forces (Eq. 14.171, Eq. 14.174, Eq. 14.177), respectively, but the computation of these end state energies and forces themselves.

In order to limit the computational effort, the unperturbed interactions are calculated only once at each time step, i.e. are not unnecessarily recalculated for each of the $\mathcal{N}^{(s)}$EDS Hamiltonian terms. Only the perturbed interactions are calculated at every time step for each of the $\mathcal{N}^{(s)}$EDS Hamiltonian terms. The overhead of an EDS calculation compared to a single, standard MD simulation is, therefore, determined by the size of the perturbed part of the system. If the number of unperturbed interactions is larger than the number of perturbed interactions, an EDS simulation of $\mathcal{N}^{(s)}$ states will take less computing time than $\mathcal{N}^{(s)}$ independent non-EDS molecular dynamics simulations. If the number of perturbed interactions is very large, the number of states is small, and the conformational changes involved in the perturbations are big, standard staging methods such as thermodynamic integration (see Sec. 14.6) would be the method of choice. EDS will work also in these cases but is not likely to be more efficient than standard approaches. However, often one is interested in perturbations involving many states, with a rather small number of perturbed interactions, and rather local conformational changes. A prototypical example would be the binding of many distinct ligands to a common receptor. Here, the number of perturbed interactions is much smaller than the number of unperturbed interactions and performing an EDS simulation will be more efficient than performing all simulations independently. Unlike in staging approaches such as TI no simulations at "unphysical" intermediate states are performed. Although the reference state itself is an "unphysical" intermediate state, it is constructed such that the sampling is focused on configurations which are of importance to the end states.

14.9.2. Accelerated EDS. Since modification of $\mathcal{V}_R(\boldsymbol{r})$ with the smoothness parameter s leads to distortion of the original energy minima with respect to the coordinates,[153] a different approach based on Accelerated MD[157,158] can be chosen to lower large potential energy barriers between end-states. The continuous accelerated EDS Hamiltonian $\mathcal{V}^*_R(\boldsymbol{r})$ smoothened with a harmonic potential energy function which preserves local energy minima reads[159,160]

$$\mathcal{V}^*_R(\boldsymbol{r}) = \begin{cases} \mathcal{V}_R(\boldsymbol{r}) - \frac{E_{max}-E_{min}}{2}, & \text{for } \mathcal{V}_R(\boldsymbol{r}) \geq E_{max} \\ \mathcal{V}_R(\boldsymbol{r}) - \frac{1}{2(E_{max}-E_{min})}\left(\mathcal{V}_R(\boldsymbol{r})-E_{min}\right)^2, & \text{for } E_{max} > \mathcal{V}_R(\boldsymbol{r}) > E_{min} \\ \mathcal{V}_R(\boldsymbol{r}), & \text{for } \mathcal{V}_R(\boldsymbol{r}) \leq E_{min} \end{cases} \tag{14.178}$$

where $\mathcal{V}_R(\boldsymbol{r})$ is the non-accelerated EDS Hamiltonian given by Eq. 14.167, E_{max} and E_{min} are the maximum and minimum borders of the accelerated region of the EDS Hamiltonian, respectively. The accelerated EDS equations of motion read

$$\dot{\boldsymbol{r}}_k(t) = m^{-1}\boldsymbol{p}_k(t) \tag{14.179}$$

$$\begin{aligned} \dot{\boldsymbol{p}}_k(t) &= \boldsymbol{f}_k(t) = \left(-\frac{\partial \mathcal{V}^*_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\right) \\ &= \begin{cases} -\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}\left(\frac{E_{max}-\mathcal{V}_R(\boldsymbol{r})}{E_{max}-E_{min}}\right), & \text{for } E_{max} > \mathcal{V}_R(\boldsymbol{r}) > E_{min} \\ -\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}, & \text{for } \mathcal{V}_R(\boldsymbol{r}) \geq E_{max}, \mathcal{V}_R(\boldsymbol{r}) \leq E_{min} \end{cases} \end{aligned} \tag{14.180}$$

where $-\frac{\partial \mathcal{V}_R(\boldsymbol{r})}{\partial \boldsymbol{r}_k}$ is the negative derivative of the non-accelerated EDS Hamiltonian given by Eq. 14.171 with a smoothness parameter $s = 1$.

Adequate accelerated EDS parameters E_{max}, E_{min} and energy offset parameters E_i^R can be determined simultaneously during a non-equilibrium parameter search simulation in which the EDS Hamiltonian is explored freely and EDS parameters are adjusted on-the-fly. An end-state is currently sampled by the EDS Hamiltonian if its energy $\left(\mathcal{V}_i(\mathbf{r}) - E_i^R\right)$ is minimal. If an end-state is sampled, its average energy $\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}$ and standard deviaton of the energy $\sigma_{\mathcal{V}_i(\mathbf{r})}$ are calculated. Moreover, the average of the maximum transition energy between states within a state-visit period (a state-visit period is defined as having seen all states at least once), $E_{max}^{\ddagger}$, is calculated. The maximum potential energy barrier between any end-states is now given by

$$\Delta E_{max} = E_{max}^{\ddagger} - \min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}\right). \tag{14.181}$$

The upper border for the accelerated region of the EDS Hamiltonian is $E_{max}^{\ddagger}$

$$E_{max} = E_{max}^{\ddagger} \tag{14.182}$$

and the lower border for the accelerated region is calculated such that the maximum potential energy barrier in an accelerated EDS Hamiltonian $\mathcal{V}_R^*(\boldsymbol{r})$ is reduced to a value ΔE_{max}^*. This value can be chosen to be a multiple of the standard deviation $\sigma_{\mathcal{V}_i(\mathbf{r})}$ of the energy of the end-state with the lowest average energy $\min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}\right)$.**[160]** The lower border E_{min} reads

$$E_{min} = 2\left\{\min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}\right) + \Delta E_{max}^*\right\} - E_{max}^{\ddagger}. \tag{14.183}$$

If E_{min} calculated according to Eq. 14.183 is smaller than $\min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}\right)$, it is given by

$$E_{min} = \frac{2E_{max}^{\ddagger}\Delta E_{max}^* + 2E_{max}^{\ddagger}\min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}\right) - \left(E_{max}^{\ddagger}\right)^2 - \min\left(\overline{\mathcal{V}_i(\mathbf{r}) - E_i^R}.\right)^2}{2\Delta E_{max}^*}. \tag{14.184}$$

The EDS energy offset parameters E_i^R are calculated explicitly from the free-energy differences between the single end-states with accelerated Hamiltonians $\mathcal{V}_i^*(\boldsymbol{r})$**[160]** which read

$$\mathcal{V}_i^*(\boldsymbol{r}) = \begin{cases} \mathcal{V}_i(\boldsymbol{r}) - \frac{E_{max}-E_{min}}{2}, & \text{for } \mathcal{V}_i(\boldsymbol{r}) \geq E_{max} + E_i^R \\ \mathcal{V}_i(\boldsymbol{r}) - \frac{1}{2(E_{max}-E_{min})}\left(\mathcal{V}_i(\boldsymbol{r}) - E_{min} - E_i^R\right)^2, & \text{for } E_{max} + E_i^R > \mathcal{V}_i(\boldsymbol{r}) > E_{min} + E_i^R \\ \mathcal{V}_i(\boldsymbol{r}), & \text{for } \mathcal{V}_i(\boldsymbol{r}) \leq E_{min} + E_i^R \end{cases} \tag{14.185}$$

The energy offset of the first state is arbitrarily set to zero ($E_1^R = 0$) and all other energy offset parameters $E_{i\neq 1}^R$ are calculated by free energy perturbation given in Eq. 14.6

$$E_{i\neq 1}^R = -k_B T \ln\left(\left\langle e^{-[\mathcal{V}_{i\neq 1}^*(\boldsymbol{r}) - \mathcal{V}_R^*(\boldsymbol{r})]/k_B T}\right\rangle_R\right) + k_B T \ln\left(\left\langle e^{-[\mathcal{V}_1^*(\boldsymbol{r}) - \mathcal{V}_R^*(\boldsymbol{r})]/k_B T}\right\rangle_R\right). \tag{14.186}$$

To allow for faster adjustment of the energy offset parameters during the parameter search simulation, a *memory relaxation time* $\tau_{\Delta\mathcal{V}}$**[58]** is implemented for the exponential averages of the potential energy differences in analogy to the time-averaged distance restraining function described in Eq. 9.14. The exponential potential energy differences with a characteristic memory decay time $\tau_{\Delta\mathcal{V}}$ read

$$\begin{aligned} \langle e^{-[\mathcal{V}_i^*(\boldsymbol{r}) - \mathcal{V}_R^*(\boldsymbol{r})]/k_B T}\rangle_t &= \left(1 - e^{-\Delta t/\tau_{\Delta\mathcal{V}}}\right) e^{-[\mathcal{V}_i^*(\boldsymbol{r}) - \mathcal{V}_R^*(\boldsymbol{r})]/k_B T}(t) \\ &+ e^{-\Delta t/\tau_{\Delta\mathcal{V}}} \langle e^{-[\mathcal{V}_i^*(\boldsymbol{r}) - \mathcal{V}_R^*(\boldsymbol{r})]/k_B T}\rangle_{t-\Delta t}, \end{aligned} \tag{14.187}$$

where t is the simulation time and Δt the simulation time-step. In MD++, the memory decay time $\tau_{\Delta\mathcal{V}}$ can be linearly interpolated over time between two values $\tau^A_{\Delta\mathcal{V}}$ at the beginning of the simulation run and $\tau^B_{\Delta\mathcal{V}}$ at the end of the simulation run:

$$\tau_{\Delta\mathcal{V}} = \tau^A_{\Delta\mathcal{V}} + \left(\tau^B_{\Delta\mathcal{V}} - \tau^A_{\Delta\mathcal{V}}\right)\frac{t}{t_{tot}}. \tag{14.188}$$

Here, t_{tot} is the total simulation time of the run. This is especially useful for systems in which the energy offset parameters are very large. In such cases, $\tau^A_{\Delta\mathcal{V}}$ can be set to a small value to allow for rapid adjustment of the energy offset parameters at the beginning of the parameter search simulation, while more statistics are used upon convergence of the energy offset parameters towards the end of the parameter search simulation.

14.9.3. Twin-system EDS. A drawback of the thermodynamic cycle shown in Fig. 14.1 is that the accuracy of $\Delta\Delta\mathcal{F}_{43} = \Delta\Delta\mathcal{F}_{21}$ can be low if two large, almost equal numbers, are subtracted, e.g. due to a change in charge state or partial charges between I_1 and I_2. This problem can be avoided by an alternative choice of the thermodynamic cycle, shown in Fig. 14.3, in which (i) one state combines the free state and computational box for one ligand with the bound state and computational box for the other ligand, and (ii) the two processes of changing ligand I_1 into I_2 in these free and bound states are carried out in opposite directions.[161] This process directly yields $\Delta\Delta\mathcal{F}_{21}$ and may lead to a smaller change of the energy of the combined state. In Fig. 14.3, the round brackets denote a periodic computational box with a particular ligand in water (free) or with a particular ligand bound to the protein in water (bound). The rectangular brackets denote that the two computational boxes are to be combined into one state or Hamiltonian,

$$\begin{aligned}\mathcal{H}_A &= \mathcal{H}\left(I_2; free\right) + \mathcal{H}\left(E; I_1; bound\right) \\ \mathcal{H}_B &= \mathcal{H}\left(I_1; free\right) + \mathcal{H}\left(E; I_2; bound\right)\end{aligned} \tag{14.189}$$

Using EDS this process can be simulated for a pair of states A and B and two computational boxes 1 and 2, employing the reference state Hamiltonian

$$\mathcal{V}_R\left(\boldsymbol{r}\right) = -\left(\beta s\right)^{-1}\ln\left\{e^{\left[-\beta s\left(\mathcal{V}_{A1}(\mathbf{r})+\mathcal{V}_{A2}(\mathbf{r})-E^R_A\right)\right]} + e^{\left[-\beta s\left(\mathcal{V}_{B1}(\mathbf{r})+\mathcal{V}_{B2}(\mathbf{r})-E^R_B\right)\right]}\right\}, \tag{14.190}$$

where $\mathcal{V}_{Xi}\left(\boldsymbol{r}\right)$ is the potential energy part corresponding to box i of the Hamiltonian $\mathcal{H}_X\left(\boldsymbol{p},\boldsymbol{r}\right)$. The corresponding equations of motion read

$$\dot{\boldsymbol{r}}_k(t) = m^{-1}\boldsymbol{p}_k(t) \tag{14.191}$$

$$\begin{aligned}\dot{\boldsymbol{p}}_k(t) &= \boldsymbol{f}_k(t) = \left(-\frac{\partial\mathcal{V}_R(\boldsymbol{r})}{\partial\boldsymbol{r}_k}\right) \\ &= \left[e^{-\beta s\left(\mathcal{V}_{B1}(\mathbf{r})+\mathcal{V}_{B2}(\mathbf{r})-\left(\mathcal{V}_{A1}(\mathbf{r})+\mathcal{V}_{A2}(\mathbf{r})\right)-\Delta E^R_{BA}\right)}+1\right]^{-1}\left(-\frac{\partial\left(\mathcal{V}_{A1}(\boldsymbol{r})+\mathcal{V}_{A2}(\boldsymbol{r})\right)}{\partial\boldsymbol{r}_k}\right) \\ &\quad + \left[e^{+\beta s\left(\mathcal{V}_{B1}(\mathbf{r})+\mathcal{V}_{B2}(\mathbf{r})-\left(\mathcal{V}_{A1}(\mathbf{r})+\mathcal{V}_{A2}(\mathbf{r})\right)-\Delta E^R_{BA}\right)}+1\right]^{-1}\left(-\frac{\partial\left(\mathcal{V}_{B1}(\boldsymbol{r})+\mathcal{V}_{B2}(\boldsymbol{r})\right)}{\partial\boldsymbol{r}_k}\right).\end{aligned} \tag{14.192}$$

Because the potential energy of box 1 does not depend on the configurations in box 2 and vice versa,

$$\begin{aligned}\frac{\partial\mathcal{V}_{A2}(\boldsymbol{r})}{\partial\boldsymbol{r}_k} = \frac{\partial\mathcal{V}_{B2}(\boldsymbol{r})}{\partial\boldsymbol{r}_k} = 0 \qquad &\text{for particles } k \text{ in system 1} \\ \frac{\partial\mathcal{V}_{A1}(\boldsymbol{r})}{\partial\boldsymbol{r}_k} = \frac{\partial\mathcal{V}_{B1}(\boldsymbol{r})}{\partial\boldsymbol{r}_k} = 0 \qquad &\text{for particles } k \text{ in system 2}\end{aligned} \tag{14.193}$$

Therefore, the coupling of the two boxes only occurs via the prefactors.

14.9.4. Configurational EDS. The EDS method can also be used to obtain the relative free energy of different conformations or configurational states. Assume we wish to calculate the free enthalpy difference between two conformations, α and β, of a molecule, and one or both of them is not the most stable one of the molecule. We may use the EDS technique to obtain the free enthalpy difference by defining the EDS reference Hamiltonian as follows. Two restraining energy function terms are defined which restrain the molecular conformations to conformation α or to conformation β, *i.e.*, $\mathcal{V}^{rest}_X(\boldsymbol{r}^N; K^{rest}_X, \boldsymbol{r}^N_{0,\xi})$ where X = A or B and $\boldsymbol{r}^N_{0,\xi}$ is the set of parameters which characterizes the conformation ξ, $\xi = \alpha$ or β, *e.g.*,

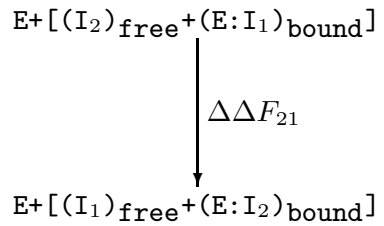


FIGURE 14.3. Alternative thermodynamic cycle with respect to the relative binding of two inhibitors I_1 and I_2 to an enzyme E. The symbol ':' indicates complexation.

through particular hydrogen-bond distance ranges or torsional-angle ranges, and K_X^{rest} is the restraining force constant. Thus, the resulting Hamiltonian for the end state X is

$$\mathcal{V}_X(\boldsymbol{r}^N) = \mathcal{V}_X^{rest}(\boldsymbol{r}^N; k_X^{rest}, \boldsymbol{r}_{0\xi}^N) + \mathcal{V}^{(phys)}(\boldsymbol{r}^N) \tag{14.194}$$

where $\mathcal{V}^{(phys)}(\boldsymbol{r}^N)$ is the interaction function of a particular force field. Then, we may construct an EDS reference-state Hamiltonian:

$$\begin{aligned} \mathcal{V}_R(\boldsymbol{r}^N; s, E_{BA}^R) &= -\beta^{-1}s^{-1}\ln\left\{e^{-\beta s(\mathcal{V}_A^{rest}(\boldsymbol{r}^N)-E_A^R)} + e^{-\beta s(\mathcal{V}_B^{rest}(\boldsymbol{r}^N)-E_B^R)}\right\} + \mathcal{V}^{(phys)}(\boldsymbol{r}^N) \\ &= \mathcal{V}^{\mathrm{EDS},rest}(\boldsymbol{r}^N; s, E_{BA}^R) + \mathcal{V}^{(phys)}(\boldsymbol{r}^N) \end{aligned} \tag{14.195}$$

where s is a smoothness parameter and $E_B^R - E_A^R = E_{BA}^R$ is an energy offset parameter difference, which are chosen such as to optimize the sampling of both end states A and B. In the original EDS implementation, the configurations $\boldsymbol{r}^N$ that are sampled by the reference Hamiltonian $\mathcal{H}_R$, *i.e.*, $\mathcal{V}_R$, are not assigned to any conformational states. They are considered[90] to belong to state A if

$$\mathcal{V}_A^{rest}(\boldsymbol{r}^N) - E_A^R) < (\mathcal{V}_B^{rest}(\boldsymbol{r}^N) - E_B^R) \tag{14.196}$$

In the case considered here, configurations must be separated into different sets, *i.e.*, different conformational states: they belong to set α if

$$\mathcal{V}_A^{rest}(\boldsymbol{r}^N) \leq E_\alpha^{thres} \text{ and } \mathcal{V}_B^{rest}(\boldsymbol{r}^N) > E_\beta^{thres} \tag{14.197}$$

they belong to set β if

$$\mathcal{V}_A^{rest}(\boldsymbol{r}^N) > E_\alpha^{thres} \text{ and } \mathcal{V}_B^{rest}(\boldsymbol{r}^N) \leq E_\beta^{thres} \tag{14.198}$$

or they may belong to neither of them, called sets γ and δ with set γ defined by

$$\mathcal{V}_A^{rest}(\boldsymbol{r}^N) > E_\alpha^{thres} \text{ and } \mathcal{V}_B^{rest}(\boldsymbol{r}^N) > E_\beta^{thres} \tag{14.199}$$

and set δ defined by

$$\mathcal{V}_A^{rest}(\boldsymbol{r}^N) \leq E_\alpha^{thres} \text{ and } \mathcal{V}_B^{rest}(\boldsymbol{r}^N) \leq E_\beta^{thres} \tag{14.200}$$

Generally, set δ should contain no or only a few configurations in order to make a meaningful distinction between sets α and β. Here, the configurations that belong to sets α and β are defined via an energy threshold criterion E_ξ^{thres}, which maps configurations $\boldsymbol{r}^N$ onto an energy $\mathcal{V}_X^{rest}(\boldsymbol{r}^N)$ using the same function $\mathcal{V}_X^{rest}(\boldsymbol{r}^N)$ that is used in the reference Hamiltonian. This means that the configurations that belong to sets α and β are defined through Eqs. 14.197 and 14.198, respectively. We note that these sets α and β differ from the conformational ensembles A and B that are through the end-state Hamiltonians defined by Eq. 14.194. Alternatively, the conformational sets α and β could be defined using a geometric measure such as an atom-positional root-mean-square deviation (RMSD) from a given configuration, either in Cartesian

or in internal torsional coordinates, instead of using the restraining functions $\mathcal{V}_X^{rest}$ and threshold energies E_ξ^{thres}. Configurations then belong to set α if

$$\mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\alpha^N) \leq \mathrm{RMSD}_\alpha^{thres} \text{ and } \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\beta^N) > \mathrm{RMSD}_\beta^{thres} \tag{14.201}$$

they belong to set β if

$$\mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\alpha^N) > \mathrm{RMSD}_\alpha^{thres} \text{ and } \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\beta^N) \leq \mathrm{RMSD}_\beta^{thres} \tag{14.202}$$

or they belong to neither of them, called sets γ and δ:

$$\gamma : \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\alpha^N) > \mathrm{RMSD}_\alpha^{thres} \text{ and } \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\beta^N) > \mathrm{RMSD}_\beta^{thres} \tag{14.203}$$

$$\delta : \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\alpha^N) \leq \mathrm{RMSD}_\alpha^{thres} \text{ and } \mathrm{RMSD}(\boldsymbol{r}^N, \boldsymbol{r}_\beta^N) \leq \mathrm{RMSD}_\beta^{thres} \tag{14.204}$$

Again, the thresholds $\mathrm{RMSD}_\xi^{thres}$ should be chosen such that set δ contains no or only a few configurations. In the procedure and expressions used in the optimization of the parameters s and $E_B^R = E_{BA}^R$ (E_A^R is standardly set to zero in two-state EDS), configurations that belong to sets γ and δ can be ignored. Thus, we get for updating the energy offset E_B^R (corresponds to eq 13 of ref[155]):

$$E_B^R(new) = -\frac{1}{\beta} \ln \left\langle \left\{ e^{-\beta(\mathcal{V}_A^{rest} - \mathcal{V}_B^{rest} + E_B^R(old))} + 1 \right\}^{-1} \right\rangle_{R,not\gamma,not\delta} + E_B^R(old) \tag{14.205}$$

where configurations of sets γ and δ are excluded when calculating the ensemble average over the ensemble of the reference state R. For updating or rather choosing a new s parameter, we calculate

$$s_A = -\left\{ \ln \left\langle e^{-\beta(|\mathcal{V}_B^{rest} - \mathcal{V}_A^{rest}| - E_{BA}^R)} \right\rangle_A \right\}^{-1} \tag{14.206}$$

and

$$s_B = -\left\{ \ln \left\langle e^{-\beta(|\mathcal{V}_A^{rest} - \mathcal{V}_B^{rest}| + E_{BA}^R)} \right\rangle_B \right\}^{-1} \tag{14.207}$$

and take the lowest s value as the new s

$$s = \min(s_A, s_B) \tag{14.208}$$

which corresponds to eq 14 of ref[155]. Ensembles A and B are obtained by reweighting the configurations generated using the reference state R to the corresponding end state A or B. For a quantity $Q(\boldsymbol{r}^N)$, which is a function of the coordinates $\boldsymbol{r}^N$, we have

$$\langle Q \rangle_X = \frac{\int Q(\boldsymbol{r}^N) e^{-\beta \mathcal{V}_X(\boldsymbol{r}^N)} d\boldsymbol{r}^N}{\int e^{-\beta \mathcal{V}_X(\boldsymbol{r}^N)} d\boldsymbol{r}^N} \tag{14.209}$$

or using the ensemble R

$$\langle Q \rangle_X = \frac{\left\langle Q e^{-\beta(\mathcal{V}_X - \mathcal{V}_R)} \right\rangle_R}{\left\langle e^{-\beta(\mathcal{V}_X - \mathcal{V}_R)} \right\rangle_R} \tag{14.210}$$

Subsequently, the ensemble averaging in Eq. 14.210 could be restricted to the sets α and β. In that case, these restricted ensemble averages can be written as

$$\langle Q \rangle_A = \frac{\left\langle Q e^{-\beta(\mathcal{V}_A - \mathcal{V}_R)} \right\rangle_{R,not\gamma,not\delta}}{\left\langle e^{-\beta(\mathcal{V}_A - \mathcal{V}_R)} \right\rangle_{R,not\gamma,not\delta}} \tag{14.211}$$

$$= \frac{\left\langle Q e^{-\beta\left(\mathcal{V}_A^{rest}-\mathcal{V}^{\mathrm{EDS},rest}(s,E_{BA}^R)\right)}\right\rangle_{R,not\gamma,not\delta}}{\left\langle e^{-\beta\left(\mathcal{V}_A^{rest}-\mathcal{V}^{\mathrm{EDS},rest}(s,E_{BA}^R)\right)}\right\rangle_{R,not\gamma,not\delta}}$$

and

$$\langle Q\rangle_B = \frac{\left\langle Q e^{-\beta(\mathcal{V}_B-\mathcal{V}_R)}\right\rangle_{R,not\gamma,not\delta}}{\left\langle e^{-\beta(\mathcal{V}_B-\mathcal{V}_R)}\right\rangle_{R,not\gamma,not\delta}} \tag{14.212}$$

$$= \frac{\left\langle Q e^{-\beta\left(\mathcal{V}_B^{rest}-\mathcal{V}^{\mathrm{EDS},rest}(s,E_{BA}^R)\right)}\right\rangle_{R,not\gamma,not\delta}}{\left\langle e^{-\beta\left(\mathcal{V}_B^{rest}-\mathcal{V}^{\mathrm{EDS},rest}(s,E_{BA}^R)\right)}\right\rangle_{R,not\gamma,not\delta}}$$

In this way, erratic irrelevant energy values due to irrelevant configurations not belonging to sets α and β are excluded from influencing the parameter optimization for sampling of sets α and β. Furthermore, configurations which belong to set δ that have low $\mathcal{V}_X^{rest}$ values are excluded too. The free enthalpy difference between two end-state Hamiltonians B and A in the EDS simulation is evaluated through[155]

$$\Delta G_{BA} = G_B - G_A = \Delta G_{BR} - \Delta G_{AR} = -\frac{1}{\beta}\ln\frac{\left\langle e^{-\beta(\mathcal{H}_B-\mathcal{H}_R)}\right\rangle_R}{\left\langle e^{-\beta(\mathcal{H}_A-\mathcal{H}_R)}\right\rangle_R} \tag{14.213}$$

The expression used to obtain the free enthalpy difference between conformational sets β and α from an ensemble generated using the reference-state Hamiltonian $\mathcal{V}_R(\boldsymbol{r}^N; s, E_{BA}^R)$ reads

$$\Delta G_{\beta\alpha} = G_\beta - G_\alpha == -\frac{1}{\beta}\ln\left\{\frac{N_\beta(\mathcal{V}^{(phys)})}{N_\alpha(\mathcal{V}^{(phys)})}\right\} \tag{14.214}$$

where $N_\xi(\mathcal{V}^{(phys)})$ is the number of configurations belonging to set ξ in an ensemble generated using $\mathcal{V}^{(phys)}$. In terms of the ensemble R generated using the reference-state potential energy $\mathcal{V}_R$, we get

$$\Delta G_{\beta\alpha} = -\frac{1}{\beta}\ln\left\{\frac{\left\langle e^{+\beta\mathcal{V}^{\mathrm{EDS},rest}}\right\rangle_{R,set\beta}}{\left\langle e^{+\beta\mathcal{V}^{\mathrm{EDS},rest}}\right\rangle_{R,set\alpha}} \times \frac{N_\beta(\mathcal{V}_R)}{N_\alpha(\mathcal{V}_R)}\right\} \tag{14.215}$$

In other words, the ensemble R that was generated using the biasing potential energy function $\mathcal{V}^{EDS,rest}$ is reweighted using Eq. 14.215, and the configurations of the sets α and β are used in the averaging via their relative populations in the ensemble R, *i.e.*, $N_\alpha(\mathcal{V}_R)$ and $N_\beta(\mathcal{V}_R)$

$$\frac{N_\beta(\mathcal{V}_R)}{N_\alpha(\mathcal{V}_R)} = \frac{\left\langle\delta(\boldsymbol{r}^N-\boldsymbol{r}_\beta^N)\right\rangle_R}{\langle\delta(\boldsymbol{r}^N-\boldsymbol{r}_\alpha^N)\rangle_R} \tag{14.216}$$

Eq. 14.215 can be simply rewritten as

$$\Delta G_{\beta\alpha} = -\frac{1}{\beta}\ln\frac{N_\beta(\mathcal{V}_R)}{N_\alpha(\mathcal{V}_R)} - \frac{1}{\beta}\ln\left\langle e^{+\beta\mathcal{V}^{\mathrm{EDS},rest}}\right\rangle_{R,set\beta} + \frac{1}{\beta}\ln\left\langle e^{+\beta\mathcal{V}^{\mathrm{EDS},rest}}\right\rangle_{R,set\alpha} \tag{14.217}$$

Eq. 14.217 is equivalent to the expression used in conformational state-specific one-step perturbation.[162] In other words, the EDS reference-state Hamiltonian can be used as the reference state in one-step perturbation, ensuring sufficient sampling of the conformational end states, which is reached by optimizing the parameters s and E_{BA}^R . If simulations based on the end state potential energy functions $\mathcal{V}_X(\boldsymbol{r}^N)$, see Eq. 14.194, are

available, these ensembles X = A and X = B can also be used to obtain the free enthalpy difference between conformational sets β and α:[163]

$$\Delta G_{\beta\alpha} = -\frac{1}{\beta} \ln \left\{ \frac{\left\langle e^{+\beta \mathcal{V}_B^{rest}} \right\rangle_{B,set\beta}}{\left\langle e^{+\beta \mathcal{V}_A^{rest}} \right\rangle_{A,set\alpha}} \times \frac{\langle 1 \rangle_{B,set\beta}}{\langle 1 \rangle_{A,set\alpha}} \times \frac{\left\langle e^{-\beta(\mathcal{V}_M - \mathcal{V}_A)} \right\rangle_A}{\left\langle e^{-\beta(\mathcal{V}_M - \mathcal{V}_B)} \right\rangle_B} \right\} \tag{14.218}$$

in which $\mathcal{V}_M$ is an intermediate state connecting two end states. If we use the EDS reference-state Hamiltonian as the intermediate state, the ensemble averages in the last factor of Eq. 14.218 can be written as

$$\left\langle e^{-\beta(\mathcal{V}_M - \mathcal{V}_X)} \right\rangle_X = \left\langle e^{-\beta(\mathcal{V}^{\mathrm{EDS},rest}(s, E_{BA}^R) - \mathcal{V}_X^{rest})} \right\rangle_X \tag{14.219}$$

In[164] it is shown that this type of EDS can be efficiently used to obtain the relative free energy of a rather unstable conformation or fold.

CHAPTER 15

QM/MM simulation

15.1. Introduction

In the combined quantum-mechanical/molecular-mechanical (QM/MM) methodology, the simulated system is divided spatially into a region that is treated quantum-chemically, e.g. the reactive center of a protein, and a region that is described by a molecular-mechanical force field, e.g. the remaining residues of the protein and the solvent.

In this chapter, the following notation is used: a subscript denotes the subsystem or region, QM or MM, that is described while a superscript indicates the type of the Hamiltonian, quantum-mechanical (QM) or classial-mechanical (CM).

15.2. Hamiltonian

The Hamiltonian, $\hat{\mathcal{H}}$, of the simulated system can be written as the sum of the Hamiltonians of the individual QM and MM subsystems and an additional term, $\hat{\mathcal{V}}_{QM/MM}$ describing the interactions between them,

$$\hat{\mathcal{H}} = \hat{\mathcal{H}}_{QM} + \mathcal{H}_{MM} + \hat{\mathcal{V}}_{QM/MM}, \tag{15.1}$$

where $\hat{\mathcal{H}} = \hat{\mathcal{K}} + \hat{\mathcal{V}}$, with $\hat{\mathcal{K}}$ the kinetic energy and $\hat{\mathcal{V}}$ the potential energy term. Here, the quantum operator $\hat{\mathcal{H}}_{MM}$ which contains a function $\mathcal{H}_{MM}$ and the unity operator will be simply denoted as $\mathcal{H}_{MM}$.

The Hamiltonian of the quantum subsystem can be expressed as the sum of a quantum term $\hat{\mathcal{H}}_{QM}^{QM}$ and a classical term $\mathcal{H}_{QM}^{CM}$,

$$\hat{\mathcal{H}}_{QM} = \hat{\mathcal{H}}_{QM}^{QM} + \mathcal{H}_{QM}^{CM}. \tag{15.2}$$

Accordingly, the coupling term, $\hat{\mathcal{V}}_{QM/MM}$, can be written as

$$\hat{\mathcal{V}}_{QM/MM} = \hat{\mathcal{V}}_{QM/MM}^{QM} + \mathcal{V}_{QM/MM}^{CM}. \tag{15.3}$$

It typically takes into account bonded as well as nonbonded interactions between the QM and MM subsystems. *However, the current implementation[165] only holds for noncovalent interactions at the QM/MM boundary.* To evaluate the electrostatic interactions between the QM and MM regions, $\hat{\mathcal{V}}_{QM/MM}^{QM}$, the MM atoms a are included in the QM Hamiltonian as positionally fixed external point charges (electrostatic embedding scheme). When applying the Born–Oppenheimer approximation, the resulting QM Hamiltonian (in atomic units) of the system reads

$$\begin{aligned}
\hat{\mathcal{H}}^{QM} &= \hat{\mathcal{H}}_{QM}^{QM} + \hat{\mathcal{V}}_{QM/MM}^{QM} \\
&= \Big(\sum_e -\tfrac{1}{2}\nabla_e^2 + \sum_e \sum_{e'>e} \tfrac{1}{|\mathbf{r}_{ee'}|} - \sum_n \sum_e \tfrac{Z_n}{|\mathbf{r}_e - \mathbf{R}_n|} \\
&+ \sum_n \sum_{n'>n} \tfrac{Z_n Z_{n'}}{|\mathbf{R}_{nn'}|}\Big) \\
&+ \Big(\sum_n \sum_a \tfrac{Z_n Z_a}{|\mathbf{R}_{na}|} - \sum_e \sum_a \tfrac{Z_a}{|\mathbf{r}_e - \mathbf{R}_a|}\Big),
\end{aligned} \tag{15.4}$$

where e and e' run over QM treated electrons, n and n' run over QM nuclei, Z is the charge of the QM nuclei and MM atoms, respectively and a runs over all MM atoms within a given cutoff $\mathrm{R_{QM/MM}}$ around the QM region or solute. This Hamiltonian is expressed using the standard (non-SI) units commonly used

in the quantum chemistry community. Van der Waals interactions between the QM nuclei and MM atoms, $\mathcal{V}^{CM}_{QM/MM}$ or $\mathcal{V}^{vdW}_{QM/MM}$, are treated on the basis of classical mechanics.

The classical potential energy term of the system is given by

$$\begin{aligned}
\mathcal{V}^{CM}(\mathbf{r}^n,\mathbf{r}^a) &= \mathcal{V}^{CM}_{MM}(\mathbf{r}^a)+\mathcal{V}^{vdW}_{QM/MM}(\mathbf{r}^n,\mathbf{r}^a)+\mathcal{V}^{vdW}_{QM}(\mathbf{r}^n)\\
&= \Big(\textstyle\sum_{\text{bonds i}}\frac{1}{2}K^b_i[b_i-b^0_i]^2+\sum_{\text{bond angles i}}\frac{1}{2}K^\theta_i[\theta_i-\theta^0_i]^2\\
&+ \textstyle\sum_{\text{torsions i}}K^\varphi_i[1+cos[m_i\varphi_i-\delta_i]]+\sum_{\text{improper torsions i}}\frac{1}{2}K^\zeta_i[\zeta_i-\zeta^0_i]^2\\
&+ \textstyle\sum_a\sum_{a'>a}4\epsilon_{aa'}\left[\left(\frac{\sigma_{aa'}}{r_{aa'}}\right)^{12}-\left(\frac{\sigma_{aa'}}{r_{aa'}}\right)^{6}\right]+\sum_a\sum_{a'>a}\frac{1}{4\pi\epsilon_0\epsilon_{cs}}\frac{q_aq'_a}{r_{aa'}}\\
&+ \textstyle\sum_a\sum_{a'>a}\frac{q_aq_{a'}}{4\pi\epsilon_0\epsilon_{cs}}\frac{(-\frac{1}{2}C_{rf}r^2_{aa'})}{R^3_{rf}}+\sum_a\sum_{a'>a}\frac{q_aq_{a'}}{4\pi\epsilon_0\epsilon_{cs}}\frac{(\frac{1}{2}C_{rf}-1)}{R_{rf}}\Big)\\
&+ \left(\textstyle\sum_n\sum_a 4\epsilon_{na}\left[\left(\frac{\sigma_{na}}{r_{na}}\right)^{12}-\left(\frac{\sigma_{na}}{r_{na}}\right)^{6}\right]\right)\\
&+ \left(\textstyle\sum_n\sum_{n'}4\epsilon_{nn'}\left[\left(\frac{\sigma_{nn'}}{r_{nn'}}\right)^{12}-\left(\frac{\sigma_{nn'}}{r_{nn'}}\right)^{6}\right]\right),
\end{aligned} \tag{15.5}$$

where the first four terms describe covalent, bonded interactions with K denoting the particular force constant and the index ^{0} indicating an ideal bond length, angle or torsional angle value[166,167]. The next four terms describe the classical nonbonded interactions within the MM region, namely the van der Waals interactions in the form of a Lennard-Jones term, the electrostatic Coulomb interactions between (partial) atomic charges q, and the distance-dependent and distance-independent interaction terms for the reaction field with R_{rf} being the reaction field cutoff radius. The ninth term accounts for the aforementioned van der Waals interactions between the nuclei of the QM region and the atoms of the MM region and the last term specifies the van der Waals interactions between the nuclei of the QM region.

For the integration of Newton's equations of motion in molecular dynamics simulations, the force on each particle, i, is determined from the gradient of the total potential energy of the system at the position of particle i, where i can be either a QM nucleus or a MM atom,

$$\mathbf{f}_i=-\frac{\partial}{\partial\mathbf{r}_i}\mathcal{V}(\mathbf{r}^n,\mathbf{r}^a). \tag{15.6}$$

Using Eq. 15.4 and Eq. 15.5, the total force on particle i can be written as the sum of the different contributions from the QM and CM Hamiltonians, respectively,

$$\mathbf{f}_i=-\frac{\partial}{\partial\mathbf{r}_i}\mathcal{V}^{QM}(\mathbf{r}^n,\mathbf{r}^a)-\frac{\partial}{\partial\mathbf{r}_i}\mathcal{V}^{CM}(\mathbf{r}^n,\mathbf{r}^a), \tag{15.7}$$

where $\mathcal{V}^{\mathrm{QM}}$ is given by

$$\mathcal{V}^{QM}=\frac{\left\langle\Psi|\hat{\mathcal{H}}^{QM}_{QM}+\hat{\mathcal{V}}^{QM}_{QM/MM}|\Psi\right\rangle}{\langle\Psi|\Psi\rangle}. \tag{15.8}$$

The implementation of the GROMOS QM/MM scheme[165] is based on direct communication between GROMOS and the particular quantum-chemical program via code-integrated interfaces that execute system calls to the particular QM executable of MNDO[168] or TURBOMOLE[169], respectively. A general description of the object-oriented C++ architecture of the GROMOS software can be found in[170]. The QM/MM functionality of GROMOS was implemented in the new class, `QMMM_Interaction`, within the namespace *interaction*. At every MD step, single-point full-SCF calculations are carried out by the specified QM program, MNDO or TURBOMOLE. The GROMOS software automatically generates the program specific input files for the QM calculation including information about the type and position of the QM atoms as well as the position and charges of the MM atoms to be included in the QM Hamiltonian as external point charges. If periodic boundary conditions are applied, the system is gathered beforehand, with respect to the first atom of the list of QM atoms, in order to determine the MM atoms within the specified QM/MM cutoff radius for evaluation of the electrostatic interactions. The charge group based QM/MM cutoff radius is a variable input parameter (*RCUTQ*). Subsequent to the QM calculation, the QM energies and gradients are

extracted from the corresponding output files and converted to SI-like units which are generally used in the GROMOS data files.

15.3. Initialization, simulation and analysis

Preliminary to a GROMOS QM/MM simulation the following change with respect to a conventional GROMOS topology is needed: explicit hydrogen atoms are to be added for the part of the system to be treated quantum-mechanically (no united atoms). During the course of a GROMOS QM/MM simulation GROMOS topological information considering bonded and electrostatic interactions of atoms that are part of the defined QM region will not be used (see Eq. 15.5).

To apply the GROMOS QM/MM functionality the switch $NTQMMM$ has to be set to 1 with $NTQMSW$ being either 0 (MNDO) or 1 (TURBOMOLE). In QM/MM simulation the timestep should be chosen smaller (e.g. 0.5 fs) than in classical MD simulation. If $NTQMMM = 1$, an additional QMMM specification file has to be provided. This file has to contain the specific input parameters for the quantum-chemical program package to be used as well as the paths to their executables.

Analysis of the GROMOS QM/MM trajectories can be performed straightforward using the GROMOS++ program package.

CHAPTER 16

Replica Exchange (RE) Molecular Dynamics

16.1. Introduction

Replica exchange method (REMD)[171–174,175] is developed to enhance sampling of the conformational space. With this technique, a number of replicas of a system that do not interact with each other are simulated simultaneously. These replicas may represent different thermodynamic state points (T-REMD) or different Hamiltonians ($\mathcal{H}$-REMD).

For a system of $\mathcal{N}_a$ atoms of mass m_k (k = 1,...,$\mathcal{N}_a$) with coordinate $\boldsymbol{q} \equiv \{\boldsymbol{q}_1, ..., \boldsymbol{q}_{\mathcal{N}_a}\}$ and momentum vectors $\boldsymbol{p_q} \equiv \{\mathbf{p}_1, ..., \mathbf{p}_{\mathcal{N}_a}\}$, the Hamiltonian $\mathcal{H}(\boldsymbol{q}, \boldsymbol{p_q})$ is the sum of the kinetic energy $\mathcal{K}(\boldsymbol{p_q})$ and the potential energy $\mathcal{V}$ $(\boldsymbol{q})$:

$$\mathcal{H}(\boldsymbol{q}, \boldsymbol{p_q}) = \mathcal{K}(\boldsymbol{p_q}) + \mathcal{V}(\boldsymbol{q}) \tag{16.1}$$

where

$$\mathcal{K}(\boldsymbol{p_q}) = \sum_{k=1}^{\mathcal{N}_a} \frac{\mathbf{p}_k^2}{2m_k} \tag{16.2}$$

In the canonical ensemble of a REMD simulation, the state is specified as

$$\mathrm{X} = \{..., x_m^i, ..., x_n^j, ...\} \tag{16.3}$$

where $x \equiv (\boldsymbol{q}, \boldsymbol{p_q})$, x_m^i the ith replica simulated at the mth condition. The average kinetic energy at temperature T is given by

$$< \mathcal{K}(\boldsymbol{p_q}) >_T \; =< \sum_{k=1}^{\mathcal{N}_a} \frac{\mathbf{p}_k^2}{2m_k} >= \frac{3}{2}\mathcal{N}_a k_B T \tag{16.4}$$

The weight factors for this state is given by the product of Boltzmann factor for individual non-interacting replicas,

$$\mathrm{W_{RE}(X)} = exp\{-\sum_{i=1}^{M} \beta_{m(i)} \mathcal{H}_m(\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]})\} \tag{16.5}$$

During the simulations, after a predefined time interval, a Monte Carlo exchange between two replicas s' and s'' is attempted with exchange probability

$$P(s' \leftrightarrow s'') = \frac{\mathrm{W}(S'')}{\mathrm{W}(S')} = \frac{\mathrm{w}(S' \leftrightarrow S'')}{\mathrm{w}(S'' \leftrightarrow S')} = exp(-\Delta) \tag{16.6}$$

where Δ is defined as:

T-REMD:

$$\Delta = ((k_B T_{s'})^{-1} - (k_B T_{s''})^{-1})(U(x) - U(x)) \tag{16.7}$$

$\mathcal{H}$-REMD:

$$\Delta = [(U(x_{s''}; \lambda_{s'}) - U(x_{s'}; \lambda s')) - (U(x_{s''}; \lambda_{s''}) - U(x_{s'}; \lambda s''))]/(k_B T) \tag{16.8}$$

16.2. Temperature replica exchange MD

Using T-REMD, N independent copies (replicas) of the systems are propagated simultaneously at different fixed temperatures. At regular time intervals, pairs of replicas at successive temperatures are exchanged according to a Metropolis criterion allowing individual replicas to sample a range of temperatures. At higher temperatures the increased thermal energy facilitates the exploration of conformational space and allows the system to cross barriers that are difficult to cross at the temperature of interest on a given time scale.

Of the *M non-interacting* copies (or replicas) of the original system in the canonical ensemble (NVT) at M different temperatures T_m (m=1,...,M), there is a one-to-one correspondence between replicas (i=1,...,M) and temperatures (m=1,...,M),

$$\mathrm{X} = (x_1^{[i(1)]}, ..., x_M^{[i(M)]}) = (x_{m(1)}^{[1]}, ..., x_{m(M)}^{[M]}) \tag{16.9}$$

where X stands for a "state" in this generalized ensemble, and $x_m^{[i]}$ the replica i at the temperature T_m, and

$$x_m^{[i]} \equiv (\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]})_m \tag{16.10}$$

We can write a permutation function to show the one-to-one correspondence:

$$\begin{cases} i = i(m) \equiv f(m); \\ m = m(i) \equiv f^{-1}(i) \end{cases} \tag{16.11}$$

Since the replicas are non-interacting, the weight factor for the state X is given by the product of Boltzmann factors for each replica (or at each temperature):

$$\mathrm{W_{RE}(X)} = exp\{-\sum_{m=1}^{M} \beta_{m(i)} H(\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]})\} = exp\{-\sum_{i=1}^{M} \beta_m H(\boldsymbol{q}^{[i(m)]}, \boldsymbol{p_q}^{[i(m)]})\} \tag{16.12}$$

For the exchange of a pair of replicas i (at T_m) and j (at T_n),

$$\mathrm{X} = (..., x_m^{[i]}, ..., x_n^{[j]}, ...) \rightarrow \mathrm{X'} = (..., x_m^{[j]'}, ..., x_n^{[i]'}, ...) \tag{16.13}$$

The new permutation function f' is:

$$\begin{cases} i = f(m) \rightarrow j = f'(m); \\ j = f(n) \rightarrow i = f'(n) \end{cases} \tag{16.14}$$

In detail, the two exchanged replicas are

$$\begin{cases} x_m^{[i]} \equiv (\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]})_m \rightarrow x_m^{[j]'} \equiv (\boldsymbol{q}^{[j]}, \boldsymbol{p_q}^{[j]'})_m \\ x_n^{[j]} \equiv (\boldsymbol{q}^{[j]}, \boldsymbol{p_q}^{[j]})_n \rightarrow x_n^{[i]'} \equiv (\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]'})_n \end{cases} \tag{16.15}$$

This is equal to the exchange of a pair of temperatures T_m and T_n for the corresponding replicas i and j:

$$\begin{cases} x_m^{[i]} \equiv (\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]})_m \rightarrow x_n^{[i]'} \equiv (\boldsymbol{q}^{[i]}, \boldsymbol{p_q}^{[i]'})_n \\ x_n^{[j]} \equiv (\boldsymbol{q}^{[j]}, \boldsymbol{p_q}^{[j]})_n \rightarrow x_m^{[j]'} \equiv (\boldsymbol{q}^{[j]}, \boldsymbol{p_q}^{[j]'})_m \end{cases} \tag{16.16}$$

where

$$\begin{cases} \boldsymbol{p_q}^{[i]'} \equiv \sqrt{\frac{T_n}{T_m}} \boldsymbol{p_q}^{[i]}) \\ \boldsymbol{p_q}^{[j]'} \equiv \sqrt{\frac{T_m}{T_n}} \boldsymbol{p_q}^{[j]}) \end{cases} \tag{16.17}$$

This means that the velocities of all atoms in the replicas will be uniformly rescaled by a factor defined by the square root of the ratio of the two temperatures.

To ensure the convergence of this exchange process towards an equilibrium distribution, a detailed balance condition is imposed on the transition probability $\mathrm{w(X \rightarrow X')}$:

$$\mathrm{W_{RE}(X)\ w(X \rightarrow X') = W_{RE}(X')\ w(X' \rightarrow X)} \tag{16.18}$$

then

$$\begin{aligned} \tfrac{\mathrm{w(X \rightarrow X')}}{\mathrm{w(X' \rightarrow X)}} &= exp\{-\beta_m[\mathcal{K}(\boldsymbol{p_q}^{[j]'}) + \mathcal{V}(\boldsymbol{q}^{[j]})] - \beta_n[\mathcal{K}(\boldsymbol{p_q}^{[i]'}) + \mathcal{V}(\boldsymbol{q}^{[i]})] \\ &\quad +\beta_m[\mathcal{K}(\boldsymbol{p_q}^{[i]}) + \mathcal{V}(\boldsymbol{q}^{[i]})] + \beta_n[\mathcal{K}(\boldsymbol{p_q}^{[j]}) + \mathcal{V}(\boldsymbol{q}^{[j]})]\} \\ &= exp[-(\beta_n - \beta_m)(\mathcal{V}_i - \mathcal{V}_j)] \\ &= exp(-\Delta) \end{aligned} \tag{16.19}$$

and i, j, m, n are related by the permutation function before the exchange

$$i = f(m), j = f(n) \tag{16.20}$$

This can be satisfied by the usual Metropolis criterion:

$$\mathrm{w}(\mathrm{X} \to \mathrm{X}') \equiv \mathrm{w}(x_m^{[i]} \mid x_n^{[j]}) = \begin{cases} 1, & for \Delta \leq 0 \\ exp(-\Delta) & for \Delta > 0 \end{cases} \tag{16.21}$$

By driving replicas to explore temperature space, T-REMD allows the system to cross energetic barriers and access regions of the conformational space difficult to reach at low temperatures.

In order to ensure a uniform exchange probability the temperatures were chosen according to the relation:

$$T_i = T_0 \; exp(i \; c) \tag{16.22}$$

where T_0 and c can be varied to give a desired exchange ratio.

Assuming $\beta_1 < \beta_2 < ... < \beta_M$, a T-REMD simulation is then done by alternately performing the two steps:

1. each replica in canonical ensemble of the fixed temperature T_m is simulated simultaneously and independently for a certain number of MC or MD steps;
2. a pair of replicas at neighboring temperatures, say $x_m^{[i]}$ and $x_{m+1}^{[j]}$, are exchanged with the probability $\mathrm{w}(x_m^{[i]} \mid x_n^{[j]})$.

For optimal performance of T-REMD, it is necessary to choose an appropriate temperature distribution. This can be done through an iterative procedure.

For the average at any intermediate temperature of R independent simulations, one can use the multiple-histogram reweighting techniques (WHAM), and the average of a physical quantity A at any intermediate temperature $T = 1/k_B\beta$ is given by

$$< A >= \frac{\sum\limits_{\mathcal{V}} A(\mathcal{V}) P(\mathcal{V}; \beta)}{\sum\limits_{\mathcal{V}} P(\mathcal{V}; \beta)} \tag{16.23}$$

where

$$P(\mathcal{V}; \beta) = \frac{\sum\limits_{m=1}^{R} g_m^{-1} N_m(\mathcal{V}) e^{-\beta \mathcal{V}}}{\sum\limits_{m=1}^{R} n_m g_m^{-1} e^{f_m - \beta_m \mathcal{V}}} \tag{16.24}$$

and

$$e^{-f_m} = \sum_E P(\mathcal{V}; \beta_m) \tag{16.25}$$

Here $g_m = 1+2\tau_m$, and τ_m is the integrated autocorrelation time at temperature T_m. $N_m(\mathcal{V})$ and n_m are the energy histogram and the total number of samples obtained in the mth run, respectively. In T-REMD $n_m = N_{sim}$. $P(\mathcal{V}; \beta)$ in Eqs. 16.24 and 16.25 are solved self-consistently by iteration.

16.2.1. Simulation checks. To examine whether a replica-exchange simulation indeed performed properly, there are three points to check:[171]

- whether the temperatures are optimally distributed;
  the optimal temperature distributions imply that all the acceptance ratios are the same, resulting in a free random walk in the replica temperature space.
- whether the number of replica temperatures is sufficient;
  the number of replica temperatures is sufficient if the acceptance ratios are not too small, say, greater than 0.1.
- whether the highest temperature is sufficiently high so that no trapping in a local energy-minimum occurs;

16.2.2. Factors determining the efficiency. T-REMD has been shown to be efficient at the lowest temperature, and meanwhile it provides converged distributions over the range of temperatures used in T-REMD simulations.[175] The efficiency and convergence are determined by the mixing/sorting of the replicas, which is the source of the main gain of this method, and can also bring false precision and be misinterpreted as true convergence or increased sampling efficiency.[176].

A sensible choice of intervals between exchange trials, ideally equal to the correlation time of the potential energy following an exchange, strongly depends on the size of the system. For small systems, the exchanges were mainly determined by the fluctuations within the solvent rather than the conformation of the solute, which is not the case for large systems, and for large systems, the conformation of the solute may require much longer time to reach the convergence.[127]

16.3. Hamiltonian replica exchange MD

Hamiltonian replica exchange MD can be realized by means of perturbation of the force field. Hamiltonians can be perturbed in different ways, see Sec. 14.2. In GROMOS, Hamiltonian replica exchange MD is performed by assigning different values of the coupling parameter λ to the different replicas. Note that the use of individual λ-values (Sec. 14.4) offers additional flexibility in this definition. For example, simulations may be set in which the replicas differ only in the softness of specific interactions, but not in the nonbonded interaction parameters.[174]

Use of $\mathcal{H}$-REMD in free-energy calculations using e.g. thermodynamic integration can improve the convergence, in particular when slow relaxation of some degrees of freedom is to be expected.[177] A condition for improved convergence is that the barriers leading to slow relaxation are absent in at least one replica. By using individual λ-values, such a replica may be explicitly designed. Furthermore, to improve the chances of relaxation in this replica, multiple replicas at the same value of λ may be included.[174]

While in T-REMD the optimal distribution of temperatures may be derived using the iterative procedures outlined above, this may be more cumbersome for $\mathcal{H}$-REMD. Using some preliminary simulations, one may design a mimicking approach from which the optimal sampling efficiency may be obtained.[178]

16.4. Initialization, simulation and analysis

In GROMOS, REMD is implemented through the program `repex_mpi`. This will start the replicas as separate MPI processes and control all the file handling and exchanges.

16.4.1. Set up of a RE simulation. The time between replica-exchange switches is set by the switch NSTLIM in the STEP block. Specific settings for the REMD simulation are defined in the REPLICA block. To do a T-REMD, the total number of temperatures is defined by the switch NRET and the temperature of each replica by RET:

`NRET`	number of replica exchange temperatures
`RET(1..NRET)`	temperature for each replica

and the scaling of temperature can be achieved by the switch LRESCALE:

`LRESCALE`	0: don't scale temperatures after exchange trial 1: scale temperatures after exchange trial

To do $\mathcal{H}$-REMD, one needs to define the number of λ-values by NRELAM, and the λ-value of each replica by RELAM. In addition, the timestep for each λ-replica can be specified separately, to allow for changes in the grain-level between the replicas:[51]

`NRELAM`	number of replica exchange lambda values
`RELAM(1..NRELAM)`	lambda value for each lambda-replica
`RETS(1..NRELAM)`	timestep of every λ-replica

Furthermore, the following switches in the REPLICA block control the REMD simulation

`NRETRIAL`	number of overall exchange trials
`NREQUIL`	number of exchange periods to equilibrate; no switches are performed in the first NREQUIL exchange periods
`CONT`	specifies if the simulation is a continuation from a previous REMD simulation. 0: input coordinates for all replicas are read from a single file 1: input coordinates for all replicas are read from separate files (see below)

Replica exchanges are attempted every NSTLIM steps. In `repex_mpi`, switches are only attempted between neighbouring replicas (in T for T-REMD, or in λ for $\mathcal{H}$-REMD). At odd trial attempts (i.e. after NSTLIM, 3xNSTLIM, 5xNSTLIM, ... timesteps), switches are attempted between replica pairs (1,2), (3,4), etc. At even trial attempts (i.e. after 2xNSTLIM, 4xNSTLIM, ... timesteps) switches are attempted between replica pairs (2,3), (4,5), etc.

The simulation may start with a single structure as the starting coordinates for all replicas, or each replica with a specified structure individually. If CONT = 0 in the REPLICA block, `repex_mpi` will read a single coordinate file as starting structure for all replicas. If CONT = 1 in the REPLICA block, `repex_mpi` will take the input parameter specified for the `@conf` flag, add the replica number to the filename and try to read the structure for every replica individually.

16.4.2. Analysis of a RE **trajectory.** For all output files, `repex_mpi` takes the filename as specified by the appropriate option and adds the replica number to the specified filename. Standard output for the individual replicas is written to files, specified by the `@repout` option. Furthermore, through the `@repdata` option an additional output-file is specified, which gives an overview of the switching attempts, the relevant energies, switching probabilities and switching acceptances.

The configurations generated in a REMD simulation of M replicas can be analyzed to obtain equilibrium averages for a target distribution P, where we can further derive an equilibrium property with eq. Eq. 16.23. Note that the trajectory files written out by `repex_mpi` contain the data pertaining to a single state of the simulation (temperature or λ-value), i.e. the coordinates and velocities are not continuous in time.

It is also possible to obtain kinetic information from a REMD simulation.[179] Due to the discontinuous feature of REMD simulations, it is necessary to use short-time propagators for the analysis, concerning that it is possible to calculate short-time correlation functions accurately with REMD. For a replica in $T-$REMD or $\mathcal{H}-$REMD, the maximum time scale is given by the simulation time between two accepted exchanges. In the sense of conformational transitions, a master equation may be constructed by dividing the whole space into N states,

$$\frac{dP_i(t)}{dt} = \sum_{j=1}^{N}[k_{ij}P_j(t) - k_{ji}P_i(t)] \tag{16.26}$$

where $P_i(t)$ the population of state i, $k_{ij} \geq 0$ the transition rate from states j to i.

The Eq. 16.26 may be rewriten as

$$\frac{d\mathbf{P}(t)}{dt} = \mathbf{K}\mathbf{P}(t) \tag{16.27}$$

where $\mathbf{K}$ is the transition rate matrix, and its diagonal elements are

$$k_{ii} = -\sum_{j\neq i} k_{ji} < 0 \tag{16.28}$$

The propagators are defined as the probability of being in state i at time 0 and j at time t and have the form

$$p(j,t|i,0) = [exp(\mathbf{K}t)]_{ji} \tag{16.29}$$

The elements of the matrix $\mathbf{K}$ may be estimated from a maximum-likelihood procedure. First the number of transitions N_{ji} from state i to j is determined within a time interval Δt, then the coefficients of the master

equation may estimated from the likelihood maximization,[180] where the logarithm of likelihood is defined as

$$ln\Lambda = \sum_{i=1}^{N}\sum_{j=1}^{N} N_{ji} ln\ p(j, \Delta t|i, 0) \tag{16.30}$$

The above approach has been used to study the transition rates of a helical peptide in water between its microscopic conformational states or between folding and unfolding.[179]

CHAPTER 17

Derivatives of the force-field terms

17.1. Bond stretching force-field term

Quartic case: the forces on atoms i and j due to formula (Eq. 5.5) are

$$\begin{aligned} \mathbf{f}_i &= -\frac{\partial V_n^{bond,q}}{\partial b_n^2}\,\frac{\partial b_n^2}{\partial \mathbf{r}_i} \\ &= -K_{b_n}\,[b_n^2 - b_{0_n}^2]\mathbf{r}_{ij} \end{aligned} \tag{17.1}$$

and

$$\mathbf{f}_j = -\mathbf{f}_i\ . \tag{17.2}$$

Harmonic case: the forces on atoms i and j due to formula (Eq. 5.6) are

$$\begin{aligned} \mathbf{f}_i &= -\frac{\partial V_n^{bond}}{\partial b_n}\,\frac{\partial b_n}{\partial \mathbf{r}_i} \\ &= -K_{b_n}^{harm}\,[b_n - b_{0_n}]\frac{\mathbf{r}_{ij}}{r_{ij}} \end{aligned} \tag{17.3}$$

and

$$\mathbf{f}_j = -\mathbf{f}_i. \tag{17.4}$$

17.2. Bond-angle bending force-field term

Cosine-harmonic form:

The forces on atoms i, j and k due to the n-th term in Eq. 5.11 are

$$\begin{aligned} \mathbf{f}_i &= -\frac{\partial V^{(\theta,c)}}{\partial \cos\theta_n}\,\frac{\partial \cos\theta_n}{\partial \mathbf{r}_i} \\ &= -k_n^{(\theta,c)}\ [\ \cos\theta_n - \cos\theta_{0_n}\ ]\ \left[\ \frac{\mathbf{r}_{kj}}{r_{kj}} - \frac{\mathbf{r}_{ij}}{r_{ij}}\ \cos\theta_n\ \right]\ \frac{1}{r_{ij}} \end{aligned} \tag{17.5}$$

and

$$\begin{aligned} \mathbf{f}_k &= -\frac{\partial V^{(\theta,c)}}{\partial \cos\theta_n}\,\frac{\partial \cos\theta_n}{\partial \mathbf{r}_k} \\ &= -k_n^{(\theta,c)}\ [\ \cos\theta_n - \cos\theta_{0_n}\ ]\ \left[\ \frac{\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{kj}}{r_{kj}}\ \cos\theta_n\ \right]\ \frac{1}{r_{kj}} \end{aligned} \tag{17.6}$$

and

$$\mathbf{f}_j\ =\ -\mathbf{f}_i\ -\mathbf{f}_k \tag{17.7}$$

Angle-harmonic form:

For harmonic bond angles the forces on atoms i, j and k due to the n-th term in Eq. 5.12 are

$$\mathbf{f}_i = k_n^{(\theta,h)} \cdot \frac{\theta - \theta_0}{\sin\theta} \cdot \frac{1}{r_{ij}} \left(\frac{\mathbf{r}_{kj}}{r_{kj}} - \frac{\mathbf{r}_{ij}}{r_{ij}} \cos\theta \right) \tag{17.8}$$

and

$$\mathbf{f}_k = k_n^{(\theta,h)} \cdot \frac{\theta - \theta_0}{\sin\theta} \cdot \frac{1}{r_{kj}} \left(\frac{\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{kj}}{r_{kj}} \cos\theta \right) \tag{17.9}$$

and

$$\mathbf{f}_j = -\mathbf{f}_i - \mathbf{f}_k \tag{17.10}$$

17.3. Improper dihedral-angle bending force-field term

The forces on atoms i, j, k and l due to the n-th term in formula (Eq. 5.17)

$$\begin{aligned} \mathbf{f}_i &= -\frac{\partial V^{(\xi)}}{\partial \xi_n}\ \frac{\partial \xi_n}{\partial \mathbf{r}_i} \\ &= -k_n^{(\xi)}\ [\ \xi_n\ -\xi_{0_n}]\ \frac{r_{kj}}{r_{mj}^2}\ \mathbf{r}_{mj} \end{aligned} \tag{17.11}$$

$$\begin{aligned} \mathbf{f}_l &= -\frac{\partial V^{(\xi)}}{\partial \xi_n}\ \frac{\partial \xi_n}{\partial \mathbf{r}_l} \\ &= +k_n^{(\xi)}\ [\ \xi_n\ -\xi_{0_n}]\ \frac{r_{kj}}{r_{nk}^2}\ \mathbf{r}_{nk} \end{aligned} \tag{17.12}$$

$$\begin{aligned} \mathbf{f}_j &= -\frac{\partial V^{(\xi)}}{\partial \xi_n}\ \frac{\partial \xi_n}{\partial \mathbf{r}_j} \\ &= \left[\frac{(\mathbf{r}_{ij}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2}\ -1\right]\ \mathbf{f}_i\ -\frac{(\mathbf{r}_{kl}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2}\ \mathbf{f}_l \end{aligned} \tag{17.13}$$

and

$$\mathbf{f}_k\ =\ -\mathbf{f}_i\ -\mathbf{f}_j\ -\mathbf{f}_l \tag{17.14}$$

17.4. Proper dihedral-angle torsion force-field term

The cosine expansions in terms of derivatives ($\partial cos(m\varphi)/\partial cos\varphi$):

$$\begin{aligned} &\partial\cos(0\varphi)/\partial\cos\varphi = 0 \\ &\partial\cos(1\varphi)/\partial\cos\varphi = 1 \\ &\partial\cos(2\varphi)/\partial\cos\varphi = 4\cos\varphi \\ &\partial\cos(3\varphi)/\partial\cos\varphi = 12\cos^2\varphi - 3 \\ &\partial\cos(4\varphi)/\partial\cos\varphi = 32\cos^3\varphi - 16\cos\varphi \\ &\partial\cos(5\varphi)/\partial\cos\varphi = 80\cos^4\varphi - 60\cos^2\varphi + 5 \\ &\partial\cos(6\varphi)/\partial\cos\varphi = 192\cos^5\varphi - 192\cos^3\varphi + 36\cos\varphi \end{aligned} \tag{17.15}$$

Symmetric form:

The forces on atoms i, j, k and l due to the n-th term in formula (Eq. 5.22 are

$$\mathbf{f}_i = -\frac{\partial V^{(\varphi,s)}}{\partial\cos(m_n\ \varphi_n)}\ \frac{\partial\cos(m_n\ \varphi_n)}{\partial\cos\varphi_n}\ \frac{\partial\cos\varphi_n}{\partial\mathbf{r}_i} \tag{17.16}$$

$$= -k_n^{(\varphi,s)}\cos(\delta_n)\frac{\partial\cos(m_n\ \varphi_n)}{\partial\cos\varphi_n}\ \left[\frac{\mathbf{r}_{ln'}}{r_{ln'}} - \frac{\mathbf{r}_{im'}}{r_{im'}}\ \cos\varphi_n\right]\ \frac{1}{r_{im'}} \tag{17.17}$$

$$\mathbf{f}_l = -k_n^{(\varphi,s)}\cos(\delta_n)\frac{\partial\cos(m_n\ \varphi_n)}{\partial\cos\varphi_n}\ \left[\frac{\mathbf{r}_{im'}}{r_{im'}}\ -\frac{\mathbf{r}_{ln'}}{r_{ln'}}\cos\varphi_n\right]\frac{1}{r_{ln'}} \tag{17.18}$$

$$\mathbf{f}_j = \left[\frac{(\mathbf{r}_{ij}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2}\ -1\right]\ \mathbf{f}_i\ -\frac{(\mathbf{r}_{kl}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2}\ \mathbf{f}_l \tag{17.19}$$

and

$$\mathbf{f}_k\ =\ -\mathbf{f}_i\ -\mathbf{f}_j\ -\mathbf{f}_l \tag{17.20}$$

Generalized form:

When doing so, the forces on atoms i, j, k and l due to the n-th term in (Eq. 5.23) are

$$\mathbf{f}_i = -\frac{\partial V^{(\varphi,g)}}{\partial \varphi_n}\frac{\partial \varphi_n}{\partial \mathbf{r}_i} \tag{17.21}$$

$$= k_n^{(\varphi,g)} m_n sin(m_n\varphi_n - \delta_n)\frac{r_{kj}}{r_{mj}^2}\ \mathbf{r}_{mj} \tag{17.22}$$

$$\mathbf{f}_l = -\frac{\partial V^{(\varphi,g)}}{\partial \varphi_n}\frac{\partial \varphi_n}{\partial \mathbf{r}_l} \tag{17.23}$$

$$= -k_n^{(\varphi,g)} m_n sin(m_n\varphi_n - \delta_n)\frac{r_{kj}}{r_{nk}^2}\ \mathbf{r}_{nk} \tag{17.24}$$

$$\mathbf{f}_j = -\frac{\partial V^{(\varphi,g)}}{\partial \varphi_n}\frac{\partial \varphi_n}{\partial \mathbf{r}_j} \tag{17.25}$$

$$= \left[\frac{(\mathbf{r}_{ij}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2} - 1\right]\ \mathbf{f}_i - \frac{(\mathbf{r}_{kl}\ \cdot\ \mathbf{r}_{kj})}{r_{kj}^2}\ \mathbf{f}_l$$

and

$$\mathbf{f}_k = -\mathbf{f}_i - \mathbf{f}_j - \mathbf{f}_{ll} \tag{17.26}$$

with $\mathbf{r}_{mj}$ and $\mathbf{r}_{nk}$ calculated according to Eq. 5.15, and

$$sin(m_n\varphi_n - \delta_n) = \text{sign}\ \sqrt{1 - cos^2(m_n\varphi_n - \delta_n)} \tag{17.27}$$

17.5. LJ interaction terms

The forces on atoms i and j due to van der Waals interactions are

$$\mathbf{f}_i^{LJ} = \left[\frac{2C_{12}(i,j)}{\mathbf{r}_{ij}^6} - C_6(i,j)\right]\cdot\frac{6\mathbf{r}_{ij}}{r_{ij}^8} \tag{17.28}$$

and

$$\mathbf{f}_j = -\mathbf{f}_i \quad . \tag{17.29}$$

17.6. Electrostatic interaction terms: Coulomb plus reactive field

The forces on atoms i and j due to Coulomb and reaction field are

$$\mathbf{f}_i^{CRF} = \frac{q_i q_j}{4\pi\epsilon_0\epsilon_{cs}}\left[\frac{1}{r_{ij}^3} + \frac{C_{rf}}{{R_{rf}}^3}\right]\mathbf{r}_{ij} \tag{17.30}$$

and

$$\mathbf{f}_j = -\mathbf{f}_i \quad . \tag{17.31}$$

17.7. Electrostatic interaction terms: lattice sum

The force on atom i corresponding to Eq. 7.44 is given by

$$\mathbf{F}_{\eta,i} = (4\pi\epsilon_0)^{-1}\sum_{j=1\,,\,j\neq i\,,\,\overline{r}_{ij}<R_p\,,\,j\notin Exc(i)}^{N_q} q_i\,q_j\,[\overline{r}_{ij}^{-1}\eta(a^{-1}\overline{r}_{ij}) - a^{-1}\eta'(a^{-1}\overline{r}_{ij})]\overline{r}_{ij}^{-2}\overline{\mathbf{r}}_{ij} \tag{17.32}$$

$$+(4\pi\epsilon_0)^{-1}\sum_{i=1}^{N_q}\sum_{j=1\,,\,j>i\,,\,j\in Exc(i)}^{N_q} q_i\,q_j\,\{\overline{r}_{ij}^{-1}[\eta(a^{-1}\overline{r}_{ij}) - 1] - a^{-1}\eta'(a^{-1}\overline{r}_{ij})\}\overline{r}_{ij}^{-2}\overline{\mathbf{r}}_{ij}\ .$$

The derivatives η' of the switch functions corresponding to the charge-shaping functions implemented are listed in Tab. 7.3.

The force on atom i corresponding to Eq. 7.45 is given by

$$\mathbf{F}_{\gamma,i} = (\epsilon_0 V)^{-1} q_i \sum_{\mathbf{l}\in\mathbb{W}\,,\,l\neq 0} k^{-2}\,\hat{\gamma}(ak)\,\mathbf{k}\,[\,C(\mathbf{k})\sin\mathbf{k}\cdot\mathbf{r}_i - S(\mathbf{k})\cos\mathbf{k}\cdot\mathbf{r}_i\,]\;. \tag{17.33}$$

Eq. 17.33 is rewritten

$$\mathbf{F}_{\gamma,i} = 8(\epsilon_0 V)^{-1} q_i \sum_{\mathbf{l}\in\mathbb{W}'\,,\,l\neq 0} k^{-2}\,\hat{\gamma}(ak)\,\sigma_{\mathbf{k}}\,\underline{\mathbf{D}}\mathbf{k} \tag{17.34}$$

where $\underline{\mathbf{D}}$ is a diagonal matrix with elements

$$\begin{aligned} D_{xx} \;=\;& -c_{x,i,l_x}c_{y,i,l_y}[S_x(\mathbf{k})c_{z,i,l_z} + C_y(\mathbf{k})s_{z,i,l_z}] + s_{x,i,l_x}s_{y,i,l_y}[S_y(\mathbf{k})c_{z,i,l_z} + C_x(\mathbf{k})s_{z,i,l_z}] \\ & -c_{x,i,l_x}s_{y,i,l_y}[C_z(\mathbf{k})c_{z,i,l_z} + S_o(\mathbf{k})s_{z,i,l_z}] + s_{x,i,l_x}c_{y,i,l_y}[C_o(\mathbf{k})c_{z,i,l_z} + S_z(\mathbf{k})s_{z,i,l_z}]\;, \end{aligned} \tag{17.35}$$

$$\begin{aligned} D_{yy} \;=\;& -c_{x,i,l_x}c_{y,i,l_y}[S_y(\mathbf{k})c_{z,i,l_z} + C_x(\mathbf{k})s_{z,i,l_z}] + s_{x,i,l_x}s_{y,i,l_y}[S_x(\mathbf{k})c_{z,i,l_z} + C_y(\mathbf{k})s_{z,i,l_z}] \\ & +c_{x,i,l_x}s_{y,i,l_y}[C_o(\mathbf{k})c_{z,i,l_z} + S_z(\mathbf{k})s_{z,i,l_z}] - s_{x,i,l_x}c_{y,i,l_y}[C_z(\mathbf{k})c_{z,i,l_z} + S_o(\mathbf{k})s_{z,i,l_z}]\;, \end{aligned} \tag{17.36}$$

and

$$\begin{aligned} D_{zz} \;=\;& -c_{x,i,l_x}c_{y,i,l_y}[S_z(\mathbf{k})c_{z,i,l_z} - C_o(\mathbf{k})s_{z,i,l_z}] - s_{x,i,l_x}s_{y,i,l_y}[S_o(\mathbf{k})c_{z,i,l_z} - C_z(\mathbf{k})s_{z,i,l_z}] \\ & -c_{x,i,l_x}s_{y,i,l_y}[C_x(\mathbf{k})c_{z,i,l_z} - S_y(\mathbf{k})s_{z,i,l_z}] - s_{x,i,l_x}c_{y,i,l_y}[C_y(\mathbf{k})c_{z,i,l_z} - S_x(\mathbf{k})s_{z,i,l_z}]\;. \end{aligned} \tag{17.37}$$

The reciprocal-space Ewald contribution to the atomic virial W_μ (corresponding to the energy term E_γ defined by Eqs. 7.26 and evaluated as Eq. 7.45) reads[118]

$$W_{ew,\mu} = -(4\epsilon_0\epsilon_{ls}V)^{-1} \sum_{\mathbf{l}\in\mathbb{W}\,,\,l\neq 0} \hat{\Gamma}_\mu(\mathbf{k})\,[\,C^2(\mathbf{k}) + S^2(\mathbf{k})\,]\;, \tag{17.38}$$

where the $C(\mathbf{k})$ and $S(\mathbf{k})$ functions are given by Eq. 7.46, and

$$\hat{\Gamma}_\mu(\mathbf{k}) = \frac{(k^2 - 2k_\mu^2)\hat{\gamma}(ak) + akk_\mu^2\hat{\gamma}'(ak)}{k^4}\;, \tag{17.39}$$

with $\hat{\gamma}'(\kappa) = \partial\hat{\gamma}(\kappa)/\partial\kappa$ (Tab. 7.5). As discussed in Sec. 7.4.3, an increase in computational efficiency can be obtained by restricting the summation to a half-space and doubling the result, or even, summing over an octant and multiplying the result by eight.

The virial associated with the energy contribution E_γ (Eq. 7.72) can be calculated as

$$\underline{\mathbf{W}}_\gamma = \frac{1}{2}\frac{\partial E_\gamma}{\partial\underline{\mathbf{L}}}\,{}^t\underline{\mathbf{L}}\;. \tag{17.40}$$

Inserting Eq. 7.72 into Eq. 17.40, noting that $\hat{s}_g$ is independent of $\underline{\mathbf{L}}$ when the particle coordinates are scaled together with the box dimensions, and making use of Eq. 4.47, one obtains

$$\underline{\mathbf{W}}_\gamma = -(4\epsilon_0\,V)^{-1} \sum_{\mathbf{l}\in G\,,\,l\neq 0} [\hat{G}_g^\dagger(\mathbf{k_l})\underline{\mathbf{1}} + \widehat{\underline{\mathbf{\Gamma}}}_g^{\,o}(\mathbf{k_l})]\mid\hat{s}_g(\mathbf{k_l})\mid^2 \tag{17.41}$$

where $\hat{G}_g^\dagger(\mathbf{k_l})$ and $\widehat{\underline{\mathbf{\Gamma}}}_g^{\,o}(\mathbf{k_l})$ are defined as in Eqs. 7.69 and 7.70.

CHAPTER 18

Appendices

18.1. Conversion of force constants: bond-stretching and bond-angle bending interactions

In GROMOS the bond-stretching interaction may be chosen harmonic,

$$V_n^{(b,h)}(b_n, k_n^{(b,h)}, b_n^0) = \frac{1}{2} k_n^{(b,h)} (b_n - b_n^0)^2 \tag{18.1}$$

or quartic,

$$V_n^{(b,q)}(b_n, k_n^{(b,q)}, b_n^0) = \frac{1}{4} k_n^{(b,q)} ({b_n}^2 - {b_n^0}^2)^2 = \frac{1}{4} k_n^{(b,q)} (b_n + b_n^0)^2 (b_n - b_n^0)^2 \quad . \tag{18.2}$$

Comparison of the two forms leads to the conversion formula

$$k_n^{(b,h)} = 2 {b_n^0}^2 k_n^{(b,q)} \tag{18.3}$$

because generally $b_n \simeq b_n^0$, and

$$k_n^{(b,q)} = \frac{k_n^{(b,h)}}{2 {b_n^0}^2} \tag{18.4}$$

In GROMOS the bond-angle bending interaction may be chosen harmonic,

$$V_n^{(\theta,h)}(\theta_n; k_n^{(\theta,h)}; \theta_n^0) = \frac{1}{2} k_n^{(\theta,h)} (\theta_n - \theta_n^0)^2 \tag{18.5}$$

or cosine-harmonic,

$$V_n^{(\theta,c)}(\theta_n; k_n^{(\theta,c)}; \theta_n^0) = \frac{1}{2} k_n^{(\theta,c)} (\cos\theta_n - \cos\theta_n^0)^2 \quad . \tag{18.6}$$

The force constants of the two forms can be related by the requirement that for the two angles θ_n for which

$$V_n^{(\theta,h)}(\theta_n; k_n^{(\theta,h)}; \theta_n^0) = V_n^{(\theta,c)}(\theta_n; k_n^{(\theta,c)}; \theta_n^0) \tag{18.7}$$

the average of the energies (Eq. 18.7) for the two angles θ_n should be equal to $\frac{1}{2} k_B T$.

Eq. 18.5 yields for the two θ_n angles

$$\theta_n = \theta_n^0 \pm \left(\frac{k_B T}{k_n^{(\theta,h)}} \right)^{\frac{1}{2}} \tag{18.8}$$

which after insertion into Eq. 18.6 and averaging over the two angles θ_n yields

$$k_n^{(\theta,c)} = \frac{2 k_B T}{\left[\cos\left(\theta_n^0 + \left(\frac{k_B T}{k_n^{(\theta,h)}} \right)^{\frac{1}{2}} \right) - \cos\theta_n^0 \right]^2 + \left[\cos\left(\theta_n^0 - \left(\frac{k_B T}{k_n^{(\theta,h)}} \right)^{\frac{1}{2}} \right) - \cos\theta_n^0 \right]^2} \tag{18.9}$$

Of course if $k_n^{(\theta,h)} = 0$, then $k_n^{(\theta,c)} = 0$ and Eq. 18.9 should not be used. An inverse relation can be derived analogously. Eq. 18.6 yields for the two θ_n angles

$$\theta_n = \begin{cases} \arccos\left(\cos\theta_n^0 \pm \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) & \text{if } -1 \leq \cos\theta_n^0 \pm \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}} \leq 1 \\ \begin{cases} \arccos\left(\cos\theta_n^0 + \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) \\ 2\theta_n^0 - \arccos\left(\cos\theta_n^0 + \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) \end{cases} & \text{if } -1 \leq \cos\theta_n^0 + \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}} \leq 1 \\ \begin{cases} \arccos\left(\cos\theta_n^0 - \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) \\ 2\theta_n^0 - \arccos\left(\cos\theta_n^0 - \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) \end{cases} & \text{if } -1 \leq \cos\theta_n^0 - \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}} \leq 1 \\ \text{not defined} & \text{else} \end{cases} \tag{18.10}$$

which after insertion into Eq. 18.5 and averaging over the two angles θ_n yields

$$k_n^{(\theta,h)} = \frac{2k_B T}{\left[\arccos\left(\cos\theta_n^0 + \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) - \theta_n^0\right]^2 + \left[\arccos\left(\cos\theta_n^0 - \left(\frac{k_B T}{k_n^{(\theta,c)}}\right)^{\frac{1}{2}}\right) - \theta_n^0\right]^2} \tag{18.11}$$

Again if $k_n^{(\theta,c)} = 0$, $k_n^{(\theta,h)} = 0$ and Eq. 18.11 should not be used.

Eq. 18.11 is the physical inverse of Eq. 18.9, *not* the mathematical one.

For $k_B T$ one can chose the values $k_B = 8.3144110^{-3}\,\mathrm{kJ\,mol^{-1}\,K^{-1}}$ and $T = 300\,\mathrm{K}$, leading to $2.494323\,\mathrm{kJ\,mol^{-1}}$ or $0.5961575\,\mathrm{kcal\,mol^{-1}}$.

Bibliography

[1] W.F. van Gunsteren and H.J.C. Berendsen. Computer Simulation of Molecular Dynamics: Methodology, Applications and Perspectives in Chemistry. *Angew. Chem. Int. Ed.*, 29:992–1023, 1990.

[2] W.F. van Gunsteren, D. Bakowies, R. Baron, I. Chandrasekhar, M. Christen, X. Daura, P. Gee, D.P. Geerke, A. Glättli, P.H. Hünenberger, M.A. Kastenholz, C. Oostenbrink, M. Schenk, D. Trzesniak, N.F.A. van der Vegt, and H.B. Yu. Biomolecular modelling: goals, problems, perspectives Angew. Chem ¡strong¿118¡/strong¿ (2006) 4168-4198. *Angew. Chem. Int. Ed.*, 45:4064–4092, 2006.

[3] K. Meier, A. Choutko, J. Dolenc, A.P. Eichenberger, S. Riniker, and W.F. van Gunsteren. Multi-resolution simulation of biomolecular systems: a review of methodological issues. *Angew. Chem. Int. Ed.*, 52:2820–2834, 2013.

[4] H. J. C. Berendsen. *Simulating the Physical World, Hierarchical Modeling from Quantum Mechanics to Fluid Dynamics*. Cambridge University Press, 2007.

[5] W.F. van Gunsteren, T. Huber, and A.E. Torda. Biomolecular Modelling: Overview of Types of Methods to Search and Sample Conformational Space. *European Conference on Computational Chemistry (E.C.C.C 1), American Institute of Physics, Conference Proceedings*, 330:253–268, 1995.

[6] M. Christen and W.F. van Gunsteren. On searching in, sampling of, and dynamically moving through conformational space of biomolecular systems: a review. *J. Comput. Chem.*, 29:157–166, 2007.

[7] P.H. Hünenberger. Thermostat Algorithms for Molecular-Dynamics Simulations. *Adv. Polym. Sci.*, 173:105–149, 2005.

[8] M. Parrinello R. Car. Unified Approach for Molecular Dynamics and Density-Functional Theory. *Phys. Rev. Lett.*, 55:2471–2474, 1985.

[9] H. Yu and W.F. van Gunsteren. Accounting for polarization in molecular simulation. *Comput. Phys. Commun.*, 172:69–85, 2005.

[10] S.R. Billeter, P.M. King, and W.F. van Gunsteren. Can the density maximum of water be found by computer simulation? *J. Chem. Phys.*, 100:6692–6699, 1994.

[11] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, A. DiNola, and J.R. Haak. Molecular dynamics with coupling to an external bath. *J. Chem. Phys.*, 81:3684–3690, 1984.

[12] W.F. van Gunsteren, P.M. King, and A.E. Mark. Fundamentals of drug design from a biophysical viewpoint. *Quart. Rev. Biophysics*, 27:435–481, 1994.

[13] T. Huber, A.E. Torda, and W.F. van Gunsteren. Optimization Methods for Conformational Sampling Using a Boltzmann-Weighted Mean Field Approach. *Biopolymers*, 39:103–114, 1996.

[14] W.F. van Gunsteren, H.J.C. Berendsen, F. Colonna, D. Perahia, J.P. Hollenberg, and D. Lellouch. On Searching Neighbours in Computer Simulations of Macromolecular Systems. *J. Comput. Chem.*, 5:272–279, 1984.

[15] H.J.C. Berendsen. Treatment of Long-Range Forces in Molecular Dynamics. In J. Hermans, editor, *Molecular Dynamics and Protein Structure*, pages 18–22. Polycrystal Book Service, Western Springs, Ill, USA, 1985.

[16] J. L. Finney. Long-range forces in molecular dynamics calculations on water. *J. Comput. Chem.*, 28:92–102, 1978.

[17] W. B. Streett; D. J. Tildesley; G. Saville. Multiple time-step methods in molecular dynamics Molecular Physics: An International Journal at the Interface Between Chemistry and Physics. *Mol. Phys.*, 35:639–648, 1978.

[18] P. G. Debenedetti A. A. Chialvo. An automated Verlet neighbor list algorithm with a multiple time-step approach for the simulation of large systems. *Comput. Phys. Commun.*, 70:467–477, 1992.

[19] T. Heinz and P.H. Hünenberger. A fast pairlist construction algorithm for molecular simulations under periodic boundary conditions. *J. Comput. Chem.*, 25:1474, 2004.

[20] H. Bekker, H.J.C. Berendsen, E.J. Dijkstra, S. Achterop, R. v. Drunen, D. v.d. Spoel, A. Sijbers, H. Keegstra, B. Reitsma, and M.K.R. Renardus. GROMACS Method of Virial Calculation Using a Single Sum. In R.A. de Groot and J. Nadrchal, editors, *Proceedings of the 4th Intl. Conference Physics Computing '92*, pages 257–261. World Scientific Publishing Company, Singapore, 1993.

[21] W.F. van Gunsteren and M. Karplus. Protein Dynamics in Solution and in a Crystalline Environment: A Molecular Dynamics Study. *Biochemistry*, 21:2259–2274, 1982.

[22] F. Fraternali and W.F. van Gunsteren. An Efficient Mean Solvation Force Model for Use in Molecular Dynamics Simulations of Proteins in Aqueous Solution. *J. Mol. Biol.*, 256:939–948, 1996.

[23] H. Bekker. Unification of box shapes in molecular simulations. *J. Comput. Chem.*, 18:1930–1942, 1997.

[24] C. Oostenbrink, A. Villa, A.E. Mark, and W.F. van Gunsteren. A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. *J. Comput. Chem.*, 25:1656, 2004.

[25] IUPAC-IUB commission on biochemical nomenclature. Abbreviations and symbols for the description of the conformation of polypeptide chains. Tentative rules (1969). *Biochemistry*, 9:3471–3479, 1970.

[26] T.C. Beutler, A.E. Mark, R.C. van Schaik, P.R. Gerber, and W.F. van Gunsteren. Avoiding singularities and numerical instabilities in free energy calculations based on molecular simulations. *Chem. Phys. Lett.*, 222:529–539, 1994.

[27] W. H. Press; S. A. Teukolsky; W. T. Vetterling; B. P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.

[28] P.H. Hünenberger. *Simulation and theory of electrostatic interactions in solution: Computational chemistry, biophysics, and aqueous solution*, chapter Lattice-sum methods for computing electrostatic interactions in molecular simulations. American Institute of Physics, New York, U.S.A., 1999.
[29] P.P. Ewald. Die Berechnung optischer und elektrostatischer Gitterpotentiale. *Ann. Phys.*, 369:253–287, 1921.
[30] R.W. Hockney and J.W. Eastwood. *Computer simulation using particles.* McGraw-Hill, New York, U.S.A., 1981.
[31] E. Wigner. Effect of the electron interaction on the energy levels of electrons in metals. *Trans. Faraday Soc.*, 34:678–685, 1938.
[32] B.U. Felderhof. Wigner solids and diffusion controlled reactions in a regular array of spheres. *Physica A*, 130:34–56, 1985.
[33] B.R.A. Nijboer and T.W. Ruijgrok. On the energy per particle in three- and two-dimensional Wigner lattices. *J. Stat. Phys.*, 53:361–382, 1988.
[34] M. Deserno and C. Holm. How to mesh up Ewald sums. I. A theoretical and numerical comparison of various particle mesh routines. *J. Chem. Phys.*, 109:7678–7693, 1998.
[35] M. Deserno and C. Holm. How to mesh up Ewald sums. II. An accurate error estimate for the particle-particle-particle-mesh algorithm. *J. Chem. Phys.*, 109:7694–7701, 1998.
[36] T. P. Straatsma; J. A. McCammon. Molecular Dynamics Simulations with Interaction Potentials Including Polarization Development of a Noniterative Method and Application to Water. *Mol. Simul.*, 5:181–192, 1990.
[37] P. Drude. *The Theory of Optics*. New York [etc.] Longmans, Green, and Co., 1901.
[38] M. Born K. Huang. *Dynamic Theory of Crystal Lattices*. Oxford University Press , Oxford, UK, 1954.
[39] D.P. Geerke and W.F. van Gunsteren. Calculation of the free energy of polarization: quantifying the effect of explicitly treating electronic polarization on the transferability of force-field parameters. *J. Phys. Chem. B*, 111:6425–6436, 2007.
[40] S. Riniker, A.P.E. Kunz, and W.F. van Gunsteren. On the calculation of the dielectric permittivity of molecular models in the liquid phase. *J. Chem. Theory Comput.*, 7:1469–1475, 2011.
[41] H. Yu, T Hansson, and W.F. van Gunsteren. Development of a Simple, Self-Consistent Polarizable Model for Liquid Water. *J. Chem. Phys.*, 118:221–234, 2003.
[42] Z. Lin, S.J. Bachmann, and W.F. van Gunsteren. GROMOS polarisable charge-on-spring models for liquid urea: COS/U and COS/U2. *J. Chem. Phys.*, 142:094117, 2015.
[43] H. Yu and W.F. van Gunsteren. Charge-on-spring polarizable water models revisited: From water clusters to liquid water to ice. *J. Chem. Phys.*, 121:9549–9564, 2004.
[44] A.P. Kunz and W.F. van Gunsteren. Development of a non-linear classical polarisation model for liquid water and aqueous solutions: COS/D. *J. Phys. Chem. A*, 113:11570–11579, 2009.
[45] S.J. Bachmann and W.F. van Gunsteren. On the compatibility of polarisable and non-polarisable models for liquid water. *Mol. Phys.*, 112:2761–2780, 2014.
[46] A. Glättli, X. Daura, and W.F. van Gunsteren. Derivation of an improved SPC model for liquid water: SPC/A and SPC/L. *J. Chem. Phys.*, 116:9811–9828, 2002.
[47] W.F. van Gunsteren and H.J.C. Berendsen. Algorithms for macromolecular dynamics and constraint dynamics. *Mol. Phys.*, 34:1311–1327, 1977.
[48] S. Riniker and W.F. van Gunsteren. A simple, efficient polarisable coarse-grained water model for molecular dynamics simulations. *J. Chem. Phys.*, 134:084110, 2011.
[49] W. Huang and W.F. van Gunsteren. Challenge of representing entropy at different levels of resolution in molecular simulation. *J. Phys. Chem. B*, 119:753–763, 2015.
[50] S.Riniker, J.R. Allison, and W.F. van Gunsteren. On developing coarse-grained models for biomolecular simulation: a review:. *Phys. Chem. Chem. Phys.*, 14:12423–12430, 2012.
[51] M. Christen and W.F. van Gunsteren. Multigraining: an algorithm for simultaneous fine-grained and coarse-grained simulation of molecular systems. *J. Chem. Phys.*, 124:7, 2006.
[52] S. Riniker and W.F. van Gunsteren. Mixing coarse-grained and fine-grained water in molecular dynamics simulations of a single system. *J. Chem. Phys.*, 137:044120, 2012.
[53] S. Riniker, A.P. Eichenberger, and W.F. van Gunsteren. Structural effects of an atomic-level layer of water molecules around proteins solvated in supra-molecular coarse-grained water. *J. Phys. Chem. B*, 116:8873–8879, 2012.
[54] R. Kaptein, E.R.P. Zuiderweg, R.M. Scheek, R. Boelens, and W.F. van Gunsteren. A Protein Structure from Nuclear Magnetic Resonance Data lac Repressor Headpiece. *J. Mol. Biol.*, 182:179–182, 1985.
[55] W.F. van Gunsteren, R. Boelens, R. Kaptein, R.M. Scheek, and E.R.P. Zuiderweg. An Improved Restrained Molecular Dynamics Technique to Obtain Protein Tertiary Structure from Nuclear Magnetic Resonance Data. In J. Hermans, editor, *Molecular Dynamics and Protein Structure*, pages 92–99. Polycrystal Book Service, Western Springs, Ill, USA, 1985.
[56] A.E. Torda and W.F. van Gunsteren. Molecular Modeling Using Nuclear Magnetic Resonance Data. In K.B. Lipkowitz and D.B. Boyd, editors, *Reviews in Computational Chemistry*, volume III, pages 143–172. VCH Publishers, Inc. New York, 1992.
[57] W.F. van Gunsteren, R.M. Brunne, P. Gros, R.C. van Schaik, C.A. Schiffer, and A.E. Torda. Accounting for Molecular Mobility in Structure Determination Based on Nuclear Magnetic Resonance Spectroscopic and X-Ray Diffraction Data. In T.L. James and N.J. Oppenheimer, editors, *Nuclear Magnetic Resonance*, volume 239 of *Methods in Enzymology*, pages 619–654. Academic Press, New York, 1994.
[58] A.E. Torda, R.M. Scheek, and W.F. van Gunsteren. Time-dependent distance restraints in molecular dynamics simulations. *Chem. Phys. Lett.*, 157:289–294, 1989.
[59] A.E. Torda, R.M. Scheek, and W.F. van Gunsteren. Time-averaged Nuclear Overhauser Effect Distance Restraints Applied to Tendamistat. *J. Mol. Biol.*, 214:223–235, 1990.
[60] A.P. Nanzer, W.F. van Gunsteren, and A.E. Torda. Parametrisation of time-averaged distance restraints in MD simulations. *J. Biomol. NMR*, 6:313–320, 1995.
[61] F. Fraternali and W.F. van Gunsteren. Conformational Transitions of a Dipeptide in Water: Effects of Imposed Pathways Using Umbrella Sampling Techniques. *Biopolymers*, 34:347–355, 1994.

[62] T.C. Beutler, T. Bremi, R.R. Ernst, and W.F. van Gunsteren. Motion and Conformation of Side Chains in Peptides. A Comparison of 2D Umbrella-Sampling Molecular Dynamics and NMR Results. *J. Phys. Chem.*, 100:2637–2645, 1996.
[63] W.R.P. Scott, A.E. Mark, and W.F. van Gunsteren. On using time-averaging restraints in molecular dynamics simulation. *J. Biomol. NMR*, 12:501–508, 1998.
[64] A.E. Torda, R.M. Brunne, T. Huber, H. Kessler, and W.F. van Gunsteren. Structure refinement using time-averaged J-coupling constant restraints. *J. Biomol. NMR*, 3:55–66, 1993.
[65] A.P. Nanzer, A.E. Torda, C. Bisang, C. Weber, J.A. Robinson, and W.F. van Gunsteren. Dynamical Studies of Peptide Motifs in the Plasmodium falciparum Circumsporozoite Surface Protein by Restrained and Unrestrained MD Simulations. *J. Mol. Biol.*, 267:1012–1025, 1997.
[66] B. Keller, M. Christen, C. Oostenbrink, and W.F. van Gunsteren. On using oscillating time-dependent restraints in MD simulation. *J. Biomol. NMR*, 37:1–14, 2007.
[67] A. Pardi, M. Billeter, and K. Wüthrich. Calibration of the angular dependence of the amide proton-Cα proton coupling constants, $^3J_{\mathrm{HN}\alpha}$, in a globular protein. Use of $^3J_{\mathrm{HN}\alpha}$ for identification of helical secondary structure. *J. Mol. Biol.*, 180:741–751, 1984.
[68] A. DeMarco, M. Llinás, and K. Wüthrich. Analysis of the ^{1}H-NMR spectra of ferrichrome peptides. I. The non-amide protons. *Biopolymers*, 17:617–636, 1978.
[69] A. DeMarco, M. Llinás, and K. Wüthrich. ^{1}H-^{1}5N spin-spin couplings in alumichrome. *Biopolymers*, 17:2727–2742, 1978.
[70] V. F. Bystrov. Spin-spin coupling and the conformational states of peptide systems. *Prog. NMR Spectr.*, 10:41–81, 1976.
[71] M. Christen, B. Keller, and W.F. van Gunsteren. Biomolecular structure refinement based on adaptive restraints using local-elevation simulation. *J. Biomol. NMR*, 39:265–273, 2007.
[72] N. Hansen, F. Heller, N Schmid, and W.F. van Gunsteren. Time-averaged order parameter restraints in molecular dynamics simulations. *J. Biomol. NMR*, 60:169–187, 2014.
[73] E.R. Henry and A. Szabo. Influence of vibrational motion on solid state line shapes and NMR relaxation. *J. Chem. Phys.*, 82:4753 – 4761, 1985.
[74] D. Waasmaier and A. Kirfel. New analytical scattering-factor functions for free atoms and ions. *Acta Crystallogr.*, A51:416–431, 1995.
[75] P. Gros, W.F. van Gunsteren, and W.G.J. Hol. Inclusion of Thermal Motion in Crystallographic Structures by Restrained Molecular Dynamics. *Science*, 249:1149–1152, 1990.
[76] T. A. Jones, J. Y. Zuo, S. W. Cowan, and M. Kjeldgaard. Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallogr.*, 47A:110–119, 1991.
[77] M. S. Chapman. Restrained real-space macromolecular atomic refinement using a new resolution-dependent electron-density function. *Acta Crystallogr.*, A51:69–80, 1995.
[78] A. de Ruiter and C. Oostenbrink. Protein-ligand binding from distancefield distances and Hamiltonian replica exchange simulations. *J. Chem. Theor. Comput.*, 9:883 – 892, 2012.
[79] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1:269–271, 1959.
[80] B. Tidor. Simulated annealing on free energy surfaces by a combined molecular dynamics and Monte Carlo approach. *J. Phys. Chem.*, 97:1069–1073, 1993.
[81] Z. Liu and B.J. Berne. Method for accelerating chain folding and mixing. *J. Chem. Phys.*, 99:6071–6077, 1993.
[82] X. Kong and C.L. Brooks III. λ-dynamics: A new approach to free energy calculations. *J. Chem. Phys.*, 105:2414–2423, 1996.
[83] H. Hansen and P.H. Hünenberger. Ball-and-stick local elevation umbrella sampling: molecular simulations involving enhanced sampling within conformational or alchemical subspaces of low internal dimensionalities, minimal irrelevant volumes and problem-adapted geometries. *J. Chem. Theory Comput.*, 6:2622–2646, 2010.
[84] H. Hansen, X. Daura, and P.H. Hünenberger. Enhanced conformational sampling in molecular dynamics simulations of solvated peptides: Fragment-based local elevation umbrella sampling. *J. Chem. Theory Comput.*, 6:2598–2621, 2010.
[85] T. Huber, A.E. Torda, and W.F. van Gunsteren. Local elevation: A method for improving the searching properties of molecular dynamics simulation. *J. Comput. Aided Mol. Des.*, 8:695–708, 1994.
[86] O. Engkvist and G. Karlström. A method to calculate the probability distribution for systems with large energy barriers. *Chem. Phys.*, 213:63–76, 1996.
[87] A. Laio and M. Parrinello. Escaping free-energy minima. *Proc. Natl. Acad. Sci. USA*, 99:12562–12566, 2002.
[88] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *J. Comput. Phys.*, 23:187–199, 1977.
[89] H.S. Hansen and P.H. Hünenberger. Using the local elevation method to construct optimized umbrella sampling potentials: calculation of the relative free energies and interconversion barriers of glucopyranose ring conformers in water. *J. Comput. Chem.*, 31:1–23, 2010.
[90] C.D. Christ and W.F. van Gunsteren. Enveloping Distribution Sampling: A method to calculate free energy differences from a single simulation. *J. Chem. Phys.*, 126:184110, 2007.
[91] A. Barducci, G. Bussi, and M. Parrinello. Well-tempered metadynamics: A smoothly converging and tunable free-energy method. *Phys. Rev. Lett.*, 100:020603/1–020603/4, 2008.
[92] I.G. Tironi, R.M. Brunne, and W.F. van Gunsteren. On the relative merits of flexible versus rigid models for use in computer simulations of molecular liquids. *Chem. Phys. Lett.*, 250:19–24, 1996.
[93] W.F. van Gunsteren and M. Karplus. Effect of Constraints on the Dynamics of Macromolecules. *Macromolecules*, 15:1528–1544, 1982.
[94] W.F. van Gunsteren. Constrained dynamics of flexible molecules. *Mol. Phys.*, 40:1015–1019, 1980.
[95] J.-P. Ryckaert, G. Ciccotti, and H.J.C. Berendsen. Numerical Integration of the Cartesian Equations of Motion of a System with Constraints: Molecular Dynamics of n-Alkanes. *J. Comput. Phys.*, 23:327–341, 1977.
[96] S. Miyamoto and P. A. Kollman. SETTLE: An Analytical Version of the SHAKE and RATTLE Algorithm for Rigid Water Models. *J. Comput. Chem.*, 13(8):952–962, 1992.

[97] V. Kräutler, W.F. van Gunsteren, and P.H. Hünenberger. A Fast SHAKE Algorithm to Solve Distance Constraint Equations for Small Molecules in Molecular Dynamics Simulations. *J. Comput. Chem.*, 22:501–508, 2001.
[98] B. Hess, H. Bekker, H.J.C. Berendsen, and J.G.E.M. Fraaije. LINCS: A Linear Contstraint Solver for Molecular Simulations. *J. Comput. Chem.*, 18:1463–1472, 1997.
[99] M. Christen and W.F. van Gunsteren. An approximate but fast method to impose flexible distance constraints in molecular dynamics simulations. *J. Chem. Phys.*, 122:144106, 2005.
[100] M. Christen, A.-P.E. Kunz, and W.F. van Gunsteren. Sampling of rare events using hidden restraints. *J. Phys. Chem. B*, 110:8488–8498, 2006.
[101] R.C. van Schaik, H.J.C. Berendsen, A.E. Torda, and W.F. van Gunsteren. A Structure Refinement Method Based on Molecular Dynamics in Four Spatial Dimensions. *J. Mol. Biol.*, 234:751–762, 1993.
[102] H. Bekker, H.J.C. Berendsen, and W.F. van Gunsteren. Force and virial of torsional-angle dependent potentials. *J. Comput. Chem.*, 16:527–533, 1995.
[103] W.F. van Gunsteren, S.R. Billeter, A.A. Eising, P.H. Hünenberger, P. Krüger, A.E. Mark, W.R.P. Scott, and I.G. Tironi. *Biomolecular Simulation: The GROMOS96 Manual and User Guide*. Vdf Hochschulverlag AG an der ETH Zürich, Zürich, Switzerland, 1996.
[104] C. L. Brooks III D. J. Tobias. Molecular-dynamics with internal coordinate constraints. *J. Chem. Phys.*, 89:5115–5127, 1988.
[105] A. Amadei, G. Chillemi, M. A. Ceruso, A. Grottesi, and A. Di Nola. Molecular dynamics simulations with constrained roto-translational motions: Theoretical basis and statistical mechanical consistency. *J. Chem. Phys.*, 112:9–23, 2000.
[106] R. Fletcher and C.M. Reeves. Function minimization by conjugate gradients. *Comput. J.*, 7:149–154, 1964.
[107] W.F. van Gunsteren and M. Karplus. A Method for Constrained Energy Minimization of Macromolecules. *J. Comput. Chem.*, 1:266–274, 1980.
[108] W.F. van Gunsteren and H.J.C. Berendsen. Computer Simulation as a Tool for Tracing the Conformational Differences between Proteins in Solution and in the Crystalline State. *J. Mol. Biol.*, 176:559–564, 1984.
[109] J. Lautz, H. Kessler, R. Kaptein, and W.F. van Gunsteren. Molecular dynamics simulations of cyclosporin A: The crystal structure and dynamic modelling of a structure in apolar solution based on NMR data. *J. Comput. Aided Mol. Des.*, 1:219–241, 1987.
[110] Polak E. and G. Ribiére. Note sur la convergence de méthodes de directions conjuguées. *Rev. Fr. Inform. Rech. O.*, 3(R1):35–43, 1969.
[111] W.F. van Gunsteren, A.P. Nanzer, and A.E. Torda. Molecular simulation methods for generating ensembles or trajectories consistent with experimental data. In K. Binder and G. Ciccotti, editors, *Monte Carlo and Molecular Dynamics of Condensed Matter Systems*, volume 49 of *Proceedings of the Euroconference*, pages 777–788. SIF, Bologna, Italy, 1996.
[112] L.V. Woodcock. Isothermal molecular dynamics calculations for liquid salts. *Chem. Phys. Lett.*, 10:257–261, 1971.
[113] M.P. Allen and D.J. Tildesley. *Computer simulation of liquids*. Oxford University Press, New York, USA, 1987.
[114] S. Nosé. A molecular dynamics method for simulations in the canonical ensemble. *Mol. Phys.*, 52:255–268, 1984.
[115] S. Nosé. A unified formulation of the constant temperature molecular dynamics methods. *J. Chem. Phys.*, 81:511–519, 1984.
[116] W.G. Hoover. Canonical dynamics: Equilibrium phase-space distributions. *Phys. Rev. A*, 31:1695–1697, 1985.
[117] G.J. Martyna, M.L. Klein, and M. Tuckerman. Nosé-Hoover chains: The canonical ensemble via continuous dynamics. *J. Chem. Phys.*, 97:2635–2643, 1992.
[118] P.H. Hünenberger. Calculation of the group-based pressure in molecular simulations: I. A general formulation including Ewald and particle-particle-particle-mesh electrostatics. *J. Chem. Phys.*, 116:6880–6897, 2002.
[119] B. Oliva and P.H. Hünenberger. Calculation of the group-based pressure in molecular simulations: II. Numerical tests and application to liquid water. *J. Chem. Phys.*, 116:6898–6909, 2002.
[120] H. Bekker and P. Ahlström. The Virial of Angle Dependent Potentials in Molecular Dynamics Simulations. *Mol. Simul.*, 13:367–374, 1994.
[121] E. Paci and M. Marchi. Constant-pressure molecular dynamics techniques applied to complex molecular systems and solvated proteins. *J. Phys. Chem.*, 100:4314–4322, 1996.
[122] G. S. Kell. Precise representation of volume properties of water at 1 atmosphere. *J. Chem. Eng. Data*, 12:66–69, 1967.
[123] B. Gavish, E. Gratton, and C. J. Hardy. Adiabatic compressibility of globular proteins. *PNAS*, 80:750–754, 1983.
[124] M.A. Cuendet and W.F. van Gunsteren. On the calculation of velocity-dependent properties in molecular dynamics simulations using the leap-frog integration algorithm. *J. Chem. Phys.*, 127:184102, 2007.
[125] W.F. van Gunsteren, P.H. Hünenberger, A.E. Mark, P.E. Smith, and I.G. Tironi. Computer simulation of protein motion. *Comput. Phys. Commun.*, 91:305–319, 1995.
[126] P.H. Hünenberger, A.E. Mark, and W.F. van Gunsteren. Fluctuation and Cross-Correlation Analysis of Protein Motions Observed in Nanosecond Molecular Dynamics Simulations. *J. Mol. Biol.*, 252:492–503, 1995.
[127] Z. Lin and W.F. van Gunsteren. On the use of a weak-coupling thermostat in replica-exchange molecular dynamics simulations. *J. Chem. Phys.*, 143:034110, 2015.
[128] W.F. van Gunsteren and H.J.C. Berendsen. A leap-frog algorithm for stochastic dynamics. *Mol. Simul.*, 1:173–185, 1988.
[129] Shi Yun-yu, Wang Lu, and W.F. van Gunsteren. On the approximation of solvent effects on the conformation and dynamics of cyclosporin A by stochastic dynamics simulation techniques. *Mol. Simul.*, 1:369–383, 1988.
[130] W.F. van Gunsteren, T.C. Beutler, F. Fraternali, P.M. King, A.E. Mark, and P.E. Smith. Computation of free energy in practice: choice of approximations and accuracy limiting factors. In W.F. van Gunsteren, P.K. Weiner, and A.J. Wilkinson, editors, *Computer Simulation of Biomolecular Systems, Theoretical and Experimental Applications*, volume 2, pages 315–348. Escom Science Publishers, Leiden, The Netherlands, 1993.
[131] W.F. van Gunsteren. The role of computer simulation techniques in protein engineering. *Protein Eng.*, 2:5–13, 1988.
[132] T.C. Beutler and W.F. van Gunsteren. The computation of a potential of mean force: Choice of the biasing potential in the umbrella sampling technique. *J. Chem. Phys.*, 100:1492–1497, 1994.

[133] W.F. van Gunsteren. Methods for calculation of free energies and binding constants: Successes and problems. In W.F. van Gunsteren and P.K. Weiner, editors, *Computer Simulation of Biomolecular Systems, Theoretical and Experimental Applications*, pages 27–59. Escom Science Publishers, Leiden, The Netherlands, 1989.

[134] L. Wang, Y. Deng, Y. Wu, B. Kim, D.N. LeBard, D. Wandschneider, M. Beachy, R.A. Friesner, and R. Abel. Accurate Modeling of Scaffold Hopping Transformations in Drug Discovery. *J. Chem. Theory Comput.*, 13(1):42–54, 2017.

[135] A.E. Mark and W.F. van Gunsteren. Free Energy Calculations in Drug Design: A Practical Guide. In P.M. Dean, G. Jolles, and C.G. Newton, editors, *New Perspectives in Drug Design*, Proceedings of the 9th Intl. Roundtable, pages 185–200. Academic Press Ltd., 1995.

[136] A.E. Mark, W.F. van Gunsteren, and H.J.C. Berendsen. Calculation of Relative Free Energy via Indirect Pathways. *J. Chem. Phys.*, 94:3808–3816, 1991.

[137] Z. Lin and W.F. van Gunsteren. A comparison of pathway independent and pathway dependent methods in the calculation of conformational free enthalpy differences. *Protein Sci.*, 2015.

[138] J. A. Tembe, B. L.; McCammon. A simple theoretical approach is outlined for calculating differences in the free energy of binding of related ligand-receptor pairs. *Computers & Chemistry*, 8:281–283, 1984.

[139] X. Daura, P.H. Hünenberger, A.E. Mark, E. Querol, F.X. Avilés, and W.F. van Gunsteren. Free Energies of Transfer of Trp Analogs from Chloroform to Water: Comparison of Theory and Experiment and the Importance of Adequate Treatment of Electrostatic and Internal Interactions. *J. Am. Chem. Soc.*, 118:6285–6294, 1996.

[140] A.E. Mark, S.P. van Helden, P.E. Smith, L.H.M. Janssen, and W.F. van Gunsteren. Convergence Properties of Free Energy Calculations: Alpha-Cyclodextrin Complexes as a Case Study. *J. Am. Chem. Soc.*, 116:6293–6302, 1994.

[141] T.C. Beutler, D.R. Béguelin, and W.F. van Gunsteren. Free energy of cavity formation in solvent: Computational, methodological and physical aspects. *J. Chem. Phys.*, 102:3787–3793, 1995.

[142] Shi Yun-yu, A.E. Mark, Wang Cun-xin, Huang Fuhua, H.J.C. Berendsen, and W.F. van Gunsteren. Can the stability of protein mutants be predicted by free energy calculations? *Protein Eng.*, 6:289–295, 1993.

[143] H. Liu, A.E. Mark, and W.F. van Gunsteren. Estimating the Relative Free Energy of Different Molecular States with Respect to a Single Reference State. *J. Phys. Chem.*, 100:9485–9494, 1996.

[144] A.E. Mark, Y. Xu, H. Liu, and W.F. van Gunsteren. Rapid non-empirical approaches for estimating relative binding free energies. *Acta Biochim. Polonica*, 42:525–536, 1995.

[145] C. Peter, C. Oostenbrink, A. van Dorp, and W.F. van Gunsteren. Estimating entropies from molecular dynamics simulations. *J. Chem. Phys.*, 120:2652–2661, 2004.

[146] T.C. Beutler and W.F. van Gunsteren. Umbrella sampling along linear combinations of generalized coordinates Theory and application to a glycine dipeptide. *Chem. Phys. Lett.*, 237:308–316, 1995.

[147] K. S. Shing and K. E. Gubbins. The Chemical-Potential In Dense Fluids And Fluid Mixtures Via Computer-Simulation. *Mol. Phys.*, 46(5):1109–1128, 1982.

[148] J. G. Powles, W. A. B. Evans, and N. Quirke. Non-Destructive Molecular-Dynamics Simulation Of The Chemical-Potential Of A Fluid. *Mol. Phys.*, 46(6):1347–1370, 1982.

[149] G. Jacucci and N. Quirke. Free-Energy Calculations For Crystals. *Lect. Notes Phys.*, 166:38–57, 1982.

[150] K. K. Han. A New Monte-Carlo Method For Estimating Free-Energy And Chemical-Potential. *Phys. Lett. A*, 165(1):28–32, 1992.

[151] K. K. Han. Multiensemble sampling: An alternative efficient Monte Carlo technique. *Phys. Rev. E*, 54(6):6906–6910, 1996.

[152] Y. G. Chen and G. Hummer. Slow conformational dynamics and unfolding of the calmodulin C-terminal domain. *J. Am. Chem. Soc.*, 129(9):2414–2415, 2007.

[153] C. D. Christ and W. F. van Gunsteren. Multiple free energies from a single simulation: Extending enveloping distribution sampling to nonoverlapping phase-space distributions. *J. Chem. Phys.*, 128(17):174112, 2008.

[154] J. D. Chodera, W. C. Swope, J. W. Pitera, C. Seok, and K. A. Dill. Use of the weighted histogram analysis method for the analysis of simulated and parallel tempering simulations. *J. Chem. Theory Comput.*, 3(1):26–41, 2007.

[155] C.D. Christ and W.F. van Gunsteren. Simple, efficient, and reliable computation of multiple free energy differences from a single simulation: a reference Hamiltonian parameter update scheme for enveloping distribution sampling (EDS). *J. Chem. Theory Comput.*, 5:276–286, 2009.

[156] C.D. Christ and W.F. van Gunsteren. Comparison of three enveloping distribution sampling Hamiltonians for the estimation of multiple free energy differences from a single simulation. *J. Comput. Chem.*, 30:1664–1679, 2009.

[157] D. Hamelberg, J. Mongan, and J.A. McCammon. Accelerated molecular dynamics: A promising and efficient simulation method for biomolecules. *J. Chem. Phys.*, 120:11919 – 11929, 2004.

[158] Y. Miao, V.A. Feher, and J.A. McCammon. Gaussian accelerated molecular dynamics: Unconstrained enhanced sampling and free energy calculation. *J. Chem. Theory Comput.*, 11:3584 – 3595, 2015.

[159] J.W. Perthold and C. Oostenbrink. Accelerated enveloping distribution sampling: Enabling sampling of multiple end states while preserving local energy minima. *J. Phys. Chem. B*, 122:5030 – 5037, 2018.

[160] D. Petrov J.W. Perthold and C. Oostenbrink. Toward automated free energy calculation with accelerated enveloping distribution sampling (a-eds). *J. Chem. Inf. Model.*, XXX:XXX – XXX, 2020.

[161] N. Hansen, P.H. Hünenberger, and W.F. van Gunsteren. Efficient combination of environment change and alchemical perturbation within the enveloping distribution sampling (EDS) scheme: twin system EDS and application to the determination of octanol-water partition coefficients. *J. Chem. Theory Comput.*, 9:1334–1346, 2013.

[162] Z. Lin, J.Kornfeld, M. Mächler, and W.F. van Gunsteren. Prediction of folding equilibria of differently substituted peptides using one-step perturbation. *J. Am. Chem. Soc.*, 132:7226–7278, 2010.

[163] Z. Lin, H. Liu, S. Riniker, and W.F. van Gunsteren. On the use of enveloping distribution sampling (EDS) to compute free enthalpy differences between different conformational states of molecules: application to 3_{10}-, α, and π helices. *J. Chem. Theory. Comput.*, 7:3884–3897, 2011.

[164] Z. Lin, C. Necula, and W.F. van Gunsteren. Using enveloping distribution sampling to compute the folding free enthalpy of a β-peptide with a very unstable folded conformation in solution: The advantage of focused sampling using EDS. *Chem. Phys.*, 428:156–163, 2014.
[165] K. Meier, N. Schmid, and W.F. van Gunsteren. Interfacing the GROMOS (bio)molecular simulation software to quantum-chemical program packages. *J. Comput. Chem.*, 2012, DOI: 10.1002/jcc.23047.
[166] M. Christen, P. H. Hünenberger, D. Bakowies, R. Baron, R. Bürgi, D. P. Geerke, T. N. Heinz, M. A. Kastenholz, V. Kräutler, C. Oostenbrink, C. Peter, D. Trzesniak, and W. F. van Gunsteren. The GROMOS software for biomolecular simulation: GROMOS05. *J. Comput. Chem.*, 26:1719–1751, 2005.
[167] W. F. van Gunsteren et al. http://www.gromos.net.
[168] W. Thiel. MNDO99 v. 6.1 ed., Max-Planck-Institut für Kohlenforschung: Mülheim an der Ruhr, Germany, 2003.
[169] R. Ahlrichs et. al. http://www.cosmologic.de/turbomole.html.
[170] N. Schmid, C.D. Christ, M.Christen, A.P. Eichenberger, and W.F. van Gunsteren. Architecture and implementation and parallelization of the GROMOS software for biomolecular simulation. *Comput. Phys. Commun.*, 183:890–903, 2012.
[171] Y. Sugita and Y. Okamoto. Replica-exchange molecular dynamics method for protein folding. *Chem. Phys. Lett.*, 314:141–151, 1999.
[172] Y. Okamoto. Generalized-ensemble algorithms: enhanced sampling techniques for Monte Carlo and molecular dynamics simulations. *J. Mol. Graph. Mod.*, 22:425–439, 2004.
[173] D.; Meng Y. Sindhikara and A. E. Roitberg. Exchange frequency in replica exchange molecular dynamics. *J. Chem. Phys.*, 128:024103, 2008.
[174] J. Hritz and C. Oostenbrink. Hamiltonian replica exchange molecular dynamics using soft-core interactions. *J. Chem. Phys.*, 128:144121, 2008.
[175] X. Periole and A.E. Mark. Convergence and sampling efficiency in replica exchange simulations of peptide folding in explicit solvent. *J. Chem. Phys.*, 126:10, 2007.
[176] A.P.E. Kunz and W.F. van Gunsteren. Enhancing the configurational sampling of ions in aqueous solution using adiabatic decoupling with translational temperature scaling. *J. Phys. Chem. B*, 115:2931–2936, 2011.
[177] D. Steiner, C. Oostenbrink, F. Diederich, M. Zürcher, and W.F van Gunsteren. Calculation of Binding Free Energies of Inhibitors to Plasmepsin II. *J. Comput. Chem.*, 32:1801–1812, 2011.
[178] J. Hritz and C. Oostenbrink. Optimization of replica exchange molecular dynamics by fast mimicking. *J. Chem. Phys.*, 127:204104, 2007.
[179] N.-V. Buchete and G. Hummer. Peptide folding kinetics from replica exchange molecular dynamics. *Phys. Rev. E*, 77:030902(R), 2008.
[180] N.-V. Buchete and G. Hummer. Coarse Master Equations for Peptide Folding Dynamics. *J. Phys. Chem. B*, 112:6057–6069, 2008.

Symbols

Symbol	Meaning
	Common names and abbreviations
GROMOS	The GROMOS software package
MD++	The MD++ simulation engine in C++
GROMOS++	The GROMOS++ analysis package in C++
GROMOS96	The GROMOS96 simulation package (1996)
3D	abbreviation for three dimensions
AA	Atomistic (All Atom) models
BD	Brownian Dynamics simulation
B&S – LEUS	Ball and stick local elevation umbrella sampling
CG	Coarse Grained models
CGEM	Conjugate gradient method for energy minimization
FRCG	Fletcher-Reeves conjugate gradient method for energy minimization
PRCG	Polak-Ribiére conjugate gradient method for energy minimization
COG	Center of geometry
COS	Charge On Spring approach
CP	Car Parrinello approach
DF	Distancefield
DOF	Degrees of freedom (abbreviation)
DPD	Diffusive Particle Dynamics simulation
`doxygen`	Documentation platform
EM	Energy minimisation
EDS	Enveloping distribution sampling
FBC	Fixed boundary conditions
HBC	Hyper-spherical boundary conditions
LE	Local elevation
LEUS	Local elevation umbrella sampling
LS	Lattice-sum method
MC	Monte Carlo sampling
MD	Molecular Dynamics simulation
NOE	Nuclear Overhauser Effect
PBC	Periodic boundary conditions
PPPM	Particle-particle–particle-mesh (P^3M) method
QM	Quantum Mechanical models
QMD	Quantum Molecular Dynamics simulation
RDF	Radial distribution function
RE	Replica Exchange
REMD	Replica Exchange Molecular Dynamics simulation
RF	Reaction-field method
RMSD	Root-mean-square difference
RMSF	Root-mean-square fluctuation
SD	Stochastic Dynamics simulation
SDEM	Steepest descent method for energy minimization
TI	Thermodynamic integration
US	Umbrella sampling

Symbol	Meaning
VBC	Vacuum boundary conditions
	Physical constants
h	Planck's constant [0.3990313 kJ mol^{-1} ps]
$\hbar$	Planck's constant divided by 2π [0.06350780 kJ mol^{-1} ps]
$N_{\rm Av}$	Avogadro's number [6.02214 $\times 10^{23}$]
k_B	Boltzmann's constant [1.380662 $\times 10^{-26}$ kJ K^{-1})]
R	Ideal gas constant ($N_{\rm Av} \times k_B$)
c	Speed of light [2.99792458 $\times 10^5$ nm ps^{-1}]
	Degrees of freedom and system configuration
$\mathcal{N}_d$	Number of degrees of freedom of a system
$\mathcal{N}_a$	Number of particles in a system of particles ($\mathcal{N}_d$ =3$\mathcal{N}_a$)
$\mathcal{N}_a^{solu}$	Number of particles the solute consists of
$\boldsymbol{q}$	$3\mathcal{N}_a$-dimensional generalized coordinate vector of a system of particles
$\boldsymbol{p_q}$	$3\mathcal{N}_a$-dimensional generalized momentum vector of a system of particles
$\boldsymbol{r}$	$3\mathcal{N}_a$-dimensional Cartesian coordinate vector of a system of particles
$\boldsymbol{p}$	$3\mathcal{N}_a$-dimensional Cartesian momentum vector of a system of particles
$\boldsymbol{f}$	$3\mathcal{N}_a$-dimensional Cartesian force vector of a system of particles
$\overline{\boldsymbol{f}}$	$3\mathcal{N}_a$-dimensional Cartesian mean force vector of a system of particles
$\boldsymbol{f}^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\mathbf{f}_i^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\boldsymbol{v}$	$3\mathcal{N}_a$-dimensional Cartesian velocity vector of a system of particles
$\mathbf{r}$	3-dimensional Cartesian coordinate vector of a particle
$\mathbf{p}$	3-dimensional Cartesian momentum vector of a particle
$\mathbf{f}$	3-dimensional Cartesian force vector of a particle
$\mathbf{v}$	3-dimensional Cartesian velocity vector of a particle
Ψ $[\Psi(\boldsymbol{r})]$	Wavefunction (position representation; configuration of a quantum-mechanical system of $\mathcal{N}_a$ particles)
$\{$ $\boldsymbol{r}$,$\boldsymbol{p}$ $\}$	Phase-space point (Cartesian coordinates; configuration of a classical system of $\mathcal{N}_a$ particles)
	(Statistical) thermodynamics
$\mathcal{F}$	Free energy
G	Gibbs free energy
H	Enthalpy
$\mathcal{U}$	Energy of a system
$\mathcal{S}$	Entropy of a system
$\mathcal{Z}$	Partition function
$\mathcal{T}$	Instantaneous temperature
$\mathcal{T}_o$	Reference temperature
$\mathcal{K}$	Instantaneous kinetic energy of a system
$\mathcal{K}_{tr}$	Instantaneous translational kinetic energy
$\mathcal{K}_{ir}$	Instantaneous internal+rotational kinetic energy
$\mathcal{U}$	Instantaneous total potential energy of a system
$\mathcal{W}$	Instantaneous virial of a system
$\mathcal{P}$	Instantaneous pressure of a system
$\mathcal{V}$	Instantaneous volume of a system
ρ_J	Number particle density of particles J
	Miscellaneous

Symbol	Meaning
t	Time
Δt	discrete time step
$\mathcal{N}_t$	Number of MD steps
P	Probability
m	Mass of a particle
M	Mass of the whole system
$\underline{\mathbf{m}}$	Diagonal mass matrix of a system of $\mathcal{N}_a$ particles
γ	Friction coefficient of a particle
$\underline{\gamma}$	Diagonal friction coefficient matrix of a system of $\mathcal{N}_a$ particles
T	Absolute temperature
β	prefactor: $1/k_BT$
τ_T	relaxation time for the coupling to a temperature bath
$\mathbf{s}$	Vector denoting the collection of all force-field parameters
λ	Coupling parameter Lambda for a lambda dependent Hamiltonian
$\mathcal{N}_\lambda$	Number of λ-values in a TI simulation
H	Heaviside function defined as $H(x) = 0 \; \forall \; x < 0$ and $H(x) = 1 \; \forall \; x > 0$
sign	Sign function: $\text{sign}(x) = 1 \; \forall \; x > 0$ and $\text{sign}(x) = -1 \; \forall \; x < 0$
i	imaginary number, $i^2 = -1$
δ_{ij}	general Kronecker delta
σ	Standard deviation
σ^2	Variance
$\mathcal{N}_{conf}$	Number of configurations in an ensemble
D	Diffusion constant
R_{gyr}	radius of gyration
η	the viscosity of a system
$g(r)$	radial distribution function
s	Smoothness parameter in EDS simulations
E^R	Energy offset parameter in EDS simulations
$\mathcal{N}^{(s)}$	Number of states in EDS simulations
Spatial boundary conditions	
$\underline{\mathcal{B}}$	3×3-matrix of the box-edge vectors (columns) in the reference Cartesian coordinate system (PBC)
$\hat{\mathbf{e}}$	Unit vector
$\mathbf{a}$	First edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{b}$	Second edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{c}$	Third edge vector of a (triclinic) box (in the reference coordinate system)
a	length of first edge of a (triclinic) box
b	length of second edge of a (triclinic) box
c	length of third edge of a (triclinic) box
$\mathbf{T}$	Position vector of the reference corner of a triclinic box (components in the reference coordinate system and vector relative to the origin of this system)
$\underline{\mathbf{L}}$	Computational box matrix (columns defined by the components of edge vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ in the reference coordinate system)
$\underline{\mathbf{B}}$	Edge length matrix (diagonal, elements a, b and c)
α	First edge angle a triclinic box (between $\mathbf{b}$ and $\mathbf{c}$)
β	Second edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{c}$)
γ	Third edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{b}$)
ϕ	First Euler angle of a triclinic box

Symbol	Meaning
θ	Second Euler angle of a triclinic box
ψ	Third Euler angle of a triclinic box
$\check{\mathbf{r}}$	Oblique coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{r}}$	Oblique fractional coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{k}}$	Oblique coordinates of a reciprocal-space vector
$\check{\mathbf{k}}$	Oblique fractional coordinates of a reciprocal-space vector
$\mathbf{l}$	Lattice vector (three-dimensional vector with integer components)
$\mathbf{k}$	Reciprocal-lattice vector ($\mathbf{k} = 2\pi\underline{\mathbf{L}}^{-1}\mathbf{l}$)
$\underline{\mathbf{S}}$	Transformation matrix
$\underline{\mathbf{R}}$	Transformation matrix
$\underline{\mathbf{T}}$	Transformation matrix
Representation of the interaction	
$\hat{\mathcal{H}}$	Hamiltonian operator describing the interaction for quantum-mechanical degrees of freedom
$\hat{\mathcal{K}}$	Kinetic energy operator (kinetic energy contribution to the quantum-mechanical Hamiltonian operator)
$\hat{\mathcal{V}}$	Potential energy operator (potential energy contribution to the quantum-mechanical Hamiltonian operator)
$\mathcal{H}$ $[\mathcal{H}(\boldsymbol{r},\boldsymbol{p})]$	Hamiltonian function describing the interaction for classical degrees of freedom
$\mathcal{K}$ $[\mathcal{K}(\boldsymbol{p})]$	Kinetic energy contribution to the classical Hamiltonian function
$\mathcal{V}$ $[\mathcal{V}(\boldsymbol{r})]$	Potential energy contribution to the classical Hamiltonian function
$\overline{\mathcal{V}}$ $[\overline{\mathcal{V}}(\boldsymbol{r})]$	Potential of mean force contribution to the classical Hamiltonian function
Physical interactions	
φ [Proper dihedral-angle term]	
$\mathcal{V}^{(phys)}$ $[\mathcal{V}^{(phys)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Physical potential energy contribution to $\mathcal{V}$
$\mathcal{V}^{(cov)}$ $[\mathcal{V}^{(cov)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Covalent potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(nbd)}$ $[\mathcal{V}^{(nbd)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Non-bonded potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(b)}$ $[\mathcal{V}^{(b)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond stretching potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\theta)}$ $[\mathcal{V}^{(\theta)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\xi)}$ $[\mathcal{V}^{(\xi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Improper dihedral-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\varphi)}$ $[\mathcal{V}^{(\varphi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Proper dihedral-angle torsion potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(vdw)}$ $[\mathcal{V}^{(vdw)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Van der Waals potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(ele)}$ $[\mathcal{V}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(LJCRF)}$	Sum of the non-bonded potentials $\mathcal{V}^{(vdw)}$ and $\mathcal{V}^{(ele)}$
Physical force-field terms	
$V^{(b)}$ $[V^{(b)}(b;k^{(b)},b^0)]$	Potential energy function associated with the stretching of a single covalent bond (quartic: $V^{(b,q)}$; harmonic: $V^{(b,h)}$; soft harmonic: $V^{(bs,h)}$)
$V_n^{(b)}$ $[V^{(b)}(b_n;k_n^{(b)},b_n^0)]$	Potential energy function associated with the stretching of the nth single covalent bond (quartic: $V_n^{(b,q)}$; harmonic: $V_n^{(b,h)}$; soft harmonic: $V_n^{(bs,h)}$)
$\mathbf{f}^{(b,q)}$	Force due to the bond stretching potential (quartic)
$\mathbf{f}^{(b,h)}$	Force due to the bond stretching potential (harmonic)
$\mathbf{f}^{(bs,h)}$	Force due to the bond stretching potential (soft harmonic)
$N^{(b)}$	Number of covalent bonds in the molecular system
$N^{(bs)}$	Number of soft covalent bonds in the molecular system
$M_n^{(b)}$	Bond type code associated with covalent bond term n

Symbol	Meaning
$b_n \quad [b_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Length of covalent bond n in the considered configuration
$b_n^0 \quad [b^0(M_n^{(b)}, \mathbf{s})]$	Reference length of covalent bond term n
$k_n^{(b,q)}$	Force constant of stretching for covalent bond term n (quartic potential)
$k_n^{(b,h)}$	Force constant of stretching for covalent bond term n (harmonic potential)
$V^{(\theta)} \quad [V^{(\theta)}(\theta; k^{(\theta)}, \theta^0)]$	Potential energy function associated with the bending of a single covalent bond angle (cosine-harmonic: $V^{(\theta,c)}$; soft cosine-harmonic: $V^{(\theta s,c)}$; angle-harmonic: $V^{(\theta,h)}$)
$V_n^{(\theta)} \quad [V_n^{(\theta)}(\theta_n; k_n^{(\theta)}, \theta_n^0)]$	Potential energy function associated with the bending of the nth covalent bond angle (cosine-harmonic: $V_n^{(\theta,c)}$; soft cosine-harmonic: $V_n^{(\theta s,c)}$; angle-harmonic: $V_n^{(\theta,h)}$)
$\mathbf{f}^{(\theta,c)}$	Force due to the bond angle potential (cosine-harmonic)
$\mathbf{f}^{(\theta s,c)}$	Force due to the bond angle potential (soft cosine-harmonic)
$\mathbf{f}^{(\theta,h)}$	Force due to the bond angle potential (angle-harmonic)
$N^{(\theta)}$	Number of covalent bond angles in the molecular system
$M_n^{(\theta)}$	Bond-angle type code associated with covalent bond-angle term n
$\theta_n \quad [\theta_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent bond angle n in the considered configuration
$\theta_n^0 \quad [\theta^0(M_n^{(\theta)}, \mathbf{s})]$	Reference angle of covalent bond-angle term n
$k_n^{(\theta,c)}$	Force constant of bending for covalent bond-angle term n (cosine-harmonic potential)
$k_n^{(\theta s,c)}$	Force constant of bending for covalent bond-angle term n (soft cosine-harmonic potential)
$k_n^{(\theta,h)}$	Force constant of bending for covalent bond-angle term n (angle-harmonic potential)
$V^{(\xi)} \quad [V^{(\xi)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$V^{(\xi s)} \quad [V^{(\xi s)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$\mathbf{f}^{(\xi)}$	Force due to the improper dihedral-angle potential
$\mathbf{f}^{(\xi s)}$	Force due to the soft improper dihedral-angle potential
$N^{(\xi)}$	Number of covalent improper dihedral angles in the molecular system
$N^{(\xi s)}$	Number of covalent improper dihedral angles in the molecular system
$M_n^{(\xi)}$	Improper dihedral-angle type code associated with covalent improper dihedral-angle term n
$\xi_n \quad [\xi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent improper dihedral angle n in the considered configuration
$\xi_n^0 \quad [\xi^0(M_n^{(\xi)}, \mathbf{s})]$	Reference angle of covalent improper dihedral-angle term n
$k_n^{(\xi)}$	Force constant of bending for covalent improper dihedral-angle term n
$V^{(\varphi)} \quad [V^{(\varphi)}(\varphi; k^{(\varphi)}, \varphi^0)]$	Potential energy function associated with the torsion of a single covalent proper dihedral angle (symmetric potential: $V^{(\varphi,s)}$; generalized: $V^{(\varphi,g)}$)
$\mathbf{f}^{(\varphi,s)}$	Force due to the symmetric proper dihedral-angle potential
$\mathbf{f}^{(\varphi,g)}$	Force due to the generalized proper dihedral-angle potential
$N^{(\varphi)}$	Number of covalent proper dihedral angles in the molecular system
$M_n^{(\varphi)}$	Proper dihedral-angle type code associated with covalent proper dihedral-angle term n
$\varphi_n \quad [\varphi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent proper dihedral angle n in the considered configuration
$\varphi_n^0 \quad [\varphi^0(M_n^{(\varphi)}, \mathbf{s})]$	Reference angle (phase shift) of covalent proper dihedral-angle term n
$m_n^{(\varphi)} \quad [m_n^{(\varphi)}(M_n^{(\varphi)}, \mathbf{s})]$	Multiplicity of covalent proper dihedral-angle term n
$k_n^{(\varphi,s)}$	Force constant of torsion for covalent proper dihedral-angle term n (symmetric potential; $\varphi_n^0 = 0, \pi$; $m_n^{(\varphi)} \leq 6$)

Symbol	Meaning
$k_n^{(\varphi,g)}$	Force constant of torsion for covalent proper dihedral-angle term n (generalized potential; $\varphi_n^0 \in [0, 2\pi[$)
q	Partial charge of an atom or site
C_{12}	Van der Waals (Pauli) repulsion coefficient of an atom or site (Lennard-Jones function)
C_6	Van der Waals (London) dispersion coefficient of an atom or site (Lennard-Jones function)
C_{126}	Ratio of Van der Waals coefficients $\frac{C_{12}}{C_6}$ (Lennard-Jones function)
α_{LJ}	Lennard-Jones soft-core switching parameter
α_C	Coulomb soft-core switching parameter
$\mathcal{V}^{(ele,pws)}$ $[\mathcal{V}^{(ele,pws)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Pairwise potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,slf)}$ $[\mathcal{V}^{(ele,slf)}(\underline{\mathcal{B}};\mathbf{s})]$	Self potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,srf)}$ $[\mathcal{V}^{(ele,srf)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Surface potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathbf{f}^{(nbd)}$	Force due to the non-bonded forces
$\Psi_{ij}^{(ele)}$ $[\Psi_{ij}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic influence function associated with the particle pair $i-j$
$\delta_{ij}^{(exc)}$ $[\delta_{ij}^{(exc)}(\mathbf{s})]$	Indicator of non-bonded exclusion for the particle pair $i-j$
$\Psi^{(ele,slf)}$ $[\Psi^{(ele,slf)}(\underline{\mathcal{B}})]$	Electrostatic self influence function
$\psi^{(RF)}$ $[\psi^{(RF)}(\mathrm{x})]$	Influence function at distance x of a particle in RF electrostatics
H $[H(\mathrm{x})]$	Heaviside step function (one if x is positive, zero otherwise)
R_C	Cutoff distance (truncation)
R_{cp}	Short-range cut-off
R_{cl}	Long-range cut-off
R^{cg}	radius of a charge group
N_{cg}	number of atoms belonging to a charge group
R_{RF}	Cutoff distance (onset of the RF continuum; usually set equal to R_C)
ϵ_{RF}	Relative dielectric permittivity of the RF continuum (usually set equal to that of the solvent)
κ_{RF}	Inverse Debye screening length of the RF continuum (usually set to zero)
C_{RF}	Constant characterizing the effect of the RF continuum
$\overline{\mathbf{R}}_{ij}$ $[\overline{\mathbf{R}}_{ij}(\boldsymbol{r})]$	Vector (FBC) or minimum-image vector (PBC) connecting the center of the CG containing particle j to the center of the CG containing particle i (norm $\overline{R}_{ij}$)
$\mathcal{V}^{(ele,pws,RF-CB)}$ $[\mathcal{V}^{(ele,pws,RF-CB)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Coulombic pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RF)}$ $[\mathcal{V}^{(ele,pws,RF-RF)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-dependent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RC)}$ $[\mathcal{V}^{(ele,pws,RF-RC)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-independent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\Psi_{ij}^{(ele,LS-RS)}$ $[\Psi_{ij}^{(ele,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space component of electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\Psi_{ij}^{(ele,LS-KS)}$ $[\Psi_{ij}^{(ele,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space component of the electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-RS)}$ $[\mathcal{V}^{(ele,pws,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-KS)}$ $[\mathcal{V}^{(ele,pws,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\psi^{(LS)}$ $[\psi^{(LS)}(\mathbf{x})]$	Influence function at position $\mathbf{x}$ relative to a particle in LS electrostatics
a	Width of the charge-shaping function
γ $[\gamma(\mathrm{x})]$	Charge-shaping function

Symbol	Meaning
$\hat{\gamma}$ $[\hat{\gamma}\ (\mathrm{x})]$	Fourier transformed charge-shaping function
$\mathbf{E}$	Electric field
$\boldsymbol{\mu}$	Dipole
$\boldsymbol{J}$	
α	Electronic polarisability
$\mathbf{P}$	Polarisation
ϵ	Dielectric permittivity
γ^{pol}	γ to calculate position of off site charge
k^{ho}	harmonic force constant in the COS model
ϕ	Electrostatic potential
Unphysical force-field terms	
$\mathcal{V}^{(spec)}$	Unphysical potential energy
$\mathcal{V}^{(res)}$	Restraint energy
$\mathcal{V}^{(pr)}$	Position restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(c)}$	Force due to the position constraints
$k^{(pr)}$	Force constant of an unphysical position-restraining term
$\mathcal{N}^{(pr)}$	number of positionally restrained atoms
l	Lagrange multiplier for position constraints
$\mathcal{V}^{(dr)}$	Distance restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(dir)}$	Force due to the atom-atom distance restraints
$k^{(dr)}$	Force constant of an unphysical distance-restraining term
$\mathbf{r}^0$	Equilibrium distance of distance restraint
$N^{(dir)}$	Number of atom-atom distance restraints
d_{CH}	carbon-hydrogen distance
d_{CC}	carbon-carbon distance
τ_{dr}	decay time for time-averaged distance restraining
$\mathcal{V}^{(tr)}$	Dihedral-angle restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(tr)}$	Force constant of an unphysical dihedral-angle restraining term
$\mathcal{N}^{(tr)}$	number of restrained dihedral angles
$\mathcal{V}^{(Jr)}$	^3J-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Jr)}$	Force constant of an unphysical ^3J-value restraining term
^3J	J-value or J-coupling constant
$^3J^0$	experimental J-value
J	general representation of a J-value
J^0	experimental J-value
ΔJ^0	width of flat-bottom for J-value restraining
a	a in Karplus relation
b	b in Karplus relation
c	c in Karplus relation
τ^s_{Jr}	period of scaling in periodically-scaled J-value restraining
Δt_ω	time period for which scaling is suspended in periodically-scaled J-value restraining
N_{le}	number of bins in J-value local elevation biasing
$w_{\zeta ni}$	weight of gaussian in J-value LE
$\mathcal{V}^{(Fxr)}$	$\mid F \mid$-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(exr)}$	ρ-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(sxr)}$	symmetry restraining potential energy contribution to $\mathcal{V}^{(phys)}$

Symbol	Meaning
k^{xr}	(harmonic) force constant for the crystallographic restraining
k^{sym}	harmonic force constant for the crystallographic symmetry restraining
F	Structure factor amplitude
ρ	Electron density
S	space group of a crystal
N_{sym}	Number of symmetry operations of a space group
$\mathbb{S}$	Symmetry operator $\mathbb{S} = \mathbf{R}\mathbf{r} + \mathbf{t}$
$\mathbf{R}$	Rotation matrix of a symmetry operator
$\mathbf{t}$	Translation vector of a symmetry operator
$\mathcal{V}^{(Sr)}$	S^2-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Sr)}$	Force constant of an unphysical S^2-value restraining term
S^2	S^2-order parameter
$S^{2,0}$	experimental S^2-value
S	general representation of a S^2-value
S^0	experimental S^2-value
$\mathcal{V}^{(df)}$	Distancefield restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(df)}$	Force due to the atom-atom distance restraints
$k^{(df)}$	Force constant of an unphysical distance-restraining term
l^0	Equilibrium distance of distance restraint
g_s	Distancefield grid distance
$\mathcal{V}^{(le)}$	Local elevation (LE) energy
$\mathcal{V}^{(bias)}$	bias energy
γ	LE basis function
$k^{(le)}$	LE force constant
$\mathbf{r}^{uc}$	unconstrained atomic positions
N_c	Number of constraints
N_{sh}	number of iterations of the SHAKE algorithm
d^0	constraint length
$\mathbf{f}^{uc}$	unconstrained atomic forces

The GROMOS Software for (Bio)Molecular Simulation

Volume 3: Force Field and Topology Data Set

January 9, 2021

Contents

CHAPTER 1

Introduction

In this volume the molecular model and the force field used in GROMOS are described. The GROMOS package comes with a number of standard data files. The ones involving the definition of a force field parameter set fall into two categories: interaction function parameter files (∗.ifp) and molecular topology building block files (∗.mtb). At least one of each of these types of files are required to build a molecular topology. This chapter continues with a short overview of the history of the GROMOS force field and a description of the background for its various versions. Chap. 2 summarizes the physical potential energy terms used in the GROMOS force fields. A detailed description of the Hamiltonian can be found in Vol. 2. The parameters contained in the interaction function parameter files of the 45A4 (45B4)[1–4] and 54A7 (54B7)[5–8] versions of the GROMOS force field are presented in Chap. 3. The corresponding molecular topology building blocks for the 54A7 force field are described in Chap. 4. Standard configurations of molecules or molecular systems are listed in Chap. 5.

1.1. GROMOS force fields

The GROMOS force field is continuously being tested and, if necessary, improved. From time to time a new version is brought out. The historic sequence of GROMOS force fields is the following:

- the 26C1 force field of June 1981
- the 37C2 (and 37D2) force field(s) of January 1983 (extended in November 1983)
- the 37C4 (and 37D4) force field(s) of November 1983 (revised in December 1985)
- the 43A1 (and 43B1) force field(s) of July 1996
- the 43A2 force field of October 2000
- the 45A3 (and 45B3) force field(s) of January 2001
- the 45A4 (and 45B4) force field(s) of July 2003
- the 53A5 (and 53B5) force field of July 2003
- the 53A6 (and 53B6) force field of July 2003
- the 54A7 (and 54B7) force field of April 2011
- the 54A8 force field of June 2012

The A-version of a force field is the basic force field designed for molecules in solution or in crystalline form. The B-version is derived from the A-version in order to be used for simulating molecules in vacuo, where the dielectric screening effect of the environment is neglected. The atomic charges and van der Waals parameters are changed such that atom charge groups with a non-zero total charge are neutralized while maintaining the hydrogen-bonding capacity of the individual atoms.

Since the functional form of the 43A1 and 43B1 force fields differs from that of the previous versions and the GROMOS file structure and formats were substantially changed in 1996, the force field versions older than 1996 have not been converted and have not been kept in the GROMOS package. For every force-field version, the complete set of interaction function parameters can be found in the corresponding file ∗.ifp. Building blocks of the 43A1 and 43A2, and the 43B1 and 43B2 force fields are given in the files 43a1.mtb and 43b1.mtb, respectively, and 45a3.mtb and 45b3.mtb present building blocks for the 45A3 and 45B3 force fields. In the newer versions of the GROMOS force field, building blocks are supplied via more than one file. These files contain building blocks that are categorized according to the kind of (sub)molecule they represent. For example, 45A4 building blocks can be found in: 45a4.mtb (α-amino acids, lipids, nucleotides and solvents), 45a4_carbo.mtb (carbohydrates and sugars), 45a4_beta.mtb (β-amino acids) and 45a4_cof.mtb (cofactors and other types of molecules).

Since the introduction of the 43A1 and 43B1 versions, the basic functional form of the GROMOS force field has been kept constant. It is described in Chap. 2-5 to Chap. 2-9 of Vol. 2. Chap. 3 and Chap. 4 of this Volume present the force field parameters and building blocks of the 54A7 force field. In the remaining of the current chapter, the events leading up to these force fields are sketched.

1.2. Development of the GROMOS force field

In the first GROMOS force field, 26C1, only 26 atom types were defined[9,10]. The molecular topology building block file contained only amino acid residues and a heme group. It was meant for simulation of proteins in aqueous solution or crystalline form.

The 37C2 force field was an extension of the 26C1 force field in order to allow for simulation of nucleotides, sugars, etc. Eleven new atom types were added. Some interaction function parameters were slightly changed, which made the version number change from 1 to 2. The molecular topology building block file contained many more building blocks. It was meant for simulation of proteins, DNA, sugars in aqueous solution or crystalline form.

The 37D2 force field was the one corresponding to the 37C2 one, but adapted in order to be used for simulations of molecules in vacuo.

In the previous force fields the repulsive part of the van der Waals interaction between third neighbour atoms (1-4 interaction) was too large, in case one or both of the atoms involved was an extended (CH1, CH2, CH3, CR1) carbon atom and the torsion angle 1-2-3-4 was in a cis-conformation. This effect was redressed by changing the programs such that for 1-4 or third-neighbour interactions, van der Waals parameters can be used which are different from the normal ones. These extra 1-4 van der Waals parameters had to be given on the interaction function parameter file. This change made the force field version number change from 2 to 4. It was meant for simulation of proteins, DNA, sugars, etc. in solution or crystalline form.

The 37D4 force field was the one corresponding to the 37C4 one, but adapted in order to be used for simulations of molecules in vacuo.

The 43A1 force field constitutes a significant change with respect to the 37C4 one, and differs from it in a number of aspects.

1. The non-polar solute atoms appeared to be slightly too hydrophilic[11]. Therefore, the C12(I,J) van der Waals parameter, where I denotes a non-polar atom type and J denotes a water oxygen, was enlarged.
2. The description of aromatic rings which was based on the use of united atoms, was improved by introducing explicit hydrogen atoms on some aromatic rings[11].
3. In order to reduce the rotational motion of the peptide plane, the dihedral angle torsional force constants for the φ, ψ dihedrals were slightly increased.
4. The force field parameters in the heme-group were slightly changed.
5. The functional forms of the covalent bond-stretching interaction and bond-angle bending interaction were changed in order to improve computational efficiency and to avoid singularities in the forces for (ideal) bond angles of 180° (see section Sec. 2.3).
6. The nomenclature (definition of improper dihedral angles) of the Leu and Val side chains was changed such that it corresponds to the IUPAC-IUB convention.
7. Due to the introduction of the distinction between mass atom types and non-bonded van der Waals atom types and by relinquishing the use of non-bonded atom types for the definition of bond, bond-angle and (improper) dihedral- angle types, the number of different (van der Waals) atom types could be reduced from 37 to 22.
8. New van der Waals atom types were added, especially to allow for the use of different solvents (apart from water). This brought the number of non-bonded (van der Waals) atom types to 43.

Shortly after the release of the 43A1 and 43B1 force fields, some small changes in the torsional-angle parameters and the third-neighbour-van der Waals interaction were introduced, in order to better reproduce the distribution of the torsional-angle values in short aliphatic chains. This modification resulted in the

43A2 parameter set.[12]. As it was then shown that the density for the longer alkanes was too high, a reparametrisation of the aliphatic united atoms followed, introducing two additional atom types for branched and cyclic alkanes. This resulted in the 45A3 and 45B3 set of parameters[1].
Several parametrisation efforts on different classes of molecular systems were subsequently collected in the 45A4 and 45B4 set of parameters. The most important changes involved

- Charges and torsional dihedral angles in nucleotides and common co-factors[3]
- Charges and torsional dihedral angles in carbohydrates[4]
- Charges and torsional dihedral angles in lipids[2]
- Modifications in the choice of polar/nonpolar C12 interaction parameters for atom type 6 (NT)
- Correction to the van der Waals interaction parameters for atom type 31 (BR)
- Modifications to the heme group covalent interactions
- A new definition of the molecular topology building block at the end of polypeptide, polynucleotide or polysaccharide chains

The 53A5 and 53A6 force fields[5] are the result of a complete reparametrisation of the non-bonded interaction parameters for condensed phase simulations of pure liquids of small molecules (53A5) and solutions of molecular systems in water or apolar solvents (53A6). All interaction types have been redefined in these force fields, which also include parameters for additional solvents. In addition, bond types, bond-angle types, dihedral-angle types and atom-types have been renumbered in 53A5 and 53A6.

In the 54A7 force field[6–8]

- The 53A6 helical propensities are corrected through new phi/psi torsional angle terms and a modification of the N-H, C=O repulsion.
- A new atom type for a charged -CH3 in the choline moiety is added.
- The Na+ and Cl- ions are modified to reproduce the free energy of hydration.
- Additional improper torsional angle types for free energy calculations involving a chirality change are introduced.
- For the cofactors the files 54c7_cof.mtb and 54d7_cof.mtb were introduced in which the partial charges were updated according to analogy of functional groups. Files 54a7_cof.mtb and 54b7_cof.mtb still contain the original charge distributions.

The 54A8 force field[13] involves a recalibration of the nonbonded interaction parameters for the charged amino-acid side chains, based on ionic side chain analogs. After a thorough analysis of the available experimental data, conventional hydration free energies for the ammonium; mono-, di-, tri-, and tetramethylammonium; formate; acetate; propanoate; imidazolium; and guanidinium ions were combined with a standard absolute intrinsic proton hydration free energy to yield absolute intrinsic single-ion hydration free energies serving as experimental target data. The raw hydration free energies calculated from atomistic simulations are affected by electrostatic and finite-size artifacts, and corrections were applied to reach methodological independence prior to comparison with these experimental values.

Solvent models that are consistent with the GROMOS biomolecular force fields are available for much used (co)-solvents[14]:

- water[15,16]
- methanol[17]
- DMSO[18]
- chloroform[19]
- carbontetrachloride[20]
- urea[21]
- acetonitrile[22]
- dimethylsulfone[23]

Polarisable (solvent) models consistent with the GROMOS biomolecular force field are available for:

- water[24]
- methanol[25]
- DMSO[26]

- chloroform[27]
- carbontetrachloride[28]
- urea[29]
- acetone[30]
- n-alkanes[31]

Supra-molecular polarisable coarse-grained solvent models compatible with the GROMOS biomolecular force fields are available for:

- water[32]
- methanol[33,34]
- DMSO[33]
- chloroform[33]

Supra-atomic polarisable coarse-grained models compatible with the GROMOS biomolecular force field are available for n-alkanes[35] and cyclohexane.[36]

CHAPTER 2

Physical forces: GROMOS force field

2.1. Introduction

This chapter summarizes the functional form of the GROMOS force field terms, which are described in detail in the following chapters of Vol. 2: The bonded interaction force-field terms are described in Chap. 2-5; van der Waals interactions are described in Chap. 2-6; electrostatic interactions are described in Chap. 2-7; forces between coarse-grained particles are described in Chap. 2-8; the special force-field terms are described in Chap. 2-9.

2.2. Bond stretching force-field terms

The potential energy (force-field) term associated with *bond stretching interactions* is the term $\mathcal{V}^{(b)}(\boldsymbol{r};\mathbf{s})$ in Eq. 2.1. It is given by

$$\mathcal{V}^{(b)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N^{(b)}} V^{(b)}(b_n; k_n^{(b)}, b_n^0) \ , \tag{2.1}$$

where $N^{(b)}$ is generally equal to the total number of all covalent bonds present in the system, *i.e.* each covalent bond is associated with one and only one stretching term in the GROMOS force-field, and $V^{(b)}$ is the function describing the potential energy associated with the stretching of a single bond. The quantity $b_n \doteq b_n(\boldsymbol{r})$ represents the length of bond n in the given system configuration, *i.e.* the distance between the two atoms $i \doteq i(n)$ and $j \doteq j(n)$ connected by the covalent bond n (minimum-image distance if PBC is applied). The quantities $k_n^{(b)}$ and b_n^0 represent force-field parameters, force constant and reference length, respectively characteristic for the specific bond n, as encoded by a corresponding *bond type code* $M_n^{(b)}$, *i.e.* one may write $k_n^{(b)} \doteq k^{(b)}(M_n^{(b)}, \mathbf{s})$ and $b_n^0 \doteq b^0(M_n^{(b)}, \mathbf{s})$. Two different expression can be used for the function $V^{(b)}$ in GROMOS (see Sec. 2-5.1), a quartic function with force constant $k^{(b)}$ and a harmonic function with force constant $k^{(b,h)}$. The coarse-grained (CG) model exploits yet another (quartic) interaction term for the bond between the central particle and the dipole particle of a CG bead.[32]

For reasons of ease of analysis, the list of $N^{(b)}$ covalent bonds is split into two lists, one of bonds involving hydrogen atoms (defined as having mass atom type code 1, see Tab. 3.1), and one involving the other bonds. These lists are kept in the molecular topology file (see Vol. 4). The first list contains NBONH bonds involving hydrogen atoms. Three items are stored: IBH, JBH[1..NBONH] are the atom sequence numbers of the atoms forming bond i-j as a function of the bond sequence number n, and ICBH[1..NBONH] is the bond-type code, denoting the parameters $k_n^{(b)}$, $k_n^{(b,h)}$ and b_n^0 , as a function of the bond sequence number n. The list for the bonds involving no hydrogen atoms contains corresponding items denoted by IB, JB, ICB[1..NBON]. The force field parameters $k_n^{(b)}$, $k_n^{(b,h)}$ and b_n^0 for the various types of covalent bonds are stored in CB[1..NBTY], HB[1..NBTY] and B0[1..NBTY], as a function of the bond-type code (ICBH or ICB). They can be found in the interaction function parameter files ∗.ifp. For the GROMOS force fields 45A4 and 45B4 they are listed in Tab. 3.2, for force fields 54A7 and 54B7 they are listed in Tab. 3.17.

Program MD++ reads values for $k_n^{(b)}$, $k_n^{(b,h)}$ and b_n^0 from the BONDSTRETCHTYPE block in the molecular topology file (∗.top). It can also read $k_n^{(b)}$ and b_n^0 from the BONDTYPE block and $k_n^{(b,h)}$ and b_n^0 from the HARMBONDTYPE block. If only BONDTYPE block or HARMBONDTYPEBLOCK are given, the missing force constant is calculated using equations Eq. 2-18.3 and Eq. 2-18.4.

2.3. Bond-angle bending force-field terms

The potential energy (force-field) term associated with *bond-angle bending interactions* is the term $\mathcal{V}^{(\theta)}(\boldsymbol{r};\mathbf{s})$ in Eq. 2.2. It is given by

$$\mathcal{V}^{(\theta)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N^{(\theta)}} V^{(\theta)}(\theta_n; k_n^{(\theta)}, \theta_n^0) \ , \tag{2.2}$$

where $N^{(\theta)}$ is generally equal to the total number of all covalent bond-angles present in the system, *i.e.* each definable covalent bond-angle is associated with one and only one bending term in the GROMOS force field, and $V^{(\theta)}$ is the function describing the potential energy associated with the bending of a single bond angle. The quantity $\theta_n \doteq \theta_n(\boldsymbol{r})$ represents the value of bond angle n in the given system configuration, *i.e.* the angle formed by the three atoms $i \doteq i(n)$, $j \doteq j(n)$ and $k \doteq k(n)$ defining the covalent bond angle n (minimum-image triplet if PBC is applied). The quantities $k_n^{(\theta)}$ and θ_n^0 represent force-field parameters, force constant and reference bond angle, respectively, characteristic for the specific bond angle n, as encoded by a corresponding *bond-angle type code* $M_n^{(\theta)}$*i.e.* one may write $k_n^{(\theta)} \doteq k^{(\theta)}(M_n^{(\theta)}, \mathbf{s})$ and $\theta_n^0 \doteq \theta^0(M_n^{(\theta)}, \mathbf{s})$. Two different expression can be used for the function $V^{(\theta)}$ in GROMOS (see Sec. 2-5.2), a cosine-harmonic function with force constant $k^{(\theta)}$ and a harmonic function with force constant $k^{(\theta,h)}$.

For reasons of ease of analysis, the list of $N^{(\theta)}$ bond angles is split into two lists, one of bond angles involving hydrogen atoms (defined as having mass atom type code 1, see Tab. 3.1), and one involving the other bond angles. These lists are kept in the molecular topology file (Volume 4). The first list contains NTHEH bond angles involving hydrogen atoms. Four items are stored: ITH, JTH, KTH[1...NTHEH] are the atom sequence numbers of the atoms forming bond angle i-j-k as a function of the bond-angle sequence number n, and ICTH [1...NTHEH] is the bond-angle type code, denoting the parameters $k_n^{(\theta)}$, $k_n^{(\theta,h)}$ and θ_n^0 as a function of the bond-angle sequence number n. The list for the bond angles involving no hydrogen atoms contains corresponding items denoted by IT, JT, KT, ICT[1...NTHE]. The force field parameters $k_n^{(\theta)}$, $k_n^{(\theta,h)}$ and θ_n^0 for the various types of bond angles are stored in CT[1...NTTY], CHT[1...NTTY] and T0[1 ... NTTY] as a function of the bond-angle type code (ICTH or ICT). They can be found in the interaction parameter files $*$.ifp. For the GROMOS force fields 45A4 and 45B4, they are listed in Tab. 3.3, for force fields 54A7 and 54B7 they are listed in Tab. 3.18.

Program MD++ reads values for $k_n^{(\theta)}$, $k_n^{(\theta,h)}$ and θ_n^0 from the BONDANGLEBENDTYPE block in the molecular topology file ($*$.top). If no BONDANGLEBENDTYPE is present $k_n^{(\theta)}$ can be read from the BONDANGLETYPE block and $k_n^{(\theta,h)}$ can be read from the HARMBONDANGLETYPE block.

2.4. Improper dihedral-angle bending force-field term

The potential energy (force-field) term associated with *improper dihedral-angle bending interactions*, *i.e.* typically controlling out-of-plane or out-of-tetrahedron distortions, is the term $\mathcal{V}^{(\xi)}(\boldsymbol{r};\mathbf{s})$ in Eq. 2.3. It is given by

$$\mathcal{V}^{(\xi)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N^{(\xi)}} V^{(\xi)}(\xi_n; k_n^{(\xi)}, \xi_n^0) \ , \tag{2.3}$$

where $N^{(\xi)}$ generally corresponds to a subset of all possibly definable improper dihedral angles in the system (see below; note, however, each definable covalent improper dihedral angle is associated with at most one bending term in the GROMOS force field), and $V^{(\xi)}$ is the function describing the potential energy associated with the bending of a single improper dihedral angle. The quantity $\xi_n \doteq \xi_n(\boldsymbol{r})$ represents the value of improper dihedral angle n in the given system configuration, *i.e.* the dihedral angle formed by the four atoms $i \doteq i(n)$, $j \doteq j(n)$, $k \doteq k(n)$ and $l \doteq l(n)$ defining the covalent improper dihedral angle n (minimum-image quadruplet if PBC is applied). The quantities $k_n^{(\xi)}$ and ξ_n^0 represent force-field parameters, force constant and reference improper dihedral-angle, respectively, characteristic for the specific improper dihedral angle n, as encoded by a corresponding *improper dihedral-angle type code* $M_n^{(\xi)}$*i.e.* one may write $k_n^{(\xi)} \doteq k^{(\xi)}(M_n^{(\xi)}, \mathbf{s})$ and $\xi_n^0 \doteq \xi^0(M_n^{(\xi)}, \mathbf{s})$. The function $V^{(\xi)}$ is always a *harmonic* function in GROMOS.
The improper dihedral angle definitions can be found in the molecular topology building block files $*$.mtb.

For reasons of ease of analysis, the list of $N^{(\xi)}$ improper dihedral angles is split into two lists, one of improper dihedrals involving hydrogen atoms (defined as having mass atom type code 1, see Tab. 3.1), and one involving the other one involving the other improper dihedrals. These lists are kept in the molecular topology file (Volume 4). The first list contains NQHIH improper dihedral angles involving hydrogen atoms. Five items are stored: IQH, JQH, KQH, LQH[1...NQHIH] are the atom sequence numbers of the atoms forming improper dihedral i-j-k-l as a function of the improper dihedral sequence number n, and ICQH[1...NQHIH] is the improper dihedral type code, denoting the parameters $k_n^{(\xi)}$ and ξ_n^0 , as a function of the improper dihedral sequence number n. The list for the improper dihedral angles involving no hydrogen atoms contains corresponding items denoted by IQ, JQ, KQ, LQ, ICQ[1...NQHI]. The force field parameters $k_n^{(\xi)}$ and ξ_n^0 for the various types of improper dihedrals are stored in CQ[1...NQTY] and Q0[1...NQTY] as a function of the improper dihedral type code (ICQH or ICQ). They can be found in the interaction parameter files $*$.ifp. For the GROMOS force fields 45A4, 45B4, 54A7 and 54B7 they are listed in Tab. 3.4 (and Tab. 3.19).

Program MD++ reads values for $k_n^{(\xi)}$, and ξ_n^0 from the IMPDIHEDRALTYPE block in the molecular topology file ($*$.top).

2.5. Proper dihedral-angle torsion force-field term

The potential energy (force-field) term associated with *proper dihedral-angle bending interactions*, *i.e.* typically controlling, in balance with non-bonded interactions, the rotational barriers around covalent bonds, is the term $\mathcal{V}^{(\varphi)}(\boldsymbol{r};\mathbf{s})$ in Eq. 2.4. It is given by

$$\mathcal{V}^{(\varphi)}(\boldsymbol{r};\mathbf{s}) = \sum_{n=1}^{N^{(\varphi)}} V^{(\varphi)}(\varphi_n; k_n^{(\varphi)}, \varphi_n^0, m_n^{(\varphi)}) \ , \tag{2.4}$$

where $N^{(\varphi)}$ generally corresponds to a subset of all possibly definable proper dihedral angles in the system and $V^{(\varphi)}$ is the function describing the potential energy contribution of the term to the torsion of the corresponding proper dihedral angle. The quantity $\varphi_n \doteq \varphi_n(\boldsymbol{r})$ represents the value of proper dihedral angle n in the given system configuration, *i.e.* the dihedral angle formed by the four atoms $i \doteq i(n)$, $j \doteq j(n)$, $k \doteq k(n)$ and $l \doteq l(n)$ defining the covalent proper dihedral angle n (minimum-image quadruplet if PBC is applied). Note that the sign of the dihedral angle as defined by Eq. 2-5.19 follows the IUPAC-IUB convention[37], and that the proper dihedral angle is undefined if either $r_{im'} = 0$ or $r_{in'} = 0$. The quantities $k_n^{(\varphi)}$, φ_n^0 and $m_n^{(\varphi)}$ represent force-field parameters (force constant, reference dihedral-angle, and multiplicity, respectively; the reference dihedral angle is also called the phase shift; the multiplicity is a positive non-zero integer) characteristic for the specific proper dihedral angle term n, as encoded by a corresponding *proper dihedral-angle type code* $M_n^{(\varphi)}$. Two different expression can be used for the function $V^{(\varphi)}$ in GROMOS (see Sec. 2-5.4).

The torsional dihedral angle definitions can be found in the molecular topology building block files $*$.mtb (see Vol. 4). Examples of the special definitions involving sugar or phosphor atoms can be found in the building blocks DADE or NADPH.

Program MD++ reads values for $k_n^{(\varphi)}$, φ_n^0 and $m_n^{(\varphi)}$ from the TORSDIHEDRALTYPE block in the molecular topology file ($*$.top), also if the DIHEDRALTYPE block is given as well. If the DIHEDRALTYPE is given in the topology instead of the TORSDIHEDRALTYPE, only cos φ_n values (instead of φ_n values) are read.

For reasons of ease of analysis, the list of $N^{(\varphi)}$ torsional dihedral angles is split into two lists, one of dihedrals involving hydrogen atoms (defined as having mass atom type code 1, see Table Tab. 3.1), and one involving the other dihedrals. These lists are kept in the molecular topology file (Vol. 4). The first list contains NPHIH dihedral angles involving hydrogen atoms. Five items are stored: IPH, JPH, KPH, LPH[1..NPHIH] are the atom sequence numbers of the atoms forming dihedral i-j-k-l as a function of the dihedral sequence number n, and ICPH[1..NPHIH] is the dihedral type code, denoting the parameters $k_n^{(\varphi)}$, φ_n^0 and $m_n^{(\varphi)}$, as a function of the dihedral sequence number n. The list for the dihedrals involving no hydrogen atoms contains corresponding items denoted by IP, JP, KP, LP, ICP[1..NPHI]. The force field parameters $k_n^{(\varphi)}$, φ_n^0 and $m_n^{(\varphi)}$ for the various types of torsional dihedrals are stored in CP[1..NPTY], NP[1..NPTY] and PD[1..NPTY] as a function of the torsional dihedral type code (ICPH or ICP). They can be found in the

interaction parameter files ∗.ifp. For the GROMOS force fields 45A4 and 45B4, they are listed in Tab. 3.5, for the GROMOS force fields 54A7 and 54B7, they are listed in Tab. 3.20.

As an additional feature, the specification at so-called cross-dihedral terms is supported. The corresponding expression for this type of interaction reads

$$\begin{aligned} V^{trig,cross}(\mathbf{r};s) &= \sum_{n=1}^{N_c} V_n^{trig,cross}\ (\varphi_n;\psi_n;K_{c_n};\delta_n;m_n) \\ &= \sum_{n=1}^{N_c} K_{c_n}\ [1+\cos\ (m_n(\varphi_n+\psi_n)-\delta_n)] \end{aligned} \tag{2.5}$$

The summation runs over the set of $n = 1, \ldots, N_c$ of coupled dihedral angles φ_n and ψ_n, as specified by atoms a-b-c-d and e-f-g-h, respectively, which are specified in the CROSSDIHEDRALH and CROSSDIHEDRAL blocks in the molecular topology file (∗.top). These blocks specify coupled dihedral angles that do involve hydrogens and do not involve hydrogens, respectively. In addition, the type of cross-dihedral term n is specified in the same blocks in the topology file, defining the force constant K_{c_n}, phase-shift δ_n and multiplicity m, as read from the TORSDIHEDRALTYPE block in the molecular topology file. Accordingly, the specification of cross-dihedral terms is only possible if the TORSDIHEDRALTYPE block is specified.

The forces on atoms a, b, c, d of dihedral φ_n and atoms e, f, g, h of dihedral ψ_n due to the n-th in (Eq. 2.5) are

$$\begin{aligned} \mathbf{f}_a &= -\frac{\partial V^{\ trig,cross}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_a} \\ &= K_{c_n}m_n sin(m_n(\varphi_n+\psi_n)-\delta_n)\frac{r_{cb}}{r_{mb}^2}\ \mathbf{r}_{mj} \end{aligned} \tag{2.6}$$

$$\begin{aligned} \mathbf{f}_d &= -\frac{\partial V^{\ trig,cross}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_d} \\ &= -K_{c_n}m_n sin(m_n(\varphi_n+\psi_n)-\delta_n)\frac{r_{cb}}{r_{mb}^2}\ \mathbf{r}_{nc} \end{aligned} \tag{2.7}$$

$$\begin{aligned} \mathbf{f}_b &= -\frac{\partial V^{\ trig,cross}}{\partial\varphi_n}\frac{\partial\varphi_n}{\partial\mathbf{r}_b} \\ &= \left[\frac{(\mathbf{r}_{ab}\ \cdot\ \mathbf{r}_{cb})}{r_{cb}^2}\ -1\right]\ \mathbf{f}_i\ -\frac{\mathbf{r}_{cd}\ \cdot\ \mathbf{r}_{cb}}{r_{cb}^2}\mathbf{f}_d \end{aligned} \tag{2.8}$$

$$\mathbf{f}_c = -\mathbf{f}_a\ -\mathbf{f}_b\ -\mathbf{f}_d \tag{2.9}$$

$$\begin{aligned} \mathbf{f}_e &= -\frac{\partial V^{\ trig,cross}}{\partial\psi_n}\frac{\partial\psi_n}{\partial\mathbf{r}_e} \\ &= K_{c_n}m_n sin(m_n(\varphi_n+\psi_n)-\delta_n)\frac{r_{gf}}{r_{mf}^2}\ \mathbf{r}_{mf} \end{aligned} \tag{2.10}$$

$$\begin{aligned} \mathbf{f}_h &= -\frac{\partial V^{\ trig,cross}}{\partial\psi_n}\frac{\partial\psi_n}{\partial\mathbf{r}_h} \\ &= -K_{c_n}m_n sin(m_n(\varphi_n+\psi_n)-\delta_n)\frac{r_{gf}}{r_{ng}^2}\ \mathbf{r}_{ng} \end{aligned} \tag{2.11}$$

$$\begin{aligned}\mathbf{f}_f &= -\frac{\partial V^{\,trig,cross}}{\partial \psi_n}\frac{\partial \psi_n}{\partial \mathbf{r}_f} \\ &= \left[\frac{(\mathbf{r}_{ef}\ \cdot\ \mathbf{r}_{gf})}{r_{gf}^2}\ -1\right]\ \mathbf{f}_e\ -\frac{(\mathbf{r}_{gh}\ \cdot\ \mathbf{r}_{gf})}{r_{gf}^2}\mathbf{f}_h\end{aligned} \tag{2.12}$$

$$\mathbf{f}_g = -\mathbf{f}_e\ -\mathbf{f}_f\ -\mathbf{f}_h \tag{2.13}$$

where

$$\mathbf{r}_{mb} = \mathbf{r}_{ab} \times \mathbf{r}_{cb} \tag{2.14}$$
$$\mathbf{r}_{nc} = \mathbf{r}_{cb} \times \mathbf{r}_{cd} \tag{2.15}$$
$$\mathbf{r}_{mf} = \mathbf{r}_{ef} \times \mathbf{r}_{gf} \tag{2.16}$$
$$\mathbf{r}_{ng} = \mathbf{r}_{gf} \times \mathbf{r}_{gh} \tag{2.17}$$

and

$$sin(m_n(\varphi_n + \psi_n) - \delta_n) = \sqrt{1 - cos^2(m_n(\varphi_n + \psi_n) - \delta_n)} \tag{2.18}$$

2.6. Non-bonded interactions

The term in the interaction function that represents the *non-bonded interaction* is a sum of contributions from van der Waals and electrostatic interactions,

$$\begin{aligned}&V^{nonb}(\mathbf{r}^{\mathbf{N}}; s) = \\ &\sum_{\substack{nonbonded\\ pairs(i,j)}} \big\{V^{LJ}(r_{ij}; C_{12}(i,j), C_6(i,j), R_{cp}, R_{cl}) \\ &\qquad +V^{CRF}(r_{ij}; q_i, q_j, R_{cp}, R_{cl}, R_{rf}, \varepsilon_1, \varepsilon_2, \kappa)\big\}\end{aligned} \tag{2.19}$$

with

$$V^{LJ} = \left[\frac{C_{12}(i,j)}{(r_{ij})^6} - C_6(i,j)\right]\frac{1}{(r_{ij})^6} \tag{2.20}$$

and

$$V^{CRF} = \frac{q_i q_j}{4\pi\varepsilon_0\varepsilon_{cs}}\left[\frac{1}{r_{ij}} - \frac{\frac{1}{2}C_{rf}(r_{ij})^2}{R_{rf}^3} - \frac{1-\frac{1}{2}C_{rf}}{R_{rf}}\right] \tag{2.21}$$

The van der Waals interactions are discussed in Chap. 2-6 and the electrostatic interactions are discussed in Chap. 2-7.

2.6.1. van der Waals parameters. The non-bonded interaction van der Waals parameters C_{12}(i,j) and C_6(i,j) in formula (Eq. 2.19) depend on the atom type or more specifically the integer atom codes I = IAC[i] and J = IAC[j] of the atoms with atom sequence numbers i and j. The integer atom codes of the various types of atoms in the GROMOS force fields 45A5 and 45B4 are listed in Tab. 3.6, for GROMOS force fields 54A7 and 54B7, they are listed in Tab. 3.21.

Lists of integer atom codes are kept in the molecular topology file (Volume 4). For the NRP atoms of the "solute" part of the molecular topology the integer atom codes are stored in IAC[1..NRP]. The integer atom codes of the NRAM solvent atoms are stored in IACS[1..NRAM]. The van der Waals parameters are kept in the molecular topology file (see Volume 4). For the NRATT atom types, C12[1 .. NRATT∗(NRATT+1)/2] contains the coefficient C_{12} in (Eq. 2.19) as a function of the occurring pair codes; the sequence of atom pairs with integer atom codes ranging from 1 to NRATT is:

1-1, 1-2, 2-2, ..., 1-NRATT, 2-NRATT, ..., NRATT-NRATT.

The coefficients C_6 in (Eq. 2.19) are kept likewise in C6[1 .. NRATT∗(NRATT+1)/2]. In this way it is possible to change the van der Waals interaction between each pair of atom types independently. Basically, the GROMOS van der Waals parameters for an atom pair with integer atom codes I and J are derived from single atom van der Waals parameters using the relations

$$C_6(I,J) = \sqrt{C_6^{\frac{1}{2}}(I,I)C_6^{\frac{1}{2}}(J,J)} \tag{2.22}$$

and

$$C_{12}(I,J) = \sqrt{C_{12}^{\frac{1}{2}}(I,I)C_{12}^{\frac{1}{2}}(J,J)} \tag{2.23}$$

For the GROMOS force fields 45A4 and 45B4, the single atom van der Waals parameters $(C6(I,I))^{1/2}$ and $(C_{12}(I,I))^{1/2}$ are given in the third and fourth column of Tables Tab. 3.7 and Tab. 3.8 as a function of integer atom code or non-bonded atom type. For the GROMOS force fields 54A7 and 54B7, they are given in the third and fourth column of Tables Tab. 3.22 and Tab. 3.23.

GROMOS also offers a possibility to specify the van der Waals parameters for a specific atom pair, thereby overruling the interaction parameters as derived from the normal (or third-neighbour) interaction parameters. This can be done by introducing a LJEXCEPTIONS block in the molecular topology file (see Vol. 4).

2.6.2. Atomic charges and charge groups. Lists of atomic charges are kept in the molecular topology file (see Volume 4). For the NRP atoms in the "solute" part of the molecular topology atomic charges are stored in CG[1..NRP] (multiplied by $(4\pi\varepsilon_0)^{-1/2}$). The atomic charges of the NRAM solvent atoms are stored in CGS[1..NRAM] (multiplied by $(4\pi\varepsilon_0)^{-1/2}$).

When the (partial) atomic charges of a group of atoms add up to exactly zero, the leading term of the electrostatic interaction between two such groups of atoms is of dipolar ($1/r^3$) character. The sum of the 1/r monopole contributions of the various atom pairs to the group-group interaction will be zero. Therefore, the range of the electrostatic interaction can be considerably reduced when atoms are assembled in so-called charge groups, which have a zero net charge, and for which the electrostatic interaction with other (groups of) atoms is either calculated for all atoms of the charge group or for none.

The GROMOS force fields make use of this *concept of charge groups*. The atoms that belong to a charge group are chosen such that their partial atomic charges add up to zero. For groups of atoms with a total charge of $+e$ or $-e$, like the sidechain atoms of Arg or Asp, the partial atomic charges of the charge group may add up to $+e$ or $-e$. In GROMOS, the non-bonded interactions are calculated between charge groups only. When a cut-off radius is used, the distance between two charge groups must be defined. The *position of a charge group* is defined differently for a charge group belonging to the "solute" part of the molecular topology and one in the "solvent" part of the molecular topology.

- The position of a "solute" charge group is taken to be its centre of geometry:

  $$R_{cg} = \sum_{i=1}^{N_{cg}} \mathbf{r}_i \; / N_{cg} \tag{2.24}$$

  where the number of atoms belonging to the charge group is denoted by N_{cg}.
- The position of a "solvent" charge group is taken to be the position of the first atom of a solvent molecule. A "solvent" molecule may only contain one charge group.

Since each solvent molecule consists of one charge group, the "solvent" part of the molecular topology file does not need to contain information on "solvent" charge groups. In the "solute" part of the molecular topology file the charge group information is kept in the following way. It is assumed that atoms belonging to one charge group have sequential atom sequence numbers. The last atom of any charge group is denoted by a charge group code value of 1. All other atoms have a charge group code value of 0. This requirement of

atoms of a charge group to have sequential atom sequence numbers is a less elegant restriction to choosing the atom sequence when defining molecular topology building blocks or molecular topologies (see Vol. 4).

The atomic charges and charge group definitions for the GROMOS force fields are given in the molecular topology building block files ∗.mtb (Chap. 3). The atomic charges and charge group definitions for amino acid residues, various solvents and nucleotides of the 45A4 and 45B4 GROMOS force fields are listed in Tables Tab. 3.12-Tab. 3.16. For the GROMOS force fields 54A7 and 54B7 they are listed in Tables Tab. 3.27-Tab. 3.31.

The charges for the B-versions of the force field are given between parentheses. The atoms for which no changes are listed have zero partial charge and form single atom or multiple atom charge groups.

CHAPTER 3

GROMOS interaction function parameters

mass atom type code	mass in a.m.u.	mass atom name
1	1.008	H
3	13.019	CH1
4	14.027	CH2
5	15.035	CH3
6	16.043	CH4
12	12.011	C
14	14.0067	N
16	15.9994	O
19	18.9984	F
23	22.9898	NA
24	24.305	MG
28	28.08	SI
31	30.9738	P
32	32.06	S
35	35.453	CL
39	39.948	AR
40	40.08	CA
56	55.847	FE
63	63.546	CU
65	65.37	ZN
80	79.904	BR

TABLE 3.1. GROMOS mass atom type codes, masses and names.

Bond-type code	Force constant	Ideal bond length	Examples of usage in terms of non-bonded atom types	
$M_n^{(b)}$	$k_n^{(b,q)}$	b_n^0		$k_n^{(b,h)}$
	[10^6 kJ·mol^{-1}·nm^{-4}]	[nm]		[10^6 kJ·mol^{-1}·nm^{-2}]
ICBH[N] ICB[N]	CB[N]	B0[N]		
1	15.7	0.100	H - OA	0.314
2	18.7	0.100	H - N (all)	0.374
3	12.3	0.109	HC - C	0.292
4	16.6	0.123	C - O	0.502
5	13.4	0.125	C - OM	0.419
6	12.0	0.132	CR1 - NR (6-ring)	0.418
7	8.87	0.133	H - S	0.314
8	10.6	0.133	C - NT, NL	0.375
9	11.8	0.133	C, CR1 - N, NR, CR1, C (peptide, 5-ring)	0.417
10	10.5	0.134	C - N, NZ, NE	0.377
11	11.7	0.134	C - NR (no H) (6-ring)	0.420
12	10.2	0.136	C - OA	0.377
13	11.0	0.138	C - NR (heme)	0.419
14	8.66	0.139	CH2 - C, CR1 (6-ring)	0.335
15	10.8	0.139	C, CR1 - CH2, C, CR1 (6-ring)	0.417
16	8.54	0.140	C, CR1, CH2 - NR (6-ring)	0.335
17	8.18	0.143	CHn - OA	0.335
18	9.21	0.143	CHn - OM	0.377
19	6.10	0.1435	CHn - OA (sugar)	0.251
20	8.71	0.147	CHn - N, NT, NL, NZ, NE	0.376
21	5.73	0.148	CHn - NR (5-ring)	0.251
22	7.64	0.148	CHn - NR (6-ring)	0.335
23	8.60	0.148	O, OM - P	0.377
24	8.37	0.150	O - S	0.377
25	5.43	0.152	CHn - CHn (sugar)	0.251
26	7.15	0.153	C, CHn - C, CHn	0.335
27	4.84	0.161	OA - P	0.251
28	4.72	0.163	OA - SI	0.251
29	5.94	0.178	CH3 - S	0.376
30	5.62	0.183	CH2 - S	0.376
31	3.59	0.187	CH1 - SI	0.251
32	0.640	0.198	NR - FE	0.0502
33	5.03	0.204	S - S	0.419
34	0.628	0.200	NR (heme) - FE	0.0502
35	23.2	0.100	HWat - OWat	0.464
36	12.1	0.110	HChl - CChl	0.293
37	8.12	0.1758	CChl - CLChl	0.502
38	8.04	0.153	ODmso - SDmso	0.376
39	4.95	0.195	SDmso - CDmso	0.376
40	8.10	0.176	CCl4 - CLCl4	0.502

TABLE 3.2: continues on next page.

Bond-type code	Force constant	Ideal bond length	Examples of usage in terms of non-bonded atom types	
$M_n^{(b)}$	$k_n^{(b,q)}$	b_n^0		$k_n^{(b,h)}$
	$[10^6$ kJ·mol^{-1}·nm$^{-4}]$	[nm]		$[10^6$ kJ·mol^{-1}·nm$^{-2}]$
ICBH[N] ICB[N]	CB[N]	B0[N]		
41	8.71	0.163299	HWat - HWat	0.465
42	2.68	0.233839	HChl - CLChl	0.293
43	2.98	0.290283	CLChl - CLChl	0.502
44	2.39	0.280412	ODmso - CDmso	0.376
45	2.19	0.292993	CDmso - CDmso	0.376
46	3.97	0.198842	HMet - CMet	0.314
47	3.04	0.287407	CLCl4 - CLCl4	0.502
48	0.540	0.221	NR (His) - FE	0.0527
49	2.72	0.178	FE - C (CO bound to heme)	0.172
50	37.0	0.112	C - O (CO bound to heme)	0.928

TABLE 3.2: GROMOS 45A4 and 45B4 bond-stretching parameters ($k_n^{(b,q)} = k_n^{(b,h)}/(2b_n^{0\,2})$).

Bond-angle type code	Force constant	Ideal bond angle	Example of usage in terms of non-bonded atom types	
$M_n^{(\theta)}$	$k_n^{(\theta,c)}$	θ_n^0		$k_n^{(\theta,h)}$
	[kJ·mol^{-1}]	[deg]		[kJ·mol^{-1}·deg^{-2}]
ICTH[N] ICT[N]	CT[N]	(T0[N])		
1	420	90.0	NR(heme) - FE - NR(heme)	0.128
2	405	96.0	H - S - CH2	0.122
3	475	100.0	CH2 - S - CH3	0.140
4	420	103.0	OA - P - OA	0.121
5	490	104.0	CH2 - S - S	0.140
6	465	108.0	NR, C, CR1(5-ring)	0.128
7	285	109.5	CHn - CHn - CHn, NR(6-ring) (sugar)	0.0769
8	320	109.5	CHn, OA - CHn - OA, NR(ring) (sugar)	0.0864
9	380	109.5	H - NL, NT - H, CHn - OA - CHn(sugar)	0.103
10	425	109.5	H - NL - C, CHn H - NT - CHn	0.115
11	450	109.5	X - OA, SI - X	0.122
12	520	109.5	CHn,C - CHn - C, CHn, OA, OM, N, NE	0.141
13	450	109.6	OM - P - OA	0.121
14	530	111.0	CHn - CHn - C, CHn, OA, NR, NT, NL	0.140
15	545	113.0	CHn - CH2 - S	0.140
16	50	115.0	NR(heme) - FE - NR	0.0123
17	460	115.0	H - N - CHn	0.115
18	610	115.0	CHn, C - C - OA, N, NT, NL	0.152
19	465	116.0	H - NE - CH2	0.114
20	620	116.0	CH2 - N - CH1	0.152
21	635	117.0	CH3 - N - C, CHn - C - OM	0.153
22	390	120.0	H - NT, NZ, NE - C	0.0889
23	445	120.0	H - NT, NZ - H	0.101
24	505	120.0	H - N - CH3, H, HC - 6-ring, H - NT - CHn	0.115
25	530	120.0	P, SI - OA - CHn, P	0.121
26	560	120.0	N, C, CR1 (6-ring, no H)	0.128
27	670	120.0	NZ - C - NZ, NE	0.153
28	780	120.0	OM - P - OM	0.178
29	685	121.0	O - C - CHn, C; CH3 - N - CHn	0.153
30	700	122.0	CH1, CH2 - N - C	0.153
31	415	123.0	H - N - C	0.0887
32	730	124.0	O - C - OA, N, NT, NL C - NE - CH2	0.153
33	375	125.0	FE - NR - CR1 (5-ring)	0.0765
34	750	125.0	-	0.153

TABLE 3.3: continues on next page.

Bond-angle type code	Force constant	Ideal bond angle	Example of usage in terms of non-bonded atom types	
$M_n^{(\theta)}$	$k_n^{(\theta,c)}$	θ_n^0		$k_n^{(\theta,h)}$
	[kJ·mol^{-1}]	[deg]		[kJ·mol^{-1}·deg^{-2}]
ICTH[N] ICT[N]	CT[N]	(T0[N])		
35	575	126.0	H, HC - 5-ring	0.114
36	640	126.0	X(noH) - 5-ring	0.127
37	770	126.0	OM - C - OM	0.153
38	760	132.0	5, 6 ring connnection	0.128
39	2215	155.0	SI - OA - SI	0.121
40	434	109.5	HWat - OWat - HWat	0.117
41	484	107.57	HChl - CChl - CLChl	0.134
42	632	111.30	CLChl - CChl - CLChl	0.167
43	469	97.4	CDmso - SDmso - CDmso	0.140
44	503	106.75	CDmso - SDmso - ODmso	0.140
45	443	108.53	HMet - OMet - CMet	0.121
46	618	109.5	CLCl4 - CCl4 - CLCl4	0.167
47	380	90.0	NR (heme) - FE - C (CO bound to heme)	0.116
48	91350	180.0	Fe - C - O (CO bound to heme)	0.0726

TABLE 3.3: GROMOS 45A4 and 45B4 bond-angle bending parameters ($k_n^{(\theta,c)} = g(k_n^{(\theta,h)}, \theta_n^0, E_{k_BT})$.

Improper dihedral-angle type code	Force constant	Ideal improper dihedral angle	Example of usage	
	$k_n^{(\xi)}$	ξ_n^0		$k_n^{(\xi)}$
	[kJmol^{-1}degree^{-2}]	[degree]		[kcalmol^{-1}rad^{-2}]
ICQH[N] ICQ[N]	CQ[N]	(Q0[N])		
1	0.0510	0.0	planar groups	40
2	0.102	35.26439	tetrahedral centres	80
3	0.204	0.0	heme iron	160

TABLE 3.4. GROMOS 45A4 and 45B4 improper (harmonic) dihedral angle parameters.

Dihedral-angle type code	Force constant	Phase shift	Multiplicity	Example of usage in terms of non-bonded atom types	
	$k_n^{(\varphi,s)}$	$\cos(\varphi_n^0)$	$m_n^{(\varphi)}$		$k_n^{(\varphi,s)}$
	[kJmol^{-1}]				[kcalmol^{-1}]
ICPH[N] ICP[N]	CP[N]	PD[N]	NP[N]		
1	5.86	-1.0	2	-C-C-	1.4
2	7.11	-1.0	2	-C-OA- (at ring)	1.7
3	16.7	-1.0	2	-C-OA- (carboxyl)	4.0
4	33.5	-1.0	2	-C-N, NT, NE, NZ,NR-	8.0
5	41.8	-1.0	2	-C-CR1- (6-ring)	10.0
6	0.0	+1.0	2	-CH1 (sugar)-NR(base)-	0.0
7	0.418	+1.0	2	O-CH1-CHn-no O	0.1
8	2.09	+1.0	2	O-CH1-CHn-O	0.5
9	3.14	+1.0	2	-OA-P-	0.75
10	16.7	+1.0	2	-S-S-	4.0
11	1.05	+1.0	3	-OA-P-; -P - O5* - C5* - C4* (dna)	0.25
12	1.26	+1.0	3	-CHn-OA(no sugar)-	0.3
13	2.93	+1.0	3	-CH2-S-	0.7
14	3.77	+1.0	3	-C,CHn,SI-NT,NL,OA(sugar)-	0.9
15	4.18	+1.0	3	HC-C-S-	1.0
16	5.44	+1.0	3	HC-C-C-	1.3
17	5.86	+1.0	3	-CHn,SI-CHn-	1.4
18	0.0	+1.0	4	-NR-FE-	0.0
19	1.0	-1.0	6	-CHn-N,NE-	0.24
20	1.0	+1.0	6	-CHn-C,NR (ring), CR1-	0.24
21	3.77	+1.0	6	-CHn-NT-	0.9
22	5.35	+1.0	1	O5* - C5* - C4* - O4* (dna)	1.3
23	2.53	+1.0	3	O5* - C5* - C4* - O4* (dna)	0.60
24	5.09	+1.0	2	CHn - O - P - O (dna, phosphodiester)	1.2
25	3.19	+1.0	3	CHn - O - P - O (dna, phosphodiester)	0.76
26	2.79	+1.0	1	P - O5* - C5* - C4* (dna)	0.67
27	5.86	-1.0	1	N - CHn - CHn - OA (lipid)	1.4
28	8.62	+1.0	3	N - CHn - CHn - OA (lipid)	2.1
29	24.0	-1.0	2	CHn - OA - C - CHn (ester lipid)	5.7
30	3.90	+1.0	3	CHn - CHn - OA - H (sugar)	0.93
31	9.35	-1.0	1	OA - CHn - CHn - OA (sugar) O5 - C5 - C6 - O6^b	2.2
32	9.50	+1.0	3	OA - CHn - CHn - OA (sugar) O5 - C5 - C6 - O6^b	2.3
33	9.45	-1.0	1	OA - CHn - OA - CHn,H (αsugar) O5 - C1 - O1 - C1′,H1	2.3
34	3.41	-1.0	1	OA - CHn - OA - CHn,H (βsugar) O5 - C1 - O1 - C1',H1	0.81
35	4.69	+1.0	3	OA - CHn - OA - CHn,H (βsugar) O5 - C1 - O1 - C1',H1	1.1
36	3.65	+1.0	3	OA - CHn - OA - CHn,H (αsugar) O5 - C1 - O1 - C1',H1	0.87

TABLE 3.5: continues on next page.

Dihedral-angle type code	Force constant	Phase shift	Multiplicity	Example of usage in terms of non-bonded atom types	
	$k_n^{(\varphi,s)}$	$\cos(\varphi_n^0)$	$m_n^{(\varphi)}$		$k_n^{(\varphi,s)}$
	[kJmol^{-1}]				[kcalmol^{-1}]
ICPH[N] ICP[N]	CP[N]	PD[N]	NP[N]		
37	6.66	-1.0	1	OA - CHn - CHn - OA (sugar) O5 - C5 - C6 - O6[a]	1.2
38	7.69	+1.0	3	OA - CHn - CHn - OA (sugar) O5 - C5 - C6 - O6[a]	1.8
39	2.67	-1.0	1	CHn - CHn - CHn - OA (sugar) O5 - C5 - C6 - O6[a]	0.64
40	1.53	-1.0	2	C1 - C2 - CAB - CBB (heme)	0.37

TABLE 3.5: GROMOS 45A4 and 45B4 (trigonometric) dihedral torsional angle parameters. a) To be used if - C5 - C6 - O6 and adjacent - C4 - O4 - are axial and the other equatorial, as in galactose; b) To be used if - C5 - C6 - O6 and adjacent - Cn - On - Hn are both simultaneously axial or equatorial, as in glucose.

integer atom code	atom type	description
IAC[N]	TYPE[N]	
1	O	carbonyl oxygen (C=O)
2	OM	carboxyl oxygen (CO^-)
3	OA	hydroxyl, sugar or ester oxygen
4	OW	water oxygen
5	N	peptide nitrogen (NH)
6	NT	terminal nitrogen (NH2)
7	NL	terminal nitrogen (NH3)
8	NR	aromatic nitrogen
9	NZ	Arg NH (NH2)
10	NE	Arg NE (NH)
11	C	bare carbon
12	CH1	aliphatic or sugar CH-group
13	CH2	aliphatic or sugar CH2-group
14	CH3	aliphatic CH3-group
15	CH4	methane
16	CR1	aromatic CH-group
17	HC	hydrogen bound to carbon
18	H	hydrogen not bound to carbon
19	DUM	dummy atom
20	S	sulphur
21	CU1+	copper (charge 1+)
22	CU2+	copper (charge 2+)
23	FE	iron (heme)
24	ZN2+	zinc (charge 2+)
25	MG2+	magnesium (charge 2+)
26	CA2+	calcium (charge 2+)
27	P, SI	phosphor or silicon
28	AR	argon
29	F	fluor (non-ionic)
30	CL	chlorine (non-ionic)
31	BR	bromine (non-ionic)
32	CMet	CH3-group in methanol (solvent)
33	OMet	oxygen in methanol (solvent)
34	NA+	sodium (charge 1+)
35	CL-	chloride (charge 1-)
36	CChl	carbon in chloroform (solvent)
37	CLChl	chloride in chloroform (solvent)
38	HChl	hydrogen in chloroform (solvent)
39	SDmso	sulphur in DMSO (solvent)
40	CDmso	CH3-group in DMSO (solvent)
41	ODmso	oxygen in DMSO (solvent)
42	CCl4	carbon in carbontetrachloride (solvent)
43	CLCl4	chloride in carbontetrachloride (solvent)
44	CH2r	aliphatic or sugar CH2 group in ring

TABLE 3.6: continues on next page.

integer atom code	atom type	description
IAC[N]	TYPE[N]	
45	CH0	bare sp3 carbon, 4 bound heavy atoms

TABLE 3.6: GROMOS 45A4 and 45B4 non-bonded atom types and integer atom codes.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
1	O	0.04756	0.8611	1.125	-
2	OM	0.04756	0.8611	1.841	3.068
3	OA	0.04756	1.125	1.227	-
4	OW	0.05116	1.544	1.623	-
5	N	0.04936	1.301	1.943	-
6	NT	0.04936	1.301	2.250	-
7	NL	0.04936	1.301	3.068	-
8	NR	0.04936	1.301	1.841	-
9	NZ	0.04936	1.301	2.148	-
10	NE	0.04936	1.301	1.984	-
11	C	0.04838	1.837	-	-
12	CH1	0.07790	9.850	-	-
13	CH2	0.08642	5.828	-	-
14	CH3	0.09805	5.162	-	-
15	CH4	0.1148	5.862	-	-
16	CR1	0.07425	3.888	-	-
17	HC	0.0092	0.123	-	-
18	H	0.0	0.0	-	-
19	DUM	0.0	0.0	-	-
20	S	0.09992	3.616	-	-
21	CU1+	0.02045	0.07159	0.2250	-
22	CU2+	0.02045	0.07159	0.4091	-
23	FE	0.0	0.0	0.0	-
24	ZN2+	0.02045	0.09716	0.09716	-
25	MG2+	0.008080	0.05838	0.05838	-
26	CA2+	0.03170	0.7057	0.7057	-
27	P, SI	0.1214	4.711	-	-
28	AR	0.07915	3.138	-	-
29	F	0.03432	0.8722	1.227	-
30	CL	0.09362	3.911	-	-
31	BR	0.03434	8.092	-	-
32	CMet	0.09421	4.5665	-	-
33	OMet	0.04756	1.125	1.227	-
34	NA+	0.008489	0.1450	0.1450	-
35	CL-	0.1175	10.340	10.340	10.340
36	CChl	0.051292	2.0160	-	-
37	CLChl	0.091141	3.7101	-	-
38	HChl	0.0061400	0.065574	-	-
39	SDmso	0.10277	4.6366	-	-
40	CDmso	0.095139	4.6645	-	-
41	ODmso	0.047652	0.86686	1.125	-
42	CCl4	0.051292	2.7568		

TABLE 3.7: continues on next page.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
43	CLCl4	0.087201	3.5732		
44	CH2r	0.08564	5.297	-	-
45	CH0	0.04896	14.33	-	-

TABLE 3.7: GROMOS 45A4 normal van der Waals parameters.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
2	OM	0.04756	0.8611	1.125	1.125
7	NL	0.04936	1.301	1.943	-

TABLE 3.8. GROMOS 45B4 (vacuo) normal van der Waals parameters.

integer atom code	atom type	integer atom code	atom type	C_6 (I,J)	C_{12} (I,J)
I		J		$10^{-3}kJmol^{-1}nm^6$	$10^{-6}kJmol^{-1}nm^{12}$
36	CChl	37	CLChl	4.6754	7.4813
36	CChl	38	HChl	0.3622	0.1745
37	CLChl	38	HChl	0.6493	0.3266
39	SDmso	40	CDmso	9.7827	21.6523
39	SDmso	41	ODmso	5.2442	4.6094
40	CDmso	41	ODmso	4.9187	4.7597

TABLE 3.9. GROMOS 45A4 normal van der Waals parameters for mixed atom type pairs (I,J).

	J	1	2	3	4	5	6	7	8	9	10	21	22	23	24	25	26	27	29	30	31	33	34	35	41
I		O	OM	OA	OW	N	NT	NL	NR	NZ	NE	CU1+	CU2+	FE	ZN2+	MG2+	CA2+	P,SI	F	CL	BR	OMet	NA+	CL-	ODmso
1	O	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1
2	OM	1	1	2	2	2	2	3	2	3	3	3	3	3	3	3	3	3	1	1	1	2	3	1	1
3	OA	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
4	OW	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
5	N	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
6	NT	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	2	2	2	2	1	2	2
7	NL	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
8	NR	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
9	NZ	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
10	NE	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
21	CU1+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
22	CU2+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
23	FE	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
24	ZN2+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
25	MG2+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
26	CA2+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
27	P,SI	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	F	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1
30	CL	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
31	BR	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
33	OMet	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
34	NA+	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2
35	CL-	1	1	2	2	2	2	3	2	3	3	3	3	3	3	3	3	3	1	1	1	2	3	1	1
41	ODmso	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1

TABLE 3.10. Selection of van der Waals (repulsive) $C_{12}^{1/2}(I,I)$ parameters (GROMOS 45A4 and 45B4).

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
12	CH1	0.05396	1.933	-	-
13	CH2	0.06873	2.178	-	-
14	CH3	0.08278	2.456	-	-
16	CR1	0.07435	2.886	-	-
44	CH2r	0.06873	2.178	-	-
45	CH0	0.04838	1.837	-	-

TABLE 3.11. GROMOS 45A4 and 45B4 third-neighbour van der Waals parameters.

atom name	charge in e	occurring in
N	-0.280	all residues
H	0.280	
C	0.380	all residues
O	-0.380	
CD	0.090 (0.0)	Arg (charge +1)
NE	-0.110 (-0.240)	
HE	0.240 (0.240)	
CZ	0.340 (0.0)	
NH1/2	-0.260 (-0.480)	
HH11/12/21/22	0.240 (0.240)	
NE	-0.280	Argn (neutral)
HE	0.280	
CZ	0.150	Argn (neutral)
NH1	-0.548	
HH1	0.398	
NH2	-0.830	Argn (neutral)
HH21/22	0.415	
CG, CD	0.380	Asn, Gln
OD1, OE1	-0.380	
ND2, NE2, NZ	-0.830	Asn, Gln, Lys
HD21/22, HE21/22, HZ1/2	0.415	
CG, CD	0.270 (0.720)	Asp, Glu (charge -1)
OD1/2, OE1/2	-0.635 (-0.360)	
CG, CD	0.530	Asph, Gluh
OD1, OE1	-0.380	
OD2, OE2	-0.548	
HD2, HE2	0.398	
CB	-0.100 (0.200)	Cys (charge -.5)
SG	-0.400 (-0.200)	
SG	-0.064	Cysh
HG	0.064	
CG	0.0	Hisa (proton at D1)
ND1	0.0	
HD1	0.190	
CD2	0.130	
CE1	0.260	
NE2	-0.580	
CG	0.130	Hisb (proton at E2)
ND1	-0.580	
CD2	0.0	
CE1	0.260	
NE2	0.0	
HE2	0.190	
CG	-0.050 (0.0)	Hish (charge +1)
ND1	0.380 (-0.300)	

TABLE 3.12: continues on next page.

atom name	charge in e	occurring in
HD1	0.300 (0.300)	
CD2	0.0 (0.0)	
CE1	-0.240 (0.0)	
NE2	0.310 (-0.300)	
HE2	0.300 (0.300)	
CG, CB, CB, CZ	0.150	Hypr, Ser, Thr, Tyr
OD1, OG, OG1, OH	-0.548	
HD1, HG, HG1, HH	0.398	
CE	0.127 (0.0)	Lysh (charge +1)
NZ	0.129 (-0.744)	
HZ1/2/3	0.248 (0.248)	
CG	-0.140	Trp
CD1	-0.100	
HD1	0.100	
CD2	0.0	
NE1	-0.050	
HE1	0.190	
CE2	0.0	
C	-0.100	all aromatic C-H groups in Phe, Tyr, Trp
H	0.100	

TABLE 3.12: GROMOS 45A4 (45B4) atomic charges and charge group definitions for amino acid residues. The charges for the 45B4 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
OW	-0.8200	H_2O (SPC model)
HW1/2	0.4100	
OW	-0.8476	H_2O (SPC/E model)
HW1/2	0.4238	
CChl	0.1790	Chloroform
CLChl	-0.0870	
HChl	0.0820	
SDmso	0.1390	DMSO
CDmso	0.1600	
ODmso	-0.4590	
CMet	0.1760	Methanol
OMet	-0.5740	
HMet	0.3980	
CCl4	0.0	Carbontetrachloride
CLCl4	0.0	

TABLE 3.13. GROMOS 45A4 atomic charges for various solvents.

atom name	charge in e	occurring in
O3/5*	-0.360 (-0.360)	all nucleotides
P	0.990 (1.440)	
O1/2P	-0.635 (-0.360)	
C4*	0.160	all nucleotides
O4*	-0.360	
C1*	0.200	
N9, N9, N1, N1, N1	-0.200	dAde, dGua, dCyt, dThy, Ura
C4, C4	0.200	dAde, dGua
C6, C6, C6	0.100	dCyt, dThy, Ura
H6, H6, H6	0.100	dCyt, dThy, Ura
N1, N3, N7, N3, N7, N3	-0.540	dAde, dAde, dAde, dGua, dGua, dGua, dCyt
C2, C8, C8	0.440	dAde, dAde, dGua
H2, H8, H8	0.100	dAde, dAde, dGua
C6, C2, C4	0.540	dAde, dGua, dCyt
N6, N2, N4	-0.830	dAde, dGua, dCyt
H61/62, H21/22, H41/42	0.415	dAde, dGua, dCyt
N1, N3, N3	-0.310	dGua, dThy, Ura
H1, H3, H3	0.310	dGua, dThy, Ura
C6, C2, C2, C4, C2, C4	0.450	dGua, dCyt, dThy, dThy, Ura, Ura
O6, O2, O2, O4, O2, O4	-0.450	dGua, dCyt, dThy, dThy, Ura, Ura
C5, C5	-0.100	dCyt, Ura
H5, H5	0.100	dCyt, Ura

TABLE 3.14. GROMOS 45A4 (45B4) atomic charges and charge group definitions for nucleotides. The charges for the 45B4 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
C32, C33, C34, C35	0.250 (0.0)	dppc
N	0.0	dppc
C31	0.0	dppc
O31, O32	-0.360 (-0.360)	dppc
O33, O34	-0.635 (-0.360)	dppc
P	0.990 (1.440)	dppc
C3	0.0	dppc
C1, C2	0.160	dppc
O11, O21	-0.360	dppc
O12, O22	-0.380	dppc
C11, C21	0.580	dppc
C12, ..., C22, ...	0.0	dppc

TABLE 3.15. GROMOS 45A4 (45B4) atomic charges and charge group definitions for lipids. The charges for the 45B4 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
C1, C2, C3, C4, C6	0.232	hexopyranose, uronate
O2, O3, O4, O6	-0.642	hexopyranose, uronate
HO2, HO3, HO4, HO6	0.410	hexopyranose, uronate
C5	0.376	hexopyranose, uronate
O5	-0.480	hexopyranose, uronate
O1	-0.360	hexopyranose, uronate
C6	0.360 (0.720)	uronate
O61, O62	-0.680 (-0.360)	uronate
C1	0.232	terminal C1 - O1 - HO1 group
O1	-0.538	terminal C1 - O1 - HO1 group
HO1	0.410	terminal C1 - O1 - HO1 group
C1	0.232	terminal C1 - O1 - CM group
O1	-0.360	terminal C1 - O1 - CM group
CM (methyl)	0.232	terminal C1 - O1 - CM group
C5, C5'	0.378	terminal C1 - O1 - C1'(sugar) group
O5, O5'	-0.450	terminal C1 - O1 - C1'(sugar) group
C1, C1'	0.242	terminal C1 - O1 - C1'(sugar) group
O1	-0.340	terminal C1 - O1 - C1'(sugar) group

TABLE 3.16. GROMOS 45A4 (45B4) atomic charges and charge group definitions for carbohydrates.

Bond-type code	Force constant	Ideal bond length	Examples of usage in terms of non-bonded atom types	
$M_n^{(b)}$	$k_n^{(b,q)}$	b_n^0		$k_n^{(b,h)}$
	[10^6 kJ·mol^{-1}·nm^{-4}]	[nm]		[10^6 kJ·mol^{-1}·nm^{-2}]
ICBH[N] ICB[N]	CB[N]	B0[N]		
1	15.7	0.100	H - OA	0.314
2	18.7	0.100	H - N (all)	0.374
3	12.3	0.109	HC - C	0.292
4	37.0	0.112	C - O (CO bound to heme)	0.928
5	16.6	0.123	C - O	0.502
6	13.4	0.125	C - OM	0.419
7	12.0	0.132	CR1 - NR (6-ring)	0.418
8	8.87	0.133	H - S	0.314
9	10.6	0.133	C - NT, NL	0.375
10	11.8	0.133	C, CR1 - N, NR, CR1, C (peptide, 5-ring)	0.417
11	10.5	0.134	C - N, NZ, NE	0.377
12	11.7	0.134	C - NR (no H) (6-ring)	0.420
13	10.2	0.136	C - OA, FTfe - CTfe	0.377
14	11.0	0.138	C - NR (heme)	0.419
15	8.66	0.139	CH2 - C, CR1 (6-ring)	0.335
16	10.8	0.139	C, CR1 - CH2, C, CR1 (6-ring)	0.417
17	8.54	0.140	C, CR1, CH2 - NR (6-ring)	0.335
18	8.18	0.143	CHn - OA	0.335
19	9.21	0.143	CHn - OM	0.377
20	6.10	0.1435	CHn - OA (sugar)	0.251
21	8.71	0.147	CHn - N, NT, NL, NZ, NE	0.376
22	5.73	0.148	CHn - NR (5-ring)	0.251
23	7.64	0.148	CHn - NR (6-ring)	0.335
24	8.60	0.148	O, OM - P	0.377
25	8.37	0.150	O - S	0.377
26	5.43	0.152	CHn - CHn (sugar)	0.251
27	7.15	0.153	C, CHn - C, CHn	0.335
28	4.84	0.161	OA - P	0.251
29	4.72	0.163	OA - SI	0.251
30	2.72	0.178	FE - C (CO bound to heme)	0.172
31	5.94	0.178	CH3 - S	0.376
32	5.62	0.183	CH2 - S	0.376
33	3.59	0.187	CH1 - SI	0.251
34	0.640	0.198	NR (His) - FE (43A1)	0.0502
35	0.628	0.200	NR (heme) - FE	0.0502
36	5.03	0.204	S - S	0.419
37	0.540	0.221	NR (His) - FE	0.0527
38	23.2	0.100	HWat - OWat	0.464
39	12.1	0.110	HChl - CChl	0.293
40	8.12	0.1758	CChl - CLChl	0.502

TABLE 3.17: continues on next page.

Bond-type code	Force constant	Ideal bond length	Examples of usage in terms of non-bonded atom types	
$M_n^{(b)}$	$k_n^{(b,q)}$	b_n^0		$k_n^{(b,h)}$
	[10^6 kJ·mol^{-1}·nm^{-4}]	[nm]		[10^6 kJ·mol^{-1}·nm^{-2}]
ICBH[N] ICB[N]	CB[N]	B0[N]		
41	8.04	0.153	ODmso - SDmso	0.376
42	4.95	0.193799	SDmso - CDmso	0.372
43	8.10	0.176	CCl4 - CLCl4	0.502
44	13.1	0.1265	CUrea - OUrea	0.419
45	10.3	0.135	CUrea - NUrea	0. 375
46	8.71	0.163299	HWat - HWat	0.465
47	2.68	0.233839	HChl - CLChl	0.293
48	2.98	0.290283	CLChl - CLChl	0.502
49	2.39	0.279388	ODmso - CDmso	0.373
50	2.19	0.291189	CDmso - CDmso	0.371
51	3.97	0.2077	HMet - CMet	0.343
52	3.04	0.287407	CLCl4 - CLCl4	0.502

TABLE 3.17: GROMOS 54A7 and 54B7 bond-stretching parameters ($k_n^{(b,q)} = k_n^{(b,h)}/(2{b_n^0}^2)$).

Bond-angle type code	Force constant	Ideal bond angle	Example of usage in terms of non-bonded atom types	
$M_n^{(\theta)}$	$k_n^{(\theta,c)}$	θ_n^0		$k_n^{(\theta,h)}$
	[kJ·mol^{-1}]	[deg]		[kJ·mol^{-1}·deg^{-2}]
ICTH[N] ICT[N]	CT[N]	(T0[N])		
1	380	90.0	NR (heme) - FE - C (CO bound to heme)	0.116
2	420	90.0	NR(heme) - FE - NR(heme), NR (His)	0.128
3	405	96.0	H - S - CH2	0.122
4	475	100.0	CH2 - S - CH3	0.140
5	420	103.0	OA - P - OA	0.121
6	490	104.0	CH2 - S - S	0.140
7	465	108.0	NR, C, CR1(5-ring)	0.128
8	285	109.5	CHn - CHn - CHn, NR(6-ring) (sugar)	0.0769
9	320	109.5	CHn, OA - CHn - OA, NR(ring) (sugar)	0.0864
10	380	109.5	H - NL, NT - H; CHn - OA - CHn(sugar)	0.103
11	425	109.5	H - NL - C, CHn; H - NT - CHn	0.115
12	450	109.5	X - OA, SI - X	0.122
13	520	109.5	CH,C - CHn - C, CHn, OA, OM, N, NE	0.141
14	450	109.6	OM - P - OA	0.121
15	530	111.0	CHn - CHn - C, CHn, OA, NR, NT, NL	0.140
16	545	113.0	CHn - CH2 - S	0.140
17	50.0	115.0	NR(heme) - FE - NR (His) (43A1)	0.0123
18	460	115.0	H - N - CHn	0.115
19	610	115.0	CHn, C - C - OA, N, NT, NL	0.152
20	465	116.0	H - NE - CH2	0.114
21	620	116.0	CH2 - N - CH1	0.152
22	635	117.0	CH3 - N - C; CHn - C - OM	0.153
23	390	120.0	H - NT, NZ, NE - C	0.0889
24	445	120.0	H - NT, NZ - H	0.101
25	505	120.0	H - N - CH3, H; HC - 6-ring,; H - NT - CHn	0.115
26	530	120.0	P, SI - OA - CHn, P	0.121
27	560	120.0	N, C, CR1 (6-ring, no H)	0.128
28	670	120.0	NZ - C - NZ, NE	0.153
29	780	120.0	OM - P - OM	0.178
30	685	121.0	O - C - CHn, C; CH3 - N - CHn	0.153
31	700	122.0	CH1, CH2 - N - C	0.153
32	415	123.0	H - N - C	0.0887

TABLE 3.18: continues on next page.

Bond-angle type code	Force constant	Ideal bond angle	Example of usage in terms of non-bonded atom types	
$M_n^{(\theta)}$	$k_n^{(\theta,c)}$	θ_n^0		$k_n^{(\theta,h)}$
	[kJ·mol^{-1}]	[deg]		[kJ·mol^{-1}·deg^{-2}]
ICTH[N] ICT[N]	CT[N]	(T0[N])		
33	730	124.0	O - C - OA, N, NT, NL C - NE - CH2	0.153
34	375	125.0	FE - NR - CR1 (5-ring)	0.0765
35	750	125.0	-	0.153
36	575	126.0	H, HC - 5-ring	0.114
37	640	126.0	X(noH) - 5-ring	0.127
38	770	126.0	OM - C - OM	0.153
39	760	132.0	5, 6 ring connnection	0.128
40	2215	155.0	SI - OA - SI	0.121
41	91350	180.0	Fe - C - O (CO bound to heme)	0.0726
42	434	109.5	HWat - OWat - HWat	0.117
43	484	107.57	HChl - CChl - CLChl	0.134
44	632	111.30	CLChl - CChl - CLChl	0.167
45	469	97.4	CDmso - SDmso - CDmso	0.140
46	503	106.75	CDmso - SDmso - ODmso	0.140
47	443	108.53	HMet - OMet - CMet	0.121
48	618	109.5	CLCl4 - CCl4 - CLCl4	0.167
49	507	107.6	FTfe - CTfe - FTfe	0.140
50	448	109.5	HTfe - OTfe - CHTfe	0.121
51	524	110.3	OTfe - CHTfe - CTfe	0.140
52	532	111.4	CHTfe - CTfe - FTfe	0.140
53	636	117.2	NUrea - CUrea - NUrea	0.153
54	690	121.4	OUrea - CUrea - NUrea	0.153

TABLE 3.18: GROMOS 54A7 and 54B7 bond-angle bending parameters ($k_n^{(\theta,c)} = g(k_n^{(\theta,h)}, \theta_n^0, E_{k_BT})$).

Improper dihedral-angle type code	Force constant	Ideal improper dihedral angle	Example of usage	
	$k_n^{(\xi)}$	ξ_n^0		$k_n^{(\xi)}$
	[kJmol^{-1}degree^{-2}]	[degree]		[kcalmol^{-1}rad^{-2}]
ICQH[N] ICQ[N]	CQ[N]	(Q0[N])		
1	0.0510	0.0	planar groups	40
2	0.102	35.26439	tetrahedral centres	80
3	0.204	0.0	heme iron	160
4	0.0510	180.0	planar groups	40
5	0.102	-35.26439	tetrahedral centres	80

TABLE 3.19. GROMOS 54A7 and 54B7 improper (harmonic) dihedral angle parameters.

Dihedral-angle type code	Force constant	Phase shift	Multiplicity	Example of usage in terms of non-bonded atom types	
	$k_n^{(\varphi,s)}$	$\cos(\varphi_n^0)$	$m_n^{(\varphi)}$		$k_n^{(\varphi,s)}$
	[kJmol^{-1}]				[kcalmol^{-1}]
ICPH[N] ICP[N]	CP[N]	PD[N]	NP[N]		
1	2.67	-1.0	1	CHn-CHn-CHn-OA (sugar) C4-C5-C6-O6[a]	0.6
2	3.41	-1.0	1	OA-CHn-OA-CHn,H (βsugar) O5-C1-O1-C1',H1	0.8
3	4.97	-1.0	1	OA-CHn-CHn-OA (sugar) O5-C5-C6-O6[a]	1.2
4	5.86	-1.0	1	N-CHn-CHn-OA(lipid)	1.4
5	9.35	-1.0	1	OA-CHn-CHn-OA(sugar) O5-C5-C6-O6[b]	2.2
6	9.45	-1.0	1	OA-CHn-OA-CHn,H (α sugar) O5-C1-O1-C1$'$,H1	2.3
7	2.79	+1.0	1	P-O5*-C5*-C4* (dna)	0.7
8	5.35	+1.0	1	O5*-C5*-C4*-O4* (dna)	1.3
9	1.53	-1.0	2	C1-C2-CAB-CBB (heme)	0.4
10	5.86	-1.0	2	-C-C-	1.4
11	7.11	-1.0	2	-C-OA- (at ring)	1.7
12	16.7	-1.0	2	-C-OA- (carboxyl)	4.0
13	24.0	-1.0	2	CHn-OA-C-CHn (ester lipid)	5.7
14	33.5	-1.0	2	-C-N,NT,NE,NZ,NR-	8.0
15	41.8	-1.0	2	-C-CR1- (6-ring)	10.0
16	0.0	+1.0	2	-CH1(sugar)-NR(base)-	0.0
17	0.418	+1.0	2	O-CH1-CHn-no O	0.1
18	2.09	+1.0	2	O-CH1-CHn-O	0.5
19	3.14	+1.0	2	-OA-P-	0.75
20	5.09	+1.0	2	CHn-O-P-O (dna, phosphodiester)	1.2
21	16.7	+1.0	2	-S-S-	4.0
22	1.05	+1.0	3	-OA-P-	0.25
23	1.26	+1.0	3	-CHn-OA(no sugar)-	0.3
24	1.30	+1.0	3	HTfe-OTfe-CHTfe-CTfe	0.3
25	2.53	+1.0	3	O5*-C5*-C4*-O4* (dna)	0.6
26	2.93	+1.0	3	-CH2-S-	0.7
27	3.19	+1.0	3	CHn-O-P-O (dna, phosphodiester)	0.8
28	3.65	+1.0	3	OA-CHn-OA-CHn,H (α sugar) O5 - C1 - O1 - C1',H1	0.9
29	3.77	+1.0	3	-C,CHn,SI-NT,NL,OA(sugar)-	0.9
30	3.90	+1.0	3	CHn-CHn-OA-H (sugar)	0.9
31	4.18	+1.0	3	HC-C-S-	1.0
32	4.69	+1.0	3	OA-CHn-OA-CHn,H (β sugar) O5 - C1 - O1 - C1',H1	1.1
33	5.44	+1.0	3	HC-C-C-	1.3
34	5.92	+1.0	3	-CHn,SI-CHn-	1.4
35	7.69	+1.0	3	OA-CHn-CHn-OA (sugar) O5 - C5 - C6 - O6[a]	1.8
36	8.62	+1.0	3	N-CHn-CHn-OA (lipid)	2.1
37	9.50	+1.0	3	OA-CHn-CHn-OA (sugar) O5-C5-C6-O6[b]	2.3
38	0.0	+1.0	4	-NR-FE-	0.0
39	1.0	-1.0	6	-CHn-N,NE-	0.24
40	1.0	+1.0	6	-CHn-C,NR(ring),CR1-	0.24

TABLE 3.20: continues on next page.

Dihedral-angle type code	Force constant	Phase shift	Multiplicity	Example of usage in terms of non-bonded atom types	
	$k_n^{(\varphi,s)}$	$\cos(\varphi_n^0)$	$m_n^{(\varphi)}$		$k_n^{(\varphi,s)}$
	[kJmol^{-1}]				[kcalmol^{-1}]
ICPH[N] ICP[N]	CP[N]	PD[N]	NP[N]		
41	3.77	+1.0	6	-CHn-NT-	0.9
42	3.50	-1.0	2	-CHn-C-	0.84
43	2.80	+1.0	3	-CHn-N-	0.64
44	0.70	-1.0	6	-CHn-N-	0.17
45	0.49	+1.0	6	-CHn-C-	0.10

TABLE 3.20: GROMOS 54A7 and 54B7 (trigonometric) dihedral torsional angle parameters. a) To be used if – C5 – C6 – O6 and adjacent – C4 – O4 – are axial and the other equatorial, as in galactose; b) To be used if – C5 – C6 – O6 and adjacent – Cn – On – Hn are both simultaneously axial or equatorial, as in glucose.

integer atom code	atom type	description
IAC[N]	TYPE[N]	
1	O	carbonyl oxygen (C=O)
2	OM	carboxyl oxygen (CO^-)
3	OA	hydroxyl or sugar oxygen
4	OE	ether or ester oxygen
5	OW	water oxygen
6	N	peptide nitrogen (NH)
7	NT	terminal nitrogen (NH2)
8	NL	terminal nitrogen (NH3)
9	NR	aromatic nitrogen
10	NZ	Arg NH (NH2)
11	NE	Arg NE (NH)
12	C	bare carbon
13	CH0	bare sp3 carbon, 4 bound heavy atoms
14	CH1	aliphatic or sugar CH-group
15	CH2	aliphatic or sugar CH2-group
16	CH3	aliphatic CH3-group
17	CH4	methane
18	CH2r	aliphatic or sugar CH2 group in ring
19	CR1	aromatic CH-group
20	HC	hydrogen bound to carbon
21	H	hydrogen not bound to carbon
22	DUM	dummy atom
23	S	sulphur
24	CU1+	copper (charge 1+)
25	CU2+	copper (charge 2+)
26	FE	iron (heme)
27	ZN2+	zinc (charge 2+)
28	MG2+	magnesium (charge 2+)
29	CA2+	calcium (charge 2+)
30	P, SI	phosphor or silicon
31	AR	argon
32	F	fluor (non-ionic)
33	CL	chlorine (non-ionic)
34	BR	bromine (non-ionic)
35	CMet	CH3-group in methanol (solvent)
36	OMet	oxygen in methanol (solvent)
37	NA+	sodium (charge 1+)
38	CL-	chloride (charge 1-)
39	CChl	carbon in chloroform (solvent)
40	CLChl	chloride in chloroform (solvent)
41	HChl	hydrogen in chloroform (solvent)
42	SDmso	sulphur in DMSO (solvent)
43	CDmso	CH3-group in DMSO (solvent)
44	ODmso	oxygen in DMSO (solvent)

TABLE 3.21: continues on next page.

integer atom code	atom type	description
IAC[N]	TYPE[N]	
45	CCl4	carbon in carbontetrachloride (solvent)
46	CLCl4	chloride in carbontetrachloride (solvent)
47	FTfe	fluor in trifluorethanol
48	CTfe	carbon in trifluorethanol
49	CHTfe	CH2-group in trifluorethanol
50	OTfe	oxygen in trifluorethanol
51	CUrea	carbon in urea
52	OUrea	oxygen in urea
53	NUrea	nitrogen in urea
54	CH3p	positively charged methyl

TABLE 3.21: GROMOS 54A7 and 54B7 non-bonded atom types and integer atom codes.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
1	O	0.04756	1.000	1.130	-
2	OM	0.04756	0.8611	1.841	3.068
3	OA	0.04756	1.100	1.227	-
4	OE	0.04756	1.100	1.227	-
5	OW	0.05116	1.623	1.623	-
6	N	0.04936	1.523	1.943	-
7	NT	0.04936	1.523	2.250	-
8	NL	0.04936	1.523	3.068	-
9	NR	0.04936	1.523	1.841	-
10	NZ	0.04936	1.523	2.148	-
11	NE	0.04936	1.523	1.984	-
12	C	0.04838	2.222	-	-
13	CH0	0.04896	14.33	-	-
14	CH1	0.07790	9.850	-	-
15	CH2	0.08642	5.828	-	-
16	CH3	0.09805	5.162	-	-
17	CH4	0.1148	5.862	-	-
18	CH2r	0.08564	5.297	-	-
19	CR1	0.07425	3.888	-	-
20	HC	0.0092	0.123	-	-
21	H	0.0	0.0	-	-
22	DUM	0.0	0.0	-	-
23	S	0.09992	3.616	-	-
24	CU1+	0.02045	0.07159	0.2250	-
25	CU2+	0.02045	0.07159	0.4091	-
26	FE	0.0	0.0	0.0	-
27	ZN2+	0.02045	0.09716	0.09716	-
28	MG2+	0.008080	0.05838	0.05838	-
29	CA2+	0.03170	0.7057	0.7057	-
30	P, SI	0.1214	4.711	4.711	-
31	AR	0.07915	3.138	-	-
32	F	0.03432	0.8722	1.227	-
33	CL	0.09362	3.911	-	-
34	BR	0.1663	8.092	-	-
35	CMet	0.09421	4.400	-	-
36	OMet	0.04756	1.525	1.525	-
37	NA+	0.0088792	0.2700	0.2700	-
38	CL-	0.11318	7.776	7.776	7.776
39	CChl	0.051292	2.0160	-	-
40	CLChl	0.091141	3.7101	-	-
41	HChl	0.006140	0.065574	-	-
42	SDmso	0.10277	4.6366	-	-

TABLE 3.22: continues on next page.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
43	CDmso	0.098050	5.1620	-	-
44	ODmso	0.047652	0.86686	1.1250	-
45	CCl4	0.051292	2.7568	-	-
46	CLCl4	0.087201	3.5732	-	-
47	FTfe	0.034320	1.0000	1.0000	-
48	CTfe	0.048380	1.8370	-	-
49	CHTfe	0.084290	5.0770	-	-
50	OTfe	0.047560	1.2270	1.2270	-
51	CUrea	0.069906	3.6864	-	-
52	OUrea	0.048620	1.2609	1.2609	-
53	NUrea	0.057903	1.9877	1.9877	-
54	CH3p	0.09805	5.162	-	-

TABLE 3.22: GROMOS 54A7 normal van der Waals parameters.

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
2	OM	0.04756	0.8611	1.125	1.125
8	NL	0.04936	1.523	1.943	-

TABLE 3.23. GROMOS 54B7 (vacuo) normal van der Waals parameters. Only the changes relative to Tab. 3.22 are listed.

	J	1	2	3	4	5	6	7	8	9	10	11	24	25	26	27	28	29	30	32	33	34	36	37	38	44	47	50	52	53	54
I		O	OM	OA	OE	OW	N	NT	NL	NR	NZ	NE	CU1+	CU2+	FE	ZN2+	MG2+	CA2+	P,SI	F	CL	BR	OMet	NA+	CL-	ODmso	FTfe	OTfe	OUrea	NUrea	CH3p
1	O	1	1	2	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	1	1	2	1
2	OM	1	1	2	1	2	2	2	3	2	3	3	3	3	3	3	3	3	3	1	1	1	2	3	1	1	1	1	1	2	3
3	OA	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
4	OE	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	1	1	2	1
5	OW	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
6	N	2	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	2	1
7	NT	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	2	2	2	1
8	NL	2	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	2	1
9	NR	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
10	NZ	2	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	2	1
11	NE	2	2	2	2	2	1	2	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	2	1
24	CU1+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
25	CU2+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
26	FE	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
27	ZN2+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
28	MG2+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
29	CA2+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
30	P,SI	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	F	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	1	1	2	1
33	CL	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
34	BR	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
36	OMet	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
37	NA+	2	2	2	2	2	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	1	2	1	1
38	CL-	1	1	2	1	2	2	2	3	2	3	3	3	3	3	3	3	3	3	1	1	1	2	3	1	1	1	1	1	2	1
44	ODmso	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	2	1	2	1
47	FTfe	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	1	1	2	1
50	OTfe	1	1	2	1	2	1	2	1	2	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	2	1	2	2	2	1
52	OUrea	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	2	2	1	1	1	2	1	2	1
53	NUrea	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	2	2	2	2	1	2	2	2	2	2	2	1
54	CH3p	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

TABLE 3.24. Selection of van der Waals (repulsive) $C_{12}^{1/2}(I,I)$ parameters (GROMOS 54A7 and 54B7)

integer atom code	atom type	integer atom code	atom type	C_6 (I,J)	C_{12} (I,J)
I		J		10^{-3}kJmol^{-1}nm^6	10^{-6}kJmol^{-1}nm^{12}
39	CChl	40	CLChl	4.6754	7.4813
39	CChl	41	HChl	0.3622	0.1745
40	CLChl	41	HChl	0.6493	0.3266

TABLE 3.25. GROMOS 54A7 normal van der Waals parameters for mixed atom type pairs (I,J)

integer atom code	atom type	$C_6^{1/2}$(I,I)		$C_{12}^{1/2}$(I,I)	
		$[kJmol^{-1}\ nm^6]^{1/2}$		$10^{-3}[kJmol^{-1}\ nm^{12}]^{1/2}$	
I=IAC[N]	TYPE[N]		**1**	**2**	**3**
1	O	0.04756	0.8611	-	-
3	OA	0.04756	1.125	-	-
4	OE	0.04756	1.125	-	-
6	N	0.04936	1.301	-	-
7	NT	0.04936	1.301	-	-
8	NL	0.04936	1.301	-	-
9	NR	0.04936	1.301	-	-
10	NZ	0.04936	1.301	-	-
11	NE	0.04936	1.301	-	-
12	C	0.04838	1.837	-	-
13	CH0	0.04838	1.837	-	-
14	CH1	0.05396	1.933	-	-
15	CH2	0.06873	2.178	-	-
16	CH3	0.08278	2.456	-	-
18	CH2r	0.06873	2.178	-	-
19	CR1	0.07435	2.886	-	-
54	CH3p	0.08278	2.456	-	-

TABLE 3.26. GROMOS 54A7 and 54B7 third-neighbour van der Waals parameters

atom name	charge in e	occurring in
N	-0.310	all residues
H	0.310	
C	0.450	all residues
O	-0.450	
CD	0.090 (0.0)	Arg (charge +1)
NE	-0.110 (-0.240)	
HE	0.240 (0.240)	
CZ	0.340 (0.0)	
NH1/2	-0.260 (-0.480)	
HH11/12/21/22	0.240 (0.240)	
NE	-0.310	Argn (neutral)
HE	0.310	
CZ	0.266	Argn (neutral)
NH1	-0.674	
HH1	0.408	
NH2	-0.880	Argn (neutral)
HH21/22	0.440	
CG, CD	0.290	Asn, Gln
OD1, OE1	-0.450	
ND2, NE2, NZ	-0.720	Asn, Gln
HD21/22, HE21/22, HZ1/2	0.440	
CG, CD	0.270 (0.720)	Asp, Glu (charge -1)
OD1/2, OE1/2	-0.635 (-0.360)	
CG, CD	0.330	Asph, Gluh
OD1, OE1	-0.450	
OD2, OE2	-0.288	
HD2, HE2	0.408	
CB	-0.100 (0.200)	Cys (charge -.5)
SG	-0.400 (-0.200)	
CB	0.150	Cysh
SG	-0.370	
HG	0.220	
CG	0.0	Hisa (proton at D1)
ND1	-0.050	
HD1	0.310	
CD2	0.0	
HD2	0.140	
CE1	0.0	
HE1	0.140	
NE2	-0.540	
CG	0.0	Hisb (proton at E2)
ND1	-0.540	
CD2	0.0	
HD2	0.140	
CE1	0.0	

TABLE 3.27: continues on next page.

atom name	charge in e	occurring in
HE1	0.140	
NE2	-0.050	
HE2	0.310	
CG	-0.050 (0.0)	Hish (charge +1)
ND1	0.380 (-0.300)	
HD1	0.300 (0.300)	
CD2	-0.100 (-0.100)	
HD2	0.100 (0.100)	
CE1	-0.340 (-0.100)	
HE1	0.100 (0.100)	
NE2	0.310 (-0.300)	
HE2	0.300 (0.300)	
CG, CB, CB	0.266	Hypr, Ser, Thr
OD1, OG, OG1	-0.674	
HD1, HG, HG1	0.408	
CE	0.127 (0.0)	Lysh (charge +1)
NZ	0.129 (-0.744)	
HZ1/2/3	0.248 (0.248)	
CE	-0.240	Lys (neutral)
NZ	-0.640	
HZ1/2	0.440	
CG /CE	0.241	Met
SD	-0.482	
CG	-0.210	Trp
CD1	-0.140	
HD1	0.140	
CD2	0.0	
NE1	-0.100	
HE1	0.310	
CE2	0.0	
CZ	0.203	Tyr
OH	-0.611	
HH	0.408	
C	-0.140	all aromatic C-H groups in Phe, Tyr, Trp
H	0.140	
C	0.0	aromatic C connected to an aliphatic CHn in Phe, Tyr
CH2	0.0	aliphatic CHn connected to aromatic C in Phe, Tyr

TABLE 3.27: GROMOS 54A7 (54B7) atomic charges and charge group definitions for amino acid residues. The charges for the 54B7 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
OW	-0.82000	H_2O (SPC model)
HW1/2	0.41000	
OW	-0.84760	H_2O (SPC/E model)
HW1/2	0.42380	
OW	-0.68850	H_2O (SPC/L model)
HW1/2	0.34425	
CChl	0.17900	Chloroform
CLChl	-0.08700	
HChl	0.08200	
SDmso	0.12753	DMSO
CDmso	0.16000	
ODmso	-0.44753	
CMet	0.26600	Methanol
OMet	-0.67400	
HMet	0.40800	
CCl4	0.0	Carbontetrachloride
CLCl4	0.0	
FTfe	-0.17000	2,2,2- Trifluoroethanol
CTfe	0.45200	
CHTfe	0.27300	
OTfe	-0.62500	
HTfe	0.41000	
CUrea	0.14200	Urea
OUrea	-0.39000	
NUrea	-0.54200	
HUrea	0.33300	

TABLE 3.28. GROMOS 54A7 atomic charges for various (co)solvents.

atom name	charge in e	occurring in
O3/5*	-0.360 (-0.360)	all nucleotides
P	0.990 (1.440)	
O1/2P	-0.635 (-0.360)	
C4*	0.160	all nucleotides
O4*	-0.360	
C1*	0.200	
N9, N9, N1, N1, N1	-0.200	dAde, dGua, dCyt, dThy, Ura
C4, C4	0.200	dAde, dGua
C6, C6, C6	0.100	dCyt, dThy, Ura
H6, H6, H6	0.100	dCyt, dThy, Ura
N1, N3, N7, N3, N7, N3	-0.540	dAde, dAde, dAde, dGua, dGua, dCyt
C2, C8, C8	0.440	dAde, dAde, dGua
H2, H8, H8	0.100	dAde, dAde, dGua
C6, C2, C4	0.540	dAde, dGua, dCyt
N6, N2, N4	-0.830	dAde, dGua, dCyt
H61/62, H21/22, H41/42	0.415	dAde, dGua, dCyt
N1, N3, N3	-0.310	dGua, dThy, Ura
H1, H3, H3	0.310	dGua, dThy, Ura
C6, C2, C2, C4, C2, C4	0.450	dGua, dCyt, dThy, dThy, Ura, Ura
O6, O2, O2, O4, O2, O4	-0.450	dGua, dCyt, dThy, dThy, Ura, Ura
C5, C5	-0.100	dCyt, Ura
H5, H5	0.100	dCyt, Ura

TABLE 3.29. GROMOS 54A7 (54B7) atomic charges and charge group definitions for nucleotides. The charges for the 54B7 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
C31, C33, C34, C35	0.400	dppc
N	-0.500	dppc
C32	0.300	dppc
O31	-0.700	dppc
O32, O33, O34	-0.800	dppc
P	1.700	dppc
C3	0.400	dppc
C2	0.300	dppc
C1	0.500	dppc
O11, O21, O22	-0.700	dppc
O12	-0.600	dppc
C11	0.800	dppc
C21	0.700	dppc
C12, ..., C22, ...	0.000	dppc

TABLE 3.30. GROMOS 54A7 (54B7) atomic charges and charge group definitions for lipids. The charges for the 54B7 force field are given between parentheses. The atoms that are not listed have zero partial charge and form single atom or multiple atom charge groups.

atom name	charge in e	occurring in
C1, C2, C3, C4, C6	0.232	hexopyranose, uronate
O2, O3, O4, O6	-0.642	hexopyranose, uronate
HO2, HO3, HO4, HO6	0.410	hexopyranose, uronate
C5	0.376	hexopyranose, uronate
O5	-0.480	hexopyranose, uronate
O1	-0.360	hexopyranose, uronate
C6	0.360 (0.720)	uronate
O61, O62	-0.680 (-0.360)	uronate
C1	0.232	terminal C1 - O1 - HO1 group
O1	-0.538	terminal C1 - O1 - HO1 group
HO1	0.410	terminal C1 - O1 - HO1 group
C1	0.232	terminal C1 - O1 - CM group
O1	-0.360	terminal C1 - O1 - CM group
CM (methyl)	0.232	terminal C1 - O1 - CM group
C5, C5'	0.378	terminal C1 - O1 - C1'(sugar) group
O5, O5'	-0.450	terminal C1 - O1 - C1'(sugar) group
C1, C1'	0.242	terminal C1 - O1 - C1'(sugar) group
O1	-0.340	terminal C1 - O1 - C1'(sugar) group

TABLE 3.31. GROMOS 54A7 (54B7) atomic charges and charge group definitions for carbohydrates.

CHAPTER 4

GROMOS molecular topology building blocks

4.1. Introduction

The GROMOS molecular topology building block files ∗.mtb contain the building blocks for a number of important types of molecules, such as proteins, DNA, RNA, sugars, etc. We note that three types of building blocks exist in the molecular topology building block file:

A. Solute building blocks with blockname MTBUILDBLSOLUTE
B. Solvent building blocks with blockname MTBUILDBLSOLVENT
C. Solute end-group building blocks with blockname MTBUILDBLEND

A solvent molecule may occur under the same name in types A and B of blocks, which contain partially different information. The solute block contains data on interaction function parameters for internal degrees of freedom and charge group information not present in the solvent block. The solvent block contains data on geometric constraints not present in the solute block. Examples of molecules that can be treated as solute molecule as well as solvent molecule are water (H2O, H2OE), chloroform (CHCL3), DMSO (DMSO), methanol (CH3OH) and carbontetrachloride (CCL4).

Below we list for each building block of the file 54a7.mtb (α-amino acids, lipids, nucleotides and solvents), 54a7_beta.mtb (β-amino acids), 54a7_cof.mtb (cofactors and other types of molecules) and 54a7_carbo.mtb (carbohydrates and sugars) the building block name and a description of the residue, nucleotide, glucose unit or molecule it is representing. The building blocks that are marked with a dagger are specified in the molecular topology building block files but are not presented in graphical and tabular form in this chapter. In the pictures of the building blocks the charged state is indicated for the simulation in solution. In the force field for in vacuo simulations (file 54b7.mtb), groups of atoms bear a total charge of zero, which is not indicated here. We note that the IUPAC-IUB nomenclature has been used throughout. When no IUPAC-IUB rules were defined (HEME group) Brookhaven protein data bank nomenclature has been used.

A. *Solute Building Blocks* (Blockname: MTBUILDBLSOLUTE)

Name	**Description** (charges in e)
α-Amino Acids and Analogues (L if not indicated otherwise)	
ALA	Alanine
ARG	Arginine (protonated; charge $+e$)
ARGN	Arginine (deprotonated; neutral)
ASN	Asparagine
ASN1	Asparagine (coordinated with ZN)
ASP	Aspartic acid (deprotonated; charge -e)
ASPH	Aspartic acid (protonated; neutral)
CYS	Cysteine (deprotonated; charge -1/2e)

CYSH	Cysteine (protonated; neutral)
CYS1	Cysteine (1st member of S-S bridge)
CYS2	Cysteine (2nd member of S-S bridge)
GLN	Glutamine
GLU	Glutamic acid (deprotonated; charge -e)
GLUH	Glutamic acid (protonated; neutral)
GLY	Glycine
HISA	Histidine (protonated at ND1; neutral)
HISB	Histidine (protonated at NE2; neutral)
HISH	Histidine (protonated at ND1 and NE2; charge +e)
HIS1	Histidine (coupled to HEME at NE2; neutral)
HIS2	Histidine (coupled to HEMC at NE2: neutral)
HYPR	Hydroxyproline (R-configuration at CG)
†HYPS	Hydroxyproline (S-configuration at CG)
ILE	Isoleucine
LEU	Leucine
LYS	Lysine (deprotonated; neutral)
LYSH	Lysine (protonated; charge +e)
MET	Methionine
PHE	Phenylalanine
PRO	Proline
SER	Serine
THR	Threonine
TRP	Tryptophan
TYR	Tyrosine
VAL	Valine
DALA	D-Alanine
ABU	L-2-amino-butanoic acid
AIB	2-aminoisobutyric acid
MEBMT	(4R)-4-[(E)-2-butanyl]-4, N-dimethyl-L-threonine
MELEU	N-methyl-L-leucine
MEVAL	N-methyl-L-valine
SAR	Sarcosine or N-methylglycine

β-Amino Acids

RAF	(R)-β^2-Phenylalanine
RAV	(R)-β^2-Valine
RBCH	(R)-β^3-Cysteine (protonated; neutral)
RBI	(R)-β^3-Isoleucine
RBKH	(R)-β^3-Lysine (protonated; charge $+e$)
RBM	(R)-β^3-Methionine
RBN	(R)-β^3-Asparagine
RBS	(R)-β^3-Serine
RBSP	(R)-β^3-Serine(propylated)
RBT	(R)-β^3-Threonine
RBV	(R)-β^3-Valine
SAA	(S)-β^2-Alanine
SAF	(S)-β^2-Phenylalanine
SAFF	(S)-β^2-Phenylalanine(Cα fluorinated)
SAL	(S)-β^2-Leucine
SAM	(S)-β^2-Methionine
SAV	(S)-β^2-Valine
SBA	(S)-β^3-Alanine
SBCH	(S)-β^3-Cysteine (protonated)
SBD	(S)-β^3-Aspartic acid (deprotonated; charge -e)
SBDH	(S)-β^3-Aspartic acid (protonated; neutral)
SBE	(S)-β^3-Glutamic acid (deprotonated; charge -e)
SBEH	(S)-β^3-Glutamic acid (protonated; neutral)
SBQ	(S)-β^3-Glutamine
SBF	(S)-β^3-Phenylalanine
BGL	β-Glycine
SBHA	(S)-β^3-Histidine (protonated at NE1; neutral)
SBHH	(S)-β^3-Histidine(protonated at NE1 and NZ2; charge $+e$)
SBI	(S)-β^3-Isoleucine
SBKH	(S)-β^3-Lysine (protonated; charge $+e$)
SBL	(S)-β^3-Leucine
SBM	(S)-β^3-Methionine

SBP	(S)-β^3-Proline
SBR	(S)-β^3-Arginine (protonated; charge $+e$)
SBS	(S)-β^3-Serine
SBT	(S)-β^3-Threonine
SBV	(S)-β^3-Valine
SBY	(S)-β^3-Tyrosine
SBW	(S)-β^3-Tryptophan
SRAM	(R,S)-$\beta^(2,3)$-Alanine(αMe)
SRLM	(R,S)-$\beta^(2,3)$-Leucine(αMe)
SRVM	(R,S)-$\beta^(2,3)$-Valine(αMe)
SSAM	(S,S)-$\beta^(2,3)$-Alanine(αMe)

Nucleotides

DADE	2'-deoxyadenosine 5'-phosphoric acid (DNA, charge -e)
DGUA	2'-deoxyguanosine 5'-phosphoric acid (DNA, charge -e)
DCYT	2'-deoxycitidine 5'-phosphoric acid (DNA, charge -e)
DTHY	2'-deoxythymidine 5'-phosphoric acid (DNA, charge -e)
ADE	adenosine 5'-phosphoric acid (RNA, charge -e)
GUA	guanosine 5'-phosphoric acid (RNA, charge -e)
CYT	cytidine 5'-phosphoric acid (RNA, charge -e)
URA	uridine 5'-phosphoric acid (RNA, charge -e)
FMNO	flavin mononucleotide (oxydized, deprotonated at FN5 and FN1; charge -e, $OPOHO_2^-$)
†FMNS	flavin mononucleotide (semi-reduced, protonated at FN5; charge -e, $OPOHO_2^-$)
†FMNR	flavin mononucleotide (reduced, protonated at FN5 and FN1; charge -e, $OPOHO_2^-$)
PFN	proflavin (protonated at FN5; charge $+e$)
NADP	nicotinamide adenine dinucleotide (NAD+; charge -e)
†NADH	nicotinamide adenine dinucleotide (NADH; charge -2e)
NDPH	nicotinamide adenine dinucleotide phosphate (NADPH; charge -3e, $OPOHO_2^-$)
†NDPP	nicotinamide adenine dinucleotide phosphate (NADP+; charge -2e, $OPOHO_2^-$)
†NDPHN	nicotinamide adenine dinucleotide phosphate (NADPH; neutral, $OPO(OH)_2$)

Carbohydrates

†NA2P	-2-D-allopyranose-α-1-

†NA3P	-3-D-allopyranose-α-1-
†NA4P	-4-D-allopyranose-α-1-
†NA6P	-6-D-allopyranose-α-1-
†NB2P	-2-D-allopyranose-β-1-
†NB3P	-3-D-allopyranose-β-1-
†NB4P	-4-D-allopyranose-β-1-
†NB6P	-6-D-allopyranose-β-1-
†EA2P	-2-D-altropyranose-α-1-
†EA3P	-3-D-altropyranose-α-1-
†EA4P	-4-D-altropyranose-α-1-
†EA6P	-6-D-altropyranose-α-1-
†EB2P	-2-D-altropyranose-β-1-
†EB3P	-3-D-altropyranose-β-1-
†EB4P	-4-D-altropyranose-β-1-
†EB6P	-6-D-altropyranose-β-1-
†GA2P	-2-D-glucopyranose-α-1-
†GA3P	-3-D-glucopyranose-α-1-
GA4P	-4-D-glucopyranose-α-1-
†GA6P	-6-D-glucopyranose-α-1-
GB2P	-2-D-glucopyranose-β-1-
GB3P	-3-D-glucopyranose-β-1-
GB4P	-4-D-glucopyranose-β-1-
GB6P	-6-D-glucopyranose-β-1-
†MA2P	-2-D-mannopyranose-α-1-
†MA3P	-3-D-mannopyranose-α-1-
†MA4P	-4-D-mannopyranose-α-1-
†MA6P	-6-D-mannopyranose-α-1-
†MB2P	-2-D-mannopyranose-β-1-
†MB3P	-3-D-mannopyranose-β-1-
†MB4P	-4-D-mannopyranose-β-1-
†MB6P	-6-D-mannopyranose-β-1-
†KA2P	-2-D-gulopyranose-α-1-
†KA3P	-3-D-gulopyranose-α-1-

†KA4P	-4-D-gulopyranose-α-1-
†KA6P	-6-D-gulopyranose-α-1-
†KB2P	-2-D-gulopyranose-β-1-
†KB3P	-3-D-gulopyranose-β-1-
†KB4P	-4-D-gulopyranose-β-1-
†KB6P	-6-D-gulopyranose-β-1-
†IA2P	-2-D-idopyranose-α-1-
†IA3P	-3-D-idopyranose-α-1-
†IA4P	-4-D-idopyranose-α-1-
†IA6P	-6-D-idopyranose-α-1-
†IB2P	-2-D-idopyranose-β-1-
†IB3P	-3-D-idopyranose-β-1-
†IB4P	-4-D-idopyranose-β-1-
†IB6P	-6-D-idopyranose-β-1-
†LA2P	-2-D-galactopyranose-α-1-
†LA3P	-3-D-galactopyranose-α-1-
†LA4P	-4-D-galactopyranose-α-1-
†LA6P	-6-D-galactopyranose-α-1-
†LB2P	-2-D-galactopyranose-β-1-
†LB3P	-3-D-galactopyranose-β-1-
LB4P	-4-D-galactopyranose-β-1-
†LB6P	-6-D-galactopyranose-β-1-
†TA2P	-2-D-talopyranose-α-1-
†TA3P	-3-D-talopyranose-α-1-
†TA4P	-4-D-talopyranose-α-1-
†TA6P	-6-D-talopyranose-α-1-
†TB2P	-2-D-talopyranose-β-1-
†TB3P	-3-D-talopyranose-β-1-
†TB4P	-4-D-talopyranose-β-1-
†TB6P	-6-D-talopyranose-β-1-
†nA2P	-2-L-allopyranose-α-1-
†nA3P	-3-L-allopyranose-α-1-
†nA4P	-4-L-allopyranose-α-1-

†nA6P	-6-L-allopyranose-α-1-
†nB2P	-2-L-allopyranose-β-1-
†nB3P	-3-L-allopyranose-β-1-
†nB4P	-4-L-allopyranose-β-1-
†nB6P	-6-L-allopyranose-β-1-
†eA2P	-2-L-altropyranose-α-1-
†eA3P	-3-L-altropyranose-α-1-
†eA4P	-4-L-altropyranose-α-1-
†eA6P	-6-L-altropyranose-α-1-
†eB2P	-2-L-altropyranose-β-1-
†eB3P	-3-L-altropyranose-β-1-
†eB4P	-4-L-altropyranose-β-1-
†eB6P	-6-L-altropyranose-β-1-
†gA2P	-2-L-glucopyranose-α-1-
†gA3P	-3-L-glucopyranose-α-1-
†gA4P	-4-L-glucopyranose-α-1-
†gA6P	-6-L-glucopyranose-α-1-
†gB2P	-2-L-glucopyranose-β-1-
†gB3P	-3-L-glucopyranose-β-1-
gB4P	-4-L-glucopyranose-β-1-
†gB6P	-6-L-glucopyranose-β-1-
†mA2P	-2-L-mannopyranose-α-1-
†mA3P	-3-L-mannopyranose-α-1-
†mA4P	-4-L-mannopyranose-α-1-
†mA6P	-6-L-mannopyranose-α-1-
†mB2P	-2-L-mannopyranose-β-1-
†mB3P	-3-L-mannopyranose-β-1-
†mB4P	-4-L-mannopyranose-β-1-
†mB6P	-6-L-mannopyranose-β-1-
†kA2P	-2-L-gulopyranose-α-1-
†kA3P	-3-L-gulopyranose-α-1-
†kA4P	-4-L-gulopyranose-α-1-
†kA6P	-6-L-gulopyranose-α-1-

†kB2P	-2-L-gulopyranose-β-1-
†kB3P	-3-L-gulopyranose-β-1-
†kB4P	-4-L-gulopyranose-β-1-
†kB6P	-6-L-gulopyranose-β-1-
†iA2P	-2-L-idopyranose-α-1-
†iA3P	-3-L-idopyranose-α-1-
†iA4P	-4-L-idopyranose-α-1-
†iA6P	-6-L-idopyranose-α-1-
†iB2P	-2-L-galactopyranose-β-1-
†iB3P	-3-L-galactopyranose-β-1-
†iB4P	-4-L-galactopyranose-β-1-
†iB6P	-6-L-galactopyranose-β-1-
†lA2P	-2-L-galactopyranose-α-1-
†lA3P	-3-L-galactopyranose-α-1-
†lA4P	-4-L-galactopyranose-α-1-
†lA6P	-6-L-galactopyranose-α-1-
†lB2P	-2-L-galactopyranose-β-1-
†lB3P	-3-L-galactopyranose-β-1-
†lB4P	-4-L-galactopyranose-β-1-
†lB6P	-6-L-galactopyranose-β-1-
†tA2P	-2-L-talopyranose-α-1-
†tA3P	-3-L-talopyranose-α-1-
†tA4P	-4-L-talopyranose-α-1-
†tA6P	-6-L-talopyranose-α-1-
†tB2P	-2-L-talopyranose-β-1-
†tB3P	-3-L-talopyranose-β-1-
†tB4P	-4-L-talopyranose-β-1-
†tB6P	-6-L-talopyranose-β-1-
GB4U	-4-D-glucuronate-β-1-
LA4U	-4-D-galacturonate-α-1-
MB4U	-4-D-mannuronate-β-1-
kA4U	-4-L-guluronate-α-1-
iA4U	-4-L-iduronate-α-1-

Other Molecules

†DPPC	dipalmitoylphosphatidylcholine
HEME	heme group (charge -2e, acidic groups deprotonated)
†HEMC	heme group (charge -2e, acidic groups deprotonated, CO coordinated)
†CYT*	3',5'-O-(tetra isopropyl-1,3-disiloxanediyl)cytidine (neutral)
†MTXH	methotrexate (protonated at N1; charge -2e)
FOL	folate (charge -2e)
†DHF	7,8-dihydrofolate (charge -2e)
†THF	5,6,7,8-tetrahydrofolate (charge -2e)
TMP	trimethoprim (deprotonated at N1; neutral)
†TMPH	trimethoprim (protonated at N1; neutral)
†TMPHP	trimethoprim (protonated at N1; charge +e)
PDG	3-phosopho-D-glycerate (charge -2e)
ATP	adenosine 5'-triphosphate (ATP; charge -3e)
PMB	p-methylbenzyl alcoholate (charge -e)
†PMBH	p-methylbenzyl alcohol (neutral)
BA	benzoic acid
RTOL	retinol (neutral)
†TEMP	tetramethyl pyrrolinyl (nitroxide spin label; neutral)
†CH4	methane (united atom)
†AR	argon
†ETH	ethanolate (obsolete: removed from 45A4 onward)
†ETHH	ethanol (obsolete: removed from 45A4 onward)
†GALB	β-galactose (obsolete: removed from 45A4 onward)
†GLCA	α-glucose (obsolete: removed from 45A4 onward)
†GLCB	β-glucose (obsolete: removed from 45A4 onward)

Ions

†SO42-	SO_4^{-2} ion (charge -2e)
†ZN2+	zinc ion (charge +2e)
†NA+	sodium ion (charge +e)
†CL-	chlorine ion (charge -e)
†CA2+	calcium ion (charge +2e)

†MG2+	magnesium ion (charge $+2e$)
†CU1+	copper ion (charge $+e$)
†CU2+	copper ion (charge $+2e$)

Solvents

(equivalent to the corresponding solvent building blocks)

†TFE	2,2,2-trifluoroethanol
†UREA	urea
†H2O	water (SPC model, rigid)[15]
†H2OE	water (SPC/E model, rigid)[38]
†CHCL3	chloroform (rigid)[19]
†DMSO	dimethylsulfoxide (rigid)[39]
†CH3OH	methanol (rigid)[17]
†CCL4	carbontetrachloride (rigid)[20]

B. *Solvent Building Blocks*(Blockname: MTBUILDBLSOLVENT)

Name	**Description**
†H2O	water (SPC model)[15]
†H2OE	water (SPC/E model)[38]
†CHCL3	chloroform[19]
†DMSO	dimethylsulfoxide[39]
†CH3OH	methanol[17]
†CCL4	carbontetrachloride[20]

C. *Solute End-Group Building Blocks* (Blockname: MTBUILDBLEND)

Name	**Description**
†NH3+	N-terminal α-peptide end-group (protonated, charge $+e$)
†NH2	N-terminal α-peptide end-group (deprotonated, neutral)
†NPRO	N-terminal α-peptide end-group for proline or hydroxy proline (protonated, charge $+e$)
†COO-	C-terminal α-peptide end-group (deprotonated, charge $-e$)
†COOH	C-terminal α-peptide end-group (protonated, neutral)
†BH3+	N-terminal β-peptide end-group (protonated, charge $+e$)
†BH2	N-terminal β-peptide end-group (deprotonated, neutral)
†BOO-	C-terminal β-peptide end-group (deprotonated, charge $-e$)

†BOOH	C-terminal β-peptide end-group (protonated, neutral)
†D5OH	5'-terminal DNA end-group
†D3OH	3'-terminal DNA end-group
†5OH	5'-terminal RNA end-group
†3OH	3'-terminal RNA end-group
†CNAP	terminal group for carbohydrates: -1-α-D-allopyranose
†CNBP	terminal group for carbohydrates: -1-β-D-allopyranose
†CEAP	terminal group for carbohydrates: -1-α-D-altropyranose
†CEBP	terminal group for carbohydrates: -1-β-D-altropyranose
†CGAP	terminal group for carbohydrates: -1-α-D-glucopyranose
CGBP	terminal group for carbohydrates: -1-β-D-glucopyranose
†CMAP	terminal group for carbohydrates: -1-α-D-mannopyranose
†CMBP	terminal group for carbohydrates: -1-β-D-mannopyranose
†CKAP	terminal group for carbohydrates: -1-α-D-gulopyranose
†CKBP	terminal group for carbohydrates: -1-β-D-gulopyranose
†CIAP	terminal group for carbohydrates: -1-α-D-idopyranose
†CIBP	terminal group for carbohydrates: -1-β-D-idopyranose
†CLAP	terminal group for carbohydrates: -1-α-D-galactopyranose
†CLBP	terminal group for carbohydrates: -1-β-D-galactopyranose
†CTAP	terminal group for carbohydrates: -1-α-D-talopyranose
†CTBP	terminal group for carbohydrates: -1-β-D-talopyranose
†CnAP	terminal group for carbohydrates: -1-α-L-allopyranose
†CnBP	terminal group for carbohydrates: -1-β-L-allopyranose
†CeAP	terminal group for carbohydrates: -1-α-L-altropyranose
†CeBP	terminal group for carbohydrates: -1-β-L-altropyranose
†CgAP	terminal group for carbohydrates: -1-α-L-glucopyranose
†CgBP	terminal group for carbohydrates: -1-β-L-glucopyranose
†CmAP	terminal group for carbohydrates: -1-α-L-mannopyranose
†CmBP	terminal group for carbohydrates: -1-β-L-mannopyranose
†CkAP	terminal group for carbohydrates: -1-α-L-gulopyranose
†CkBP	terminal group for carbohydrates: -1-β-L-gulopyranose
†CiAP	terminal group for carbohydrates: -1-α-L-idopyranose
†CiAP	terminal group for carbohydrates: -1-β-L-idopyranose

†ClAP	terminal group for carbohydrates: -1-α-L-galactopyranose
†ClBP	terminal group for carbohydrates: -1-β-L-galactopyranose
†CtAP	terminal group for carbohydrates: -1-α-L-talopyranose
†CtBP	terminal group for carbohydrates: -1-β-L-talopyranose
†CGBU	terminal group for carbohydrates: -1-β-D-glucuronate
†CLAU	terminal group for carbohydrates: -1-α-D-galacturonate
†CMBU	terminal group for carbohydrates: -1-β-D-mannuronate
†CkAU	terminal group for carbohydrates: -1-α-L-guluronate
†CiAU	terminal group for carbohydrates: -1-α-L-iduronate
†C1OC	terminal group for carbohydrates: -C1-O1-CH3
†C1OH	terminal group for carbohydrates: -C1-O1-HO1
†HO2C	initial group for carbohydrates: HO2-O2-C2-
†HO3C	initial group for carbohydrates: HO3-O3-C3-
†HO4C	initial group for carbohydrates: HO4-O4-C4-
†HO6C	initial group for carbohydrates: HO6-O6-C6-

4.2. Definition of molecular topology building block pictures

a. Figure 1, *atoms*

- names: X
- numbering ^{n}X
- integer atom code X^{IAC}
- charge: X_{charge}
- charge groups: color boundaries between different charge groups

b. Figure 2, *bonds and bond angles*

- bond type codes: thin, underlined
- bond-angle type codes: bold, italics

In both building block figures the stereochemical configuration is represented such that the solid wedges indicate bonds that project above the plane of the paper and hashed wedges indicate bonds that project below the plane of the paper. The wedges are always oriented with the narrow end at the stereogenic center.

4.3. α-amino acids and analogues

Solute building block: Alanine
Name: ALA

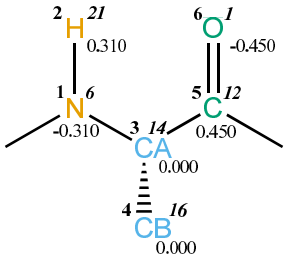


FIGURE 4.1. ALA non-bonded parameters.

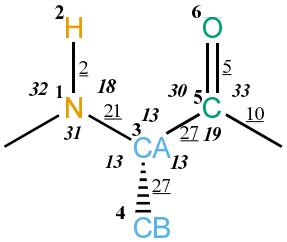


FIGURE 4.2. ALA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7
4	CB	16	5	0.00000	5
5	C	12	12	0.45000	
6	O	1	16	-0.45000	

TABLE 4.1. Atoms of building block ALA.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
5	6	5
5	7	10

TABLE 4.2. Bonds of building block ALA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
4	3	5	13
3	5	6	30
3	5	7	19
6	5	7	33

TABLE 4.3. Bond angles of building block ALA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	5	43
-1	1	3	5	44
1	3	5	7	42
1	3	5	7	45

TABLE 4.4. Dihedral angles of building block ALA.

I	J	K	L	Type
1	-1	3	2	1
3	1	5	4	2
5	3	7	6	1

TABLE 4.5. Improper dihedral angles of building block ALA.

Solute building block: Arginine (protonated; charge $+e$)
Name: ARG

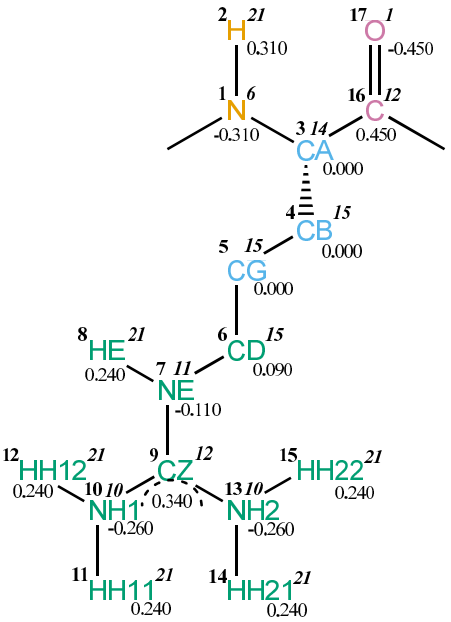


FIGURE 4.3. ARG non-bonded parameters.

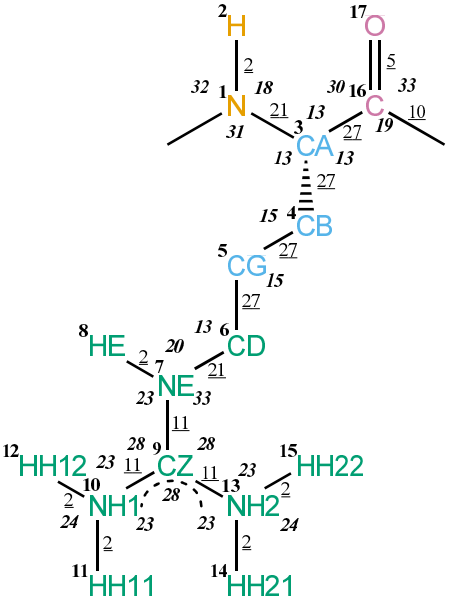


FIGURE 4.4. ARG bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 16
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 16 17 18
4	CB	15	4	0.00000	5 6 16
5	CG	15	4	0.00000	6 7
6	CD	15	4	0.09000	7 8 9
7	NE	11	14	-0.11000	8 9 10 13
8	HE	21	1	0.24000	9
9	CZ	12	12	0.34000	10 11 12 13 14 15
10	NH1	10	14	-0.26000	11 12 13
11	HH11	21	1	0.24000	12
12	HH12	21	1	0.24000	
13	NH2	10	14	-0.26000	14 15
14	HH21	21	1	0.24000	15
15	HH22	21	1	0.24000	
16	C	12	12	0.45000	
17	O	1	16	-0.45000	

TABLE 4.6. Atoms of building block ARG.

I	J	Type
1	2	2
1	3	21
3	4	27
3	16	27
4	5	27
5	6	27
6	7	21
7	8	2
7	9	11
9	10	11
9	13	11
10	11	2
10	12	2
13	14	2
13	15	2
16	17	5
16	18	10

TABLE 4.7. Bonds of building block ARG.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	16	13
4	3	16	13
3	4	5	15
4	5	6	15
5	6	7	13
6	7	8	20
6	7	9	33
8	7	9	23
7	9	10	28
7	9	13	28
10	9	13	28
9	10	11	23
9	10	12	23
11	10	12	24
9	13	14	23
9	13	15	23
14	13	15	24
3	16	17	30
3	16	18	19
17	16	18	33

TABLE 4.8. Bond angles of building block ARG.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	16	43
-1	1	3	16	44
1	3	4	5	34
1	3	16	18	42
1	3	16	18	45
3	4	5	6	34
4	5	6	7	34
5	6	7	9	39
6	7	9	10	14
7	9	10	11	14
7	9	13	14	14

TABLE 4.9. Dihedral angles of building block ARG.

I	J	K	L	Type
1	-1	3	2	1
3	1	16	4	2
7	6	9	8	1
9	10	13	7	1
10	11	12	9	1
13	14	15	9	1
16	3	18	17	1

TABLE 4.10. Improper dihedral angles of building block ARG.

Solute building block: Arginine (deprotonated; neutral)
Name: ARGN

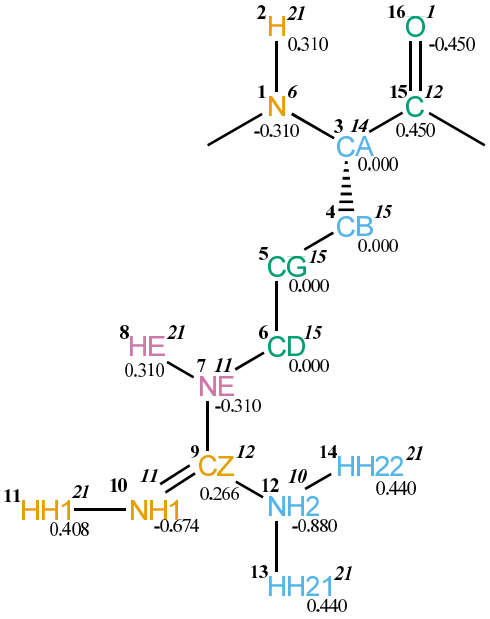


FIGURE 4.5. ARGN non-bonded parameters.

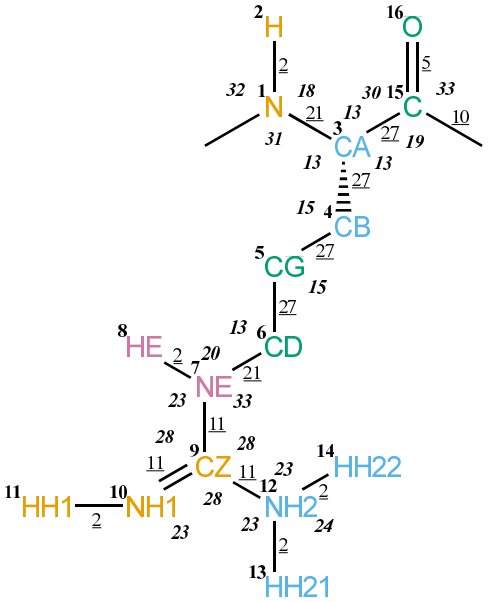


FIGURE 4.6. ARGN bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 15
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 15 16 17
4	CB	15	4	0.00000	5 6 15
5	CG	15	4	0.00000	6 7
6	CD	15	4	0.00000	7 8 9
7	NE	11	14	-0.31000	8 9 10 12
8	HE	21	1	0.31000	9
9	CZ	12	12	0.26600	10 11 12 13 14
10	NH1	11	14	-0.67400	11 12
11	HH1	21	1	0.40800	
12	NH2	10	14	-0.88000	13 14
13	HH21	21	1	0.44000	14
14	HH22	21	1	0.44000	
15	C	12	12	0.45000	
16	O	1	16	-0.45000	

TABLE 4.11. Atoms of building block ARGN.

I	J	Type
1	2	2
1	3	21
3	4	27
3	15	27
4	5	27
5	6	27
6	7	21
7	8	2
7	9	11
9	10	11
9	12	11
10	11	2
12	13	2
12	14	2
15	16	5
15	17	10

TABLE 4.12. Bonds of building block ARGN.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	15	13
4	3	15	13
3	4	5	15
4	5	6	15
5	6	7	13
6	7	8	20
6	7	9	33
8	7	9	23
7	9	10	28
7	9	12	28
10	9	12	28
9	10	11	23
9	12	13	23
9	12	14	23
13	12	14	24
3	15	16	30
3	15	17	19
16	15	17	33

TABLE 4.13. Bond angles of building block ARGN.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	15	43
-1	1	3	15	44
1	3	4	5	34
1	3	15	17	42
1	3	15	17	45
3	4	5	6	34
4	5	6	7	34
5	6	7	9	39
6	7	9	10	14
7	9	10	11	14
7	9	12	13	14

TABLE 4.14. Dihedral angles of building block ARGN.

I	J	K	L	Type
1	-1	3	2	1
3	1	15	4	2
7	6	9	8	1
9	10	12	7	1
12	13	14	9	1
15	3	17	16	1

TABLE 4.15. Improper dihedral angles of building block ARGN.

Solute building block: Asparagine
Name: ASN

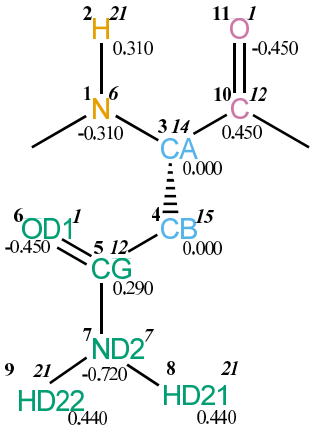


FIGURE 4.7. ASN non-bonded parameters.

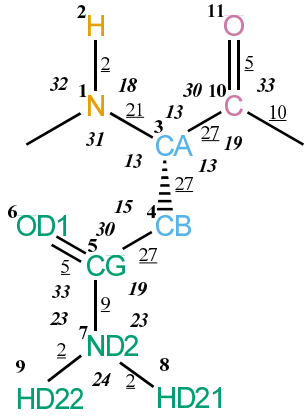


FIGURE 4.8. ASN bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 10
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 10 11 12
4	CB	15	4	0.00000	5 6 7 10
5	CG	12	12	0.29000	6 7 8 9
6	OD1	1	16	-0.45000	7
7	ND2	7	14	-0.72000	8 9
8	HD21	21	1	0.44000	9
9	HD22	21	1	0.44000	
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.16. Atoms of building block ASN.

I	J	Type
1	2	2
1	3	21
3	4	27
3	10	27
4	5	27
5	6	5
5	7	9
7	8	2
7	9	2
10	11	5
10	12	10

TABLE 4.17. Bonds of building block ASN.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	10	13
4	3	10	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33
5	7	8	23
5	7	9	23
8	7	9	24
3	10	11	30
3	10	12	19
11	10	12	33

TABLE 4.18. Bond angles of building block ASN.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	10	43
-1	1	3	10	44
1	3	4	5	34
1	3	10	12	42
1	3	10	12	45
3	4	5	7	40
4	5	7	8	14

TABLE 4.19. Dihedral angles of building block ASN.

I	J	K	L	Type
1	-1	3	2	1
3	1	10	4	2
5	6	7	4	1
7	8	9	5	1
10	3	12	11	1

TABLE 4.20. Improper dihedral angles of building block ASN.

Solute building block: Asparagine (coordinated with ZN)
Name: ASN1

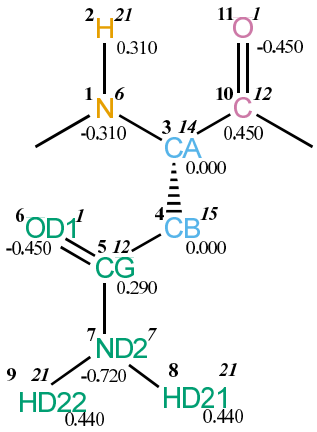


FIGURE 4.9. ASN1 non-bonded parameters.

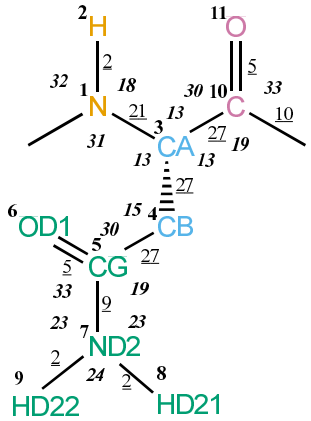


FIGURE 4.10. ASN1 bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 10
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 10 11 12
4	CB	15	4	0.00000	5 6 7 10
5	CG	12	12	0.29000	6 7 8 9
6	OD1	1	16	-0.45000	7
7	ND2	8	14	-0.72000	8 9
8	HD21	21	1	0.44000	9
9	HD22	21	1	0.44000	
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.21. Atoms of building block ASN1.

I	J	Type
1	2	2
1	3	21
3	4	27
3	10	27
4	5	27
5	6	5
5	7	9
7	8	2
7	9	2
10	11	5
10	12	10

TABLE 4.22. Bonds of building block ASN1.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	10	13
4	3	10	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33
5	7	8	23
5	7	9	23
8	7	9	24
3	10	11	30
3	10	12	19
11	10	12	33

TABLE 4.23. Bond angles of building block ASN1.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	10	43
-1	1	3	10	44
1	3	4	5	34
1	3	10	12	42
1	3	10	12	45
3	4	5	7	40
4	5	7	8	14

TABLE 4.24. Dihedral angles of building block ASN1.

I	J	K	L	Type
1	-1	3	2	1
3	1	10	4	2
5	6	7	4	1
7	8	9	5	1
10	3	12	11	1

TABLE 4.25. Improper dihedral angles of building block ASN1.

Solute building block: Aspartic acid (deprotonated; charge -e)
Name: ASP

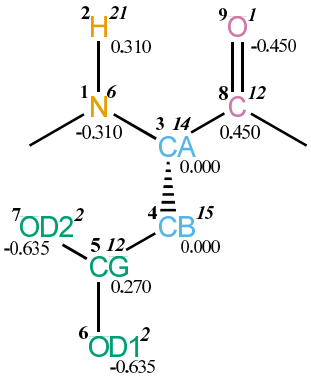


FIGURE 4.11. ASP non-bonded parameters.

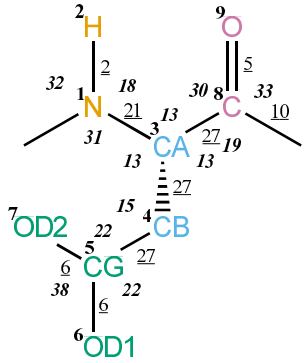


FIGURE 4.12. ASP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 8 9 10
4	CB	15	4	0.00000	5 6 7 8
5	CG	12	12	0.27000	6 7
6	OD1	2	16	-0.63500	7
7	OD2	2	16	-0.63500	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.26. Atoms of building block ASP.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	6
5	7	6
8	9	5
8	10	10

TABLE 4.27. Bonds of building block ASP.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	22
4	5	7	22
6	5	7	38
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.28. Bond angles of building block ASP.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	6	40

TABLE 4.29. Dihedral angles of building block ASP.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
5	6	7	4	1
8	3	10	9	1

TABLE 4.30. Improper dihedral angles of building block ASP.

Solute building block: Aspartic acid (protonated; neutral)
Name: ASPH

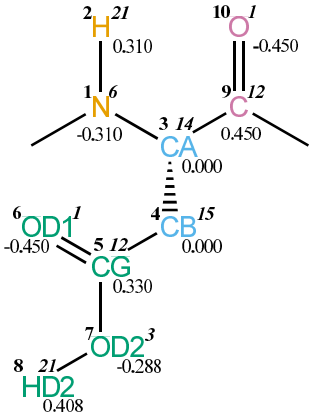


FIGURE 4.13. ASPH non-bonded parameters.

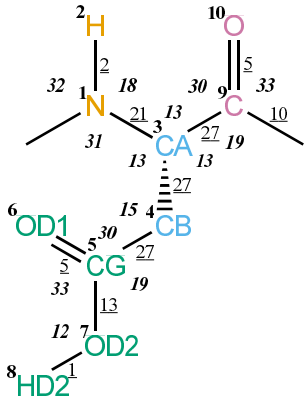


FIGURE 4.14. ASPH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 9
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 9 10 11
4	CB	15	4	0.00000	5 6 7 9
5	CG	12	12	0.33000	6 7 8
6	OD1	1	16	-0.45000	7
7	OD2	3	16	-0.28800	8
8	HD2	21	1	0.40800	
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.31. Atoms of building block ASPH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	9	27
4	5	27
5	6	5
5	7	13
7	8	1
9	10	5
9	11	10

TABLE 4.32. Bonds of building block ASPH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	9	13
4	3	9	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33
5	7	8	12
3	9	10	30
3	9	11	19
10	9	11	33

TABLE 4.33. Bond angles of building block ASPH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	9	43
-1	1	3	9	44
1	3	4	5	34
1	3	9	11	42
1	3	9	11	45
3	4	5	7	40
4	5	7	8	12

TABLE 4.34. Dihedral angles of building block ASPH.

I	J	K	L	Type
1	-1	3	2	1
3	1	9	4	2
5	6	7	4	1
9	3	11	10	1

TABLE 4.35. Improper dihedral angles of building block ASPH.

Solute building block: Cysteine (deprotonated; charge -1/2e)
Name: CYS

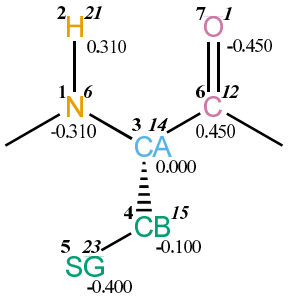


FIGURE 4.15. CYS non-bonded parameters.

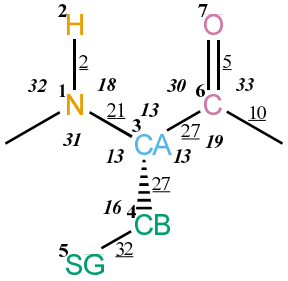


FIGURE 4.16. CYS bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 6
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8
4	CB	15	4	-0.10000	5 6
5	SG	23	32	-0.40000	
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.36. Atoms of building block CYS.

I	J	Type
1	2	2
1	3	21
3	4	27
3	6	27
4	5	32
6	7	5
6	8	10

TABLE 4.37. Bonds of building block CYS.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	6	13
4	3	6	13
3	4	5	16
3	6	7	30
3	6	8	19
7	6	8	33

TABLE 4.38. Bond angles of building block CYS.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	6	43
-1	1	3	6	44
1	3	4	5	34
1	3	6	8	42
1	3	6	8	45

TABLE 4.39. Dihedral angles of building block CYS.

I	J	K	L	Type
1	-1	3	2	1
3	1	6	4	2
6	3	8	7	1

TABLE 4.40. Improper dihedral angles of building block CYS.

Solute building block: Cysteine (protonated; neutral)
Name: CYSH

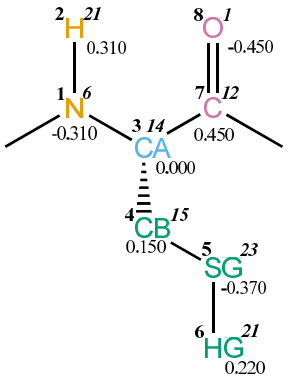


FIGURE 4.17. CYSH non-bonded parameters.

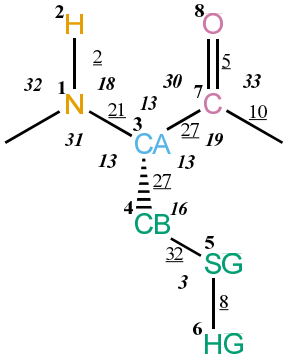


FIGURE 4.18. CYSH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 7 8 9
4	CB	15	4	0.15000	5 6 7
5	SG	23	32	-0.37000	6
6	HG	21	1	0.22000	
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.41. Atoms of building block CYSH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	32
5	6	8
7	8	5
7	9	10

TABLE 4.42. Bonds of building block CYSH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	16
4	5	6	3
3	7	8	30
3	7	9	19
8	7	9	33

TABLE 4.43. Bond angles of building block CYSH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	9	42
1	3	7	9	45
3	4	5	6	26

TABLE 4.44. Dihedral angles of building block CYSH.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
7	3	9	8	1

TABLE 4.45. Improper dihedral angles of building block CYSH.

Solute building block: Cysteine (1st member of S-S bridge)
Name: CYS1

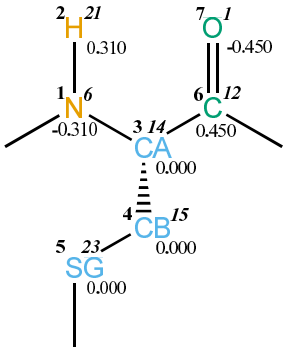


FIGURE 4.19. CYS1 non-bonded parameters.

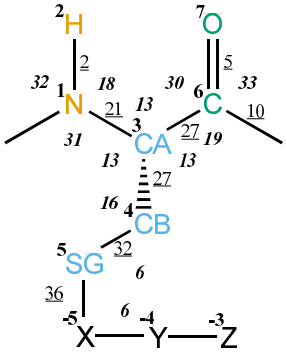


FIGURE 4.20. CYS1 bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 6
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8
4	CB	15	4	0.00000	-5 5 6
5	SG	23	32	0.00000	-5 -4
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.46. Atoms of building block CYS1.

I	J	Type
-5	5	36
1	2	2
1	3	21
3	4	27
3	6	27
4	5	32
6	7	5
6	8	10

TABLE 4.47. Bonds of building block CYS1.

I	J	K	Type
-4	-5	5	6
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	6	13
4	3	6	13
3	4	5	16
-5	5	4	6
3	6	7	30
3	6	8	19
7	6	8	33

TABLE 4.48. Bond angles of building block CYS1.

I	J	K	L	Type
5	-5	-4	-3	26
-4	-5	5	4	21
-2	-1	1	3	14
-1	1	3	6	43
-1	1	3	6	44
1	3	4	5	34
1	3	6	8	42
1	3	6	8	45
3	4	5	-5	26

TABLE 4.49. Dihedral angles of building block CYS1.

I	J	K	L	Type
1	-1	3	2	1
3	1	6	4	2
6	3	8	7	1

TABLE 4.50. Improper dihedral angles of building block CYS1.

Solute building block: Cysteine (2nd member of S-S bridge)
Name: CYS2

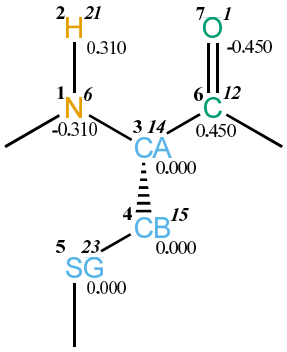


FIGURE 4.21. CYS2 non-bonded parameters.

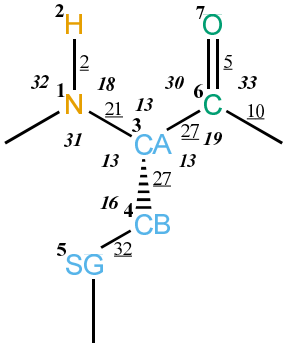


FIGURE 4.22. CYS2 bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 6
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8
4	CB	15	4	0.00000	5 6
5	SG	23	32	0.00000	
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.51. Atoms of building block CYS2.

I	J	Type
1	2	2
1	3	21
3	4	27
3	6	27
4	5	32
6	7	5
6	8	10

TABLE 4.52. Bonds of building block CYS2.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	6	13
4	3	6	13
3	4	5	16
3	6	7	30
3	6	8	19
7	6	8	33

TABLE 4.53. Bond angles of building block CYS2.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	6	43
-1	1	3	6	44
1	3	4	5	34
1	3	6	8	42
1	3	6	8	45

TABLE 4.54. Dihedral angles of building block CYS2.

I	J	K	L	Type
1	-1	3	2	1
3	1	6	4	2
6	3	8	7	1

TABLE 4.55. Improper dihedral angles of building block CYS2.

Solute building block: Glutamine
Name: GLN

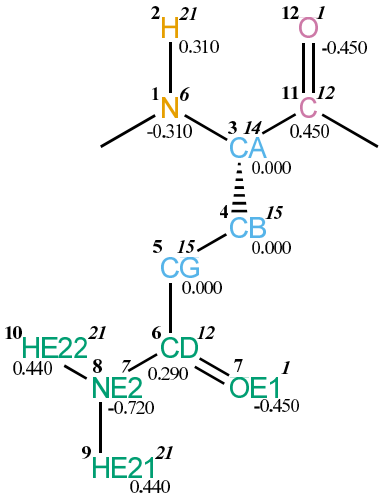


FIGURE 4.23. GLN non-bonded parameters.

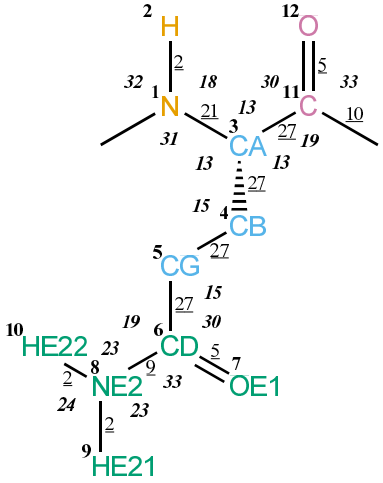


FIGURE 4.24. GLN bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 11
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 11 12 13
4	CB	15	4	0.00000	5 6 11
5	CG	15	4	0.00000	6 7 8
6	CD	12	12	0.29000	7 8 9 10
7	OE1	1	16	-0.45000	8
8	NE2	7	14	-0.72000	9 10
9	HE21	21	1	0.44000	10
10	HE22	21	1	0.44000	
11	C	12	12	0.45000	
12	O	1	16	-0.45000	

TABLE 4.56. Atoms of building block GLN.

I	J	Type
1	2	2
1	3	21
3	4	27
3	11	27
4	5	27
5	6	27
6	7	5
6	8	9
8	9	2
8	10	2
11	12	5
11	13	10

TABLE 4.57. Bonds of building block GLN.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	11	13
4	3	11	13
3	4	5	15
4	5	6	15
5	6	7	30
5	6	8	19
7	6	8	33
6	8	9	23
6	8	10	23
9	8	10	24
3	11	12	30
3	11	13	19
12	11	13	33

TABLE 4.58. Bond angles of building block GLN.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	11	43
-1	1	3	11	44
1	3	4	5	34
1	3	11	13	42
1	3	11	13	45
3	4	5	6	34
4	5	6	8	40
5	6	8	9	14

TABLE 4.59. Dihedral angles of building block GLN.

I	J	K	L	Type
1	-1	3	2	1
3	1	11	4	2
6	7	8	5	1
8	9	10	6	1
11	3	13	12	1

TABLE 4.60. Improper dihedral angles of building block GLN.

Solute building block: Glutamic acid (deprotonated; charge -e)
Name: GLU

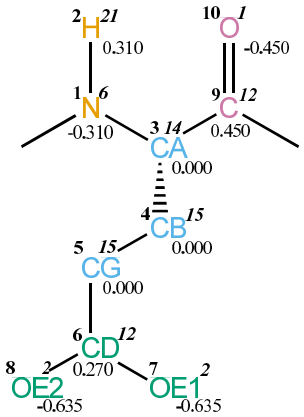


FIGURE 4.25. GLU non-bonded parameters.

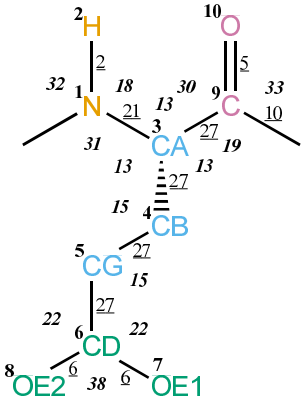


FIGURE 4.26. GLU bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 9
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 9 10 11
4	CB	15	4	0.00000	5 6 9
5	CG	15	4	0.00000	6 7 8
6	CD	12	12	0.27000	7 8
7	OE1	2	16	-0.63500	8
8	OE2	2	16	-0.63500	
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.61. Atoms of building block GLU.

I	J	Type
1	2	2
1	3	21
3	4	27
3	9	27
4	5	27
5	6	27
6	7	6
6	8	6
9	10	5
9	11	10

TABLE 4.62. Bonds of building block GLU.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	9	13
4	3	9	13
3	4	5	15
4	5	6	15
5	6	7	22
5	6	8	22
7	6	8	38
3	9	10	30
3	9	11	19
10	9	11	33

TABLE 4.63. Bond angles of building block GLU.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	9	43
-1	1	3	9	44
1	3	4	5	34
1	3	9	11	42
1	3	9	11	45
3	4	5	6	34
4	5	6	8	40

TABLE 4.64. Dihedral angles of building block GLU.

I	J	K	L	Type
1	-1	3	2	1
3	1	9	4	2
6	7	8	5	1
9	3	11	10	1

TABLE 4.65. Improper dihedral angles of building block GLU.

Solute building block: Glutamic acid (protonated; neutral)
Name: GLUH

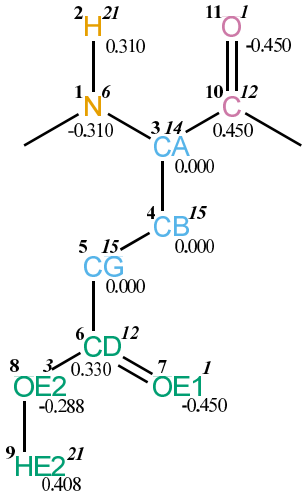


FIGURE 4.27. GLUH non-bonded parameters.

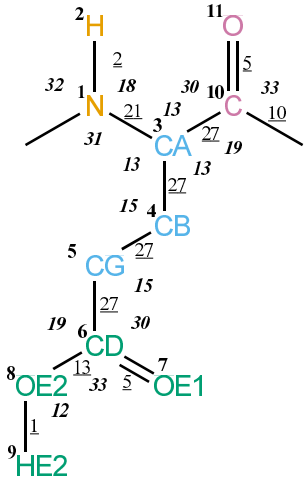


FIGURE 4.28. GLUH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 10
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 10 11 12
4	CB	15	4	0.00000	5 6 10
5	CG	15	4	0.00000	6 7 8
6	CD	12	12	0.33000	7 8 9
7	OE1	1	16	-0.45000	8
8	OE2	3	16	-0.28800	9
9	HE2	21	1	0.40800	
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.66. Atoms of building block GLUH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	10	27
4	5	27
5	6	27
6	7	5
6	8	13
8	9	1
10	11	5
10	12	10

TABLE 4.67. Bonds of building block GLUH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	10	13
4	3	10	13
3	4	5	15
4	5	6	15
5	6	7	30
5	6	8	19
7	6	8	33
6	8	9	12
3	10	11	30
3	10	12	19
11	10	12	33

TABLE 4.68. Bond angles of building block GLUH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	10	43
-1	1	3	10	44
1	3	4	5	34
1	3	10	12	42
1	3	10	12	45
3	4	5	6	34
4	5	6	8	40
5	6	8	9	12

TABLE 4.69. Dihedral angles of building block GLUH.

I	J	K	L	Type
1	-1	3	2	1
3	1	10	4	2
6	7	8	5	1
10	3	12	11	1

TABLE 4.70. Improper dihedral angles of building block GLUH.

Solute building block: Glycine
Name: GLY

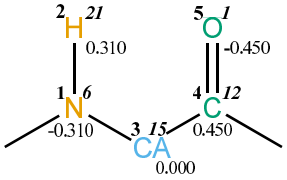


FIGURE 4.29. GLY non-bonded parameters.

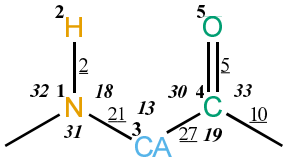


FIGURE 4.30. GLY bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CA	15	4	0.00000	4 5 6
4	C	12	12	0.45000	
5	O	1	16	-0.45000	

TABLE 4.71. Atoms of building block GLY.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	5
4	6	10

TABLE 4.72. Bonds of building block GLY.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	30
3	4	6	19
5	4	6	33

TABLE 4.73. Bond angles of building block GLY.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	6	42
1	3	4	6	45

TABLE 4.74. Dihedral angles of building block GLY.

I	J	K	L	Type
1	-1	3	2	1
4	3	6	5	1

TABLE 4.75. Improper dihedral angles of building block GLY.

Solute building block: Histidine (protonated at ND1; neutral)
Name: HISA

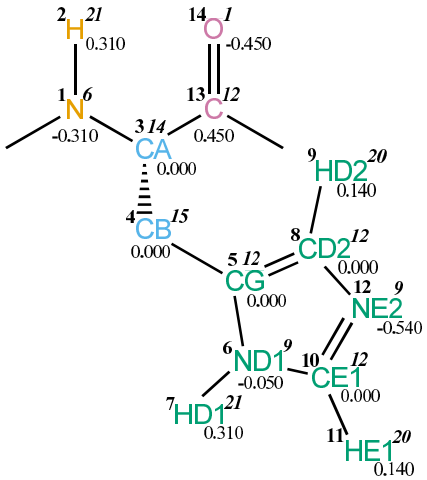


FIGURE 4.31. HISA non-bonded parameters.

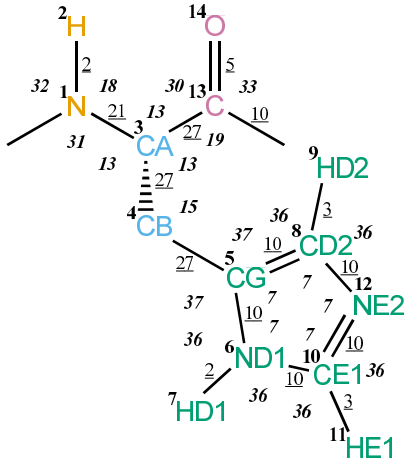


FIGURE 4.32. HISA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 13
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 13 14 15
4	CB	15	4	0.00000	5 6 7 8 9 10 12 13
5	CG	12	12	0.00000	6 7 8 9 10 11 12
6	ND1	9	14	-0.05000	7 8 9 10 11 12
7	HD1	21	1	0.31000	8 10 11 12
8	CD2	12	12	0.00000	9 10 11 12
9	HD2	20	1	0.14000	10 12
10	CE1	12	12	0.00000	11 12
11	HE1	20	1	0.14000	12
12	NE2	9	14	-0.54000	
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.76. Atoms of building block HISA.

I	J	Type
1	2	2
1	3	21
3	4	27
3	13	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
13	14	5
13	15	10

TABLE 4.77. Bonds of building block HISA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
8	12	10	7
3	13	14	30
3	13	15	19
14	13	15	33

TABLE 4.78. Bond angles of building block HISA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	13	43
-1	1	3	13	44
1	3	4	5	34
1	3	13	15	42
1	3	13	15	45
3	4	5	6	40

TABLE 4.79. Dihedral angles of building block HISA.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
13	3	15	14	1

TABLE 4.80. Improper dihedral angles of building block HISA.

Solute building block: Histidine (protonated at NE2; neutral)
Name: HISB

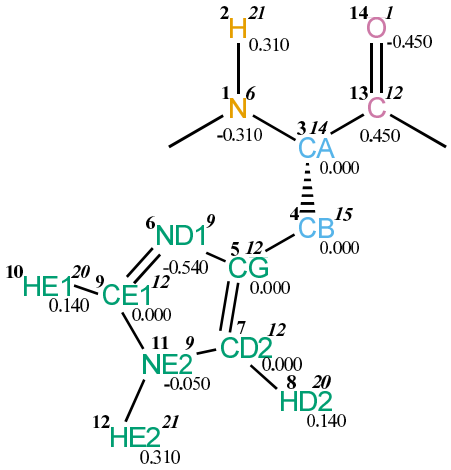


FIGURE 4.33. HISB non-bonded parameters.

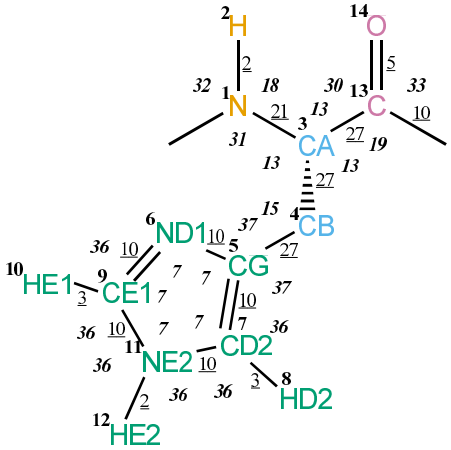


FIGURE 4.34. HISB bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 13
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 13 14 15
4	CB	15	4	0.00000	5 6 7 8 9 11 13
5	CG	12	12	0.00000	6 7 8 9 10 11 12
6	ND1	9	14	-0.54000	7 8 9 10 11 12
7	CD2	12	12	0.00000	8 9 10 11 12
8	HD2	20	1	0.14000	9 11 12
9	CE1	12	12	0.00000	10 11 12
10	HE1	20	1	0.14000	11 12
11	NE2	9	14	-0.05000	12
12	HE2	21	1	0.31000	
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.81. Atoms of building block HISB.

I	J	Type
1	2	2
1	3	21
3	4	27
3	13	27
4	5	27
5	6	10
5	7	10
6	9	10
7	8	3
7	11	10
9	10	3
9	11	10
11	12	2
13	14	5
13	15	10

TABLE 4.82. Bonds of building block HISB.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	15
4	5	6	37
4	5	7	37
6	5	7	7
5	6	9	7
5	7	8	36
5	7	11	7
8	7	11	36
6	9	10	36
6	9	11	7
10	9	11	36
7	11	9	7
7	11	12	36
9	11	12	36
3	13	14	30
3	13	15	19
14	13	15	33

TABLE 4.83. Bond angles of building block HISB.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	13	43
-1	1	3	13	44
1	3	4	5	34
1	3	13	15	42
1	3	13	15	45
3	4	5	6	40

TABLE 4.84. Dihedral angles of building block HISB.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
5	6	7	4	1
5	6	9	11	1
5	7	11	9	1
6	5	7	11	1
6	9	11	7	1
7	5	6	9	1
7	5	11	8	1
9	6	11	10	1
11	7	9	12	1
13	3	15	14	1

TABLE 4.85. Improper dihedral angles of building block HISB.

Solute building block: Histidine (protonated at ND1 and NE2; charge $+e$)
Name: HISH

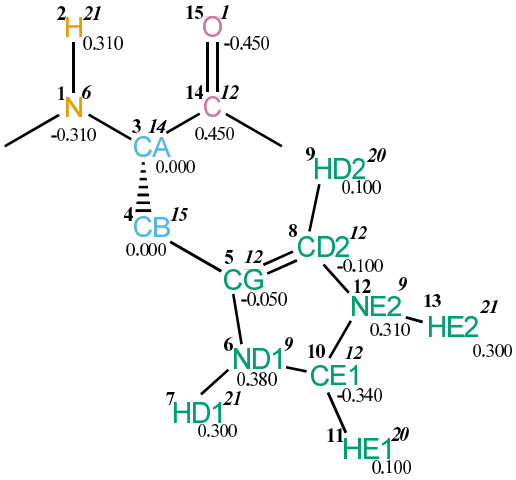


FIGURE 4.35. HISH non-bonded parameters.

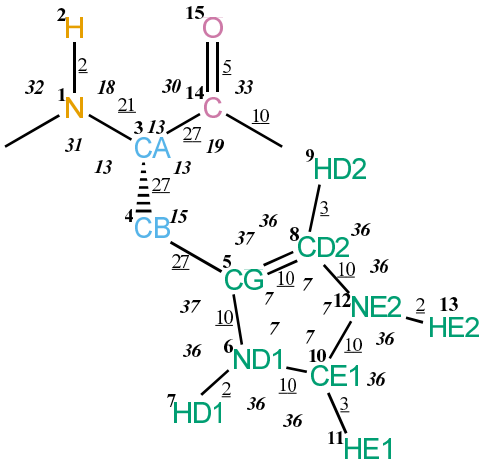


FIGURE 4.36. HISH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 14
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 14 15 16
4	CB	15	4	0.00000	5 6 7 8 9 10 12 14
5	CG	12	12	-0.05000	6 7 8 9 10 11 12 13
6	ND1	9	14	0.38000	7 8 9 10 11 12 13
7	HD1	21	1	0.30000	8 10 11 12
8	CD2	12	12	-0.10000	9 10 11 12 13
9	HD2	20	1	0.10000	10 12 13
10	CE1	12	12	-0.34000	11 12 13
11	HE1	20	1	0.10000	12 13
12	NE2	9	14	0.31000	13
13	HE2	21	1	0.30000	
14	C	12	12	0.45000	
15	O	1	16	-0.45000	

TABLE 4.86. Atoms of building block HISH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	14	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
12	13	2
14	15	5
14	16	10

TABLE 4.87. Bonds of building block HISH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	14	13
4	3	14	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
8	12	10	7
8	12	13	36
10	12	13	36
3	14	15	30
3	14	16	19
15	14	16	33

TABLE 4.88. Bond angles of building block HISH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	14	43
-1	1	3	14	44
1	3	4	5	34
1	3	14	16	42
1	3	14	16	45
3	4	5	6	40

TABLE 4.89. Dihedral angles of building block HISH.

I	J	K	L	Type
1	-1	3	2	1
3	1	14	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
12	8	10	13	1
14	3	16	15	1

TABLE 4.90. Improper dihedral angles of building block HISH.

Solute building block: Histidine (coupled to HEME at NE2; neutral)
Name: HIS1

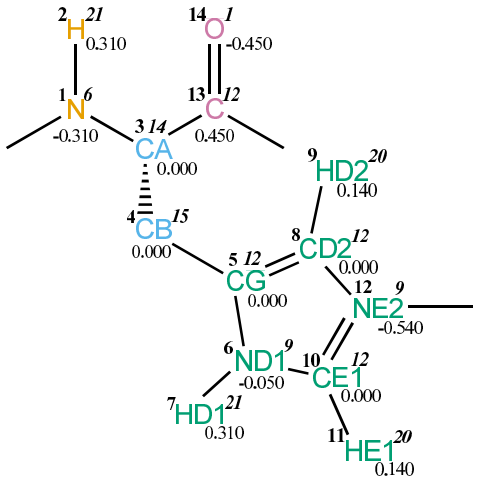


FIGURE 4.37. HIS1 non-bonded parameters.

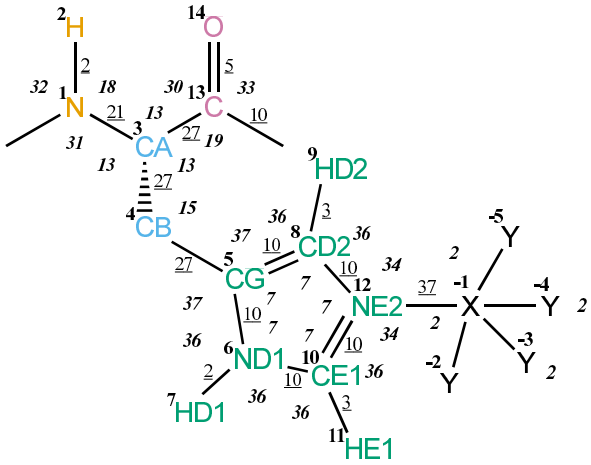


FIGURE 4.38. HIS1 bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 13
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 13 14 15
4	CB	15	4	0.00000	5 6 7 8 9 10 12 13
5	CG	12	12	0.00000	-1 6 7 8 9 10 11 12
6	ND1	9	14	-0.05000	-1 7 8 9 10 11 12
7	HD1	21	1	0.31000	8 10 11 12
8	CD2	12	12	0.00000	-1 9 10 11 12
9	HD2	20	1	0.14000	-1 10 12
10	CE1	12	12	0.00000	-1 11 12
11	HE1	20	1	0.14000	-1 12
12	NE2	9	14	-0.54000	-5 -4 -3 -2 -1
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.91. Atoms of building block HIS1.

I	J	Type
-1	12	37
1	2	2
1	3	21
3	4	27
3	13	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
13	14	5
13	15	10

TABLE 4.92. Bonds of building block HIS1.

I	J	K	Type
-5	-1	12	2
-4	-1	12	2
-3	-1	12	2
-2	-1	12	2
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
-1	12	8	34
-1	12	10	34
8	12	10	7
3	13	14	30
3	13	15	19
14	13	15	33

TABLE 4.93. Bond angles of building block HIS1.

I	J	K	L	Type
-2	-1	1	3	14
-2	-1	12	8	38
-1	1	3	13	43
-1	1	3	13	44
1	3	4	5	34
1	3	13	15	42
1	3	13	15	45
3	4	5	6	40

TABLE 4.94. Dihedral angles of building block HIS1.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
13	3	15	14	1

TABLE 4.95. Improper dihedral angles of building block HIS1.

Solute building block: Histidine (coupled to HEMC at NE2: neutral)
Name: HIS2

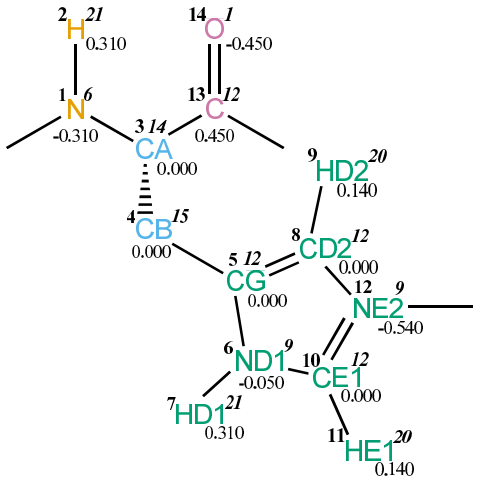


FIGURE 4.39. HIS2 non-bonded parameters.

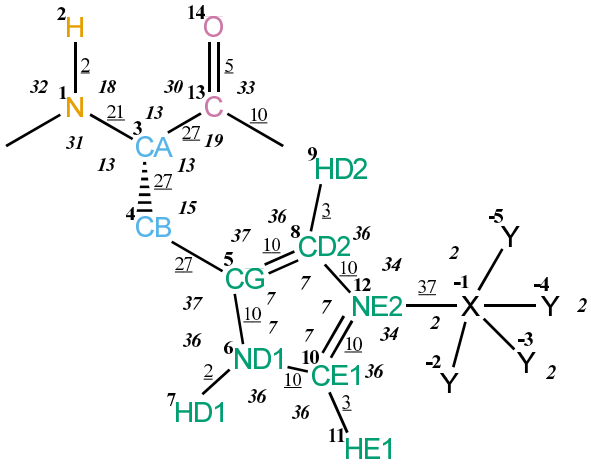


FIGURE 4.40. HIS2 bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 13
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 13 14 15
4	CB	15	4	0.00000	5 6 7 8 9 10 12 13
5	CG	12	12	0.00000	-1 6 7 8 9 10 11 12
6	ND1	9	14	-0.05000	-1 7 8 9 10 11 12
7	HD1	21	1	0.31000	8 10 11 12
8	CD2	12	12	0.00000	-1 9 10 11 12
9	HD2	20	1	0.14000	-1 10 12
10	CE1	12	12	0.00000	-1 11 12
11	HE1	20	1	0.14000	-1 12
12	NE2	9	14	-0.54000	-48 -5 -4 -3 -2 -1
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.96. Atoms of building block HIS2.

I	J	Type
-1	12	37
1	2	2
1	3	21
3	4	27
3	13	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
13	14	5
13	15	10

TABLE 4.97. Bonds of building block HIS2.

I	J	K	Type
-5	-1	12	2
-4	-1	12	2
-3	-1	12	2
-2	-1	12	2
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
-1	12	8	34
-1	12	10	34
8	12	10	7
3	13	14	30
3	13	15	19
14	13	15	33

TABLE 4.98. Bond angles of building block HIS2.

I	J	K	L	Type
-2	-1	1	3	14
-2	-1	12	8	38
-1	1	3	13	43
-1	1	3	13	44
1	3	4	5	34
1	3	13	15	42
1	3	13	15	45
3	4	5	6	40

TABLE 4.99. Dihedral angles of building block HIS2.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
13	3	15	14	1

TABLE 4.100. Improper dihedral angles of building block HIS2.

Solute building block: Hydroxyproline (R-configuration at CG)
Name: HYPR

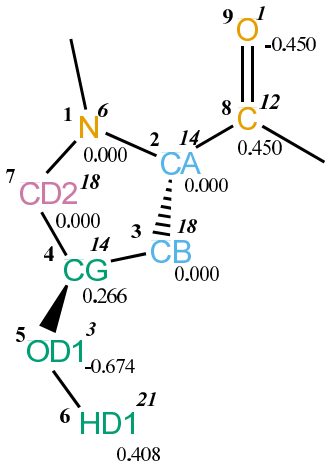


FIGURE 4.41. HYPR non-bonded parameters.

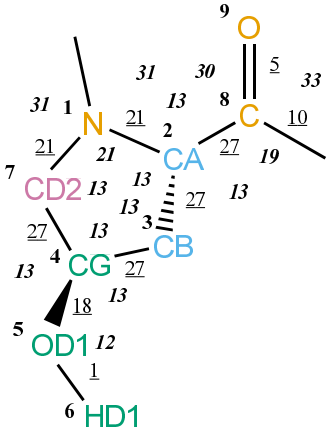


FIGURE 4.42. HYPR bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 7
0					1
1	N	6	14	0.00000	2 3 4 7 8
2	CA	14	3	0.00000	3 4 7 8 9 10
3	CB	18	4	0.00000	4 5 7 8
4	CG	14	3	0.26600	5 6 7
5	OD1	3	16	-0.67400	6 7
6	HD1	21	1	0.40800	
7	CD2	18	4	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.101. Atoms of building block HYPR.

I	J	Type
1	2	21
1	7	21
2	3	27
2	8	27
3	4	27
4	5	18
4	7	27
5	6	1
8	9	5
8	10	10

TABLE 4.102. Bonds of building block HYPR.

I	J	K	Type
-1	1	2	31
-1	1	7	31
2	1	7	21
1	2	3	13
1	2	8	13
3	2	8	13
2	3	4	13
3	4	5	13
3	4	7	13
5	4	7	13
4	5	6	12
1	7	4	13
2	8	9	30
2	8	10	19
9	8	10	33

TABLE 4.103. Bond angles of building block HYPR.

I	J	K	L	Type
-2	-1	1	2	14
-1	1	2	8	43
-1	1	2	8	44
2	1	7	4	39
1	2	3	4	34
1	2	8	10	42
1	2	8	10	45
2	3	4	7	34
3	4	5	6	23
3	4	7	1	34

TABLE 4.104. Dihedral angles of building block HYPR.

I	J	K	L	Type
1	-1	2	7	1
2	1	8	3	2
4	3	7	5	2
8	2	10	9	1

TABLE 4.105. Improper dihedral angles of building block HYPR.

Solute building block: Isoleucine
Name: ILE

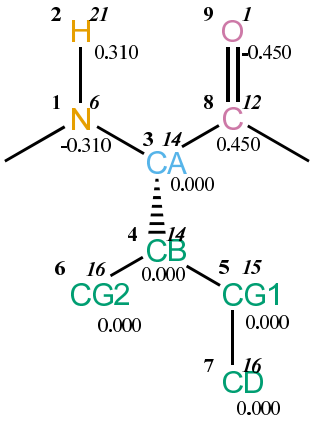


FIGURE 4.43. ILE non-bonded parameters.

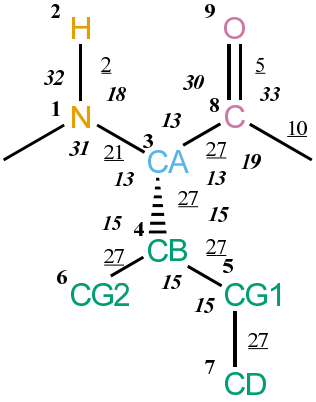


FIGURE 4.44. ILE bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 8 9 10
4	CB	14	3	0.00000	5 6 7 8
5	CG1	15	4	0.00000	6 7
6	CG2	16	5	0.00000	
7	CD	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.106. Atoms of building block ILE.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
4	6	27
5	7	27
8	9	5
8	10	10

TABLE 4.107. Bonds of building block ILE.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
3	4	6	15
5	4	6	15
4	5	7	15
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.108. Bond angles of building block ILE.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	7	34

TABLE 4.109. Dihedral angles of building block ILE.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	5	6	3	2
8	3	10	9	1

TABLE 4.110. Improper dihedral angles of building block ILE.

Solute building block: Leucine
Name: LEU

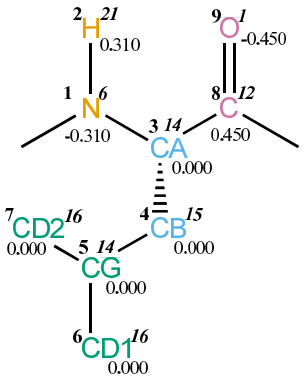


FIGURE 4.45. LEU non-bonded parameters.

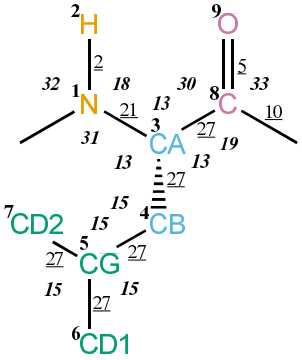


FIGURE 4.46. LEU bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 8 9 10
4	CB	15	4	0.00000	5 6 7 8
5	CG	14	3	0.00000	6 7
6	CD1	16	5	0.00000	7
7	CD2	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.111. Atoms of building block LEU.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	27
5	7	27
8	9	5
8	10	10

TABLE 4.112. Bonds of building block LEU.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	15
4	5	7	15
6	5	7	15
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.113. Bond angles of building block LEU.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	6	34

TABLE 4.114. Dihedral angles of building block LEU.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	6	7	5	2
8	3	10	9	1

TABLE 4.115. Improper dihedral angles of building block LEU.

Solute building block: Lysine (deprotonated; neutral)
Name: LYS

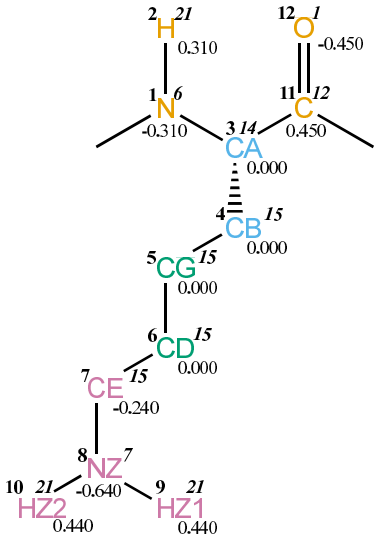


FIGURE 4.47. LYS non-bonded parameters.

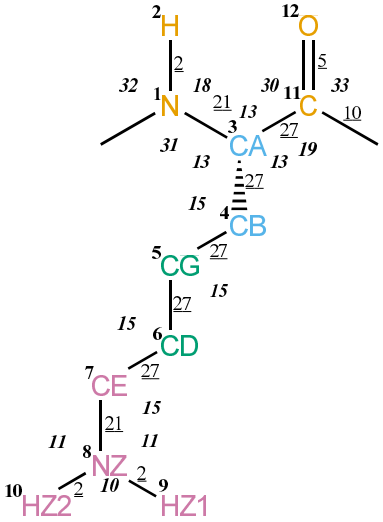


FIGURE 4.48. LYS bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 11
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 11 12 13
4	CB	15	4	0.00000	5 6 11
5	CG	15	4	0.00000	6 7
6	CD	15	4	0.00000	7 8
7	CE	15	4	-0.24000	8 9 10
8	NZ	7	14	-0.64000	9 10
9	HZ1	21	1	0.44000	10
10	HZ2	21	1	0.44000	
11	C	12	12	0.45000	
12	O	1	16	-0.45000	

TABLE 4.116. Atoms of building block LYS.

I	J	Type
1	2	2
1	3	21
3	4	27
3	11	27
4	5	27
5	6	27
6	7	27
7	8	21
8	9	2
8	10	2
11	12	5
11	13	10

TABLE 4.117. Bonds of building block LYS.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	11	13
4	3	11	13
3	4	5	15
4	5	6	15
5	6	7	15
6	7	8	15
7	8	9	11
7	8	10	11
9	8	10	10
3	11	12	30
3	11	13	19
12	11	13	33

TABLE 4.118. Bond angles of building block LYS.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	11	43
-1	1	3	11	44
1	3	4	5	34
1	3	11	13	42
1	3	11	13	45
3	4	5	6	34
4	5	6	7	34
5	6	7	8	34
6	7	8	9	29

TABLE 4.119. Dihedral angles of building block LYS.

I	J	K	L	Type
1	-1	3	2	1
3	1	11	4	2
11	3	13	12	1

TABLE 4.120. Improper dihedral angles of building block LYS.

Solute building block: Lysine (protonated; charge $+e$)
Name: LYSH

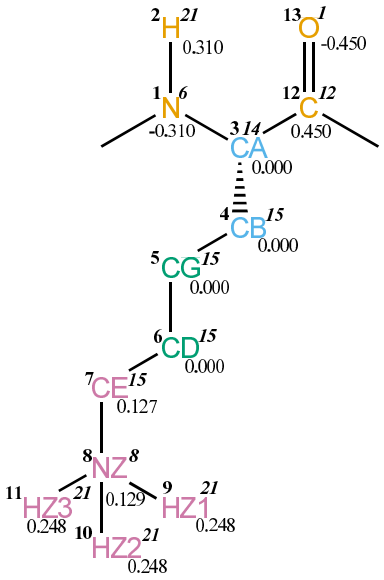


FIGURE 4.49. LYSH non-bonded parameters.

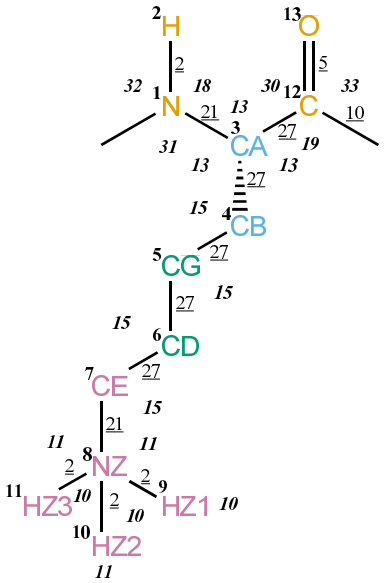


FIGURE 4.50. LYSH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 12
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 12 13 14
4	CB	15	4	0.00000	5 6 12
5	CG	15	4	0.00000	6 7
6	CD	15	4	0.00000	7 8
7	CE	15	4	0.12700	8 9 10 11
8	NZ	8	14	0.12900	9 10 11
9	HZ1	21	1	0.24800	10 11
10	HZ2	21	1	0.24800	11
11	HZ3	21	1	0.24800	
12	C	12	12	0.45000	
13	O	1	16	-0.45000	

TABLE 4.121. Atoms of building block LYSH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	12	27
4	5	27
5	6	27
6	7	27
7	8	21
8	9	2
8	10	2
8	11	2
12	13	5
12	14	10

TABLE 4.122. Bonds of building block LYSH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	12	13
4	3	12	13
3	4	5	15
4	5	6	15
5	6	7	15
6	7	8	15
7	8	9	11
7	8	10	11
7	8	11	11
9	8	10	10
9	8	11	10
10	8	11	10
3	12	13	30
3	12	14	19
13	12	14	33

TABLE 4.123. Bond angles of building block LYSH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	12	43
-1	1	3	12	44
1	3	4	5	34
1	3	12	14	42
1	3	12	14	45
3	4	5	6	34
4	5	6	7	34
5	6	7	8	34
6	7	8	9	29

TABLE 4.124. Dihedral angles of building block LYSH.

I	J	K	L	Type
1	-1	3	2	1
3	1	12	4	2
12	3	14	13	1

TABLE 4.125. Improper dihedral angles of building block LYSH.

Solute building block: Methionine
Name: MET

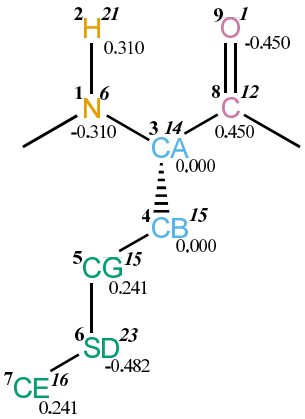


FIGURE 4.51. MET non-bonded parameters.

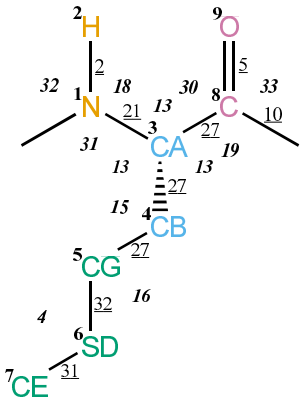


FIGURE 4.52. MET bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 8 9 10
4	CB	15	4	0.00000	5 6 8
5	CG	15	4	0.24100	6 7
6	SD	23	32	-0.48200	7
7	CE	16	5	0.24100	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.126. Atoms of building block MET.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	32
6	7	31
8	9	5
8	10	10

TABLE 4.127. Bonds of building block MET.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	16
5	6	7	4
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.128. Bond angles of building block MET.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	6	34
4	5	6	7	26

TABLE 4.129. Dihedral angles of building block MET.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
8	3	10	9	1

TABLE 4.130. Improper dihedral angles of building block MET.

Solute building block: Phenylalanine
Name: PHE

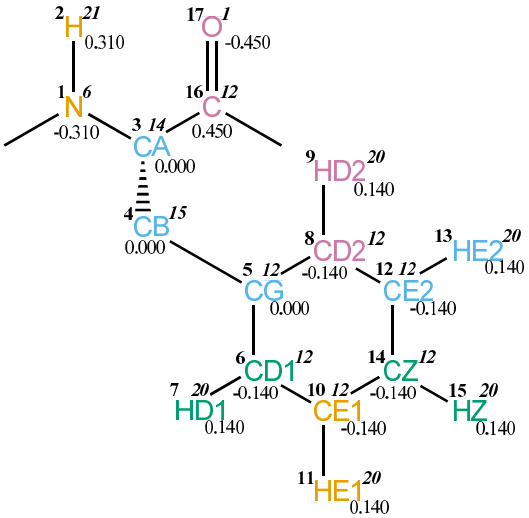


FIGURE 4.53. PHE non-bonded parameters.

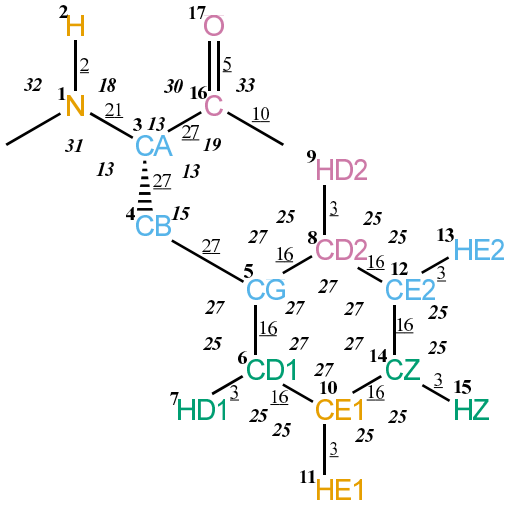


FIGURE 4.54. PHE bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 16
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 16 17 18
4	CB	15	4	0.00000	5 6 7 8 9 10 12 16
5	CG	12	12	0.00000	6 7 8 9 10 11 12 13 14
6	CD1	12	12	-0.14000	7 8 9 10 11 12 14 15
7	HD1	20	1	0.14000	8 10 11 14
8	CD2	12	12	-0.14000	9 10 12 13 14 15
9	HD2	20	1	0.14000	12 13 14
10	CE1	12	12	-0.14000	11 12 13 14 15
11	HE1	20	1	0.14000	12 14 15
12	CE2	12	12	-0.14000	13 14 15
13	HE2	20	1	0.14000	14 15
14	CZ	12	12	-0.14000	15
15	HZ	20	1	0.14000	
16	C	12	12	0.45000	
17	O	1	16	-0.45000	

TABLE 4.131. Atoms of building block PHE.

I	J	Type
1	2	2
1	3	21
3	4	27
3	16	27
4	5	27
5	6	16
5	8	16
6	7	3
6	10	16
8	9	3
8	12	16
10	11	3
10	14	16
12	13	3
12	14	16
14	15	3
16	17	5
16	18	10

TABLE 4.132. Bonds of building block PHE.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	16	13
4	3	16	13
3	4	5	15
4	5	6	27
4	5	8	27
6	5	8	27
5	6	7	25
5	6	10	27
7	6	10	25
5	8	9	25
5	8	12	27
9	8	12	25
6	10	11	25
6	10	14	27
11	10	14	25
8	12	13	25
8	12	14	27
13	12	14	25
10	14	12	27
10	14	15	25
12	14	15	25
3	16	17	30
3	16	18	19
17	16	18	33

TABLE 4.133. Bond angles of building block PHE.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	16	43
-1	1	3	16	44
1	3	4	5	34
1	3	16	18	42
1	3	16	18	45
3	4	5	6	40

TABLE 4.134. Dihedral angles of building block PHE.

I	J	K	L	Type
1	-1	3	2	1
3	1	16	4	2
5	6	8	4	1
5	6	10	14	1
5	8	12	14	1
6	5	8	12	1
6	5	10	7	1
6	10	14	12	1
8	5	6	10	1
8	5	12	9	1
8	12	14	10	1
11	6	14	10	1
13	8	14	12	1
14	10	12	15	1
16	3	18	17	1

TABLE 4.135. Improper dihedral angles of building block PHE.

Solute building block: Proline
Name: PRO

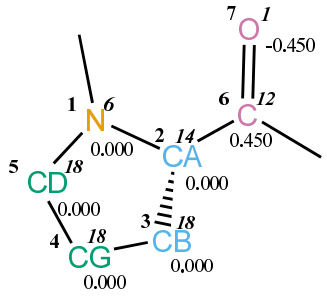


FIGURE 4.55. PRO non-bonded parameters.

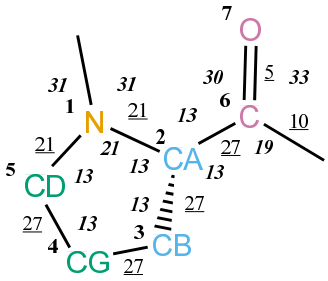


FIGURE 4.56. PRO bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 5
0					1
1	N	6	14	0.00000	2 3 4 5 6
2	CA	14	3	0.00000	3 4 5 6 7 8
3	CB	18	4	0.00000	4 5 6
4	CG	18	4	0.00000	5
5	CD	18	4	0.00000	
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.136. Atoms of building block PRO.

I	J	Type
1	2	21
1	5	21
2	3	27
2	6	27
3	4	27
4	5	27
6	7	5
6	8	10

TABLE 4.137. Bonds of building block PRO.

I	J	K	Type
-1	1	2	31
-1	1	5	31
2	1	5	21
1	2	3	13
1	2	6	13
3	2	6	13
2	3	4	13
3	4	5	13
1	5	4	13
2	6	7	30
2	6	8	19
7	6	8	33

TABLE 4.138. Bond angles of building block PRO.

I	J	K	L	Type
-2	-1	1	2	14
-1	1	2	6	43
-1	1	2	6	44
2	1	5	4	39
1	2	3	4	34
1	2	6	8	42
1	2	6	8	45
2	3	4	5	34
3	4	5	1	34

TABLE 4.139. Dihedral angles of building block PRO.

I	J	K	L	Type
1	-1	2	5	1
2	1	6	3	2
6	2	8	7	1

TABLE 4.140. Improper dihedral angles of building block PRO.

Solute building block: Serine
Name: SER

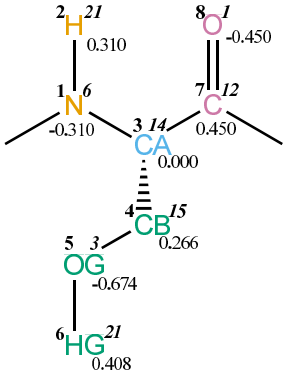


FIGURE 4.57. SER non-bonded parameters.

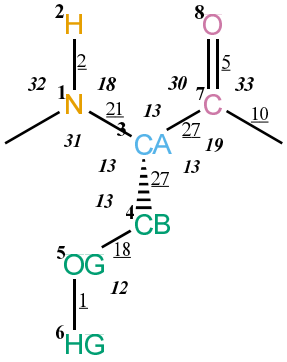


FIGURE 4.58. SER bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 7 8 9
4	CB	15	4	0.26600	5 6 7
5	OG	3	16	-0.67400	6
6	HG	21	1	0.40800	
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.141. Atoms of building block SER.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	18
5	6	1
7	8	5
7	9	10

TABLE 4.142. Bonds of building block SER.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	13
4	5	6	12
3	7	8	30
3	7	9	19
8	7	9	33

TABLE 4.143. Bond angles of building block SER.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	9	42
1	3	7	9	45
3	4	5	6	23

TABLE 4.144. Dihedral angles of building block SER.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
7	3	9	8	1

TABLE 4.145. Improper dihedral angles of building block SER.

Solute building block: Threonine
Name: THR

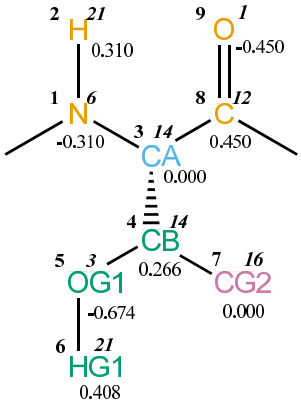


FIGURE 4.59. THR non-bonded parameters.

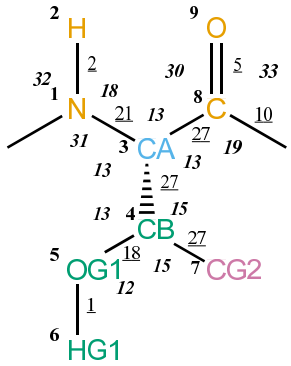


FIGURE 4.60. THR bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 7 8 9 10
4	CB	14	3	0.26600	5 6 7 8
5	OG1	3	16	-0.67400	6 7
6	HG1	21	1	0.40800	
7	CG2	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.146. Atoms of building block THR.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	18
4	7	27
5	6	1
8	9	5
8	10	10

TABLE 4.147. Bonds of building block THR.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	13
3	4	7	15
5	4	7	15
4	5	6	12
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.148. Bond angles of building block THR.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	6	23

TABLE 4.149. Dihedral angles of building block THR.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	5	7	3	2
8	3	10	9	1

TABLE 4.150. Improper dihedral angles of building block THR.

Solute building block: Tryptophan
Name: TRP

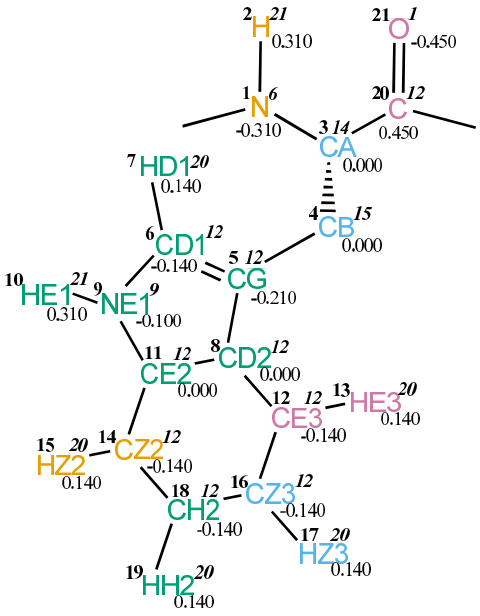


FIGURE 4.61. TRP non-bonded parameters.

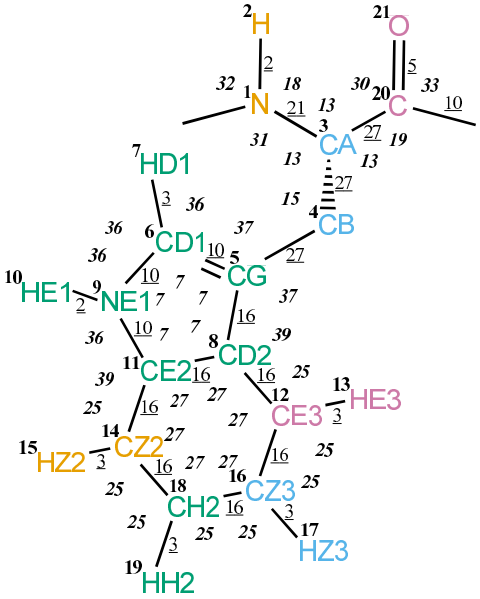


FIGURE 4.62. TRP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 20
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 20 21 22
4	CB	15	4	0.00000	5 6 7 8 9 11 12 20
5	CG	12	12	-0.21000	6 7 8 9 10 11 12 13 14 16
6	CD1	12	12	-0.14000	7 8 9 10 11 12 14
7	HD1	20	1	0.14000	8 9 10 11
8	CD2	12	12	0.00000	9 10 11 12 13 14 15 16 17 18
9	NE1	9	14	-0.10000	10 11 12 14 15 18
10	HE1	21	1	0.31000	11 14
11	CE2	12	12	0.00000	12 13 14 15 16 18 19
12	CE3	12	12	-0.14000	13 14 16 17 18 19
13	HE3	20	1	0.14000	16 17 18
14	CZ2	12	12	-0.14000	15 16 17 18 19
15	HZ2	20	1	0.14000	16 18 19
16	CZ3	12	12	-0.14000	17 18 19
17	HZ3	20	1	0.14000	18 19
18	CH2	12	12	-0.14000	19
19	HH2	20	1	0.14000	
20	C	12	12	0.45000	
21	O	1	16	-0.45000	

TABLE 4.151. Atoms of building block TRP.

I	J	Type
1	2	2
1	3	21
3	4	27
3	20	27
4	5	27
5	6	10
5	8	16
6	7	3
6	9	10
8	11	16
8	12	16
9	10	2
9	11	10
11	14	16
12	13	3
12	16	16
14	15	3
14	18	16
16	17	3
16	18	16
18	19	3
20	21	5
20	22	10

TABLE 4.152. Bonds of building block TRP.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	20	13
4	3	20	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	9	7
7	6	9	36
5	8	11	7
5	8	12	39
11	8	12	27
6	9	10	36
6	9	11	7
10	9	11	36
8	11	9	7
8	11	14	27
9	11	14	39
8	12	13	25
8	12	16	27
13	12	16	25
11	14	15	25
11	14	18	27
15	14	18	25
12	16	17	25
12	16	18	27
17	16	18	25
14	18	16	27
14	18	19	25
16	18	19	25
3	20	21	30
3	20	22	19
21	20	22	33

TABLE 4.153. Bond angles of building block TRP.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	20	43
-1	1	3	20	44
1	3	4	5	34
1	3	20	22	42
1	3	20	22	45
3	4	5	8	40

TABLE 4.154. Dihedral angles of building block TRP.

I	J	K	L	Type
1	-1	3	2	1
3	1	20	4	2
5	6	8	4	1
5	6	9	11	1
5	8	11	9	1
6	5	8	11	1
6	5	9	7	1
6	9	11	8	1
8	5	6	9	1
8	11	12	5	1
8	11	14	18	1
8	12	16	18	1
9	6	11	10	1
11	8	12	16	1
11	8	14	9	1
11	14	18	16	1
12	8	11	14	1
12	8	16	13	1
12	16	18	14	1
14	11	18	15	1
16	12	18	17	1
18	14	16	19	1
20	3	22	21	1

TABLE 4.155. Improper dihedral angles of building block TRP.

Solute building block: Tyrosine
Name: TYR

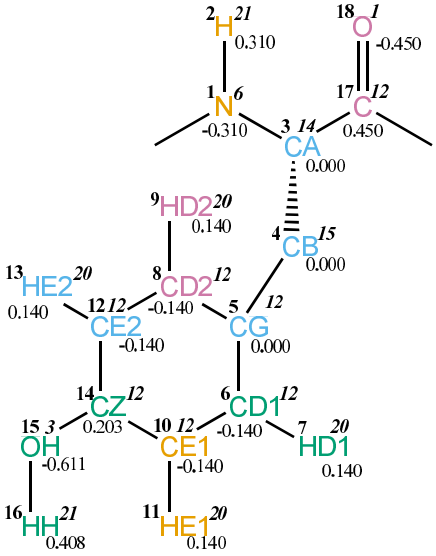


FIGURE 4.63. TYR non-bonded parameters.

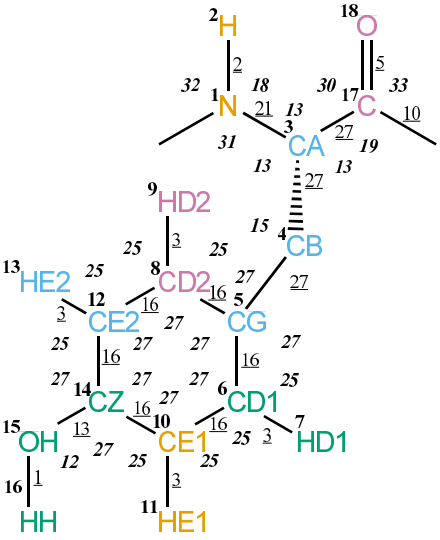


FIGURE 4.64. TYR bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 17
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 17 18 19
4	CB	15	4	0.00000	5 6 7 8 9 10 12 17
5	CG	12	12	0.00000	6 7 8 9 10 11 12 13 14
6	CD1	12	12	-0.14000	7 8 9 10 11 12 14 15
7	HD1	20	1	0.14000	8 10 11 14
8	CD2	12	12	-0.14000	9 10 12 13 14 15
9	HD2	20	1	0.14000	12 13 14
10	CE1	12	12	-0.14000	11 12 13 14 15
11	HE1	20	1	0.14000	12 14 15
12	CE2	12	12	-0.14000	13 14 15
13	HE2	20	1	0.14000	14 15
14	CZ	12	12	0.20300	15 16
15	OH	3	16	-0.61100	16
16	HH	21	1	0.40800	
17	C	12	12	0.45000	
18	O	1	16	-0.45000	

TABLE 4.156. Atoms of building block TYR.

I	J	Type
1	2	2
1	3	21
3	4	27
3	17	27
4	5	27
5	6	16
5	8	16
6	7	3
6	10	16
8	9	3
8	12	16
10	11	3
10	14	16
12	13	3
12	14	16
14	15	13
15	16	1
17	18	5
17	19	10

TABLE 4.157. Bonds of building block TYR.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	17	13
4	3	17	13
3	4	5	15
4	5	6	27
4	5	8	27
6	5	8	27
5	6	7	25
5	6	10	27
7	6	10	25
5	8	9	25
5	8	12	27
9	8	12	25
6	10	11	25
6	10	14	27
11	10	14	25
8	12	13	25
8	12	14	27
13	12	14	25
10	14	12	27
10	14	15	27
12	14	15	27
14	15	16	12
3	17	18	30
3	17	19	19
18	17	19	33

TABLE 4.158. Bond angles of building block TYR.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	17	43
-1	1	3	17	44
1	3	4	5	34
1	3	17	19	42
1	3	17	19	45
3	4	5	6	40
10	14	15	16	11

TABLE 4.159. Dihedral angles of building block TYR.

I	J	K	L	Type
1	-1	3	2	1
3	1	17	4	2
5	6	8	4	1
5	6	10	14	1
5	8	12	14	1
6	5	8	12	1
6	5	10	7	1
6	10	14	12	1
8	5	6	10	1
8	5	12	9	1
8	12	14	10	1
11	6	14	10	1
13	8	14	12	1
14	10	12	15	1
17	3	19	18	1

TABLE 4.160. Improper dihedral angles of building block TYR.

Solute building block: Valine
Name: VAL

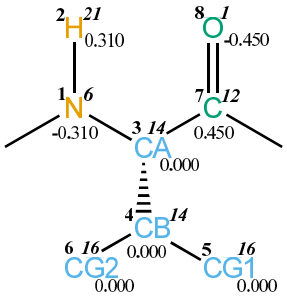


FIGURE 4.65. VAL non-bonded parameters.

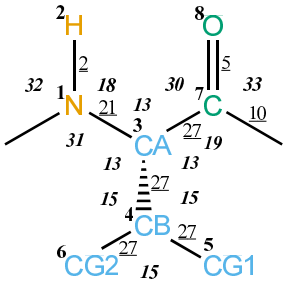


FIGURE 4.66. VAL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8 9
4	CB	14	3	0.00000	5 6 7
5	CG1	16	5	0.00000	6
6	CG2	16	5	0.00000	
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.161. Atoms of building block VAL.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	27
4	6	27
7	8	5
7	9	10

TABLE 4.162. Bonds of building block VAL.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	15
3	4	6	15
5	4	6	15
3	7	8	30
3	7	9	19
8	7	9	33

TABLE 4.163. Bond angles of building block VAL.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	9	42
1	3	7	9	45

TABLE 4.164. Dihedral angles of building block VAL.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
3	5	6	4	2
7	3	9	8	1

TABLE 4.165. Improper dihedral angles of building block VAL.

Solute building block: D-ALanine
Name: DALA

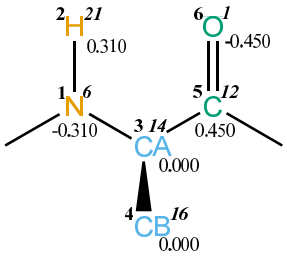


FIGURE 4.67. DALA non-bonded parameters.

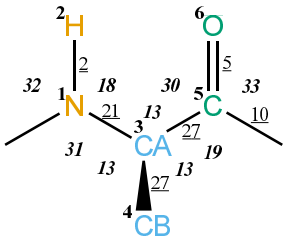


FIGURE 4.68. DALA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7
4	CB	16	5	0.00000	5
5	C	12	12	0.45000	
6	O	1	16	-0.45000	

TABLE 4.166. Atoms of building block DALA.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
5	6	5
5	7	10

TABLE 4.167. Bonds of building block DALA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
4	3	5	13
3	5	6	30
3	5	7	19
6	5	7	33

TABLE 4.168. Bond angles of building block DALA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	5	43
-1	1	3	5	44
1	3	5	7	42
1	3	5	7	45

TABLE 4.169. Dihedral angles of building block DALA.

I	J	K	L	Type
1	-1	3	2	1
4	1	5	3	2
5	3	7	6	1

TABLE 4.170. Improper dihedral angles of building block DALA.

Solute building block: L-2-amino-butanoic acid
Name: ABU

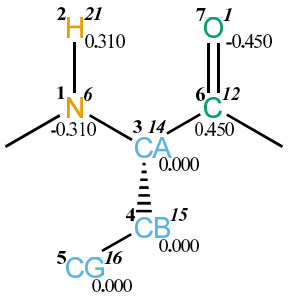


FIGURE 4.69. ABU non-bonded parameters.

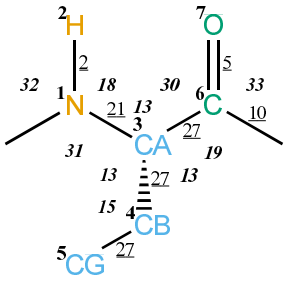


FIGURE 4.70. ABU bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 6
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8
4	CB	15	4	0.00000	5 6
5	CG	16	5	0.00000	
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.171. Atoms of building block ABU.

I	J	Type
1	2	2
1	3	21
3	4	27
3	6	27
4	5	27
6	7	5
6	8	10

TABLE 4.172. Bonds of building block ABU.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	6	13
4	3	6	13
3	4	5	15
3	6	7	30
3	6	8	19
7	6	8	33

TABLE 4.173. Bond angles of building block ABU.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	6	43
-1	1	3	6	44
1	3	4	5	34
1	3	6	8	42
1	3	6	8	45

TABLE 4.174. Dihedral angles of building block ABU.

I	J	K	L	Type
1	-1	3	2	1
3	1	6	4	2
6	3	8	7	1

TABLE 4.175. Improper dihedral angles of building block ABU.

Solute building block: 2-aminoisobutyric acid
Name: AIB

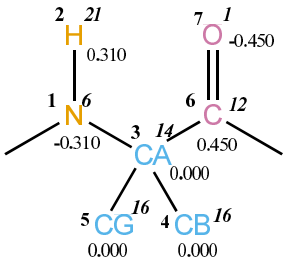


FIGURE 4.71. AIB non-bonded parameters.

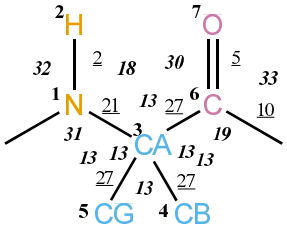


FIGURE 4.72. AIB bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5 6
2	H	21	1	0.31000	3
3	CA	14	3	0.00000	4 5 6 7 8
4	CB	16	5	0.00000	5 6
5	CG	16	5	0.00000	6
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.176. Atoms of building block AIB.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
3	6	27
6	7	5
6	8	10

TABLE 4.177. Bonds of building block AIB.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
1	3	6	13
4	3	5	13
4	3	6	13
5	3	6	13
3	6	7	30
3	6	8	19
7	6	8	33

TABLE 4.178. Bond angles of building block AIB.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	6	43
-1	1	3	6	44
1	3	6	8	42
1	3	6	8	45

TABLE 4.179. Dihedral angles of building block AIB.

I	J	K	L	Type
1	-1	3	2	1
6	3	8	7	1

TABLE 4.180. Improper dihedral angles of building block AIB.

Solute building block: (4R)-4-[(E)-2-butanyl]-4, N-dimethyl-L-threonine
Name: MEBMT

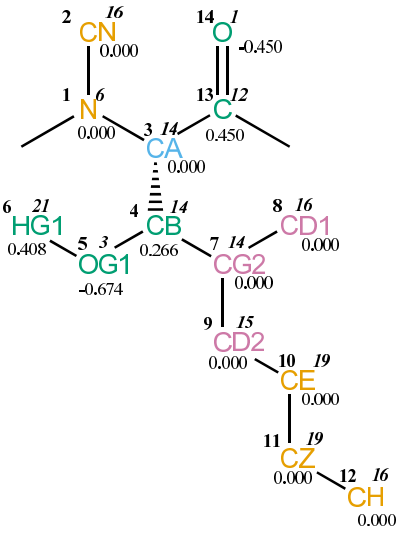


FIGURE 4.73. MEBMT non-bonded parameters.

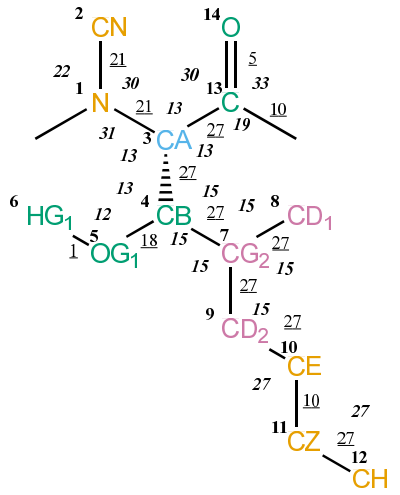


FIGURE 4.74. MEBMT bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	0.00000	2 3 4 13
2	CN	16	5	0.00000	3
3	CA	14	3	0.00000	4 5 7 13 14 15
4	CB	14	3	0.26600	5 6 7 8 9 13
5	OG1	3	16	-0.67400	6 7
6	HG1	21	1	0.40800	
7	CG2	14	3	0.00000	8 9 10
8	CD1	16	5	0.00000	9
9	CD2	15	4	0.00000	10 11
10	CE	19	3	0.00000	11 12
11	CZ	19	3	0.00000	12
12	CH	16	5	0.00000	
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.181. Atoms of building block MEBMT.

I	J	Type
1	2	21
1	3	21
3	4	27
3	13	27
4	5	18
4	7	27
5	6	1
7	8	27
7	9	27
9	10	27
10	11	10
11	12	27
13	14	5
13	15	10

TABLE 4.182. Bonds of building block MEBMT.

I	J	K	Type
-1	1	2	22
-1	1	3	31
2	1	3	30
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	13
3	4	7	15
5	4	7	15
4	5	6	12
4	7	8	15
4	7	9	15
8	7	9	15
7	9	10	15
9	10	11	27
10	11	12	27
3	13	14	30
3	13	15	19
14	13	15	33

TABLE 4.183. Bond angles of building block MEBMT.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	13	43
-1	1	3	13	44
1	3	4	7	34
1	3	13	15	42
1	3	13	15	45
3	4	5	6	23
3	4	7	9	34
4	7	9	10	34
7	9	10	11	40
9	10	11	12	14

TABLE 4.184. Dihedral angles of building block MEBMT.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
4	5	7	3	2
7	8	9	4	2
13	3	15	14	1

TABLE 4.185. Improper dihedral angles of building block MEBMT.

Solute building block: N-methyl-L-leucine
Name: MELEU

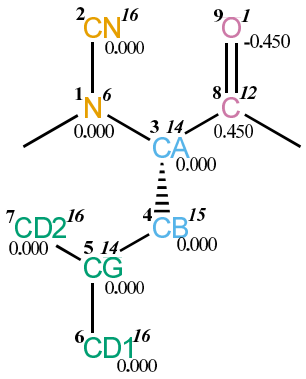


FIGURE 4.75. MELEU non-bonded parameters.

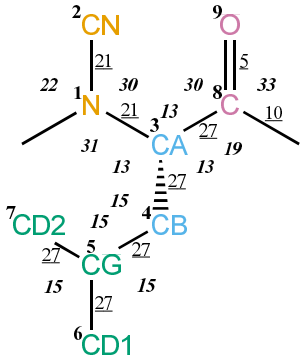


FIGURE 4.76. MELEU bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	0.00000	2 3 4 8
2	CN	16	5	0.00000	3
3	CA	14	3	0.00000	4 5 8 9 10
4	CB	15	4	0.00000	5 6 7 8
5	CG	14	3	0.00000	6 7
6	CD1	16	5	0.00000	7
7	CD2	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.186. Atoms of building block MELEU.

I	J	Type
1	2	21
1	3	21
3	4	27
3	8	27
4	5	27
5	6	27
5	7	27
8	9	5
8	10	10

TABLE 4.187. Bonds of building block MELEU.

I	J	K	Type
-1	1	2	22
-1	1	3	31
2	1	3	30
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	15
4	5	7	15
6	5	7	15
3	8	9	30
3	8	10	19
9	8	10	33

TABLE 4.188. Bond angles of building block MELEU.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	42
1	3	8	10	45
3	4	5	6	34

TABLE 4.189. Dihedral angles of building block MELEU.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	6	7	5	2
8	3	10	9	1

TABLE 4.190. Improper dihedral angles of building block MELEU.

Solute building block: N-methyl-L-valine
Name: MEVAL

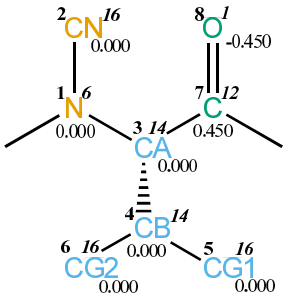


FIGURE 4.77. MEVAL non-bonded parameters.

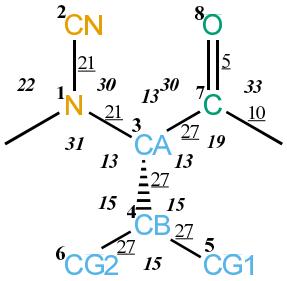


FIGURE 4.78. MEVAL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	0.00000	2 3 4 7
2	CN	16	5	0.00000	3
3	CA	14	3	0.00000	4 5 6 7 8 9
4	CB	14	3	0.00000	5 6 7
5	CG1	16	5	0.00000	6
6	CG2	16	5	0.00000	
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.191. Atoms of building block MEVAL.

I	J	Type
1	2	21
1	3	21
3	4	27
3	7	27
4	5	27
4	6	27
7	8	5
7	9	10

TABLE 4.192. Bonds of building block MEVAL.

I	J	K	Type
-1	1	2	22
-1	1	3	31
2	1	3	30
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	15
3	4	6	15
5	4	6	15
3	7	8	30
3	7	9	19
8	7	9	33

TABLE 4.193. Bond angles of building block MEVAL.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	9	42
1	3	7	9	45

TABLE 4.194. Dihedral angles of building block MEVAL.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
3	5	6	4	2
7	3	9	8	1

TABLE 4.195. Improper dihedral angles of building block MEVAL.

Solute building block: Sarcosine or N-methylglycine
Name: SAR

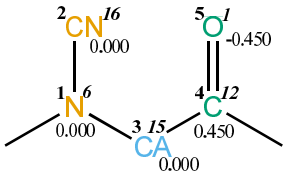


FIGURE 4.79. SAR non-bonded parameters.

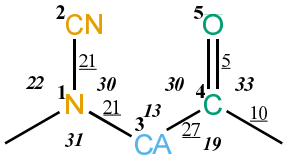


FIGURE 4.80. SAR bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	0.00000	2 3 4
2	CN	16	5	0.00000	3
3	CA	15	4	0.00000	4 5 6
4	C	12	12	0.45000	
5	O	1	16	-0.45000	

TABLE 4.196. Atoms of building block SAR.

I	J	Type
1	2	21
1	3	21
3	4	27
4	5	5
4	6	10

TABLE 4.197. Bonds of building block SAR.

I	J	K	Type
-1	1	2	22
-1	1	3	31
2	1	3	30
1	3	4	13
3	4	5	30
3	4	6	19
5	4	6	33

TABLE 4.198. Bond angles of building block SAR.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	6	42
1	3	4	6	45

TABLE 4.199. Dihedral angles of building block SAR.

I	J	K	L	Type
1	-1	3	2	1
4	3	6	5	1

TABLE 4.200. Improper dihedral angles of building block SAR.

4.4. β-amino acids

Solute building block: (R)-β^2-Phenylalanine
Name: RAF

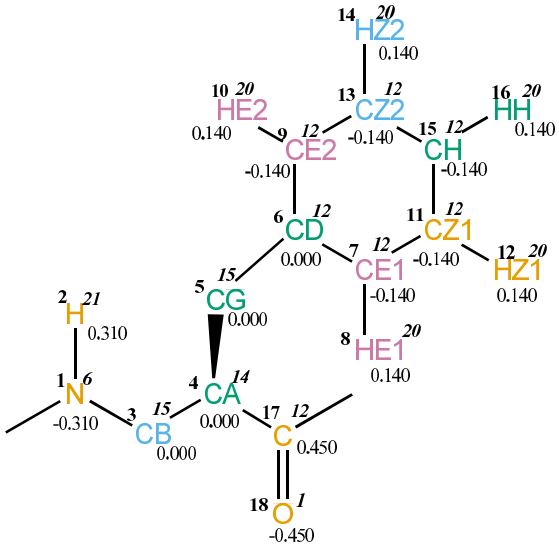


FIGURE 4.81. RAF non-bonded parameters.

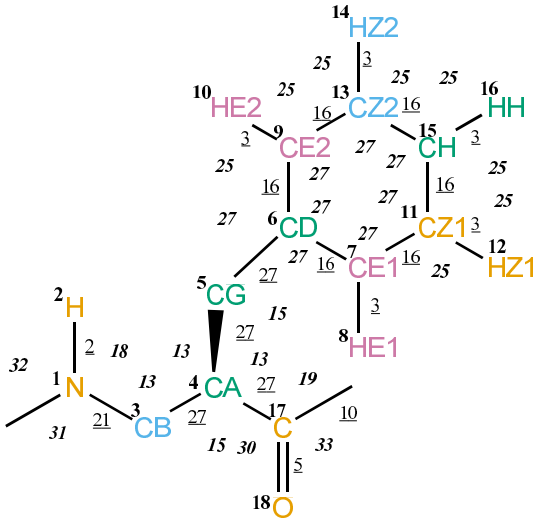


FIGURE 4.82. RAF bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 17
4	CA	14	3	0.00000	5 6 17 18 19
5	CG	15	4	0.00000	6 7 8 9 10 11 13 17
6	CD	12	12	0.00000	7 8 9 10 11 12 13 14 15
7	CE1	12	12	-0.14000	8 9 10 11 12 13 15 16
8	HE1	20	1	0.14000	9 11 12 15
9	CE2	12	12	-0.14000	10 11 13 14 15 16
10	HE2	20	1	0.14000	13 14 15
11	CZ1	12	12	-0.14000	12 13 14 15 16
12	HZ1	20	1	0.14000	13 15 16
13	CZ2	12	12	-0.14000	14 15 16
14	HZ2	20	1	0.14000	15 16
15	CH	12	12	-0.14000	16
16	HH	20	1	0.14000	
17	C	12	12	0.45000	
18	O	1	16	-0.45000	

TABLE 4.201. Atoms of building block RAF.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	17	27
5	6	27
6	7	16
6	9	16
7	8	3
7	11	16
9	10	3
9	13	16
11	12	3
11	15	16
13	14	3
13	15	16
15	16	3
17	18	5
17	19	10

TABLE 4.202. Bonds of building block RAF.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	13
3	4	17	15
5	4	17	13
4	5	6	15
5	6	7	27
5	6	9	27
7	6	9	27
6	7	8	25
6	7	11	27
8	7	11	25
6	9	10	25
6	9	13	27
10	9	13	25
7	11	12	25
7	11	15	27
12	11	15	25
9	13	14	25
9	13	15	27
14	13	15	25
11	15	13	27
11	15	16	25
13	15	16	25
4	17	18	30
4	17	19	19
18	17	19	33

TABLE 4.203. Bond angles of building block RAF.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	17	34
3	4	5	6	34
3	4	17	19	42
3	4	17	19	45
4	5	6	7	40

TABLE 4.204. Dihedral angles of building block RAF.

I	J	K	L	Type
1	-1	3	2	1
4	3	17	5	2
6	7	9	5	1
6	7	11	15	1
6	9	13	15	1
7	6	9	13	1
7	6	11	8	1
7	11	15	13	1
9	6	7	11	1
9	6	13	10	1
9	13	15	11	1
12	7	15	11	1
14	9	15	13	1
15	11	13	16	1
17	4	19	18	1

TABLE 4.205. Improper dihedral angles of building block RAF.

Solute building block: (R)-β^2-Valine
Name: RAV

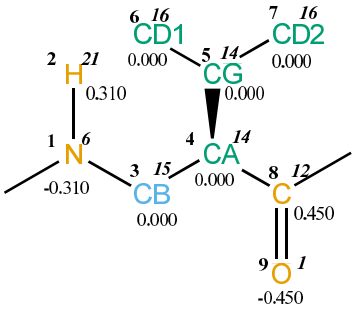


FIGURE 4.83. RAV non-bonded parameters.

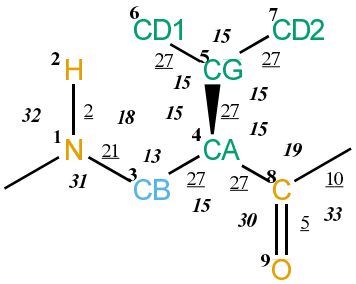


FIGURE 4.84. RAV bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 8
4	CA	14	3	0.00000	5 6 7 8 9 10
5	CG	14	3	0.00000	6 7 8
6	CD1	16	5	0.00000	7
7	CD2	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.206. Atoms of building block RAV.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	8	27
5	6	27
5	7	27
8	9	5
8	10	10

TABLE 4.207. Bonds of building block RAV.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	8	15
5	4	8	15
4	5	6	15
4	5	7	15
6	5	7	15
4	8	9	30
4	8	10	19
9	8	10	33

TABLE 4.208. Bond angles of building block RAV.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	8	34
3	4	5	6	34
3	4	8	10	42
3	4	8	10	45

TABLE 4.209. Dihedral angles of building block RAV.

I	J	K	L	Type
1	-1	3	2	1
4	3	8	5	2
4	6	7	5	2
8	4	10	9	1

TABLE 4.210. Improper dihedral angles of building block RAV.

Solute building block: (R)-β^3-Cysteine (protonated; neutral)
Name: RBCH

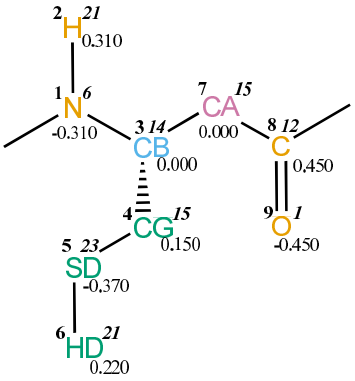


FIGURE 4.85. RBCH non-bonded parameters.

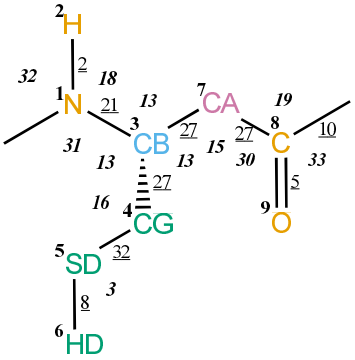


FIGURE 4.86. RBCH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8
4	CG	15	4	0.15000	5 6 7
5	SD	23	32	-0.37000	6
6	HD	21	1	0.22000	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.211. Atoms of building block RBCH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	32
5	6	8
7	8	27
8	9	5
8	10	10

TABLE 4.212. Bonds of building block RBCH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	16
4	5	6	3
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.213. Bond angles of building block RBCH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	4	5	6	26
3	7	8	10	42
3	7	8	10	45

TABLE 4.214. Dihedral angles of building block RBCH.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
8	7	10	9	1

TABLE 4.215. Improper dihedral angles of building block RBCH.

Solute building block: (R)-β^3-Isoleucine
Name: RBI

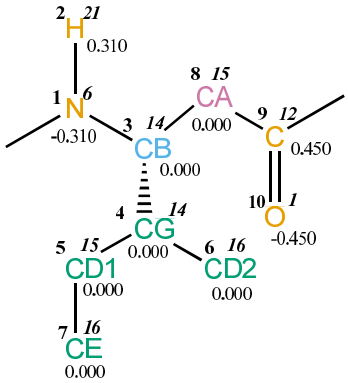


FIGURE 4.87. RBI non-bonded parameters.

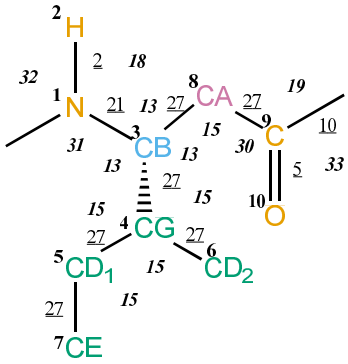


FIGURE 4.88. RBI bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 8 9
4	CG	14	3	0.00000	5 6 7 8
5	CD1	15	4	0.00000	6 7
6	CD2	16	5	0.00000	
7	CE	16	5	0.00000	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.216. Atoms of building block RBI.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
4	6	27
5	7	27
8	9	27
9	10	5
9	11	10

TABLE 4.217. Bonds of building block RBI.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
3	4	6	15
5	4	6	15
4	5	7	15
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.218. Bond angles of building block RBI.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	7	34
3	8	9	11	42
3	8	9	11	45

TABLE 4.219. Dihedral angles of building block RBI.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	5	6	3	2
9	8	11	10	1

TABLE 4.220. Improper dihedral angles of building block RBI.

Solute building block: (R)-β^3-Lysine (protonated; charge $+e$)
Name: RBKH

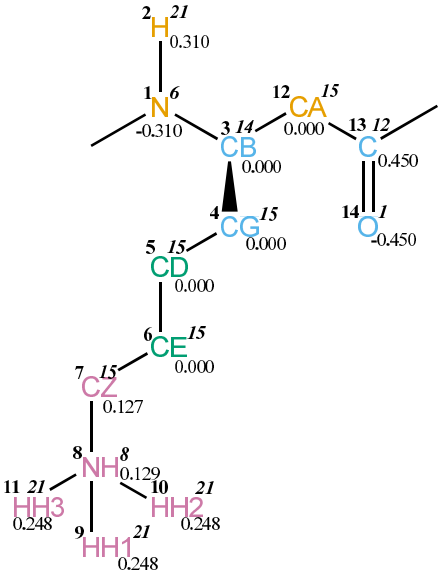


FIGURE 4.89. RBKH non-bonded parameters.

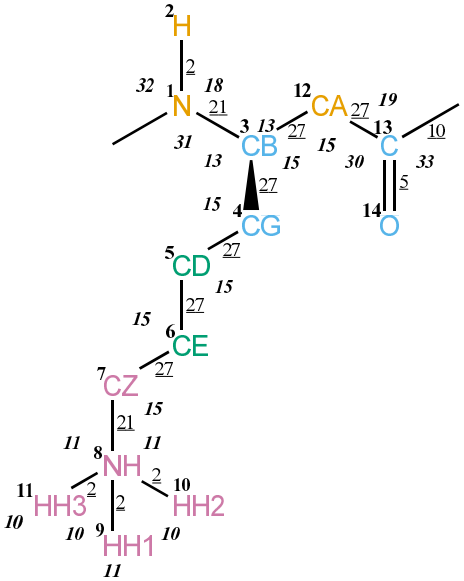


FIGURE 4.90. RBKH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 12
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 12 13
4	CG	15	4	0.00000	5 6 12
5	CD	15	4	0.00000	6 7
6	CE	15	4	0.00000	7 8
7	CZ	15	4	0.12700	8 9 10 11
8	NH	8	14	0.12900	9 10 11
9	HH1	21	1	0.24800	10 11
10	HH2	21	1	0.24800	11
11	HH3	21	1	0.24800	
12	CA	15	4	0.00000	13 14 15
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.221. Atoms of building block RBKH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	12	27
4	5	27
5	6	27
6	7	27
7	8	21
8	9	2
8	10	2
8	11	2
12	13	27
13	14	5
13	15	10

TABLE 4.222. Bonds of building block RBKH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	12	13
4	3	12	15
3	4	5	15
4	5	6	15
5	6	7	15
6	7	8	15
7	8	9	11
7	8	10	11
7	8	11	11
9	8	10	10
9	8	11	10
10	8	11	10
3	12	13	15
12	13	14	30
12	13	15	19
14	13	15	33

TABLE 4.223. Bond angles of building block RBKH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	12	43
-1	1	3	12	44
1	3	4	5	34
1	3	12	13	34
3	4	5	6	34
4	5	6	7	34
5	6	7	8	34
6	7	8	9	29
3	12	13	15	42
3	12	13	15	45

TABLE 4.224. Dihedral angles of building block RBKH.

I	J	K	L	Type
1	-1	3	2	1
4	1	12	3	2
13	12	15	14	1

TABLE 4.225. Improper dihedral angles of building block RBKH.

Solute building block: (R)-β^3-Methionine
Name: RBM

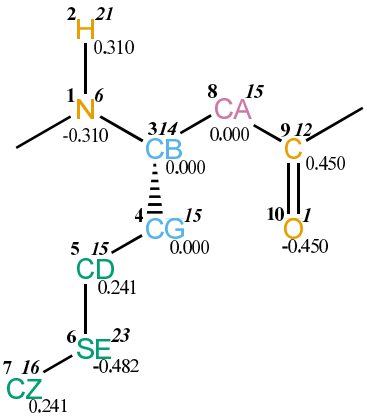


FIGURE 4.91. RBM non-bonded parameters.

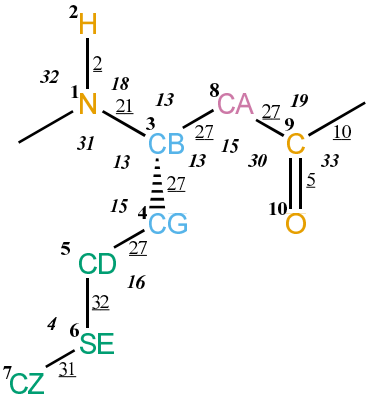


FIGURE 4.92. RBM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 8 9
4	CG	15	4	0.00000	5 6 8
5	CD	15	4	0.24100	6 7
6	SE	23	32	-0.48200	7
7	CZ	16	5	0.24100	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.226. Atoms of building block RBM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	32
6	7	31
8	9	27
9	10	5
9	11	10

TABLE 4.227. Bonds of building block RBM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	16
5	6	7	4
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.228. Bond angles of building block RBM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	34
4	5	6	7	26
3	8	9	11	42
3	8	9	11	45

TABLE 4.229. Dihedral angles of building block RBM.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
9	8	11	10	1

TABLE 4.230. Improper dihedral angles of building block RBM.

Solute building block: (R)-β^3-Asparagine
Name: RBN

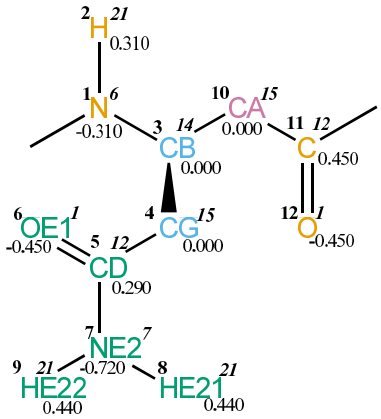


FIGURE 4.93. RBN non-bonded parameters.

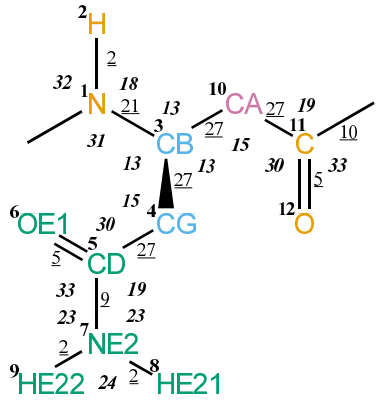


FIGURE 4.94. RBN bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 10
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 10 11
4	CG	15	4	0.00000	5 6 7 10
5	CD	12	12	0.29000	6 7 8 9
6	OE1	1	16	-0.45000	7
7	NE2	7	14	-0.72000	8 9
8	HE21	21	1	0.44000	9
9	HE22	21	1	0.44000	
10	CA	15	4	0.00000	11 12 13
11	C	12	12	0.45000	
12	O	1	16	-0.45000	

TABLE 4.231. Atoms of building block RBN.

I	J	Type
1	2	2
1	3	21
3	4	27
3	10	27
4	5	27
5	6	5
5	7	9
7	8	2
7	9	2
10	11	27
11	12	5
11	13	10

TABLE 4.232. Bonds of building block RBN.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	10	13
4	3	10	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33
5	7	8	23
5	7	9	23
8	7	9	24
3	10	11	15
10	11	12	30
10	11	13	19
12	11	13	33

TABLE 4.233. Bond angles of building block RBN.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	10	43
-1	1	3	10	44
1	3	4	5	34
1	3	10	11	34
3	4	5	7	40
4	5	7	8	14
3	10	11	13	42
3	10	11	13	45

TABLE 4.234. Dihedral angles of building block RBN.

I	J	K	L	Type
1	-1	3	2	1
4	1	10	3	2
5	6	7	4	1
7	8	9	5	1
11	10	13	12	1

TABLE 4.235. Improper dihedral angles of building block RBN.

Solute building block: (R)-β^3-Serine
Name: RBS

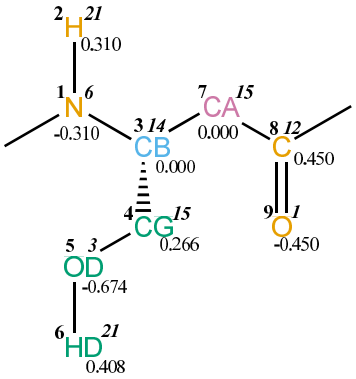


FIGURE 4.95. RBS non-bonded parameters.

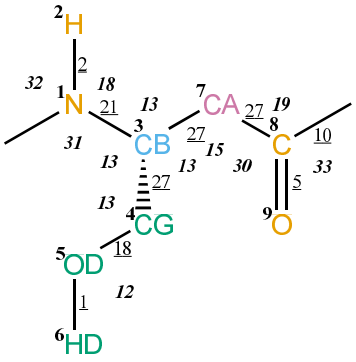


FIGURE 4.96. RBS bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8
4	CG	15	4	0.26600	5 6 7
5	OD	3	16	-0.67400	6
6	HD	21	1	0.40800	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.236. Atoms of building block RBS.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	18
5	6	1
7	8	27
8	9	5
8	10	10

TABLE 4.237. Bonds of building block RBS.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	13
4	5	6	12
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.238. Bond angles of building block RBS.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	4	5	6	23
3	7	8	10	42
3	7	8	10	45

TABLE 4.239. Dihedral angles of building block RBS.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
8	7	10	9	1

TABLE 4.240. Improper dihedral angles of building block RBS.

Solute building block: (R)-β^3-Serine(propylated)
Name: RBSP

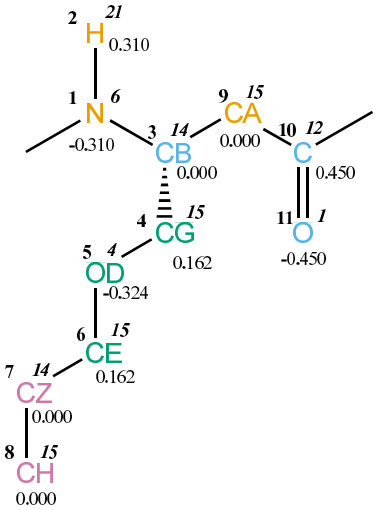


FIGURE 4.97. RBSP non-bonded parameters.

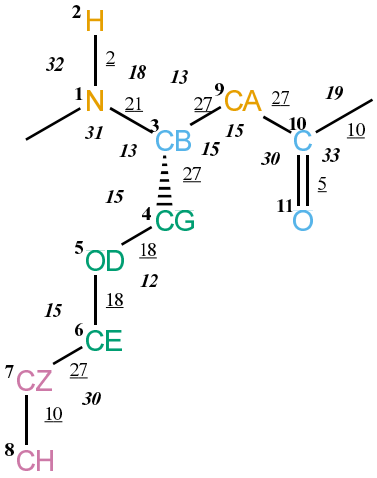


FIGURE 4.98. RBSP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 9
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 9 10
4	CG	15	4	0.16200	5 6 9
5	OD	4	16	-0.32400	6 7
6	CE	15	4	0.16200	7 8
7	CZ	14	3	0.00000	8
8	CH	15	4	0.00000	
9	CA	15	4	0.00000	10 11 12
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.241. Atoms of building block RBSP.

I	J	Type
1	2	2
1	3	21
3	4	27
3	9	27
4	5	18
5	6	18
6	7	27
7	8	10
9	10	27
10	11	5
10	12	10

TABLE 4.242. Bonds of building block RBSP.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	9	13
4	3	9	15
3	4	5	15
4	5	6	12
5	6	7	15
6	7	8	30
3	9	10	15
9	10	11	30
9	10	12	19
11	10	12	33

TABLE 4.243. Bond angles of building block RBSP.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	9	43
-1	1	3	9	44
1	3	4	5	34
1	3	9	10	34
3	4	5	6	23
4	5	6	7	23
5	6	7	8	40
3	9	10	12	42
3	9	10	12	45

TABLE 4.244. Dihedral angles of building block RBSP.

I	J	K	L	Type
1	-1	3	2	1
3	1	9	4	2
10	9	12	11	1

TABLE 4.245. Improper dihedral angles of building block RBSP.

Solute building block: (R)-β^3-Threonine
Name: RBT

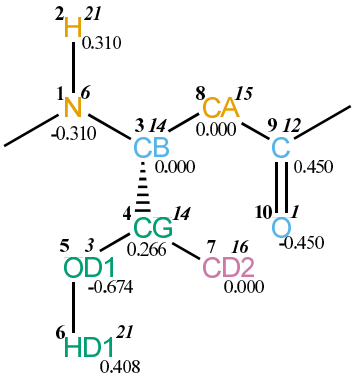


FIGURE 4.99. RBT non-bonded parameters.

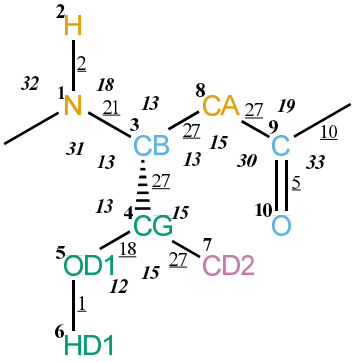


FIGURE 4.100. RBT bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8 9
4	CG	14	3	0.26600	5 6 7 8
5	OD1	3	16	-0.67400	6 7
6	HD1	21	1	0.40800	
7	CD2	16	5	0.00000	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.246. Atoms of building block RBT.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	18
4	7	27
5	6	1
8	9	27
9	10	5
9	11	10

TABLE 4.247. Bonds of building block RBT.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	13
3	4	7	15
5	4	7	15
4	5	6	12
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.248. Bond angles of building block RBT.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	23
3	8	9	11	42
3	8	9	11	45

TABLE 4.249. Dihedral angles of building block RBT.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	5	7	3	2
9	8	11	10	1

TABLE 4.250. Improper dihedral angles of building block RBT.

Solute building block: (R)-β^3-Valine
Name: RBV

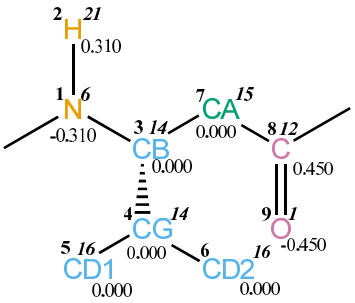


FIGURE 4.101. RBV non-bonded parameters.

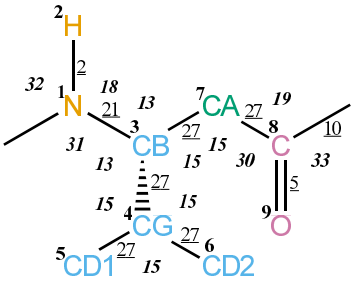


FIGURE 4.102. RBV bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 7 8
4	CG	14	3	0.00000	5 6 7
5	CD1	16	5	0.00000	6
6	CD2	16	5	0.00000	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.251. Atoms of building block RBV.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	27
4	6	27
7	8	27
8	9	5
8	10	10

TABLE 4.252. Bonds of building block RBV.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	15
3	4	5	15
3	4	6	15
5	4	6	15
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.253. Bond angles of building block RBV.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	7	8	10	42
3	7	8	10	45

TABLE 4.254. Dihedral angles of building block RBV.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
3	5	6	4	2
8	7	10	9	1

TABLE 4.255. Improper dihedral angles of building block RBV.

Solute building block: (S)-β^2-Alanine
Name: SAA

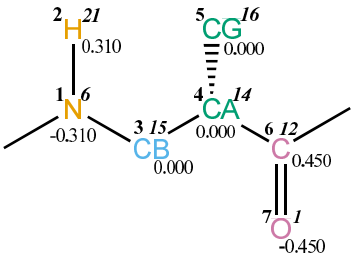


FIGURE 4.103. SAA non-bonded parameters.

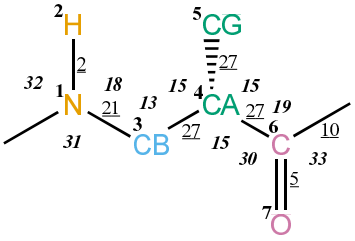


FIGURE 4.104. SAA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 6
4	CA	14	3	0.00000	5 6 7 8
5	CG	16	5	0.00000	6
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.256. Atoms of building block SAA.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	6	27
6	7	5
6	8	10

TABLE 4.257. Bonds of building block SAA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	6	15
5	4	6	15
4	6	7	30
4	6	8	19
7	6	8	33

TABLE 4.258. Bond angles of building block SAA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	6	34
3	4	6	8	42
3	4	6	8	45

TABLE 4.259. Dihedral angles of building block SAA.

I	J	K	L	Type
1	-1	3	2	1
5	3	6	4	2
6	4	8	7	1

TABLE 4.260. Improper dihedral angles of building block SAA.

Solute building block: (S)-β^2-Phenylalanine
Name: SAF

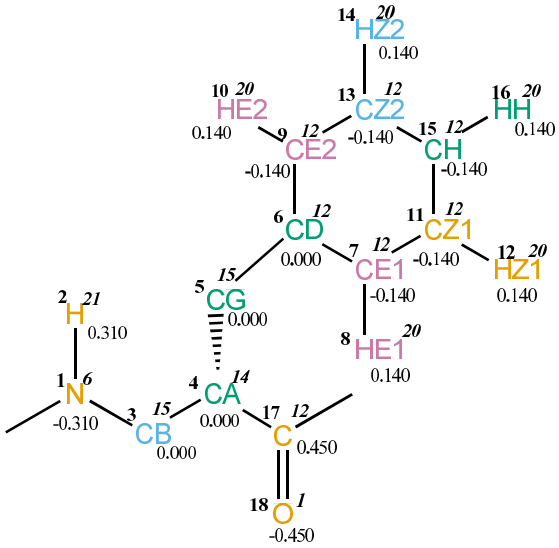


FIGURE 4.105. SAF non-bonded parameters.

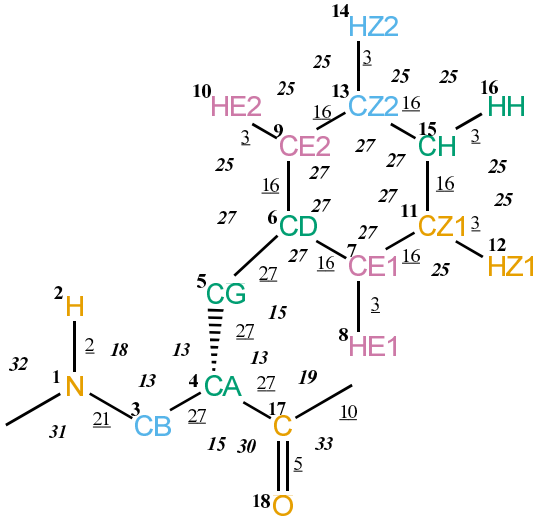


FIGURE 4.106. SAF bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 17
4	CA	14	3	0.00000	5 6 17 18 19
5	CG	15	4	0.00000	6 7 8 9 10 11 13 17
6	CD	12	12	0.00000	7 8 9 10 11 12 13 14 15
7	CE1	12	12	-0.14000	8 9 10 11 12 13 15 16
8	HE1	20	1	0.14000	9 11 12 15
9	CE2	12	12	-0.14000	10 11 13 14 15 16
10	HE2	20	1	0.14000	13 14 15
11	CZ1	12	12	-0.14000	12 13 14 15 16
12	HZ1	20	1	0.14000	13 15 16
13	CZ2	12	12	-0.14000	14 15 16
14	HZ2	20	1	0.14000	15 16
15	CH	12	12	-0.14000	16
16	HH	20	1	0.14000	
17	C	12	12	0.45000	
18	O	1	16	-0.45000	

TABLE 4.261. Atoms of building block SAF.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	17	27
5	6	27
6	7	16
6	9	16
7	8	3
7	11	16
9	10	3
9	13	16
11	12	3
11	15	16
13	14	3
13	15	16
15	16	3
17	18	5
17	19	10

TABLE 4.262. Bonds of building block SAF.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	13
3	4	17	15
5	4	17	13
4	5	6	15
5	6	7	27
5	6	9	27
7	6	9	27
6	7	8	25
6	7	11	27
8	7	11	25
6	9	10	25
6	9	13	27
10	9	13	25
7	11	12	25
7	11	15	27
12	11	15	25
9	13	14	25
9	13	15	27
14	13	15	25
11	15	13	27
11	15	16	25
13	15	16	25
4	17	18	30
4	17	19	19
18	17	19	33

TABLE 4.263. Bond angles of building block SAF.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	17	34
3	4	5	6	34
3	4	17	19	42
3	4	17	19	45
4	5	6	7	40

TABLE 4.264. Dihedral angles of building block SAF.

I	J	K	L	Type
1	-1	3	2	1
5	3	17	4	2
6	7	9	5	1
6	7	11	15	1
6	9	13	15	1
7	6	9	13	1
7	6	11	8	1
7	11	15	13	1
9	6	7	11	1
9	6	13	10	1
9	13	15	11	1
12	7	15	11	1
14	9	15	13	1
15	11	13	16	1
17	4	19	18	1

TABLE 4.265. Improper dihedral angles of building block SAF.

Solute building block: (S)-β^2-Phenylalanine(Cα fluorinated)
Name: SAFF

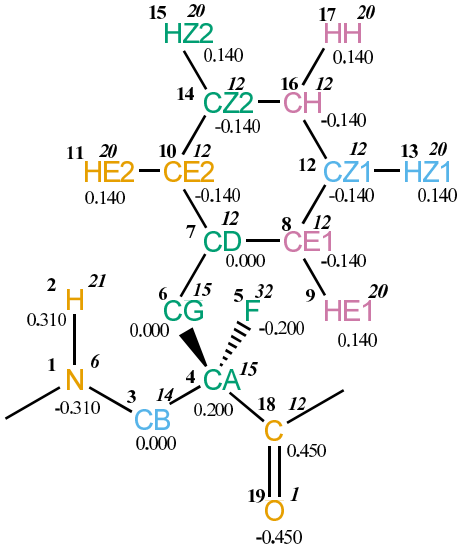


FIGURE 4.107. SAFF non-bonded parameters.

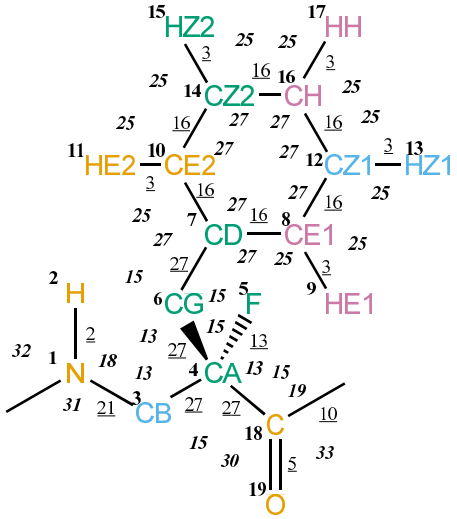


FIGURE 4.108. SAFF bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 18
4	CA	15	4	0.20000	5 6 7 18 19 20
5	F	32	19	-0.20000	6 18 19 20
6	CG	15	4	0.00000	7 8 9 10 11 12 14 18
7	CD	12	12	0.00000	8 9 10 11 12 13 14 15 16
8	CE1	12	12	-0.14000	9 10 11 12 13 14 16 17
9	HE1	20	1	0.14000	10 12 13 16
10	CE2	12	12	-0.14000	11 12 14 15 16 17
11	HE2	20	1	0.14000	14 15 16
12	CZ1	12	12	-0.14000	13 14 15 16 17
13	HZ1	20	1	0.14000	14 16 17
14	CZ2	12	12	-0.14000	15 16 17
15	HZ2	20	1	0.14000	16 17
16	CH	12	12	-0.14000	17
17	HH	20	1	0.14000	
18	C	12	12	0.45000	
19	O	1	16	-0.45000	

TABLE 4.266. Atoms of building block SAFF.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	13
4	6	27
4	18	27
6	7	27
7	8	16
7	10	16
8	9	3
8	12	16
10	11	3
10	14	16
12	13	3
12	16	16
14	15	3
14	16	16
16	17	3
18	19	5
18	20	10

TABLE 4.267. Bonds of building block SAFF.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	6	13
3	4	18	15
5	4	6	15
5	4	18	15
6	4	18	13
4	6	7	15
6	7	8	27
6	7	10	27
8	7	10	27
7	8	9	25
7	8	12	27
9	8	12	25
7	10	11	25
7	10	14	27
11	10	14	25
8	12	13	25
8	12	16	27
13	12	16	25
10	14	15	25
10	14	16	27
15	14	16	25
12	16	14	27
12	16	17	25
14	16	17	25
4	18	19	30
4	18	20	19
19	18	20	33

TABLE 4.268. Bond angles of building block SAFF.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	18	34
3	4	6	7	34
3	4	18	20	42
3	4	18	20	45
4	6	7	8	40

TABLE 4.269. Dihedral angles of building block SAFF.

I	J	K	L	Type
1	-1	3	2	1
4	3	18	6	2
7	8	10	6	1
7	8	12	16	1
7	10	14	16	1
8	7	10	14	1
8	7	12	9	1
8	12	16	14	1
10	7	8	12	1
10	7	14	11	1
10	14	16	12	1
13	8	16	12	1
15	10	16	14	1
16	12	14	17	1
18	4	20	19	1

TABLE 4.270. Improper dihedral angles of building block SAFF.

Solute building block: (S)-β^2-Leucine
Name: SAL

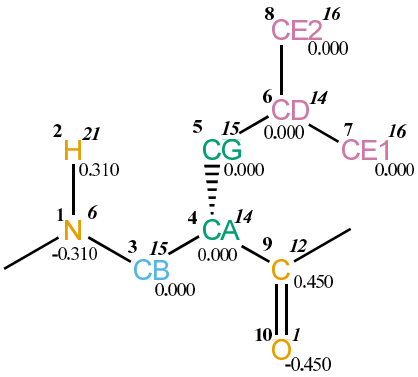


FIGURE 4.109. SAL non-bonded parameters.

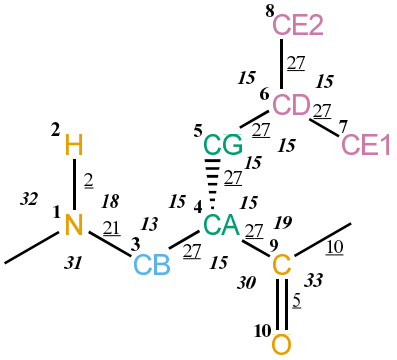


FIGURE 4.110. SAL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 9
4	CA	14	3	0.00000	5 6 9 10 11
5	CG	15	4	0.00000	6 7 8 9
6	CD	14	3	0.00000	7 8
7	CE1	16	5	0.00000	8
8	CE2	16	5	0.00000	
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.271. Atoms of building block SAL.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	9	27
5	6	27
6	7	27
6	8	27
9	10	5
9	11	10

TABLE 4.272. Bonds of building block SAL.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	9	15
5	4	9	15
4	5	6	15
5	6	7	15
5	6	8	15
7	6	8	15
4	9	10	30
4	9	11	19
10	9	11	33

TABLE 4.273. Bond angles of building block SAL.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	9	34
3	4	5	6	34
3	4	9	11	42
3	4	9	11	45
4	5	6	7	34

TABLE 4.274. Dihedral angles of building block SAL.

I	J	K	L	Type
1	-1	3	2	1
5	3	9	4	2
5	7	8	6	2
9	4	11	10	1

TABLE 4.275. Improper dihedral angles of building block SAL.

Solute building block: (S)-β^2-Methionine
Name: SAM

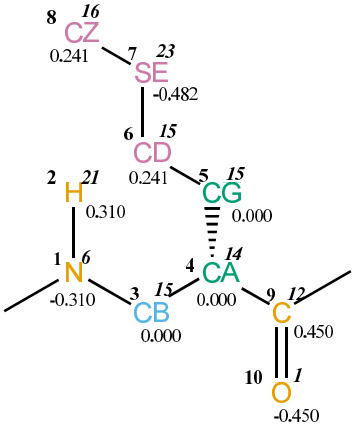


FIGURE 4.111. SAM non-bonded parameters.

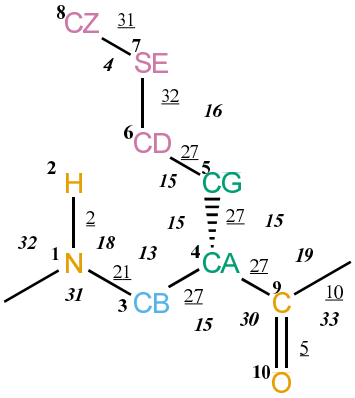


FIGURE 4.112. SAM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 9
4	CA	14	3	0.00000	5 6 9 10 11
5	CG	15	4	0.00000	6 7 9
6	CD	15	4	0.24100	7 8
7	SE	23	32	-0.48200	8
8	CZ	16	5	0.24100	
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.276. Atoms of building block SAM.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	9	27
5	6	27
6	7	32
7	8	31
9	10	5
9	11	10

TABLE 4.277. Bonds of building block SAM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	9	15
5	4	9	15
4	5	6	15
5	6	7	16
6	7	8	4
4	9	10	30
4	9	11	19
10	9	11	33

TABLE 4.278. Bond angles of building block SAM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	9	34
3	4	5	6	34
3	4	9	11	42
3	4	9	11	45
4	5	6	7	34
5	6	7	8	26

TABLE 4.279. Dihedral angles of building block SAM.

I	J	K	L	Type
1	-1	3	2	1
5	3	9	4	2
9	4	11	10	1

TABLE 4.280. Improper dihedral angles of building block SAM.

Solute building block: (S)-β^2-Valine
Name: SAV

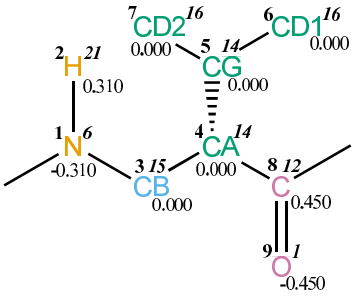


FIGURE 4.113. SAV non-bonded parameters.

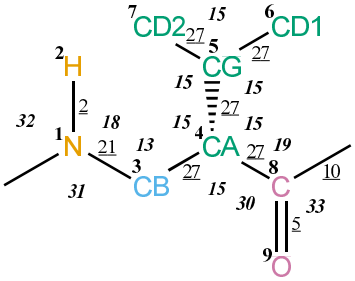


FIGURE 4.114. SAV bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5 8
4	CA	14	3	0.00000	5 6 7 8 9 10
5	CG	14	3	0.00000	6 7 8
6	CD1	16	5	0.00000	7
7	CD2	16	5	0.00000	
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.281. Atoms of building block SAV.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
4	8	27
5	6	27
5	7	27
8	9	5
8	10	10

TABLE 4.282. Bonds of building block SAV.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
3	4	8	15
5	4	8	15
4	5	6	15
4	5	7	15
6	5	7	15
4	8	9	30
4	8	10	19
9	8	10	33

TABLE 4.283. Bond angles of building block SAV.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	8	34
3	4	5	6	34
3	4	8	10	42
3	4	8	10	45

TABLE 4.284. Dihedral angles of building block SAV.

I	J	K	L	Type
1	-1	3	2	1
4	6	7	5	2
5	3	8	4	2
8	4	10	9	1

TABLE 4.285. Improper dihedral angles of building block SAV.

Solute building block: (S)-β^3-Alanine
Name: SBA

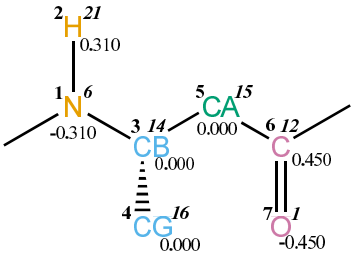


FIGURE 4.115. SBA non-bonded parameters.

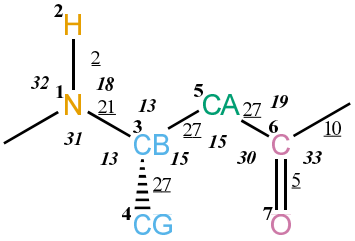


FIGURE 4.116. SBA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6
4	CG	16	5	0.00000	5
5	CA	15	4	0.00000	6 7 8
6	C	12	12	0.45000	
7	O	1	16	-0.45000	

TABLE 4.286. Atoms of building block SBA.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
5	6	27
6	7	5
6	8	10

TABLE 4.287. Bonds of building block SBA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
4	3	5	15
3	5	6	15
5	6	7	30
5	6	8	19
7	6	8	33

TABLE 4.288. Bond angles of building block SBA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	5	43
-1	1	3	5	44
1	3	5	6	34
3	5	6	8	42
3	5	6	8	45

TABLE 4.289. Dihedral angles of building block SBA.

I	J	K	L	Type
1	-1	3	2	1
3	1	5	4	2
6	5	8	7	1

TABLE 4.290. Improper dihedral angles of building block SBA.

Solute building block: (S)-β^3-Cysteine (protonated)
Name: SBCH

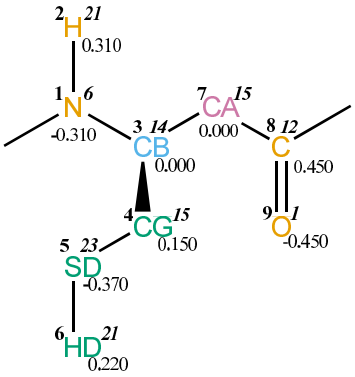


FIGURE 4.117. SBCH non-bonded parameters.

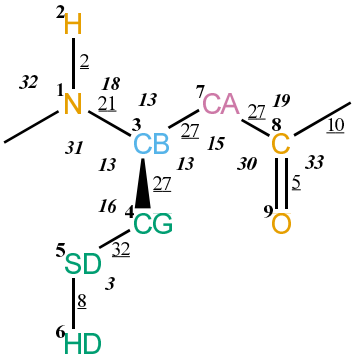


FIGURE 4.118. SBCH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8
4	CG	15	4	0.15000	5 6 7
5	SD	23	32	-0.37000	6
6	HD	21	1	0.22000	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.291. Atoms of building block SBCH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	32
5	6	8
7	8	27
8	9	5
8	10	10

TABLE 4.292. Bonds of building block SBCH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	16
4	5	6	3
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.293. Bond angles of building block SBCH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	4	5	6	26
3	7	8	10	42
3	7	8	10	45

TABLE 4.294. Dihedral angles of building block SBCH.

I	J	K	L	Type
1	-1	3	2	1
4	1	7	3	2
8	7	10	9	1

TABLE 4.295. Improper dihedral angles of building block SBCH.

Solute building block: (S)-β^3-Aspartic acid (deprotonated; charge -e)
Name: SBD

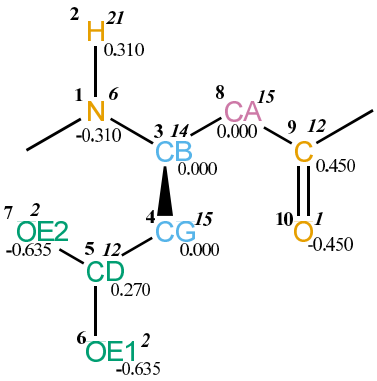


FIGURE 4.119. SBD non-bonded parameters.

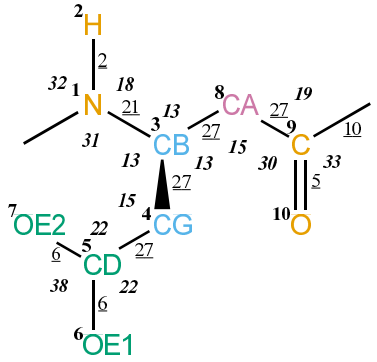


FIGURE 4.120. SBD bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 8 9
4	CG	15	4	0.00000	5 6 7 8
5	CD	12	12	0.27000	6 7
6	OE1	2	16	-0.63500	7
7	OE2	2	16	-0.63500	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.296. Atoms of building block SBD.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	6
5	7	6
8	9	27
9	10	5
9	11	10

TABLE 4.297. Bonds of building block SBD.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	22
4	5	7	22
6	5	7	38
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.298. Bond angles of building block SBD.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	40
3	8	9	11	42
3	8	9	11	45

TABLE 4.299. Dihedral angles of building block SBD.

I	J	K	L	Type
1	-1	3	2	1
4	1	8	3	2
5	6	7	4	1
9	8	11	10	1

TABLE 4.300. Improper dihedral angles of building block SBD.

Solute building block: (S)-β^3-Aspartic acid (protonated; neutral)
Name: SBDH

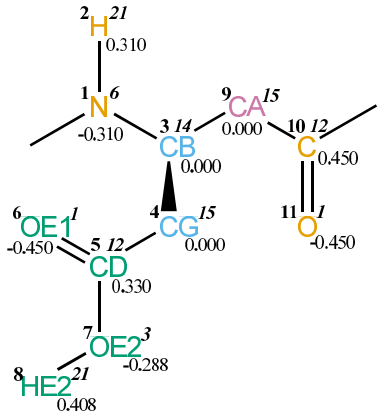


FIGURE 4.121. SBDH non-bonded parameters.

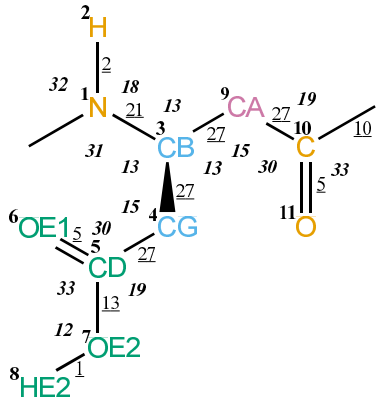


FIGURE 4.122. SBDH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 9
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 9 10
4	CG	15	4	0.00000	5 6 7 9
5	CD	12	12	0.33000	6 7 8
6	OE1	1	16	-0.45000	7
7	OE2	3	16	-0.28800	8
8	HE2	21	1	0.40800	
9	CA	15	4	0.00000	10 11 12
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.301. Atoms of building block SBDH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	9	27
4	5	27
5	6	5
5	7	13
7	8	1
9	10	27
10	11	5
10	12	10

TABLE 4.302. Bonds of building block SBDH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	9	13
4	3	9	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33
5	7	8	12
3	9	10	15
9	10	11	30
9	10	12	19
11	10	12	33

TABLE 4.303. Bond angles of building block SBDH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	9	43
-1	1	3	9	44
1	3	4	5	34
1	3	9	10	34
3	4	5	7	40
4	5	7	8	12
3	9	10	12	42
3	9	10	12	45

TABLE 4.304. Dihedral angles of building block SBDH.

I	J	K	L	Type
1	-1	3	2	1
4	1	9	3	2
5	6	7	4	1
10	9	12	11	1

TABLE 4.305. Improper dihedral angles of building block SBDH.

Solute building block: (S)-β^3-Glutamic acid (deprotonated; charge -e)
Name: SBE

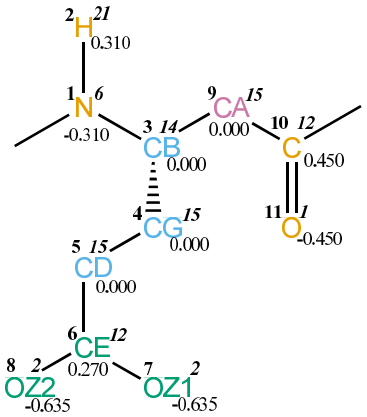


FIGURE 4.123. SBE non-bonded parameters.

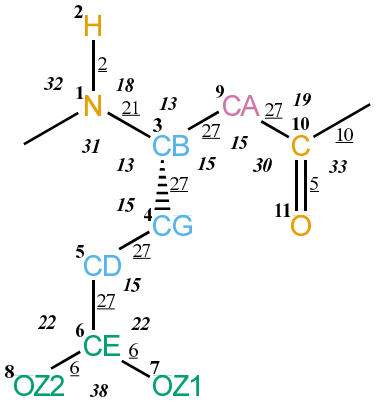


FIGURE 4.124. SBE bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 9
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 9 10
4	CG	15	4	0.00000	5 6 9
5	CD	15	4	0.00000	6 7 8
6	CE	12	12	0.27000	7 8
7	OZ1	2	16	-0.63500	8
8	OZ2	2	16	-0.63500	
9	CA	15	4	0.00000	10 11 12
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.306. Atoms of building block SBE.

I	J	Type
1	2	2
1	3	21
3	4	27
3	9	27
4	5	27
5	6	27
6	7	6
6	8	6
9	10	27
10	11	5
10	12	10

TABLE 4.307. Bonds of building block SBE.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	9	13
4	3	9	15
3	4	5	15
4	5	6	15
5	6	7	22
5	6	8	22
7	6	8	38
3	9	10	15
9	10	11	30
9	10	12	19
11	10	12	33

TABLE 4.308. Bond angles of building block SBE.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	9	43
-1	1	3	9	44
1	3	4	5	34
1	3	9	10	34
3	4	5	6	34
4	5	6	8	40
3	9	10	12	42
3	9	10	12	45

TABLE 4.309. Dihedral angles of building block SBE.

I	J	K	L	Type
1	-1	3	2	1
3	1	9	4	2
6	7	8	5	1
10	9	12	11	1

TABLE 4.310. Improper dihedral angles of building block SBE.

Solute building block: (S)-β^3-Glutamic acid (protonated; neutral)
Name: SBEH

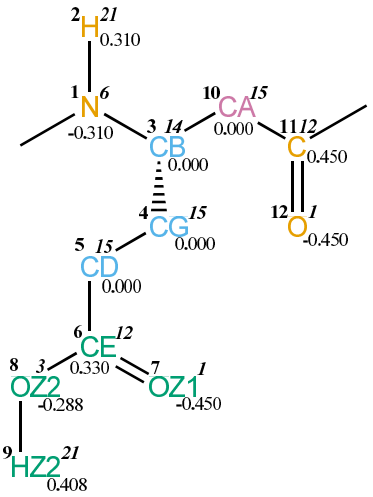


FIGURE 4.125. SBEH non-bonded parameters.

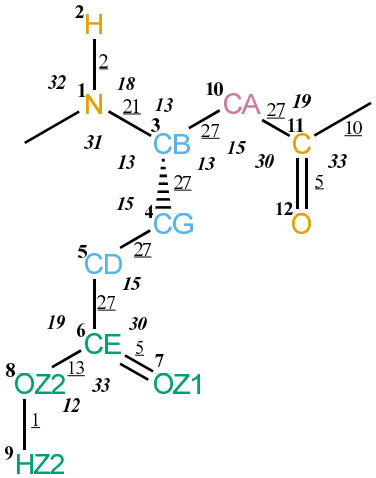


FIGURE 4.126. SBEH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 10
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 10 11
4	CG	15	4	0.00000	5 6 10
5	CD	15	4	0.00000	6 7 8
6	CE	12	12	0.33000	7 8 9
7	OZ1	1	16	-0.45000	8
8	OZ2	3	16	-0.28800	9
9	HZ2	21	1	0.40800	
10	CA	15	4	0.00000	11 12 13
11	C	12	12	0.45000	
12	O	1	16	-0.45000	

TABLE 4.311. Atoms of building block SBEH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	10	27
4	5	27
5	6	27
6	7	5
6	8	13
8	9	1
10	11	27
11	12	5
11	13	10

TABLE 4.312. Bonds of building block SBEH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	10	13
4	3	10	13
3	4	5	15
4	5	6	15
5	6	7	30
5	6	8	19
7	6	8	33
6	8	9	12
3	10	11	15
10	11	12	30
10	11	13	19
12	11	13	33

TABLE 4.313. Bond angles of building block SBEH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	10	43
-1	1	3	10	44
1	3	4	5	34
1	3	10	11	34
3	4	5	6	34
4	5	6	8	40
5	6	8	9	12
3	10	11	13	42
3	10	11	13	45

TABLE 4.314. Dihedral angles of building block SBEH.

I	J	K	L	Type
1	-1	3	2	1
3	1	10	4	2
6	7	8	5	1
11	10	13	12	1

TABLE 4.315. Improper dihedral angles of building block SBEH.

Solute building block: (S)-β^3-Glutamine
Name: SBQ

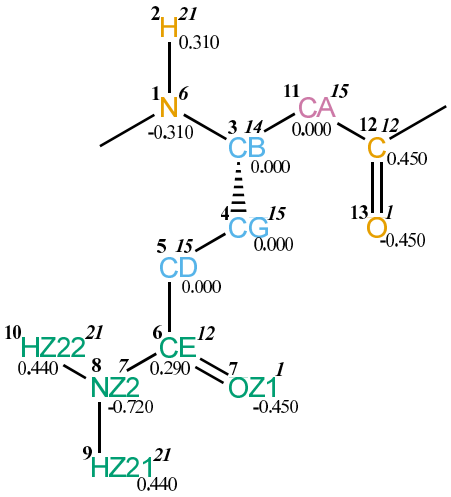


FIGURE 4.127. SBQ non-bonded parameters.

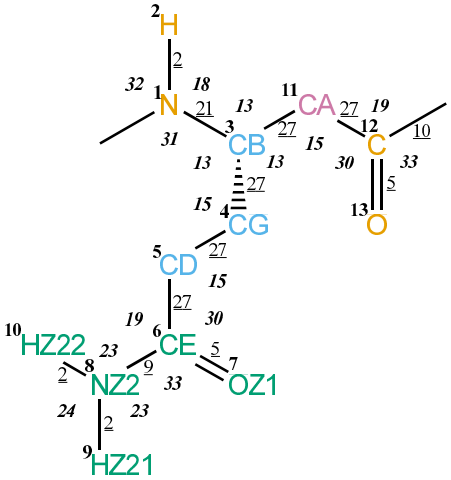


FIGURE 4.128. SBQ bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 11
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 11 12
4	CG	15	4	0.00000	5 6 11
5	CD	15	4	0.00000	6 7 8
6	CE	12	12	0.29000	7 8 9 10
7	OZ1	1	16	-0.45000	8
8	NZ2	7	14	-0.72000	9 10
9	HZ21	21	1	0.44000	10
10	HZ22	21	1	0.44000	
11	CA	15	4	0.00000	12 13 14
12	C	12	12	0.45000	
13	O	1	16	-0.45000	

TABLE 4.316. Atoms of building block SBQ.

I	J	Type
1	2	2
1	3	21
3	4	27
3	11	27
4	5	27
5	6	27
6	7	5
6	8	9
8	9	2
8	10	2
11	12	27
12	13	5
12	14	10

TABLE 4.317. Bonds of building block SBQ.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	11	13
4	3	11	13
3	4	5	15
4	5	6	15
5	6	7	30
5	6	8	19
7	6	8	33
6	8	9	23
6	8	10	23
9	8	10	24
3	11	12	15
11	12	13	30
11	12	14	19
13	12	14	33

TABLE 4.318. Bond angles of building block SBQ.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	11	43
-1	1	3	11	44
1	3	4	5	34
1	3	11	12	34
3	4	5	6	34
4	5	6	8	40
5	6	8	9	14
3	11	12	14	42
3	11	12	14	45

TABLE 4.319. Dihedral angles of building block SBQ.

I	J	K	L	Type
1	-1	3	2	1
3	1	11	4	2
6	7	8	5	1
8	9	10	6	1
12	11	14	13	1

TABLE 4.320. Improper dihedral angles of building block SBQ.

Solute building block: (S)-β^3-Phenylalanine
Name: SBF

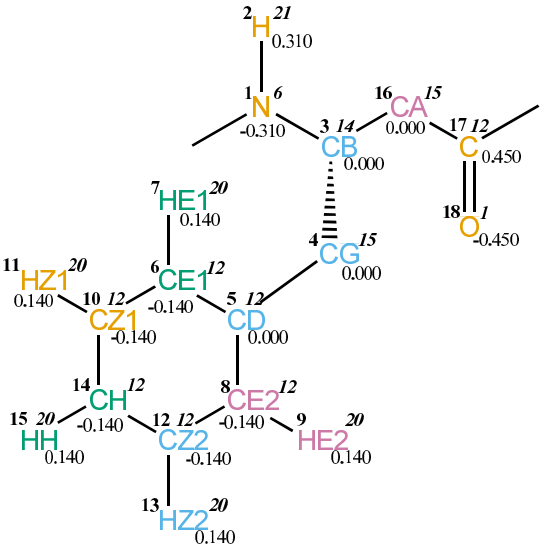


FIGURE 4.129. SBF non-bonded parameters.

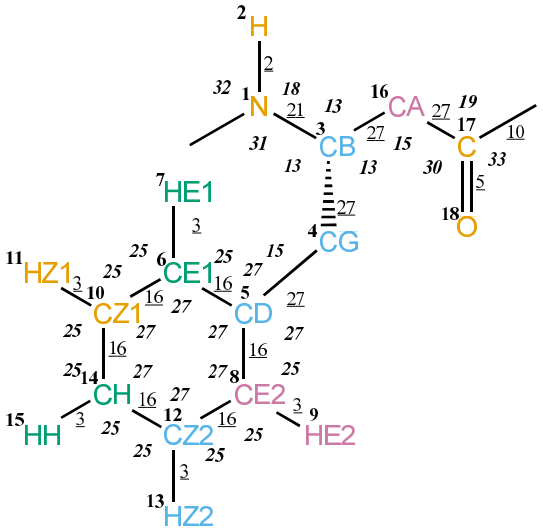


FIGURE 4.130. SBF bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 16
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 16 17
4	CG	15	4	0.00000	5 6 7 8 9 10 12 16
5	CD	12	12	0.00000	6 7 8 9 10 11 12 13 14
6	CE1	12	12	-0.14000	7 8 9 10 11 12 14 15
7	HE1	20	1	0.14000	8 10 11 14
8	CE2	12	12	-0.14000	9 10 12 13 14 15
9	HE2	20	1	0.14000	12 13 14
10	CZ1	12	12	-0.14000	11 12 13 14 15
11	HZ1	20	1	0.14000	12 14 15
12	CZ2	12	12	-0.14000	13 14 15
13	HZ2	20	1	0.14000	14 15
14	CH	12	12	-0.14000	15
15	HH	20	1	0.14000	
16	CA	15	4	0.00000	17 18 19
17	C	12	12	0.45000	
18	O	1	16	-0.45000	

TABLE 4.321. Atoms of building block SBF.

I	J	Type
1	2	2
1	3	21
3	4	27
3	16	27
4	5	27
5	6	16
5	8	16
6	7	3
6	10	16
8	9	3
8	12	16
10	11	3
10	14	16
12	13	3
12	14	16
14	15	3
16	17	27
17	18	5
17	19	10

TABLE 4.322. Bonds of building block SBF.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	16	13
4	3	16	13
3	4	5	15
4	5	6	27
4	5	8	27
6	5	8	27
5	6	7	25
5	6	10	27
7	6	10	25
5	8	9	25
5	8	12	27
9	8	12	25
6	10	11	25
6	10	14	27
11	10	14	25
8	12	13	25
8	12	14	27
13	12	14	25
10	14	12	27
10	14	15	25
12	14	15	25
3	16	17	15
16	17	18	30
16	17	19	19
18	17	19	33

TABLE 4.323. Bond angles of building block SBF.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	16	43
-1	1	3	16	44
1	3	4	5	34
1	3	16	17	34
3	4	5	6	40
3	16	17	19	42
3	16	17	19	45

TABLE 4.324. Dihedral angles of building block SBF.

I	J	K	L	Type
1	-1	3	2	1
3	1	16	4	2
5	6	8	4	1
5	6	10	14	1
5	8	12	14	1
6	5	8	12	1
6	5	10	7	1
6	10	14	12	1
8	5	6	10	1
8	5	12	9	1
8	12	14	10	1
11	6	14	10	1
13	8	14	12	1
14	10	12	15	1
17	16	19	18	1

TABLE 4.325. Improper dihedral angles of building block SBF.

Solute building block: β-Glycine
Name: BGL

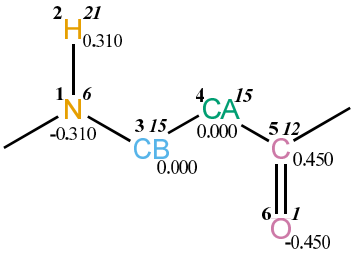


FIGURE 4.131. BGL non-bonded parameters.

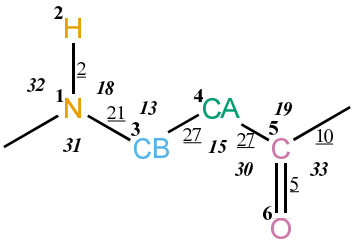


FIGURE 4.132. BGL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4
2	H	21	1	0.31000	3
3	CB	15	4	0.00000	4 5
4	CA	15	4	0.00000	5 6 7
5	C	12	12	0.45000	
6	O	1	16	-0.45000	

TABLE 4.326. Atoms of building block BGL.

I	J	Type
1	2	2
1	3	21
3	4	27
4	5	27
5	6	5
5	7	10

TABLE 4.327. Bonds of building block BGL.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
3	4	5	15
4	5	6	30
4	5	7	19
6	5	7	33

TABLE 4.328. Bond angles of building block BGL.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	4	43
-1	1	3	4	44
1	3	4	5	34
3	4	5	7	42
3	4	5	7	45

TABLE 4.329. Dihedral angles of building block BGL.

I	J	K	L	Type
1	-1	3	2	1
5	4	7	6	1

TABLE 4.330. Improper dihedral angles of building block BGL.

Solute building block: (S)-β^3-Histidine (protonated at NE1; neutral)
Name: SBHA

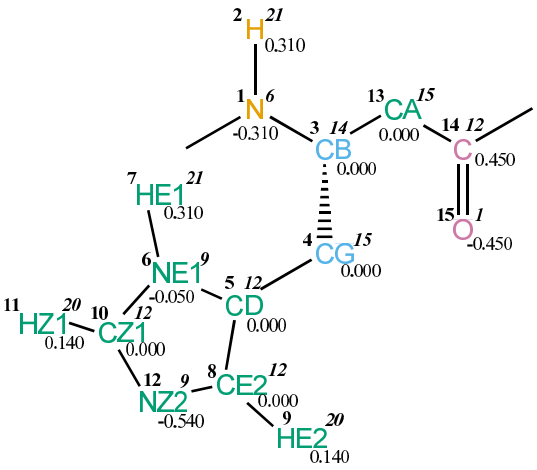


FIGURE 4.133. SBHA non-bonded parameters.

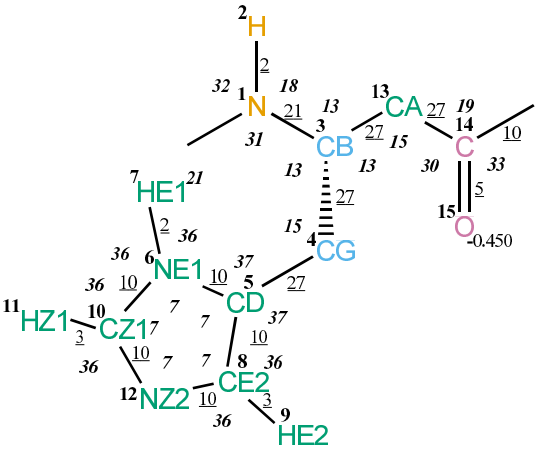


FIGURE 4.134. SBHA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 13
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 13 14
4	CG	15	4	0.00000	5 6 7 8 9 10 12 13
5	CD	12	12	0.00000	6 7 8 9 10 11 12
6	NE1	9	14	-0.05000	7 8 9 10 11 12
7	HE1	21	1	0.31000	8 10 11 12
8	CE2	12	12	0.00000	9 10 11 12
9	HE2	20	1	0.14000	10 12
10	CZ1	12	12	0.00000	11 12
11	HZ1	20	1	0.14000	12
12	NZ2	9	14	-0.54000	
13	CA	15	4	0.00000	14 15 16
14	C	12	12	0.45000	
15	O	1	16	-0.45000	

TABLE 4.331. Atoms of building block SBHA.

I	J	Type
1	2	2
1	3	21
3	4	27
3	13	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
13	14	27
14	15	5
14	16	10

TABLE 4.332. Bonds of building block SBHA.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	13	13
4	3	13	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
8	12	10	7
3	13	14	15
13	14	15	30
13	14	16	19
15	14	16	33

TABLE 4.333. Bond angles of building block SBHA.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	13	43
-1	1	3	13	44
1	3	4	5	34
1	3	13	14	34
3	4	5	6	40
3	13	14	16	42
3	13	14	16	45

TABLE 4.334. Dihedral angles of building block SBHA.

I	J	K	L	Type
1	-1	3	2	1
3	1	13	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
14	13	16	15	1

TABLE 4.335. Improper dihedral angles of building block SBHA.

Solute building block: (S)-β^3-Histidine(protonated at NE1 and NZ2; charge $+e$)
Name: SBHH

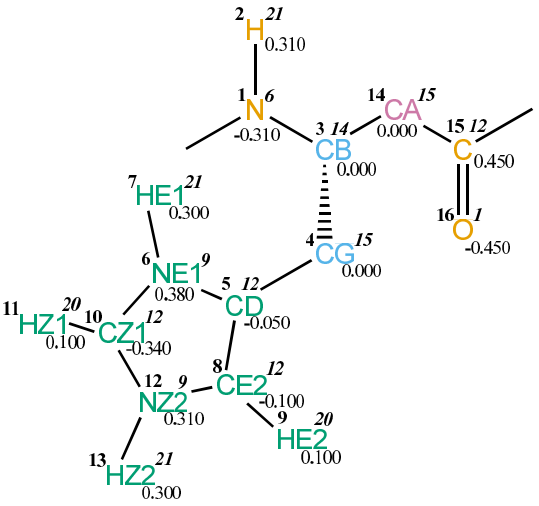


FIGURE 4.135. SBHH non-bonded parameters.

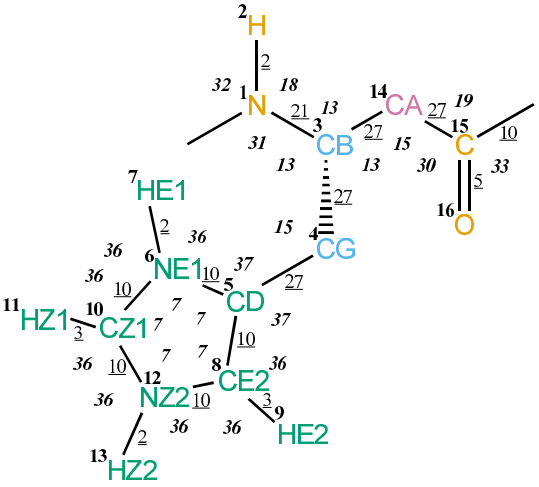


FIGURE 4.136. SBHH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 14
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 14 15
4	CG	15	4	0.00000	5 6 7 8 9 10 12 14
5	CD	12	12	-0.05000	6 7 8 9 10 11 12 13
6	NE1	9	14	0.38000	7 8 9 10 11 12 13
7	HE1	21	1	0.30000	8 10 11 12
8	CE2	12	12	-0.10000	9 10 11 12 13
9	HE2	20	1	0.10000	10 12 13
10	CZ1	12	12	-0.34000	11 12 13
11	HZ1	20	1	0.10000	12 13
12	NZ2	9	14	0.31000	13
13	HZ2	21	1	0.30000	
14	CA	15	4	0.00000	15 16 17
15	C	12	12	0.45000	
16	O	1	16	-0.45000	

TABLE 4.336. Atoms of building block SBHH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	14	27
4	5	27
5	6	10
5	8	10
6	7	2
6	10	10
8	9	3
8	12	10
10	11	3
10	12	10
12	13	2
14	15	27
15	16	5
15	17	10

TABLE 4.337. Bonds of building block SBHH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	14	13
4	3	14	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	10	7
7	6	10	36
5	8	9	36
5	8	12	7
9	8	12	36
6	10	11	36
6	10	12	7
11	10	12	36
8	12	10	7
8	12	13	36
10	12	13	36
3	14	15	15
14	15	16	30
14	15	17	19
16	15	17	33

TABLE 4.338. Bond angles of building block SBHH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	14	43
-1	1	3	14	44
1	3	4	5	34
1	3	14	15	34
3	4	5	6	40
3	14	15	17	42
3	14	15	17	45

TABLE 4.339. Dihedral angles of building block SBHH.

I	J	K	L	Type
1	-1	3	2	1
3	1	14	4	2
5	6	8	4	1
5	6	10	12	1
5	8	12	10	1
6	5	8	12	1
6	5	10	7	1
6	10	12	8	1
8	5	6	10	1
8	5	12	9	1
10	6	12	11	1
12	8	10	13	1
15	14	17	16	1

TABLE 4.340. Improper dihedral angles of building block SBHH.

Solute building block: (S)-β^3-Isoleucine
Name: SBI

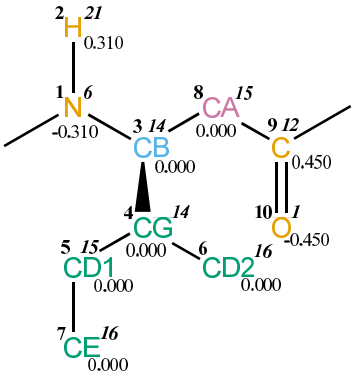


FIGURE 4.137. SBI non-bonded parameters.

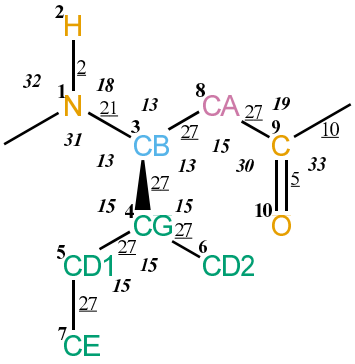


FIGURE 4.138. SBI bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 8 9
4	CG	14	3	0.00000	5 6 7 8
5	CD1	15	4	0.00000	6 7
6	CD2	16	5	0.00000	
7	CE	16	5	0.00000	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.341. Atoms of building block SBI.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
4	6	27
5	7	27
8	9	27
9	10	5
9	11	10

TABLE 4.342. Bonds of building block SBI.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
3	4	6	15
5	4	6	15
4	5	7	15
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.343. Bond angles of building block SBI.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	7	34
3	8	9	11	42
3	8	9	11	45

TABLE 4.344. Dihedral angles of building block SBI.

I	J	K	L	Type
1	-1	3	2	1
4	1	8	3	2
4	5	6	3	2
9	8	11	10	1

TABLE 4.345. Improper dihedral angles of building block SBI.

Solute building block: (S)-β^3-Lysine (protonated; charge $+e$)
Name: SBKH

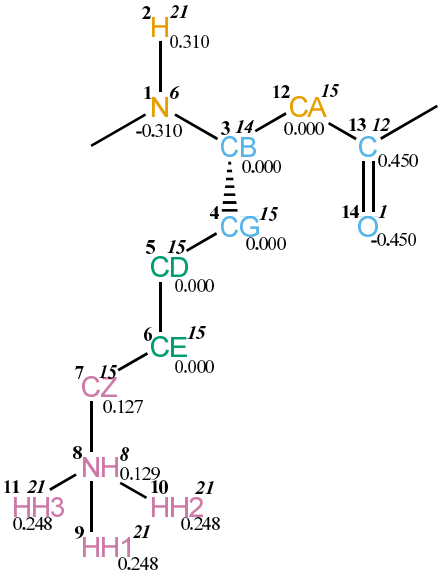


FIGURE 4.139. SBKH non-bonded parameters.

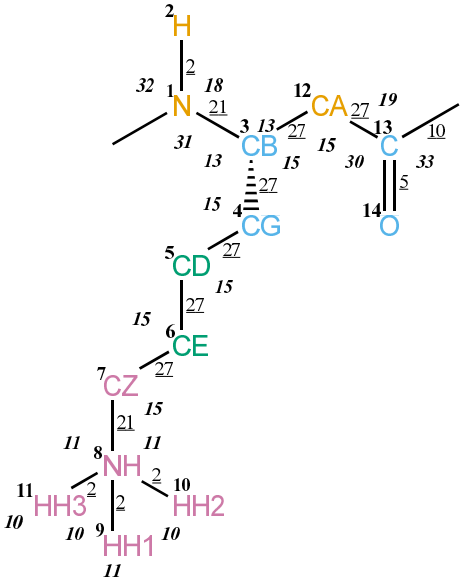


FIGURE 4.140. SBKH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 12
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 12 13
4	CG	15	4	0.00000	5 6 12
5	CD	15	4	0.00000	6 7
6	CE	15	4	0.00000	7 8
7	CZ	15	4	0.12700	8 9 10 11
8	NH	8	14	0.12900	9 10 11
9	HH1	21	1	0.24800	10 11
10	HH2	21	1	0.24800	11
11	HH3	21	1	0.24800	
12	CA	15	4	0.00000	13 14 15
13	C	12	12	0.45000	
14	O	1	16	-0.45000	

TABLE 4.346. Atoms of building block SBKH.

I	J	Type
1	2	2
1	3	21
3	4	27
3	12	27
4	5	27
5	6	27
6	7	27
7	8	21
8	9	2
8	10	2
8	11	2
12	13	27
13	14	5
13	15	10

TABLE 4.347. Bonds of building block SBKH.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	12	13
4	3	12	15
3	4	5	15
4	5	6	15
5	6	7	15
6	7	8	15
7	8	9	11
7	8	10	11
7	8	11	11
9	8	10	10
9	8	11	10
10	8	11	10
3	12	13	15
12	13	14	30
12	13	15	19
14	13	15	33

TABLE 4.348. Bond angles of building block SBKH.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	12	43
-1	1	3	12	44
1	3	4	5	34
1	3	12	13	34
3	4	5	6	34
4	5	6	7	34
5	6	7	8	34
6	7	8	9	29
3	12	13	15	42
3	12	13	15	45

TABLE 4.349. Dihedral angles of building block SBKH.

I	J	K	L	Type
1	-1	3	2	1
3	1	12	4	2
13	12	15	14	1

TABLE 4.350. Improper dihedral angles of building block SBKH.

Solute building block: (S)-β^3-Leucine
Name: SBL

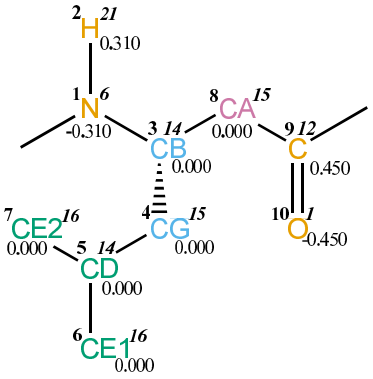


FIGURE 4.141. SBL non-bonded parameters.

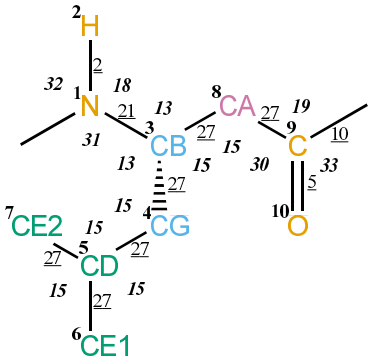


FIGURE 4.142. SBL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 8 9
4	CG	15	4	0.00000	5 6 7 8
5	CD	14	3	0.00000	6 7
6	CE1	16	5	0.00000	7
7	CE2	16	5	0.00000	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.351. Atoms of building block SBL.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	27
5	7	27
8	9	27
9	10	5
9	11	10

TABLE 4.352. Bonds of building block SBL.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	15
3	4	5	15
4	5	6	15
4	5	7	15
6	5	7	15
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.353. Bond angles of building block SBL.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	34
3	8	9	11	42
3	8	9	11	45

TABLE 4.354. Dihedral angles of building block SBL.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	6	7	5	2
9	8	11	10	1

TABLE 4.355. Improper dihedral angles of building block SBL.

Solute building block: (S)-β^3-Methionine
Name: SBM

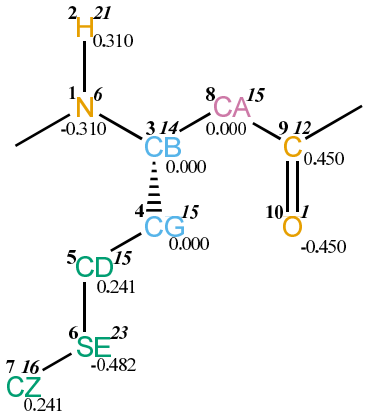


FIGURE 4.143. SBM non-bonded parameters.

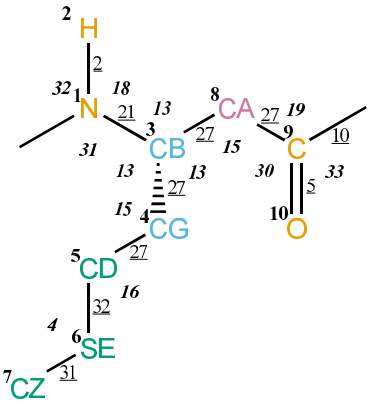


FIGURE 4.144. SBM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 8 9
4	CG	15	4	0.00000	5 6 8
5	CD	15	4	0.24100	6 7
6	SE	23	32	-0.48200	7
7	CZ	16	5	0.24100	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.356. Atoms of building block SBM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	32
6	7	31
8	9	27
9	10	5
9	11	10

TABLE 4.357. Bonds of building block SBM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	15
4	5	6	16
5	6	7	4
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.358. Bond angles of building block SBM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	34
4	5	6	7	26
3	8	9	11	42
3	8	9	11	45

TABLE 4.359. Dihedral angles of building block SBM.

I	J	K	L	Type
1	-1	3	2	1
4	1	8	3	2
9	8	11	10	1

TABLE 4.360. Improper dihedral angles of building block SBM.

Solute building block: (S)-β^3-Proline
Name: SBP

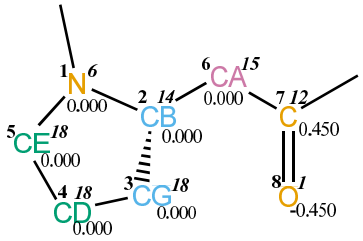


FIGURE 4.145. SBP non-bonded parameters.

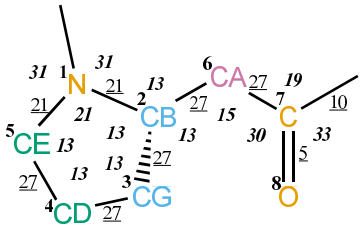


FIGURE 4.146. SBP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 5
0					1
1	N	6	14	0.00000	2 3 4 5 6
2	CB	14	3	0.00000	3 4 5 6 7
3	CG	18	4	0.00000	4 5 6
4	CD	18	4	0.00000	5
5	CE	18	4	0.00000	
6	CA	15	4	0.00000	7 8 9
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.361. Atoms of building block SBP.

I	J	Type
1	2	21
1	5	21
2	3	27
2	6	27
3	4	27
4	5	27
6	7	27
7	8	5
7	9	10

TABLE 4.362. Bonds of building block SBP.

I	J	K	Type
-1	1	2	31
-1	1	5	31
2	1	5	21
1	2	3	13
1	2	6	13
3	2	6	13
2	3	4	13
3	4	5	13
1	5	4	13
2	6	7	15
6	7	8	30
6	7	9	19
8	7	9	33

TABLE 4.363. Bond angles of building block SBP.

I	J	K	L	Type
-2	-1	1	2	14
-1	1	2	6	43
-1	1	2	6	44
2	1	5	4	39
1	2	3	4	34
1	2	6	7	34
2	3	4	5	34
3	4	5	1	34
2	6	7	9	42
2	6	7	9	45

TABLE 4.364. Dihedral angles of building block SBP.

I	J	K	L	Type
1	-1	2	5	1
2	1	6	3	2
7	6	9	8	1

TABLE 4.365. Improper dihedral angles of building block SBP.

Solute building block: (S)-β^3-Arginine (protonated; charge $+e$)
Name: SBR

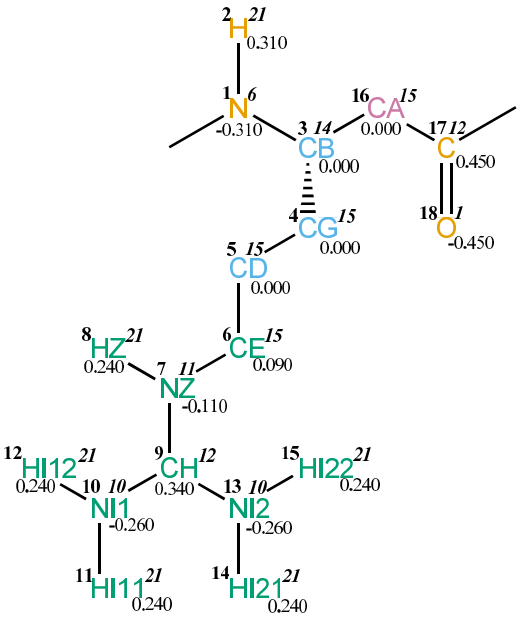


FIGURE 4.147. SBR non-bonded parameters.

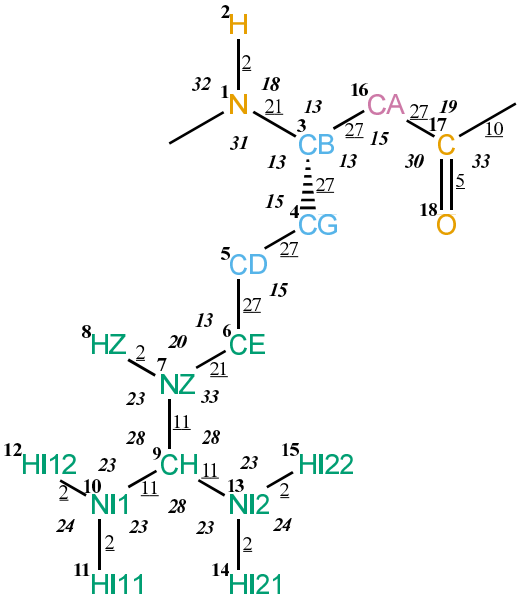


FIGURE 4.148. SBR bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 16
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 16 17
4	CG	15	4	0.00000	5 6 16
5	CD	15	4	0.00000	6 7
6	CE	15	4	0.09000	7 8 9
7	NZ	11	14	-0.11000	8 9 10 13
8	HZ	21	1	0.24000	9
9	CH	12	12	0.34000	10 11 12 13 14 15
10	NI1	10	14	-0.26000	11 12 13
11	HI11	21	1	0.24000	12
12	HI12	21	1	0.24000	
13	NI2	10	14	-0.26000	14 15
14	HI21	21	1	0.24000	15
15	HI22	21	1	0.24000	
16	CA	15	4	0.00000	17 18 19
17	C	12	12	0.45000	
18	O	1	16	-0.45000	

TABLE 4.366. Atoms of building block SBR.

I	J	Type
1	2	2
1	3	21
3	4	27
3	16	27
4	5	27
5	6	27
6	7	21
7	8	2
7	9	11
9	10	11
9	13	11
10	11	2
10	12	2
13	14	2
13	15	2
16	17	27
17	18	5
17	19	10

TABLE 4.367. Bonds of building block SBR.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	16	13
4	3	16	13
3	4	5	15
4	5	6	15
5	6	7	13
6	7	8	20
6	7	9	33
8	7	9	23
7	9	10	28
7	9	13	28
10	9	13	28
9	10	11	23
9	10	12	23
11	10	12	24
9	13	14	23
9	13	15	23
14	13	15	24
3	16	17	15
16	17	18	30
16	17	19	19
18	17	19	33

TABLE 4.368. Bond angles of building block SBR.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	16	43
-1	1	3	16	44
1	3	4	5	34
1	3	16	17	34
3	4	5	6	34
4	5	6	7	34
5	6	7	9	39
6	7	9	10	14
7	9	10	11	14
7	9	13	14	14
3	16	17	19	42
3	16	17	19	45

TABLE 4.369. Dihedral angles of building block SBR.

I	J	K	L	Type
1	-1	3	2	1
3	1	16	4	2
7	6	9	8	1
9	10	13	7	1
10	11	12	9	1
13	14	15	9	1
17	16	19	18	1

TABLE 4.370. Improper dihedral angles of building block SBR.

Solute building block: (S)-β^3-Serine
Name: SBS

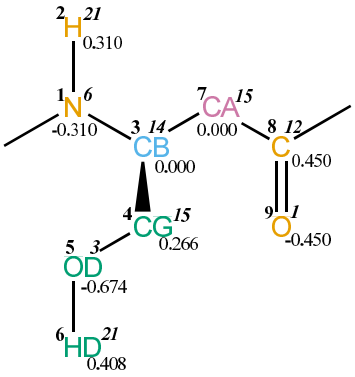


FIGURE 4.149. SBS non-bonded parameters.

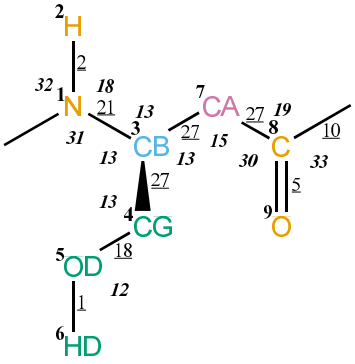


FIGURE 4.150. SBS bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8
4	CG	15	4	0.26600	5 6 7
5	OD	3	16	-0.67400	6
6	HD	21	1	0.40800	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.371. Atoms of building block SBS.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	18
5	6	1
7	8	27
8	9	5
8	10	10

TABLE 4.372. Bonds of building block SBS.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	13
3	4	5	13
4	5	6	12
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.373. Bond angles of building block SBS.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	4	5	6	23
3	7	8	10	42
3	7	8	10	45

TABLE 4.374. Dihedral angles of building block SBS.

I	J	K	L	Type
1	-1	3	2	1
4	1	7	3	2
8	7	10	9	1

TABLE 4.375. Improper dihedral angles of building block SBS.

Solute building block: (S)-β^3-Threonine
Name: SBT

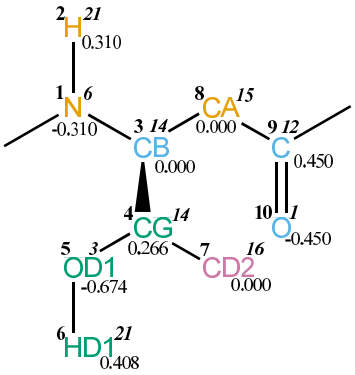


FIGURE 4.151. SBT non-bonded parameters.

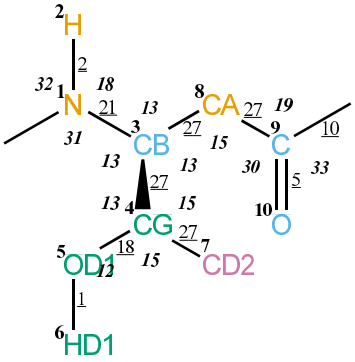


FIGURE 4.152. SBT bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 7 8 9
4	CG	14	3	0.26600	5 6 7 8
5	OD1	3	16	-0.67400	6 7
6	HD1	21	1	0.40800	
7	CD2	16	5	0.00000	
8	CA	15	4	0.00000	9 10 11
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.376. Atoms of building block SBT.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	18
4	7	27
5	6	1
8	9	27
9	10	5
9	11	10

TABLE 4.377. Bonds of building block SBT.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	13
3	4	5	13
3	4	7	15
5	4	7	15
4	5	6	12
3	8	9	15
8	9	10	30
8	9	11	19
10	9	11	33

TABLE 4.378. Bond angles of building block SBT.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	9	34
3	4	5	6	23
3	8	9	11	42
3	8	9	11	45

TABLE 4.379. Dihedral angles of building block SBT.

I	J	K	L	Type
1	-1	3	2	1
4	1	8	3	2
4	5	7	3	2
9	8	11	10	1

TABLE 4.380. Improper dihedral angles of building block SBT.

Solute building block: (S)-β^3-Valine
Name: SBV

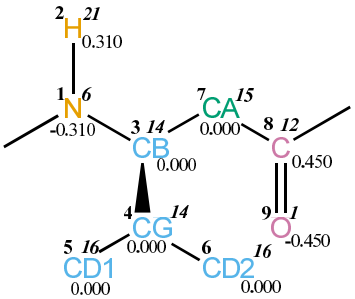


FIGURE 4.153. SBV non-bonded parameters.

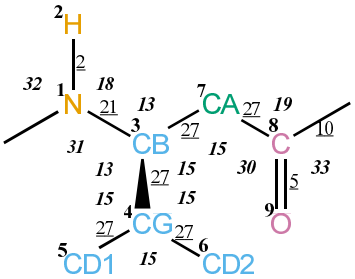


FIGURE 4.154. SBV bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 7 8
4	CG	14	3	0.00000	5 6 7
5	CD1	16	5	0.00000	6
6	CD2	16	5	0.00000	
7	CA	15	4	0.00000	8 9 10
8	C	12	12	0.45000	
9	O	1	16	-0.45000	

TABLE 4.381. Atoms of building block SBV.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	27
4	6	27
7	8	27
8	9	5
8	10	10

TABLE 4.382. Bonds of building block SBV.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	15
3	4	5	15
3	4	6	15
5	4	6	15
3	7	8	15
7	8	9	30
7	8	10	19
9	8	10	33

TABLE 4.383. Bond angles of building block SBV.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	8	34
3	7	8	10	42
3	7	8	10	45

TABLE 4.384. Dihedral angles of building block SBV.

I	J	K	L	Type
1	-1	3	2	1
3	5	6	4	2
4	1	7	3	2
8	7	10	9	1

TABLE 4.385. Improper dihedral angles of building block SBV.

Solute building block: (S)-β^3-Tyrosine
Name: SBY

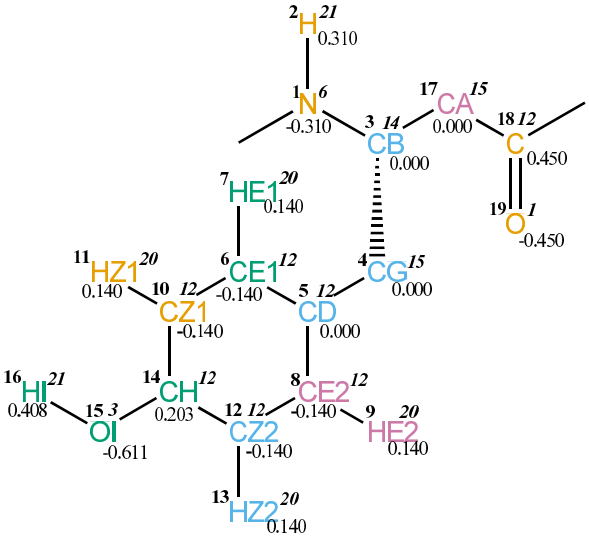


FIGURE 4.155. SBY non-bonded parameters.

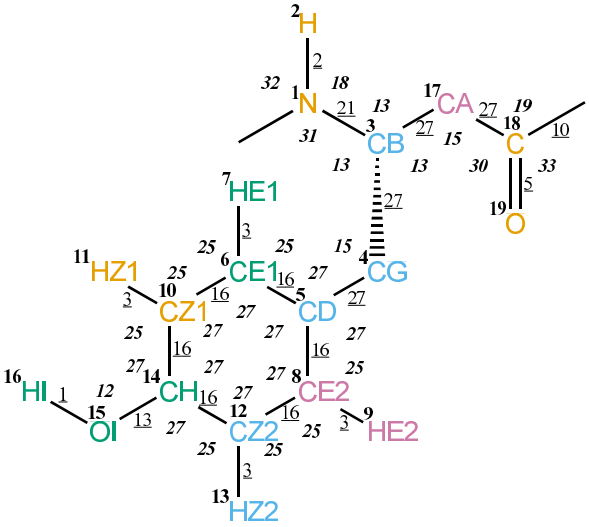


FIGURE 4.156. SBY bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 17
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 17 18
4	CG	15	4	0.00000	5 6 7 8 9 10 12 17
5	CD	12	12	0.00000	6 7 8 9 10 11 12 13 14
6	CE1	12	12	-0.14000	7 8 9 10 11 12 14 15
7	HE1	20	1	0.14000	8 10 11 14
8	CE2	12	12	-0.14000	9 10 12 13 14 15
9	HE2	20	1	0.14000	12 13 14
10	CZ1	12	12	-0.14000	11 12 13 14 15
11	HZ1	20	1	0.14000	12 14 15
12	CZ2	12	12	-0.14000	13 14 15
13	HZ2	20	1	0.14000	14 15
14	CH	12	12	0.20300	15 16
15	OI	3	16	-0.61100	16
16	HI	21	1	0.40800	
17	CA	15	4	0.00000	18 19 20
18	C	12	12	0.45000	
19	O	1	16	-0.45000	

TABLE 4.386. Atoms of building block SBY.

I	J	Type
1	2	2
1	3	21
3	4	27
3	17	27
4	5	27
5	6	16
5	8	16
6	7	3
6	10	16
8	9	3
8	12	16
10	11	3
10	14	16
12	13	3
12	14	16
14	15	13
15	16	1
17	18	27
18	19	5
18	20	10

TABLE 4.387. Bonds of building block SBY.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	17	13
4	3	17	13
3	4	5	15
4	5	6	27
4	5	8	27
6	5	8	27
5	6	7	25
5	6	10	27
7	6	10	25
5	8	9	25
5	8	12	27
9	8	12	25
6	10	11	25
6	10	14	27
11	10	14	25
8	12	13	25
8	12	14	27
13	12	14	25
10	14	12	27
10	14	15	27
12	14	15	27
14	15	16	12
3	17	18	15
17	18	19	30
17	18	20	19
19	18	20	33

TABLE 4.388. Bond angles of building block SBY.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	17	43
-1	1	3	17	44
1	3	4	5	34
1	3	17	18	34
3	4	5	6	40
10	14	15	16	11
3	17	18	20	42
3	17	18	20	45

TABLE 4.389. Dihedral angles of building block SBY.

I	J	K	L	Type
1	-1	3	2	1
3	1	17	4	2
5	6	8	4	1
5	6	10	14	1
5	8	12	14	1
6	5	8	12	1
6	5	10	7	1
6	10	14	12	1
8	5	6	10	1
8	5	12	9	1
8	12	14	10	1
11	6	14	10	1
13	8	14	12	1
14	10	12	15	1
18	17	20	19	1

TABLE 4.390. Improper dihedral angles of building block SBY.

Solute building block: (S)-β^3-Tryptophan
Name: SBW

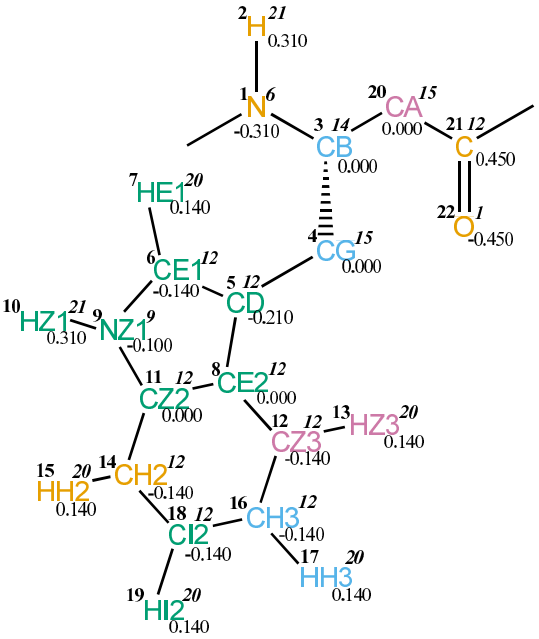


FIGURE 4.157. SBW non-bonded parameters.

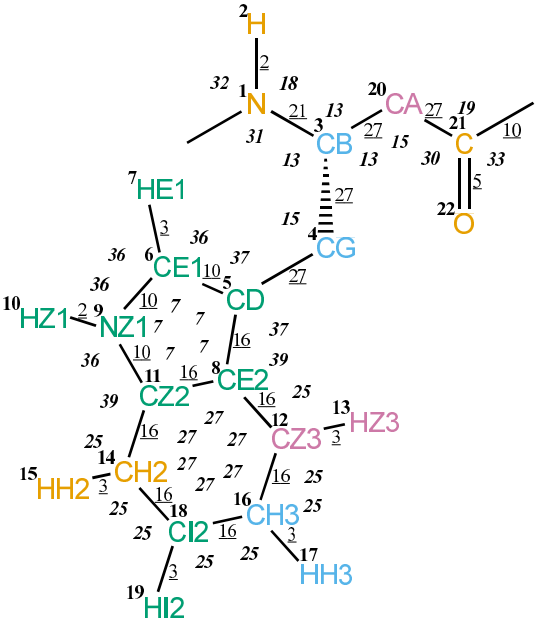


FIGURE 4.158. SBW bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 20
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 20 21
4	CG	15	4	0.00000	5 6 7 8 9 11 12 20
5	CD	12	12	-0.21000	6 7 8 9 10 11 12 13 14 16
6	CE1	12	12	-0.14000	7 8 9 10 11 12 14
7	HE1	20	1	0.14000	8 9 10 11
8	CE2	12	12	0.00000	9 10 11 12 13 14 15 16 17 18
9	NZ1	9	14	-0.10000	10 11 12 14 15 18
10	HZ1	21	1	0.31000	11 14
11	CZ2	12	12	0.00000	12 13 14 15 16 18 19
12	CZ3	12	12	-0.14000	13 14 16 17 18 19
13	HZ3	20	1	0.14000	16 17 18
14	CH2	12	12	-0.14000	15 16 17 18 19
15	HH2	20	1	0.14000	16 18 19
16	CH3	12	12	-0.14000	17 18 19
17	HH3	20	1	0.14000	18 19
18	CI2	12	12	-0.14000	19
19	HI2	20	1	0.14000	
20	CA	15	4	0.00000	21 22 23
21	C	12	12	0.45000	
22	O	1	16	-0.45000	

TABLE 4.391. Atoms of building block SBW.

I	J	Type
1	2	2
1	3	21
3	4	27
3	20	27
4	5	27
5	6	10
5	8	16
6	7	3
6	9	10
8	11	16
8	12	16
9	10	2
9	11	10
11	14	16
12	13	3
12	16	16
14	15	3
14	18	16
16	17	3
16	18	16
18	19	3
20	21	27
21	22	5
21	23	10

TABLE 4.392. Bonds of building block SBW.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	20	13
4	3	20	13
3	4	5	15
4	5	6	37
4	5	8	37
6	5	8	7
5	6	7	36
5	6	9	7
7	6	9	36
5	8	11	7
5	8	12	39
11	8	12	27
6	9	10	36
6	9	11	7
10	9	11	36
8	11	9	7
8	11	14	27
9	11	14	39
8	12	13	25
8	12	16	27
13	12	16	25
11	14	15	25
11	14	18	27
15	14	18	25
12	16	17	25
12	16	18	27
17	16	18	25
14	18	16	27
14	18	19	25
16	18	19	25
3	20	21	15
20	21	22	30
20	21	23	19
22	21	23	33

TABLE 4.393. Bond angles of building block SBW.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	20	43
-1	1	3	20	44
1	3	4	5	34
1	3	20	21	34
3	4	5	8	40
3	20	21	22	42
3	20	21	22	45

TABLE 4.394. Dihedral angles of building block SBW.

I	J	K	L	Type
1	-1	3	2	1
3	1	20	4	2
5	6	8	4	1
5	6	9	11	1
5	8	11	9	1
6	5	8	11	1
6	5	9	7	1
6	9	11	8	1
8	5	6	9	1
8	11	12	5	1
8	11	14	18	1
8	12	16	18	1
9	6	11	10	1
11	8	12	16	1
11	8	14	9	1
11	14	18	16	1
12	8	11	14	1
12	8	16	13	1
12	16	18	14	1
14	11	18	15	1
16	12	18	17	1
18	14	16	19	1
21	20	23	22	1

TABLE 4.395. Improper dihedral angles of building block SBW.

Solute building block: (R,S)-$\beta^{(}2,3$)-Alanine(αMe)
Name: SRAM

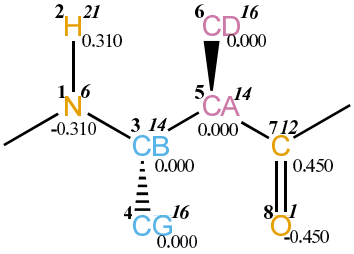


FIGURE 4.159. SRAM non-bonded parameters.

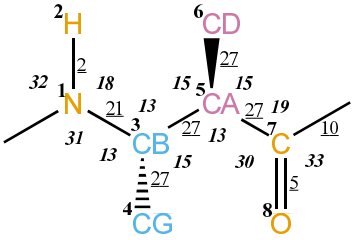


FIGURE 4.160. SRAM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 7
4	CG	16	5	0.00000	5
5	CA	14	3	0.00000	6 7 8 9
6	CD	16	5	0.00000	7
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.396. Atoms of building block SRAM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
5	6	27
5	7	27
7	8	5
7	9	10

TABLE 4.397. Bonds of building block SRAM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
4	3	5	15
3	5	6	15
3	5	7	13
6	5	7	15
5	7	8	30
5	7	9	19
8	7	9	33

TABLE 4.398. Bond angles of building block SRAM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	5	43
-1	1	3	5	44
1	3	5	7	34
3	5	7	9	42
3	5	7	9	45

TABLE 4.399. Dihedral angles of building block SRAM.

I	J	K	L	Type
1	-1	3	2	1
3	1	5	4	2
5	3	7	6	2
7	5	9	8	1

TABLE 4.400. Improper dihedral angles of building block SRAM.

Solute building block: (R,S)-$\beta^{(}2,3$)-Leucine(αMe)
Name: SRLM

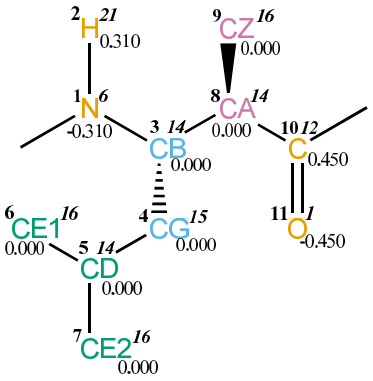


FIGURE 4.161. SRLM non-bonded parameters.

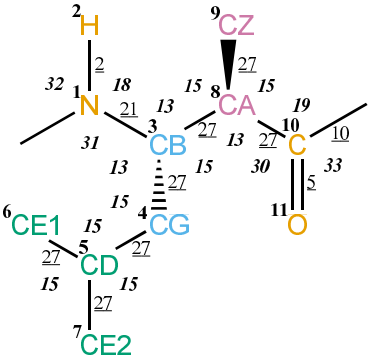


FIGURE 4.162. SRLM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 8
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 8 9 10
4	CG	15	4	0.00000	5 6 7 8
5	CD	14	3	0.00000	6 7
6	CE1	16	5	0.00000	7
7	CE2	16	5	0.00000	
8	CA	14	3	0.00000	9 10 11 12
9	CZ	16	5	0.00000	10
10	C	12	12	0.45000	
11	O	1	16	-0.45000	

TABLE 4.401. Atoms of building block SRLM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	8	27
4	5	27
5	6	27
5	7	27
8	9	27
8	10	27
10	11	5
10	12	10

TABLE 4.402. Bonds of building block SRLM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	8	13
4	3	8	15
3	4	5	15
4	5	6	15
4	5	7	15
6	5	7	15
3	8	9	15
3	8	10	13
9	8	10	15
8	10	11	30
8	10	12	19
11	10	12	33

TABLE 4.403. Bond angles of building block SRLM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	8	43
-1	1	3	8	44
1	3	4	5	34
1	3	8	10	34
3	4	5	6	34
3	8	10	12	42
3	8	10	12	45

TABLE 4.404. Dihedral angles of building block SRLM.

I	J	K	L	Type
1	-1	3	2	1
3	1	8	4	2
4	6	7	5	2
8	3	10	9	2
10	8	12	11	1

TABLE 4.405. Improper dihedral angles of building block SRLM.

Solute building block: (R,S)-$\beta^{(}2,3$)-Valine(αMe)
Name: SRVM

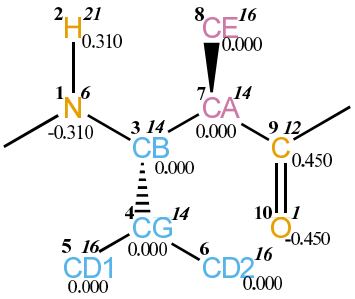


FIGURE 4.163. SRVM non-bonded parameters.

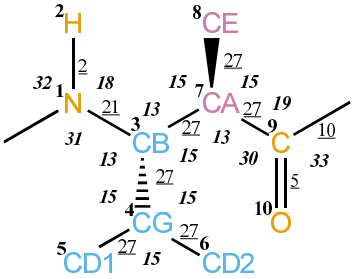


FIGURE 4.164. SRVM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 7
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 7 8 9
4	CG	14	3	0.00000	5 6 7
5	CD1	16	5	0.00000	6
6	CD2	16	5	0.00000	
7	CA	14	3	0.00000	8 9 10 11
8	CE	16	5	0.00000	9
9	C	12	12	0.45000	
10	O	1	16	-0.45000	

TABLE 4.406. Atoms of building block SRVM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	7	27
4	5	27
4	6	27
7	8	27
7	9	27
9	10	5
9	11	10

TABLE 4.407. Bonds of building block SRVM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	7	13
4	3	7	15
3	4	5	15
3	4	6	15
5	4	6	15
3	7	8	15
3	7	9	13
8	7	9	15
7	9	10	30
7	9	11	19
10	9	11	33

TABLE 4.408. Bond angles of building block SRVM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	7	43
-1	1	3	7	44
1	3	4	5	34
1	3	7	9	34
3	7	9	11	42
3	7	9	11	45

TABLE 4.409. Dihedral angles of building block SRVM.

I	J	K	L	Type
1	-1	3	2	1
3	1	7	4	2
3	5	6	4	2
7	3	9	8	2
9	7	11	10	1

TABLE 4.410. Improper dihedral angles of building block SRVM.

Solute building block: (S,S)-$\beta^{(}2,3$)-Alanine(αMe)
Name: SSAM

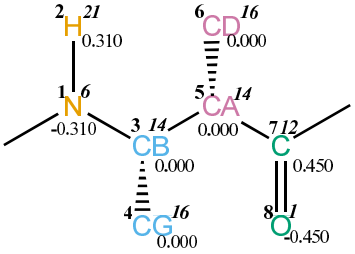


FIGURE 4.165. SSAM non-bonded parameters.

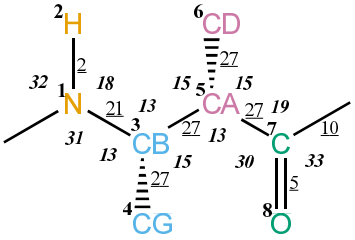


FIGURE 4.166. SSAM bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1 2 3
0					1
1	N	6	14	-0.31000	2 3 4 5
2	H	21	1	0.31000	3
3	CB	14	3	0.00000	4 5 6 7
4	CG	16	5	0.00000	5
5	CA	14	3	0.00000	6 7 8 9
6	CD	16	5	0.00000	7
7	C	12	12	0.45000	
8	O	1	16	-0.45000	

TABLE 4.411. Atoms of building block SSAM.

I	J	Type
1	2	2
1	3	21
3	4	27
3	5	27
5	6	27
5	7	27
7	8	5
7	9	10

TABLE 4.412. Bonds of building block SSAM.

I	J	K	Type
-1	1	2	32
-1	1	3	31
2	1	3	18
1	3	4	13
1	3	5	13
4	3	5	15
3	5	6	15
3	5	7	13
6	5	7	15
5	7	8	30
5	7	9	19
8	7	9	33

TABLE 4.413. Bond angles of building block SSAM.

I	J	K	L	Type
-2	-1	1	3	14
-1	1	3	5	43
-1	1	3	5	44
1	3	5	7	34
3	5	7	9	42
3	5	7	9	45

TABLE 4.414. Dihedral angles of building block SSAM.

I	J	K	L	Type
1	-1	3	2	1
3	1	5	4	2
6	3	7	5	2
7	5	9	8	1

TABLE 4.415. Improper dihedral angles of building block SSAM.

4.5. Nucleotides

Solute building block: 2'-deoxyadenosine 5'-phosphoric acid (DNA, charge -e)
Name: DADE

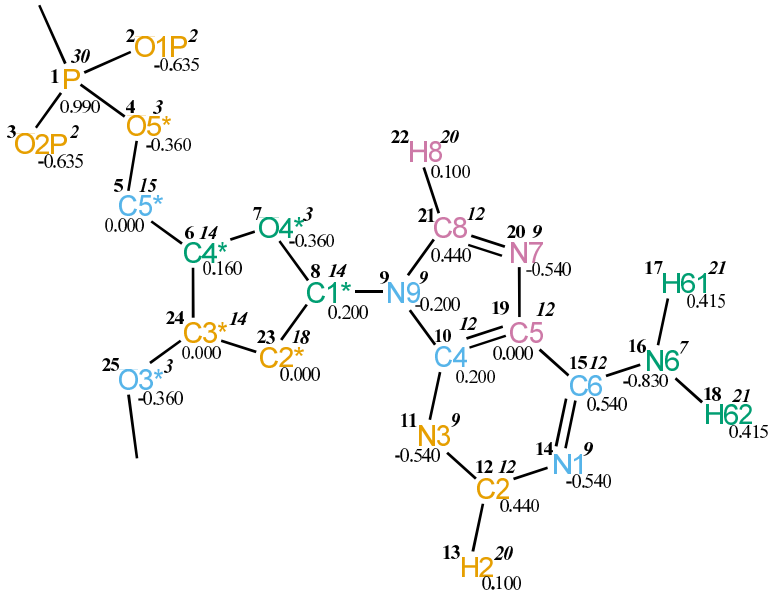


FIGURE 4.167. DADE non-bonded parameters.

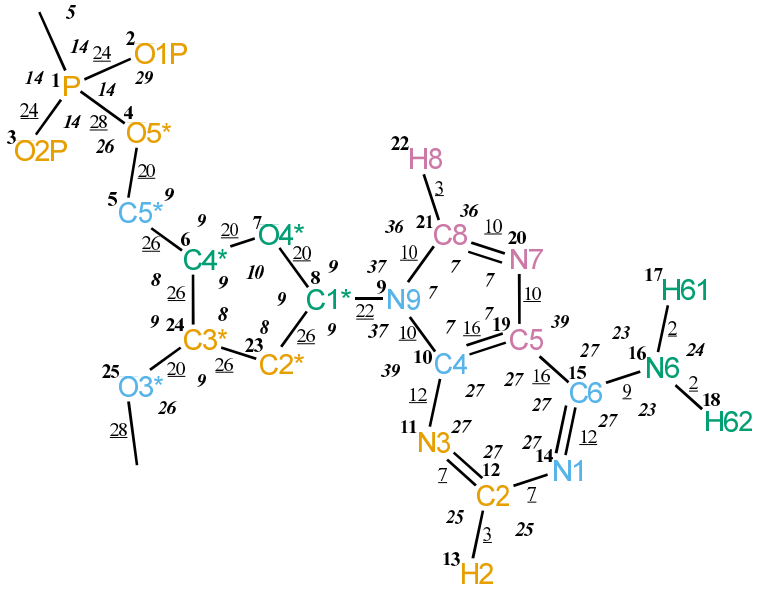


FIGURE 4.168. DADE bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 24
6	C4*	14	3	0.16000	7 8 23 24 25
7	O4*	3	16	-0.36000	8 9 23 24
8	C1*	14	3	0.20000	9 10 11 19 20 21 22 23 24
9	N9	9	14	-0.20000	10 11 12 15 19 20 21 22 23
10	C4	12	12	0.20000	11 12 13 14 15 16 19 20 21 22
11	N3	9	14	-0.54000	12 13 14 15 19 20 21
12	C2	12	12	0.44000	13 14 15 16 19
13	H2	20	1	0.10000	14 15
14	N1	9	14	-0.54000	15 16 19 20
15	C6	12	12	0.54000	16 17 18 19 20 21
16	N6	7	14	-0.83000	17 18 19 20
17	H61	21	1	0.41500	18 20
18	H62	21	1	0.41500	20
19	C5	12	12	0.00000	20 21 22
20	N7	9	14	-0.54000	21 22
21	C8	12	12	0.44000	22
22	H8	20	1	0.10000	
23	C2*	18	4	0.00000	24 25
24	C3*	14	3	0.00000	
25	O3*	3	16	-0.36000	

TABLE 4.416. Atoms of building block DADE.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	24	26
7	8	20
8	9	22
8	23	26
9	10	10
9	21	10
10	11	12
10	19	16
11	12	7
12	13	3
12	14	7
14	15	12
15	16	9
15	19	16
16	17	2
16	18	2
19	20	10
20	21	10
21	22	3
23	24	26
24	25	20
25	26	28

TABLE 4.417. Bonds of building block DADE.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	24	8
7	6	24	9
6	7	8	10
7	8	9	9
7	8	23	9
9	8	23	9
8	9	10	37
8	9	21	37
10	9	21	7
9	10	11	39
9	10	19	7
11	10	19	27
10	11	12	27
11	12	13	25
11	12	14	27
13	12	14	25
12	14	15	27
14	15	16	27
14	15	19	27
16	15	19	27
15	16	17	23
15	16	18	23
17	16	18	24
10	19	15	27
10	19	20	7
15	19	20	39
19	20	21	7
9	21	20	7
9	21	22	36
20	21	22	36
8	23	24	8
6	24	23	8
6	24	25	9
23	24	25	9
24	25	26	26

TABLE 4.418. Bond angles of building block DADE.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	24	17
4	5	6	24	34
24	6	7	8	29
5	6	24	23	34
5	6	24	25	17
7	6	24	23	17
7	6	24	25	18
6	7	8	23	29
7	8	9	10	16
7	8	23	24	17
7	8	23	24	34
19	15	16	17	14
8	23	24	6	34
8	23	24	25	17
6	24	25	26	29

TABLE 4.419. Dihedral angles of building block DADE.

I	J	K	L	Type
8	10	21	9	1
9	10	19	20	1
10	9	11	19	1
10	9	21	20	1
10	11	12	14	1
10	19	20	21	1
11	10	19	15	1
11	12	14	15	1
12	11	13	14	1
12	14	15	19	1
14	15	19	10	1
16	14	19	15	1
16	17	18	15	1
19	10	11	12	1
19	15	20	10	1
19	20	21	9	1
21	9	10	19	1
21	9	20	22	1
23	7	9	8	2
24	5	7	6	2
24	23	25	6	2

TABLE 4.420. Improper dihedral angles of building block DADE.

Solute building block: 2'-deoxyguanosine 5'-phosphoric acid (DNA, charge -*e*)
Name: DGUA

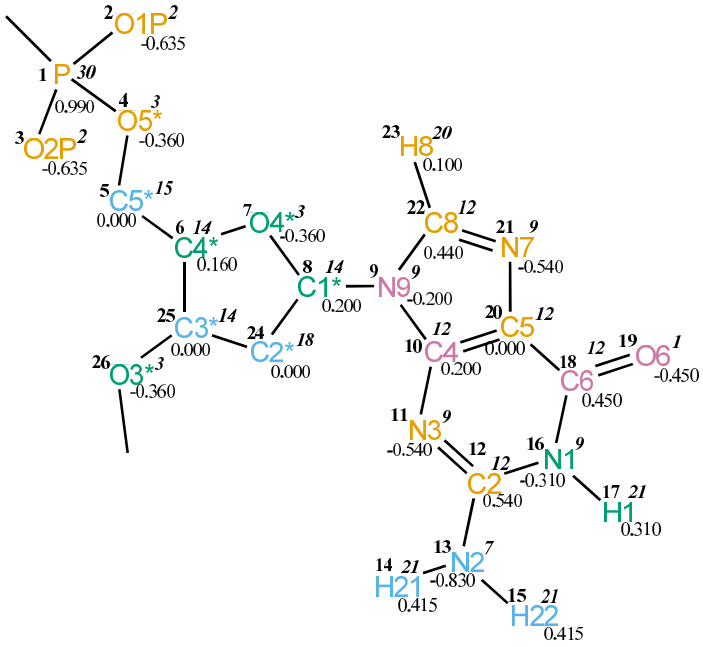


FIGURE 4.169. DGUA non-bonded parameters.

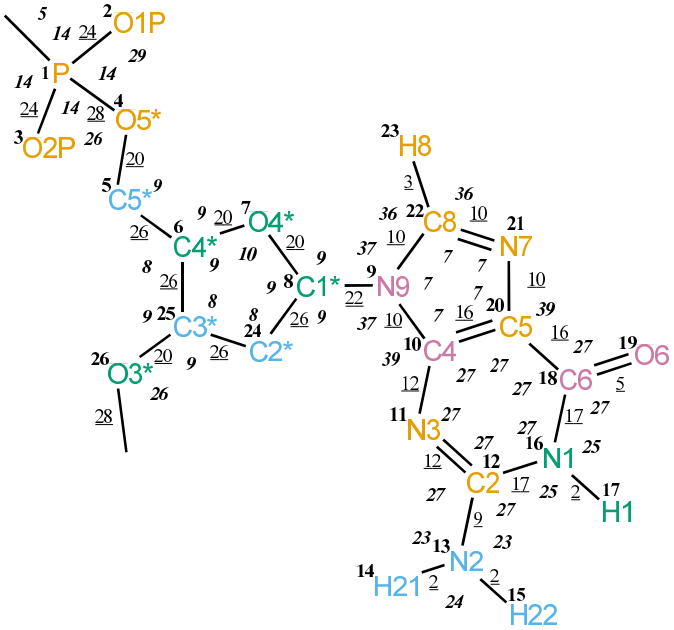


FIGURE 4.170. DGUA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 25
6	C4*	14	3	0.16000	7 8 24 25 26
7	O4*	3	16	-0.36000	8 9 24 25
8	C1*	14	3	0.20000	9 10 11 20 21 22 23 24 25
9	N9	9	14	-0.20000	10 11 12 18 20 21 22 23 24
10	C4	12	12	0.20000	11 12 13 16 18 19 20 21 22 23
11	N3	9	14	-0.54000	12 13 16 17 18 20 21 22
12	C2	12	12	0.54000	13 14 15 16 17 18 19 20
13	N2	7	14	-0.83000	14 15 16 17 18
14	H21	21	1	0.41500	15
15	H22	21	1	0.41500	
16	N1	9	14	-0.31000	17 18 19 20 21
17	H1	21	1	0.31000	18 19 20
18	C6	12	12	0.45000	19 20 21 22
19	O6	1	16	-0.45000	20 21
20	C5	12	12	0.00000	21 22 23
21	N7	9	14	-0.54000	22 23
22	C8	12	12	0.44000	23
23	H8	20	1	0.10000	
24	C2*	18	4	0.00000	25 26
25	C3*	14	3	0.00000	
26	O3*	3	16	-0.36000	

TABLE 4.421. Atoms of building block DGUA.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	25	26
7	8	20
8	9	22
8	24	26
9	10	10
9	22	10
10	11	12
10	20	16
11	12	12
12	13	9
12	16	17
13	14	2
13	15	2
16	17	2
16	18	17
18	19	5
18	20	16
20	21	10
21	22	10
22	23	3
24	25	26
25	26	20
26	27	28

TABLE 4.422. Bonds of building block DGUA.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	25	8
7	6	25	9
6	7	8	10
7	8	9	9
7	8	24	9
9	8	24	9
8	9	10	37
8	9	22	37
10	9	22	7
9	10	11	39
9	10	20	7
11	10	20	27
10	11	12	27
11	12	13	27
11	12	16	27
13	12	16	27
12	13	14	23
12	13	15	23
14	13	15	24
12	16	17	25
12	16	18	27
17	16	18	25
16	18	19	27
16	18	20	27
19	18	20	27
10	20	18	27
10	20	21	7
18	20	21	39
20	21	22	7
9	22	21	7
9	22	23	36
21	22	23	36
8	24	25	8
6	25	24	8
6	25	26	9
24	25	26	9
25	26	27	26

TABLE 4.423. Bond angles of building block DGUA.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	25	17
4	5	6	25	34
25	6	7	8	29
5	6	25	24	34
5	6	25	26	17
7	6	25	24	17
7	6	25	26	18
6	7	8	24	29
7	8	9	10	16
7	8	24	25	17
7	8	24	25	34
11	12	13	14	14
8	24	25	6	34
8	24	25	26	17
6	25	26	27	29

TABLE 4.424. Dihedral angles of building block DGUA.

I	J	K	L	Type
8	10	22	9	1
9	10	20	21	1
10	9	11	20	1
10	9	22	21	1
10	11	12	16	1
10	20	21	22	1
11	10	20	18	1
11	12	16	18	1
12	16	18	20	1
13	11	16	12	1
13	14	15	12	1
16	18	20	10	1
17	12	18	16	1
19	16	20	18	1
20	10	11	12	1
20	18	21	10	1
20	21	22	9	1
22	9	10	20	1
22	9	21	23	1
24	7	9	8	2
25	5	7	6	2
25	24	26	6	2

TABLE 4.425. Improper dihedral angles of building block DGUA.

Solute building block: 2'-deoxycytidine 5'-phosphoric acid (DNA, charge -e)
Name: DCYT

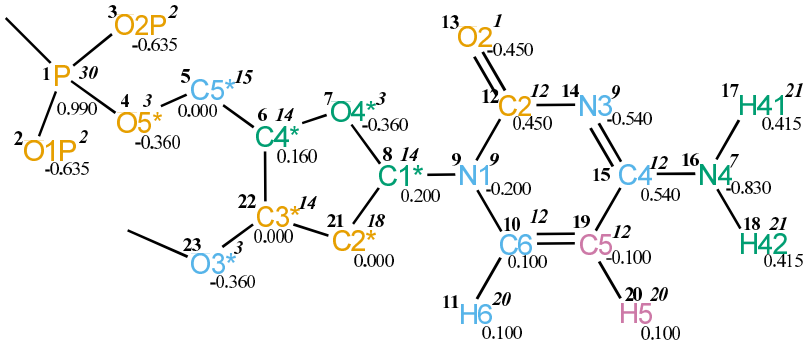


FIGURE 4.171. DCYT non-bonded parameters.

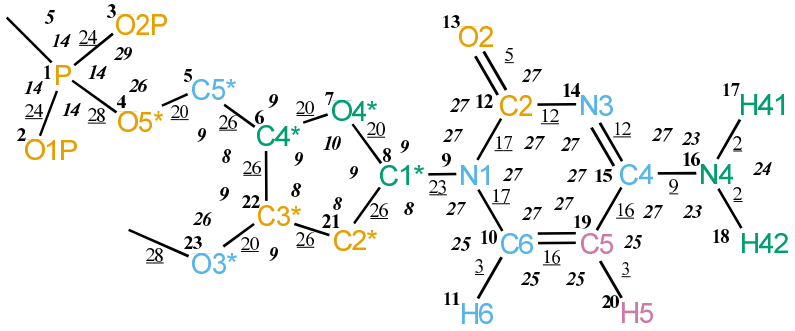


FIGURE 4.172. DCYT bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 22
6	C4*	14	3	0.16000	7 8 21 22 23
7	O4*	3	16	-0.36000	8 9 21 22
8	C1*	14	3	0.20000	9 10 11 12 13 14 19 21 22
9	N1	9	14	-0.20000	10 11 12 13 14 15 19 20 21
10	C6	12	12	0.10000	11 12 13 14 15 16 19 20
11	H6	20	1	0.10000	12 15 19 20
12	C2	12	12	0.45000	13 14 15 16 19
13	O2	1	16	-0.45000	14 15
14	N3	9	14	-0.54000	15 16 19 20
15	C4	12	12	0.54000	16 17 18 19 20
16	N4	7	14	-0.83000	17 18 19 20
17	H41	21	1	0.41500	18
18	H42	21	1	0.41500	
19	C5	12	12	-0.10000	20
20	H5	20	1	0.10000	
21	C2*	18	4	0.00000	22 23
22	C3*	14	3	0.00000	
23	O3*	3	16	-0.36000	

TABLE 4.426. Atoms of building block DCYT.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	22	26
7	8	20
8	9	23
8	21	26
9	10	17
9	12	17
10	11	3
10	19	16
12	13	5
12	14	12
14	15	12
15	16	9
15	19	16
16	17	2
16	18	2
19	20	3
21	22	26
22	23	20
23	24	28

TABLE 4.427. Bonds of building block DCYT.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	22	8
7	6	22	9
6	7	8	10
7	8	9	9
7	8	21	9
9	8	21	8
8	9	10	27
8	9	12	27
10	9	12	27
9	10	11	25
9	10	19	27
11	10	19	25
9	12	13	27
9	12	14	27
13	12	14	27
12	14	15	27
14	15	16	27
14	15	19	27
16	15	19	27
15	16	17	23
15	16	18	23
17	16	18	24
10	19	15	27
10	19	20	25
15	19	20	25
8	21	22	8
6	22	21	8
6	22	23	9
21	22	23	9
22	23	24	26

TABLE 4.428. Bond angles of building block DCYT.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	22	17
4	5	6	22	34
22	6	7	8	29
5	6	22	21	34
5	6	22	23	17
7	6	22	21	17
7	6	22	23	18
6	7	8	21	29
7	8	9	12	16
7	8	21	22	17
7	8	21	22	34
14	15	16	17	14
8	21	22	6	34
8	21	22	23	17
6	22	23	24	29

TABLE 4.429. Dihedral angles of building block DCYT.

I	J	K	L	Type
9	10	12	8	1
9	10	19	15	1
9	12	14	15	1
10	9	12	14	1
10	9	19	11	1
12	9	10	19	1
12	14	15	19	1
13	9	14	12	1
14	15	19	10	1
16	14	19	15	1
16	17	18	15	1
19	10	15	20	1
21	7	9	8	2
22	5	7	6	2
22	21	23	6	2

TABLE 4.430. Improper dihedral angles of building block DCYT.

Solute building block: 2'-deoxythymidine 5'-phosphoric acid (DNA, charge -*e*)
Name: DTHY

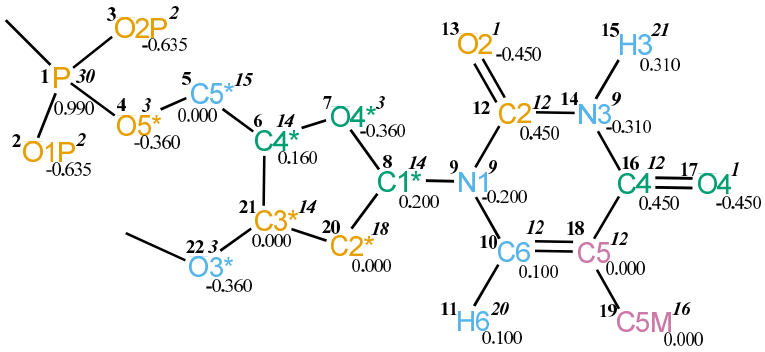


FIGURE 4.173. DTHY non-bonded parameters.

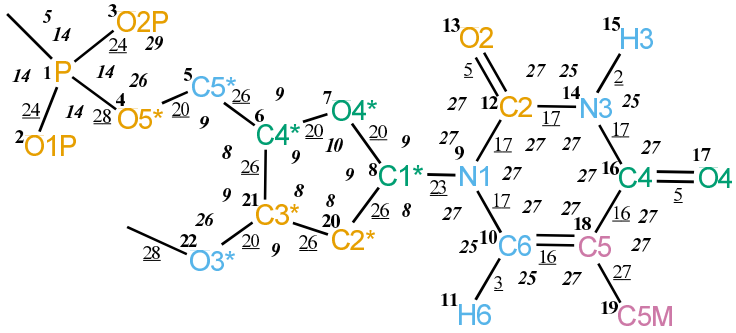


FIGURE 4.174. DTHY bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 21
6	C4*	14	3	0.16000	7 8 20 21 22
7	O4*	3	16	-0.36000	8 9 20 21
8	C1*	14	3	0.20000	9 10 11 12 13 14 18 20 21
9	N1	9	14	-0.20000	10 11 12 13 14 15 16 18 19 20
10	C6	12	12	0.10000	11 12 13 14 16 17 18 19
11	H6	20	1	0.10000	12 16 18 19
12	C2	12	12	0.45000	13 14 15 16 17 18
13	O2	1	16	-0.45000	14 15 16
14	N3	9	14	-0.31000	15 16 17 18 19
15	H3	21	1	0.31000	16 17 18
16	C4	12	12	0.45000	17 18 19
17	O4	1	16	-0.45000	18 19
18	C5	12	12	0.00000	19
19	C5M	16	5	0.00000	
20	C2*	18	4	0.00000	21 22
21	C3*	14	3	0.00000	
22	O3*	3	16	-0.36000	

TABLE 4.431. Atoms of building block DTHY.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	21	26
7	8	20
8	9	23
8	20	26
9	10	17
9	12	17
10	11	3
10	18	16
12	13	5
12	14	17
14	15	2
14	16	17
16	17	5
16	18	16
18	19	27
20	21	26
21	22	20
22	23	28

TABLE 4.432. Bonds of building block DTHY.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	21	8
7	6	21	9
6	7	8	10
7	8	9	9
7	8	20	9
9	8	20	8
8	9	10	27
8	9	12	27
10	9	12	27
9	10	11	25
9	10	18	27
11	10	18	25
9	12	13	27
9	12	14	27
13	12	14	27
12	14	15	25
12	14	16	27
15	14	16	25
14	16	17	27
14	16	18	27
17	16	18	27
10	18	16	27
10	18	19	27
16	18	19	27
8	20	21	8
6	21	20	8
6	21	22	9
20	21	22	9
21	22	23	26

TABLE 4.433. Bond angles of building block DTHY.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	21	17
4	5	6	21	34
21	6	7	8	29
5	6	21	20	34
5	6	21	22	17
7	6	21	20	17
7	6	21	22	18
6	7	8	20	29
7	8	9	12	16
7	8	20	21	17
7	8	20	21	34
8	20	21	6	34
8	20	21	22	17
6	21	22	23	29

TABLE 4.434. Dihedral angles of building block DTHY.

I	J	K	L	Type
9	10	12	8	1
9	10	18	16	1
9	12	14	16	1
10	9	12	14	1
10	9	18	11	1
12	9	10	18	1
12	14	16	18	1
13	9	14	12	1
14	16	18	10	1
15	12	16	14	1
17	14	18	16	1
18	10	16	19	1
20	7	9	8	2
21	5	7	6	2
21	20	22	6	2

TABLE 4.435. Improper dihedral angles of building block DTHY.

Solute building block: adenosine 5'-phosphoric acid (RNA, charge -e)
Name: ADE

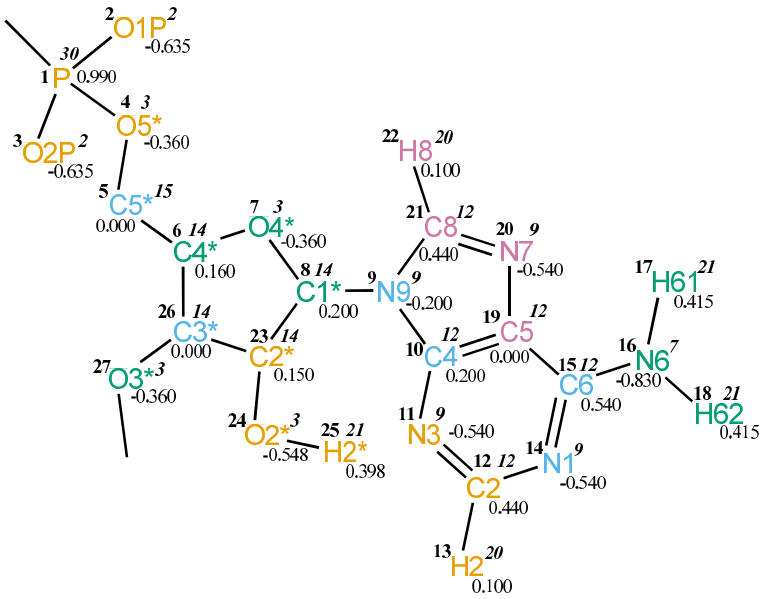


FIGURE 4.175. ADE non-bonded parameters.

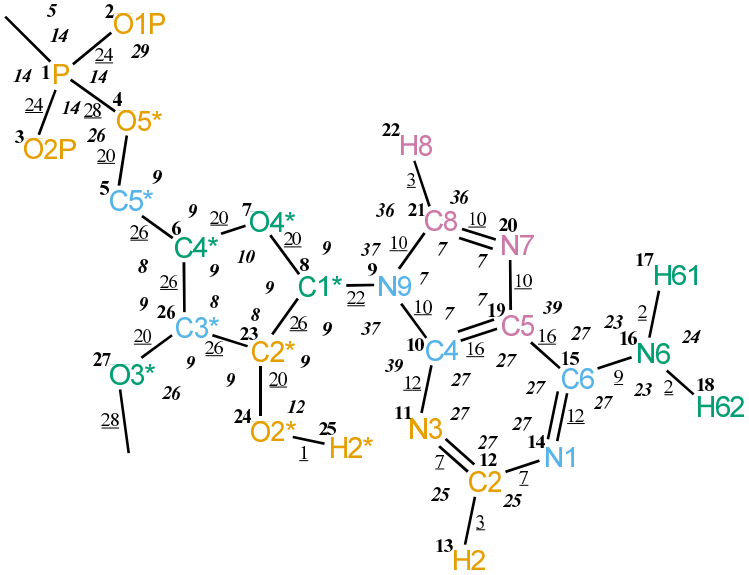


FIGURE 4.176. ADE bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 26
6	C4*	14	3	0.16000	7 8 23 26 27
7	O4*	3	16	-0.36000	8 9 23 26
8	C1*	14	3	0.20000	9 10 11 19 20 21 22 23 24 26
9	N9	9	14	-0.20000	10 11 12 15 19 20 21 22 23
10	C4	12	12	0.20000	11 12 13 14 15 16 19 20 21 22
11	N3	9	14	-0.54000	12 13 14 15 19 20 21
12	C2	12	12	0.44000	13 14 15 16 19
13	H2	20	1	0.10000	14 15
14	N1	9	14	-0.54000	15 16 19 20
15	C6	12	12	0.54000	16 17 18 19 20 21
16	N6	7	14	-0.83000	17 18 19 20
17	H61	21	1	0.41500	18 20
18	H62	21	1	0.41500	20
19	C5	12	12	0.00000	20 21 22
20	N7	9	14	-0.54000	21 22
21	C8	12	12	0.44000	22
22	H8	20	1	0.10000	
23	C2*	14	3	0.15000	24 25 26 27
24	O2*	3	16	-0.54800	25 26
25	H2*	21	1	0.39800	
26	C3*	14	3	0.00000	
27	O3*	3	16	-0.36000	

TABLE 4.436. Atoms of building block ADE.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	26	26
7	8	20
8	9	22
8	23	26
9	10	10
9	21	10
10	11	12
10	19	16
11	12	7
12	13	3
12	14	7
14	15	12
15	16	9
15	19	16
16	17	2
16	18	2
19	20	10
20	21	10
21	22	3
23	24	20
23	26	26
24	25	1
26	27	20
27	28	28

TABLE 4.437. Bonds of building block ADE.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	26	8
7	6	26	9
6	7	8	10
7	8	9	9
7	8	23	9
9	8	23	9
8	9	10	37
8	9	21	37
10	9	21	7
9	10	11	39
9	10	19	7
11	10	19	27
10	11	12	27
11	12	13	25
11	12	14	27
13	12	14	25
12	14	15	27
14	15	16	27
14	15	19	27
16	15	19	27
15	16	17	23
15	16	18	23
17	16	18	24
10	19	15	27
10	19	20	7
15	19	20	39
19	20	21	7
9	21	20	7
9	21	22	36
20	21	22	36
8	23	24	9
8	23	26	8
24	23	26	9
23	24	25	12
6	26	23	8
6	26	27	9
23	26	27	9
26	27	28	26

TABLE 4.438. Bond angles of building block ADE.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	26	17
4	5	6	26	34
26	6	7	8	29
5	6	26	23	34
5	6	26	27	17
7	6	26	23	17
7	6	26	27	18
6	7	8	23	29
7	8	9	10	16
7	8	23	24	18
7	8	23	26	17
7	8	23	26	34
9	8	23	24	17
19	15	16	17	14
8	23	24	25	23
8	23	26	6	34
8	23	26	27	17
24	23	26	6	17
24	23	26	27	18
6	26	27	28	29

TABLE 4.439. Dihedral angles of building block ADE.

I	J	K	L	Type
8	10	21	9	1
9	10	19	20	1
10	9	11	19	1
10	9	21	20	1
10	11	12	14	1
10	19	20	21	1
11	10	19	15	1
11	12	14	15	1
12	11	13	14	1
12	14	15	19	1
14	15	19	10	1
16	14	19	15	1
16	17	18	15	1
19	10	11	12	1
19	15	20	10	1
19	20	21	9	1
21	9	10	19	1
21	9	20	22	1
23	7	9	8	2
23	24	26	8	2
26	5	7	6	2
26	23	27	6	2

TABLE 4.440. Improper dihedral angles of building block ADE.

Solute building block: guanosine 5'-phosphoric acid (RNA, charge -*e*)
Name: GUA

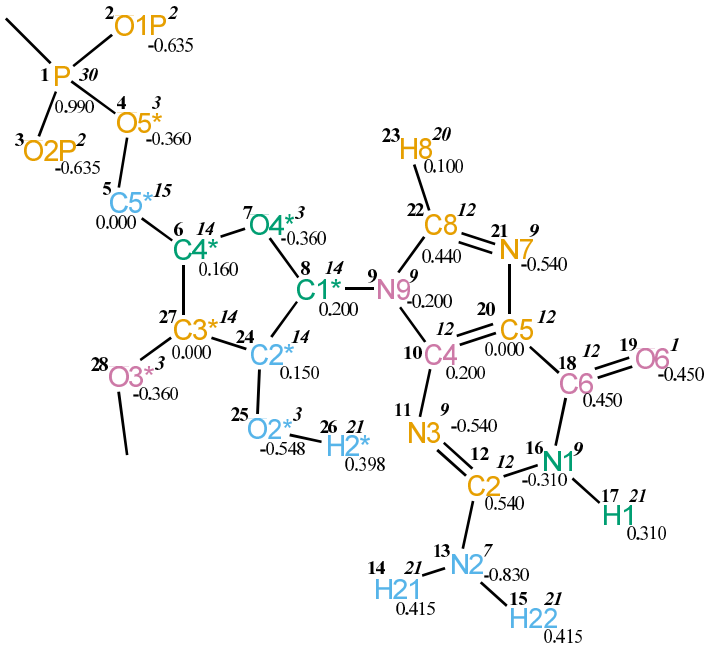


FIGURE 4.177. GUA non-bonded parameters.

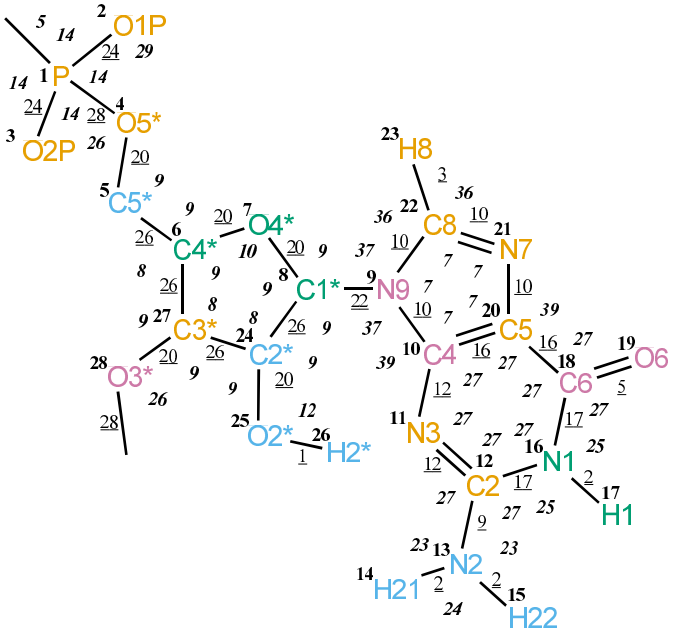


FIGURE 4.178. GUA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 27
6	C4*	14	3	0.16000	7 8 24 27 28
7	O4*	3	16	-0.36000	8 9 24 27
8	C1*	14	3	0.20000	9 10 11 20 21 22 23 24 25 27
9	N9	9	14	-0.20000	10 11 12 18 20 21 22 23 24
10	C4	12	12	0.20000	11 12 13 16 18 19 20 21 22 23
11	N3	9	14	-0.54000	12 13 16 17 18 20 21 22
12	C2	12	12	0.54000	13 14 15 16 17 18 19 20
13	N2	7	14	-0.83000	14 15 16 17 18
14	H21	21	1	0.41500	15
15	H22	21	1	0.41500	
16	N1	9	14	-0.31000	17 18 19 20 21
17	H1	21	1	0.31000	18 19 20
18	C6	12	12	0.45000	19 20 21 22
19	O6	1	16	-0.45000	20 21
20	C5	12	12	0.00000	21 22 23
21	N7	9	14	-0.54000	22 23
22	C8	12	12	0.44000	23
23	H8	20	1	0.10000	
24	C2*	14	3	0.15000	25 26 27 28
25	O2*	3	16	-0.54800	26 27
26	H2*	21	1	0.39800	
27	C3*	14	3	0.00000	
28	O3*	3	16	-0.36000	

TABLE 4.441. Atoms of building block GUA.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26

TABLE 4.443: continues on next page.

I	J	K	Type
4	5	6	9
5	6	7	9
5	6	27	8
7	6	27	9
6	7	8	10
7	8	9	9
7	8	24	9
9	8	24	9
8	9	10	37
8	9	22	37
10	9	22	7
9	10	11	39
9	10	20	7
11	10	20	27
10	11	12	27
11	12	13	27
11	12	16	27
13	12	16	27
12	13	14	23
12	13	15	23
14	13	15	24
12	16	17	25
12	16	18	27
17	16	18	25
16	18	19	27
16	18	20	27
19	18	20	27
10	20	18	27
10	20	21	7
18	20	21	39
20	21	22	7
9	22	21	7
9	22	23	36
21	22	23	36
8	24	25	9
8	24	27	8
25	24	27	9
24	25	26	12
6	27	24	8
6	27	28	9
24	27	28	9
27	28	29	26

TABLE 4.443: Bond angles of building block GUA.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	27	26
7	8	20
8	9	22
8	24	26
9	10	10
9	22	10
10	11	12
10	20	16
11	12	12
12	13	9
12	16	17
13	14	2
13	15	2
16	17	2
16	18	17
18	19	5
18	20	16
20	21	10
21	22	10
22	23	3
24	25	20
24	27	26
25	26	1
27	28	20
28	29	28

TABLE 4.442. Bonds of building block GUA.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	27	17
4	5	6	27	34
27	6	7	8	29
5	6	27	24	34
5	6	27	28	17
7	6	27	24	17
7	6	27	28	18
6	7	8	24	29
7	8	9	10	16
7	8	24	25	18
7	8	24	27	17
7	8	24	27	34
9	8	24	25	17
11	12	13	14	14
8	24	25	26	23
8	24	27	6	34
8	24	27	28	17
25	24	27	6	17
25	24	27	28	18
6	27	28	29	29

TABLE 4.444. Dihedral angles of building block GUA.

I	J	K	L	Type
8	10	22	9	1
9	10	20	21	1
10	9	11	20	1
10	9	22	21	1
10	11	12	16	1
10	20	21	22	1
11	10	20	18	1
11	12	16	18	1
12	16	18	20	1
13	11	16	12	1
13	14	15	12	1
16	18	20	10	1
17	12	18	16	1
19	16	20	18	1
20	10	11	12	1
20	18	21	10	1
20	21	22	9	1
22	9	10	20	1
22	9	21	23	1
24	7	9	8	2
24	25	27	8	2
27	5	7	6	2
27	24	28	6	2

TABLE 4.445. Improper dihedral angles of building block GUA.

Solute building block: cytidine 5'-phosphoric acid (RNA, charge -e)
Name: CYT

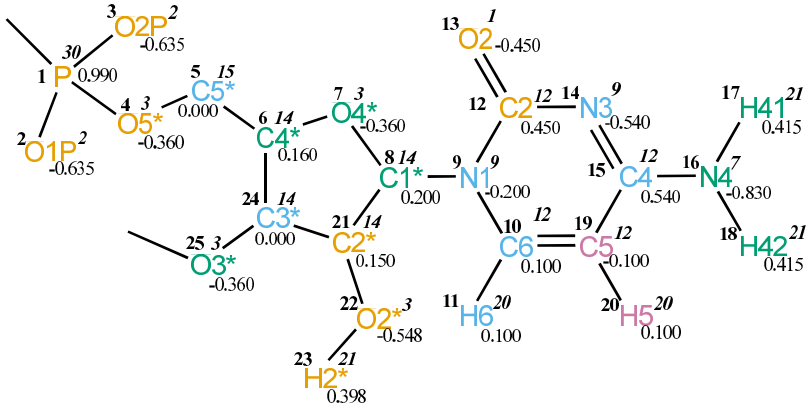


FIGURE 4.179. CYT non-bonded parameters.

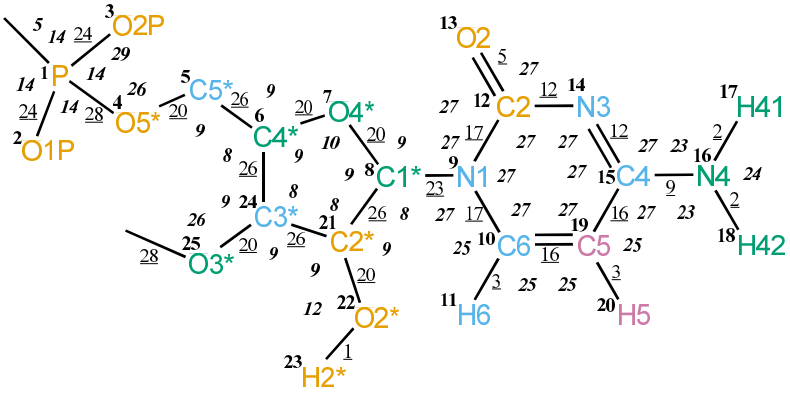


FIGURE 4.180. CYT bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 24
6	C4*	14	3	0.16000	7 8 21 24 25
7	O4*	3	16	-0.36000	8 9 21 24
8	C1*	14	3	0.20000	9 10 11 12 13 14 19 21 22 24
9	N1	9	14	-0.20000	10 11 12 13 14 15 19 20 21
10	C6	12	12	0.10000	11 12 13 14 15 16 19 20
11	H6	20	1	0.10000	12 15 19 20
12	C2	12	12	0.45000	13 14 15 16 19
13	O2	1	16	-0.45000	14 15
14	N3	9	14	-0.54000	15 16 19 20
15	C4	12	12	0.54000	16 17 18 19 20
16	N4	7	14	-0.83000	17 18 19 20
17	H41	21	1	0.41500	18
18	H42	21	1	0.41500	
19	C5	12	12	-0.10000	20
20	H5	20	1	0.10000	
21	C2*	14	3	0.15000	22 23 24 25
22	O2*	3	16	-0.54800	23 24
23	H2*	21	1	0.39800	
24	C3*	14	3	0.00000	
25	O3*	3	16	-0.36000	

TABLE 4.446. Atoms of building block CYT.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	24	26
7	8	20
8	9	23
8	21	26
9	10	17
9	12	17
10	11	3
10	19	16
12	13	5
12	14	12
14	15	12
15	16	9
15	19	16
16	17	2
16	18	2
19	20	3
21	22	20
21	24	26
22	23	1
24	25	20
25	26	28

TABLE 4.447. Bonds of building block CYT.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	24	8
7	6	24	9
6	7	8	10
7	8	9	9
7	8	21	9
9	8	21	8
8	9	10	27
8	9	12	27
10	9	12	27
9	10	11	25
9	10	19	27
11	10	19	25
9	12	13	27
9	12	14	27
13	12	14	27
12	14	15	27
14	15	16	27
14	15	19	27
16	15	19	27
15	16	17	23
15	16	18	23
17	16	18	24
10	19	15	27
10	19	20	25
15	19	20	25
8	21	22	9
8	21	24	8
22	21	24	9
21	22	23	12
6	24	21	8
6	24	25	9
21	24	25	9
24	25	26	26

TABLE 4.448. Bond angles of building block CYT.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	24	17
4	5	6	24	34
24	6	7	8	29
5	6	24	21	34
5	6	24	25	17
7	6	24	21	17
7	6	24	25	18
6	7	8	21	29
7	8	9	12	16
7	8	21	22	18
7	8	21	24	17
7	8	21	24	34
9	8	21	22	17
14	15	16	17	14
8	21	22	23	23
8	21	24	6	34
8	21	24	25	17
22	21	24	6	17
22	21	24	25	18
6	24	25	26	29

TABLE 4.449. Dihedral angles of building block CYT.

I	J	K	L	Type
9	10	12	8	1
9	10	19	15	1
9	12	14	15	1
10	9	12	14	1
10	9	19	11	1
12	9	10	19	1
12	14	15	19	1
13	9	14	12	1
14	15	19	10	1
16	14	19	15	1
16	17	18	15	1
19	10	15	20	1
21	7	9	8	2
21	22	24	8	2
24	5	7	6	2
24	21	25	6	2

TABLE 4.450. Improper dihedral angles of building block CYT.

Solute building block: uridine 5'-phosphoric acid (RNA, charge -e)
Name: URA

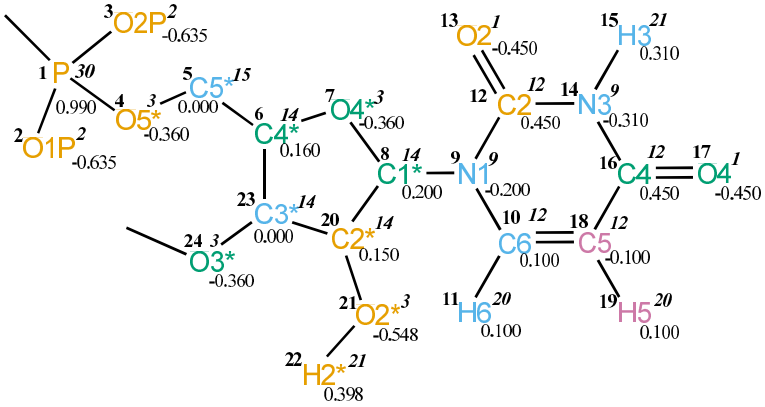


FIGURE 4.181. URA non-bonded parameters.

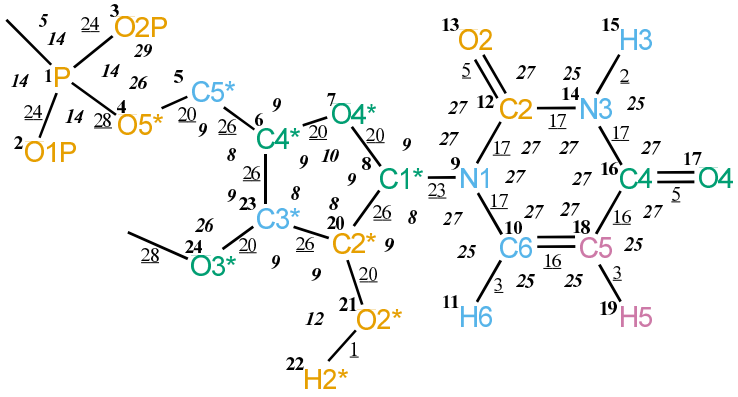


FIGURE 4.182. URA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 3 4
1	P	30	31	0.99000	2 3 4 5
2	O1P	2	16	-0.63500	3 4
3	O2P	2	16	-0.63500	4
4	O5*	3	16	-0.36000	5 6
5	C5*	15	4	0.00000	6 7 23
6	C4*	14	3	0.16000	7 8 20 23 24
7	O4*	3	16	-0.36000	8 9 20 23
8	C1*	14	3	0.20000	9 10 11 12 13 14 18 20 21 23
9	N1	9	14	-0.20000	10 11 12 13 14 15 16 18 19 20
10	C6	12	12	0.10000	11 12 13 14 16 17 18 19
11	H6	20	1	0.10000	12 16 18 19
12	C2	12	12	0.45000	13 14 15 16 17 18
13	O2	1	16	-0.45000	14 15 16
14	N3	9	14	-0.31000	15 16 17 18 19
15	H3	21	1	0.31000	16 17 18
16	C4	12	12	0.45000	17 18 19
17	O4	1	16	-0.45000	18 19
18	C5	12	12	-0.10000	19
19	H5	20	1	0.10000	
20	C2*	14	3	0.15000	21 22 23 24
21	O2*	3	16	-0.54800	22 23
22	H2*	21	1	0.39800	
23	C3*	14	3	0.00000	
24	O3*	3	16	-0.36000	

TABLE 4.451. Atoms of building block URA.

I	J	Type
1	2	24
1	3	24
1	4	28
4	5	20
5	6	26
6	7	20
6	23	26
7	8	20
8	9	23
8	20	26
9	10	17
9	12	17
10	11	3
10	18	16
12	13	5
12	14	17
14	15	2
14	16	17
16	17	5
16	18	16
18	19	3
20	21	20
20	23	26
21	22	1
23	24	20
24	25	28

TABLE 4.452. Bonds of building block URA.

I	J	K	Type
0	1	2	14
0	1	3	14
0	1	4	5
2	1	3	29
2	1	4	14
3	1	4	14
1	4	5	26
4	5	6	9
5	6	7	9
5	6	23	8
7	6	23	9
6	7	8	10
7	8	9	9
7	8	20	9
9	8	20	8
8	9	10	27
8	9	12	27
10	9	12	27
9	10	11	25
9	10	18	27
11	10	18	25
9	12	13	27
9	12	14	27
13	12	14	27
12	14	15	25
12	14	16	27
15	14	16	25
14	16	17	27
14	16	18	27
17	16	18	27
10	18	16	27
10	18	19	25
16	18	19	25
8	20	21	9
8	20	23	8
21	20	23	9
20	21	22	12
6	23	20	8
6	23	24	9
20	23	24	9
23	24	25	26

TABLE 4.453. Bond angles of building block URA.

I	J	K	L	Type
-1	0	1	4	20
-1	0	1	4	27
0	1	4	5	20
0	1	4	5	27
1	4	5	6	7
1	4	5	6	22
4	5	6	7	8
4	5	6	7	25
4	5	6	23	17
4	5	6	23	34
23	6	7	8	29
5	6	23	20	34
5	6	23	24	17
7	6	23	20	17
7	6	23	24	18
6	7	8	20	29
7	8	9	12	16
7	8	20	21	18
7	8	20	23	17
7	8	20	23	34
9	8	20	21	17
8	20	21	22	23
8	20	23	6	34
8	20	23	24	17
21	20	23	6	17
21	20	23	24	18
6	23	24	25	29

TABLE 4.454. Dihedral angles of building block URA.

I	J	K	L	Type
9	10	12	8	1
9	10	18	16	1
9	12	14	16	1
10	9	12	14	1
10	9	18	11	1
12	9	10	18	1
12	14	16	18	1
13	9	14	12	1
14	16	18	10	1
15	12	16	14	1
17	14	18	16	1
18	10	16	19	1
20	7	9	8	2
20	21	23	8	2
23	5	7	6	2
23	20	24	6	2

TABLE 4.455. Improper dihedral angles of building block URA.

Solute building block: flavin mononucleotide (oxydized,deprotonated at FN5 and FN1;charge -e, $OPOHO_2^-$)
Name: FMNO

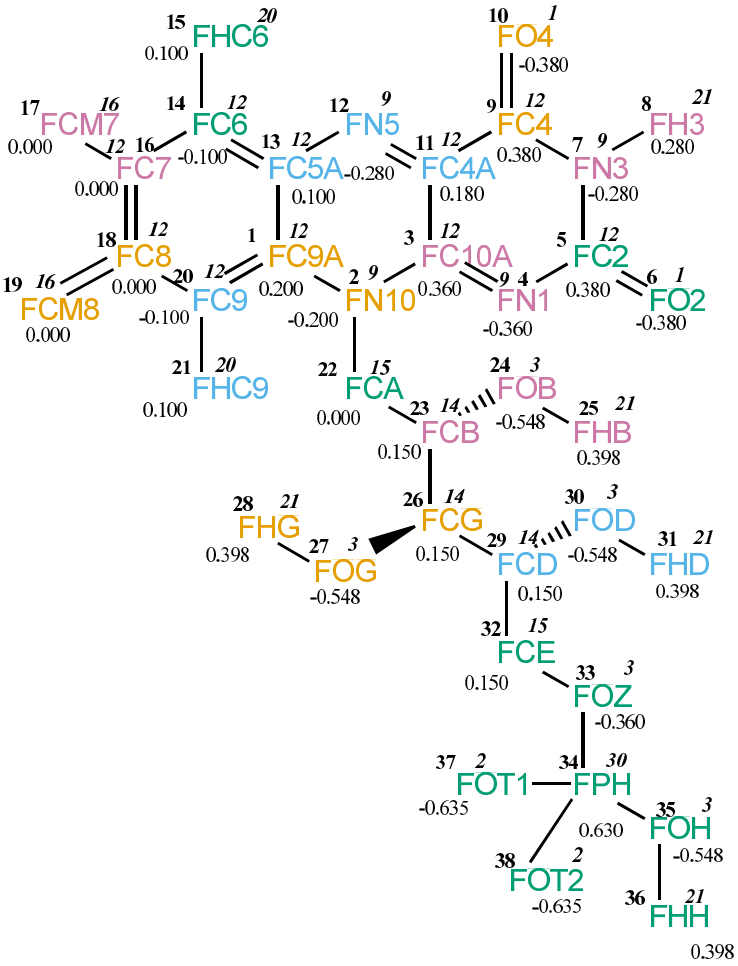


FIGURE 4.183. FMNO non-bonded parameters.

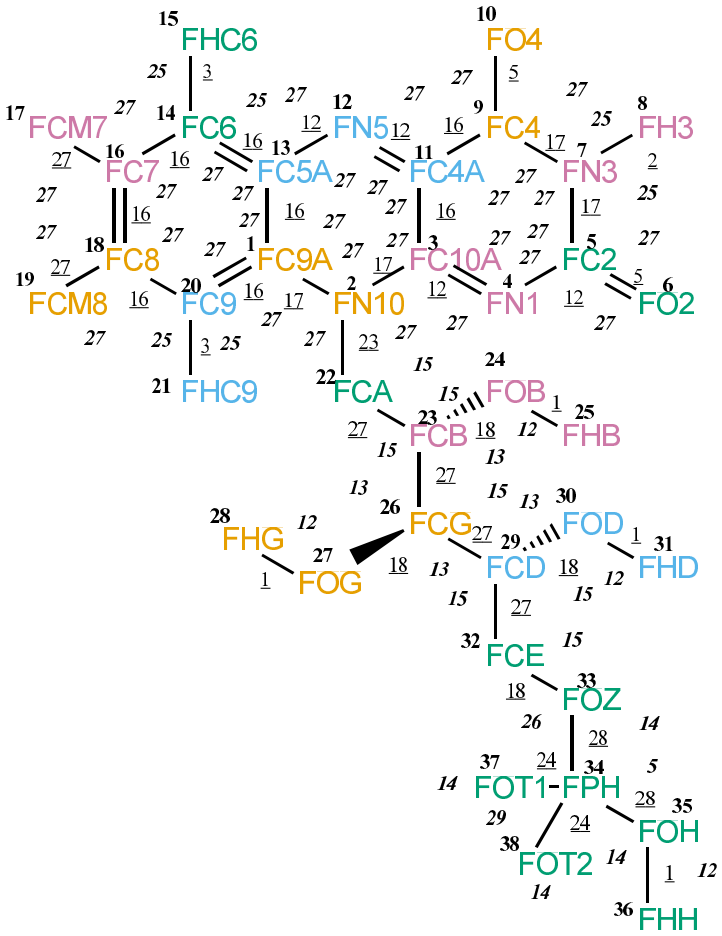


FIGURE 4.184. FMNO bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	FC9A	12	12	0.20000	2 3 4 11 12 13 14 15 16 18 19 20 21 22
2	FN10	9	14	-0.20000	3 4 5 9 11 12 13 14 18 20 21 22 23
3	FC10A	12	12	0.36000	4 5 6 7 9 10 11 12 13 20 22
4	FN1	9	14	-0.36000	5 6 7 8 9 11 12 22
5	FC2	12	12	0.38000	6 7 8 9 10 11
6	FO2	1	16	-0.38000	7 8 9
7	FN3	9	14	-0.28000	8 9 10 11 12
8	FH3	21	1	0.28000	9 10 11
9	FC4	12	12	0.38000	10 11 12 13
10	FO4	1	16	-0.38000	11 12
11	FC4A	12	12	0.18000	12 13 14 22
12	FN5	9	14	-0.28000	13 14 15 16 20
13	FC5A	12	12	0.10000	14 15 16 17 18 20 21 22
14	FC6	12	12	-0.10000	15 16 17 18 19 20
15	FHC6	20	1	0.10000	16 17 18
16	FC7	12	12	0.00000	17 18 19 20 21
17	FCM7	16	5	0.00000	18 19 20
18	FC8	12	12	0.00000	19 20 21
19	FCM8	16	5	0.00000	20 21
20	FC9	12	12	-0.10000	21 22
21	FHC9	20	1	0.10000	
22	FCA	15	4	0.00000	23 24 26
23	FCB	14	3	0.15000	24 25 26 27 29
24	FOB	3	16	-0.54800	25 26
25	FHB	21	1	0.39800	
26	FCG	14	3	0.15000	27 28 29 30 32
27	FOG	3	16	-0.54800	28 29
28	FHG	21	1	0.39800	
29	FCD	14	3	0.15000	30 31 32 33
30	FOD	3	16	-0.54800	31 32
31	FHD	21	1	0.39800	
32	FCE	15	4	0.15000	33 34
33	FOZ	3	16	-0.36000	34 35 36 37 38
34	FPH	30	31	0.63000	35 36 37 38
35	FOH	3	16	-0.54800	36 37 38
36	FHH	21	1	0.39800	37 38
37	FOT1	2	16	-0.63500	38
38	FOT2	2	16	-0.63500	

TABLE 4.456: Atoms of building block FMNO.

I	J	Type
1	2	17
1	13	16
1	20	16
2	3	17
2	22	23
3	4	12
3	11	16
4	5	12
5	6	5
5	7	17
7	8	2
7	9	17
9	10	5
9	11	16
11	12	12
12	13	12
13	14	16
14	15	3
14	16	16
16	17	27
16	18	16
18	19	27
18	20	16
20	21	3
22	23	27
23	24	18
23	26	27
24	25	1
26	27	18
26	29	27
27	28	1
29	30	18
29	32	27
30	31	1
32	33	18
33	34	28
34	35	28
34	37	24
34	38	24
35	36	1

TABLE 4.457: Bonds of building block FMNO.

I	J	K	Type
2	1	13	27
2	1	20	27
13	1	20	27
1	2	3	27
1	2	22	27
3	2	22	27
2	3	4	27
2	3	11	27
4	3	11	27
3	4	5	27
4	5	6	27
4	5	7	27
6	5	7	27
5	7	8	25
5	7	9	27
8	7	9	25
7	9	10	27
7	9	11	27
10	9	11	27
3	11	9	27
3	11	12	27
9	11	12	27
11	12	13	27
1	13	12	27
1	13	14	27
12	13	14	27
13	14	15	25
13	14	16	27
15	14	16	25
14	16	17	27
14	16	18	27
17	16	18	27
16	18	19	27
16	18	20	27
19	18	20	27
1	20	18	27
1	20	21	25
18	20	21	25
2	22	23	15
22	23	24	15
22	23	26	15
24	23	26	13
23	24	25	12
23	26	27	13
23	26	29	15
27	26	29	13

TABLE 4.458: continues on next page.

I	J	K	Type
26	27	28	12
26	29	30	13
26	29	32	15
30	29	32	15
29	30	31	12
29	32	33	15
32	33	34	26
33	34	35	5
33	34	37	14
33	34	38	14
35	34	37	14
35	34	38	14
37	34	38	29
34	35	36	12

TABLE 4.458: Bond angles of building block FMNO.

I	J	K	L	Type
13	1	2	3	14
1	2	3	11	14
1	2	22	23	40
3	11	12	13	14
11	12	13	1	14
2	22	23	26	34
22	23	24	25	23
22	23	26	29	34
23	26	27	28	23
23	26	29	32	34
26	29	30	31	23
26	29	32	33	34
29	32	33	34	7
29	32	33	34	22
32	33	34	35	20
32	33	34	35	27
33	34	35	36	20
33	34	35	36	27

TABLE 4.459: Dihedral angles of building block FMNO.

I	J	K	L	Type
1	2	20	13	1
1	13	14	16	1
2	1	13	12	1
2	3	11	12	1
3	4	5	7	1
4	3	11	9	1
4	5	7	9	1
5	4	7	6	1
5	7	9	11	1
7	5	9	8	1
7	9	11	3	1
9	7	11	10	1
11	2	4	3	1
11	3	4	5	1
11	9	12	3	1
13	1	20	18	1
13	12	14	1	1
13	14	16	18	1
14	16	18	20	1
15	13	16	14	1
16	14	18	17	1
16	18	20	1	1
18	16	20	19	1
20	1	13	14	1
20	1	18	21	1
23	24	26	22	2
26	27	29	23	2
29	30	32	26	2

TABLE 4.460: Improper dihedral angles of building block FMNO.

Solute building block: proflavin (protonated at FN5; charge $+e$)
Name: PFN

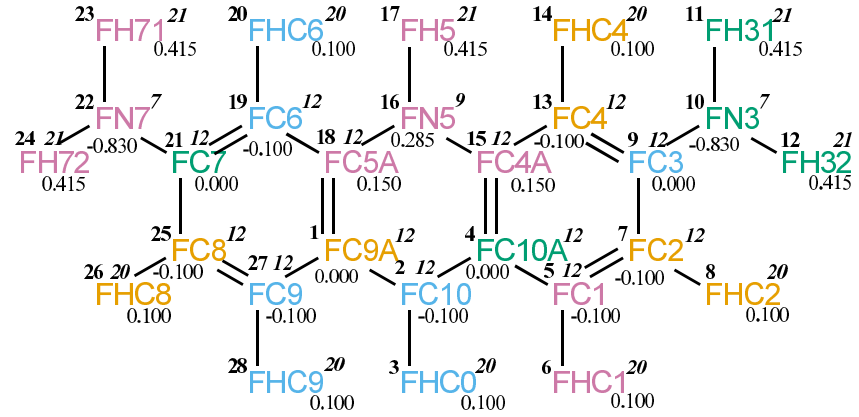


FIGURE 4.185. PFN non-bonded parameters.

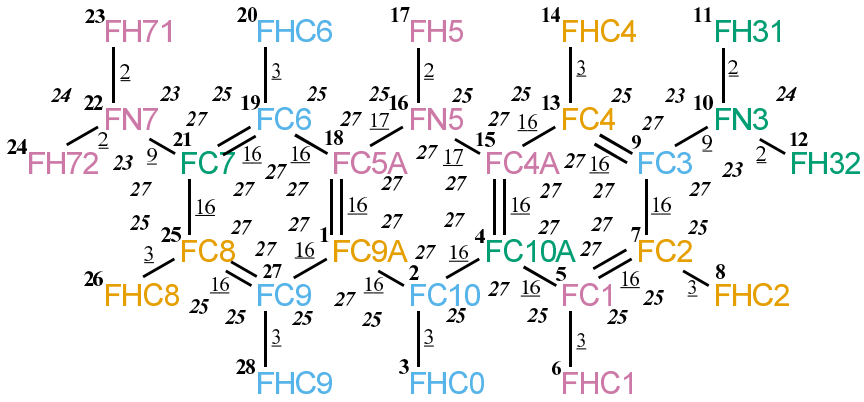


FIGURE 4.186. PFN bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	FC9A	12	12	0.00000	2 3 4 5 15 16 17 18 19 20 21 25 26 27 28
2	FC10	12	12	-0.10000	3 4 5 6 7 13 15 16 18 19 25 27 28
3	FHC0	20	1	0.10000	4 5 15 18 27
4	FC10A	12	12	0.00000	5 6 7 8 9 13 14 15 16 17 18 27
5	FC1	12	12	-0.10000	6 7 8 9 10 13 15 16
6	FHC1	20	1	0.10000	7 8 9 15
7	FC2	12	12	-0.10000	8 9 10 13 14 15
8	FHC2	20	1	0.10000	9 10 13
9	FC3	12	12	0.00000	10 11 12 13 14 15 16
10	FN3	7	14	-0.83000	11 12 13 14 15
11	FH31	21	1	0.41500	12
12	FH32	21	1	0.41500	
13	FC4	12	12	-0.10000	14 15 16 17 18
14	FHC4	20	1	0.10000	15 16
15	FC4A	12	12	0.15000	16 17 18 19
16	FN5	9	14	0.28500	17 18 19 20 21 27
17	FH5	21	1	0.41500	18 19
18	FC5A	12	12	0.15000	19 20 21 22 25 27 28
19	FC6	12	12	-0.10000	20 21 22 25 26 27
20	FHC6	20	1	0.10000	21 22 25
21	FC7	12	12	0.00000	22 23 24 25 26 27 28
22	FN7	7	14	-0.83000	23 24 25 26 27
23	FH71	21	1	0.41500	24
24	FH72	21	1	0.41500	
25	FC8	12	12	-0.10000	26 27 28
26	FHC8	20	1	0.10000	27 28
27	FC9	12	12	-0.10000	28
28	FHC9	20	1	0.10000	

TABLE 4.461: Atoms of building block PFN.

I	J	Type
1	2	16
1	18	16
1	27	16
2	3	3
2	4	16
4	5	16
4	15	16
5	6	3
5	7	16
7	8	3
7	9	16
9	10	9
9	13	16
10	11	2
10	12	2
13	14	3
13	15	16
15	16	17
16	17	2
16	18	17
18	19	16
19	20	3
19	21	16
21	22	9
21	25	16
22	23	2
22	24	2
25	26	3
25	27	16
27	28	3

TABLE 4.462: Bonds of building block PFN.

I	J	K	Type
2	1	18	27
2	1	27	27
18	1	27	27
1	2	3	25
1	2	4	27
3	2	4	25
2	4	5	27
2	4	15	27
5	4	15	27
4	5	6	25
4	5	7	27
6	5	7	25
5	7	8	25
5	7	9	27
8	7	9	25
7	9	10	27
7	9	13	27
10	9	13	27
9	10	11	23
9	10	12	23
11	10	12	24
9	13	14	25
9	13	15	27
14	13	15	25
4	15	13	27
4	15	16	27
13	15	16	27
15	16	17	25
15	16	18	27
17	16	18	25
1	18	16	27
1	18	19	27
16	18	19	27
18	19	20	25
18	19	21	27
20	19	21	25
19	21	22	27
19	21	25	27
22	21	25	27
21	22	23	23
21	22	24	23
23	22	24	24
21	25	26	25
21	25	27	27
26	25	27	25
1	27	25	27

TABLE 4.463: continues on next page.

I	J	K	Type
1	27	28	25
25	27	28	25

TABLE 4.463: Bond angles of building block PFN.

I	J	K	L	Type
7	9	10	11	14
25	21	22	23	14

TABLE 4.464: Dihedral angles of building block PFN.

I	J	K	L	Type
1	2	4	15	1
1	2	27	18	1
1	18	19	21	1
2	1	4	3	1
2	1	18	16	1
2	4	15	16	1
4	5	7	9	1
4	15	16	18	1
5	4	7	6	1
5	4	15	13	1
5	7	9	13	1
7	5	9	8	1
7	9	13	15	1
9	7	13	10	1
9	13	15	4	1
10	11	12	9	1
13	9	15	14	1
15	2	5	4	1
15	4	5	7	1
15	13	16	4	1
15	16	18	1	1
16	15	18	17	1
18	1	2	4	1
18	1	27	25	1
18	16	19	1	1
18	19	21	25	1
19	18	21	20	1
19	21	25	27	1
21	25	27	1	1
22	19	25	21	1
22	23	24	21	1
25	21	27	26	1
27	1	18	19	1
27	1	25	28	1

TABLE 4.465: Improper dihedral angles of building block PFN.

Solute building block: nicotinamide adenine dinucleotide (NAD+; charge -*e*)
Name: NADP

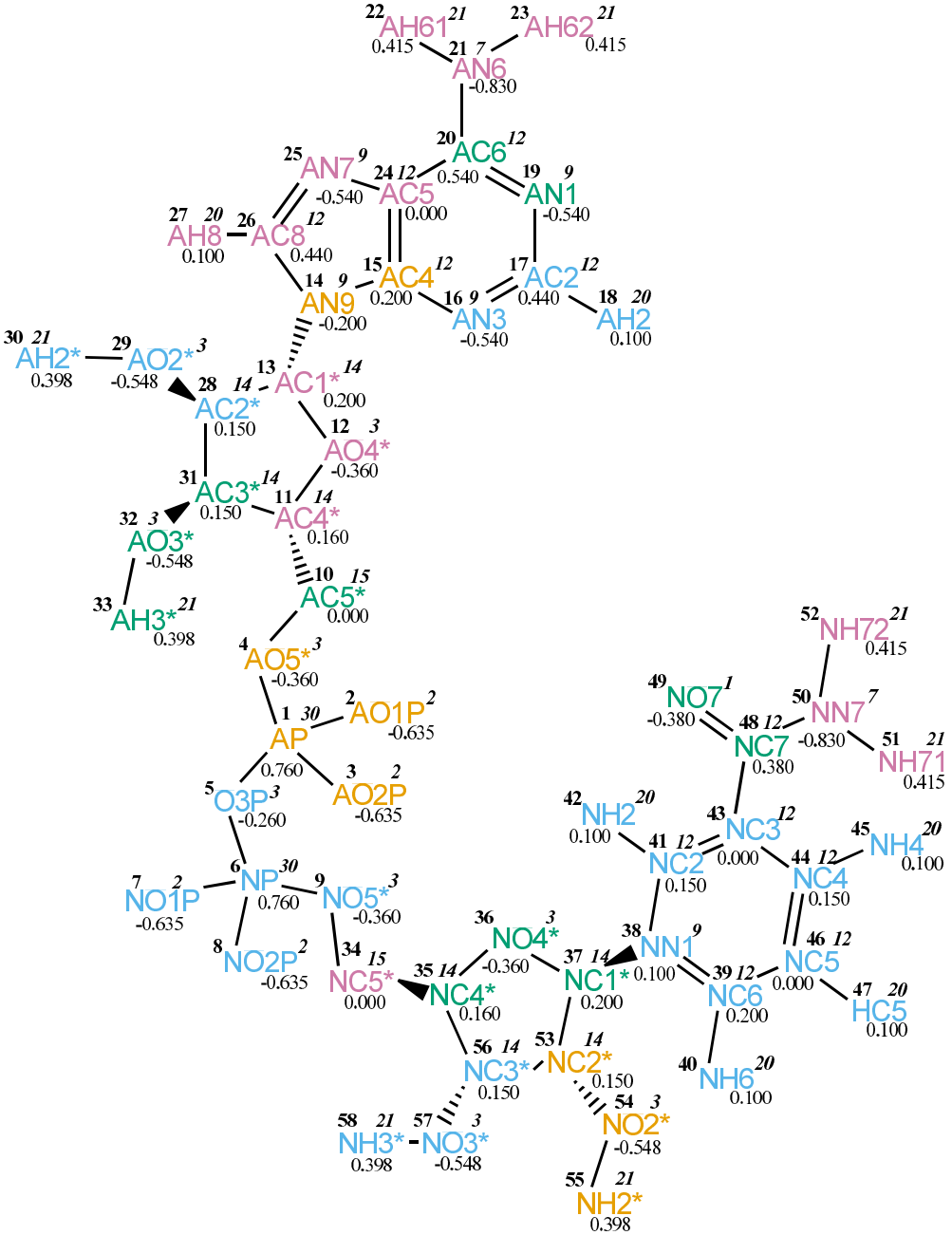


FIGURE 4.187. NADP non-bonded parameters.

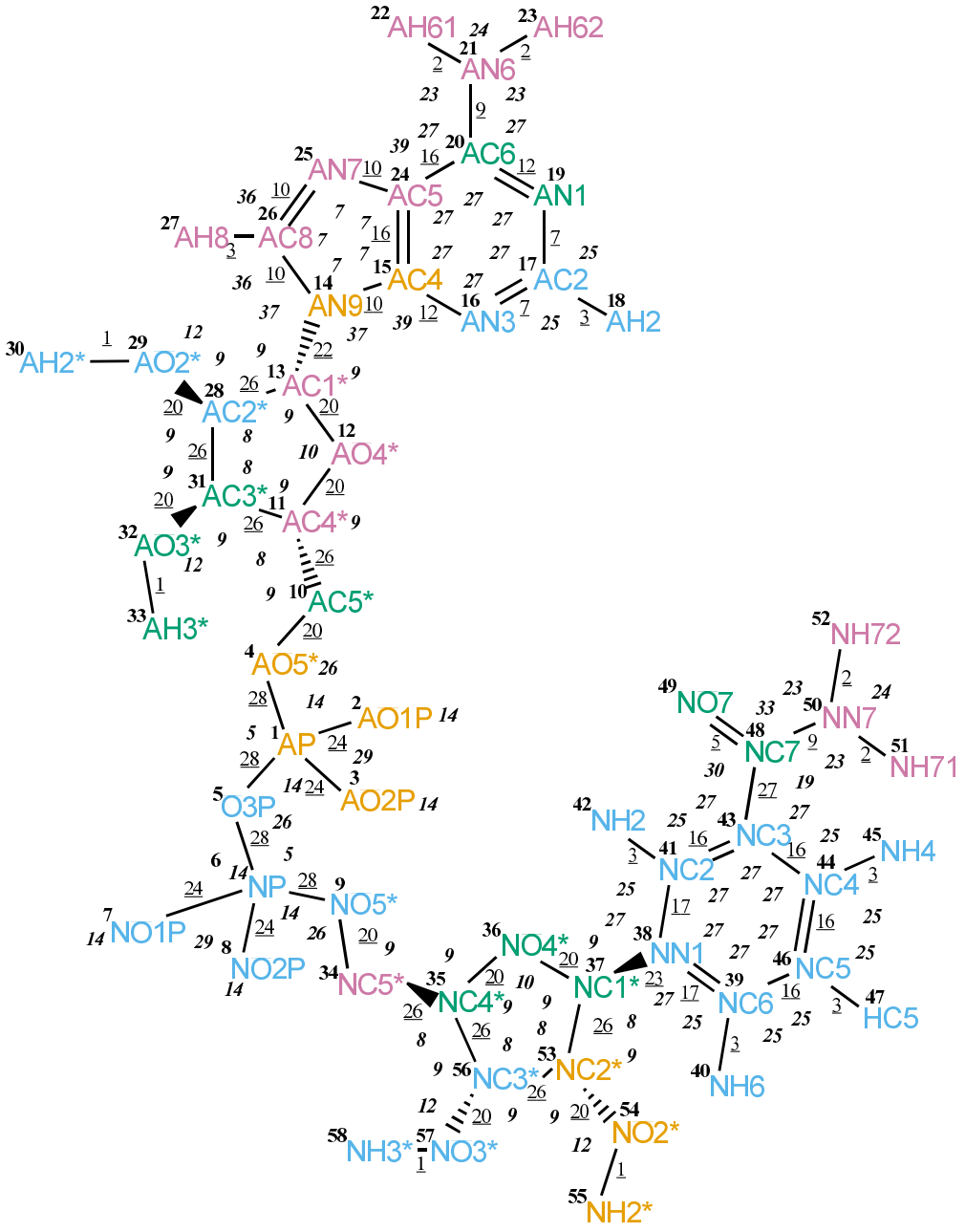


FIGURE 4.188. NADP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	AP	30	31	0.76000	2 3 4 5 6 10
2	AO1P	2	16	-0.63500	3 4 5
3	AO2P	2	16	-0.63500	4 5
4	AO5*	3	16	-0.36000	5 10 11
5	O3P	3	16	-0.26000	6 7 8 9
6	NP	30	31	0.76000	7 8 9 34
7	NO1P	2	16	-0.63500	8 9
8	NO2P	2	16	-0.63500	9
9	NO5*	3	16	-0.36000	34 35
10	AC5*	15	4	0.00000	11 12 31
11	AC4*	14	3	0.16000	12 13 28 31 32
12	AO4*	3	16	-0.36000	13 14 28 31
13	AC1*	14	3	0.20000	14 15 16 24 25 26 27 28 29 31
14	AN9	9	14	-0.20000	15 16 17 20 24 25 26 27 28
15	AC4	12	12	0.20000	16 17 18 19 20 21 24 25 26 27
16	AN3	9	14	-0.54000	17 18 19 20 24 25 26
17	AC2	12	12	0.44000	18 19 20 21 24
18	AH2	20	1	0.10000	19 20
19	AN1	9	14	-0.54000	20 21 24 25
20	AC6	12	12	0.54000	21 22 23 24 25 26
21	AN6	7	14	-0.83000	22 23 24 25
22	AH61	21	1	0.41500	23 25
23	AH62	21	1	0.41500	25
24	AC5	12	12	0.00000	25 26 27
25	AN7	9	14	-0.54000	26 27
26	AC8	12	12	0.44000	27
27	AH8	20	1	0.10000	
28	AC2*	14	3	0.15000	29 30 31 32
29	AO2*	3	16	-0.54800	30 31
30	AH2*	21	1	0.39800	
31	AC3*	14	3	0.15000	32 33
32	AO3*	3	16	-0.54800	33
33	AH3*	21	1	0.39800	
34	NC5*	15	4	0.00000	35 36 56
35	NC4*	14	3	0.16000	36 37 53 56 57
36	NO4*	3	16	-0.36000	37 38 53 56
37	NC1*	14	3	0.20000	38 39 40 41 42 43 46 53 54 56
38	NN1	9	14	0.10000	39 40 41 42 43 44 46 47 48 53
39	NC6	12	12	0.20000	40 41 42 43 44 45 46 47
40	NH6	20	1	0.10000	41 44 46 47
41	NC2	12	12	0.15000	42 43 44 45 46 48
42	NH2	20	1	0.10000	43 44 48
43	NC3	12	12	0.00000	44 45 46 47 48 49 50
44	NC4	12	12	0.15000	45 46 47 48
45	NH4	20	1	0.10000	46 47 48
46	NC5	12	12	0.00000	47 48

TABLE 4.466: continues on next page.

Seq.	Name	IAC	Mass	Charge	Exclusions
47	HC5	20	1	0.10000	
48	NC7	12	12	0.38000	49 50 51 52
49	NO7	1	16	-0.38000	50
50	NN7	7	14	-0.83000	51 52
51	NH71	21	1	0.41500	52
52	NH72	21	1	0.41500	
53	NC2*	14	3	0.15000	54 55 56 57
54	NO2*	3	16	-0.54800	55 56
55	NH2*	21	1	0.39800	
56	NC3*	14	3	0.15000	57 58
57	NO3*	3	16	-0.54800	58
58	NH3*	21	1	0.39800	

TABLE 4.466: Atoms of building block NADP.

I	J	Type
1	2	24
1	3	24
1	4	28
1	5	28
4	10	20
5	6	28
6	7	24
6	8	24
6	9	28
9	34	20
10	11	26
11	12	20
11	31	26
12	13	20
13	14	22
13	28	26
14	15	10
14	26	10
15	16	12
15	24	16
16	17	7
17	18	3
17	19	7
19	20	12
20	21	9
20	24	16
21	22	2
21	23	2
24	25	10
25	26	10
26	27	3
28	29	20
28	31	26
29	30	1
31	32	20
32	33	1
34	35	26
35	36	20
35	56	26
36	37	20
37	38	23
37	53	26
38	39	17
38	41	17
39	40	3
39	46	16

TABLE 4.467: continues on next page.

I	J	Type
41	42	3
41	43	16
43	44	16
43	48	27
44	45	3
44	46	16
46	47	3
48	49	5
48	50	9
50	51	2
50	52	2
53	54	20
53	56	26
54	55	1
56	57	20
57	58	1

TABLE 4.467: Bonds of building block NADP.

I	J	K	Type
2	1	3	29
2	1	4	14
2	1	5	14
3	1	4	14
3	1	5	14
4	1	5	5
1	4	10	26
1	5	6	26
5	6	7	14
5	6	8	14
5	6	9	5
7	6	8	29
7	6	9	14
8	6	9	14
6	9	34	26
4	10	11	9
10	11	12	9
10	11	31	8
12	11	31	9
11	12	13	10
12	13	14	9
12	13	28	9
14	13	28	9
13	14	15	37
13	14	26	37
15	14	26	7
14	15	16	39
14	15	24	7
16	15	24	27
15	16	17	27
16	17	18	25
16	17	19	27
18	17	19	25
17	19	20	27
19	20	21	27
19	20	24	27
21	20	24	27
20	21	22	23
20	21	23	23
22	21	23	24
15	24	20	27
15	24	25	7
20	24	25	39
24	25	26	7
14	26	25	7
14	26	27	36

TABLE 4.468: continues on next page.

I	J	K	Type
25	26	27	36
13	28	29	9
13	28	31	8
29	28	31	9
28	29	30	12
11	31	28	8
11	31	32	9
28	31	32	9
31	32	33	12
9	34	35	9
34	35	36	9
34	35	56	8
36	35	56	9
35	36	37	10
36	37	38	9
36	37	53	9
38	37	53	8
37	38	39	27
37	38	41	27
39	38	41	27
38	39	40	25
38	39	46	27
40	39	46	25
38	41	42	25
38	41	43	27
42	41	43	25
41	43	44	27
41	43	48	27
44	43	48	27
43	44	45	25
43	44	46	27
45	44	46	25
39	46	44	27
39	46	47	25
44	46	47	25
43	48	49	30
43	48	50	19
49	48	50	33
48	50	51	23
48	50	52	23
51	50	52	24
37	53	54	9
37	53	56	8
54	53	56	9
53	54	55	12
35	56	53	8
35	56	57	9

TABLE 4.468: continues on next page.

I	J	K	Type
53	56	57	9
56	57	58	12

TABLE 4.468: Bond angles of building block NADP.

I	J	K	L	Type
5	1	4	10	20
5	1	4	10	27
4	1	5	6	20
4	1	5	6	27
1	4	10	11	7
1	4	10	11	22
1	5	6	9	20
1	5	6	9	27
5	6	9	34	20
5	6	9	34	27
6	9	34	35	7
6	9	34	35	22
4	10	11	12	8
4	10	11	12	25
4	10	11	31	17
4	10	11	31	34
31	11	12	13	29
10	11	31	28	34
10	11	31	32	17
12	11	31	28	17
12	11	31	32	18
11	12	13	28	29
12	13	14	15	16
12	13	28	29	18
12	13	28	31	17
12	13	28	31	34
14	13	28	29	17
24	20	21	22	14
13	28	29	30	23
13	28	31	11	34
13	28	31	32	17
29	28	31	11	17
29	28	31	32	18
11	31	32	33	23
9	34	35	36	8
9	34	35	36	25
9	34	35	56	17
9	34	35	56	34
56	35	36	37	29
34	35	56	53	34
34	35	56	57	17
36	35	56	53	17
36	35	56	57	18
35	36	37	53	29
36	37	38	41	16
36	37	53	54	18

TABLE 4.469: continues on next page.

I	J	K	L	Type
36	37	53	56	17
36	37	53	56	34
38	37	53	54	17
41	43	48	50	10
43	48	50	51	14
37	53	54	55	23
37	53	56	35	34
37	53	56	57	17
54	53	56	35	17
54	53	56	57	18
35	56	57	58	23

TABLE 4.469: Dihedral angles of building block NADP.

I	J	K	L	Type
14	13	15	26	1
14	15	24	25	1
15	14	16	24	1
15	14	26	25	1
15	16	17	19	1
15	24	25	26	1
16	15	24	20	1
16	17	19	20	1
17	16	18	19	1
17	19	20	24	1
19	20	24	15	1
20	19	21	24	1
21	20	22	23	1
24	15	16	17	1
24	15	20	25	1
24	25	26	14	1
26	14	15	24	1
26	14	25	27	1
28	12	14	13	2
28	29	31	13	2
31	10	12	11	2
31	28	32	11	2
38	37	39	41	1
38	39	46	44	1
38	41	43	44	1
39	38	40	46	1
39	38	41	43	1
41	38	39	46	1
41	38	42	43	1
41	43	44	46	1
43	41	44	48	1
43	44	46	39	1
44	43	46	45	1
46	39	44	47	1
48	43	49	50	1
50	48	51	52	1
53	36	38	37	2
53	54	56	37	2
56	34	36	35	2
56	53	57	35	2

TABLE 4.470: Improper dihedral angles of building block NADP.

Solute building block: nicotinamide adenine dinucleotide phosphate(NADPH;charge -3e, $OPOHO_2^-$)
Name: NDPH

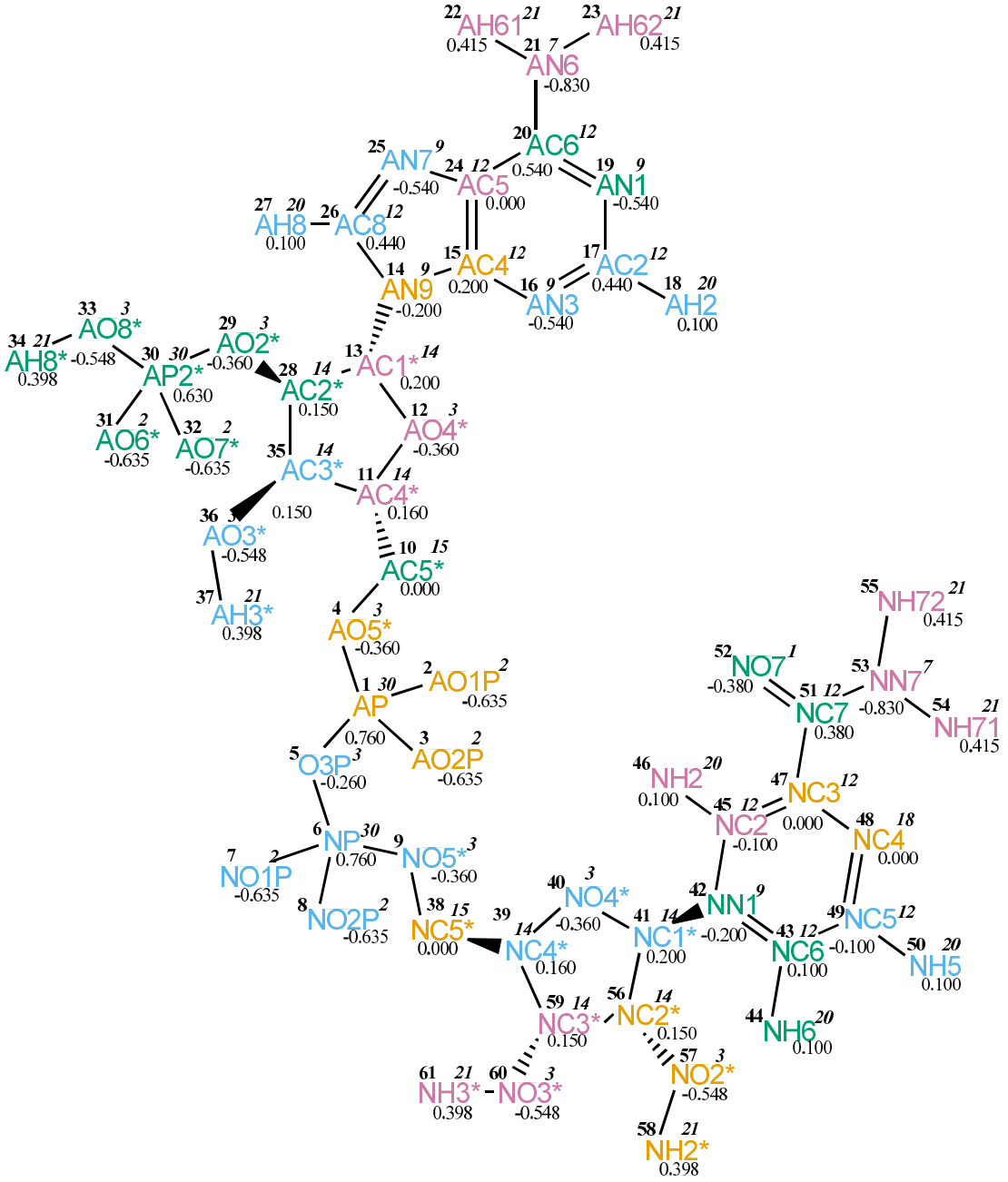


FIGURE 4.189. NDPH non-bonded parameters.

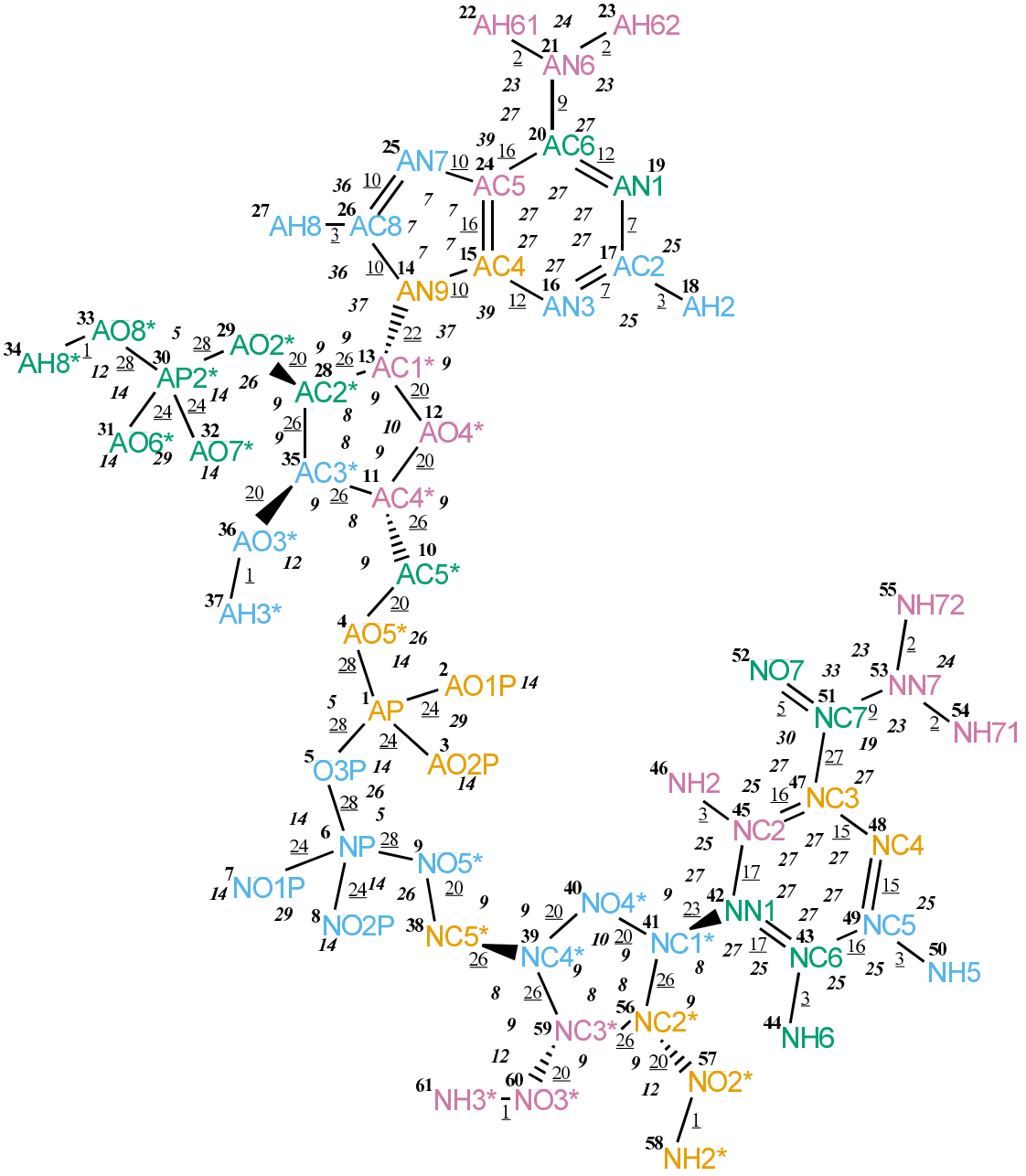


FIGURE 4.190. NDPH bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	AP	30	31	0.76000	2 3 4 5 6 10
2	AO1P	2	16	-0.63500	3 4 5
3	AO2P	2	16	-0.63500	4 5
4	AO5*	3	16	-0.36000	5 10 11
5	O3P	3	16	-0.26000	6 7 8 9
6	NP	30	31	0.76000	7 8 9 38
7	NO1P	2	16	-0.63500	8 9
8	NO2P	2	16	-0.63500	9
9	NO5*	3	16	-0.36000	38 39
10	AC5*	15	4	0.00000	11 12 35
11	AC4*	14	3	0.16000	12 13 28 35 36
12	AO4*	3	16	-0.36000	13 14 28 35
13	AC1*	14	3	0.20000	14 15 16 24 25 26 27 28 29 35
14	AN9	9	14	-0.20000	15 16 17 20 24 25 26 27 28
15	AC4	12	12	0.20000	16 17 18 19 20 21 24 25 26 27
16	AN3	9	14	-0.54000	17 18 19 20 24 25 26
17	AC2	12	12	0.44000	18 19 20 21 24
18	AH2	20	1	0.10000	19 20
19	AN1	9	14	-0.54000	20 21 24 25
20	AC6	12	12	0.54000	21 22 23 24 25 26
21	AN6	7	14	-0.83000	22 23 24 25
22	AH61	21	1	0.41500	23 25
23	AH62	21	1	0.41500	25
24	AC5	12	12	0.00000	25 26 27
25	AN7	9	14	-0.54000	26 27
26	AC8	12	12	0.44000	27
27	AH8	20	1	0.10000	
28	AC2*	14	3	0.15000	29 30 35 36
29	AO2*	3	16	-0.36000	30 31 32 33 34 35
30	AP2*	30	31	0.63000	31 32 33 34
31	AO6*	2	16	-0.63500	32 33 34
32	AO7*	2	16	-0.63500	33 34
33	AO8*	3	16	-0.54800	34
34	AH8*	21	1	0.39800	
35	AC3*	14	3	0.15000	36 37
36	AO3*	3	16	-0.54800	37
37	AH3*	21	1	0.39800	
38	NC5*	15	4	0.00000	39 40 59
39	NC4*	14	3	0.16000	40 41 56 59 60
40	NO4*	3	16	-0.36000	41 42 56 59
41	NC1*	14	3	0.20000	42 43 44 45 46 47 49 56 57 59
42	NN1	9	14	-0.20000	43 44 45 46 47 48 49 50 51 56
43	NC6	12	12	0.10000	44 45 46 47 48 49 50
44	NH6	20	1	0.10000	45 48 49 50
45	NC2	12	12	-0.10000	46 47 48 49 51
46	NH2	20	1	0.10000	47 48 51

TABLE 4.471: continues on next page.

Seq.	Name	IAC	Mass	Charge	Exclusions
47	NC3	12	12	0.00000	48 49 50 51 52 53
48	NC4	18	4	0.00000	49 50 51
49	NC5	12	12	-0.10000	50 51
50	NH5	20	1	0.10000	
51	NC7	12	12	0.38000	52 53 54 55
52	NO7	1	16	-0.38000	53
53	NN7	7	14	-0.83000	54 55
54	NH71	21	1	0.41500	55
55	NH72	21	1	0.41500	
56	NC2*	14	3	0.15000	57 58 59 60
57	NO2*	3	16	-0.54800	58 59
58	NH2*	21	1	0.39800	
59	NC3*	14	3	0.15000	60 61
60	NO3*	3	16	-0.54800	61
61	NH3*	21	1	0.39800	

TABLE 4.471: Atoms of building block NDPH.

I	J	Type
1	2	24
1	3	24
1	4	28
1	5	28
4	10	20
5	6	28
6	7	24
6	8	24
6	9	28
9	38	20
10	11	26
11	12	20
11	35	26
12	13	20
13	14	22
13	28	26
14	15	10
14	26	10
15	16	12
15	24	16
16	17	7
17	18	3
17	19	7
19	20	12
20	21	9
20	24	16
21	22	2
21	23	2
24	25	10
25	26	10
26	27	3
28	29	20
28	35	26
29	30	28
30	31	24
30	32	24
30	33	28
33	34	1
35	36	20
36	37	1
38	39	26
39	40	20
39	59	26
40	41	20
41	42	23
41	56	26

TABLE 4.472: continues on next page.

I	J	Type
42	43	17
42	45	17
43	44	3
43	49	16
45	46	3
45	47	16
47	48	15
47	51	27
48	49	15
49	50	3
51	52	5
51	53	9
53	54	2
53	55	2
56	57	20
56	59	26
57	58	1
59	60	20
60	61	1

TABLE 4.472: Bonds of building block NDPH.

I	J	K	Type
2	1	3	29
2	1	4	14
2	1	5	14
3	1	4	14
3	1	5	14
4	1	5	5
1	4	10	26
1	5	6	26
5	6	7	14
5	6	8	14
5	6	9	5
7	6	8	29
7	6	9	14
8	6	9	14
6	9	38	26
4	10	11	9
10	11	12	9
10	11	35	8
12	11	35	9
11	12	13	10
12	13	14	9
12	13	28	9
14	13	28	9
13	14	15	37
13	14	26	37
15	14	26	7
14	15	16	39
14	15	24	7
16	15	24	27
15	16	17	27
16	17	18	25
16	17	19	27
18	17	19	25
17	19	20	27
19	20	21	27
19	20	24	27
21	20	24	27
20	21	22	23
20	21	23	23
22	21	23	24
15	24	20	27
15	24	25	7
20	24	25	39
24	25	26	7
14	26	25	7
14	26	27	36

TABLE 4.473: continues on next page.

I	J	K	Type
25	26	27	36
13	28	29	9
13	28	35	8
29	28	35	9
28	29	30	26
29	30	31	14
29	30	32	14
29	30	33	5
31	30	32	29
31	30	33	14
32	30	33	14
30	33	34	12
11	35	28	8
11	35	36	9
28	35	36	9
35	36	37	12
9	38	39	9
38	39	40	9
38	39	59	8
40	39	59	9
39	40	41	10
40	41	42	9
40	41	56	9
42	41	56	8
41	42	43	27
41	42	45	27
43	42	45	27
42	43	44	25
42	43	49	27
44	43	49	25
42	45	46	25
42	45	47	27
46	45	47	25
45	47	48	27
45	47	51	27
48	47	51	27
47	48	49	27
43	49	48	27
43	49	50	25
48	49	50	25
47	51	52	30
47	51	53	19
52	51	53	33
51	53	54	23
51	53	55	23
54	53	55	24
41	56	57	9

TABLE 4.473: continues on next page.

I	J	K	Type
41	56	59	8
57	56	59	9
56	57	58	12
39	59	56	8
39	59	60	9
56	59	60	9
59	60	61	12

TABLE 4.473: Bond angles of building block NDPH.

I	J	K	L	Type
5	1	4	10	20
5	1	4	10	27
4	1	5	6	20
4	1	5	6	27
1	4	10	11	7
1	4	10	11	22
1	5	6	9	20
1	5	6	9	27
5	6	9	38	20
5	6	9	38	27
6	9	38	39	7
6	9	38	39	22
4	10	11	12	8
4	10	11	12	25
4	10	11	35	17
4	10	11	35	34
35	11	12	13	29
10	11	35	28	34
10	11	35	36	17
12	11	35	28	17
12	11	35	36	18
11	12	13	28	29
12	13	14	15	16
12	13	28	29	18
12	13	28	35	17
12	13	28	35	34
14	13	28	29	17
24	20	21	22	14
13	28	29	30	23
13	28	35	11	34
13	28	35	36	17
29	28	35	11	17
29	28	35	36	18
28	29	30	33	20
28	29	30	33	27
29	30	33	34	20
29	30	33	34	27
11	35	36	37	23
9	38	39	40	8
9	38	39	40	25
9	38	39	59	17
9	38	39	59	34
59	39	40	41	29
38	39	59	56	34
38	39	59	60	17
40	39	59	56	17

TABLE 4.474: continues on next page.

I	J	K	L	Type
40	39	59	60	18
39	40	41	56	29
40	41	42	45	16
40	41	56	57	18
40	41	56	59	17
40	41	56	59	34
42	41	56	57	17
45	47	51	53	10
47	51	53	54	14
41	56	57	58	23
41	56	59	39	34
41	56	59	60	17
57	56	59	39	17
57	56	59	60	18
39	59	60	61	23

TABLE 4.474: Dihedral angles of building block NDPH.

I	J	K	L	Type
14	13	15	26	1
14	15	24	25	1
15	14	16	24	1
15	14	26	25	1
15	16	17	19	1
15	24	25	26	1
16	15	24	20	1
16	17	19	20	1
17	16	18	19	1
17	19	20	24	1
19	20	24	15	1
20	19	21	24	1
21	20	22	23	1
24	15	16	17	1
24	15	20	25	1
24	25	26	14	1
26	14	15	24	1
26	14	25	27	1
28	12	14	13	2
28	29	35	13	2
35	10	12	11	2
35	28	36	11	2
42	41	43	45	1
42	43	49	48	1
42	45	47	48	1
43	42	44	49	1
43	42	45	47	1
45	42	43	49	1
45	42	46	47	1
45	47	48	49	1
47	45	48	51	1
47	48	49	43	1
49	43	48	50	1
51	47	52	53	1
53	51	54	55	1
56	40	42	41	2
56	57	59	41	2
59	38	40	39	2
59	56	60	39	2

TABLE 4.475: Improper dihedral angles of building block NDPH.

4.6. Carbohydrates

Solute building block: -2-D-glucopyranose-β-1-
Name: GB2P

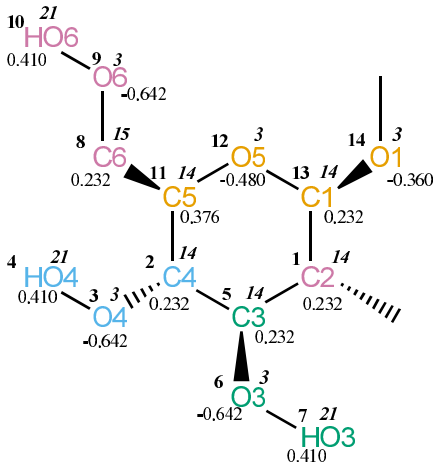


FIGURE 4.191. GB2P non-bonded parameters.

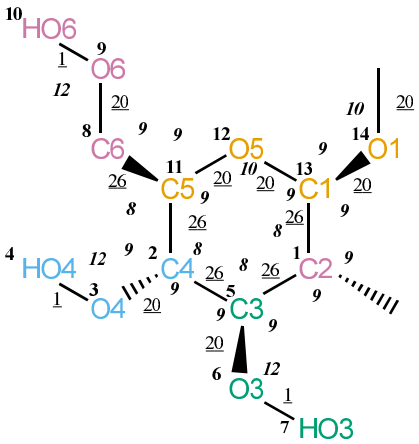


FIGURE 4.192. GB2P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 5 13
1	C2	14	3	0.23200	2 5 6 12 13 14
2	C4	14	3	0.23200	3 4 5 6 8 11 12
3	O4	3	16	-0.64200	4 5 11
4	HO4	21	1	0.41000	
5	C3	14	3	0.23200	6 7 11 13
6	O3	3	16	-0.64200	7
7	HO3	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.476. Atoms of building block GB2P.

I	J	Type
1	5	26
1	13	26
2	3	20
2	5	26
2	11	26
3	4	1
5	6	20
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.477. Bonds of building block GB2P.

I	J	K	Type
0	1	5	9
0	1	13	9
5	1	13	8
3	2	5	9
3	2	11	9
5	2	11	8
2	3	4	12
1	5	2	8
1	5	6	9
2	5	6	9
5	6	7	12
9	8	11	9
8	9	10	12
2	11	8	8
2	11	12	9
8	11	12	9
11	12	13	10
1	13	12	9
1	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.478. Bond angles of building block GB2P.

I	J	K	L	Type
-1	0	1	13	30
0	1	5	2	17
0	1	5	6	18
13	1	5	2	34
13	1	5	6	17
0	1	13	14	18
5	1	13	12	17
5	1	13	12	34
5	1	13	14	17
5	2	3	4	30
3	2	5	1	17
3	2	5	6	18
11	2	5	1	34
11	2	5	6	17
3	2	11	8	17
5	2	11	12	17
5	2	11	12	34
1	5	6	7	30
11	8	9	10	30
9	8	11	12	5
9	8	11	12	37
2	11	12	13	29
11	12	13	1	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.479. Dihedral angles of building block GB2P.

I	J	K	L	Type
1	0	5	13	2
1	12	14	13	2
2	1	6	5	2
2	8	12	11	2
11	3	5	2	2

TABLE 4.480. Improper dihedral angles of building block GB2P.

Solute building block: -3-D-glucopyranose-β-1-
Name: GB3P

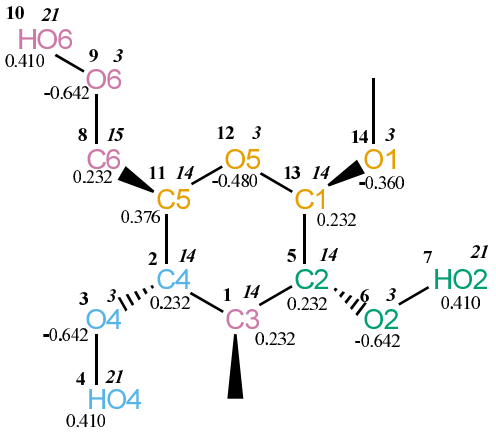


FIGURE 4.193. GB3P non-bonded parameters.

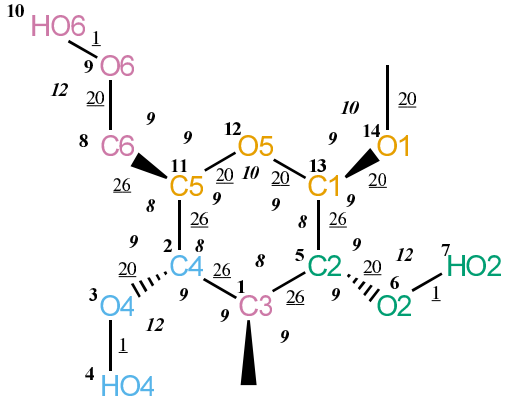


FIGURE 4.194. GB3P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 5
1	C3	14	3	0.23200	2 3 5 6 11 13
2	C4	14	3	0.23200	3 4 5 8 11 12
3	O4	3	16	-0.64200	4 11
4	HO4	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.481. Atoms of building block GB3P.

I	J	Type
1	2	26
1	5	26
2	3	20
2	11	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.482. Bonds of building block GB3P.

I	J	K	Type
0	1	2	9
0	1	5	9
2	1	5	8
1	2	3	9
1	2	11	8
3	2	11	9
2	3	4	12
1	5	6	9
1	5	13	8
6	5	13	9
5	6	7	12
9	8	11	9
8	9	10	12
2	11	8	8
2	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.483. Bond angles of building block GB3P.

I	J	K	L	Type
-1	0	1	5	30
0	1	2	3	18
0	1	2	11	17
5	1	2	3	17
5	1	2	11	34
0	1	5	6	18
0	1	5	13	17
2	1	5	6	17
2	1	5	13	34
1	2	3	4	30
1	2	11	12	17
1	2	11	12	34
3	2	11	8	17
13	5	6	7	30
1	5	13	12	17
1	5	13	12	34
1	5	13	14	17
6	5	13	14	18
11	8	9	10	30
9	8	11	12	5
9	8	11	12	37
2	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.484. Dihedral angles of building block GB3P.

I	J	K	L	Type
1	0	5	2	2
2	1	3	11	2
2	8	12	11	2
5	12	14	13	2
13	1	6	5	2

TABLE 4.485. Improper dihedral angles of building block GB3P.

Solute building block: -6-D-glucopyranose-β-1-
Name: GB6P

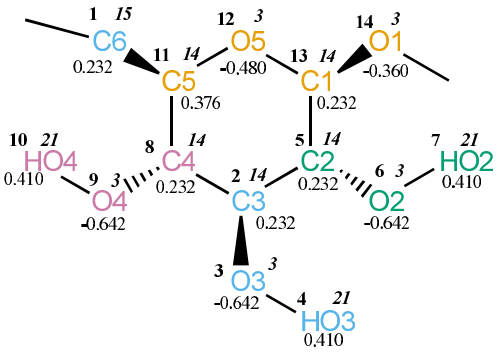


FIGURE 4.195. GB6P non-bonded parameters.

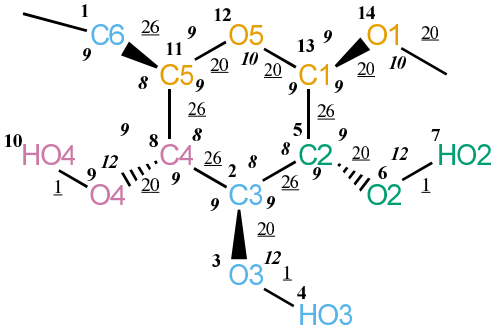


FIGURE 4.196. GB6P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 11
1	C6	15	4	0.23200	8 11 12
2	C3	14	3	0.23200	3 4 5 6 8 9 11 13
3	O3	3	16	-0.64200	4 5 8
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 8 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C4	14	3	0.23200	9 10 11 12
9	O4	3	16	-0.64200	10 11
10	HO4	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.486. Atoms of building block GB6P.

I	J	Type
1	11	26
2	3	20
2	5	26
2	8	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.487. Bonds of building block GB6P.

I	J	K	Type
0	1	11	9
3	2	5	9
3	2	8	9
5	2	8	8
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
2	8	9	9
2	8	11	8
9	8	11	9
8	9	10	12
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.488. Bond angles of building block GB6P.

I	J	K	L	Type
-1	0	1	11	30
0	1	11	12	5
0	1	11	12	37
5	2	3	4	30
3	2	5	6	18
3	2	5	13	17
8	2	5	6	17
8	2	5	13	34
3	2	8	9	18
3	2	8	11	17
5	2	8	9	17
5	2	8	11	34
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
2	8	9	10	30
2	8	11	12	17
2	8	11	12	34
9	8	11	1	17
8	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.489. Dihedral angles of building block GB6P.

I	J	K	L	Type
2	3	5	8	2
5	12	14	13	2
8	1	12	11	2
8	2	9	11	2
13	2	6	5	2

TABLE 4.490. Improper dihedral angles of building block GB6P.

Solute building block: -4-D-glucopyranose-β-1-
Name: GB4P

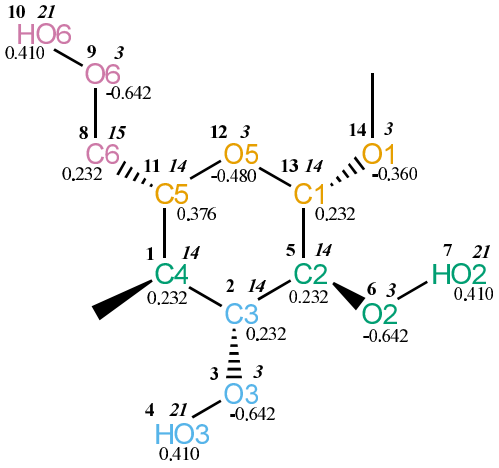


FIGURE 4.197. GB4P non-bonded parameters.

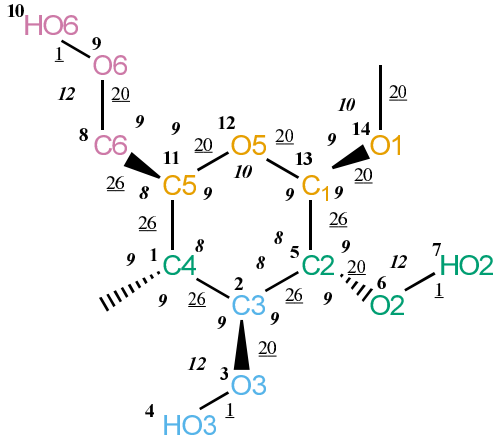


FIGURE 4.198. GB4P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.491. Atoms of building block GB4P.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.492. Bonds of building block GB4P.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	11	9
8	9	10	12
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.493. Bond angles of building block GB4P.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
11	8	9	10	30
9	8	11	12	5
9	8	11	12	37
1	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.494. Dihedral angles of building block GB4P.

I	J	K	L	Type
1	8	12	11	2
2	3	5	1	2
5	12	14	13	2
11	0	2	1	2
13	2	6	5	2

TABLE 4.495. Improper dihedral angles of building block GB4P.

Solute building block: -4-D-glucopyranose-α-1-
Name: GA4P

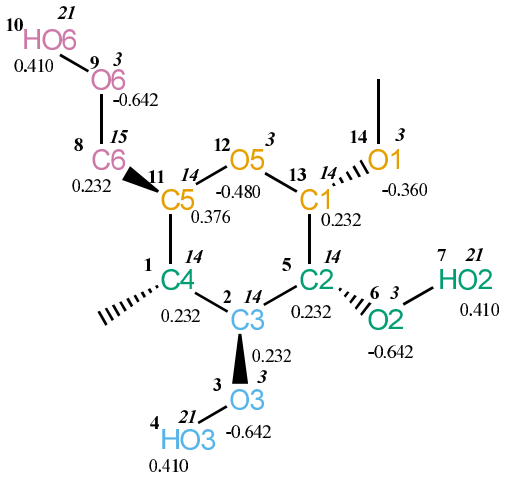


FIGURE 4.199. GA4P non-bonded parameters.

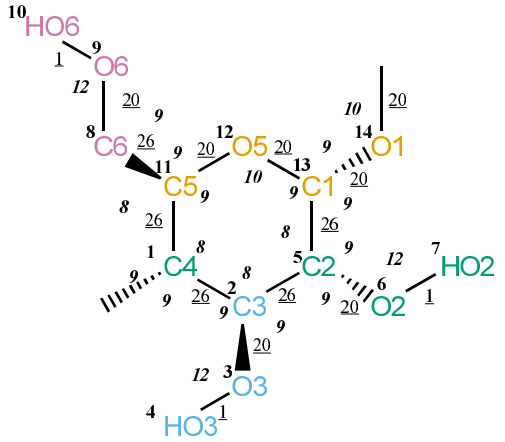


FIGURE 4.200. GA4P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.496. Atoms of building block GA4P.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.497. Bonds of building block GA4P.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	11	9
8	9	10	12
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.498. Bond angles of building block GA4P.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
11	8	9	10	30
9	8	11	12	5
9	8	11	12	37
1	11	12	13	29
11	12	13	5	29
12	13	14	15	6
12	13	14	15	28

TABLE 4.499. Dihedral angles of building block GA4P.

I	J	K	L	Type
1	8	12	11	2
2	3	5	1	2
11	0	2	1	2
13	2	6	5	2
13	12	14	5	2

TABLE 4.500. Improper dihedral angles of building block GA4P.

Solute building block: -4-L-glucopyranose-β-1-
Name: gB4P

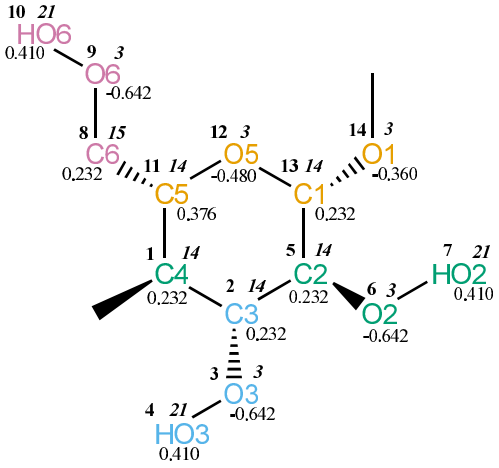


FIGURE 4.201. gB4P non-bonded parameters.

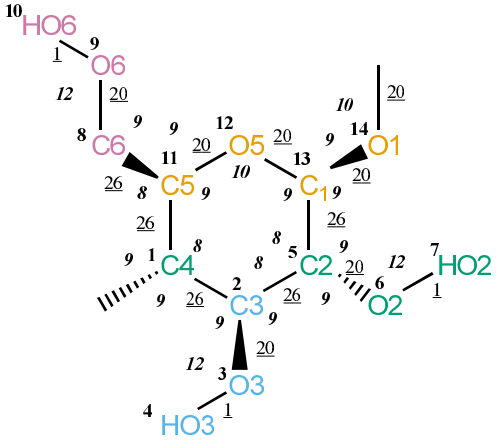


FIGURE 4.202. gB4P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.501. Atoms of building block gB4P.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.502. Bonds of building block gB4P.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	11	9
8	9	10	12
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.503. Bond angles of building block gB4P.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
11	8	9	10	30
9	8	11	12	5
9	8	11	12	37
1	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.504. Dihedral angles of building block gB4P.

I	J	K	L	Type
1	0	2	11	2
1	3	5	2	2
5	2	6	13	2
11	8	12	1	2
13	12	14	5	2

TABLE 4.505. Improper dihedral angles of building block gB4P.

Solute building block: -4-D-galactopyranose-β-1-
Name: LB4P

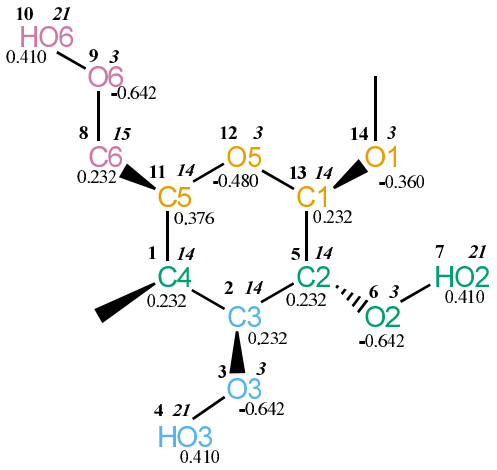


FIGURE 4.203. LB4P non-bonded parameters.

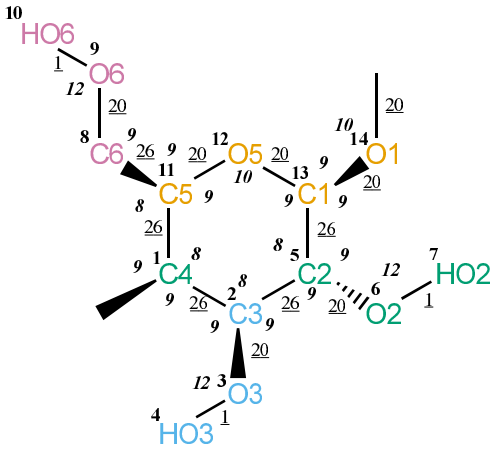


FIGURE 4.204. LB4P bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	15	4	0.23200	9 10 11 12
9	O6	3	16	-0.64200	10 11
10	HO6	21	1	0.41000	
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.506. Atoms of building block LB4P.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	20
8	11	26
9	10	1
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.507. Bonds of building block LB4P.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	11	9
8	9	10	12
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.508. Bond angles of building block LB4P.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
11	8	9	10	30
9	8	11	1	1
9	8	11	12	3
9	8	11	12	35
1	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.509. Dihedral angles of building block LB4P.

I	J	K	L	Type
1	0	2	11	2
1	8	12	11	2
2	3	5	1	2
5	12	14	13	2
13	2	6	5	2

TABLE 4.510. Improper dihedral angles of building block LB4P.

Solute building block: -4-D-glucuronate-β-1-
Name: GB4U

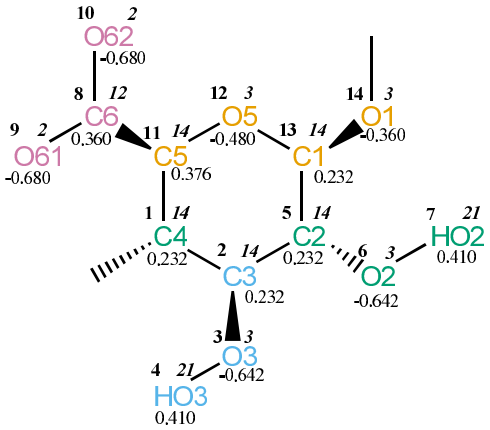


FIGURE 4.205. GB4U non-bonded parameters.

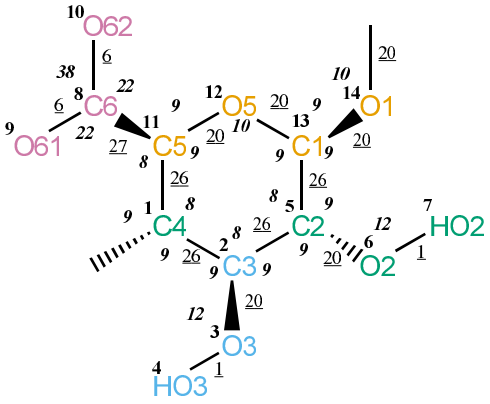


FIGURE 4.206. GB4U bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	12	12	0.36000	9 10 11 12
9	O61	2	16	-0.68000	10 11
10	O62	2	16	-0.68000	11
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.511. Atoms of building block GB4U.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	6
8	10	6
8	11	27
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.512. Bonds of building block GB4U.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	10	38
9	8	11	22
10	8	11	22
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.513. Bond angles of building block GB4U.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
9	8	11	12	40
1	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.514. Dihedral angles of building block GB4U.

I	J	K	L	Type
1	8	12	11	2
2	3	5	1	2
5	12	14	13	2
8	9	10	11	1
11	0	2	1	2
13	2	6	5	2

TABLE 4.515. Improper dihedral angles of building block GB4U.

Solute building block: -4-D-mannuronate-β-1-
Name: MB4U

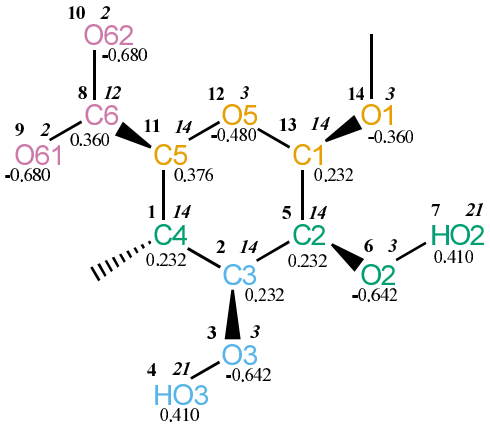


FIGURE 4.207. MB4U non-bonded parameters.

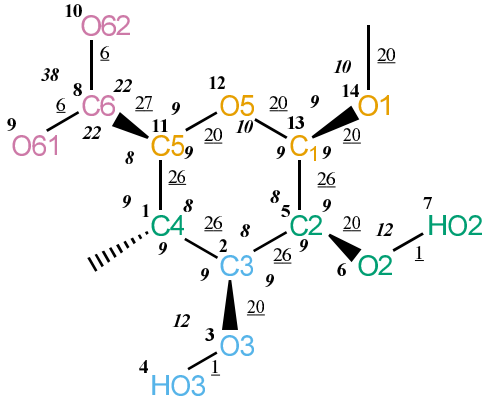


FIGURE 4.208. MB4U bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	12	12	0.36000	9 10 11 12
9	O61	2	16	-0.68000	10 11
10	O62	2	16	-0.68000	11
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.516. Atoms of building block MB4U.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	6
8	10	6
8	11	27
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.517. Bonds of building block MB4U.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	10	38
9	8	11	22
10	8	11	22
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.518. Bond angles of building block MB4U.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
9	8	11	12	40
1	11	12	13	29
11	12	13	5	29
12	13	14	15	2
12	13	14	15	32

TABLE 4.519. Dihedral angles of building block MB4U.

I	J	K	L	Type
1	8	12	11	2
2	3	5	1	2
5	2	6	13	2
5	12	14	13	2
8	9	10	11	1
11	0	2	1	2

TABLE 4.520. Improper dihedral angles of building block MB4U.

Solute building block: -4-D-galacturonate-α-1-
Name: LA4U

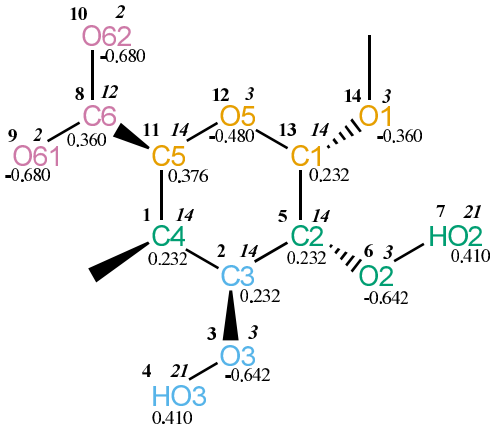


FIGURE 4.209. LA4U non-bonded parameters.

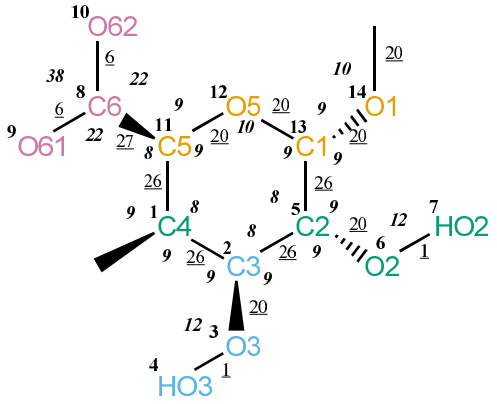


FIGURE 4.210. LA4U bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	12	12	0.36000	9 10 11 12
9	O61	2	16	-0.68000	10 11
10	O62	2	16	-0.68000	11
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.521. Atoms of building block LA4U.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	6
8	10	6
8	11	27
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.522. Bonds of building block LA4U.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	10	38
9	8	11	22
10	8	11	22
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.523. Bond angles of building block LA4U.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
9	8	11	12	40
1	11	12	13	29
11	12	13	5	29
12	13	14	15	6
12	13	14	15	28

TABLE 4.524. Dihedral angles of building block LA4U.

I	J	K	L	Type
1	0	2	11	2
1	8	12	11	2
2	3	5	1	2
8	9	10	11	1
13	2	6	5	2
13	12	14	5	2

TABLE 4.525. Improper dihedral angles of building block LA4U.

Solute building block: -4-L-guluronate-α-1-
Name: kA4U

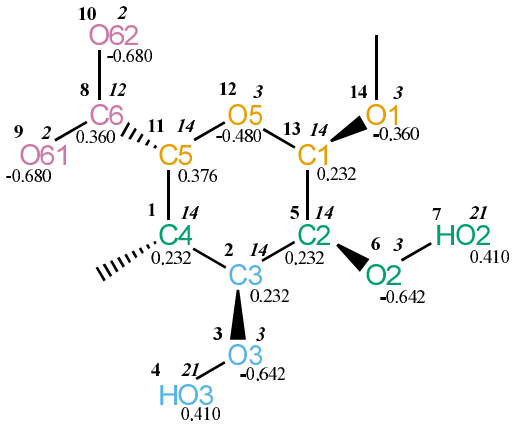


FIGURE 4.211. kA4U non-bonded parameters.

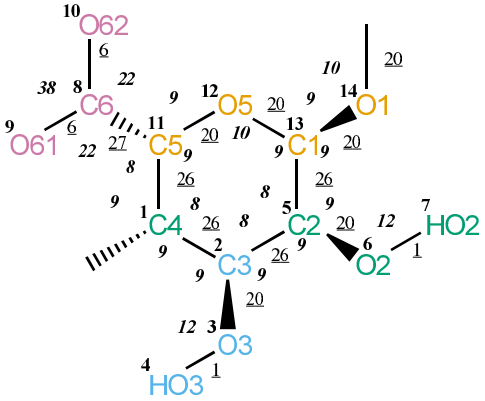


FIGURE 4.212. kA4U bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	12	12	0.36000	9 10 11 12
9	O61	2	16	-0.68000	10 11
10	O62	2	16	-0.68000	11
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.526. Atoms of building block kA4U.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	6
8	10	6
8	11	27
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.527. Bonds of building block kA4U.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	10	38
9	8	11	22
10	8	11	22
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.528. Bond angles of building block kA4U.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
9	8	11	12	40
1	11	12	13	29
11	12	13	5	29
12	13	14	15	6
12	13	14	15	28

TABLE 4.529. Dihedral angles of building block kA4U.

I	J	K	L	Type
2	3	5	1	2
5	2	6	13	2
5	12	14	13	2
8	9	10	11	1
11	0	2	1	2
11	8	12	1	2

TABLE 4.530. Improper dihedral angles of building block kA4U.

Solute building block: -4-L-iduronate-α-1-
Name: iA4U

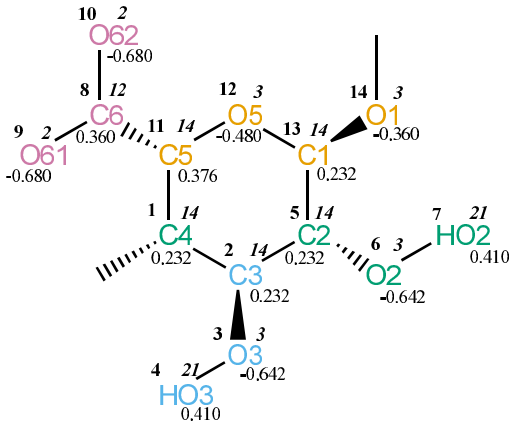


FIGURE 4.213. iA4U non-bonded parameters.

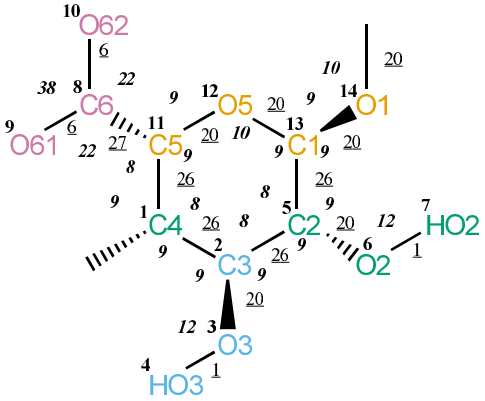


FIGURE 4.214. iA4U bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
-1					0 1
0					1 2 11
1	C4	14	3	0.23200	2 3 5 8 11 12
2	C3	14	3	0.23200	3 4 5 6 11 13
3	O3	3	16	-0.64200	4 5
4	HO3	21	1	0.41000	
5	C2	14	3	0.23200	6 7 12 13 14
6	O2	3	16	-0.64200	7 13
7	HO2	21	1	0.41000	
8	C6	12	12	0.36000	9 10 11 12
9	O61	2	16	-0.68000	10 11
10	O62	2	16	-0.68000	11
11	C5	14	3	0.37600	12 13
12	O5	3	16	-0.48000	13 14
13	C1	14	3	0.23200	
14	O1	3	16	-0.36000	

TABLE 4.531. Atoms of building block iA4U.

I	J	Type
1	2	26
1	11	26
2	3	20
2	5	26
3	4	1
5	6	20
5	13	26
6	7	1
8	9	6
8	10	6
8	11	27
11	12	20
12	13	20
13	14	20
14	15	20

TABLE 4.532. Bonds of building block iA4U.

I	J	K	Type
0	1	2	9
0	1	11	9
2	1	11	8
1	2	3	9
1	2	5	8
3	2	5	9
2	3	4	12
2	5	6	9
2	5	13	8
6	5	13	9
5	6	7	12
9	8	10	38
9	8	11	22
10	8	11	22
1	11	8	8
1	11	12	9
8	11	12	9
11	12	13	10
5	13	12	9
5	13	14	9
12	13	14	9
13	14	15	10

TABLE 4.533. Bond angles of building block iA4U.

I	J	K	L	Type
-1	0	1	2	30
0	1	2	3	18
0	1	2	5	17
11	1	2	3	17
11	1	2	5	34
0	1	11	8	17
2	1	11	12	17
2	1	11	12	34
5	2	3	4	30
1	2	5	6	17
1	2	5	13	34
3	2	5	6	18
3	2	5	13	17
13	5	6	7	30
2	5	13	12	17
2	5	13	12	34
2	5	13	14	17
6	5	13	14	18
9	8	11	12	40
1	11	12	13	29
11	12	13	5	29
12	13	14	15	6
12	13	14	15	28

TABLE 4.534. Dihedral angles of building block iA4U.

I	J	K	L	Type
2	3	5	1	2
5	12	14	13	2
8	9	10	11	1
11	0	2	1	2
11	8	12	1	2
13	2	6	5	2

TABLE 4.535. Improper dihedral angles of building block iA4U.

Solute building block: -1-β-D-glucopyranose
Name: CGBP

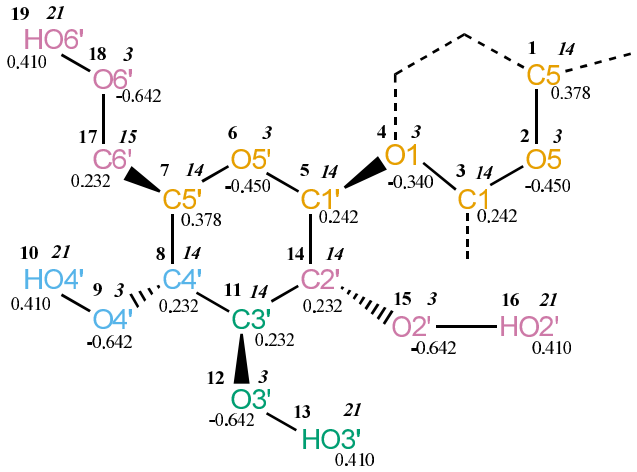


FIGURE 4.215. CGBP non-bonded parameters.

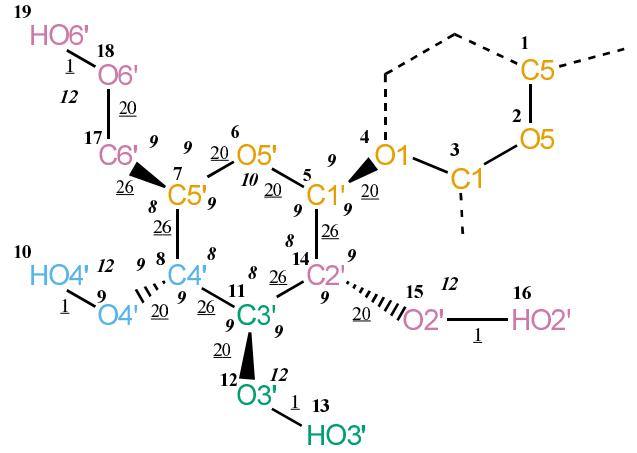


FIGURE 4.216. CGBP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	C5	14	3	0.37800	2 3
2	O5	3	16	-0.45000	3 4
3	C1	14	3	0.24200	4 5
4	O1	3	16	-0.34000	5 6 14
5	C1'	14	3	0.24200	6 7 11 14 15
6	O5'	3	16	-0.45000	7 8 14 17
7	C5'	14	3	0.37800	8 9 11 17 18
8	C4'	14	3	0.23200	9 10 11 12 14 17
9	O4'	3	16	-0.64200	10 11
10	HO4'	21	1	0.41000	
11	C3'	14	3	0.23200	12 13 14 15
12	O3'	3	16	-0.64200	13 14
13	HO3'	21	1	0.41000	
14	C2'	14	3	0.23200	15 16
15	O2'	3	16	-0.64200	16
16	HO2'	21	1	0.41000	
17	C6'	15	4	0.23200	18 19
18	O6'	3	16	-0.64200	19
19	HO6'	21	1	0.41000	

TABLE 4.536. Atoms of building block CGBP.

I	J	Type
5	6	20
5	14	26
6	7	20
7	8	26
7	17	26
8	9	20
8	11	26
9	10	1
11	12	20
11	14	26
12	13	1
14	15	20
15	16	1
17	18	20
18	19	1

TABLE 4.537. Bonds of building block CGBP.

I	J	K	Type
4	5	6	9
4	5	14	9
6	5	14	9
5	6	7	10
6	7	8	9
6	7	17	9
8	7	17	8
7	8	9	9
7	8	11	8
9	8	11	9
8	9	10	12
8	11	12	9
8	11	14	8
12	11	14	9
11	12	13	12
5	14	11	8
5	14	15	9
11	14	15	9
14	15	16	12
7	17	18	9
17	18	19	12

TABLE 4.538. Bond angles of building block CGBP.

I	J	K	L	Type
3	4	5	6	2
3	4	5	6	32
14	5	6	7	29
4	5	14	11	17
4	5	14	15	18
6	5	14	11	17
6	5	14	11	34
5	6	7	8	29
6	7	8	11	17
6	7	8	11	34
17	7	8	9	17
6	7	17	18	5
6	7	17	18	37
11	8	9	10	30
7	8	11	12	17
7	8	11	14	34
9	8	11	12	18
9	8	11	14	17
14	11	12	13	30
8	11	14	5	34
8	11	14	15	17
12	11	14	5	17
12	11	14	15	18
5	14	15	16	30
7	17	18	19	30

TABLE 4.539. Dihedral angles of building block CGBP.

I	J	K	L	Type
5	4	6	14	2
5	11	15	14	2
7	6	17	8	2
7	9	11	8	2
11	12	14	8	2

TABLE 4.540. Improper dihedral angles of building block CGBP.

4.7. Other molecules

Solute building block: heme group (charge -2, acidic groups deprotonated)
Name: HEME

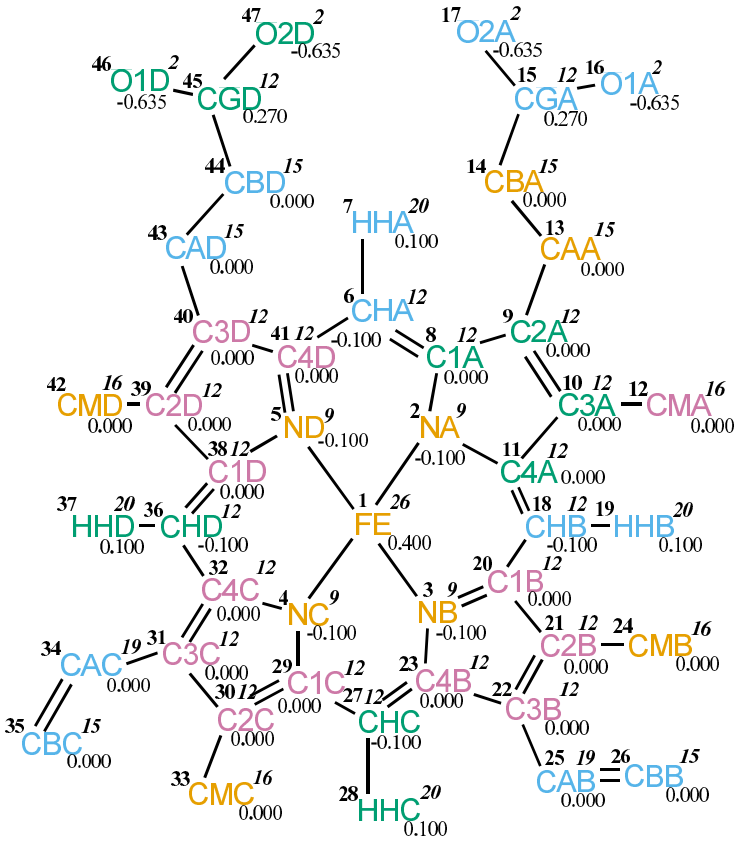


FIGURE 4.217. HEME non-bonded parameters.

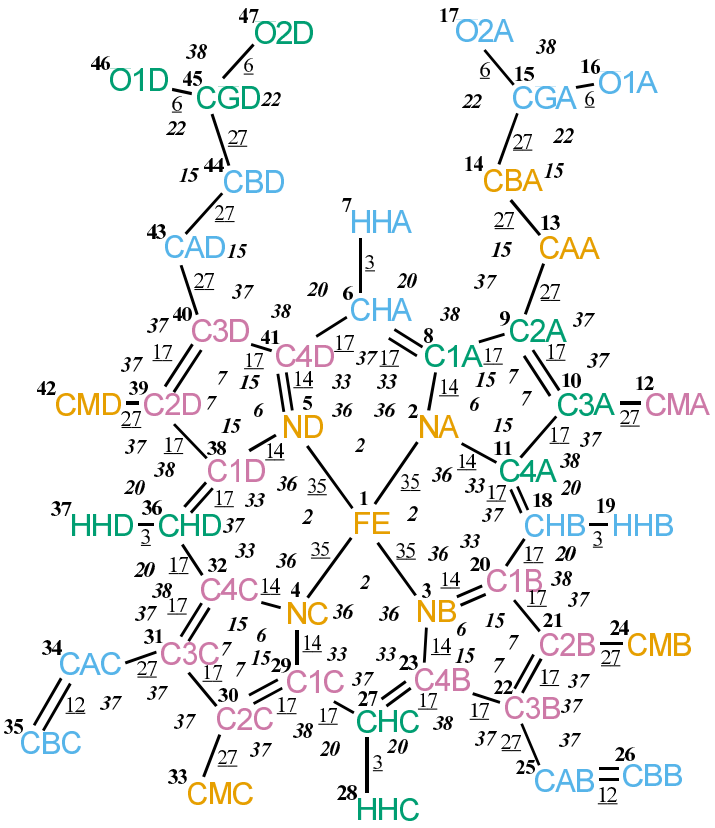


FIGURE 4.218. HEME bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	FE	26	56	0.40000	2 3 4 5 6 8 9 10 11 18 20 21 22 23 27 29 30 31 32 36 38 39 40 41
2	NA	9	14	-0.10000	3 4 5 6 7 8 9 10 11 12 13 18 19 20 23 29 32 38 41
3	NB	9	14	-0.10000	4 5 8 11 18 19 20 21 22 23 24 25 27 28 29 32 38 41
4	NC	9	14	-0.10000	5 8 11 20 23 27 28 29 30 31 32 33 34 36 37 38 41
5	ND	9	14	-0.10000	6 7 8 11 20 23 29 32 36 37 38 39 40 41 42 43
6	CHA	12	12	-0.10000	7 8 9 10 11 13 38 39 40 41 43
7	HHA	20	1	0.10000	8 9 40 41
8	C1A	12	12	0.00000	9 10 11 12 13 18 40 41
9	C2A	12	12	0.00000	10 11 12 13 14 18 41
10	C3A	12	12	0.00000	11 12 13 18 19 20
11	C4A	12	12	0.00000	12 13 18 19 20 21
12	CMA	16	5	0.00000	13 18
13	CAA	15	4	0.00000	14 15
14	CBA	15	4	0.00000	15 16 17
15	CGA	12	12	0.27000	16 17
16	O1A	2	16	-0.63500	17
17	O2A	2	16	-0.63500	
18	CHB	12	12	-0.10000	19 20 21 22 23 24
19	HHB	20	1	0.10000	20 21
20	C1B	12	12	0.00000	21 22 23 24 25 27
21	C2B	12	12	0.00000	22 23 24 25 27
22	C3B	12	12	0.00000	23 24 25 26 27 28 29
23	C4B	12	12	0.00000	24 25 27 28 29 30
24	CMB	16	5	0.00000	25
25	CAB	19	3	0.00000	26 27
26	CBB	15	4	0.00000	
27	CHC	12	12	-0.10000	28 29 30 31 32 33
28	HHC	20	1	0.10000	29 30
29	C1C	12	12	0.00000	30 31 32 33 34 36
30	C2C	12	12	0.00000	31 32 33 34 36
31	C3C	12	12	0.00000	32 33 34 35 36 37 38
32	C4C	12	12	0.00000	33 34 36 37 38 39
33	CMC	16	5	0.00000	34
34	CAC	19	3	0.00000	35 36
35	CBC	15	4	0.00000	
36	CHD	12	12	-0.10000	37 38 39 40 41 42
37	HHD	20	1	0.10000	38 39
38	C1D	12	12	0.00000	39 40 41 42 43
39	C2D	12	12	0.00000	40 41 42 43
40	C3D	12	12	0.00000	41 42 43 44
41	C4D	12	12	0.00000	42 43
42	CMD	16	5	0.00000	43
43	CAD	15	4	0.00000	44 45
44	CBD	15	4	0.00000	45 46 47
45	CGD	12	12	0.27000	46 47
46	O1D	2	16	-0.63500	47

TABLE 4.541: continues on next page.

Seq.	Name	IAC	Mass	Charge	Exclusions
47	O2D	2	16	-0.63500	

TABLE 4.541: Atoms of building block HEME.

I	J	Type
1	2	35
1	3	35
1	4	35
1	5	35
2	8	14
2	11	14
3	20	14
3	23	14
4	29	14
4	32	14
5	38	14
5	41	14
6	7	3
6	8	17
6	41	17
8	9	17
9	10	17
9	13	27
10	11	17
10	12	27
11	18	17
13	14	27
14	15	27
15	16	6
15	17	6
18	19	3
18	20	17
20	21	17
21	22	17
21	24	27
22	23	17
22	25	27
23	27	17
25	26	12
27	28	3
27	29	17
29	30	17
30	31	17
30	33	27
31	32	17
31	34	27
32	36	17
34	35	12
36	37	3
36	38	17
38	39	17

TABLE 4.542: continues on next page.

I	J	Type
39	40	17
39	42	27
40	41	17
40	43	27
43	44	27
44	45	27
45	46	6
45	47	6

TABLE 4.542: Bonds of building block HEME.

I	J	K	Type
2	1	3	2
2	1	5	2
3	1	4	2
4	1	5	2
1	2	8	36
1	2	11	36
8	2	11	6
1	3	20	36
1	3	23	36
20	3	23	6
1	4	29	36
1	4	32	36
29	4	32	6
1	5	38	36
1	5	41	36
38	5	41	6
7	6	8	20
7	6	41	20
8	6	41	37
2	8	6	33
2	8	9	15
6	8	9	38
8	9	10	7
8	9	13	37
10	9	13	37
9	10	11	7
9	10	12	37
11	10	12	37
2	11	10	15
2	11	18	33
10	11	18	38
9	13	14	15
13	14	15	15
14	15	16	22
14	15	17	22
16	15	17	38
11	18	19	20
11	18	20	37
19	18	20	20
3	20	18	33
3	20	21	15
18	20	21	38
20	21	22	7
20	21	24	37
22	21	24	37
21	22	23	7

TABLE 4.543: continues on next page.

I	J	K	Type
21	22	25	37
23	22	25	37
3	23	22	15
3	23	27	33
22	23	27	38
22	25	26	37
23	27	28	20
23	27	29	37
28	27	29	20
4	29	27	33
4	29	30	15
27	29	30	38
29	30	31	7
29	30	33	37
31	30	33	37
30	31	32	7
30	31	34	37
32	31	34	37
4	32	31	15
4	32	36	33
31	32	36	38
31	34	35	37
32	36	37	20
32	36	38	37
37	36	38	20
5	38	36	33
5	38	39	15
36	38	39	38
38	39	40	7
38	39	42	37
40	39	42	37
39	40	41	7
39	40	43	37
41	40	43	37
5	41	6	33
5	41	40	15
6	41	40	38
40	43	44	15
43	44	45	15
44	45	46	22
44	45	47	22
46	45	47	38

TABLE 4.543: Bond angles of building block HEME.

I	J	K	L	Type
41	6	8	2	15
8	6	41	5	15
8	9	13	14	40
2	11	18	20	15
9	13	14	15	34
13	14	15	16	40
11	18	20	3	15
21	22	25	26	9
3	23	27	29	15
23	27	29	4	15
30	31	34	35	9
4	32	36	38	15
32	36	38	5	15
39	40	43	44	40
40	43	44	45	34
43	44	45	46	40

TABLE 4.544: Dihedral angles of building block HEME.

I	J	K	L	Type
1	8	11	2	3
1	20	23	3	3
1	29	32	4	3
1	38	41	5	3
2	8	9	10	1
3	20	21	22	1
4	29	30	31	1
5	38	39	40	1
6	2	9	8	1
6	5	40	41	1
7	8	41	6	1
8	2	11	10	1
8	9	10	11	1
9	8	10	13	1
9	10	11	2	1
10	9	11	12	1
11	2	8	9	1
14	16	17	15	1
18	2	10	11	1
18	3	21	20	1
18	11	20	19	1
20	3	23	22	1
20	21	22	23	1
21	20	22	24	1
21	22	23	3	1
22	21	23	25	1
23	3	20	21	1
27	3	22	23	1
27	4	30	29	1
27	23	29	28	1
29	4	32	31	1
29	30	31	32	1
30	29	31	33	1
30	31	32	4	1
31	30	32	34	1
32	4	29	30	1
36	4	31	32	1
36	5	39	38	1
36	32	38	37	1
38	5	41	40	1
38	39	40	41	1
39	38	40	42	1
39	40	41	5	1
40	39	41	43	1
41	5	38	39	1
44	46	47	45	1

TABLE 4.545: Improper dihedral angles of building block HEME.

Solute building block: folate (charge -2e)
Name: FOL

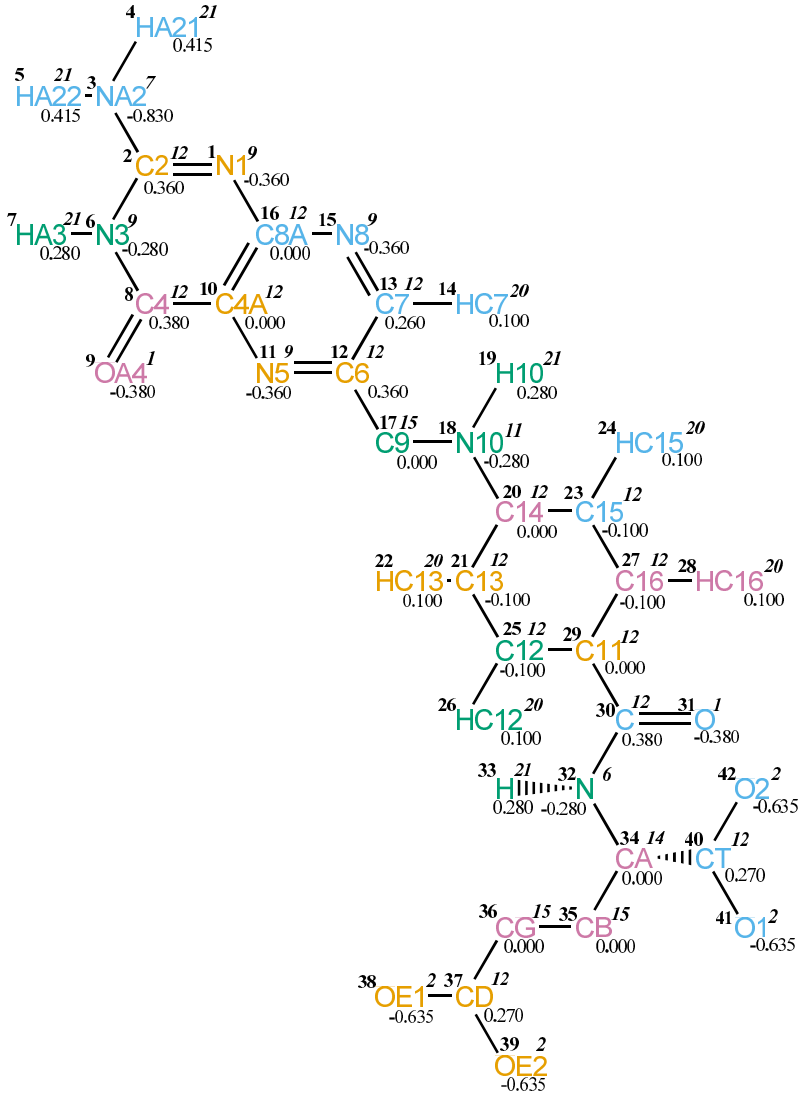


FIGURE 4.219. FOL non-bonded parameters.

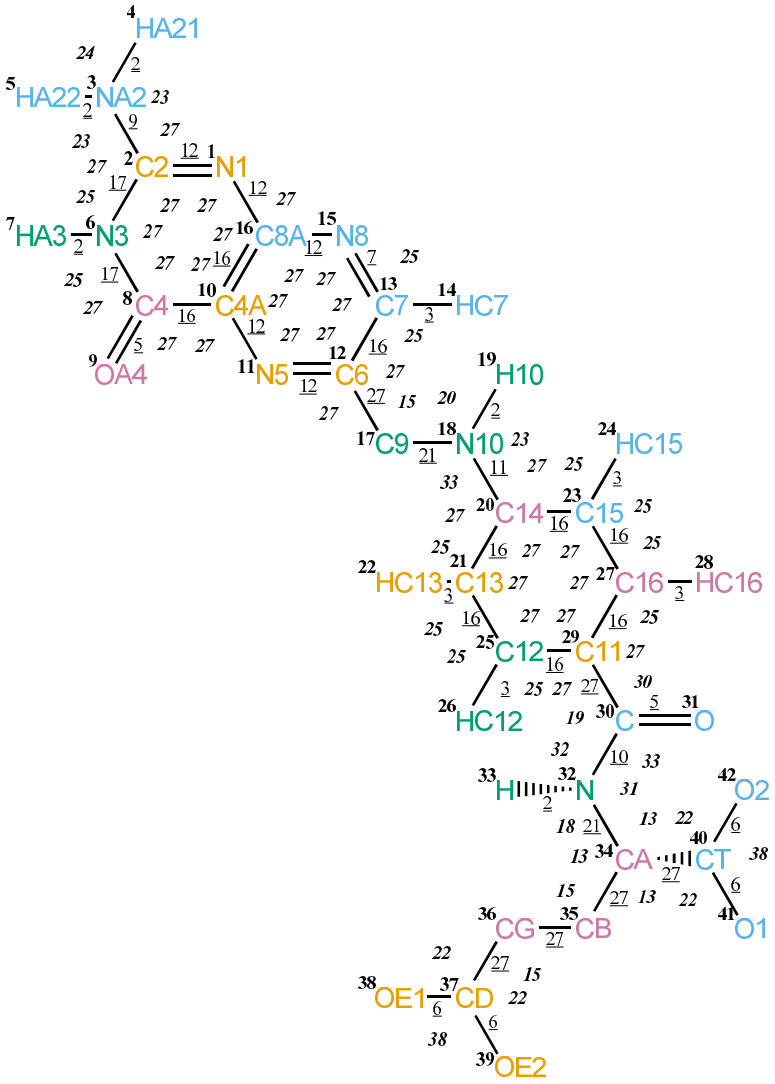


FIGURE 4.220. FOL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	N1	9	14	-0.36000	2 3 6 7 8 10 11 13 15 16
2	C2	12	12	0.36000	3 4 5 6 7 8 9 10 15 16
3	NA2	7	14	-0.83000	4 5 6 7 8 16
4	HA21	21	1	0.41500	5
5	HA22	21	1	0.41500	
6	N3	9	14	-0.28000	7 8 9 10 11 16
7	HA3	21	1	0.28000	8 9 10
8	C4	12	12	0.38000	9 10 11 12 15 16
9	OA4	1	16	-0.38000	10 11 16
10	C4A	12	12	0.00000	11 12 13 15 16 17
11	N5	9	14	-0.36000	12 13 14 15 16 17
12	C6	12	12	0.36000	13 14 15 16 17 18
13	C7	12	12	0.26000	14 15 16 17
14	HC7	20	1	0.10000	15 16 17
15	N8	9	14	-0.36000	16 17
16	C8A	12	12	0.00000	
17	C9	15	4	0.00000	18 19 20
18	N10	11	14	-0.28000	19 20 21 22 23 24 25 27
19	H10	21	1	0.28000	20
20	C14	12	12	0.00000	21 22 23 24 25 26 27 28 29
21	C13	12	12	-0.10000	22 23 24 25 26 27 29 30
22	HC13	20	1	0.10000	23 25 26 29
23	C15	12	12	-0.10000	24 25 27 28 29 30
24	HC15	20	1	0.10000	27 28 29
25	C12	12	12	-0.10000	26 27 28 29 30
26	HC12	20	1	0.10000	27 29 30
27	C16	12	12	-0.10000	28 29 30
28	HC16	20	1	0.10000	29 30
29	C11	12	12	0.00000	30 31 32
30	C	12	12	0.38000	31 32 33 34
31	O	1	16	-0.38000	32
32	N	6	14	-0.28000	33 34 35 40
33	H	21	1	0.28000	34
34	CA	14	3	0.00000	35 36 40 41 42
35	CB	15	4	0.00000	36 37 40
36	CG	15	4	0.00000	37 38 39
37	CD	12	12	0.27000	38 39
38	OE1	2	16	-0.63500	39
39	OE2	2	16	-0.63500	
40	CT	12	12	0.27000	41 42
41	O1	2	16	-0.63500	42
42	O2	2	16	-0.63500	

TABLE 4.546: Atoms of building block FOL.

I	J	Type
1	2	12
1	16	12
2	3	9
2	6	17
3	4	2
3	5	2
6	7	2
6	8	17
8	9	5
8	10	16
10	11	12
10	16	16
11	12	12
12	13	16
12	17	27
13	14	3
13	15	7
15	16	12
17	18	21
18	19	2
18	20	11
20	21	16
20	23	16
21	22	3
21	25	16
23	24	3
23	27	16
25	26	3
25	29	16
27	28	3
27	29	16
29	30	27
30	31	5
30	32	10
32	33	2
32	34	21
34	35	27
34	40	27
35	36	27
36	37	27
37	38	6
37	39	6
40	41	6
40	42	6

TABLE 4.547: Bonds of building block FOL.

I	J	K	Type
2	1	16	27
1	2	3	27
1	2	6	27
3	2	6	27
2	3	4	23
2	3	5	23
4	3	5	24
2	6	7	25
2	6	8	27
7	6	8	25
6	8	9	27
6	8	10	27
9	8	10	27
8	10	11	27
8	10	16	27
11	10	16	27
10	11	12	27
11	12	13	27
11	12	17	27
13	12	17	27
12	13	14	25
12	13	15	27
14	13	15	25
13	15	16	27
1	16	10	27
1	16	15	27
10	16	15	27
12	17	18	15
17	18	19	20
17	18	20	33
19	18	20	23
18	20	21	27
18	20	23	27
21	20	23	27
20	21	22	25
20	21	25	27
22	21	25	25
20	23	24	25
20	23	27	27
24	23	27	25
21	25	26	25
21	25	29	27
26	25	29	25
23	27	28	25
23	27	29	27
28	27	29	25

TABLE 4.548: continues on next page.

I	J	K	Type
25	29	27	27
25	29	30	27
27	29	30	27
29	30	31	30
29	30	32	19
31	30	32	33
30	32	33	32
30	32	34	31
33	32	34	18
32	34	35	13
32	34	40	13
35	34	40	13
34	35	36	15
35	36	37	15
36	37	38	22
36	37	39	22
38	37	39	38
34	40	41	22
34	40	42	22
41	40	42	38

TABLE 4.548: Bond angles of building block FOL.

I	J	K	L	Type
1	2	3	4	14
11	12	17	18	40
12	17	18	20	39
17	18	20	21	14
25	29	30	32	10
29	30	32	34	14
30	32	34	40	39
32	34	35	36	34
32	34	40	41	40
34	35	36	37	34
35	36	37	38	40

TABLE 4.549: Dihedral angles of building block FOL.

I	J	K	L	Type
1	2	6	8	1
2	1	6	3	1
2	1	16	10	1
2	6	8	10	1
3	4	5	2	1
6	2	8	7	1
6	8	10	16	1
8	6	10	9	1
8	10	16	1	1
10	8	11	16	1
10	11	12	13	1
11	10	16	15	1
11	12	13	15	1
12	11	13	17	1
12	13	15	16	1
13	12	15	14	1
13	15	16	10	1
16	1	2	6	1
16	1	15	10	1
16	10	11	12	1
18	17	20	19	1
20	21	23	18	1
20	21	25	29	1
20	23	27	29	1
21	20	23	27	1
21	20	25	22	1
21	25	29	27	1
23	20	21	25	1
23	20	27	24	1
23	27	29	25	1
25	21	29	26	1
27	23	29	28	1
29	25	27	30	1
30	29	32	31	1
32	30	34	33	1
34	32	40	35	2
37	38	39	36	1
40	34	42	41	1

TABLE 4.550: Improper dihedral angles of building block FOL.

Solute building block: trimethoprim (deprotonated at N1; neutral)
Name: TMP

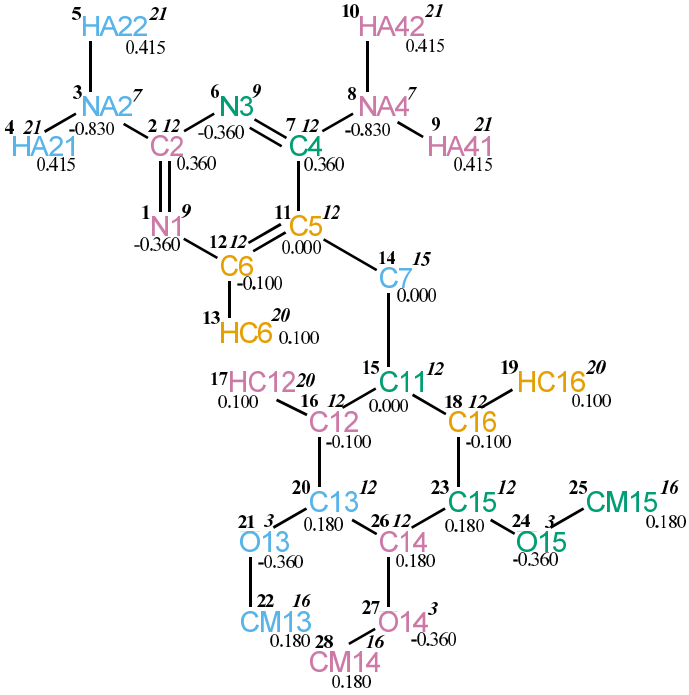


FIGURE 4.221. TMP non-bonded parameters.

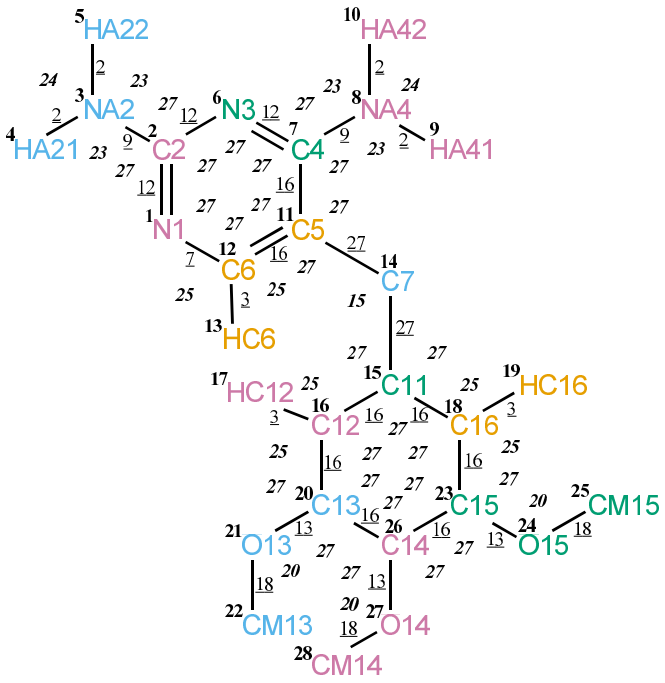


FIGURE 4.222. TMP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	N1	9	14	-0.36000	2 3 6 7 11 12 13 14
2	C2	12	12	0.36000	3 4 5 6 7 8 11 12 13
3	NA2	7	14	-0.83000	4 5 6 7 12
4	HA21	21	1	0.41500	5
5	HA22	21	1	0.41500	
6	N3	9	14	-0.36000	7 8 11 12 14
7	C4	12	12	0.36000	8 9 10 11 12 13 14
8	NA4	7	14	-0.83000	9 10 11 12 14
9	HA41	21	1	0.41500	10
10	HA42	21	1	0.41500	
11	C5	12	12	0.00000	12 13 14 15
12	C6	12	12	-0.10000	13 14
13	HC6	20	1	0.10000	14
14	C7	15	4	0.00000	15 16 17 18 19 20 23
15	C11	12	12	0.00000	16 17 18 19 20 21 23 24 26
16	C12	12	12	-0.10000	17 18 19 20 21 23 26 27
17	HC12	20	1	0.10000	18 20 21 26
18	C16	12	12	-0.10000	19 20 23 24 26 27
19	HC16	20	1	0.10000	23 24 26
20	C13	12	12	0.18000	21 22 23 24 26 27
21	O13	3	16	-0.36000	22 23 26 27
22	CM13	16	5	0.18000	
23	C15	12	12	0.18000	24 25 26 27
24	O15	3	16	-0.36000	25 26 27
25	CM15	16	5	0.18000	
26	C14	12	12	0.18000	27 28
27	O14	3	16	-0.36000	28
28	CM14	16	5	0.18000	

TABLE 4.551. Atoms of building block TMP.

I	J	Type
1	2	12
1	12	7
2	3	9
2	6	12
3	4	2
3	5	2
6	7	12
7	8	9
7	11	16
8	9	2
8	10	2
11	12	16
11	14	27
12	13	3
14	15	27
15	16	16
15	18	16
16	17	3
16	20	16
18	19	3
18	23	16
20	21	13
20	26	16
21	22	18
23	24	13
23	26	16
24	25	18
26	27	13
27	28	18

TABLE 4.552. Bonds of building block TMP.

I	J	K	Type
2	1	12	27
1	2	3	27
1	2	6	27
3	2	6	27
2	3	4	23
2	3	5	23
4	3	5	24
2	6	7	27
6	7	8	27
6	7	11	27
8	7	11	27
7	8	9	23
7	8	10	23
9	8	10	24
7	11	12	27
7	11	14	27
12	11	14	27
1	12	11	27
1	12	13	25
11	12	13	25
11	14	15	15
14	15	16	27
14	15	18	27
16	15	18	27
15	16	17	25
15	16	20	27
17	16	20	25
15	18	19	25
15	18	23	27
19	18	23	25
16	20	21	27
16	20	26	27
21	20	26	27
20	21	22	20
18	23	24	27
18	23	26	27
24	23	26	27
23	24	25	20
20	26	23	27
20	26	27	27
23	26	27	27
26	27	28	20

TABLE 4.553. Bond angles of building block TMP.

I	J	K	L	Type
1	2	3	4	14
11	7	8	9	14
7	11	14	15	40
11	14	15	16	40
16	20	21	22	11
16	20	21	22	12
18	23	24	25	11
18	23	24	25	12
20	26	27	28	11

TABLE 4.554. Dihedral angles of building block TMP.

I	J	K	L	Type
1	2	6	7	1
2	1	6	3	1
2	1	12	11	1
2	6	7	11	1
3	4	5	2	1
6	7	11	12	1
7	6	11	8	1
7	11	12	1	1
8	9	10	7	1
11	7	12	14	1
12	1	2	6	1
12	1	11	13	1
15	16	18	14	1
15	16	20	26	1
15	18	23	26	1
16	15	18	23	1
16	15	20	17	1
16	20	26	23	1
18	15	16	20	1
18	15	23	19	1
18	23	26	20	1
20	16	21	26	1
23	18	24	26	1
26	20	27	23	1

TABLE 4.555. Improper dihedral angles of building block TMP.

Solute building block: 3-phospho-D-glycerate (charge -2e)
Name: PDG

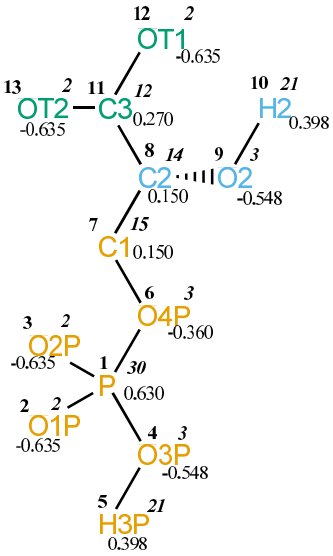


FIGURE 4.223. PDG non-bonded parameters.

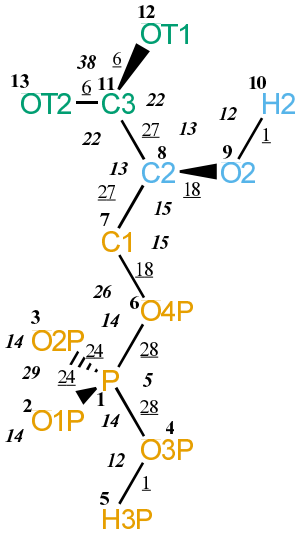


FIGURE 4.224. PDG bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	P	30	31	0.63000	2 3 4 5 6 7
2	O1P	2	16	-0.63500	3 4 5 6
3	O2P	2	16	-0.63500	4 5 6
4	O3P	3	16	-0.54800	5 6
5	H3P	21	1	0.39800	6
6	O4P	3	16	-0.36000	7 8
7	C1	15	4	0.15000	8 9 11
8	C2	14	3	0.15000	9 10 11 12 13
9	O2	3	16	-0.54800	10 11
10	H2	21	1	0.39800	
11	C3	12	12	0.27000	12 13
12	OT1	2	16	-0.63500	13
13	OT2	2	16	-0.63500	

TABLE 4.556. Atoms of building block PDG.

I	J	Type
1	2	24
1	3	24
1	4	28
1	6	28
4	5	1
6	7	18
7	8	27
8	9	18
8	11	27
9	10	1
11	12	6
11	13	6

TABLE 4.557. Bonds of building block PDG.

I	J	K	Type
2	1	3	29
2	1	4	14
2	1	6	14
3	1	4	14
3	1	6	14
4	1	6	5
1	4	5	12
1	6	7	26
6	7	8	15
7	8	9	15
7	8	11	13
9	8	11	13
8	9	10	12
8	11	12	22
8	11	13	22
12	11	13	38

TABLE 4.558. Bond angles of building block PDG.

I	J	K	L	Type
6	1	4	5	20
6	1	4	5	27
4	1	6	7	20
4	1	6	7	27
1	6	7	8	7
1	6	7	8	22
6	7	8	11	34
7	8	9	10	23
7	8	11	12	40

TABLE 4.559. Dihedral angles of building block PDG.

I	J	K	L	Type
8	7	11	9	2
11	8	13	12	1

TABLE 4.560. Improper dihedral angles of building block PDG.

Solute building block: adenosine-5'-triphosphate (ATP; charge -3e)
Name: ATP

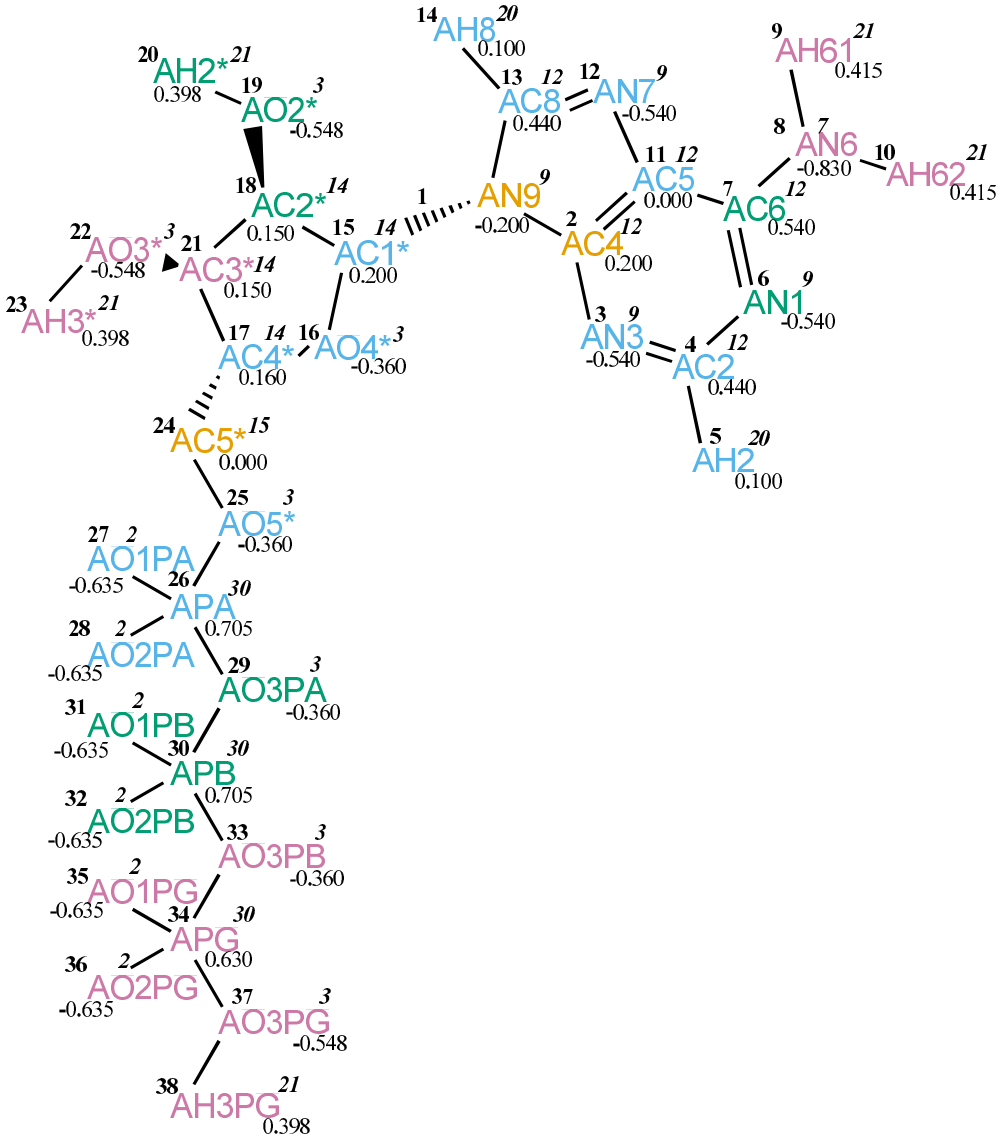


FIGURE 4.225. ATP non-bonded parameters.

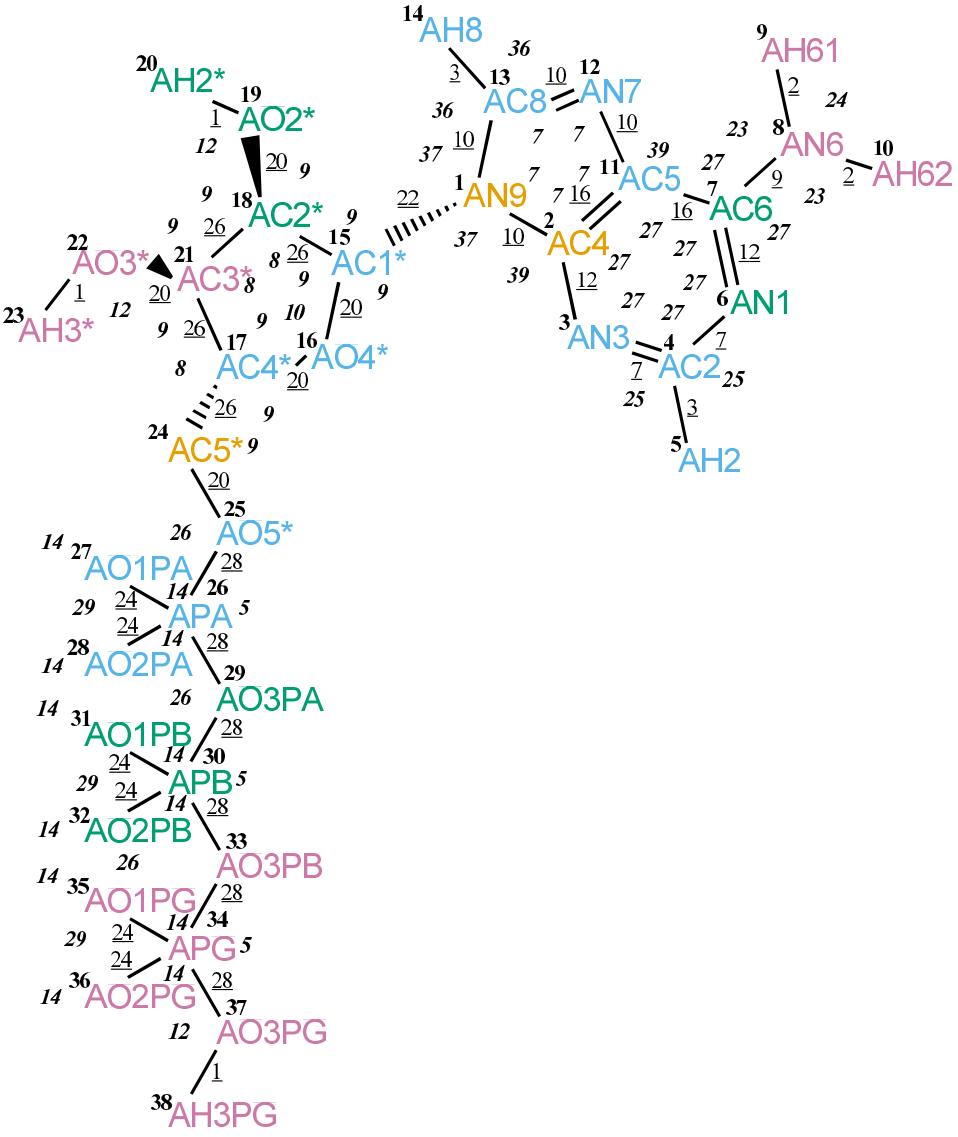


FIGURE 4.226. ATP bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	AN9	9	14	-0.20000	2 3 4 7 11 12 13 14 15 16 18
2	AC4	12	12	0.20000	3 4 5 6 7 8 11 12 13 14 15
3	AN3	9	14	-0.54000	4 5 6 7 11 12 13 15
4	AC2	12	12	0.44000	5 6 7 8 11
5	AH2	20	1	0.10000	6 7
6	AN1	9	14	-0.54000	7 8 11 12
7	AC6	12	12	0.54000	8 9 10 11 12 13
8	AN6	7	14	-0.83000	9 10 11 12
9	AH61	21	1	0.41500	10 12
10	AH62	21	1	0.41500	12
11	AC5	12	12	0.00000	12 13 14 15
12	AN7	9	14	-0.54000	13 14 15
13	AC8	12	12	0.44000	14 15
14	AH8	20	1	0.10000	15
15	AC1*	14	3	0.20000	16 17 18 19 21
16	AO4*	3	16	-0.36000	17 18 21 24
17	AC4*	14	3	0.16000	18 21 22 24 25
18	AC2*	14	3	0.15000	19 20 21 22
19	AO2*	3	16	-0.54800	20 21
20	AH2*	21	1	0.39800	
21	AC3*	14	3	0.15000	22 23 24
22	AO3*	3	16	-0.54800	23
23	AH3*	21	1	0.39800	
24	AC5*	15	4	0.00000	25 26
25	AO5*	3	16	-0.36000	26 27 28 29
26	APA	30	31	0.70500	27 28 29 30
27	AO1PA	2	16	-0.63500	28 29
28	AO2PA	2	16	-0.63500	29
29	AO3PA	3	16	-0.36000	30 31 32 33
30	APB	30	31	0.70500	31 32 33 34
31	AO1PB	2	16	-0.63500	32 33
32	AO2PB	2	16	-0.63500	33
33	AO3PB	3	16	-0.36000	34 35 36 37
34	APG	30	31	0.63000	35 36 37 38
35	AO1PG	2	16	-0.63500	36 37 38
36	AO2PG	2	16	-0.63500	37 38
37	AO3PG	3	16	-0.54800	38
38	AH3PG	21	1	0.39800	

TABLE 4.561: Atoms of building block ATP.

I	J	Type
1	2	10
1	13	10
1	15	22
2	3	12
2	11	16
3	4	7
4	5	3
4	6	7
6	7	12
7	8	9
7	11	16
8	9	2
8	10	2
11	12	10
12	13	10
13	14	3
15	16	20
15	18	26
16	17	20
17	21	26
17	24	26
18	19	20
18	21	26
19	20	1
21	22	20
22	23	1
24	25	20
25	26	28
26	27	24
26	28	24
26	29	28
29	30	28
30	31	24
30	32	24
30	33	28
33	34	28
34	35	24
34	36	24
34	37	28
37	38	1

TABLE 4.562: Bonds of building block ATP.

I	J	K	Type
2	1	13	7
2	1	15	37
13	1	15	37
1	2	3	39
1	2	11	7
3	2	11	27
2	3	4	27
3	4	5	25
3	4	6	27
5	4	6	25
4	6	7	27
6	7	8	27
6	7	11	27
8	7	11	27
7	8	9	23
7	8	10	23
9	8	10	24
2	11	7	27
2	11	12	7
7	11	12	39
11	12	13	7
1	13	12	7
1	13	14	36
12	13	14	36
1	15	16	9
1	15	18	9
16	15	18	9
15	16	17	10
16	17	21	9
16	17	24	9
21	17	24	8
15	18	19	9
15	18	21	8
19	18	21	9
18	19	20	12
17	21	18	8
17	21	22	9
18	21	22	9
21	22	23	12
17	24	25	9
24	25	26	26
25	26	27	14
25	26	28	14
25	26	29	5
27	26	28	29
27	26	29	14

TABLE 4.563: continues on next page.

I	J	K	Type
28	26	29	14
26	29	30	26
29	30	31	14
29	30	32	14
29	30	33	5
31	30	32	29
31	30	33	14
32	30	33	14
30	33	34	26
33	34	35	14
33	34	36	14
33	34	37	5
35	34	36	29
35	34	37	14
36	34	37	14
34	37	38	12

TABLE 4.563: Bond angles of building block ATP.

I	J	K	L	Type
2	1	15	16	16
11	7	8	9	14
18	15	16	17	29
1	15	18	19	17
16	15	18	19	18
16	15	18	21	17
16	15	18	21	34
15	16	17	21	29
16	17	21	18	17
16	17	21	22	18
24	17	21	18	34
24	17	21	22	17
16	17	24	25	8
16	17	24	25	25
21	17	24	25	17
21	17	24	25	34
15	18	19	20	23
15	18	21	17	34
15	18	21	22	17
19	18	21	17	17
19	18	21	22	18
17	21	22	23	23
17	24	25	26	7
17	24	25	26	22
24	25	26	29	20
24	25	26	29	27
25	26	29	30	20
25	26	29	30	27
26	29	30	33	20
26	29	30	33	27
29	30	33	34	20
29	30	33	34	27
30	33	34	37	20
30	33	34	37	27
33	34	37	38	20
33	34	37	38	27

TABLE 4.564: Dihedral angles of building block ATP.

I	J	K	L	Type
1	2	11	12	1
1	2	13	15	1
2	1	3	11	1
2	1	13	12	1
2	3	4	6	1
2	11	12	13	1
3	2	11	7	1
3	4	6	7	1
4	3	5	6	1
4	6	7	11	1
6	7	11	2	1
7	6	8	11	1
8	7	9	10	1
11	2	3	4	1
11	2	7	12	1
11	12	13	1	1
13	1	2	11	1
13	1	12	14	1
15	1	16	18	2
17	16	24	21	2
18	19	21	15	2
21	18	22	17	2

TABLE 4.565: Improper dihedral angles of building block ATP.

Solute building block: p-methylbenzyl alcoholate (charge -e)
Name: PMB

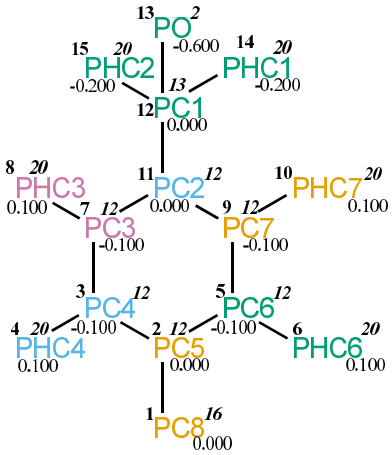


FIGURE 4.227. PMB non-bonded parameters.

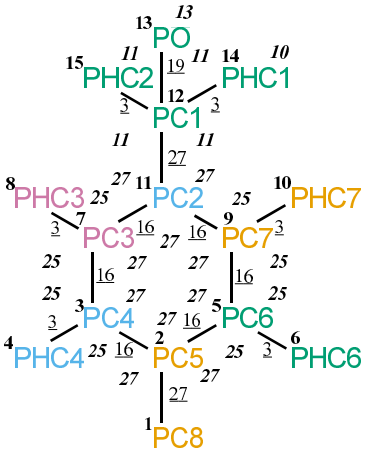


FIGURE 4.228. PMB bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	PC8	16	5	0.00000	2 3 4 5 6 7 9
2	PC5	12	12	0.00000	3 4 5 6 7 8 9 10 11
3	PC4	12	12	-0.10000	4 5 6 7 8 9 11 12
4	PHC4	20	1	0.10000	5 7 8 11
5	PC6	12	12	-0.10000	6 7 9 10 11 12
6	PHC6	20	1	0.10000	9 10 11
7	PC3	12	12	-0.10000	8 9 10 11 12
8	PHC3	20	1	0.10000	9 11 12
9	PC7	12	12	-0.10000	10 11 12
10	PHC7	20	1	0.10000	11 12
11	PC2	12	12	0.00000	12 13 14 15
12	PC1	13	12	0.00000	13 14 15
13	PO	2	16	-0.60000	14 15
14	PHC1	20	1	-0.20000	15
15	PHC2	20	1	-0.20000	

TABLE 4.566. Atoms of building block PMB.

I	J	Type
1	2	27
2	3	16
2	5	16
3	4	3
3	7	16
5	6	3
5	9	16
7	8	3
7	11	16
9	10	3
9	11	16
11	12	27
12	13	19
12	14	3
12	15	3

TABLE 4.567. Bonds of building block PMB.

I	J	K	Type
1	2	3	27
1	2	5	27
3	2	5	27
2	3	4	25
2	3	7	27
4	3	7	25
2	5	6	25
2	5	9	27
6	5	9	25
3	7	8	25
3	7	11	27
8	7	11	25
5	9	10	25
5	9	11	27
10	9	11	25
7	11	9	27
7	11	12	27
9	11	12	27
11	12	13	13
11	12	14	11
11	12	15	11
13	12	14	11
13	12	15	11
14	12	15	10

TABLE 4.568. Bond angles of building block PMB.

I	J	K	L	Type
7	11	12	13	40

TABLE 4.569. Dihedral angles of building block PMB.

I	J	K	L	Type
2	3	5	1	1
2	3	7	11	1
2	5	9	11	1
3	2	5	9	1
3	2	7	4	1
3	7	11	9	1
5	2	3	7	1
5	2	9	6	1
5	9	11	7	1
7	3	11	8	1
9	5	11	10	1
11	7	9	12	1

TABLE 4.570. Improper dihedral angles of building block PMB.

Solute building block: benzoic acid
Name: BA

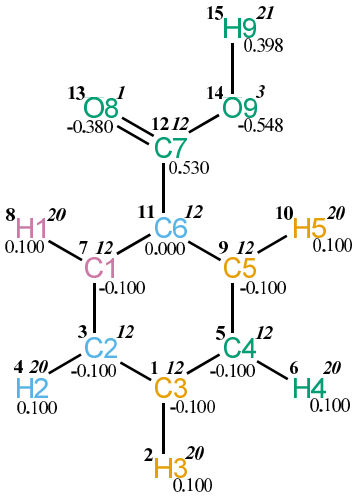


FIGURE 4.229. BA non-bonded parameters.

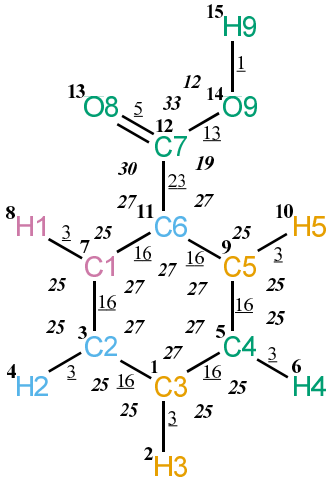


FIGURE 4.230. BA bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	C3	12	12	-0.10000	2 3 4 5 6 7 8 9 10 11
2	H3	20	1	0.10000	3 4 5 6 7 9
3	C2	12	12	-0.10000	4 5 6 7 8 9 11 12
4	H2	20	1	0.10000	5 7 8 11
5	C4	12	12	-0.10000	6 7 9 10 11 12
6	H4	20	1	0.10000	9 10 11
7	C1	12	12	-0.10000	8 9 10 11 12
8	H1	20	1	0.10000	9 11 12
9	C5	12	12	-0.10000	10 11 12
10	H5	20	1	0.10000	11 12
11	C6	12	12	0.00000	12 13 14
12	C7	12	12	0.53000	13 14 15
13	O8	1	16	-0.38000	14
14	O9	3	16	-0.54800	15
15	H9	21	1	0.39800	

TABLE 4.571. Atoms of building block BA.

I	J	Type
1	2	3
1	3	16
1	5	16
3	4	3
3	7	16
5	6	3
5	9	16
7	8	3
7	11	16
9	10	3
9	11	16
11	12	23
12	13	5
12	14	13
14	15	1

TABLE 4.572. Bonds of building block BA.

I	J	K	Type
2	1	3	25
2	1	5	25
3	1	5	27
1	3	4	25
1	3	7	27
4	3	7	25
1	5	6	25
1	5	9	27
6	5	9	25
3	7	8	25
3	7	11	27
8	7	11	25
5	9	10	25
5	9	11	27
10	9	11	25
7	11	9	27
7	11	12	27
9	11	12	27
11	12	13	30
11	12	14	19
13	12	14	33
12	14	15	12

TABLE 4.573. Bond angles of building block BA.

I	J	K	L	Type
7	11	12	14	10
11	12	14	15	12

TABLE 4.574. Dihedral angles of building block BA.

I	J	K	L	Type
1	3	5	2	1
1	3	7	11	1
1	5	9	11	1
3	1	5	9	1
3	7	11	9	1
4	1	7	3	1
5	1	3	7	1
5	9	11	7	1
6	1	9	5	1
7	3	11	8	1
9	5	11	10	1
11	7	9	12	1
12	13	14	11	1

TABLE 4.575. Improper dihedral angles of building block BA.

Solute building block: retinol(neutral)
Name: RTOL

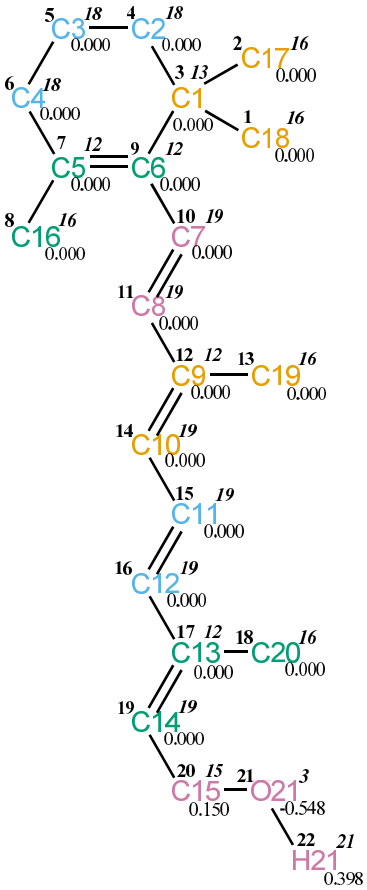


FIGURE 4.231. RTOL non-bonded parameters.

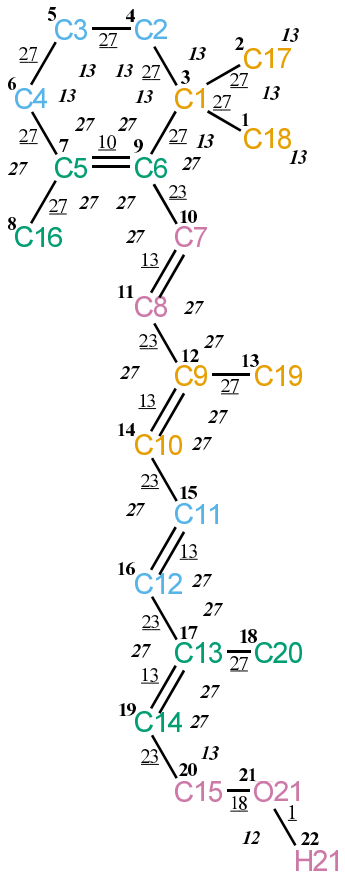


FIGURE 4.232. RTOL bonded parameters.

Seq.	Name	IAC	Mass	Charge	Exclusions
1	C18	16	5	0.00000	2 3 4 9
2	C17	16	5	0.00000	3 4 9
3	C1	13	12	0.00000	4 5 7 9 10
4	C2	18	4	0.00000	5 6 9
5	C3	18	4	0.00000	6 7
6	C4	18	4	0.00000	7 8 9
7	C5	12	12	0.00000	8 9 10
8	C16	16	5	0.00000	9
9	C6	12	12	0.00000	10 11
10	C7	19	3	0.00000	11 12
11	C8	19	3	0.00000	12 13 14
12	C9	12	12	0.00000	13 14 15
13	C19	16	5	0.00000	14
14	C10	19	3	0.00000	15 16
15	C11	19	3	0.00000	16 17
16	C12	19	3	0.00000	17 18 19
17	C13	12	12	0.00000	18 19 20
18	C20	16	5	0.00000	19
19	C14	19	3	0.00000	20 21
20	C15	15	4	0.15000	21 22
21	O21	3	16	-0.54800	22
22	H21	21	1	0.39800	

TABLE 4.576. Atoms of building block RTOL.

I	J	Type
1	3	27
2	3	27
3	4	27
3	9	27
4	5	27
5	6	27
6	7	27
7	8	27
7	9	10
9	10	23
10	11	13
11	12	23
12	13	27
12	14	13
14	15	23
15	16	13
16	17	23
17	18	27
17	19	13
19	20	23
20	21	18
21	22	1

TABLE 4.577. Bonds of building block RTOL.

I	J	K	Type
1	3	2	13
1	3	4	13
1	3	9	13
2	3	4	13
2	3	9	13
4	3	9	13
3	4	5	13
4	5	6	13
5	6	7	13
6	7	8	27
6	7	9	27
8	7	9	27
3	9	7	27
3	9	10	27
7	9	10	27
9	10	11	27
10	11	12	27
11	12	13	27
11	12	14	27
13	12	14	27
12	14	15	27
14	15	16	27
15	16	17	27
16	17	18	27
16	17	19	27
18	17	19	27
17	19	20	27
19	20	21	13
20	21	22	12

TABLE 4.578. Bond angles of building block RTOL.

I	J	K	L	Type
9	3	4	5	34
4	3	9	7	34
3	4	5	6	34
4	5	6	7	34
5	6	7	9	34
6	7	9	3	14
7	9	10	11	34
9	10	11	12	14
10	11	12	14	12
11	12	14	15	14
12	14	15	16	12
14	15	16	17	14
15	16	17	19	12
16	17	19	20	14
17	19	20	21	40
19	20	21	22	23

TABLE 4.579. Dihedral angles of building block RTOL.

I	J	K	L	Type
7	6	9	8	1
9	3	10	7	1
12	11	14	13	1
17	16	19	18	1

TABLE 4.580. Improper dihedral angles of building block RTOL.

CHAPTER 5

GROMOS standard configurations

When simulating a molecule in solution or in crystalline form, the initial positions of solvent molecules surrounding it are to be generated in some way. The GROMOS package comes with files with *standard configurations* of a box with *solvent* molecules. This can be done using the GROMOS++ program sim_box, which requires a standard GROMOS atomic coordinate file containing a number of solvent molecules. Data on the solvent configuration files are given in Tab. 5.1. The configurations are taken from MD simulations at the indicated temperature at constant volume. The atomic coordinates are in nm.

5.1. Water

See Tab. 5.1.

5.2. Chloroform

See Tab. 5.1.

5.3. DMSO

See Tab. 5.1.

5.4. Methanol

See Tab. 5.1.

5.5. Carbontetrachloride

See Tab. 5.1.

solvent	number of molecules	temperature	periodic box type	length of box edge	cut-off radius	file name
		(K)		(nm)	(nm)	
water	5384	300	cubic	5.4937	1.4	spc.cnf
chloroform	1000	293	cubic	5.0816	1.4	chcl3.cnf
DMSO	1024	298	cubic	4.9505	1.4	dmso.cnf
methanol	1000	300	cubic	4.0669	1.4	ch3oh.cnf
CCl4	1000	293	cubic	5.4260	1.4	ccl4.cnf

TABLE 5.1. Standard Solvent Configuration Files.

Bibliography

[1] L.D. Schuler, X. Daura, and W.F. van Gunsteren. An Improved GROMOS96 Force Field for Aliphatic Hydrocarbons in the Condensed Phase. *J. Comput. Chem.*, 22:1205–1218, 2001.

[2] I. Chandrasekhar, M. Kastenholz, R.D. Lins, C. Oostenbrink, L.D. Schuler, D.P. Tieleman, and W.F. van Gunsteren. A consistent potential energy parameter set for lipids: Dipalmitoylphosphatidylcholine as a benchmark of the GROMOS96 45A3 force field. *Eur. Biophys. J.*, 32:67–77, 2003.

[3] T.A. Soares, P.H. Hünenberger, M.A. Kastenholz, V. Kräutler, T. Lenz, R.D. Lins, C. Oostenbrink, and W.F. van Gunsteren. An Improved Nucleic-Acid Parameter Set for the GROMOS Force Field. *J. Comput. Chem.*, 26:725–737, 2005.

[4] R.D. Lins and P.H. Hünenberger. A new GROMOS force field for hexopyranose-based cardohydrates. *J. Comput. Chem.*, 26:1400–1412, 2005.

[5] C. Oostenbrink, A. Villa, A.E. Mark, and W.F. van Gunsteren. A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. *J. Comput. Chem.*, 25:1656, 2004.

[6] D. Poger, W.F. van Gunsteren, and A.E. Mark. A new force field for simulating phosphatidylcholine bilayers. *J. Comput. Chem.*, 31:1117–1125, 2010.

[7] H. Hansen and P.H. Hünenberger. A reoptimized GROMOS force field for hexopyranose-based carbohydrates accounting for the relative free energies of ring conformers, anomers, epimers, hydroxymethyl rotamers and glycosidic linkage conformers. *J. Comput. Chem.*, 32:998–1032, 2011.

[8] N. Schmid, A. P. Eichenberger, A. Choutko, S. Riniker, M. Winger, A.E.. Mark, and W.F. van Gunsteren. Definition and testing of the GROMOS force-field versions: 54A7 and 54B7. *Eur. Biophys. J.*, 40:843–856, 2011.

[9] W.F. van Gunsteren and M. Karplus. Effect of Constraints on the Dynamics of Macromolecules. *Macromolecules*, 15:1528–1544, 1982.

[10] J. Hermans, H.J.C. Berendsen, W.F. van Gunsteren, and J.P.M. Postma. A Consistent Empirical Potential for Water-Protein Interactions. *Biopolymers*, 23:1513–1518, 1984.

[11] L.J. Smith, A.E. Mark, C.M. Dobson, and W.F. van Gunsteren. Comparison of MD simulations and NMR experiments for hen lysozyme: Analysis of local fluctuations, cooperative motions and global changes. *Biochemistry*, 34:10918–10931, 1995.

[12] L.D. Schuler and W.F. van Gunsteren. On the Choice of Dihedral Angle Potential Energy Functions for n-Alkanes. *Mol. Simul.*, 25:301–319, 2000.

[13] M.M. Reif, P.H. Hünenberger, and C. Oostenbrink. New interaction parameters for charged amino acid side chains in the GROMOS force field. *J. Chem. Theory Comput.*, 8:3705–3723, 2012.

[14] D.P. Geerke and W.F. van Gunsteren. Force Field Evaluation for Biomolecular Simulation: Free Enthalpies of Solvation of Polar and Apolar Compounds in Various Solvents. *ChemPhysChem*, 7:671–678, 2006.

[15] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, and J. Hermans. Interaction models for water in relation to protein hydration. In B. Pullman, editor, *Intermolecular Forces*, pages 331–342. Reidel, Dordrecht, 1981.

[16] A. Glättli, X. Daura, and W.F. van Gunsteren. Derivation of an improved SPC model for liquid water: SPC/A and SPC/L. *J. Chem. Phys.*, 116:9811–9828, 2002.

[17] R. Walser, A.E. Mark, W.F. van Gunsteren, M. Lauterbach, and G. Wipff. The effect of force-field parameters on properties of liquids: Parametrization of a simple three-site model for methanol. *J. Chem. Phys.*, 112:10450–10459, 2000.

[18] D.P. Geerke, C. Oostenbrink, N.F.A. van der Vegt, and W.F. van Gunsteren. An Effective Force Field for Molecular Dynamics Simulations of Dimethyl Sulfoxide and Dimethyl Sulfoxide-Water Mixtures. *J. Phys. Chem. B*, 108:1436, 2004.

[19] I.G. Tironi and W.F. van Gunsteren. A molecular dynamics simulation study of chloroform. *Mol. Phys.*, 83:381–403, 1994.

[20] I.G. Tironi, P. Fontana, and W.F. van Gunsteren. A molecular dynamics simulation study of liquid carbon tetrachloride. *Mol. Simul.*, 18:1–11, 1996.

[21] L.J. Smith, H.J.C. Berendsen, and W.F. van Gunsteren. Computer Simulation of Urea-Water Mixtures: A Test of Force Field Parameters for Use in Biomolecular Simulations. *J. Phys. Chem. B*, 108:1065–1071, 2004.

[22] P.J. Gee and W.F. van Gunsteren. Acetonitrile revisited: a molecular dynamics study of the liquid phase. *Mol. Phys.*, 104:477–483, 2006.

[23] N. Hansen, P. Kraus, H. Sassmannshausen, T. Timmerscheidt, and W.F. van Gunsteren. An effective force field for molecular dynamics simulations of dimethyl sulfone. *Mol. Phys.*, 109:2593–2605, 2011.

[24] S.J. Bachmann and W.F. van Gunsteren. An improved polarisable water model for use in biomolecular simulation. *J. Chem. Phys.*, 141:22D515, 2014.

[25] H. Yu, D.P. Geerke, H. Liu, and W.F. van Gunsteren. Molecular dynamics simulations of liquid methanol and methanol-water mixtures with polarizable models. *J. Comput. Chem.*, 27:1494–1504, 2006.

[26] S.J. Bachmann and W.F. van Gunsteren. Polarisable model for DMSO and DMSO-water mixtures. *J. Phys. Chem. B*, 118:10175–10186, 2014.

[27] Z. Lin, A.P. Kunz, and W.F. van Gunsteren. A one-site polarizable model for liquid chloroform: COS/C. *Mol. Phys.*, 108:1749–1757, 2010.

[28] A.P.E. Kunz, A.P. Eichenberger, and W.F. van Gunsteren. A simple, efficient polarisable molecular model for liquid carbon tetrachloride. *Mol. Phys.*, 109:365–372, 2011.
[29] Z. Lin, S.J. Bachmann, and W.F. van Gunsteren. GROMOS polarisable charge-on-spring models for liquid urea: COS/U and COS/U2. *J. Chem. Phys.*, 142:094117, 2015.
[30] V.H. Rusu, S.J. Bachmann, and W.F. van Gunsteren. GROMOS polarisable model for acetone. *Mol. Phys.*, 114:845–854, 2016.
[31] O.M. Szklarczyk, S.J. Bachmann, and W.F. van Gunsteren. A polarisable empirical force field for molecular dynamics simulation of liquid hydrocarbons. *J. Comput. Chem.*, 35:789–801, 2014.
[32] S. Riniker and W.F. van Gunsteren. A simple, efficient polarisable coarse-grained water model for molecular dynamics simulations. *J. Chem. Phys.*, 134:084110, 2011.
[33] J.R. Allison, S. Riniker, and W.F. van Gunsteren. Coarse-grained models for the solvents dimethyl sulfoxide, chloroform and methanol. *J. Chem. Phys.*, 136:054505, 2012.
[34] W. Huang, S. Riniker, and W.F. van Gunsteren. Rapid sampling of folding equilibria of β-peptides in methanol using a supramolecular solvent model. *J. Chem. Theory Comput.*, 10:2213–2223, 2014.
[35] A.P. Eichenberger, W. Huang, S. Riniker, and W.F. van Gunsteren. A supra-atomic coarse-grained GROMOS force field for aliphatic hydrocarbons in the liquid phase. *J. Chem. Theory Comput.*, 11:2925–2937, 2015.
[36] O. Szklarczyk, E. Arvaniti, and W.F. van Gunsteren. Polarisable coarse-grained models for molecular dynamics simulation of liquid cyclohexane. *J. Comput. Chem.*, 36:1311–1321, 2015.
[37] IUPAC-IUB commission on biochemical nomenclature. Abbreviations and symbols for the description of the conformation of polypeptide chains. Tentative rules (1969). *Biochemistry*, 9:3471–3479, 1970.
[38] H.J.C. Berendsen, J.R. Grigera, and T.P. Straatsma. The Missing Term in Effective Pair Potentials. *J. Phys. Chem.*, 91:6269–6271, 1987.
[39] H. Liu, F. Müller-Plathe, and W.F. van Gunsteren. A Force Field for Liquid Dimethyl Sulfoxide and Physical Properties of Liquid Dimethyl Sulfoxide Calculated Using Molecular Dynamics Simulation. *J. Am. Chem. Soc.*, 117:4363–4366, 1995.

Symbols

Symbol	Meaning
	Common names and abbreviations
GROMOS	The GROMOS software package
MD++	The MD++ simulation engine in C++
GROMOS++	The GROMOS++ analysis package in C++
GROMOS96	The GROMOS96 simulation package (1996)
3D	abbreviation for three dimensions
AA	Atomistic (All Atom) models
BD	Brownian Dynamics simulation
B&S – LEUS	Ball and stick local elevation umbrella sampling
CG	Coarse Grained models
CGEM	Conjugate gradient method for energy minimization
FRCG	Fletcher-Reeves conjugate gradient method for energy minimization
PRCG	Polak-Ribiére conjugate gradient method for energy minimization
COG	Center of geometry
COS	Charge On Spring approach
CP	Car Parrinello approach
DF	Distancefield
DOF	Degrees of freedom (abbreviation)
DPD	Diffusive Particle Dynamics simulation
`doxygen`	Documentation platform
EM	Energy minimisation
EDS	Enveloping distribution sampling
FBC	Fixed boundary conditions
HBC	Hyper-spherical boundary conditions
LE	Local elevation
LEUS	Local elevation umbrella sampling
LS	Lattice-sum method
MC	Monte Carlo sampling
MD	Molecular Dynamics simulation
NOE	Nuclear Overhauser Effect
PBC	Periodic boundary conditions
PPPM	Particle-particle–particle-mesh (P^3M) method
QM	Quantum Mechanical models
QMD	Quantum Molecular Dynamics simulation
RDF	Radial distribution function
RE	Replica Exchange
REMD	Replica Exchange Molecular Dynamics simulation
RF	Reaction-field method
RMSD	Root-mean-square difference
RMSF	Root-mean-square fluctuation
SD	Stochastic Dynamics simulation
SDEM	Steepest descent method for energy minimization
TI	Thermodynamic integration
US	Umbrella sampling

Symbol	Meaning
VBC	Vacuum boundary conditions
Physical constants	
h	Planck's constant [0.3990313 kJ mol^{-1} ps]
$\hbar$	Planck's constant divided by 2π [0.06350780 kJ mol^{-1} ps]
$N_{\rm Av}$	Avogadro's number [6.02214 $\times 10^{23}$]
k_B	Boltzmann's constant [1.380662 $\times 10^{-26}$ kJ K^{-1})]
R	Ideal gas constant ($N_{\rm Av} \times k_B$)
c	Speed of light [2.99792458 $\times 10^5$ nm ps^{-1}]
Degrees of freedom and system configuration	
$\mathcal{N}_d$	Number of degrees of freedom of a system
$\mathcal{N}_a$	Number of particles in a system of particles ($\mathcal{N}_d$ =3$\mathcal{N}_a$)
$\mathcal{N}_a^{solu}$	Number of particles the solute consists of
$\boldsymbol{q}$	$3\mathcal{N}_a$-dimensional generalized coordinate vector of a system of particles
$\boldsymbol{p_q}$	$3\mathcal{N}_a$-dimensional generalized momentum vector of a system of particles
$\boldsymbol{r}$	$3\mathcal{N}_a$-dimensional Cartesian coordinate vector of a system of particles
$\boldsymbol{p}$	$3\mathcal{N}_a$-dimensional Cartesian momentum vector of a system of particles
$\boldsymbol{f}$	$3\mathcal{N}_a$-dimensional Cartesian force vector of a system of particles
$\overline{\boldsymbol{f}}$	$3\mathcal{N}_a$-dimensional Cartesian mean force vector of a system of particles
$\boldsymbol{f}^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\mathbf{f}_i^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\boldsymbol{v}$	$3\mathcal{N}_a$-dimensional Cartesian velocity vector of a system of particles
$\mathbf{r}$	3-dimensional Cartesian coordinate vector of a particle
$\mathbf{p}$	3-dimensional Cartesian momentum vector of a particle
$\mathbf{f}$	3-dimensional Cartesian force vector of a particle
$\mathbf{v}$	3-dimensional Cartesian velocity vector of a particle
Ψ $[\Psi(\boldsymbol{r})]$	Wavefunction (position representation; configuration of a quantum-mechanical system of $\mathcal{N}_a$ particles)
$\{$ $\boldsymbol{r}$,$\boldsymbol{p}$ $\}$	Phase-space point (Cartesian coordinates; configuration of a classical system of $\mathcal{N}_a$ particles)
(Statistical) thermodynamics	
$\mathcal{F}$	Free energy
G	Gibbs free energy
H	Enthalpy
$\mathcal{U}$	Energy of a system
$\mathcal{S}$	Entropy of a system
$\mathcal{Z}$	Partition function
$\mathcal{T}$	Instantaneous temperature
$\mathcal{T}_o$	Reference temperature
$\mathcal{K}$	Instantaneous kinetic energy of a system
$\mathcal{K}_{tr}$	Instantaneous translational kinetic energy
$\mathcal{K}_{ir}$	Instantaneous internal+rotational kinetic energy
$\mathcal{U}$	Instantaneous total potential energy of a system
$\mathcal{W}$	Instantaneous virial of a system
$\mathcal{P}$	Instantaneous pressure of a system
$\mathcal{V}$	Instantaneous volume of a system
ρ_J	Number particle density of particles J
Miscellaneous	

Symbol	Meaning
t	Time
Δt	discrete time step
$\mathcal{N}_t$	Number of MD steps
P	Probability
m	Mass of a particle
M	Mass of the whole system
$\underline{\mathbf{m}}$	Diagonal mass matrix of a system of $\mathcal{N}_a$ particles
γ	Friction coefficient of a particle
$\underline{\gamma}$	Diagonal friction coefficient matrix of a system of $\mathcal{N}_a$ particles
T	Absolute temperature
β	prefactor: $1/k_BT$
τ_T	relaxation time for the coupling to a temperature bath
$\mathbf{s}$	Vector denoting the collection of all force-field parameters
λ	Coupling parameter Lambda for a lambda dependent Hamiltonian
$\mathcal{N}_\lambda$	Number of λ-values in a TI simulation
H	Heaviside function defined as $H(x) = 0 \; \forall \; x < 0$ and $H(x) = 1 \; \forall \; x > 0$
sign	Sign function: $\text{sign}(x) = 1 \; \forall \; x > 0$ and $\text{sign}(x) = -1 \; \forall \; x < 0$
i	imaginary number, $i^2 = -1$
δ_{ij}	general Kronecker delta
σ	Standard deviation
σ^2	Variance
$\mathcal{N}_{conf}$	Number of configurations in an ensemble
D	Diffusion constant
R_{gyr}	radius of gyration
η	the viscosity of a system
$g(r)$	radial distribution function
s	Smoothness parameter in EDS simulations
E^R	Energy offset parameter in EDS simulations
$\mathcal{N}^{(s)}$	Number of states in EDS simulations
Spatial boundary conditions	
$\underline{\mathcal{B}}$	3×3-matrix of the box-edge vectors (columns) in the reference Cartesian coordinate system (PBC)
$\hat{\mathbf{e}}$	Unit vector
$\mathbf{a}$	First edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{b}$	Second edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{c}$	Third edge vector of a (triclinic) box (in the reference coordinate system)
a	length of first edge of a (triclinic) box
b	length of second edge of a (triclinic) box
c	length of third edge of a (triclinic) box
$\mathbf{T}$	Position vector of the reference corner of a triclinic box (components in the reference coordinate system and vector relative to the origin of this system)
$\underline{\mathbf{L}}$	Computational box matrix (columns defined by the components of edge vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ in the reference coordinate system)
$\underline{\mathbf{B}}$	Edge length matrix (diagonal, elements a, b and c)
α	First edge angle a triclinic box (between $\mathbf{b}$ and $\mathbf{c}$)
β	Second edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{c}$)
γ	Third edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{b}$)
ϕ	First Euler angle of a triclinic box

Symbol	Meaning
θ	Second Euler angle of a triclinic box
ψ	Third Euler angle of a triclinic box
$\check{\mathbf{r}}$	Oblique coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{r}}$	Oblique fractional coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{k}}$	Oblique coordinates of a reciprocal-space vector
$\check{\mathbf{k}}$	Oblique fractional coordinates of a reciprocal-space vector
$\mathbf{l}$	Lattice vector (three-dimensional vector with integer components)
$\mathbf{k}$	Reciprocal-lattice vector ($\mathbf{k} = 2\pi\underline{\mathbf{L}}^{-1}\mathbf{l}$)
$\underline{\mathbf{S}}$	Transformation matrix
$\underline{\mathbf{R}}$	Transformation matrix
$\underline{\mathbf{T}}$	Transformation matrix
Representation of the interaction	
$\hat{\mathcal{H}}$	Hamiltonian operator describing the interaction for quantum-mechanical degrees of freedom
$\hat{\mathcal{K}}$	Kinetic energy operator (kinetic energy contribution to the quantum-mechanical Hamiltonian operator)
$\hat{\mathcal{V}}$	Potential energy operator (potential energy contribution to the quantum-mechanical Hamiltonian operator)
$\mathcal{H}$ $[\mathcal{H}(\boldsymbol{r},\boldsymbol{p})]$	Hamiltonian function describing the interaction for classical degrees of freedom
$\mathcal{K}$ $[\mathcal{K}(\boldsymbol{p})]$	Kinetic energy contribution to the classical Hamiltonian function
$\mathcal{V}$ $[\mathcal{V}(\boldsymbol{r})]$	Potential energy contribution to the classical Hamiltonian function
$\overline{\mathcal{V}}$ $[\overline{\mathcal{V}}(\boldsymbol{r})]$	Potential of mean force contribution to the classical Hamiltonian function
Physical interactions	
φ [Proper dihedral-angle term]	
$\mathcal{V}^{(phys)}$ $[\mathcal{V}^{(phys)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Physical potential energy contribution to $\mathcal{V}$
$\mathcal{V}^{(cov)}$ $[\mathcal{V}^{(cov)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Covalent potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(nbd)}$ $[\mathcal{V}^{(nbd)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Non-bonded potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(b)}$ $[\mathcal{V}^{(b)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond stretching potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\theta)}$ $[\mathcal{V}^{(\theta)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\xi)}$ $[\mathcal{V}^{(\xi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Improper dihedral-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\varphi)}$ $[\mathcal{V}^{(\varphi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Proper dihedral-angle torsion potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(vdw)}$ $[\mathcal{V}^{(vdw)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Van der Waals potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(ele)}$ $[\mathcal{V}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(LJCRF)}$	Sum of the non-bonded potentials $\mathcal{V}^{(vdw)}$ and $\mathcal{V}^{(ele)}$
Physical force-field terms	
$V^{(b)}$ $[V^{(b)}(b;k^{(b)},b^0)]$	Potential energy function associated with the stretching of a single covalent bond (quartic: $V^{(b,q)}$; harmonic: $V^{(b,h)}$; soft harmonic: $V^{(bs,h)}$)
$V_n^{(b)}$ $[V^{(b)}(b_n;k_n^{(b)},b_n^0)]$	Potential energy function associated with the stretching of the nth single covalent bond (quartic: $V_n^{(b,q)}$; harmonic: $V_n^{(b,h)}$; soft harmonic: $V_n^{(bs,h)}$)
$\mathbf{f}^{(b,q)}$	Force due to the bond stretching potential (quartic)
$\mathbf{f}^{(b,h)}$	Force due to the bond stretching potential (harmonic)
$\mathbf{f}^{(bs,h)}$	Force due to the bond stretching potential (soft harmonic)
$N^{(b)}$	Number of covalent bonds in the molecular system
$N^{(bs)}$	Number of soft covalent bonds in the molecular system
$M_n^{(b)}$	Bond type code associated with covalent bond term n

Symbol	Meaning
b_n $[b_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Length of covalent bond n in the considered configuration
b_n^0 $[b^0(M_n^{(b)}, \mathbf{s})]$	Reference length of covalent bond term n
$k_n^{(b,q)}$	Force constant of stretching for covalent bond term n (quartic potential)
$k_n^{(b,h)}$	Force constant of stretching for covalent bond term n (harmonic potential)
$V^{(\theta)}$ $[V^{(\theta)}(\theta; k^{(\theta)}, \theta^0)]$	Potential energy function associated with the bending of a single covalent bond angle (cosine-harmonic: $V^{(\theta,c)}$; soft cosine-harmonic: $V^{(\theta s,c)}$; angle-harmonic: $V^{(\theta,h)}$)
$V_n^{(\theta)}$ $[V_n^{(\theta)}(\theta_n; k_n^{(\theta)}, \theta_n^0)]$	Potential energy function associated with the bending of the nth covalent bond angle (cosine-harmonic: $V_n^{(\theta,c)}$; soft cosine-harmonic: $V_n^{(\theta s,c)}$; angle-harmonic: $V_n^{(\theta,h)}$)
$\mathbf{f}^{(\theta,c)}$	Force due to the bond angle potential (cosine-harmonic)
$\mathbf{f}^{(\theta s,c)}$	Force due to the bond angle potential (soft cosine-harmonic)
$\mathbf{f}^{(\theta,h)}$	Force due to the bond angle potential (angle-harmonic)
$N^{(\theta)}$	Number of covalent bond angles in the molecular system
$M_n^{(\theta)}$	Bond-angle type code associated with covalent bond-angle term n
θ_n $[\theta_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent bond angle n in the considered configuration
θ_n^0 $[\theta^0(M_n^{(\theta)}, \mathbf{s})]$	Reference angle of covalent bond-angle term n
$k_n^{(\theta,c)}$	Force constant of bending for covalent bond-angle term n (cosine-harmonic potential)
$k_n^{(\theta s,c)}$	Force constant of bending for covalent bond-angle term n (soft cosine-harmonic potential)
$k_n^{(\theta,h)}$	Force constant of bending for covalent bond-angle term n (angle-harmonic potential)
$V^{(\xi)}$ $[V^{(\xi)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$V^{(\xi s)}$ $[V^{(\xi s)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$\mathbf{f}^{(\xi)}$	Force due to the improper dihedral-angle potential
$\mathbf{f}^{(\xi s)}$	Force due to the soft improper dihedral-angle potential
$N^{(\xi)}$	Number of covalent improper dihedral angles in the molecular system
$N^{(\xi s)}$	Number of covalent improper dihedral angles in the molecular system
$M_n^{(\xi)}$	Improper dihedral-angle type code associated with covalent improper dihedral-angle term n
ξ_n $[\xi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent improper dihedral angle n in the considered configuration
ξ_n^0 $[\xi^0(M_n^{(\xi)}, \mathbf{s})]$	Reference angle of covalent improper dihedral-angle term n
$k_n^{(\xi)}$	Force constant of bending for covalent improper dihedral-angle term n
$V^{(\varphi)}$ $[V^{(\varphi)}(\varphi; k^{(\varphi)}, \varphi^0)]$	Potential energy function associated with the torsion of a single covalent proper dihedral angle (symmetric potential: $V^{(\varphi,s)}$; generalized: $V^{(\varphi,g)}$)
$\mathbf{f}^{(\varphi,s)}$	Force due to the symmetric proper dihedral-angle potential
$\mathbf{f}^{(\varphi,g)}$	Force due to the generalized proper dihedral-angle potential
$N^{(\varphi)}$	Number of covalent proper dihedral angles in the molecular system
$M_n^{(\varphi)}$	Proper dihedral-angle type code associated with covalent proper dihedral-angle term n
φ_n $[\varphi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent proper dihedral angle n in the considered configuration
φ_n^0 $[\varphi^0(M_n^{(\varphi)}, \mathbf{s})]$	Reference angle (phase shift) of covalent proper dihedral-angle term n
$m_n^{(\varphi)}$ $[m_n^{(\varphi)}(M_n^{(\varphi)}, \mathbf{s})]$	Multiplicity of covalent proper dihedral-angle term n
$k_n^{(\varphi,s)}$	Force constant of torsion for covalent proper dihedral-angle term n (symmetric potential; $\varphi_n^0 = 0, \pi$; $m_n^{(\varphi)} \leq 6$)

Symbol	Meaning
$k_n^{(\varphi,g)}$	Force constant of torsion for covalent proper dihedral-angle term n (generalized potential; $\varphi_n^0 \in [0, 2\pi[$)
q	Partial charge of an atom or site
C_{12}	Van der Waals (Pauli) repulsion coefficient of an atom or site (Lennard-Jones function)
C_6	Van der Waals (London) dispersion coefficient of an atom or site (Lennard-Jones function)
C_{126}	Ratio of Van der Waals coefficients $\frac{C_{12}}{C_6}$ (Lennard-Jones function)
α_{LJ}	Lennard-Jones soft-core switching parameter
α_C	Coulomb soft-core switching parameter
$\mathcal{V}^{(ele,pws)}$ $[\mathcal{V}^{(ele,pws)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Pairwise potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,slf)}$ $[\mathcal{V}^{(ele,slf)}(\underline{\mathcal{B}};\mathbf{s})]$	Self potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,srf)}$ $[\mathcal{V}^{(ele,srf)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Surface potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathbf{f}^{(nbd)}$	Force due to the non-bonded forces
$\Psi_{ij}^{(ele)}$ $[\Psi_{ij}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic influence function associated with the particle pair $i-j$
$\delta_{ij}^{(exc)}$ $[\delta_{ij}^{(exc)}(\mathbf{s})]$	Indicator of non-bonded exclusion for the particle pair $i-j$
$\Psi^{(ele,slf)}$ $[\Psi^{(ele,slf)}(\underline{\mathcal{B}})]$	Electrostatic self influence function
$\psi^{(RF)}$ $[\psi^{(RF)}(\mathrm{x})]$	Influence function at distance x of a particle in RF electrostatics
H $[H(\mathrm{x})]$	Heaviside step function (one if x is positive, zero otherwise)
R_C	Cutoff distance (truncation)
R_{cp}	Short-range cut-off
R_{cl}	Long-range cut-off
R^{cg}	radius of a charge group
N_{cg}	number of atoms belonging to a charge group
R_{RF}	Cutoff distance (onset of the RF continuum; usually set equal to R_C)
ϵ_{RF}	Relative dielectric permittivity of the RF continuum (usually set equal to that of the solvent)
κ_{RF}	Inverse Debye screening length of the RF continuum (usually set to zero)
C_{RF}	Constant characterizing the effect of the RF continuum
$\overline{\mathbf{R}}_{ij}$ $[\overline{\mathbf{R}}_{ij}(\boldsymbol{r})]$	Vector (FBC) or minimum-image vector (PBC) connecting the center of the CG containing particle j to the center of the CG containing particle i (norm $\overline{R}_{ij}$)
$\mathcal{V}^{(ele,pws,RF-CB)}$ $[\mathcal{V}^{(ele,pws,RF-CB)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Coulombic pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RF)}$ $[\mathcal{V}^{(ele,pws,RF-RF)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-dependent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RC)}$ $[\mathcal{V}^{(ele,pws,RF-RC)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-independent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\Psi_{ij}^{(ele,LS-RS)}$ $[\Psi_{ij}^{(ele,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space component of electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\Psi_{ij}^{(ele,LS-KS)}$ $[\Psi_{ij}^{(ele,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space component of the electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-RS)}$ $[\mathcal{V}^{(ele,pws,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-KS)}$ $[\mathcal{V}^{(ele,pws,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\psi^{(LS)}$ $[\psi^{(LS)}(\mathbf{x})]$	Influence function at position $\mathbf{x}$ relative to a particle in LS electrostatics
a	Width of the charge-shaping function
γ $[\gamma(\mathrm{x})]$	Charge-shaping function

Symbol	Meaning
$\hat{\gamma}$ $[\hat{\gamma}\ (\mathrm{x})]$	Fourier transformed charge-shaping function
$\mathbf{E}$	Electric field
$\boldsymbol{\mu}$	Dipole
$\boldsymbol{J}$	
α	Electronic polarisability
$\mathbf{P}$	Polarisation
ϵ	Dielectric permittivity
γ^{pol}	γ to calculate position of off site charge
k^{ho}	harmonic force constant in the COS model
ϕ	Electrostatic potential
Unphysical force-field terms	
$\mathcal{V}^{(spec)}$	Unphysical potential energy
$\mathcal{V}^{(res)}$	Restraint energy
$\mathcal{V}^{(pr)}$	Position restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(c)}$	Force due to the position constraints
$k^{(pr)}$	Force constant of an unphysical position-restraining term
$\mathcal{N}^{(pr)}$	number of positionally restrained atoms
l	Lagrange multiplier for position constraints
$\mathcal{V}^{(dr)}$	Distance restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(dir)}$	Force due to the atom-atom distance restraints
$k^{(dr)}$	Force constant of an unphysical distance-restraining term
$\mathbf{r}^0$	Equilibrium distance of distance restraint
$N^{(dir)}$	Number of atom-atom distance restraints
d_{CH}	carbon-hydrogen distance
d_{CC}	carbon-carbon distance
τ_{dr}	decay time for time-averaged distance restraining
$\mathcal{V}^{(tr)}$	Dihedral-angle restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(tr)}$	Force constant of an unphysical dihedral-angle restraining term
$\mathcal{N}^{(tr)}$	number of restrained dihedral angles
$\mathcal{V}^{(Jr)}$	^3J-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Jr)}$	Force constant of an unphysical ^3J-value restraining term
^3J	J-value or J-coupling constant
$^3J^0$	experimental J-value
J	general representation of a J-value
J^0	experimental J-value
ΔJ^0	width of flat-bottom for J-value restraining
a	a in Karplus relation
b	b in Karplus relation
c	c in Karplus relation
τ^s_{Jr}	period of scaling in periodically-scaled J-value restraining
Δt_ω	time period for which scaling is suspended in periodically-scaled J-value restraining
N_{le}	number of bins in J-value local elevation biasing
$w_{\zeta ni}$	weight of gaussian in J-value LE
$\mathcal{V}^{(Fxr)}$	$\mid F \mid$-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(exr)}$	ρ-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(sxr)}$	symmetry restraining potential energy contribution to $\mathcal{V}^{(phys)}$

Symbol	Meaning
k^{xr}	(harmonic) force constant for the crystallographic restraining
k^{sym}	harmonic force constant for the crystallographic symmetry restraining
F	Structure factor amplitude
ρ	Electron density
S	space group of a crystal
N_{sym}	Number of symmetry operations of a space group
$\mathbb{S}$	Symmetry operator $\mathbb{S} = \mathbf{R}\mathbf{r} + \mathbf{t}$
$\mathbf{R}$	Rotation matrix of a symmetry operator
$\mathbf{t}$	Translation vector of a symmetry operator
$\mathcal{V}^{(Sr)}$	S^2-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Sr)}$	Force constant of an unphysical S^2-value restraining term
S^2	S^2-order parameter
$S^{2,0}$	experimental S^2-value
S	general representation of a S^2-value
S^0	experimental S^2-value
$\mathcal{V}^{(df)}$	Distancefield restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(df)}$	Force due to the atom-atom distance restraints
$k^{(df)}$	Force constant of an unphysical distance-restraining term
l^0	Equilibrium distance of distance restraint
g_s	Distancefield grid distance
$\mathcal{V}^{(le)}$	Local elevation (LE) energy
$\mathcal{V}^{(bias)}$	bias energy
γ	LE basis function
$k^{(le)}$	LE force constant
$\mathbf{r}^{uc}$	unconstrained atomic positions
N_c	Number of constraints
N_{sh}	number of iterations of the SHAKE algorithm
d^0	constraint length
$\mathbf{f}^{uc}$	unconstrained atomic forces

The GROMOS Software for (Bio)Molecular Simulation

Volume 4: Data Structures and Formats

January 9, 2021

Contents

CHAPTER 1

Introduction

GROMOS knows different types of data and data files, which are described in this volume. Two types of information concerning a molecular system can be distinguished.

1. *Topological information*: data on the covalent structure, atomic masses, charges, van der Waals parameters, atom-atom distance restraints specification, ^3J-value restraints specification, local-elevation coordinate specification, etc.
2. *Configurational information*: atomic coordinates and atomic coordinate dependent or related quantities, such as velocities and forces, atom-atom distances, dihedral angles, ^3J-values, energies, size of the computational box, etc.

These two types of information are generally stored in separate files, since configurations change continuously during a simulation, whereas molecular topological data generally do not change. Both types of files, *topological files* and *configurational files,* for a specific molecular system are related through the requirement that in both the sequence of the quantities is the same, e.g.

1. sequence of atoms
2. sequence of atom-atom distance restraints
3. sequence of dihedral angle restraints
4. sequence of ^3J-value restraints

This identity of sequence could be checked e.g. by comparing atom names occurring in topological files with those from the configurational files. However, in order to avoid dependence on naming conventions and to maintain maximum flexibility, this is not done in the GROMOS programs. When molecular information, such as residue numbers and names or atom sequence numbers or names, is present both in a topological file and in a configurational file of a molecular system, the program generally uses the data from the topological file and ignores the corresponding data on the configurational file.

The units of the quantities contained in the different files are all derived from the basic units: nm (length), ps (time), atomic mass units and electronic charge, leading in particular to $\mathrm{kJ} \cdot \mathrm{mol}^{-1}$ as unit of energies. The angles are always given in degrees in the files.

GROMOS data files have a block structure, which is defined in Chap. 2. Topological quantities, variables, blocks and files are described in Chap. 3. Configurational quantities, variables, blocks and files are described in Chap. 4. Two other types of data, molecular topology building blocks and interaction function parameters are described in Chap. 5 and Chap. 6. Library files to be used by the analysis programs of GROMOS++ are described in Chap. 7. Chap. 8 describes the MD++ input file.

CHAPTER 2

Block structure and title record of GROMOS files

GROMOS files are composed of a sequence of blocks, which may be of different type. A block begins with a line (record) containing the *blockname* or *blockidentifier* beginning in the first position. The block ends with a line (record) containing the character string *END* beginning in the first position. A blockname or blockidentifier is a unique string of maximally 25 characters. It may not contain a # symbol in the first position and may not be an END string. Block names are given in upper case only. The currently defined blocknames and their functions are listed in Chap. 12.

Each input or output file of the program MD++, which executes a simulation, starts with a *Title block* (Blockname: TITLE), which may contain any character type of data and is meant to specify the contents of the file. When MD++ reads a file, the title record is always printed in order to check whether the wanted file has been assigned to a specific (reading) unit number. This convention is not followed by GROMOS++, which relies on file names rather than units, and ignores title blocks.

Generally, blocks may be listed in any order. However, when there are obvious dependences between data in different blocks, e.g. the definition of bond-angle types and sequence of bond-angles of a molecule, a specific order is required (the latter after the former).

Data files may contain *comment lines*, which may occur at any position and in any number. A comment line is recognized by the # symbol *in the first position* of the line. GROMOS++ also recognizes as comment any text following a # symbol anywhere on the line. In GROMOS, files are *written* using *fixed format* and are *read* using *free format*.

CHAPTER 3

Topological information

3.1. Introduction

A *molecular topology* file contains information about the topology of a molecular system. In its simplest form it would contain lists of covalent bonds, angles, masses, charges, etc. for all the atoms in the molecular system. When the system contains topologically identical molecules, like water molecules in an aqueous solution or corresponding molecules in different asymmetric units in a unit cell in a crystal, these atom lists would contain redundant information. For MD++ and GROMOS++ the topology has to be specified explicitly for all identical solute units. Since a solvent generally consists of simple molecules like H_2O or CCl_4, it would generally be advantageous to avoid the overhead of handling the possibility of occurrence of internal dihedral angle degrees of freedom, non-bonded interactions, etc. within a solvent molecule. Therefore, a distinction is made between a general part (solutes) and a more restricted part (solvent) of a molecular topology file.

For historical reasons the *general part* of a molecular topology file is denoted by the notation *"solute" molecular topology*, although it may contain any collection of molecules including solvent molecules. The *restricted part* of a molecular topology file is denoted by the notation *"solvent" molecular topology*. In general, this part contains topological data on a single type of solvent molecule, unless a solvent molecule does not fit with the following *restrictions*:

- a solvent molecule must be rigid: no internal interactions like bond-stretching, bond-angle bending, (improper) dihedral torsion and non-bonded interactions are allowed,
- the internal structure of a solvent molecule is maintained by application of distance constraint forces between its atoms,
- a solvent molecule consists of one charge group, the position of the first atom of a solvent molecule is taken to represent the position of this charge group,
- a solvent molecule corresponds to a single "temperature group" and a single "pressure (virial) group",
- the residue or molecule name cannot be specified, it is predefined as SOL,
- position restraining should only be applied to the first atom of a solvent molecule,
- fixed position constraints cannot be applied to solvent atoms,
- solvent parameters cannot be changed using a molecular topology perturbation file for obtaining free energy differences.

If a solvent molecule does not comply with these rules, its topological data must be included in the general or solute part of the molecular topology file.

A molecular topology file often contains fewer atoms than a corresponding configuration file. Let us assume that the former contains a set of molecules forming a "solute" of NRP atoms and a solvent molecule with NRAM atoms. In order to *match* this *molecular topology file*, a *configuration file* must contain the following sequence of atoms (for each block of atomic quantities):

1. the atomic coordinates or related quantities of the NRP "solute" atoms,
2. if the molecular system contains NSM solvent molecules, the atomic coordinates or related quantities of the NSM*NRAM solvent atoms.

Solvent coordinates always appear after solute coordinates in the various blocks of configuration files. All solute atoms should be included explicitly in the molecular topology file.The solvent parts of the molecular topology file are to be chosen as the smallest topologically identical units of each type.

In Sec. 3.2 the content of a molecular topology file is specified. This information is always kept in formatted form. In some applications of GROMOS, like calculating the free energy difference between two different states A and B of a system, it is required to change the molecular topology of the system from one corresponding to state A to another one corresponding to state B. In Sec. 3.3 the way a perturbation (change from A to B) of a molecular topology is to be specified will be discussed. The topological specification of atom-atom distance restraints and distance-field restraints is described in Sec. 3.4. The topological specification of dihedral angle restraints or constraints is described in Sec. 3.5. The topological definition of ^3J-value restraints is described in Sec. 3.6. The topological definition of S^2-order parameter restraints is described in Sec. 3.7. The topological specification of coordinates to be used in the local-elevation search technique is described in Sec. 3.8. The following three sections contain atomic property specifications: friction coefficients and position restraining or fixing indicators (Sec. 3.9-Sec. 3.11).

3.2. Molecular topology

A molecular topology is characterized by some or all of the following quantities, which are stored in a molecular topology file.

`FPEPSI`	$(4\pi\varepsilon_0)^{-1}$, ε_0 = permittivity of vacuum
`HBAR`	$\hbar = h/(2\pi), h$ = Planck's constant
`SPDL`	c = speed of light
`BOLTZ`	k_B = Boltzmann's constant
`TPVER`	real number characterizing the version of the molecular topology
`NRATT`	number of (van der Waals) atom types
`TYPE[1..NRATT]`	names of the different atom types as a function of the integer atom code that defines an atom type (at most 5 characters)
`NRAA2`	number of residues in a solute
`AANM[1..NRAA2]`	residue names as a function of the residue sequence number (at most 5 characters)
`NRP`	number of solute atoms
`MRES[1..NRP]`	residue sequence number of solute atoms ($\leqslant$NRAA2)
`PANM[1..NRP]`	atom name of solute atoms
`IAC[1..NRP]`	integer atom code of solute atoms, determining the type of van der Waals interaction of an atom ($\leqslant$NRATT)
`MASS[1..NRP]`	mass of solute atoms
`CG[1..NRP]`	charge of solute atoms
`CGC[1...NRP]`	Atomic charge group codes. The last atom of a charge group is defined by CGC=1, the others must have CGC=0
`INE[1..NRP]`	number of neighbour atoms that are excluded from the non-bonded interaction with a solute atom
`JNE[1..NRP][1..INE[]]`	excluded neighbours (solute, $\leqslant$NRP); sequence numbers J of atoms that are excluded from the non-bonded interaction with the atom with sequence number I; it is assumed that I<J and that the J's appear in ascending order
`INE14[1..NRP]`	number of third-neighbour atoms of solute atoms, for which special 1-4 van der Waals interaction parameters are used when evaluating the non-bonded interaction

JNE14[1..NRP][1..INE14[]]
: third neighbours (solute, ⩽NRP); sequence numbers J of atoms for which the 1-4 van der Waals parameters are used when calculating the non-bonded interaction with the atom with sequence number I; it is assumed that I<J and that the J's appear in ascending order

NCGB[L]
: number of coarse-grained regions

NRCGF[1..NCGB]
: sequence number of the first coarse-grained solute particle in range

NRCGL[1..NCGB]
: sequence number of the last coarse-grained solute particle in range

MSCAL[1..NCGB]
: scaling factor for pressure correction of a coarse-grained region

NBTY
: number of covalent bond types

CB[1..NBTY]
: force constant of the bond-stretching term of the interaction as a function of the bond-type code, based on a quartic potential

CHB [1..NBTY]
: force constant of the bond-stretching term of the interaction as a function of the bond-type code, based on a harmonic potential

B0[1..NBTY]
: bond length at minimum energy of the bond-stretching term as a function of the bond-type code

NBONH
: number of bonds involving H-atoms in the solute

IBH, JBH[1..NBONH]
: atom sequence numbers of the atoms forming a bond i-j as a function of the bond sequence number (⩽NRP), i is always smaller than j

ICBH[1..NBONH]
: bond-type code as a function of the bond sequence number (⩽NBTY)

NBON
: number of bonds *NOT* involving H-atoms in the solute

IB, JB[1..NBON]
: atom sequence numbers of the atoms forming a bond i-j as a function of the bond sequence number (⩽NRP), i is always smaller than j

ICB[1..NBON]
: bond-type code as a function of the bond sequence number (⩽NBTY)

NBONDP
: number of bonds involving coarse grained particles in the solute

IBDP, JBDP[1..NBONDP]
: sequence numbers of the coarse grained particles forming a bond i-j as a function of the bond sequence number (⩽NRP)

ICBC[1..NBON]
: bond-type code as a function of the bond sequence number (⩽NBTY)

NTTY
: number of bond-angle types

CT[1..NTTY]
: force constant of the bond-angle bending term of the interaction as a function of the bond-angle type code, based on a potential harmonic in the angle cosine

CHT[1..NTTY]
: force constant of the bond-angle bending term of the interaction as a function of the bond-angle type code, based on a potential harmonic in the angle (in energy units per degree^2)

T0[1..NTTY]
: bond angle (in degrees) at minimum energy of the bond-angle bending term as a function of the bond-angle type code; upon reading a molecular topology file by MD++, the bond angle is converted from degrees to radians; this conversion is not performed in GROMOS++

NTHEH
: number of bond-angles involving H-atoms in the solute

ITH,JTH,KTH[1..NTHEH]
: atom sequence numbers of the atoms forming a bond-angle i-j-k as a function of the bond-angle sequence number ($\leqslant$NRP), i is always smaller than k

ICTH[1..NTHEH]
: bond-angle type code as a function of the bond-angle sequence number ($\leqslant$NTTY)

NTHE
: number of bond-angles *NOT* involving H-atoms in the solute

IT,JT,KT[1..NTHE]
: atom sequence numbers of the atoms forming a bond-angle i-j-k as a function of the bond-angle sequence number ($\leqslant$NRP), i is always smaller than k

ICT[1..NTHE]
: bond-angle type code as a function of the bond-angle sequence number ($\leqslant$NTTY)

NQTY
: number of improper (harmonic) dihedral-angle types

CQ[1..NQTY]
: force constant of the harmonic improper dihedral term of the interaction as a function of the improper dihedral-angle type code (in energy units per degree^2); upon reading a molecular topology file by MD++, the force constant is converted to energy per rad^2; this conversion is not performed by GROMOS++

Q0[1..NQTY]
: improper dihedral (in degrees) at minimum energy of the harmonic improper dihedral term as a function of the improper dihedral-angle type code; upon reading a molecular topology file by MD++, the improper dihedral angle is converted from degrees to radians; this conversion is not performed by GROMOS++

NQHIH
: number of improper dihedrals involving H-atoms in the solute

IQH,JQH,KQH,LQH[1..NQHIH]
: atom sequence numbers of the atoms forming improper dihedral i-j-k-l as a function of the improper dihedral sequence number ($\leqslant$NRP), j is always smaller than k

ICQH[1..NQHIH]
: improper dihedral type code as a function of the improper dihedral sequence number ($\leqslant$NQTY)

NQHI
: number of improper dihedrals *NOT* involving H-atoms in the solute

IQ,JQ,KQ,LQ[1..NQHI]
: atom sequence numbers of the atoms forming improper dihedral i-j-k-l as a function of the improper dihedral sequence number ($\leqslant$NRP), j is always smaller than k

ICQ[1..NQHI]
: improper dihedral type code as a function of the improper dihedral sequence number ($\leqslant$NQTY)

NPTY
: number of (trigonometric) dihedral-angle types

CP[1..NPTY]
: force constant of the trigonometric dihedral term of the interaction as a function of the dihedral-angle type code

PD[1..NPTY]
: phase-shift angle (in degrees) of the trigonometric dihedral term as a function of the dihedral-angle type code; upon reading a molecular topology file by MD++, the phase-shift angle is converted from degrees to radians; this conversion is not performed in GROMOS++

NP[1..NPTY]
: multiplicity of the trigonometric dihedral term as a function of the dihedral-angle type code (1, 2, 3, 4, 5, or 6)

NPHIH
: number of dihedrals involving H-atoms in the solute

IPH,JPH,KPH,LPH[1..NPHIH]
: atom sequence numbers of the atoms forming dihedral i-j-k-l as a function of the dihedral sequence number ($\leqslant$NRP), j is always smaller than k

`ICPH[1..NPHIH]` dihedral type code as a function of the dihedral sequence number (⩽NPTY)

`NPHI` number of dihedrals *NOT* involving H-atoms in the solute

`IP,JP,KP,LP[1..NPHI]`
atom sequence numbers of the atoms forming dihedral i-j-k-l as a function of the dihedral sequence number (⩽NRP), j is always smaller than k

`ICP[1..NPHI]` dihedral type code as a function of the dihedral sequence number (⩽NPTY)

`NPPCH` number of cross-dihedrals involving H-atoms in the solute

`APH,BPH,CPH,DPH,EPH,FPH,GPH,HPH[1..NPPCH]`
atom sequence numbers of the atoms forming cross-dihedrals a-b-c-d and e-f-g-h as a function of the dihedral sequence numbers of the separate dihedrals (⩽NRP), a,b,c,d are always smaller or equal to e,f,g,h respectively

`ICCH[1..NPPCH]` dihedral type code as a function of the cross-dihedral sequence number (⩽NPTY)

`NPPC` number of cross-dihedrals *NOT* involving H-atoms in the solute

`AP,BP,CP,DP,EP,FP,GP,HP[1..NPPC]`
atom sequence numbers of the atoms forming cross-dihedrals a-b-c-d and e-f-g-h as a function of the dihedral sequence numbers of the separate dihedrals (⩽NRP), a,b,c,d are always smaller or equal to e,f,g,h respectively

`ICC[1..NPPC]` dihedral type code as a function of the cross-dihedral sequence number (⩽NPTY)

`NRATT2` number of unique pairwise combinations of atom types (=NRATT*(NRATT +1)/2)

`C12[1..NRATT2]` coefficient of the $1/r^{12}$ term in the non-bonded interaction as a function of the occurring pair codes; so, the sequence of atom pairs with integer atom codes ranging from 1 to NRATT is: 1-1, 1-2, ...,1-NRATT, 2-2, 2-NRATT, ..., NRATT-NRATT

`C6[1..NRATT2]` coefficient of the $-1/r^6$ term in the non-bonded interaction as a function of the occurring pair codes

`CS12[1..NRATT2]` coefficient of the $1/r^{12}$ term in the 1-4 non-bonded interaction between third-neighbour atoms as a function of the occurring pair codes

`CS6[1..NRATT2]` coefficient of the $-1/r^6$ term in the 1-4 non-bonded interaction between third-neighbour atoms as a function of the occurring pair codes

`NPPOL` number of polarisable solute atoms (⩽NRP)

`IPOLP[1..NPPOL]` atom sequence numbers of the polarisable solute atoms (⩽NRP)

`ALPP[1..NPPOL]` polarisabilities of solute atoms IPOLP[1..NPPOL]

`QPOLP[1..NPPOL]` size of charge-on-spring connected to polarisable solute atoms IPOLP[1..NPPOL]

`ENOTP[1..NPPOL]` damping level for polarisation of solute atoms IPOLP[1..NPPOL]

`EPP[1..NPPOL]` damping parameter for polarisation of solute atoms IPOLP[1..NPPOL]

`NSPM` number of all separate (covalently linked) molecules within the solute topology (e.g. separate protein chains, co-solute molecules, counterions, co-solvent molecules)

`NSP[1..NSPM]` atom sequence number of the last atom of the successive submolecules (⩽NRP)

NSTM	number of temperature atom groups (used to separate translational from internal-plus-rotational velocity components for kinetic energy evaluation and thermostatting) within the solute topology
NST[1..NSTM]	atom sequence number of the last atom of the successive temperature atom groups ($\leqslant$NRP)
NSVM	number of pressure (virial) atom groups (used to define a group-based pressure) within the solute topology
NSV[1..NSVM]	atom sequence number of the last atom of the successive pressure (virial) atom groups ($\leqslant$NRP)
NLJEX	number of LJ-exceptions
ILJEX, JLJEX[1..NLJEX]	atom sequence numbers of atoms i and j to interact with special LJ-interactions given by LJ-exceptions, i is always smaller than j
LJEXC12[1..NLJEX]	coefficient of the $1/r^{12}$ term in the non-bonded interaction for the corresponding atom pair
LJEXC6[1..NLJEX]	coefficient of the $1/r^{6}$ term in the non-bonded interaction for the corresponding atom pair
NRAM	number of atoms per solvent molecule
ANMS[1..NRAM]	atom name of solvent atoms
IACS[1..NRAM]	integer atom code of solvent atoms determining the type of van der Waals interaction of an atom ($\leqslant$NRATT)
WMASS[1..NRAM]	mass of solvent atoms
CGS[1..NRAM]	charge of solvent atoms
NCONS	number of distance constraints within a solvent molecule
ICONS,JCONS[1..NCONS]	atom sequence numbers of the atoms forming the constraint i-j as a function of the constraint sequence number ($\leqslant$NRAM), i is always smaller than j
CONS[1..NCONS]	constraint length as a function of the constraint sequence number
NVPOL	number of polarisable solvent atoms ($\leqslant$NRAM)
IPOLV[1..NVPOL]	atom sequence number of the polarisable solvent atoms ($\leqslant$NRAM)
ALPV[1..NVPOL]	polarisabilities of solvent atoms IPOLV[1..NVPOL]
QPOLV[1..NVPOL]	size of charge-on-spring connected to polarisable solvent atoms IPOLV[1..NVPOL]
ENOTV[1..NVPOL]	damping level for polarisation of solvent atoms IPOLV[1..NVPOL]
EPV[1..NVPOL]	damping parameter for polarisation of solvent atoms IPOLV[1..NVPOL]
NRSASAA	number of atoms to be considered for SASA implicit solvent interaction function
ISASA[1..NRSASAA]	atom sequence numbers of the atoms to be included in the SASA implicit solvent interaction function
RADI[1..NRSASAA]	atomic radii of the SASA atoms
PI[1..NRSASAA]	atom type-specific parameters P in SASA calculation

`SIGMAI[1..NRSASAA]` atom type-specific scaling parameters for SASA energy term

The blocks of a *molecular topology file* are (apart from the *Title block*) the following:

Physical constants block
Blockname: PHYSICALCONSTANTS

```
     WRITE  (unit,12) FPEPSI
     WRITE  (unit,12) HBAR
     WRITE  (unit,12) SPDL
     WRITE  (unit,12) BOLTZ
12   FORMAT (E15.7)
```

This block replaces the TOPPHYSCON block of GROMOS96.

Version block
Blockname: TOPVERSION

```
     WRITE  (unit,13) TPVER
13   FORMAT (F3.1)
```

The version number expected by GROMOS is 2.0.

Van der Waals atom type sequence and name block
Blockname: ATOMTYPENAME

```
     WRITE  (unit,14) NRATT
     DO 10  K=1, NRATT
10   WRITE  (unit,15) TYPE[K]
14   FORMAT (5I5)
15   FORMAT (A5)
```

Solute residue sequence and name block
Blockname: RESNAME

```
     WRITE  (unit,14) NRAA2
     DO 10  K=1, NRAA2
10   WRITE  (unit,15) AANM[K]
```

Solute atom information block
Blockname: SOLUTEATOM

```
     WRITE  (unit,14) NRP
     DO 10  I=1, NRP
     WRITE  (unit,17) I, MRES[I], PANM[I], IAC[I], MASS[I], CG[I],
                      CGC[I], INE[I], JNE[I][K], K=1, INE[I])
10   WRITE  (unit,18) INE14[I],(JNE14[I][K], K=1, INE14[I])
17   FORMAT (2I5,1X,A5,I4,2F11.5,2I4,6I5)
18   FORMAT (46X,I4,6I5)
```

If INE[I] > 6, then the remaining JNE values are written on the next line using 16I5 as format. Likewise for the JSNE14 values if INE14[I] > 6.

Coarse grained solute information block
Blockname: CGSOLUTE

```
     WRITE  (unit,14) NCGB
     DO 10  K=1, NCGB
10   WRITE  (unit,15) NRCGF[K], NRCGL[K], MSCAL[K]
```

Lennard-Jones interaction exception block
Blockname: LJEXCEPTIONS

```
     WRITE  (unit,14) NLJEX
     DO 10  N=1, NLJEX
10   WRITE  (unit,19) ILJEX[N], JLJEX[N], LJEXC12[N], LJEXC6[N]
19   FORMAT (2I5,2F15.7)
```

Bond interaction type block
Blockname: BONDSTRETCHTYPE

```
     WRITE  (unit,14) NBTY
     DO 10  N=1, NBTY
10   WRITE  (unit,19) CB[N], CHB[N], B0[N]
19   FORMAT (3F15.7)
```

The GROMOS96 BONDTYPE block is still accepted, it only contains force constants for the quartic interaction form, the HARMBONDTYPE block contains only force constants for the harmonic form. If the BONDSTRETCHTYPE block is present, the other two are not allowed.

Solute bonds involving H-atoms block
Blockname: BONDH

```
     WRITE  (unit,14) NBONH
     DO 10  N=1, NBONH
10   WRITE  (unit,14) IBH[N], JBH[N], ICBH[N]
```

Solute bonds NOT involving H-atoms block
Blockname: BOND

```
     WRITE  (unit,14) NBON
     DO 10  N=1, NBON
10   WRITE  (unit,14) IB[N], JB[N], ICB[N]
```

Coarse grained solute bonds block
Blockname: BONDDP

```
     WRITE  (unit,14) NBONCG
     DO 10  N=1, NBONCG
10   WRITE  (unit,14) IBCG[N], JBCG[N], ICBCG[N]
```

Solute distance constraints
Blockname: CONSTRAINT

```
     WRITE  (unit,14) NCON
```

```
     DO 10  N=1, NCON
10   WRITE  (unit,14) IC[N], JC[N], ICC[N]
```

Bond angle interaction type block
Blockname: BONDANGLEBENDTYPE

```
     WRITE  (unit,14) NTTY
     DO 10  N=1, NTTY
10   WRITE  (unit,19) CT[N], CHT[N], T0[N]
19   FORMAT (3F15.7)
```

The GROMOS96 BONDANGLETYPE block is still accepted, it only contains force constants for the cosine harmonic interaction form, the HARMBONDANGLETYPE block contains only force constants for the harmonic form. If the BONDANGLEBENDTYPE block is present, the other two are not allowed.

Solute bond angles involving H-atoms block
Blockname: BONDANGLEH

```
     WRITE  (unit,14) NTHEH
     DO 10  N=1, NTHEH
10   WRITE  (unit,14) ITH[N], JTH[N], KTH[N], ICTH[N]
```

Solute bond angles NOT involving H-atoms block
Blockname: BONDANGLE

```
     WRITE  (unit,14) NTHE
     DO 10  N=1, NTHE
10   WRITE  (unit,14) IT[N], JT[N], KT[N], ICT[N]
```

Improper (harmonic) dihedral angle interaction type block
Blockname: IMPDIHEDRALTYPE

```
     WRITE  (unit,14) NQTY
     DO 10  N=1, NQTY
10   WRITE  (unit,19) CQ[N], Q0[N]
19   FORMAT (3F15.7)
```

Solute improper dihedrals involving H-atoms block
Blockname: IMPDIHEDRALH

```
     WRITE  (unit,14) NQHIH
     DO 10  N=1, NQHIH
10   WRITE  (unit,14) IQH[N], JQH[N], KQH[N], LQH[N], ICQH[N]
```

Solute improper dihedrals NOT involving H-atoms block
Blockname: IMPDIHEDRAL

```
     WRITE  (unit,14) NQHI
     DO 10  N=1, NQHI
10   WRITE  (unit,14) IQ[N], JQ[N], KQ[N], LQ[N], ICQ[N]
```

Proper (trigonometric) dihedral angle interaction type block
Blockname: TORSDIHEDRALTYPE

```
     WRITE  (unit,14) NPTY
     DO 10  N=1, NPTY
10   WRITE  (unit,20) CP[N], PDL[N], NP[N]
20   FORMAT (2F10.5,I5)
```

The GROMOS96 DIHEDRALTYPE block is still accepted, it expects cosine values for the phase shifts allowing only values of -1 and 1 (0 or 180°). If both blocks are specified, only the TORSDIHEDRALTYPE block is read in.

Solute dihedrals involving H-atoms block
Blockname: DIHEDRALH

```
     WRITE  (unit,14) NPHIH
     DO 10  N=1, NPHIH
10   WRITE  (unit,14) IPH[N], JPH[N], KPH[N], LPH[N], ICPH[N]
```

Solute dihedrals NOT involving H-atoms block
Blockname: DIHEDRAL

```
     WRITE  (unit,14) NPHI
     DO 10  N=1, NPHI
10   WRITE  (unit,14) IP[N], JP[N], KP[N], LP[N], ICP[N]
```

Solute cross-dihedrals involving H-atoms block
Blockname: CROSSDIHEDRALH

```
     WRITE  (unit,14) NPPCH
     DO 10  N=1, NPPCH
10   WRITE  (unit,14) APH[N], BPH[N], CPH[N], DPH[N], EPH[N],
                      FPH[N], GPH[N], HPH[N], ICCH[N]
```

Solute cross-dihedrals NOT involving H-atoms block
Blockname: CROSSDIHEDRAL

```
     WRITE  (unit,14) NPPC
     DO 10  N=1, NPPC
10   WRITE  (unit,14) AP[N], BP[N], CP[N], DP[N], EP[N],
                      FP[N], GP[N], HP[N], ICC[N]
```

Van der Waals (Lennard-Jones) interaction block
Blockname: LJPARAMETERS

```
     NRATT2 = NRATT*(NRATT+1)/2
     WRITE  (unit,14) NRATT2
     DO 10  J=1, NRATT
     DO 9   I=1, J
 9   WRITE  (unit,16) I, J, C12[I,J], C6[I,J], CS12[I,J], CS6[I,J]
10   CONTINUE
16   FORMAT (2I5,4E15.7)
```

Coarse grain (Lennard-Jones) interaction block
Blockname: CGPARAMETERS

```
     NRATT2 = NRATT*(NRATT+1)/2
     WRITE  (unit,14) NRATT2
     DO 10  J=1, NRATT
     DO 9   I=1, J
 9   WRITE  (unit,16) I, J, C12[I,J], C6[I,J]
10   CONTINUE
16   FORMAT (2I5,2E15.7)
```

Solute polarisation specification block (md++ only, optional)
Blockname: SOLUTEPOLARISATION

```
     WRITE  (unit,14) NPPOL
     DO 10  N=1, NPPOL
10   WRITE  (unit,99) IPOLP[N], ALPP[N], QPOLP[N], ENOTP[N], EPP[N]
99   FORMAT (I5,4F15.7)
```

Separate solute molecules specification block
Blockname: SOLUTEMOLECULES

```
     WRITE  (unit,14) NSPM
     DO 10  N=1, NSPM
10   WRITE  (unit,14) NSP[N]
```

Temperature atom groups specification block
Blockname: TEMPERATUREGROUPS

```
     WRITE  (unit,14) NSTM
     DO 10  N=1, NSTM
10   WRITE  (unit,14) NST[N]
```

Pressure groups specification block
Blockname: PRESSUREGROUPS

```
     WRITE  (unit,14) NSVM
     DO 10  N=1, NSVM
10   WRITE  (unit,14) NSV[N]
```

Solvent atom information block
Blockname: SOLVENTATOM

```
     WRITE  (unit,14) NRAM
     DO 10  I=1, NRAM
10   WRITE  (unit,21) I, ANMS[I], IACS[I], WMASS[I], CGS[I]
21   FORMAT (I5,1X,A5,I4,2F11.5)
```

Solvent distance constraint block
Blockname: SOLVENTCONSTR

```
     WRITE  (unit,14) NCONS
```

```
     DO 10  K=1, NCONS
10   WRITE  (unit,22) ICONS[K], JCONS[K], CONS[K]
22   FORMAT (2I5,F15.7)
```

Solvent polarisation specification block (optional)
Blockname: SOLVENTPOLARISATION

```
     WRITE  (unit,14) NVPOL
     DO 10  N=1, NVPOL
10   WRITE  (unit,99) IPOLV[N], ALPV[N], QPOLV[N], ENOTV[N], EPV[N]
```

SASA implicit solvent model parameter block
Blockname: SASAPARAMETERS

```
     WRITE  (unit,20) NRSASAA
     DO 10  I=1, NRSASAA
10   WRITE  (unit,21) ISASA[I], RADI[I], PI[I], SIGMAI[I]
20   FORMAT (I5)
21   FORMAT (I6,3X,F5.3,3X,F5.3,3X,F8.3)
```

Examples of molecular topology files are named:

*.top

3.3. Perturbation molecular topology

When simulating a molecular system or when analyzing a set of conformations of a molecule, the molecular topology file of the system remains unchanged. This is the rationale for separating topological and force field information resident in a molecular topology file from conformational information resident in configurational files. If a change of topological data or force field parameters is required, a new changed molecular topology file has to be generated by one of the molecular topology building or conversion programs.

However, when applying the thermodynamic integration formalism based on the coupling parameter (λ) approach in order to determine the difference in free energy between two states A and B of a molecular system, the molecular topology (Hamiltonian) of the system becomes a function of the coupling parameter λ such that it may change in a continuous way from the one corresponding to state A to the one corresponding to state B or vice versa. In general the difference between state B and state A is limited to a restricted part of the system, that is, a few tens of atoms. Therefore, this difference is represented by a *perturbation molecular topology*, which contains information on how to change or perturb the molecular topology of state A in order to obtain the one of state B.

The implementation of the parametrisation of the Hamiltonian of a molecular system in terms of a parameter λ has been described in Sec. 2-14.2. It has been implemented in MD++. Note that the GROMOS++ program *pt_top* can be used to merge a topology (A) and a perturbation topology (B-A) into a new topology (B). Similarly, GROMOS++ program *make_pt_top* can be used to create the perturbation topology (B-A) from the specified topologies (A) and (B). The contents of the file containing the perturbation Hamiltonian or molecular topology is described below. Here, a few comments are given:

1. The *molecular topology* that is read, the unperturbed one, corresponds to *state A.*
2. The value $\lambda = RLAM = 0$ corresponds to *state A* of the system (*unperturbed* molecular topology); the value $\lambda = RLAM = 1$ corresponds to *state B* of the system (*perturbed* molecular topology).
3. Since atoms cannot be created or destroyed, only their interaction with other atoms can be modified or perturbed. Thus, the unperturbed topology corresponding to *state A must contain all atoms* involved in the perturbation *as either real or dummy* (i.e. non-interacting) *atoms.* So, state B has the same number of atoms as state A.

4. The perturbation of non-bonded interaction is specified by giving the NJLA atom sequence number of the perturbed atoms (JLA) and the integer atom codes (IAC(A), IAC(B)), masses (MASS(A), MASS(B) and charges (CHARGE(A), CHARGE(B)) in both states A and B. The *force field parameters for state A* given in the perturbation molecular topology must not necessarily match those given in the (unperturbed) molecular topology. In the perturbation calculation, interactions in state A will be described according to state A given in the perturbation topology and a warning will be printed if state A in the perturbed topology does not match state A in the unperturbed topology.
5. The change from state A to state B may involve the *breaking or formation of a covalent bond between two atoms*. In that case, the excluded neighbours and the third neighbours of these atoms will be different in state A and in state B. The type of interaction, i.e. normal interaction, 1-4 or third-neighbour interaction, must be changeable. The standard non-bonded interaction subroutines only allow for a continuous change from one integer atom code (IAC(A)) to another (IAC(B)), but not for a change of type in the sense of normal, third-neighbour or excluded-neighbour interaction. A change of type is implemented by specifying the NEB pairs of atoms (IEB, JEB) for which the type is to be changed when moving from state A to state B. The perturbation molecular topology file contains the variables IETA and IETB for each pair, denoting which of the three types of interaction is applicable in state A and in state B. The interaction for these specified atom pairs is evaluated in special subroutines in MD++. In order to avoid double counting, *all these specified pairs must be excluded atom pairs in the unperturbed molecular topology* (state A).
6. In the most common case, all interactions within the molecular system are made λ-dependent. However, in special cases, one may wish to restrict the λ-dependence to a specified subset of interactions. This can be done by defining individual λ values per interaction which can be different for interactions within or between every energy group. The individual λ values are defined as a polynomial function of order 4 of the overall λ value (see Sec. 2-14.4).
7. In a number of applications (e.g. creation or annihilation of atoms by conversion from or into a dummy, or free-energy extrapolation from an unphysical reference state) it is useful to make perturbed interactions soft. This is achieved by a modification of the λ-dependent Lennard-Jones and electrostatic interaction functions through the introduction of two corresponding soft-core parameters $\alpha_{LJ}(I,J)$ and $\alpha_{EL}(I,J)$. These parameters are calculated from atomic soft-core parameters $\alpha_{LJ}(I,I)$ and $\alpha_{EL}(I,I)$ using the combination rules described in Sec. 2-14.2.8.
8. The perturbation of the bond-stretching, bond-angle bending, improper dihedral or dihedral interaction terms is specified by giving the sequence numbers of the atoms involved and the type codes determining force field parameters in state A as well as in state B. The *force field parameters for state A* given in the perturbation molecular topology must not necessarily match those given in the (unperturbed) molecular topology. In the perturbation calculation, interactions in state A will be described according to state A given in perturbation topology and a warning will be printed if state A in the perturbed topology does not mach the unperturbed topology. The occurrence of multiple force field terms involving the same atoms and the same type code in the unperturbed topology is not allowed in this case.
9. For bond stretching, bond-angle bending and improper dihedrals a soft potential energy function can be chosen to reduce numerical instabilities when force constants are being reduced to 0. Apart from the regular type codes, determining the force field parameters for states A and B, a type code of 0 can be given for either state A or state B, indicating an interaction with a force constant of 0. An additional softness parameter (α_b, α_θ or α_ξ, respectively) is added to the definition of the perturbed interaction (see Sec. 2-14.2.2).
10. Note that the *units* of the perturbation molecular topology file *must match* the units of the unperturbed molecular topology file.
11. For some GROMOS++ programs and for enveloping distribution sampling (EDS) in MD++, it is convenient to handle multiple perturbation topologies simultaneously. For the non-bonded interactions this is implemented in a special MPERTATOMS block in which only the interaction parameters IACB and charges CGB for state B are specified for different perturbations. The GROMOS++ program *pt_top* can convert a *multiple perturbation topology file* into a perturbation topology file.

A perturbation molecular topology file is characterized by the following quantities:

`NJLA` number of perturbed atoms

NPTB	number of listed perturbations in MPERTATOM block
PTNAME[1..NPTB]	name to identify a perturbation in MPERTATOM block
NR[1..NJLA]	atom sequence numbers of the perturbed atoms ($\leqslant$ NRP)
RES[1..NJLA]	residue sequence number of atom (only read, not used)
NAME[1..NJLA]	atom name of atom (only read, not used)
IAC(A)[1..NJLA]	integer atom code of perturbed atoms in state A, determining the type of van der Waals interaction($\leqslant$ NRATT)
IAC(B)[1..NPTB,1..NJLA]	integer atom code of perturbed atoms in state B, determining the type of van der Waals interaction ($\leqslant$NRATT)
MASS(A)[1..NJLA]	mass of the perturbed atoms in state A
MASS(B)[1..NJLA]	mass of the perturbed atoms in state B
CHARGE(A)[1..NJLA]	charge of the perturbed atoms in state A
CHARGE(B)[1..NPTB,1..NJLA]	charge of the perturbed atoms in state B
ALJ[1..NJLA]	atomic soft-core parameter for the Lennard-Jones interaction
ACRF[1..NJLA]	atomic soft-core parameters for the Coulomb-Reaction field interaction
NEB	number of atom pairs for which the non-bonded interaction changes the exclusion state; these pairs must be excluded pairs in the molecular topology file
IEB, JEB[1..NEB]	atom sequence numbers of the two atoms forming the pairs ($\leqslant$NRP)
IETA[1..NEB]	determines type of non-bonded interaction in state A for the pairs; zero for no non-bonded interaction; one for non-bonded interaction of normal type; two for non-bonded interaction of 1-4 (van der Waals) type
IETB[1..NEB]	likewise, but for state B
NBONHG	number of perturbed bonds involving H-atoms ($\leqslant$NBONH)
IBHG, JBHG[1..NBONHG]	atom sequence numbers of the atoms forming the perturbed bond i-j ($\leqslant$NRP)
ICBHA[1..NBONHG]	bond-type code corresponding to bond-stretching interaction term in state A ($\leqslant$NBTY)
ICBHB[1..NBONHG]	as ICBHA, but for state B ($\leqslant$NBTY)
NBONG	number of perturbed bonds NOT involving H-atoms ($\leqslant$NBON)
IBG, JBG[1..NBONG]	atom sequence numbers of the atoms forming the perturbed bonds i-j ($\leqslant$NRP)
ICBA[1..NBONG]	bond-type code corresponding to bond-stretching interaction term in state A ($\leqslant$NBTY)
ICBB[1..NBONG]	as ICBA, but for state B ($\leqslant$NBTY)
NBONSG	number of perturbed bonds with a soft potential energy function ($\leqslant$NBON)

IBSG, JBSG[1..NBONSG]
: atom sequence numbers of the atoms forming the perturbed bonds i-j (⩽NRP)

ICBSA[1..NBONSG]
: bond-type code corresponding to bond-stretching interaction term in state A (⩽NBTY)

ICBSB[1..NBONSG]
: as ICBSA, but for state B (⩽NBTY)

ALB[1..NBONSG]
: softness parameter for soft harmonic bond

NTHEHG
: number of perturbed bond angles involving H-atoms (⩽NTHEH)

ITHG, JTHG, KTHG[1..NTHEHG]
: atom sequence numbers of the atoms forming the perturbed bond angle i-j-k (⩽NRP)

ICTHA[1..NTHEHG]
: bond-angle type code corresponding to bond-angle bending interaction term in state A (⩽NTTY)

ICTHB[1..NTHEHG]
: as ICTHA, but for state B (⩽NTTY)

NTHEG
: number of perturbed bond angles NOT involving H-atoms (⩽NTHE)

ITG, JTG, KTG[1..NTHEG]
: atom sequence numbers of the atoms forming the perturbed bond angle i-j-k (⩽NRP)

ICTA[1..NTHEG]
: bond-angle type code corresponding to bond-angle bending interaction term in state A (⩽NTTY)

ICTB[1..NTHEG]
: as ICTA, but for state B (⩽NTTY)

NTHESG
: number of perturbed bond angles with a soft potential energy function (⩽NTHE)

ITSG, JTSG, KTSG[1..NTHESG]
: atom sequence numbers of the atoms forming the perturbed bond angle i-j-k (⩽NRP)

ICTSA[1..NTHESG]
: bond-angle type code corresponding to bond-angle bending interaction term in state A (⩽NTTY)

ICTSB[1..NTHESG]
: as ICTSA, but for state B (⩽NTTY)

ALA[1..NTHESG]
: softness parameter for soft bond angle

NQHIHG
: number of perturbed improper (harmonic) dihedrals involving H-atoms (⩽NQHIH)

IQHG, JQHG, KQHG, LQHG[1..NQHIHG]
: atom sequence numbers of the atoms forming the perturbed improper (harmonic) dihedral i-j-k-l (⩽NRP)

ICQHA[1..NQHIHG]
: improper dihedral type code corresponding to improper-dihedral interaction term in state A (⩽NQTY)

ICQHB[1..NQHIHG]
: as ICQHA, but for state B (⩽NQTY)

NQHIG
: number of perturbed improper (harmonic) dihedrals NOT involving H-atoms (⩽NQHI)

IQG, JQG, KQG, LQG[1..NQHIG]
: atom sequence numbers of the atoms forming the perturbed improper (harmonic) dihedral i-j-k-l (⩽NRP)

ICQA[1..NQHIG]
: improper-dihedral type code corresponding to improper-dihedral interaction term in state A (⩽NQTY)

ICQB[1..NQHIG]	as ICQA, but for state B (⩽NQTY)
NQHISG	number of perturbed improper (harmonic) dihedrals with a soft potential energy function (⩽NQHI)
IQSG, JQSG, KQSG, LQSG[1..NQHISG]	atom sequence numbers of the atoms forming the perturbed improper (harmonic) dihedral i-j-k-l (⩽NRP)
ICQSA[1..NQHISG]	improper-dihedral type code corresponding to improper-dihedral interaction term in state A (⩽NQTY)
ICQSB[1..NQHISG]	as ICQSA, but for state B (⩽NQTY)
ALI[1..NQHISG]	softness parameter for improper dihedral
NPHIHG	number of perturbed (trigonometric) dihedrals involving H-atoms (⩽NPHIH)
IPHG, JPHG, KPHG, LPHG[1..NPHIHG]	atom sequence numbers of the atoms forming the perturbed (trigonometric) dihedral i-j-k-l (⩽NRP)
ICPHA[1..NPHIHG]	dihedral-type code corresponding to trigonometric dihedral interaction term in state A (⩽NPTY)
ICPHB[1..NPHIHG]	as ICPHA, but for state B (⩽NPTY)
NPHIG	number of perturbed (trigonometric) dihedrals NOT involving H-atoms (⩽NPHI)
IPG, JPG, KPG,LPG[1..NPHIG]	atom sequence numbers of the atoms forming the perturbed (trigonometric) dihedral i-j-k-l (⩽NRP)
ICPA[1..NPHIG]	dihedral-type code corresponding to trigonometric dihedral interaction term in state A (⩽NPTY)
ICPB[1..NPHIG]	as ICPA, but for state B (⩽NPTY)
NPOLG	number of perturbed polarisabilities of perturbed atoms with atom sequence number JLA (NPOLG⩽NJLA)
ALPA[1..NPOLG]	polarisability of perturbed atoms in state A
ENOTA[1..NPOLG]	damping level for polarisation of perturbed atoms in state B
ALPB[1..NPOLG]	polarisability of perturbed atoms in state A
ENOTB[1..NPOLG]	damping level for polarisation of perturbed atoms in state B

The blocks of a *perturbation molecular topology file* are (apart from the *Title block*) the following:

Perturbed atom information block
Blockname: PERTATOMPARAM

```
     WRITE  (unit,14) NJLA
     DO 10  N=1, NJLA
10   WRITE  (unit,23) NR[N],RES[N],NAME[N],IAC(A)[N],MASS(A)[N],CHARGE(A)[N],
```

```
                    IAC(B)[N],MASS(B)[N],CHARGE(B)[N],ALJ[N],ACRF[N]
23   FORMAT (2I5,1X,A5,2F11.5,I4,4F11.5)
```

Multiple perturbed atom information block (for use in GROMOS++ only)
Blockname: MPERTATOM

```
     WRITE  (unit,14) NJLA, NPTB
     WRITE  (unit 21) (PTNAME[I],I=1,NPTB)
     DO 10  N=1, NJLA
10   WRITE  (unit,22) NR[N],NAME[N],((IAC(B)[I,N],CHARGE(B)[I,N]),I=1,NPTB),ALJ[N],ACRF[N]
21   FORMAT (16A5)
22   FORMAT (I5,1X,A5,16(I4,F11.5))
```

Perturbed NONBPL atom pair block
Blockname: PERTATOMPAIR

```
     WRITE  (unit,14) NEB
     DO 10  N=1, NEB
10   WRITE  (unit,14) IEB[N], JEB[N], IETA[N], IETB[N]
```

Perturbed bonds involving H-atoms block
Blockname: PERTBONDSTRETCHH

```
     WRITE  (unit,14) NBONHG
     DO 10  N=1, NBONHG
10   WRITE  (unit,14) IBHG[N], JBHG[N], ICBHA[N], ICBHB[N]
```

Perturbed bonds NOT involving H-atoms block
Blockname: PERTBONDSTRETCH

```
     WRITE  (unit,14) NBONG
     DO 10  N=1, NBONG
10   WRITE  (unit,14) IBG[N], JBG[N], ICBA[N],ICBB[N]
```

Perturbed bonds with a soft potential
Blockname: PERTBONDSOFT

```
     WRITE  (unit,14) NBONSG
     DO 10  N=1, NBONSG
10   WRITE  (unit,14) IBSG[N], JBSG[N], ICBSA[N],ICBSB[N], ALB[N]
```

Perturbed bond angles involving H-atoms block
Blockname: PERTBONDANGLEH

```
     WRITE  (unit,14) NTHEHG
     DO 10  N=1, NTHEHG
10   WRITE  (unit,14) ITHG[N], JTHG[N], KTHG[N], ICTHA[N], ICTHB[N]
```

Perturbed bond angles NOT involving H-atoms block
Blockname: PERTBONDANGLE

```
     WRITE  (unit,14) NTHEG
     DO 10  N=1, NTHEG
10   WRITE  (unit,14) ITG[N], JTG[N], KTG[N], ICTA[N], ICTB[N]
```

Perturbed bond angles with a soft potential
Blockname: PERTANGLESOFT

```
     WRITE  (unit,14) NTHESG
     DO 10  N=1, NTHESG
10   WRITE  (unit,14) ITSG[N], JTSG[N], KTSG[N], ICTSA[N], ICTSB[N], ALA[N]
```

Perturbed improper (harmonic) dihedrals involving H-atoms block
Blockname: PERTIMPROPERDIHH

```
     WRITE  (unit,14) NQHIHG
     DO 10  N=1, NQHIHG
10   WRITE  (unit,26) IQHG[N], JQHG[N], KQHG[N], LQHG[N], ICQHA[N],
                      ICQHB[N]
26   FORMAT (6I5)
```

Perturbed improper (harmonic) dihedrals NOT involving H-atoms block
Blockname: PERTIMPROPERDIH

```
     WRITE  (unit,14) NQHIG
     DO 10  N=1, NQHIG
10   WRITE  (unit,26) IQG[N], JQG[N], KQG[N], LQG[N], ICQA[N], ICQB[N]
```

Perturbed improper (harmonic) dihedrals with a soft potential
Blockname: PERTIMPROPERDIHSOFT

```
     WRITE  (unit,14) NQHISG
     DO 10  N=1, NQHISG
10   WRITE  (unit,26) IQG[N], JQG[N], KQG[N], LQG[N], ICQA[N], ICQB[N], ALI[N]
```

Perturbed (trigonometric) dihedrals involving H-atoms block
Blockname: PERTPROPERDIHH

```
     WRITE  (unit,14) NPHIHG
     DO 10  N=1, NPHIHG
10   WRITE  (unit,26) IPHG[N], JPHG[N], KPHG[N], LPHG[N], ICPHA[N],
                      ICPHB[N]
```

Perturbed (trigonometric) dihedrals NOT involving H-atoms block
Blockname: PERTPROPERDIH

```
     WRITE  (unit,14) NPHIG
     DO 10  N=1, NPHIG
10   WRITE  (unit,26) IPG[N], JPG[N], KPG[N], LPG[N], ICPA[N], ICPB[N]
```

Perturbed atomic polarisabilities block
Blockname: PERTPOLPARAM

```
     WRITE  (unit,14) NPOLG
     DO 10  N=1, NPOLG
10   WRITE  (unit,99) JLA[N], RESNR[N], ATNAME[N], ALPA[N], ENOTA[N],
                      ALPB[N], ENOTB[N]
```

Examples of perturbation molecular topology files are named:

*.ptp

3.4. Atom-atom and distance-field distance restraints

When performing a simulation or energy minimization, a special interaction function term that restrains atom-atom distances can be added to the interaction function, see Sec. 2-9.3 and Sec. 2-9.12. Such a term may be used to make a molecule satisfy a given set of atom-atom distance upper or lower bounds, or to direct a molecule into the active site of a protein. A slight complication is that an atom involved in an atom-atom distance restraint pair may be a virtual or a pseudo atom (Sec. 2-9.4). In terms of a molecular topology or a molecular configuration such an atom is non-existing. As discussed in Sec. 2-9.4, its geometric position is defined in terms of the positions of its non-virtual neighbour atoms. For a virtual or pseudo atom the atom-atom distance restraint specification will contain the atom sequence numbers of the real atoms that define the virtual or pseudo atom position together with a geometry code denoting the specific geometric definition.

A set of atom-atom distance restraints in an atom-atom distance restraints file is characterized by the following quantities:

`NDR` number of distance restraint atom pairs per "solute" molecule

`I1, J1, K1, L1[1..NDR]`
atom sequence numbers of the real atoms defining the geometric position of the first atom of a distance restraint pair ($\leqslant$NRP)

`TYPE1[1..NDR]` geometric code defining the position of the first atom of a distance restraint pair [-2, -1, ..., 7]

`I2, J2, K2, L2[1..NDR]`
atom sequence numbers of the real atoms defining the geometric position of the second atom of a distance restraint pair ($\leqslant$NRP)

`TYPE2[1..NDR]` geometric code defining the position of the second atom of a distance restraint pair [-2, -1, ..., 7]

`R0[1..NDR]` in case of a full-harmonic distance restraint (RAH = 0), R0 is the minimum-energy distance; in case of an attractive or repulsive half-harmonic restraint (RAH = $\pm$1), R0 is the upper or lower bound, respectively, beyond which the restraining forces become non-zero. When using distance restraints for NMR-NOE distance restraining, pseudo-atom corrections should already be included in R0 (see Sec. 2-9.4)

`W0[1..NDR]` individual distance restraint weight factor, by which the distance restraint interaction term may be multiplied.

`DIM` dimensionality-code for distance restraints. See below for allowed options. The value of DIM is determined from the value of RAH and is not stored separately.

`RAH[1..NDR]` type of distance restraint; this parameter sets both the dimensions in which the restraint is applied as well as the shape of the functional form. if RAH = DIM - 1, a half-harmonic repulsive distance restraint is applied; if RAH = DIM, a full harmonic distance restraint is applied; if RAH = DIM + 1, a half-harmonic attractive distance restraint is applied.

DISH	carbon-hydrogen distance, used for geometries TYPE = 1-6
DISC	carbon-carbon distance, used for geometry TYPE = 6

In MD++ a distance restraint can also be modified in the course of a free energy perturbation. The perturbed distance restraints make use of the additional parameters

NDRP	number of perturbed distance restraint atom pairs
M[1..NDRP]	hidden restraint parameter: exponent of λ in state superposition prefactor
N[1..NDRP]	hidden restraint parameter: exponent of (1-λ) in state superposition prefactor
A_R0[1..NDRP]	upper or lower bound beyond which the restraining forces become non-zero for state A
A_W0[1..NDRP]	individual distance restraint weight factor by which the distance restraint interaction term may be multiplied for state A
B_R0[1..NDRP]	as A_R0, but for state B
B_W0[1..NDRP]	as A_W0, but for state B

As discussed in Sec. 2-9.4, the allowed geometries are the following ones. The notation is given in terms of hydrogen atoms.

TYPE = 0	real atom; its atom sequence number is given by IDR
TYPE = 1	virtual H-atom, aliphatic CH; it is bound to real atom I (carbon, atom sequence number IDR) and the three covalently-bound real neighbours of atom I are the real atoms J, K and L (atom sequence numbers JDR, KDR and LDR)
TYPE = 2	virtual H-atom, aromatic CH; it is bound to real atom I (carbon) and the two covalently-bound real neighbours of atom I are the real atoms J and K (LDR is not used)
TYPE = 3	pseudo H-atom, geometric mean of the two H-atoms of an aliphatic CH_2; it is (pseudo) bound to real atom I (carbon) and the two covalently-bound real neighbours of atom I are the real atoms J and K (LDR is not used)
TYPE = 4	virtual H-atom, one of the two H-atoms of an aliphatic CH_2; it is bound to real atom I (carbon) and the two covalently-bound real neighbours of atom I are the real atoms J and K (LDR is not used); the definition is the following: looking along covalent bond vector J-I from atom J to the central (carbon) atom I, the direction of the virtual bond I-H is obtained from the direction of the bond I-K by a counter-clockwise rotation over 120° around bond J-I; the other virtual H-atom can be obtained by interchanging the sequence numbers JDR and KDR
TYPE = 5	pseudo H-atom, geometric mean of the three H-atoms of a CH_3 group; it is (pseudo) bound to real atom I (carbon) and the one covalently-bound real neighbour of atom I is the real atom J (KDR and LDR are not used)
TYPE = 6	pseudo H-atom, geometric mean of the six H-atoms of two CH_3 groups that are both bound to a common third carbon atom; it is (pseudo) bound to this real third carbon atom I and the carbon atoms of the two CH_3 groups are the real atoms J and K (LDR is not used)
TYPE = 7	pseudo H-atom, geometric mean of the nine H-atoms of three CH_3 groups that are all three bound to a common fourth carbon atom I; it is (pseudo) bound to I and the fifth atom J is the real atom that is bound to I as well (KDR and LDR not used)

`TYPE = -1` virtual atom, centre of geometry of the atoms I,J,K and L if their specifications are non-zero. (Example: the two (δ or ϵ) H-atoms I and J of an aromatic ring, or the two H-atoms I and J of a planar NH_2-group.)

`TYPE = -2` virtual atom, centre of mass of the atoms I,J,K and L if their specifications are non-zero.

Atom-atom distance restraints may be applied in selected dimensions only. This is specified by the parameter RAH, from which the nearest integer code DIM is deduced. The following values of DIM are implemented in MD++:

`DIM = 0` dimensions to apply distance restraint: X, Y, Z.

`DIM = 10` dimensions to apply distance restraint: X, Y.

`DIM = 20` dimensions to apply distance restraint: X, Z.

`DIM = 30` dimensions to apply distance restraint: Y, Z.

`DIM = 40` dimension to apply distance restraint: X.

`DIM = 50` dimension to apply distance restraint: Y.

`DIM = 60` dimension to apply distance restraint: Z.

Atom-atom distance restraint specification block
Blockname: DISTANCERESSPEC

```
     WRITE  (unit,11) DISH, DISC
     DO 10  N=1, NDR
10   WRITE  (unit,12) I1[N], J1[N], K1[N], L1[N], TYPE1[N],
                      I2[N], J2[N], K2[N], L2[N], TYPE2[N],
                      R0[N], W0[N], RAH[N]
11   FORMAT (2F10.5)
12   FORMAT (5I5,5X,5I5,3F10.5)
```

Perturbed atom-atom distance restraint specification block
Blockname:PERTDISRESSPEC

```
     WRITE  (unit,11) DISH, DISC
     DO 10  N=1, NDRP
10   WRITE  (unit,13) I1[N], J1[N], K1[N], L1[N], TYPE1[N],
                      I2[N], J2[N], K2[N], L2[N], TYPE2[N], M[N], N[N]
                      A_R0[N], A_W0[N], B_R0[N], B_W0[N], RAH[N]
13   FORMAT (5I5,5X,7I5,5F10.5)
```

For enveloping distribution sampling (EDS) it is convenient to define multiple perturbed distances. For this the MDISRESSPEC block may be used.

Multiple atom-atom distance restraint specification block
Blockname:MDISRESSPEC

```
     WRITE  (unit,11) DISH, DISC
     DO 10  N=1, NDRP
10   WRITE  (unit,13) I1[N], J1[N], K1[N], L1[N], TYPE1[N],
                      I2[N], J2[N], K2[N], L2[N], TYPE2[N],
```

```
                    (R0[N,M],M=1,NEDS), (W0[N,M],M=1,NEDS), RAH[N]
13   FORMAT (5I5,5X,5I5,NEDS(F10.5),NEDS(F10.5),I5)
```

A distance-field distance restraint in an atom-atom distance restraints file is characterized by the following additional quantities:

PROTEINATOMS	last atom of the set of atoms to be defined as being part of the protein ($\leqslant$NRP)
K	force constant for the harmonic distance-field distance restraint
R0	minimum-energy distance on the distance-field grid
TYPE_I	geometric code defining the position of the first atom of the distance-field distance restraint, typically the protein [-2, -1, ..., 7]
NUM_I	number of atoms used to define the virtual atom I
ATOM_I[1..NUM_I]	atom sequence numbers of the atoms used to define the virutal atom I
TYPE_J	geometric code defining the position of the first atom of the distance-field distance restraint, typically the ligand [-2, -1, ..., 7]
NUM_J	number of atoms used to define the virtual atom I
ATOM_J[1..NUM_J]	atom sequence numbers of the atoms used to define the virutal atom I

In MD++ a distance-field distance restraint can also be modified in the course of a free energy perturbation. The perturbed distance-field distance restraint makes use of the additional parameters

K_A	force constant for the harmonic distance-field distance restraint in state A
K_B	force constant for the harmonic distance-field distance restraint in state B
A_R0	minimum-energy distance on the distance-field grid in state A
B_R0	minimum-energy distance on the distance-field grid in state B

Distance-field distance restraint specification block
Blockname: DFRESSPEC

```
     WRITE  (unit,11) DISH, DISC
     WRITE  (unit,12) PROTEINATOMS, K, R0
     WRITE  (unit,13) TYPE_I, NUM_I, (ATOM_I[K],K=1,NUM_I)
     WRITE  (unit,13) TYPE_J, NUM_J, (ATOM_J[K],K=1,NUM_J)
11   FORMAT (2F10.5)
12   FORMAT (I5,2F10.5)
13   FORMAT (16I5)
```

Perturbed distance-field distance restraint specification block
Blockname: PERTDFRESSPEC

```
     WRITE  (unit,11) DISH, DISC
     WRITE  (unit,12) PROTEINATOMS, A_R0, K_A, B_R0, K_B, M, N
     WRITE  (unit,13) TYPE_I, NUM_I, (ATOM_I[K],K=1,NUM_I)
     WRITE  (unit,13) TYPE_J, NUM_J, (ATOM_J[K],K=1,NUM_I)
11   FORMAT (2F10.5)
12   FORMAT (I5,4F10.5, 2I5)
13   FORMAT (16I5)
```

Examples of atom-atom distance restrained files are named:

*.dsr

Program *prep_noe* can produce an atom-atom distance restrained file for virtual and pseudo atoms from a list of proton-proton distances and a library file. See also Sec. 7.8.

3.5. Dihedral-angle restraints or constraints

When performing a simulation or energy minimization, a special interaction function term that restrains dihedral angles can be added to the interaction function. Dihedral angles can also be constraint, see Chap. 2. This approach may be used to make a molecule satisfy a given set of dihedral angle values.

A set of dihedral-angle restraints or constraints in a dihedral-angle restraints file is characterized by the following quantities:

`NDLR` number of dihedral-angle restraints

`IPLR, JPLR, KPLR, LPLR [1..NDLR]`
atom sequence numbers of the atoms defining the restrained dihedral i-j-k-l ($\leqslant$NPM*NRP), j is always smaller than k

`WDLR[1..NDLR]` individual dihedral restraint weight factor by which the harmonic dihedral restraining term may be multiplied.

`PDLR[1..NDLR]` dihedral angle value (in degrees) at minimum energy of the harmonic dihedral restraining term; upon reading a dihedral angle restraints file, the dihedral angle is converted from degrees to radians and stored in PDLR

`DELTA[1..NDLR]` dihedral angle value (in degrees) defining the periodic dihedral angle interval. The current dihedral angle value is shifted to the interval [PDLR[I] + DELTA[I] - 360.0 ,PDLR[I] + DELTA[I]] before force calculation

In MD++ a dihedral angle restraint can also be modified in the course of a free energy perturbation. The perturbed dihedral angle restraints make use of the additional parameters

`NDLRP` number of perturbated dihedral-angle restraints

`MLR[1..NDLRP]` hidden restraint parameter: exponent of λ in state superposition prefactor

`NLR[1..NDLRP]` hidden restraint parameter: exponent of (1-λ) in state superposition prefactor

`APDLR[1..NDLRP]` dihedral angle value (in degrees) at minimum energy of the harmonic dihedral restraining term in state A

`AWDLR[1..NDLRP]` Individual dihedral restraint weight factor by which the harmonic dihedral restraining term may be multiplied in state A

`BPDLR[1..NDLRP]` as APDLR, but for state B

`BWDLR[1..NDLRP]` as AWDLR, but for state B

Dihedral angle restraint specification block
Blockname: DIHEDRALRESSPEC

```
     DO 10  N=1, NDLR
10   WRITE  (unit,11) IPLR[N], JPLR[N], KPLR[N], LPLR[N], WDLR[N],
                      PDLR[N], DELTA[N]
11   FORMAT (4I5,2F15.7)
```

Perturbed dihedral angle restraint specification block
Blockname: PERTDIHRESSPEC

```
     DO 10  N=1, NDLRP
10   WRITE  (unit,11) IPLR[N], JPLR[N], KPLR[N], LPLR[N], MLR[N], NLR[N], DELTA[N],
                      APDLR[N], AWDLR[N], BPDLR[N], BWDLR[N]
11   FORMAT (4I5,5F15.7)
```

Examples of dihedral angle restraint files are named:

*.dhr

3.6. ^3J-coupling constant restraints

When performing a simulation or energy minimization, a special interaction function term that restrains NMR ^3J-coupling constants can be added to the interaction function, see Sec. 2-9.7. Such a term may be used to make a molecule satisfy a given set of ^3J-values.

A set of ^3J-coupling constant restraints in a ^3J-coupling constant restraints file is characterized by the following quantities:

`NDJV` number of ^3J-coupling constant restraints.

`IPJV, JPJV, KPJV, LPJV [1..NDJV]`
atom sequence numbers of the real atoms present in the simulation that define the dihedral angle related to the restrained ^3J-value ($\leqslant$NRP).

`WJVR[1..NDJV]` individual ^3J-value restraint weight factor by which the restraining term for each ^3J-value may be multiplied.

`PJR0[1..NDJV]` experimental or reference ^3J-value, J_0 (≥ 0). In the case of a full-harmonic ^3J-value restraint (NHJV = 0), PJR0 is the minimum-energy ^3J-value; in the case of an attractive or repulsive half-harmonic ^3J-value restraint (NHJV = $\pm$1), it is the upper or lower bound, respectively, beyond which the restraining force becomes non-zero.

`PSJR[1..NDJV]` phase shift or difference $\delta = \theta - \phi$ between the dihedral angle θ formed by the possibly non-existant atoms defining the experimental ^3J-coupling and the dihedral angle $\phi(i-j-k-l)$ formed by the real atoms present in the simulation (in degrees); upon reading a ^3J-coupling constant restraints file, the phase shift is converted from degrees to radians and stored in PSJR.

`AJV, BJV, CJV[1..NDJV]`
Karplus parameters a, b and c for the ^3J-coupling constant expressed as function of the dihedral angle θ

`NHJV[1..NDJV]` the type of ^3J-value restraint; if H = -1, a half-harmonic repulsive ^3J-value restraint is applied; if H = 0, a full harmonic ^3J-value restraint is applied; if H = 1, a half-harmonic attractive ^3J-value is applied. Note that the half-harmonic forms of the potential are only implemented in analogy to distance restraining and make little sense for restraining ^3J-values, which depend on a periodic structural parameter.

^3J-coupling constant restraint specification block
Blockname: JVALRESSPEC

```
     DO 10  N=1, NDJV
10   WRITE  (unit,11) IPJV[N], JPJV[N], KPJV[N], LPJV[N], WJVR[N],
                      PJR0[N], PSJR [N], AJV[N], BJV[N], CJV[N], NHJV[N]
11   FORMAT (4I5,7F10.5)
```

Examples of ^3J-coupling constant restraint files are named:

*.jvr

3.7. S^2-order parameter restraining

When performing a simulation or energy minimization, a special interaction function term that restrains NMR S^2-order parameters can be added to the interaction function, see Sec. 2-9.8. Such a term may be used to make a molecule satisfy a given set of S^2-values.

A set of S^2-order parameter restraints in a S^2-order parameter restraints file is characterized by the following quantities:

`NOPR` number of S^2-order parameter restraints.

`I1, J1, K1, L1 [1..NOPR]`
atom sequence numbers of the real atoms defining the geometric position of the first atom of the order parameter restraint pair ($\leqslant$NRP).

`TYPE1 [1..NOPR]` geometric code defining the position of the first atom of a order parameter restraint pair [-2, -1, ..., 7]

`I2, J2, K2, L2 [1..NOPR]`
atom sequence numbers of the real atoms defining the geometric position of the second atom of the order parameter restraint pair ($\leqslant$NRP).

`TYPE2 [1..NOPR]` geometric code defining the position of the second atom of a order parameter restraint pair [-2, -1, ..., 7]

`RN [1..NOPR]` effective distance used to make the order parameter dimensionless.

`S0 [1..NOPR]` experimental or reference S^2-value

`DS0 [1..NOPR]` size of flat bottom region in one direction

`WOPR [1..NOPR]` individual order parameter restraint weight factor, by which the order parameter restraint term may be multiplied

`DISH` carbon-hydrogen distance, used for geometries ICOPR = 1-6

`DISC` carbon-carbon distance, used for geometry ICOPR = 6

S^2-order parameter restraint specification block
Blockname: ORDERPARAMRESSPEC

```
     WRITE  (unit,11) DISH, DISC
     DO 10  N=1, NOPR
10   WRITE  (unit,12) IOPR1[N], JOPR1[N], KOPR1[N], LOPR1[N], ICOPR1[N],
                      IOPR2[N], JOPR2[N], KOPR2[N], LOPR2[N], ICOPR2[N],
                      RN[N], S0[N], DS0[N], WOPR[N]
11   FORMAT (2F10.5)
12   FORMAT (5I5,5X,5I5,3F10.5)
```

Examples of S^2-order parameter restraint files are named:

*.opr

Symmetry restraining block If the symmetry within a unit cell is to be restrained additional parameters, a description of the asymmetric unit and the atoms to be restrained have to be given.

`NTSYM` A switch to determine the method of symmetry restraining. 0: no symmetry restraining, 1: harmonic symmetry restraining, 2: symmetry constraining

`CSYM` The force constant for the symmetry restraints

`SYMSPGR` The space group in Hermann-Mauguin format

`SYMNUMSYM` The number of asymmetric units combined to form the unit cell

`ASUDEF[1..SYMNUMSYM]`
The index of the first atom of each asymmetric unit

`NSYMATOMS` The number of atoms to be symmetry restrained

`SYMATOMS[1..NSYMATOMS]`
The index of the atom to be symmetry restrained. Only the atoms in the first asymmetric unit are to be specified. The indices of the atoms of the other asymmetric units are determined automatically.

Blockname: XRAYSYMRESSPEC

```
8    WRITE  (unit,9)  NTSYM, CSYM
9    FORMAT (I5,F10.5)
10   WRITE  (unit,11) SYMSPGR
11   FORMAT (A20)
12   WRITE  (unit,13) ASUDEF[N], N = 1, SYMNUMSYM
13   FORMAT (100I5)
     DO 10  N=1, NSYMATOMS
14   WRITE  (unit,15) SYMATOMS[N]
15   FORMAT (17X, I5)
```

B-factor optimisation blocks
For the optimisation of atomic B-factors additional parameters can be specified. In addition, groups of equal atoms can be specified.

`BFOPTS` Optimise B-factos every BFOPTSth step

`BFOPTTI` Terminate after BFOPTTI minimisation iterations

`BFOPTTG` Terminate if the gradient is smaller than BFOPTTG

`BFOPTMN` The minimal B-factor

`BFOPTMX` The maximal B-factor

`BFOPTNG` The number of B-factor groups

`BFOPTGS[1..BFOPTNG]`
The size of a B-factor group

`BFOPTGM[1..BFOPTNG][1..BFOPTGS]`
The index of the atom being member in this group

Blockname: XRAYBFACTOROPTIMISATION

```
1    WRITE  (unit, 2) BFOPTS, BFOPTTI, BFOPTTG, BFOPTMN, BFOPTMX
2    FORMAT (2I5, 3F10.5)
```

```
3    WRITE  (unit, 4) BFOPTNG
4    FORMAT (I5)
     DO 5   N=1, BFOPTNG
5    WRITE  (unit, 6) BFOPTGS[N], BFOPTGM[N][K], K=1, BFOPTGS[N]
6    FORMAT (1001I5)
```

Structure factor computation As the computation of structure factors is computationally demanding, it can be either carried out every selected step or whenever an atom has moved by some distance.

SFCTOL	The distance an atom is allowed to move before the structure factors are recalculated
SFCST	recalculate the structure factors every SFCST steps

Blockname: XRAYSFCALC

```
1    WRITE  (unit, 2) SFCTOL, SFCST
2    FORMAT (2I5)
```

Replica exchange properties block This block is used to couple the crystallographic restraints with lambda for Hamiltonian replica-exchange simulations.

NTXRRE	Determines the coupling method to be used. 0: do not couple the X-ray restraints with lambda, 1: couple the force constant, 2: couple the resolution
CXREEMN	The RLAM= 0 value of CXR or RESO
CXREEMX	The RLAM= 1 value of CXR or RESO

Blockname: XRAYREPLICAEXCHANGE

```
1    WRITE  (unit, 2) NTXRRE, CXREEMN, CXREEMX
2    FORMAT (I5, 2F10.5)
```

Examples of crystallographic restraint files are named:

*.xrs

3.8. Local-elevation coordinates

When performing a simulation, a special (changing) interaction function term that memorizes the values adopted during the simulation by a specified set of so-called local-elevation (LE) coordinates and increasingly penalizes readopting of these values, can be added to the interaction function, see Chap. 2. Such a local-elevation term may be used to drive a molecule out of a low energy conformation (LE searching).

The biasing potential energy term may also be frozen at some point and used to perform umbrella sampling (LEUS).

A set of local-elevation coordinates in a local-elevation coordinates file is characterized by the following quantities:

NPHILE	number of local-elevation (LE) coordinates
NLEPID	ID of LE potential-energy functions which will be associated to this dihedral. For n-dimensional potentials, the n dihedrals will have to be listed sequentially, using the same ID. Multiple sets of dihedrals may be associated with the same potential-energy function (thus multiple sets of dihedrals may build on the same potential-energy function).
VARTYPE	Integer defining the variable type.
NVARAT	Number of atoms needed to define the local elevation coordinate

`AI[1..NVARAT,1..NPHILE]`
atom sequence numbers of the atoms defining the local-elevation coordinate ($\leqslant$NRP)

The variable VARTYPE can take the following values

`VARTYPE = 1` Dihedral angle, NVARAT = 4

`VARTYPE = 2` Distance, NVARAT = 2

`VARTYPE = 6` Distance-field distance, according to specification in distance restraint file, NVARAT = 0

local-elevation coordinate specification block
Blockname: LOCALELEVSPEC

```
     DO 10  N=1, NPHILE
10   WRITE  (unit,11) NLEPID[N], IPLE[N], JPLE[N], KPLE[N], LPLE[N]
11   FORMAT (6I5)
```

Note that also if NVARAT $\neq$ 4, four values are read from the file.

Examples of local-elevation coordinate files are named:

*.led

3.9. Local elevation umbrella sampling database file

The LEUS database file contains a set of biasing potential energy functions that can be applied to specific subsets of collective variables so as to improve sampling (LEUS sampling).This file contains a title block and a LEUSBIAS and/or LEUSBIASBAS block (see page 4-46 and/or 4-48)

Samples of LEUS database files are named:

*.lud

3.10. Atomic friction coefficients

When performing stochastic dynamics simulations, atomic friction coefficients γ_i must be specified or calculated in some way, see Chap. 2. They may either be calculated in subroutine FRIC or specified in an atomic friction coefficient block or file.

A set of atomic friction coefficients is specified as follows:

`NR` number of atoms (=NATTOT)

`GAM[1..NR]` atomic friction coefficients

Atomic friction coefficient block
Blockname: FRICTIONSPEC

```
     DO 10  J=1, NR
10   WRITE  (unit,12) GAM[J]
12   FORMAT (24X,F15.9)
```

The first 24 positions are reserved for atom identification information.

Examples of atomic friction coefficient files are named:

*.frc

3.11. Position restraining or constraining atom specification list

When performing a simulation or energy minimization, a special interaction function term that restrains atomic positions can be added to the interaction function, see Chap. 2. Such a term may be used to keep parts of a molecular system relatively rigid. Another possibility is to keep atom positions fixed (constrained positions).

A set of atoms that are to be positionally restrained or constrained is specified as follows:

`NRC`	number of position restrained or constrained atoms ($\leqslant$NRP +NSM*NRAM)
`JRC[1..NRC]`	atom sequence numbers of the position-restrained or constrained atoms($\leqslant$NRP+NSM*NRAM for restraining)

Position restraining or constrained atom specification block
Blockname: POSRESSPEC

```
     DO 10  N=1, NRC
10   WRITE  (unit,12) JRC[N]
12   FORMAT (17X,I7)
```

The first 17 positions are reserved for residue number, residue name and atom name.

Examples of position restraining (or constraining) atom specification files are named:

*.por

3.12. B-factor restraining

Atomic mobilities or positional fluctuations can be stored in the form of isotropic B-factors

$$B_i = (8\pi^2/3) < (r_i - < r_i >)^2 >$$

The quantities characterising fluctuations or coordinate distributions are the following:

`NR`	number of atoms (= NATTOT)
`X[1..3, 1..NR]`	atomic Cartesian coordinates
`BFAC[1..NR]`	atomic isotropic B-factors

`DXY[1..3, 1..3, 1..NR] atomic positional fluctuation tensors (6 components)`

The B-factor of fluctuation blocks are the following:

Isotropic B-factor block
Blockname: BFACTOR

Formatted form

```
     DO 10  J=1, NR
     WRITE  (unit,25) MRES[J], AANMA[J], PANM[J], J, BFAC[J]
25   FORMAT (I5,2(1X,A5),I7,6F9.5)
```

Anisotropic B-factor block
Blockname: BFACTORANISO

Formatted form

```
     DO 10  J=1, NR
     WRITE  (unit,25) MRES[J], AANMA[J], PANM[J], J,
            ((8*pye**2*DXY[K1,K2,J], K1=1,K2), K2=1,3)
```

3.13. Backwards compatibility with GROMOS96

The file formats used by MD++ for topological information differ in a number of respects from those of GROMOS96:

- Molecular Topology (section Sec. 3.2)
  - the TOPVERSION block is expected to contain the version number 2.0
  - the GROMOS 96 TOPPHYSCON block must be replaced by a GROMOS05 PHYSICALCONSTANTS block (including the value of Boltzmann's constant in addition)
  - the GROMOS96 BONDTYPE block may be replaced by a BONDSTRETCHTYPE block (including force constants for a harmonic interaction form in addition to the quartic one). The two types of blocks cannot be present simultaneously. A HARMBONDTYPE block, containing only force constants for the harmonic interaction form can be used with the BONDTYPE block.
  - the GROMOS96 BONDANGLETYPE block may be replaced by a BONDANGLEBENDTYPE block (including force constants for an interaction form harmonic in the bond angle, in addition to those for the form harmonic in the angle cosine. The two types of blocks cannot be present simultaneously. A HARMBONDANGLETYPE block, containing only force constants for the harmonic interaction form can be used with the BONDANGLETYPE block.
  - the GROMOS96 DIHEDRALTYPE block may be replaced by a TORSDIHEDRALTYPE block (including arbitrary phase-shift angles in degrees within the range 0 to 360°, rather than phase-shift cosines restricted to the values -1 or +1). The two types of blocks cannot be present simultaneously.
  - a SOLUTEMOLECULES block must be included that defines all separate (covalently-linked) solute molecules (per solute unit)
  - a TEMPERATUREGROUP block must be included that defines groups of solute atoms (per solute unit) used to separate translational from internal-plus-rotational velocity components for the application of thermostatting and/or roto-translational constraints
  - a PRESSUREGROUPS block must be included that defines groups of solute atoms (per solute unit) used for the definition of a group-based virial
- Perturbation Molecular Topology (section Sec. 3.3)

- the GROMOS96 PERTATOM block must be replaced by a PERTATOMPARAM block containing in addition parameters for state A (for consistency checking).
- the GROMOS96 blocks
  PERTBONDH, PERTBOND, PERTANGLEH, PERTANGLE, PERTIMPDIHEDRALH, PERTIMPDIHEDRAL, PERTDIHEDRALH and PERTDIHEDRAL
  must be replaced by
  PERTBONDSTRETCHH, PERTBONDSTRETCH, PERTBONDANGLEH, PERTBONDANGLE, PERTIMPROPERDIHH, PERTIMPROPERDIH, PERTPROPERDIHH and PERTPROPERDIH
  respectively, containing the corresponding information in the form of type codes rather than interaction parameters.

The other types of topological information are essentially unaltered with respect to GROMOS96.

CHAPTER 4

Configurational information

4.1. Introduction

Here, it is described in which form configurational quantities, atomic coordinates and atomic coordinate dependent (e.g. energies, internal coordinates) or related (e.g. velocities, forces, atomic stochastic integrals) quantities are stored on file.

Generally, the blocks on a file are written in formatted form. A number of quantities can be written in two ways (different blocktypes):

1. *standard formatted form*
2. *reduced-information formatted form*

The former form is used when writing a file containing data concerning a *single configuration* or *time frame* of a molecular system. When performing MD, a whole time series of configurations or a *trajectory* of a molecular system is produced. Since trajectories require much more storage capacity, they are stored using the reduced-information formatted form, in which redundant information has been omitted. The extra information in the *standard formatted form block* is helpful for interpretation, but is redundant, since it is also present in the topologicaql files for the molecular system.

There is no structural difference between single configuration and trajectory files. On the latter, a specific block will occur more than once.

We note that the sequence of data (atoms, distance restraints, etc.) within a block on a configurational file must match the sequence of the same data in the corresponding topological file (molecular topology, distance restraint specification, etc.) for the molecular system, see Chap. 1.

Quantities are generally stored with one atom or quantity per line, thereby repeating the atom or quantity identification information in different blocks in order to allow for easy identification of atoms or quantities. When a program has read a topological file, it takes the topological information, such as MRES[J], AANMA[J], PANM[J], J from there and it ignores these quantities on the configurational file.

A molecular configuration may not only be characterized by atom coordinates, and atom sequence numbers, but also by other quantities such as crystallographic B-factors.

In MD or SD simulations quantities such as velocities, stochastic integrals and random number generator seed need to be stored with a final configuration in order to be able to later continue the simulation. When the volume of the system varies with time, i.e. when pressure coupling is applied, the dimensions of the periodic box need to be stored, as well as the (possibly changing) reference positions if position restraining is also applied.

For simulations under periodic boundary conditions (where particles diffuse in an infinite periodic system) lattice-shift vectors are stored along with the atomic coordinates translated into the reference box (these are used for the calculation of a group-based pressure). Application of Nosé-Hoover chains for thermostatting and barostatting require the storage of the thermostat variables. Note that MD++ only supports Nosé-Hoover chains for thermostatting. Application of roto-translational constraints require the storage of parameters defining the position and orientation of specific atom groups.

Free energy calculations using the slow-growth or continuous λ-change technique require the storage of the actual λ-value and the cumulative derivatives of the Hamiltonian terms with respect to λ. When using time-averaging in atom-atom distance restraining or in ^3J-value restraining, the current averaged distances or ^3J-values need to be stored. When applying local-elevation search, the information with respect to parts of the energy hypersurface that have been elevated so far need to be stored in order to use it when continuing the simulation.

In the next sections the various quantities and their mode of storage is described.

4.2. Atomic coordinates

The atomic Cartesian coordinates of a molecular configuration can be stored as follows:

`NR`	number of atoms
`NDIM`	dimensionality of the Cartesian space
`X[1..3, 1..NR]`	atomic Cartesian coordinates (MD: trajectory at time t, final configuration at time t+Δt)
`MRES[1..NR]`	residue number
`AANMA[1..NR]`	residue name
`PANM[1..NR]`	atom name
`J`	atom sequence number
`XC[1..3, 1..NR]`	atomic Cartesian reference positions for position restraining (MD: final configuration at time t+Δt)
`NLSHFT[1..3,1..NR]`	lattice-shift vectors defining the position of particles in the infinite periodic system relative to their position in the reference box (in units of the box edge vectors for rectangular and triclinic boxes; in units of half the box edge for a truncated-octahedron box)
`RPOL[1..3, 1..NR]`	displacement of charge-on-spring from atomic centres

The atomic coordinate blocks are the following:

3-dimensional coordinate block
Blockname: POSITION

```
     DO 10  J=1, NR
10   WRITE  (unit,12) MRES[J], AANMA[J], PANM[J], J, (X[M,J], M=1,3)
12   FORMAT (I5,2(1X,A5),I7,3F15.9)
```

Blockname: POSITIONRED

Reduced-information form

```
     DO 10  J=1, NR
10   WRITE  (unit,13) (X[M,J], M=1,3)
13   FORMAT (3F15.9)
```

Reference coordinate block

Blockname: REFPOSITION

```
     DO 10  J=1, NR
10   WRITE  (unit,12) MRES[J], AANMA[J], PANM[J], J, (XC[M,J], M=1,3)
12   FORMAT (I5,2(1X,A5),I7,3F15.9)
```

Coordinates after SHAKE failure

Blockname: SHAKEFAILPOSITION

same format as POSITION (other quantities are written to the final coordinate file in the usual blocks)

Coordinates just before SHAKE failure (i.e. before the coordinate resetting was attempted)

Blockname: SHAKEFAILPREVPOSITION

same format as POSITION (other quantities are written to the final coordinate file in the usual blocks)

Blockname: LATTICESHIFTS

```
     DO 10  J=1, NR
10   WRITE  (unit,30) (NLSHFT[M,J], M=1,3)
30   FORMAT (3I10)
```

Displacement of charge-on-spring from polarizable centres
Blockname: COSDISPLACEMENTS

```
     DO 10  J=1, NR
10   WRITE  (unit,13) RPOL[M,J], M=1,3)
```

4.3. Atomic velocities

The atomic velocities (at time t-Δt/2) belonging to a molecular configuration (at time t) can be stored as follows:

`V[1..NDIM, 1..NR]` atomic velocities (trajectory at time t-Δt/2, final velocities at t+Δt/2)

3-dimensional velocity block
Blockname: VELOCITY

```
     DO 10  J=1, NR
10   WRITE  (unit,12) MRES[J], AANMA[J], PANM[J], J, (V[M,J], M=1,3)
12   FORMAT (I5,2(1X,A5),I7,3F15.9)
```

Blockname: VELOCITYRED

Reduced-information form

```
     DO 10  J=1, NR
10   WRITE  (unit,13) (V[M,J], M=1,3)
13   FORMAT (3F15.9)
```

4.4. Atomic forces

The atomic forces (at time t) belonging to a molecular configuration (at time t) can be stored as follows:

`FF[1..NDIM, 1..NR]` atomic free-flight forces (trajectory at time t); these are the raw atomic forces (physical forces and possibly including special forces) prior to the application of any constraint

`FC[1..NDIM,1..NR]` atomic constraint forces (trajectory at time t); these are the atomic forces induced by the application of all constraints on the system (typically SHAKE, but also possibly including special constraints)

The sum of the two contributions represents the actual total force used to propagate the system via the integration scheme.

3-dimensional free-flight force block
Blockname: FREEFORCE

```
     DO 10  J=1, NR
10   WRITE  (unit,12) MRES[J], AANMA[J], PANM[J], J, (FF[M,J], M=1,3)
12   FORMAT (I5,2(1X,A5),I7,3F15.9)
```

Blockname: FREEFORCERED

Reduced-information form

```
     DO 10  J=1, NR
10   WRITE  (unit,14) (FF[M,J], M=1,3)
14   FORMAT (3F 20.9)
```

3-dimensional constraint force block
Blockname: CONSFORCE

```
     DO 10  J=1, NR
10   WRITE  (unit,12) MRES[J], AANMA[J], PANM[J], J, (FC[M,J], M=1,3)
12   FORMAT (I5,2(1X,A5),I7,3F15.9)
```

Blockname: CONSFORCERED

Reduced-information form

```
     DO 10  J=1, NR
10   WRITE  (unit,13) (FC[M,J], M=1,3)
13   FORMAT (3F15.9)
```

4.5. Atomic stochastic integrals

When performing stochastic dynamics (SD) using the leap frog algorithm, the integrals of the stochastic forces over time are correlated between successive time steps. Therefore, the stochastic integrals SX[1..NDIM, 1..NR] over the time interval (t+Δt/2, t+Δt) are stored to allow for continuation runs. For the same purpose the random number generator seed needs to be stored.

`SX[1..NDIM, 1..NR]` atomic stochastic integrals (interval t+Δt/2 to t+Δt)

IG	random number generator seed (at time t+Δt)

3-dimensional stochastic integral block
Blockname: STOCHINT

```
     DO 10  J=1, NR
10   WRITE  (unit,15) MRES[J], AANMA[J], PANM[J], J, (SX[M,J], M=1,3)
     WRITE  (unit,16) IG
15   FORMAT (I5,2(1X,A5),I7,3E15.7)
16   FORMAT (I10)
```

4.6. Periodic box

When using pressure coupling in a MD simulation, the parameters characterizing the size and shape of the periodic box that contains the molecular system as well as its orientation in space are a function of time, so these parameters need to be stored.

NTB	type of boundary conditions; truncated-octahedron (-1), vacuum (0), rectangular (1), or triclinic (2)
BOX[1..3]	lengths of the a-, b- and c-edges of the periodic box (trajectory at time t, final configuration at time t+Δt)
ALPHA	angle between the b and c axes in degrees
BETA	angle between the a and c axes in degrees
GAMMA	angle between the a and b axes in degrees
PHI	Euler yaw angle of the box (z-axis rotation) in degrees
THETA	Euler pitch angle of the box (y-axis rotation) in degrees
PSI	Euler roll angle of the box (x-axis rotation) in degrees
OX	(Cartesian) x-coordinate of origin of triclinic box
OY	(Cartesian) y-coordinate of origin of triclinic box
OZ	(Cartesian) z-coordinate of origin of triclinic box

Boxsize block
Blockname: GENBOX

```
     WRITE  (unit,17) NTB
     WRITE  (unit,18) (BOX[M], M=1,3)
     WRITE  (unit,18) ALPHA, BETA, GAMMA
     WRITE  (unit,18) PHI, THETA, PSI
     WRITE  (unit,18) OX, OY, OZ
17   FORMAT (I5)
18   FORMAT (3F15.9)
```

4.7. Nose-Hoover chain thermostat variables

When using temperature coupling in an MD simulation based on the Nosé-Hoover chain thermostat, the values of the thermostat variables need to be stored.

`NUM_NHC_TMP_BTH` number of heat baths employing Nosé-Hoover chain thermostat coupling

`NUM_VAR_NHC_TMP_BTH[1..NUM_NHC_TMP_BTH]`
number of variables used for each bath

`VAL_VAR_NHC_TMP_BTH[I=1..NUM_NHC_TMP_BTH,1..NUM_VAR_NHC_TMP_BTH[I]]`
values of the corresponding thermostat variables

Nosé-Hoover chain thermostat variables block
Blockname: NHCVARIABLES

```
     DO 10  I=1, NUM_NHC_TMP_BTH
10   WRITE  (unit,41) (VAL_VAR_NHC_TMP_BTH[I,J], J=1..NUM_VAR_NHC_TMP_BTH[I])
41   FORMAT (2I5,5F15.9)
```

4.8. Roto-translational constraints reference variables

When using translational or/and rotational constraining in an MD simulation, the values of the variables defining the reference position or/and orientation of all constrained atoms need to be stored. In MD++ roto-translational constraints are always applied on the first specified number of atoms. Therefore, only the translation and rotation matrices and reference positions of the first atoms are written.

`RTCLAST` The first RTCLAST atoms are roto-translationally contrained.

`RTCTM[1..3,1..3]` The translation matrix.

`RTCRM[1..3,1..3]` The rotation matrix.

`RTCREF[1..3,1..NUMRTC]`
The reference atomic coordinates.

Rototranslational reference matrices and positions block
Blockname: ROTTRANSREFPOS

```
     DO 10  J=1, 3
10   WRITE  (unit,13) (RTCTM[M,J], M=1,3)
     DO 11  J=1, 3
11   WRITE  (unit,13) (RTCRM[M,J], M=1,3)
13   FORMAT (9F15.9)
     DO 20  I=1, RTCLAST
20   WRITE  (unit,40) (RTCREF[M,I], M=1,3)
40   FORMAT (3F15.9)
```

4.9. Perturbation data

When applying a perturbation to the Hamiltonian in a simulation in order to determine a free energy difference between two states of a molecular system using the so-called slow-growth or continuous λ-change technique, the value for the thermodynamic integration coupling parameter λ at time t+Δt needs to be stored to allow for a continuation run. This value is stored in the PERTDATA block of the molecular configuration file.

RLAM	thermodynamic integration coupling parameter λ (at time t+Δt)

Perturbation data block
Blockname: PERTDATA

```
     WRITE  (unit,19) RLAM
19   FORMAT (E15.7)
```

4.10. Atom-atom distance restraints

When applying time-averaging in atom-atom distance restraining in a simulation, the exponentially weighted average of r_{ij}^{-3} for the restrained atom-atom distance r_{ij} needs to be stored in order to allow for continuation runs.

NDR	number of distance restraint atom pairs
RIIAVE[1..NDR]	minus $\frac{1}{3}$ power of the exponentially weighted average of r_{ij}^{-3} for the restrained atom-atom distances r_{ij} (at time t)

Exponentially averaged distance restraint block
Blockname: DISRESEXPAVE

```
     DO 10  N=1, NDR
10   WRITE  (unit,19) RIIAVE[N]
19   FORMAT (E15.7)
```

4.11. ^3J-coupling constant restraints

When applying time-averaging in ^3J-coupling constant restraining in a simulation, the exponentially weighted time-average of the ^3J-coupling value(s) need to be stored in order to allow for continuation runs.

NJR	number of ^3J-coupling constant restraints.
JVALAV[1..NJR]	exponentially weighted average of each ^3J-coupling value (at time t).

Exponentially averaged ^3J-value restraint block
Blockname: JVALUERESEXPAVE

```
     DO 10  N=1, NJR
10   WRITE  (unit,19) JVALAV[N]
19   FORMAT (E15.7)
```

In MD++, a local-elevation interaction term can be applied (along with time-averaging) to perform ^3J-coupling constant restraining. In this case, the time-averaged, weighted heights at time t of the Gaussians describing the potential-energy penalty function at grid points [1..NJLEGR] need to be stored in order to allow for continuation runs.

NJLEGR	number of grid points.

`HJLEG[1..NJRJ, 1..NJLEGR]`
time-averaged, weighted height of the local-elevation Gaussian at grid point M=[1..NJLEGR] for the ^3J-coupling-related dihedral angle θ (at time t).

Time-average weighted height of the local elevation penalty functions used in combination with ^3J-coupling constant restraining
Blockname: JVALUERESEPS

```
     DO 10  N=1, NJR
     DO 11  M=1, NJLEGR
11   WRITE  (unit,19) HJLEG[N][M]
10   CONTINUE
19   FORMAT (E15.7)
```

In MD++, the force constant can be periodically scaled. In this case the scaling constant and time need to be stored in order to allow for continuation runs.

`JVALSC[1..NJR]` integer (0 or 1) which indicates whether the ^3J-value's force constant is scaled.

`JVALT[1..NJR]` current time in scaling period.

Periodoc scaling ^3J-value restraint block
Blockname: JVALUEPERSCALE

```
     DO 10  N=1, NJR
10   WRITE  (unit,19) JVALSC[N], JVALT[N]
19   FORMAT (I10, F15.9)
```

4.12. S^2-order parameter restraints

When applying S^2-order paramter restraining in a simulation, the exponentially weighted time averages of $Q_{\alpha\beta}$ and D need to be stored in order to allow for continuation runs.

`NOPR` number of order parameter restraints

`QABAVE[1..NOPR,A,B]`
exponentially weighted average of matrix elements $Q_{\alpha\beta}$ (9 elements) (at time t)

`DAVE[1..NOPR]` exponentially weighted average D (at time t)

Exponentially averaged S^2-order parameter restraint block
Blockname: ORDERPARAMRESEXPAVE

```
     DO 9   N=1, NOPR
     WRITE  (unit,19) QABAVE[N,1,1],QABAVE[N,1,2],QABAVE[N,1,3],QABAVE[N,2,1],QABAVE[N,2,2]
     WRITE  (unit,19) QABAVE[N,2,3],QABAVE[N,3,1],QABAVE[N,3,2],QABAVE[N,3,3],DAVE[N]
9    CONTINUE
19   FORMAT (E15.7)
```

4.13. Crystallographic restraints

When applying time-averaging in structure-factor amplitude or electron density restraining in a simulation, the exponentially weighted average of the complex structure factor needs to be stored in order to allow for continuation runs.

`NFXR` number of structure-factor amplitudes

`FXRAVE[1..NFXR]` time-averaged structure-factor amplitude

`PHXRAVE[1..NFXR]` time-averaged structure-factor phase

Exponentially averaged structure factor block
Blockname: XRAYRESEXPAVE

```
     DO 10  N=1, NFXR
10   WRITE  (unit,19) FXRAVE[N], PHXRAVE[1..NFXR]
19   FORMAT (E15.7, E15.7)
```

When refining the atomic B-factors they are written to the configuration as well (XRAYBFOCCSPEC block).

4.14. Local-elevation data

When performing a simulation in which the local-elevation interaction term is switched on and the memory progressively builds up (LE searching), data on the local-elevation conformations that have been visited so far during the simulation needs to be stored in order to allow for continuation runs.

`NUMB` number of umbrella potential energy functions contained in the block

`NLEPID[1..NUMB]` potential energy function ID

`NLEDIM[1..NUMB]` dimensionality of the potential energy function

`CLES[1..NUMB]` force constant associated with the local functions

`VARTYPE[1..NUMB,1..NLEDIM]`
type of variable (1: Dihedral angle, 2: Distance, 6: Distance-field distance)

`NTLEFU[1..NUMB,1..NLEDIM]`
functional form of the local functions (0: truncated polynomials; 1: Gaussian)

`WLES[1..NUMB,1..NLEDIM]`
width of the local functions in units of the grid spacing

`RLES[1..NUMB,1..NLEDIM]`
cutoff applied to the range of action of the local functions in units of grid spacing

`NGRID[1..NUMB,1..NLEDIM]`
number of grid points used along each dimension

`GRIDMIN[1..NUMB,1..NLEDIM]`
minimum grid point used along each dimension

`GRIDMAX[1..NUMB,1..NLEDIM]`
maximum grid point used along each dimension. If GRIDMAX[N]==GRIDMIN[N] a cyclic variable is assumed, applying NLEGRD[N] unidistant grid points over the whole variable range, with first grid point at GRIDMIN[N]

`NCONLE[1..NUMB]` number of LE conformations visited and stored so far in memory (at time t)

`ICONF[1..NUMB,1..NLEDIM, 1..NCONLE]`
integer coded representation of LE conformations. Position of grid point in dimension N for potential K is given as POS = GRIDMIN[N] + (ICONF[N, K] − 1)GRIDSPACING[N]

`NVISLE[1..NUMB,1..NCONLE]`
number of times LE conformations have been visited so far (at time t)

local-elevation memory block
Blockname: LEUSBIAS

```
     WRITE  (unit,20) NUMB
DO 9 I=1, NUMB
     WRITE  (unit,21) NLEPID[I], NDIM[I], CLES[I]
     DO 10  N=1, NDIM[I]
10   WRITE  (unit,22) VARTYPE[I,N],NTLEFU[I,N],WLES[I,N],RLES[I,N], &
            NGRID[I,N],GRIDMIN[I,N],GRIDMAX[I,N]
     WRITE  (unit,20) NCONLE[I]
     DO 11  N=1, NCONLE[I]
11   WRITE  (unit,23) (NVISLE[I,N], (ICONF[I,M,N], M=1, NDIM[I])
9    CONTINUE
20   FORMAT (1I8)
21   FORMAT (2I8, 1E18.10)
22   FORMAT (2I8, 2E18.10,I8,2E18.10)
23   FORMAT (8I8)
```

4.15. Ball and stick local-elevation data

When performing a simulation in which the Ball and Stick local-elevation interaction term is switched on and the memory progressively builds up (LE searching), data on the local-elevation conformations that have been visited so far during the simulation needs to be stored in order to allow for continuation runs.

`NUMB` number of umbrella potential energy functions contained in the block

`NLEPID[1..NUMB]` potential energy function ID

`NLEDIM[1..NUMB]` dimensionality of the potential energy function

`ACTPOT[1..NUMB]` the ID of the active potential energy function (not used when EDS combination is applied)

`BETA[1..NUMB]` the factor $s\beta$ used for EDS combination of bias potential energy functions

`VARTYPE[1..NUMB,1..NLEDIM]`
type of variable (1: Dihedral angle, 2: Distance, 6: Distance-field distance)

`DIMSCALE[1..NUMB,1..NLEDIM]`
by which factor should the respective variable be divided

`SHIFTTYPE[1..NUMB,1..NLEDIM]`
0: Do not shift; 1: Shift to nearest image; 2: Apply shift vectors

`REFSHIFT[1..NUMB,1..NLEDIM]`
Value of coordinate at last time step, used for updating shift vectors (only used for SHIFTTYPE=2)

`NSPHERES[1..NUMB]` Number of defined spherical potentials

`SID[1..NUMB,1..NSPHERES]`
ID of defined sphere

`NPRAD[1..NUMB,1..NSPHERES]`
Number of radial grid points of defined sphere

`DR[1..NUMB,1..NSPHERES]`
Radial distance between grid points

`IBUILD[1..NUMB,1..NSPHERES]`
0: Do not build; 1: Build proportional to EDS weight w; 3: Build proportional to EDS weight and grid index to the power of SCALEVAL

`SCALEVAL[1..NUMB,1..NSPHERES]`
Value for scaling the build-up as function of grid index

`CLES[1..NUMB,1..NSPHERES]`
(Current) build-up force constant [kJ/mol]

`REDFAC[1..NUMB,1..NSPHERES]`
Factor for reduction of build-up force constant

`CRES[1..NUMB,1..NSPHERES]`
Force constant [kJ/mol] for restraint

`VADD[1..NUMB,1..NSPHERES]`
Potential energy [kJ/mol] added to the energy of sphere (grid-point independent)

`CENTRE[1..NUMB,1..NSPHERES,1..NDIM]`
Value of LE coordinates defining centre of the sphere

`VSPHERE[1..NUMB,1..NSPHERES,1..NPRAD]`
Value of LE potential energy function at radial grid point [kJ/mol]

`VISSPHERE[1..NUMB,1..NSPHERES,1..NPRAD]`
Number of visits at grid point

`NLINES[1..NUMB]` Number of defined lines

`LID[1..NUMB,1..NLINES]`
ID of defined line

NPLINE[1..NUMB,1..NLINES]
: Number of grid points on line

IBUILD[1..NUMB,1..NLINES]
: 0: Do not build 1: Build

CLES[1..NUMB,1..NLINES]
: LE force constant [kJ/mol]

REDFAC[1..NUMB,1..NLINES]
: Factor to reduce the LE force constant

VADD[1..NUMB,1..NLINES]
: Potential energy [kJ/mol] added to the energy of line (grid-point independent)

PSTART[1..NUMB,1..NLINES,1..NDIM]
: LE coordinates defining starting point of line

PEND[1..NUMB,1..NLINES,1..NDIM]
: LE coordinates defining end point of line

NDIS[1..NUMB,1..NLINES]
: Number of displacement vectors

DISVEC[1..NUMB,1..NLINES,1..NDIS,1..NDIM]
: Vector components of displacement vectors

VLINE[1..NUMB,1..NLINES,1..NPLINE]
: Potential energy [kJ/mol] at grid point

VISLINE[1..NUMB,1..NLINES,1..NPLINE]
: Number of visits at grid point

WIDTH[1..NUMB,1..NLINES,1..NPLINE]
: Width orthogonal to the line before start of restraining potential energy function

CRES[1..NUMB,1..NLINES,1..NPLINE]
: Restraining force constant

LAM[1..NUMB,1..NLINES,1..NPLINE]
: Value of λ for fixed λ simulations (currently unused)

DIS[1..NUMB,1..NLINES,1..NPLINE,1..NDIS]
: Displacememt of current grid point along respective (orthonormalised) displacement vectors

local-elevation ball and stick memory block
Blockname: LEUSBIASBAS

```
     WRITE  (unit,21) NUMB
DO 9 I=1, NUMB
     WRITE  (unit,22) NLEPID[I], NDIM[I], ACTPOT[I], BETA[I]
     DO 11  N=1, NDIM[I]
11   WRITE  (unit,23) VARTYPE[I,N],DIMSCALE[I,N],SHIFTTYPE[I,N],REFSHIFT[I,N]
     WRITE  (unit,21) NSPHERES[I]
     DO 12  N=1, NSPHERES[I]
     WRITE  (unit,24) ID[I,N], NPRAD[I,N], DR[I,N], IBUILD[I,N], RBUILD[I,N], CLES[I,N], &
            REDFAC[I,N], CRES[I,N], VADD[I,N], (CENTRE[I,N,M], M=1, NDIM[I])
     WRITE  (unit,25) (VSPHERE[I,M], M=1,NPRAD[I,N])
     WRITE  (unit,25) (VISSPHERE[I,M], M=1,NPRAD[I,N])
12   CONTINUE
```

```
      WRITE  (unit,21) NLINES[I]
      DO 13  N=1, NLINES[I]
      WRITE  (unit,26) ID[I,N], NPLINE[I,N], IBUILD[I,N], CLES[I,N], REDFAC[I,N], VADD[I,N]
      WRITE  (unit,25) (PSTART[I,N,M], M=1,NDIM[I])
      WRITE  (unit,25) (PEND[I,N,M], M=1,NDIM[I])
      WRITE  (unit,21) NDIS[I,N]
      DO 14  K=1, NDIS[I,N]
      WRITE  (unit,25) (DISVEC[I,N,K,M], M=1,NDIM[I])
14    CONTINUE
      DO 15  M=1, NPLINE[I,N]
      WRITE  (unit,25) VLINE[I,N,M], VISLINE[I,N,M], WIDTH[I,N,M], CRES[I,N,M], LAM[I,N,M], &
             (DIS[I,N,M,K], K=1,NDIS[I,N])
15    CONTINUE
13    CONTINUE
      WRITE  (unit,27) NSTATES[I], NCHECK[I], NCHECKCUR[I]
      DO 16  N=1, NSTATES[I]
      WRITE  (unit,28/29) TYPE[I,N], NVISITS[I,N], (PARAMS[I,N,M], NPAR)
16    CONTINUE
9     CONTINUE
21    FORMAT (1I8)
22    FORMAT (3I8,1E18.10)
23    FORMAT (1I8,1E18.10,I8,5E18.10)
24    FORMAT (2I8,1E18.10,I8,50E18.10)
25    FORMAT (50E18.10)
26    FORMAT (3I8,5E18.10)
27    FORMAT (3I8)
28    FORMAT (3I8,50E18.10)
29    FORMAT (3I8,E18.10,I8)
```

4.16. Time or step number data

Generally, trajectory files are written such that the time frames are equidistant in time, i.e. correspond to a multiple of a specified number of simulation steps. So, time or step number of a block are known in that case. When selecting configurations to be stored using a criterion such as low potential energy, the configurations will not be equidistant in time. In that case time or step number information should be added to a configuration.

`T`	time in the simulation ($t = t_n$)
`NSTEP`	step number since the beginning of the current runs (n)

Time and step number data block
Blockname: TIMESTEP

```
      WRITE  (unit,21) NSTEP, T
21    FORMAT (I15,F20.9)
```

4.17. Energies, pressure, volume and free-energy data

Program MD++ writes out the following arrays:

`ENER[1]`	total energy of the molecular system (at time t)
`ENER[2]`	total kinetic energy of the molecular system (at time t)
`ENER[3]`	total potential energy of the molecular system (at time t)

ENER[4]	total energy of covalent terms (solutes, at time t)
ENER[5]	total energy of bond-stretching terms (solutes, at time t)
ENER[6]	total energy of bond-angle bending terms (solutes, at time t)
ENER[7]	total energy of improper (harmonic) dihedral angle terms (solutes, at time t)
ENER[8]	total energy of (trigonometric) dihedral angle terms (solutes, at time t)
ENER[9]	total energy of crossdihedral angle terms (solutes, at time t)
ENER[10]	total energy of nonbonded terms (solutes, at time t)
ENER[11]	total energy of van der Waals interaction terms (at time t)
ENER[12]	total energy of electrostatic interaction terms (at time t)
ENER[13]	total energy of lattice sum terms (at time t)
ENER[14]	total energy of lattice sum pair term (at time t)
ENER[15]	total energy of lattice sum real space term (at time t)
ENER[16]	total energy of lattice sum reciprocal space term (at time t)
ENER[17]	total energy of lattice sum A term (at time t)
ENER[18]	total energy of lattice sum self term (at time t)
ENER[19]	total energy of lattice sum surface term (at time t)
ENER[20]	total energy of polarisation self term (at time t)
ENER[21]	total energy of special terms (at time t)
ENER[22]	total energy of SASA term (at time t)
ENER[23]	total energy of SASA volume term (at time t)
ENER[24]	total energy due to constraints in the molecular system (at time t)
ENER[25]	total energy of atom-atom distance restraint terms (at time t)
ENER[26]	total energy of distance-field restraining terms (at time t)
ENER[27]	total energy of dihedral angle restraining terms (at time t)
ENER[28]	total energy of atom position restraining terms (at time t)
ENER[29]	total energy of ^3J-value restraining terms (at time t)
ENER[30]	total energy of X-ray restraining terms (at time t)
ENER[31]	total energy of local-elevation terms (at time t)
ENER[32]	total energy of S^2 order parameter restraining terms (at time t)
ENER[33]	total energy of symmetry restraining terms (at time t)
ENER[34]	total energy of non-accelerated EDS reference state in accelerated EDS (at time t)
ENER[35]	total energy of EDS reference state (at time t)
ENER[36]	accelerated EDS parameter E_{max} (at time t)

ENER[37] accelerated EDS parameter E_{min} (at time t)

ENER[38] average energy of the end-state with the lowest energy in accelerated EDS parameter search (at time t)

ENER[39] standard deviation of the energy of the end-state with the lowest energy in accelerated EDS parameter search (at time t)

ENER[40] total entropy term (at time t)

ENER[41] total energy QM

ENER[42] total energy of ball and stick local elevation (at time t)

ENER[43] total energy of RDC restraining terms (at time t)

NBATHS number of temperature baths defined for constant temperature simulations

KINENER[1,1..NBATHS]
total kinetic energy of individual temperature baths (at time t)

KINENER[2,1..NBATHS]
total translational kinetic energy of the centres of mass of the molecules coupled to the individual baths (at time t)

KINENER[3,1..NBATHS]
total internal-rotational kinetic energy of the individual temperature baths (at time t)

NEGR number of groups G_i of atoms for which the energy terms are separately stored

BONDED[1,1..NEGR] total energy of bond-stretching terms of which the first atom belongs to the energy group G

BONDED[2,1..NEGR] total energy of bond-angle bending terms of which the first atom belongs to the energy group G

BONDED[3,1..NEGR] total energy of improper (harmonic) dihedral angle terms of which the first atom belongs to the energy group G

BONDED[4,1..NEGR] total energy of dihedral (trigonometric) angle terms of which the first atom belongs to the energy group G

BONDED[5,1..NEGR] total energy of crossdihedral angle terms of which the first atom belongs to the energy group G

NONBONDED[1,1..NEGR*(NEGR+1)/2]
total Van der Waals interaction energies between atoms belonging to the different groups G_i (at time t); the order of the group-group energies is 1-1, 1-2, 2-2, ..., 1-NEGR, 2-NEGR, ..., NEGR-NEGR

NONBONDED[2,1..NEGR*(NEGR+1)/2]
idem, but for the total electrostatic interaction

NONBONDED[3,1..NEGR*(NEGR+1)/2]
idem, but for the total lattice sum real space energy

NONBONDED[4,1..NEGR*(NEGR+1)/2]
idem, but for the total lattice sum reciprocal space energy

SPECIAL[1,1..NEGR] total constraint energy per energy group G

SPECIAL[2,1..NEGR] total energy of position restraining terms per energy group G

SPECIAL[3,1..NEGR] total energy of distance restraints per energy group G

SPECIAL[4,1..NEGR] total energy of distance-field restraints per energy group G

SPECIAL[5,1..NEGR] total energy of dihedral restraints per energy group G

SPECIAL[6,1..NEGR] total energy of SASA term per energy group G

SPECIAL[7,1..NEGR] total energy of SASA volume term per energy group G

SPECIAL[8,1..NEGR] total energy of ^3J-value restraints per energy group G (= 0; the ^3J-value restraints are not split up per energy group)

SPECIAL[9,1..NEGR] total energy of RDC restraints per energy group G

SPECIAL[10,1..NEGR]
total energy of local-elevation terms per energy group G (= 0; the local-elevation terms are not split up per energy group)

SPECIAL[11,1..NEGR]
total energy of X-ray restraining terms per energy group G (= 0; the X-ray restraining terms are not split up per energy group)

NEDS number of EDS states

EDSENER[1,1..NEDS] total potential energy per EDS state

EDSENER[2,1..NEDS] total nonbonded energy per EDS state

EDSENER[3,1..NEDS] total special energy functions per EDS state

EDSENER[4,1..NEDS] energy offset parameter per EDS state in accelerated EDS

MASS total mass of all particles in the system

TEMPERATURE[1,1..NBATHS
temperature of the part of the system that is coupled to every temperature bath

TEMPERATURE[2,1..NBATHS
temperature associated with the centre of mass translational degrees of freedom of the submolecules that are coupled to every temperature bath

TEMPERATURE[3,1..NBATHS
temperature associated with the internal and rotational degrees of freedom of the part of the system that that is coupled to every temperature bath

TEMPERATURE[4,1..NBATHS
scaling factor for scaling the corresponding degrees of freedom for every temperature bath (used at time t $t + \Delta t/2$)

VOLUME total volume of the computational box.

BOX[1..3,1..3] triclinic unit vectors K, L, M

PRESSURE[1] total pressure of the system

PRESSURE[2] total virial of the system

PRESSURE[3] total translational kinetic energy matrix for centre of mass for all submolecules

PRES[1..3,1..3] pressure tensor

VIRIAL[1..3,1..3] virial matrix

`KINETIC[1..3,1..3]` translational kinetic energy matrix for centre of mass for all submolecules

`RLAM` perturbation parameter λ (at time t)

`FREEENER[1..38]` derivatives of the various terms of the Hamiltonian with respect to λ; the energy terms are the same as in ENER[1..38]

`FREEKINENER[1..3,1..NBATHS]`
derivatives of the kinetic energy terms with respect to λ; the energy terms are the same as in KINENER[1..3,1..NBATHS]

`FREEBONDED[1..5,1..NEGR]`
derivatives of the bonded energy terms with respect to λ; the energy terms are the same as in BONDED[1..5,1..NBATHS]

`FREENONBONDED[1..4,1..NEGR*(NEGR+1)2]`
derivatives of the various terms of the Hamiltonian with respect to λ; the energy terms are the same as in NONBONDED[1..4,1..NEGR*(NEGR+1)2]

`FREESPECIAL[1..11,1..NEGR]`
derivatives of the special interaction energy terms with respect to λ; the energy terms are the same as in SPECIAL[1..11, 1..NEGR]

`FREEEDSENER[1..3,1..NEDS]`
derivatives of the EDS energies with respect to λ; the energy terms are the same as in EDSENER[1..3,1..NEDS]

Energy block
Blockname: ENERGY03

```
     DO 10  N=1,43
10   WRITE  (unit,23) ENER[N]
     WRITE  (unit,22) NBATHS
     DO 11  N=1,NBATHS
11   WRITE  (unit,23) KINENER[1,N],KINENER[2,N],KINENER[3,N]
     WRITE  (unit,22) NEGR
     DO 12  N=1, NEGR
12   WRITE  (unit,23) BONDED[1,N],BONDED[2,N],BONDED[3,N],
                      BONDED[4,N],BONDED[5,N]
     DO 13  N=1, NEGR*(NEGR+1)/2
13   WRITE  (unit,23) NONBONDED[1,N], NONBONDED[2,N],
                      NONBONDED[3,N], NONBONDED[4,N]
     DO 14  N=1, NEGR
14   WRITE  (unit,23) SPECIAL[1,N],SPECIAL[2,N],SPECIAL[3,N],SPECIAL[4,N],
                      SPECIAL[5,N],SPECIAL[6,N],SPECIAL[7,N],SPECIAL[8,N]
                      SPECIAL[9,N],SPECIAL[10,N],SPECIAL[11,N]
     WRITE  (unit,22) NEDS
     DO 15  N=1,NEDS
15   WRITE  (unit,23) EDSENER[1,N],EDSENER[2,N],EDSENER[3,N],EDSENER[4,N]
22   FORMAT (I5)
23   FORMAT (11E17.9)
```

Volume, pressure block
Blockname: VOLUMEPRESSURE03

```
     WRITE  (unit,23) MASS
```

```
     WRITE  (unit,22) NBATHS
     DO 10  N=1, NBATHS
10   WRITE  (unit,23) TEMPERATURE[1,N], TEMPERATURE[2,N], TEMPERATURE
                      [3,N], TEMPERATURE[4,N]
     WRITE  (unit,23) VOLUME
     DO 11  N=1, 3
11   WRITE  (unit,23) BOX[1,N], BOX[2,N], BOX[3,N]]
     DO 12  N=1, 3
12   WRITE  (unit,23) PRESSURE[N]
     DO 13  N=1, 3
13   WRITE  (unit,23) PRESS[1,N], PRESS[2,N], PRESS[3,N]
     DO 14  N=1, 3
14   WRITE  (unit,23) VIRIAL[1,N], VIRIAL[2,N], VIRIAL[3,N]
     DO 15  N=1, 3
15   WRITE  (unit,23) KINETIC[1,N], KINETIC[2,N], KINETIC[3,N]
```

Free energy derivative lambda block
Blockname: FREEENERGYDERIVS03

```
     WRITE  (unit, 23)RLAM
     DO 10  N=1,38
10   WRITE  (unit,23) FREEENER[N]
     WRITE  (unit,22) NBATHS
     DO 11  N=1,NBATHS
11   WRITE  (unit,23) FREEKINENER[1,N], FREEKINENER[2,N], FREEKINENER
                      [3,N]
     WRITE  (unit,22) NEGR
     DO 12  N=1, NEGR
12   WRITE  (unit,23) FREEBONDED[1,N],FREEBONDED[2,N],FREEBONDED[3,N],
                      FREEBONDED[4,N],FREEBONDED[5,N]
     DO 13  N=1, NEGR*(NEGR+1)/2
13   WRITE  (unit23)  FREENONBONDED[1,N],FREENONBONDED[2,N],
                      FREENONBONDED[3,N],FREENONBONDED[4,N]
     DO 14  N=1, NEGR
14   WRITE  (unit, 23)FREESPECIAL[1,N], FREESPECIAL[2,N], FREESPECIAL[3,N]
                      FREESPECIAL[4,N],FREESPECIAL[5,N],FREESPECIAL[6,N]
                      FREESPECIAL[7,N],FREESPECIAL[8,N],FREESPECIAL[9,N]
                      FREESPECIAL[10,N],FREESPECIAL[11,N]
     DO 15  N=1,NEDS
15   WRITE  (unit,23) FREEEDSENER[1,N],EDSENER[2,N],EDSENER[3,N]
22   FORMAT (I5)
23   FORMAT (11E17.9)
```

4.18. Atomic B-factors and positional fluctuations

Atomic mobilities or positional fluctuations can be stored in the form of isotropic B-factors

$$B_i = (8\,\pi^2\,/3) < (r_i - < r_i >)^2 > .$$

The quantities characterizing fluctuations of coordinate distributions are the following:

`NR`	number of atoms
`X[1..3, 1..NR]`	atomic Cartesian coordinates

`BFAC[1..NR]` atomic isotropic B-factors

The B-factor or fluctuation blocks are the following:

Isotropic B-factor block
Blockname: BFACTOR

Formatted form

```
    DO 10  J=1, NR
10  WRITE  (unit,25) MRES[J], AANMA[J], PANM[J], J, BFAC[J]
25  FORMAT (I5,2(1X,A5),I7,6F9.5)
```

4.19. Accelerated EDS parameter search data

For accelerated EDS parameter search simulations, several data are stored to allow for continuation of the search in a new run. The quantities are:

`AEDSS[1]` accelerated EDS parameter E_{max}

`AEDSS[2]` accelerated EDS parameter E_{min}

`AEDSS[3]` current maximum transition energy within this state-visit period

`AEDSS[4]` number of found maximum transitions energies (i.e. number of completed state-visit periods)

`AEDSS[5]` number of the end-state sampled in the last simulation step

`AEDSS[6]` should E_{min} be allowed to be smaller than the average energy of the end-state with the lowest energy?

`NEDS` number of EDS end-states

`AEDSS[7,1..NEDS]` energy offset parameter per EDS state

`AEDSS[8,1..NEDS]` natural logarithm of the exponentially averaged energy difference between the accelerated EDS reference state and this accelerated end-state

`AEDSS[9,1..NEDS]` free-energy difference between the accelerated EDS reference state and this accelerated end-state

`AEDSS[10,1..NEDS]` has this state already been visited within the current state-visit period?

`AEDSS[11,1..NEDS]` number of visits of this end-state

`AEDSS[12,1..NEDS]` average energy of this end-state

`AEDSS[13,1..NEDS]` average of the energy of this end-state minus the energy offset parameter of this end-state

`AEDSS[14,1..NEDS]` helper variable for the calculation of the standard deviation of the energy of this end-state

`AEDSS[15,1..NEDS]` standard deviation of the energy of this end-state

The accelerated EDS parameter search block is the following:

Accelerated EDS parameter search block
Blockname: AEDSS

Formatted form

```
     DO 10  N=1,6
10   WRITE  (unit,24) AEDSS[N]
     DO 11  N=1,NEDS
11   WRITE  (unit,42) AEDSS[7,N],AEDSS[8,N],AEDSS[9,N],AEDSS[10,N],AEDSS[11,N],
                      AEDSS[12,N],AEDSS[13,N],AEDSS[14,N],AEDSS[15,N]
24   FORMAT (3F15.9,3I15)
42   FORMAT (3F15.9,2I15,4F15.9)
```

4.20. Backwards compatibility with GROMOS96

The changes with respect to GROMOS96 have been detailed above. They consist of alterations in a number of blocks, introduction of a number of new blocks, and the deletion of a number of blocks.

For the alterations:

the GROMOS96 blocks	have been replaced by
BOX,	GENBOX,TRICLINICBOX
ENERGY,	ENERGIES,ENERY03
VOLUMEPRESSURE,	RUNDATA,VOLUMEPRESSURE03
FREEENERGYLAMBDA	FREELAMBDADATA,FREEENERGY-DERIVS03
and	and
FREENERGY3D4	FREE3D4DDATA respectively.

Additional blocks include:

SHAKEFAILPOSITION,	CONSFORCE,
SHAKEFAILPREVPOSITION,	CONSFORCERED,
LATTICESHIFTS,	NHCVARIABLES.
FREEFORCE,	
FREEFORCERED,	

Deleted blocks include:

BFACTORANISO,	QUANSUMENEAVER,
POSITIONSEDONDM,	QUANTIMESERIES,
POSITIONTHIRDM,	QUANDISTRIB,
POSITIONFOURTHM,	QUANTIMECORR and
POSITIONSECONDMT,	QUANTIMECORRSPE.
QUANTITYAVER,	
QUANENEAVER,	

CHAPTER 5

Molecular topology building blocks

5.1. Introduction

Most programs of GROMOS do require a molecular topology file containing the topological and force field data concerning the molecular system that is considered. Specifying a complete molecular topology for a large molecule like a protein is a tedious task. Long lists of atomic properties have to be typed. Therefore, GROMOS contains a program *make_top* that is able to generate a complete molecular topology from *molecular topology building blocks*, that is, molecules or parts of molecules like amino acid residues, nucleotides, etc., which are constituting the molecular system that is considered. The building blocks are linked in order to form the wanted molecular topology.

Linking of building blocks consisting of *separate*, non-covalently connected, *molecules* is straightforward. This will be discussed in Sec. 5.2 together with the content and format of a *molecular topology building block file*. The linking of *covalently connected building blocks* by make_top demands a set of rules to be satisfied by the molecular topology building blocks. These rules will be discussed in Sec. 5.3 to Sec. 5.5.

Reading a molecular topology building block file occurs in programs:

make_top, check_top.

Examples of molecular topology building block files are named:

*.mtb

5.2. Separate molecules

A molecular topology building block file contains two types of information.

1. information regarding all building blocks:

`FPEPSI`	$(4\pi\varepsilon_0)^{-1}$, ε_0 = permittivity of vacuum
`HBAR`	$\hbar = h/(2\pi), h$ = Planck's constant
`SPDL`	c = speed of light
`BOLTZ`	k_B = Boltzmann's constant

2. information specifying a building block:

`L`	sequence number of the solute building block in the molecular topology building block file; below, the *primary sequence number* of the residue or nucleotide in the protein or polynucleotide is denoted by M
`RNME[L]`	name of residue or nucleotide or molecule (at most 5 characters); these names are to be used in the input of make_top to select building blocks
`NMAT[L]`	number of atoms

`NLIN[L]` number of atoms for which the exclusions are given before the exclusions of the building block atoms themselves:
= 0 for a separate molecule
≠ 0 for a residue, nucleotide or monosaccharide unit

`ANM[1..NMAT[L], L]` atom names (at most 5 characters)

`IMCM[1..NMAT[L], L]`
integer mass type codes for selection of atomic masses

`ATOM[1..NMAT[L], L]`
atom sequence number

`NREP[L]` number of atoms that are replacing existing atoms at the beginning (NREP[L]>0) or at the end (NREP[L]<0) of a chain of building blocks

`IACM[1..NMAT[L], L]`
integer atom type codes for selection of van der Waals parameters

`CGM[1..NMAT[L], L]` atomic charges

`ICGM[1..NMAT[L], L]`
atomic charge group codes; the atoms forming a charge group must have sequential sequence numbers; the last atom of a charge group is denoted by ICGM=1, the others must have ICGM=0

`MAE[1..NMAT[L], L]` number of neighbours of atom i that are excluded from non-bonded interaction with atom i

`MSAE[1..MAE[ATOM],1..NMAT[L], L]`
atom sequence numbers j of the excluded neighbours of atom i; it is required that i<j and that the j's occur in ascending order. The exclusions of the last NLIN[L] atoms of the building block are not specified here, but handled in the next building block of the molecular chain.
Exclusions between different building blocks that do not go over the systems main chain (e.g. disulfide bridges, covalently bound Heme groups) are described in section Sec. 5.3.

`NCGB[L]` number of coarse-grained regions

`NRCGF[1..NCGB], NRCGL[1..NCGB]`
first and last atom sequence number of a coarse-grained region

`MSCAL[1..NCGB]` scaling factor for pressure correction of a coarse-grained region

`NPPOL[L]` number of polarisable solute atoms

`IPOLP[1..NPPOL], ALPP[1..NPPOL]`
atom sequence number and polarisability of the polarisable solute atom

`QPOLP[1..NPPOL]` size of charge-on-spring connected to polarisable solute atoms

`ENOTP[1..NPPOL], EPP[1..NPPOL]`
damping level and power for polarisation

`GAMP[1..NPPOL], IP[1..NPPOL], JP[1..NPPOL]`
gamma and the first and second atom for off-site polarisable centre construction

`NEX[L]` number of LJ-exceptions

`AEX[1..2, 1..NEX[L], L]`
: atom sequence numbers of the atoms to have special LJ-interactions defined by LJ-exceptions

`AEXTYPE[1..NEX[L], L]`
: LJ-exception type codes for selection of interaction parameters

`NMB[L]`
: number of bonds

`MB[1..2, 1..NMB[L], L]`
: atom sequence numbers of the atoms forming the bonds i-j, i is always smaller than j

`MCBL[1..NMB[L], L]`
: bond-type codes for selection of interaction parameters

`NMBDP[L]`
: number of dipole bonds

`MBDP[1..2, 1..NMBDP[L], L]`
: atom sequence numbers of the atoms forming the dipole bonds i-j, i is always smaller than j

`NMBA[L]`
: number of bond angles

`MBA[1..3, 1..NMBA[L], L]`
: atom sequence numbers of the atoms forming the bond angles i-j-k, i is always smaller than k

`MCBA[1..NMBA[L], L]`
: bond-angle type codes for selection of interaction parameters

`NMIDA[L]`
: number of improper dihedral angles

`MIDA[1..4, 1..NMIDA[L], L]`
: atom sequence numbers of the atoms forming the improper dihedrals i-j-k-l, j is always smaller than k

`MCIA[1..NMIDA[L], L]`
: improper dihedral angle type codes for selection of interaction parameters

`NMDA[L]`
: number of dihedral angles

`MDA[1..4, 1..NMDA[L], L]`
: atom sequence numbers of the atoms forming the dihedrals i-j-k-l, j is always smaller than k

`MCDA[1..NMDA[L], L]`
: dihedral angle type codes for selection of interaction parameters

`LL`
: sequence number of the solvent building block in the molecular topology building block file

`RNMES[LL]`
: name of solvent molecule (at most 5 characters): one of these names is used in the input of make_top to select a solvent building block

`NMATS[LL]`
: number of atoms

`ANMMS[1..NMATS[LL], LL]`
: atoms names (at most 5 characters)

`IMCMS[1..NMATS[LL], LL]`
: integer mass type codes for selection of atomic masses

`IACMS[1..NMATS[LL], LL]`
: integer atom type codes for selection of van der Waals parameters

`CGMS[1..NMATS[LL], LL]`
: atomic charges

`NVPOL[L]`
: number of polarisable solvent atoms

`IPOLV[1..NVPOL], ALPV[1..NVPOL]`
: atom sequence number and polarisability of the polarisable solvent atom

`QPOLV[1..NVPOL]`
: size of charge-on-spring connected to polarisable solvent atoms

`ENOTV[1..NPVOL], EPV[1..NVPOL]`
: damping level and power for polarisation

`GAMV[1..NVPOL], IV[1..NVPOL], JV[1..NVPOL]`
: gamma and the first and second atom for off-site polarisable centre construction

`NCONM[LL]`
: number of distance constraints

`ICONM, JCONM[1..NCONM[LL], LL]`
: atom sequence numbers of the atoms forming the constraint i-j, i is always smaller than j

`CONM[1..NCONM[LL], LL]`
: constraint length of the constraint i-j

The blocks of a *molecular topology building block file* are (apart from the *Title block*) the following:

Physical constants block
Blockname: PHYSICALCONSTANTS

```
     WRITE  (unit,12) FPEPSI
     WRITE  (unit,12) HBAR
     WRITE  (unit,12) SPDL
     WRITE  (unit,12) BOLTZ
12   FORMAT (E15.7)
```

Molecular topology solute building block
Blockname: MTBUILDBLSOLUTE

```
     WRITE  (unit,30) RNME[L]
     WRITE  (unit,33) NMAT[L], NLIN[L]
```

for every preceding exclusion I:

```
     DO 19  I=1, NLIN[L]
19   WRITE  (unit,31) I-NLIN[L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for atom I $<$ NMATL[L]-NLIN[L]:

```
     DO 20  I=1, NMAT[L]-NLIN[L]
20   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for every atom I with NMAT[L]-NLIN[L] $<$ I $\leqslant$ NMAT[L]:

```
     DO 21  I=1, NMAT[L]-NLIN[L] + 1, NMAT[L]
21   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L]
```

```
     WRITE  (unit,33) NLJEX[L]
```
for every LJ-exception N:
```
     DO 22  N=1, NLJEX[L]
22   WRITE  (unit,33) (AEX[M,N,L], M=1,2), AEXTYPE[N,L]

     WRITE  (unit,33) NMB[L]
```
for every bond N:
```
     DO 23  N=1, NMB[L]
23   WRITE  (unit,33) (MB[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
```
for every bond angle N:
```
     DO 24  N=1, NMBA[L]
24   WRITE  (unit,33) (MBA[M,N,L], M=1,3), MCBA[N,L]

     WRITE  (unit,33) NMIDA[L]
```
for every improper dihedral angle N:
```
     DO 25  N=1, NMIDA[L]
25   WRITE  (unit,33) (MIDA[M,N,L], M=1,4), MCIA[N,L]

     WRITE  (unit,33) NMDA[L]
```
for every proper torsional dihedral N:
```
     DO 26  N=1, NMDA[L]
26   WRITE  (unit,33) (MDA[M,N,L], M=1,4), MCDA[N,L]
30   FORMAT (A5)
31   FORMAT (I5,30X,I4,8I5)
32   FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33   FORMAT (16I5)
```

If MAE[I,L] $>$ 8, then the remaining MSAE values are written on the next line using 16I5 as format.

Molecular topology coarse-grained (CG) solute building block
Blockname: MTBUILDBLCGSOLUTE

```
     WRITE  (unit,30) RNME[L]
     WRITE  (unit,33) NMAT[L], NLIN[L]
```

for every preceding exclusion I:
```
     DO 19  I=1, NLIN[L]
19   WRITE  (unit,31) I-NLIN[L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for atom I $<$ NMATL[L]-NLIN[L]:
```
     DO 20  I=1, NMAT[L]-NLIN[L]
20   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for every atom I with NMAT[L]-NLIN[L] $<$ I $\leqslant$ NMAT[L]:
```
     DO 21  I=1, NMAT[L]-NLIN[L] + 1, NMAT[L]
21   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L]

     WRITE  (unit,33) NCGB[L]
```
for every coarse-grained region N:
```
     DO 22  N=1, NCGB[L]
```

```
22   WRITE  (unit,34) NRCGF[N], NRCGL[N], MSCAL

     WRITE  (unit,33) NLJEX[L]
```
for every LJ-exception N:
```
     DO 23  N=1, NLJEX[L]
23   WRITE  (unit,33) (AEX[M,N,L], M=1,2), AEXTYPE[N,L]

     WRITE  (unit,33) NMB[L]
```
for every dipole bond N:
```
     DO 24  N=1, NMBDP[L]
24   WRITE  (unit,33) (MBDP[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
```
for every bond N:
```
     DO 24  N=1, NMB[L]
24   WRITE  (unit,33) (MB[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
```
for every bond angle N:
```
     DO 25  N=1, NMBA[L]
25   WRITE  (unit,33) (MBA[M,N,L], M=1,3), MCBA[N,L]

     WRITE  (unit,33) NMIDA[L]
```
for every improper dihedral angle N:
```
     DO 26  N=1, NMIDA[L]
26   WRITE  (unit,33) (MIDA[M,N,L], M=1,4), MCIA[N,L]

     WRITE  (unit,33) NMDA[L]
```
for every proper torsional dihedral N:
```
     DO 27  N=1, NMDA[L]
27   WRITE  (unit,33) (MDA[M,N,L], M=1,4), MCDA[N,L]
30   FORMAT (A5)
31   FORMAT (I5,30X,I4,8I5)
32   FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33   FORMAT (16I5)
34   FORMAT (2I5,F15.7)
```

If MAE[I,L] > 8, then the remaining MSAE values are written on the next line using 16I5 as format.

Molecular topology polarisable solute building block
Blockname: MTBUILDBLPOLSOLUTE

```
     WRITE  (unit,30) RNME[L]
     WRITE  (unit,33) NMAT[L], NLIN[L]
```

for every preceding exclusion I:
```
     DO 19  I=1, NLIN[L]
19   WRITE  (unit,31) I-NLIN[L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for atom I $<$ NMATL[L]-NLIN[L]:
```
     DO 20  I=1, NMAT[L]-NLIN[L]
20   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
```

for every atom I with NMAT[L]-NLIN[L] $<$ I $\leqslant$ NMAT[L]:

```
     DO 21  I=1, NMAT[L]-NLIN[L] + 1, NMAT[L]
21   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L]

     WRITE  (unit,33) NCGB[L]
```
for every polarisable atom I:
```
     DO 22  I=1, NPPOL[L]
22   WRITE  (unit,35) IPOLP[I], ALPP[I], QPOLP[I], ENOTP[I], EPP[I], GAMP[I], IP[I], JP[I]

     WRITE  (unit,33) NLJEX[L]
```
for every LJ-exception N:
```
     DO 23  N=1, NLJEX[L]
23   WRITE  (unit,33) (AEX[M,N,L], M=1,2), AEXTYPE[N,L]

     WRITE  (unit,33) NMB[L]
```
for every bond N:
```
     DO 24  N=1, NMB[L]
24   WRITE  (unit,33) (MB[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
```
for every bond angle N:
```
     DO 25  N=1, NMBA[L]
25   WRITE  (unit,33) (MBA[M,N,L], M=1,3), MCBA[N,L]

     WRITE  (unit,33) NMIDA[L]
```
for every improper dihedral angle N:
```
     DO 26  N=1, NMIDA[L]
26   WRITE  (unit,33) (MIDA[M,N,L], M=1,4), MCIA[N,L]

     WRITE  (unit,33) NMDA[L]
```
for every proper torsional dihedral N:
```
     DO 27  N=1, NMDA[L]
27   WRITE  (unit,33) (MDA[M,N,L], M=1,4), MCDA[N,L]
30   FORMAT (A5)
31   FORMAT (I5,30X,I4,8I5)
32   FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33   FORMAT (16I5)
35   FORMAT (I5,5F11.5,2I5)
```

If MAE[I,L] $>$ 8, then the remaining MSAE values are written on the next line using 16I5 as format.

Molecular topology solute end group building block
Blockname: MTBUILDBLEND

```
     WRITE  (unit,30) RNME[L]
     WRITE  (unit,33) NMAT[L], NREP[L]
```

if NREP[L] $>$ 0 then
for atom I $\leqslant$ NMATL[L]-NREP[L]:
```
     DO 20  I=1, NMAT[L]-NREP[L]
20   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])
     DO 21  I=NMAT[L]-NREP[L]+1, NMAT[L]
21   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], ICMCM[I,L], CGM[I,L],
                      ICGM[I,L]
```

if NREP[L] < 0 then
for every atom I

```
     DO 22  I= NMAT[L]-NREP[L]+1, NMAT[L]
22   WRITE  (unit,32) I, ANM[I,L], IACM[I,L], IMCM[I,L], CGM[I,L],
                      ICGM[I,L], MAE[I,L], (MSAE[N,I,L],N=1,MAE[I,L)

     WRITE  (unit,33) NMB[L]
```

for every bond N:

```
     DO 23  N=1, NMB[L]
23   WRITE  (unit,33) (MB[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
```

for every bond angle N:

```
     DO 24  N=1, NMBA[L]
24   WRITE  (unit,33) (MBA[M,N,L], M=1,3), MCBA[N,L]

     WRITE  (unit,33) NMIDA[L]
```

for every improper dihedral angle N:

```
     DO 25  N=1, NMIDA[L]
25   WRITE  (unit,33) (MIDA[M,N,L], M=1,4), MCIA[N,L]

     WRITE  (unit,33) NMDA[L]
```

for every proper torsional dihedral N:

```
     DO 26  N=1, NMDA[L]
26   WRITE  (unit,33) (MDA[M,N,L], M=1,4), MCDA[N,L]
30   FORMAT (A5)
31   FORMAT (I5,30X,I4,8I5)
32   FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33   FORMAT (16I5)
```

If MAE[I,L] > 8, then the remaining MSAE values are written on the next line using 16I5 as format.

Molecular topology solvent building block
Blockname: MTBUILDBLSOLVENT

```
     WRITE  (unit,30) RNMES[LL]
     WRITE  (unit,33) NMATS[LL]
     DO 20  J=1, NMATS[LL]
20   WRITE  (unit,32) J, ANMMS[J,LL], IACMS[J,LL], IMCMS[J,LL],
                      CGMS[J,LL]
     WRITE  (unit,33) NCONM[LL]
     DO 21  N=1, NCONM[LL]
21   WRITE  (unit,34) ICONM[N,LL], JCONM[N,LL], CONM[N,LL]
30   FORMAT (A5)
32   FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33   FORMAT (16I5)

34   FORMAT (2I5,F15.7)
```

Molecular topology polarisable solvent building block
Blockname: MTBUILDBLPOLSOLVENT

```
      WRITE  (unit,30) RNMES[LL]
      WRITE  (unit,33) NMATS[LL]
      DO 20  J=1, NMATS[LL]
20    WRITE  (unit,32) J, ANMMS[J,LL], IACMS[J,LL], IMCMS[J,LL],
                       CGMS[J,LL]
      WRITE  (unit,33) NVPOL[LL]
      DO 21  I=1, NVPOL[LL]
21    WRITE  (unit,35) IPOLV[J], ALPV[J], QPOLV[J], ENOTV[J], EPV[J], GAMV[J], IV[J], JV[J]

      WRITE  (unit,33) NCONM[LL]
      DO 22  N=1, NCONM[LL]
22    WRITE  (unit,34) ICONM[N,LL], JCONM[N,LL], CONM[N,LL]
30    FORMAT (A5)
32    FORMAT (I5,1X,A5,I4,I5,F11.5,2I4,8I5)
33    FORMAT (16I5)

34    FORMAT (2I5,F15.7)
35    FORMAT (I5,5F11.5,2I5)
```

5.3. Linking of building blocks

When several building blocks need to be covalently linked to obtain the required molecular topology, a few rules must be satisfied compared to the case of separate molecules. These rules are due to the fact that in a chain of building blocks the bonds, bond-angles and (improper) dihedral angles involve atoms from *different* building blocks. Also excluded neighbours may reside in different building blocks. These rules are the following:

- When listing a bond (i-j), bond-angle (i-j-k), improper dihedral (i-j-k-l) or dihedral (i-j-k-l) connecting atoms with sequence numbers i, j, k or l in two residues with residue sequence numbers M-1 and M or M and M+1, through a *peptide C-N link*, the following rules apply:
  - for the bond i-j, neither i nor j may lie in residue M-1 and only j may lie in residue M+1;
  - for the bond-angle i-j-k, only i may lie in residue M-1, and only k may lie in residue M+1;
  - for the improper dihedral i-j-k-l, only j or k may lie in residue M-1, and only i or j or k or l may lie in residue M+1;
  - for the dihedral i-j-k-l, only i or i and j may lie in residue M-1. When i and j lie in residue M-1, atom i always refers to the first atom bound to j with i<j, it can be specified by -2. Only l may lie in residue M+1.

- Cross links such as disulfide bridges or the covalent link between a histidine and the heme group can be made between different building blocks. For a cross-link between building blocks M and N (M<N) the rules for listing the bond, bond-angles and dihedrals are the following:
  - In M, the atoms of the building block N are identified by a negative sign of atom sequence numbers
  - for the bond i-j, only in M -j may denote atom j in N
  - for the bond-angle i-j-k, only in M -k or -j and -k may denote atoms j and k in N
  - for the dihedral i-j-k-l, only in M -l, or -h and -l or -j, -k and -l may denote atoms j,k and l in N
  - for the improper dihedral i-j-k-l, only in M -i or -l may denote atoms i and l in N
  - In M, the excluded neighbours residing in N are denoted by a negative sign of their atom sequence numbers

- More general cross links between molecules can be made by post-processing an unlinked topology using GROMOS++ program link_top. The link is defined according to the MTBUILDBLLINK block outlined below. The rules for creating cross links with *link_top* are the following:
  - the atoms are identified in the original toplogy by the residue sequence number indicated in the input and the name of the atom specified in the MTBUILDBLLINK block

- all atoms for which the IAC is set to 0 will be removed from the original topology. For all remaining atoms specified in the MTBUILBLLINK block, the values for the IAC, the MASS, the CHARGE and the charge group code are updated. Note that the actual MASS is given in the MTBUILDBLLINK file and not the integer mass type code
- the exlusions of the original topology (without the removed atom) remain, and exclusions that are specified in the MTBUILDBLLINK block are *added*
- all covalent interactions that are specified in the MTBUILDBLLINK block are first removed from the original topology (if present) and subsequently added according to the current definitions.
- for dihedral angles, *link_top* allows the user to refer to the first and/or last atom with an atom sequence number 0. For these atoms, the program will search in the topology that is bound to the second or third atom in the dihedral angle definition, respectively, and assign the dihedral angle to this atom. Any dihedral angles that were already defined for this group is replced.

Molecular topology solute building block
Blockname: MTBUILDBLLINK

```
     WRITE  (unit,30) RNME[L]
     WRITE  (unit,33) NMAT[L]

     DO 20  I=1, NMAT[L]
20   WRITE  (unit,32) I, RES[I,L], ANM[I,L], IACM[I,L], MASS[I,L],
                      CGM[I,L],ICGM[I,L], MAE[I,L],(MSAE[N,I,L],N=1,MAE[I,L])

     DO 23  N=1, NMB[L]
23   WRITE  (unit,33) (MB[M,N,L], M=1,2), MCBL[N,L]

     WRITE  (unit,33) NMBA[L]
     DO 24  N=1, NMBA[L]
24   WRITE  (unit,33) (MBA[M,N,L], M=1,3), MCBA[N,L]

     WRITE  (unit,33) NMIDA[L]
     DO 25  N=1, NMIDA[L]
25   WRITE  (unit,33) (MIDA[M,N,L], M=1,4), MCIA[N,L]

     WRITE  (unit,33) NMDA[L]
     DO 26  N=1, NMDA[L]
26   WRITE  (unit,33) (MDA[M,N,L], M=1,4), MCDA[N,L]
30   FORMAT (A5)
31   FORMAT (I5,30X,I4,8I5)
32   FORMAT (I5,1X,I5,A5,I4,F10.5,F11.5,2I4,8I5)
33   FORMAT (16I5)
```

An example of a molecular topology building block containing an amino acid residue is the Alanine building block to be found under the name ALA in the *.mtb files.

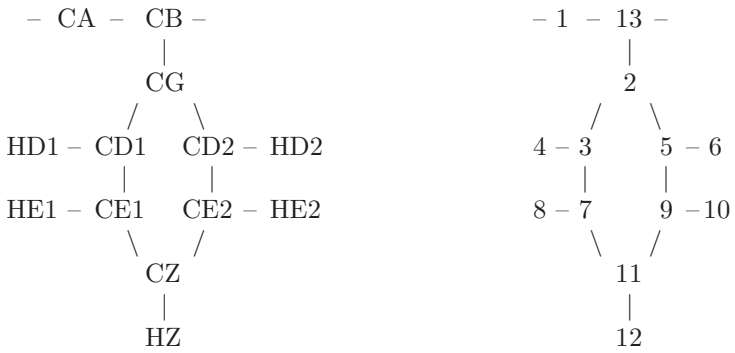
An example of a molecular topology building block containing a nucleotide is the Adenosine building block to be found under the name DADE in the *.mtb files.

An example of a molecular topology building block containing a glucose unit is the sugar building block to be found under the name GLCA in the *.mtb files.

5.4. Other building blocks

From the previous paragraphs it has become clear that program *make_top* may link any type of molecular topology building blocks into a linear co-valently connected chain, as long as the characteristics of the link

satisfy the given rules. A Styrene residue topology building block may serve as an example.



By choosing the displayed atom sequence numbers, all the rules for connecting these building blocks into a polystyrene chain using *make_top* are satisfied.

It should always be checked in the complete molecular topology generated from the building blocks by program *make_top* whether the linking has been carried out correctly.

5.5. End groups

The linking of molecular topology building blocks has been described in the previous sections. This leaves open the question of how to treat the head and tail of the molecular chain one is interested in. Often the bonded and nonbonded parameters at the head and tail of e.g. a protein will be different from the parameters that are usually describing amino acid interactions. For this reason GROMOS knows the end-group building blocks, which describe which atoms to change and add or remove at the head and tail of the sequence. The following rules apply:

- if NREP > 0, the end-group building block describes the head of the chain. The last NREP atoms will replace the first NREP atoms in the next building block. Only the exclusions that are specified remain from the next building block, all other parameters are overwritten.
- if NREP < 0, the end-group building block describes the tail of the chain. The first -NREP atoms will replace the last -NREP atoms in the previous building block.
- in order to remove atoms from the next or previous building block they need to be specified in the end-group building block as a replacing atom with a negative IAC
- for all covalent interactions that cross between an end-group and a regular building block the same rules apply as for linking two building blocks.

Examples of protein end-group building blocks can be found under the names NH3+ or COOH in the *.mtb files.

Examples of nucleotide end-group building blocks can be found under the names D3OH and D5OH in the *.mtb files.

Examples of saccharide end-group building blocks can be found under the names C1OH and C6OH in the *.mtb files.

It should always be checked in the complete molecular topology generated from the building blocks by program make_top whether the end-groups have been implemented correctly.

5.6. Contents of the MTB file

MTB file:

TITLE
LINKEXCLUSIONS
PHYSICALCONSTANTS
MTBUILDBLSOLUTE
MTBUILDBLSOLVENT
MTBUILDBLEND

CHAPTER 6

Interaction function parameters

6.1. Introduction

The molecular topology file of a molecular system does not only contain topological information about the system, but also force field parameters. These parameters have been listed in Chap. 3 (Force Field and Topology Data Set). They are part of a molecular topology as described in Sec. 3.2. The molecular topology can be generated using the program *make_top*. All the force field parameters that belong to a specific force field are kept in two different files. The force field parameters that are related to the molecular topology, like atomic charges and third or excluded nearest neighbour information, are included in the *molecular topology building block file*, which has been described in Sec. 5.2. The remaining force field parameters, which are independent of the molecular topology, are kept in another file, the *interaction function parameter file*. Both files are combined by program *make_top* to generate a complete molecular topology file (Sec. 3.2) corresponding to the molecular system that is considered.

The various blocks of an interaction function parameter file are (apart from the *Title block*) described in CSec. 6.2 to Sec. 6.7.

Reading an interaction parameter file occurs in programs:

make_top, check_top, con_top

Examples of interaction function parameter files are named:

*.ifp

6.2. Mass atom types

The mass atom type codes, mass values and names are stored as follows:

`NRMATY`	number of defined mass atom types
`NMATY`	largest (integer) mass atom type code
`ATMAS[1..NMATY]`	atomic mass as a function of the (integer) mass atom type code
`ATMASN [1..NMATY]`	(mass) atom names as a function of the (integer) mass atom type code (at most 5 characters)

Mass atom type code block
Blockname: MASSATOMTYPECODE

```
     WRITE  (unit,11) NRMATY, NMATY
     DO 10  M=1, NRMATY
10   WRITE  (unit,12) N, ATMAS[N], ATMASN[N]
11   FORMAT (2I5)
12   FORMAT (I5,F10.5,1X,A5)
```

6.3. Covalent bond-stretching interaction parameters

The parameters concerning the bond-stretching interaction are stored as follows:

`NRBTY`	number of defined covalent bond types
`NBTY`	largest (integer) bond-type code
`CB[1..NBTY]`	force constant of the bond-stretching term of the interaction as a function of the bond-type code, based on a quartic potential energy function
`CHB [1..NTBY]`	force constant of the bond-stretching term of the interaction as a function of the bond-type code, based on a harmonic potential energy function
`B0[1..NBTY]`	bond length at minimum energy of the bond-stretching term as a function of the bond-type code

Bond-type code and parameters block
Blockname: BONDSTRETCHTYPECODE

```
     WRITE  (unit,11) NRBTY, NBTY
     DO 10  M=1, NRBTY
10   WRITE  (unit,13) N, CB[N], CHB [N], B0[N]
13   FORMAT (I5,3F15.7)
```

6.4. Covalent bond-angle bending interaction parameters

The parameters concerning the bond-angle bending interaction are stored as follows:

`NRTTY`	number of defined bond-angle types
`NTTY`	largest (integer) bond-angle type code
`CT[1..NTTY]`	force constant of the bond-angle bending term of the interaction as a function of the bond-angle type code, based on a potential energy function harmonic in the angle cosine
`CHT[1..NTTY]`	force constant of the bond-angle bending term of the interaction as a function of the bond-angle type code, based on a potential energy function harmonic in the angle (in energy units per degree^2)
`T0[1..NTTY]`	bond angle (in degrees) at minimum energy of the bond-angle bending term as a function of the bond-angle type code

Bond-angle type code and parameters block
Blockname: BONDANGLEBENDTYPECODE

```
     WRITE  (unit,11) NRTTY, NTTY
     DO 10  M=1, NRTTY
10   WRITE  (unit,13) N, CT[N], CHT [N], T0[N]
13   FORMAT (I5,3F15.7)
```

6.5. Improper dihedral-angle interaction parameters

The parameters concerning the harmonic improper dihedral-angle interaction are stored as follows:

NRQTY	number of defined (harmonic) improper dihedral-angle types
NQTY	largest (integer) improper dihedral-angle type code
CQ[1..NQTY]	force constant of the harmonic improper dihedral term of the interaction as a function of the improper dihedral-angle type code (in energy units per degree^2)
Q0[1..NQTY]	improper dihedral-angle (in degrees) at minimum energy of the harmonic improper dihedral term as a function of the improper dihedral-angle type code

Improper (harmonic) dihedral-angle type code and parameters block
Blockname: IMPDIHEDRALTYPECODE

```
     WRITE  (unit,11) NRQTY, NQTY
     DO 10  M=1, NRQTY
10   WRITE  (unit,13) N, CQ[N], Q0[N]
13   FORMAT (I5,3F15.7)
```

6.6. Dihedral-angle torsional interaction parameters

The parameters concerning the trigonometric dihedral-angle interaction are stored as follows:

NRPTY	number of defined (trigonometric) dihedral-angle types
NPTY	largest (integer) dihedral-angle type code
CP[1..NPTY]	force constant of the trigonometric dihedral term of the interaction as a function of the dihedral-angle type code
PDL [1..NPTY]	phase-shift angle (in degrees) of the trigonometric dihedral term as a function of the dihedral-angle type code
NP[1..NPTY]	multiplicity of the trigonometric dihedral term as a function of the dihedral-angle type code (1,2,3,4,5,6)

Proper (trigonometric) dihedral-angle type code and parameters block
Blockname: TORSDIHEDRALTYPECODE

```
     WRITE  (unit,11) NRPTY, NPTY
     DO 10  M=1, NRPTY
10   WRITE  (unit,14) N, CP[N], PDL[N], NP[N]
14   FORMAT (I5,2F10.5,I5)
```

6.7. Van der Waals interaction parameters and integer atom codes

The interaction function parameter file contains information on the van der Waals interaction parameters. These are C_{12}(i,j), the coefficient of the $1/r^{12}$ term, and C_6(i,j), the coefficient of the $-1/r^6$ term in the non-bonded interaction. These coefficients depend on the integer atom codes I and J (1..NRATT) of atoms i and j. In a molecular topology file these parameters are stored in the arrays C12, C6[1..NRATT*(NRATT+1)/2]. The corresponding parameters for the 1-4 or third neighbour non-bonded interaction are stored in arrays CS12, CS6 [1..NRATT* (NRATT+1)/2].

If NRATT is large, direct specification of all these parameters becomes tedious. Therefore, the information on the van der Waals parameters is stored in a different manner.

1. A first block contains single atom type normal van der Waals parameters $C_6^{1/2}$ (I,I) and (maximally 3 values) $C_{12}^{1/2}$ (I,I) and third-neighbour parameters $C_6^{1/2}$ (I,I) and $C_{12}^{1/2}$ (I,I), from which the van der Waals parameters for all atom pairs are calculated (in *make_top*) using geometric combination rules.
2. A second block contains van der Waals parameters for a given set of atom pairs, which will replace the combination rule values (upon reading in *make_top*).

The following variables are used to define the van der Waals interaction parameters:

`NRATT` number of (van der Waals) atom types

`TYPE[1..NRATT]` names of the different atom types as a function of the integer atom code that defines an atom type (at most 5 characters)

`C612[1..NRATT]` square root of the single atom coefficient of the $-1/r^6$ term in the normal van der Waals interaction as a function of the integer atom code

`C1212[1..NRATT, 1..3]` three values for the square root of the single atom coefficient of the $1/r^{12}$ term in the normal van der Waals interaction as a function of the integer atom code

`LPAIR[1..NRATT, 1..NRATT]` pointer matrix for selection of one of the three C1212 values when applying the combination rules: C12 [I,J] = C1212[I, LPAIR[I,J] * C1212[J, LPAIR[J,I]]; LPAIR[I,J] = 1,2 or 3

`CS612[1..NRATT]` square root of the single atom coefficient of the $-1/r^6$ term in the third-neighbour van der Waals interaction as a function of the integer atom code

`CS1212[1..NRATT]` square root of the single atom coefficient of the $1/r^{12}$ term in the third-neighbour van der Waals interaction as a function of the integer atom code

`NRPAIR` number of atom type pairs for which the van der Waals parameters are explicitly given

`MPAC[1..NRATT, 1..NRATT]` pair codes for atom pairs as a function of their integer atom codes I and J ($\leqslant$NRATT * (NRATT+1)/2 = NRATT2), the pair code is defined as I+J*(J-1)/2 when I$\leqslant$J and as J+I*(I-1)/2 when J$\leqslant$I; these values are not stored in the interaction function parameter file, since they can be and are calculated upon reading the file

`C12[1..NRATT*(NRATT+1)/2]` coefficient of the $1/r^{12}$ term in the non-bonded interaction as a function of the occurring pair codes; so, the sequence of atom pairs with integer atom codes ranging from 1 to NRATT is: 1-1, 1-2, 2-2, ...1-NRATT, ...2-NRATT, ... NRATT-NRATT

`C6[1..NRATT*(NRATT+1)/2]` coefficient of the $-1/r^6$ term in the non-bonded interaction as a function of the occurring pair codes

`CS12[1..NRATT*(NRATT+1)/2]` coefficient of the $1/r^{12}$ term in the 1-4 non-bonded interaction between third-neighbour atoms as a function of the occurring pair codes

`CS6[1..NRATT*(NRATT+1)/2]` coefficient of the $-1/r^6$ term in the 1-4 non-bonded interaction between third-neighbour atoms as a function of the occurring pair codes

Single atom type van der Waals (Lennard-Jones) parameters block
Blockname: SINGLEATOMLJPAIR

```
     WRITE  (unit,11) NRATT
     DO 10  M=1, NRATT
     WRITE  (unit,15) I, TYPE[I], C612[I], (C1212[I,K], K=1,3),
     WRITE  (unit,16) CS612[I], CS1212[I]
10   WRITE  (unit,17) (LPAIR[I,K], K=1, NRATT)
15   FORMAT (I5,1X,A5,4E15.7)
16   FORMAT (11X,2E15.7)
17   FORMAT (20I2)
```

If NRATT $>$ 20, the LPAIR values are written with 20 entries on each line using 20I2 as format.

Mixed atom type van der Waals (Lennard-Jones) parameters block
Blockname: MIXEDATOMLJPAIR

```
     WRITE  (unit,11) NRPAIR
     DO 10  M=1, NRPAIR
     WRITE  (unit,18) I, J, C6[I,J], C12[I,J], CS6[I,J], CS12[I,J]
18   FORMAT (2I5, 4E15.7)
```

The MIXEDATOMLJPAIR block must occur after the SINGLEATOMLJPAIR block on the interaction function parameter file.

GROMOS integer atom codes, single atom type van der Waals parameters for normal and third-neighbour interactions, and mixed atom type van der Waals parameters are given in Vol. 3.

Special atom pair based van der Waals interactions (LJ-exceptions) parameters block
Blockname: LJEXCEPTIONTYPE

```
     WRITE  (unit,11) NLJEXTYPE
     DO 10  M=1, NLJEXTYPE
     WRITE  (unit,18) M, LJEXC12[M], LJEXC6[M]
18   FORMAT (1I5, 2E15.7)
```

6.8. Atomic charges and charge group codes

The atomic charges and the charge group codes are to be specified with the atoms in the molecular topology building blocks, in the molecular topology building block file. This is discussed in Sec. 5.2.

6.9. Excluded neighbours

The information about which atoms j will be excluded from non-bonded interaction with atom i based on the proximity of atom i and j measured along the covalently bound chain (nearest neighbours), is to be specified with the atomic information in the molecular topology building blocks in the molecular topology building block file. This is discussed in Sec. 5.2.

6.10. Contents of the IFP file

IFP file:

TITLE
MASSATOMTYPECODE
BONDSTRETCHTYPECODE
BONDANGLEBENDTYPECODE
IMPDIHEDRALTYPEC

DIHEDRALTYPECODE
SINGLEATOMLJPAIR
MIXEDATOMLJPAIR

CHAPTER 7

Library files for GROMOS++

7.1. Introduction

The pre- and post-processing programs of GROMOS++ that are described in Chap. 5 make use of different additional library files that are described in the following sections.

7.2. Interaction function parameter renumbering

Several parameter sets of the GROMOS force field are available. Program `con_top` (see Sec. 5-2.6) is able to convert existing topologies to a different parameter set. From parameter set 45A4 to 53A5 all interaction parameter types have been renumbered. To convert topologies that were generated with a parameter set older than 53A5 to the new numbering a renumber file needs to be specified.

The renumbering information is stored as follows:

`BTFROM`	bond-stretch parameter type in original topology
`BTTO`	bond-stretch parameter type in resulting topology
`ATFROM`	bond-angle bend parameter type in original topology
`ATTO`	bond-angle bend parameter type in resulting topology
`IDTFROM`	improper (harmonic) dihedral angle parameter type in original topology
`IDTTO`	improper (harmonic) dihedral angle parameter type in resulting topology
`DTFROM`	dihedral (trigonometric) angle parameter type in original topology
`DTTO`	dihedral (trigonometric) angle parameter type in resulting topology
`ATOMFROM`	Lennard-Jones interaction type code (IAC) in original topology
`ATOMTO`	Lennard-Jones interaction type code (IAC) in resulting topology

The renumber-file contains the following blocks (apart from the title):

Bondtype conversion block
Blockname: BONDTYPECONV

```
     DO 10  N=1, NBT
10   WRITE  (unit,11) BTFROM, BTTO
11   FORMAT (2I5)
```

Bond-angle bend conversion block
Blockname: ANGLETYPECONV

```
     DO 10  N=1, NAT
10   WRITE  (unit,11) ATFROM, ATTO
```

Improper dihedral conversion block
Blockname IMPROPERTYPECONV

```
     DO 10  N=1, NIMP
10   WRITE  (unit,11) IDTFROM, IDTTO
```

Dihedral angle conversion block
Blockname: DIHEDRALTYPECONV

```
     DO 10  N=1, NDIH
10   WRITE  (unit,11) DTFROM, DTTO
```

Atomtype conversion block
Blockname: ATOMTYPECONV

```
     DO 10  N=1, NATOM
10   WRITE  (unit,11) ATOMFROM, ATOMTO
```

An example of a force field renumber file is ren45a4_to_53a5.dat.

7.3. Atomic naming conventions

Program `pdb2g96` can be used to convert molecular coordinate files in pdb-format to GROMOS-format (see Sec. 5-2.19). This program matches residue and atom names in the pdb-file with the names of residues and atoms specified in the molecular topology of the system. For proteins and nucleotides, the names by which residues or nucleotides and atoms are denoted will correspond exactly in the two files. However, some often occurring differences are known. These can be defined in the pdb2g96-library file which is to be specified when using the `pdb2g96` program.

The library file is defined by the following variables:

`RESPDB`	The name of a residue that is encountered in a pdb-file
`RESTOPO`	The name by which the corresponding residue is denoted in the topology RESAT The name of a residue for which an atom name difference is listed, according to the topology.
`ATMPDB`	The name of an atom in residue RESAT as it may be encountered in a pdb-file
`ATMTOPO`	The name of the corresponding atom in residue RESAT will be denoted in the topology

Apart from the title block, the pdb2g96 library file contains the following blocks:

Residue name block
Blockname: RESIDUENAMELIB

```
     DO 10 N=1, NRES
10   WRITE  (unit,11) RESPDB, RESTOPO
11   FORMAT (3A6)
```

Atomic name block
Blockname: ATOMNAMELIB

```
     DO 10  N=1, NATOM
10   WRITE  (unit,11) RESAT, ATMPDB, ATMTOPO
```

An example of this library file is pdb2g96.lib.

7.4. Definition of file-names and joblists

Program `mk_script` can generate jobscripts and input files to run MD++ (see Sec. 5-2.18). Although there are recommended file-names for the different GROMOS files that are used in a molecular simulation, there is no requirement to use these names. Program `mk_script` can generate names for files according to user-defined rules, that use the simulation time or a simulation sequence number. The rules to define these files are given in a `mk_script` template file. Some additional string constants to be used in the scripts can also be defined in the template file. The following types are recognized:

`script`	A rule to define the scriptname
`qinput`	A rule to define the name of a MD++ input file
`output`	A rule to define the name of a MD++ output file
`coord`	A rule to define the name of a single structure coordinate file
`pttopo`	A rule to define the name of a perturbation topology
`refpos`	A rule to define the name of a reference position coordinate file
`posresspec`	A rule to define the name of a position restraints specification file
`disres`	A rule to define the name of an atom-atom distance restraint file
`dihres`	A rule to define the name of a dihedral angle restraining file
`jvalue`	A rule to define the name of a ^3J-value restraints specification file
`ledih`	A rule to define the name of a local-elevation specification file
`outtrx`	A rule to define the name of a molecular coordinate trajectory file
`outtrv`	A rule to define the name of a molecular velocity trajectory file
`outtre`	A rule to define the name of an energy trajectory file
`outbae`	A rule to define the name of a block-averaged energy trajectory file
`outtrg`	A rule to define the name of a free energy trajectory file
`outbag`	A rule to define the name of a block-averaged free energy trajectory file
`workdir`	A rule to define a directory where the simulation can be run locally
`mpicommand`	A rule to define the command used to run in an MPI parallel environment
`firstcommand`	A rule to define an initial command that needs to be performed before the call to MD++
`lastcommand`	A rule to define a final command that is to be called when the simulation script finishes

The variables to store the rules are as follows:

`FILETYPE`	The type of a file for which the rule will be specified.

FILENAME	The rule to form the corresponding file name
MISCTYPE	The type of another stringconstant that needs to be formed based on the simulation time or sequence number
MISCNAME	The rule to form the corresponding stringconstant

The template file is built up with the following blocks (apart from the title block):

Filename specification block
Blockname: FILENAMES

```
     DO 10  N=1, NSPEC
10   WRITE  (unit,11) FILETYPE, FILENAME
11   FORMAT (A20, A60)
```

Miscellaneous specification block
Blockname: MISCELLANEOUS

```
     DO 10  N=1, NSPEC
10   WRITE  (unit,11) MISCTYPE, MISCNAME
```

An example of a template file is mkscript.lib.

Program `mk_script` can not only write a single script with the appropriate naming conventions for the files that are involved, but it can also generate a consistent set of simulations that perform a specific task. This is done by specifying a joblist in which specific variable of the input file can be given different values from simulation script to simulation script.

A joblist is specified by the following variables:

NVAR	The number of variables that are to be modified between scripts
NSCRIPTS	The number of scripts and input files that will be written
VARID[1..NVAR+3]	An identifying string for every variable. The value of VARID[1] is required to be "job_id", the value of VARID[NVAR+2] to be "subdir" and the value of VARID[NVAR+3] to be "run_after".
VARVAL[1..NVAR+3,1..NSCRIPTS]	The value of the specified variables in the input files that are to be generated. VARVAL[1,1..NSCRIPTS] contains the job-sequence number by which the scripts can be identified. VARVAL[NVAL+2,1..NSCRIPTS] contains a string constant that refers to the subdirectory where the simulations will be run and VARVAL[NVAL+3, 1..NSCRIPTS] specifies which script-id should end with a call (or submission) of this script (defined through variable lastcommand in the template file).

Joblist specification file
Blockname: JOBSCRIPTS

```
     WRITE  (unit, 11)(VARID[N], N=1, NVAR)
     DO 10  M=1, NSCRIPTS
10   WRITE  (unit, 11)VARVAL[M,N], N=1, NVAR)
11   FORMAT (20A10)
```

Examples for joblist specification files are joblist.startup and joblist.perturbation.

7.5. Energy trajectory block definition

Energy and free energy data is written at a user specified interval to (free) energy trajectory files. MD++ can also write block averaged trajectory files. Program `ene_ana` can be used to extract time series of properties derived from variables stored in these trajectory files (see Sec. 5-4.21). `ene_ana` has been written such that is can read any block-based trajectory file. The block format is specified in a library file that can be modified by the user. In addition, this library file contains definitions to calculate properties from the values that are stored in the trajectory files.

The format of the library file depends on the format of the free energy trajectory it is defining. It contains three blocks, ENERTRJ, FRENERTRJ and VARIABLES. The blocks ENERTRJ and FRENERTRJ define the format of the energy trajectory and the free energy trajectory respectively. Every line in this block contains one entry, with a first keyword specifying what kind of entry this is. The following keywords are recognized:

`block`	followed by a block name. For every configuration that is written to the trajectory file, program `ene_ana` will try to read these blocks. The entries in the library file that follow on subsequent lines specify the expected format of the block.
`size`	followed by a variable name. This entry tells `ene_ana` that it should read an integer number and store this in the specified name. This number can subsequently be used in the definition of arraysizes
`subblock`	followed by a variable name and two dimensions. This entry tells `ene_ana` to read a block of data of the specified dimensions and store the data under the specified name. The dimensions can be either specified by an integer number, by a previously defined size-variable or by such a variable preceded by the word "matrix_". For example, if the size-variable NEGR was defined previously, the dimension specification "matrix_NEGR" will expand to the value NEGR*(NEGR+1)/2.

In the VARIABLES block properties can be specified based on the data that was read in from the trajectory files. A new property is defined if the second word on a line consists of the character '='. All string constants read in until the next definition of a property will considered to be part of the same definition. The raw data read in from the energy trajectory files are referred to by the name of the subblocks followed by rectangular brackets to indicate the individual elements of the arrays. If the second dimension of a subblock is one, the second set of brackets may be omitted. Properties can be defined using subblock names, the characters +, -, *, /, (,) and any properties that were previously defined.

An examples of an energy trajectory specification file is ene_ana.md++.lib

7.6. Hydrogen-bond donors and acceptors

Program `hbond` monitors the presence of hydrogen bonds throughout a simulation (see Sec. 5-4.32). The explicit hydrogen atoms and H-bond acceptor atoms to be monitored can be specified individually, or can be obtained by applying a mass-filter on a larger set of specified atoms. The definition of a mass representing a hydrogen atom and possibly H-bond acceptors is done through a massfile, which is defined by the following variables

`NHMASS`	Number of masses that represent hydrogen atoms
`HMASS[1..NHMASS]`	Mass representative for hydrogen atoms

`NACCMASS` Number of masses that represent H-bond acceptor atoms

`ACCMASS[1..NACCMASS]`
Mass representative for H-bond acceptor atoms

This information is stored in the following blocks

Hydrogenmass block
Blockname: HYDROGENMASS

```
     DO 10  N=1, NHMASS
10   WRITE  (unit,11) HMASS[N]
11   FORMAT (F15.7)
```

Acceptormass block
Blockname: ACCEPTORMASS

```
     DO 10  N=1, NACCMASS
10   WRITE  (unit,11) ACCMASS[N]
```

An example of a massfile is hbond.massfile.

7.7. Crystallographic transformations

Program `cry` can construct crystallographic unit cells by applying the appropriate symmetry transformations on a given molecular structure (see Sec. 5-2.8). The symmetry transformations are specified by a rotation matrix, M, and a translation vector, V in a specification file. The following variables are required

`NSOP` Number of symmetry transformations that are defined

`M[1..3,1..3,1..NSOP]`
Rotation matrix M for every transformation

`V[1..3, 1..NSOP]` Translation vector V for every transformation

These variables are stored in the following block:

Symmetry transformation block
Blockname: TRANSFORM

```
     WRITE  (unit,20) NSOP
     DO 10  I=1, NSOP
10   DO 11  J=1,3
11   WRITE  (unit,21) M[1,J,I], M[2,J,I], M[3,J,I], V[J,I]
20   FORMAT (I5)
21   FORMAT (3F11.5,4X,F11.5)
```

An example of a transformation file is cry.spec.

7.8. NOE analysis

The programs `prep_noe` (Sec. 5-4.45), `noe` (Sec. 5-4.41) and `post_noe` (Sec. 5-4.42) analyse a trajectory for atom-atom distances and compare to experimentally determined upper-bounds to such distances. The NOE's are specified using virtual and pseudo-atoms as described in section Sec. 3.4. Program `prep_noe` can generate this NOE specification file from a list of proton-proton distances and a library file. Corrections to the experimentally determined upper-bounds for pseudo-atoms and multiplicities are defined in a correction file.

The proton-proton distances can be specified using a XPLOR like NOE specification file, which can be easily generated from e.g. an XPLOR-format. This format usually uses three distances, from which the upper- and lower-bounds for the atom-atom distances can be calculated. It uses the following variables:

`NNOE`	Number of NOE distances specified
`SEQN`	Sequential NOE number starting from 1 to NNOE
`RESI[1..NNOE]`	Residue number of atom I of the NOE distance
`NAMEI[1..NNOE]`	Atom name of atom I of the NOE distance
`RESJ[1..NNOE]`	Residue number of atom J of the NOE distance
`NAMEJ[1..NNOE]`	Atom name of atom J of the NOE distance
`ANOE[1..NNOE]`	XPLOR distance 1
`BNOE[1..NNOE]`	XPLOR distance 2
`CNOE[1..NNOE]`	XPLOR distance 3
`NUMAMB[1..NNOE]`	Number of ambiguous NOEs to which this NOE is linked
`AMBNOE[1..NOE,1..NUMAMB]`	SEQN of NOEs to which this NOE is linked

XPLOR like NOE specification block
Blockname: NOESPEC

```
     DO 10  N=1, NNOE
10   WRITE  (unit,11) SEQN[N], RESI[N], NAMEI[N], RESJ[N], NAMEJ[N],
                      ANOE[N], BNOE[N], CNOE[N], SEQN[N], NUMAMB[N],
                      AMBNOE[N,1], AMBNOE[N,2], ...
11   FORMAT (2I5,A5,I5,A5,3F8.3,I4,I4,10I4)
```

An example of a XPLOR like NOE specification file can be found in examples/prep.noe.

For unambiguous NOEs, only the first eight columns of this file are to be specified. For ambiguous restraints, the 9th column repeats the number of the NOE (first column), the 10th column contains the number of NOEs this NOE may be linked to and the remaining columns lists the numbers of the NOEs to which it is linked.

An NOE library file determines what type of virtual or pseudo-atom needs to be used to represent the proton-proton distances. The NOE library is defined by the following variables:

`NNLIB`	Number of entries in the library file
`RSNM[1..NNLIB]`	Residue name of for the atom

PNMIU[1..NNLIB]	IUPAC name of the proton (replace * with @)
CANM[1..NNLIB]	Name of the central atom of a virtual or pseudo atom description.
PATP[1..NNLIB]	Explicit, virtual or pseudo-atom type as described in section Sec. 3.4.

NOE library block
Blockname: NOELIB

```
     DO 10  N=1, NNLIB
10   WRITE  (unit,11) RSNM[N], PNMIU[N], CANM[N], PATP[N]
11   FORMAT (3A6,I5)
```

Examples of NOE library files are noelib.45a3 and noelib.53a6.

Virtual and pseudo atoms may require corrections to the upper bounds due to the position of the atom or the multiplicity of the signal. Program `prep_noe` can either add or remove such correction from a given set of distances. The corrections are defined in a NOE correction file, which contains the following variables:

NPAC	Number of pseudo-atom corrections in the file
NSPAC	NOE suptype to which the pseudo-atom correction applies (set to 0 if no subtype defined)
NTPAC[1..NPAC]	NOE type to which the pseudo-atom correction applies
FTPAC[1..NPAC]	Distance of the pseudo-atom correction
NMPC	Number of multiplicity corrections in the file
NTMPC[1..NPAC]	NOE type to which the multiplicity correction applies
FTMPC[1..NPAC]	Factor for the multiplicity correction

This information is written in the following blocks:

Pseudo atom correction block
Blockname: NOECORGROMOS

```
     DO 10  N=1, NPAC
10   WRITE  (unit, 11)NTPAC[N], NSPAC[N], FTPAC[N]
11   FORMAT (I5, F15.8)
```

Multiplicity correction block
Blockname: MULTIPLICITY

```
     DO 10  N=1, NMPC
10   WRITE  (unit,11) NTMPC[N], NSPAC[N], FTMPC[N]
```

Examples of NOE correction files are called noecor.*.

The program `prep_noe` generates the NOE specification file which can be used as input for programs `noe` and `post_noe`. A NOE distance in the NOE specification file is characterised by the following quantities:

DISH	carbon-hydrogen distance

DISC carbon-carbon distance

NNOE Number of NOE distances specified

IDR1, JDR1, KDR1, LDR1 [1..NNOE]
atom sequence numbers of the real atoms defining the geometric position of the first atom of a NOE distance pair

ICDR1 [1..NNOE] geometric code defining the position of the first atom of a distance restraint pair [-2, -1, ..., 7] (see Sec. 3.4)

VACS1 [1..NNOE] subtype of first virtual atom. Possible subtypes are:
0: no subtype defined (for ICDR1 = -2, 0-7)
1: aromatic flipping ring (for ICDR1 = -1)
2: non-stereospecific NH2 group (for ICDR1 = -1)

IDR2, JDR2, KDR2, LDR2 [1..NNOE]
atom sequence numbers of the real atoms defining the geometric position of the second atom of a NOE distance pair

ICDR2 [1..NNOE] geometric code defining the position of the second atom of a distance restraint pair [-2, -1, ..., 7] (see Sec. 3.4)

VACS2 [1..NNOE] subtype of second virtual atom. Possible subtypes are:
0: no subtype defined (for ICDR2 = -2, 0-7)
1: aromatic flipping ring (for ICDR2 = -1)
2: non-stereospecific NH2 group (for ICDR2 = -1)

R0 [1..NNOE] corrected upper bound for NOE distance.

NOE specification block
Blockname: NOECALCSPEC

```
     WRITE  (unit,11) DISH, DISC
     DO 10  N=1, NNOE
10   WRITE  (unit,12) IDR1[N], JDR1[N], KDR1[N], LDR1[N], ICDR1[N], VACS1[N],
                      IDR2[N], JDR2[N], KDR2[N], LDR2[N], ICDR2[N], VACS2[N],
                      R0[N]
11   FORMAT (2F10.5)
12   FORMAT (12I5,1F10.5)
```

7.9. SASA implicit solvent model

Program make_sasa_top adds the atom-specific information required to use the SASA implicit solvent model to the molecular topology file (see Sec. 5-2.16. It reads in an existing molecular topology file created using make_top, along with a SASA specification library file, which contains the atom-specific SASA parameters. The specification library file must be for the same force field as was used to create the molecular topology file. The inclusion of hydrogen atoms in the calculation of the SASA during the simulation can also be specified. The following variables are known

`NRSASAT` Number of atom types with a unique set of SASA parameters (not given in file).

`RADI[1..NRSASAT]` Atomic radius for each SASA atom type.

`PI[1..NRSASAT]` Atom type-specific parameter for reduction in SASA.

`SIGMAI[1..NRSASAT]` Scaling parameter for SASA energy term ($kJ \cdot mol^{-1} \cdot nm^{-2}$)

`NRIACI[1..NRSASAT]` Number of integer atomic codes corresponding to this SASA atom type.

`IAC[1..NRSASAT,1..NRIACI]`
Integer atomic code for each atom corresponding to this SASA atom type.

These variables are stored in the following block of the library file:

SASA parameter specification block
Blockname: SASASPEC

```
     DO 10  I=1, NSASAT
10   WRITE  (unit,20) RADI[I], PI[I], SIGMAI[I], NRIACI[I], (IAC[I,J], J=1, NRIACI)
20   FORMAT (F5.3,3X,F5.3,4X,5I,3X,2I,3X,5I3)
```

The 5I3 in format statement 20 is for NIAC ≤ 5; this should be altered if NIAC > 5. Examples of a SASA specification library file can be found in data/sasa45b3.spec and data/sasa53a6.spec. The values of RADI and PI in these files were optimised by Hasel et al.[1] for RSOLV = 0.14 nm. They should not be changed without justification. Different values of SIGMAI are required if the SASA implicit solvent model is used alone (see[2]) or with the VOLUME correction (see[3]). The IAC values will depend on the force field that is used. They are listed in Vol. 3.

7.10. DISICL angle, region and segment definitions

Program `disicl` classifies protein and nucleic acid secondary structure based on dihedral angles (see Sec. 5-4.11.[4,5] Angle, region and segment definitions are read in from a user-specified library file.

The library file is defined by the following variables:

`DIHNAME` Name of the dihedral to define.

`ATOM[1..4]` Atom names defining a dihedral angle, either simply by the name or by an expression in the following format: D;RES1,RES2:B;RES3:C , where RES1..3 are residue names for which atom B, B and C, respectively will be used, whereas atom D is the default which will be used for all other residues. The most common case will be different atoms for purines and pyrimidines in nucleic acids, e.g. N1;GUA,ADE:N9.

`SHIFT[1..4]` Relative residue number for each atom.

`REGNAME` The name of a DISICL region.

`REGMIN[1..NDIH]` The lower limit of the region.

`REGMAX[1..NDIH]` The upper limit of the region.

CLASSNAME	The name of a DISICL segment.
SEGDEF1	The region the current residue has to fall into.
SEGDEF2	The region the following residue has to fall into.
CLASSSHORT	Shortname of the class.

Apart from the title block, the disicl library file contains the following blocks:

Dihedral angle definition block
Blockname: DSCLANG

```
     DO 10 N=1, NDIH
10   WRITE  (unit,11) DIHNAME, (ATOM[N], N=1, 4), (SHIFT[N], N=1, 4)
11   FORMAT (A6, 4A4, 4I4)
```

Region definition block
Blockname: DSCLREG

```
     DO 10  N=1, NREG
10   WRITE  (unit,11) REGNAME ((REGMIN[M], REGMAX[M]), M=1, NDIH)
11   FORMAT (A8, 16F3.1)
```

Class definition block
Blockname: DSCLCLASS

```
     DO 10  N=1, NSEG
10   WRITE  (unit,11) CLASSNAME, SEGDEF1, SEGDEF2, CLASSSHORT
11   FORMAT (A20, 2A8, A6)
```

An example of this library file is DISICL_prot_detailed.lib.

CHAPTER 8

Input file for MD++

The data structure of the *input file* (input flag @input for MD++) is as follows:

MD++ QUICK REFERENCE SHEET

- Blocks can appear in any order
- Compulsory blocks are marked by a star
- When an optional block is not given all switches will be set to their "DEFAULT" value.
- Linebreaks in the variable list should match linebreaks in the input file
- Error checking is performed in three phases:
- Phase I: unknown, duplicate or missing compulsory blocks, switches defining array lengths
- Phase II: incorrect switch values or variable ranges within the blocks
- Phase III: incompatible switches within and among the blocks
- EM, MD, SD, or RT denote the energy minimisation, molecular dynamics, stochastic dynamics and trajectory reading modes of the program
- NRP(> 0): number of atoms of the solute
- NRAM(> 0): number of atoms per solvent molecule
- NATTOT=NRP+NSM*NRAM: total number of atoms in the system

TITLE*
text

- Arbitray text that can be used to identify the simulation.

AEDS

AEDS
ALPHLJ,ALPHC,FORM,NUMSTATES
EMAX,EMIN
EIR(1..NUMSTATES)
NTIAEDSS,RESTREMIN,BMAXTYPE,BMAX,ASTEPS,BSTEPS

AEDS 0,1 controls accelerated enveloping distribution sampling (A-EDS)
0: no accelerated enveloping distribution sampling (EDS) [DEFAULT]
1: accelerated enveloping distribution sampling
ALPHLJ $\geq$ 0.0 Lennard-Jones soft-core parameter
ALPHC $\geq$ 0.0 Coulomb soft-core parameter
FORM 1..4 defines type of A-EDS simulation
1: A-EDS with fixed parameters
2: fixed Emax and Emin parameters, search for offset parameters
3: search for Emax and Emin parameters, fixed offset parameters
4: search for Emax, Emin and offset parameters
NUMSTATES $\geq$ 2 number of (end)states
EMAX A-EDS parameter Emax
EMIN A-EDS parameter Emin
EIR energy offsets for states

NTIAEDSS 0..1 controls startup of the A-EDS parameter search
0: read A-EDS parameter search configuration from input configuration
1: initialize A-EDS parameter search
RESTREMIN 0..1 controls restriction of parameter Emin during parameter search
0: do not restrict Emin $\geq$ minimum average end-state energy
1: restrict Emin $\geq$ minimum average end-state energy before all states have been visited at least once
BMAXTYPE 1..2 controls type of given anticipated maximum energy barrier between the states
1: absolute maximum energy barrier between the states in energy units
2: multiples of the standard deviation of the energy of the end-state with the lowest average energy
BMAX maximum energy barrier parameter
ASTEPS have-life in simulation steps of the exponential averaged energy difference between the end-states at the begining of the run
BSTEPS have-life in simulation steps of the exponential averaged energy difference between the end-states at the end of the run

- The parameter Emax must be $\geq$ Emin
- NBATHS= 0 results in an error, in addition all baths must have the same temperature TEMP0
- A-EDS cannot be applied with replica exchange
- A-EDS cannot be applied to solvent atoms

BOUNDCOND*

NTB,NDFMIN

NTB -1..2 controls type of boundary conditions
-1: truncated-octahedral periodic boundary conditions
0: vacuum boundary conditions
1: rectangular periodic boundary conditions
2: triclinic periodic boundary conditions
NDFMIN ≥ 0 number of degrees of freedom subtracted for temperature

- NTM$\neq$ 0 requires NTB= 1,2
- pressure coupling requires NTB$\neq$ 0
- (semi-)anisotropic pressure coupling (SCALE=2,4) requires NTB=1 or 2
- full anisotropic pressure coupling (SCALE=3) requires NTB=2
- abs(NLRELE)$>$ 1 requires NTB$\neq 0, -1$
- NTISHI=0 requires NTB$\neq$0
- NTRD$\neq$ 0 and NTRB$\neq$ 0 require NTB$\neq$ 0
- Initial box parameters (GENBOX) are read from @conf

CGRAIN

NTCGRAN,EPS,EPSM

NTCGRAN $= 0..3$ Coarse grain selection
0: No coarse graining [DEFAULT]
1: Coarse grain simulation using the MARTINI model
2: Coarse grain simulation using the GROMOS model
3: Mixed-grain simulation using the GROMOS model
EPS ≥ 0.0 Dielectric constant for coarse grained - coarse grained coulombic interactions
EPSM ≥ 0.0 Dielectric constant for coarse grained - fine grained coulombic interactions

COMTRANSROT

NSCM

NSCM controls system center-of-mass (com) motion removal
0: no com motion removal [DEFAULT]
< 0: com translation and rotation are removed every abs(NSCM) steps
> 0: com translation is removed every NSCM steps

- NSCM≠0 should not be used with roto-translational constraints (RTC=1)

CONSTRAINT*

NTC
NTCP,NTCP0(1),[NTCP0(2),NTCP0(3)]
NTCS,[NTCS0(1),NTCS0(2),NTCS0(3)]

NTC 0,1,2,3,4 controls application of constraints to bonds
1: constraints are applied to solvent only
2: constraints are applied to solvent and solute bonds involving hydrogen atoms and to bonds specified in the topology CONSTRAINT block
3: constraints are applied to solvent and solute bonds
4: constraints are applied to bonds specified in the CONSTRAINT block in the topology and to solvent

NTCP shake, lincs, flexshake controls algorithms to apply solute constraints
shake(1) apply shake for solute
lincs(2) apply lincs for solute
flexshake(3) apply flexible shake for solute

NTCP0(1) > 0 option parameters for constraint algorithm Shake: Tolerance, Lincs: Order

NTCP0(2..3) ≥ 0 option parameters for flexible shake algorithm: readin, mode [only supply when flexible shake is selected]

NTCS shake, lincs, flexshake, settle, m_shake, gpu_shake controls algorithm to apply solvent constraints
shake(1) apply shake for solvent
lincs(2) apply lincs for solvent
flexshake(3) apply flexible shake for solvent
settle(4) apply settle for solvent
m_shake(5) apply m_shake for solvent
gpu_shake(6) apply m_shake for solvent using GPU

NTCS0(1) ≥ 0 option parameter for constraint algorithm: (flexible) Shake or M-Shake: Tolerance; Lincs: order; Settle: do not specify

NTCS0 (2..3) ≥ 0 option parameters for flexible shake algorithm: readin, mode [only supply when flexible shake is selected]

NTCG ≥ 0 number of GPUs [only supply when GPU shake is selected]

NTCD ≥ −1 device number of the GPU; if -1 given driver will determine [only supply when GPU shake is selected]

COVALENTFORM

NTBBH,NTBAH,NTBDN

NTBBH 0,1 controls bond-stretching potential energy function
0: quartic potential energy function [DEFAULT]
1: harmonic potential energy function

NTBAH 0,1 controls bond-angle bending potential energy function
0: cosine-harmonic potential energy function [DEFAULT]
1: harmonic potential energy function

NTBDN 0,1 controls torsional dihedral potential energy function
0: arbitrary phase shifts [DEFAULT]
1: phase shifts limited to 0 and 180 degrees

- A topology containing bond types only in the form of a BONDANGLETYPE block and no BONDANGLEBENDTYPE block requires NTBAH= 0, the HARMBONDANGLETYPE block requires NTBAH= 1.

- A topology containing a DIHEDRALTYPE and no TORSDIHEDRALTYPE block requires NTBDN=1
- NTBDN=1 along with the presence of a topology block TORSDIHEDRALTYPE requires that all phase shifts are 0 or 180 degrees in this block

DIHEDRALRES

NTDLR,CDLR,PHILIN

NTDLR 0...3 controls dihedral-angle restraining or constraining
- 0: no dihedral restraining [DEFAULT]
- 1: dihedral restraining using CDLR (WDLR ignored)
- 2: dihedral restraining using CDLR×WDLR
- 3: dihedral constraining

CDLR ≥0.0 force constant for dihedral restraining (multiplied by WDLR)

PHILIN 0...180 absolute deviation (degrees) after which the potential energy function is linearised. If zero no linearisation performed.

- Dihedral restraints and weights WDLR in DIHEDRALRESSPEC and PERTDIHRESSPEC block read from @dihrest

DISTANCEFIELD

NTDFR,GRID,PROTEINOFFSET,PROTEINCUTOFF,PROTECT,UPDATE,SMOOTH,RL,NTWDF, PRINTGRID

NTDFR 0,1 controls distance-field restraining
- 0: no distance-field restraining [DEFAULT]
- 1: apply distance-field restraining

GRID > 0.0 grid size for distance-field

PROTEINOFFSET > 0.0 penalty for distances through the host

PROTEINCUTOFF > 0.0 distance to host atoms to be considered inside

PROTECT ≥ 0 protect grid points within this radius around the zero-distance point from being flagged as protein

UPDATE > 0 update frequency for grid

RL ≥ 0.0 potential energy function for distances larger than RL

SMOOTH ≥ 0 smoothen the host boundary after grid construction by SMOOTH layers

NTWDF ≥ 0 write distance-field information to special trajectory every NTWDF steps

PRINTGRID 0,1 write grid to final configuration file

- Distance-field specification read from distance restraints specification file (@distrest)
- To use distance-field coordinate in local elevation, turn off the restraining potential energy function (NTDFR = 0)
- Distance-field restraints require NTB=1.

DISTANCERES

NTDIR,NTDIRA,CDIR,DIR0,TAUDIR,FORCESCALE,VDIR,NTWDIR

NTDIR -2..3 controls distance restraining
- -2: time-averaged restraining using force constant CDIR×W0
- -1: time-averaged restraining using force constant CDIR (W0 ignored)
- 0: no distance restraining [DEFAULT]
- 1: instantaneous restraining using force constant CDIR (W0 ignored)
- 2: instantaneous restraining using force constant CDIR×W0

NTDIRA 0,1 controls values of initial distance averages
- 0: zero initial averages [DEFAULT]
- 1: read current averages from startup file

CDIR ≥ 0.0 force constant for distance restraining

DIR0 ≥ 0.0 distance offset in restraining function
TAUDIR > 0.0 coupling time for time averaging
FORCESCALE 0..2 controls approximation of force scaling
0: approximate $d\langle r\rangle/dr = 1$
1: approximate $d\langle r\rangle/dr = (1.0 - exp(-\Delta t/\tau))$
2: use $d\langle r\rangle/dr = (1.0 - exp(-\Delta t/\tau)) * (\langle r\rangle/r)^4$
VDIR 0,1 controls contribution to virial
0: no contribution
1: distance restraints contribute to virial
NTWDIR ≥ 0 write every NTWDIRth step distance restraining information to external file

- NTDIRA= 1 requires NTDIR≤ 0
- List of distance restraints and weights W0 (DISTANCERESSPEC) read from @distrest (24)
- Average distances (DISRESEXPAVE) read from @conf if NTDIRA= 1
- NTWDIR> 0 requires the specification of a special trajectory file with @trs

EDS

EDS,ALPHLJ,ALPHC,FORM,NUMSTATES,S,EIR

EDS 0,1 controls enveloping distribution sampling
0: no enveloping distribution sampling (EDS) [DEFAULT]
1: enveloping distribution sampling
ALPHLJ ≥ 0.0 Lennard-Jones soft-core parameter
ALPHC ≥ 0.0 Coulomb soft-core parameter
FORM 1..3 defines functional form of the Hamiltonian
1: Single s Hamiltonian
2: Hamiltonian with NUMSTATES*(NUMSTATES-1)/2 (pairwise) s parameters
3: Hamiltonian with (NUMSTATES-1) s parameters
NUMSTATES ≥ 2 number of (end)states
S > 0.0 smoothness parameter(s) (number according to functional form)
EIR ≥ 0.0 energy offsets for states

- FORM=3 requires the specification of a tree: "S" becomes "i j S", where i and j are the pair of states for which the S is applied
- NBATHS= 0 results in an error, in addition all baths must have the same temperature TEMP0
- EDS cannot be applied with replica exchange
- EDS cannot be applied to solvent atoms

ELECTRIC

FIELD,DIPOLE,CURRENT
EF_x,EF_y,EF_z
DIPGRP,NTWDIP
NTWCUR,NCURGRP,CURGRP(1..NCURGRP)

FIELD 0..1 controls the use of applied electric field
0: not used [DEFAULT]
1: electric field is applied
DIPOLE 0..1 controls the calculation of the box dipole
0: not used [DEFAULT]
1: box dipole is calculated and written to special trajectory
CURRENT 0..1 controls the calculation of electric (ionic) currents
0: not used [DEFAULT]
1: electric (ionic) current is calculated and written to special trajectory
EF_x *double* x-component of the electric-field vector
EF_y *double* y-component of the electric-field vector
EF_z *double* z-component of the electric-field vector
DIPGRP 0..2 define the groups for which the box dipole is calculated

0: solute only
1: solvent only
2: all

NTWDIP ≥ 0 write box dipole to special trajectory every NTWDIPth step
NTWCUR ≥ 0 write box currents to special trajectory every NTWCURth step
NCURGRP≥ 0 number of current groups
CURGRP(1..NCURGRP) ≥ 0 last atom of each current group

ENERGYMIN

NTEM,NCYC,DELE,DX0,DXM
NMIN,FLIM,CGIC,CGIM

NTEM 0..3 controls energy minimisation mode
0: do not do energy minimisation [DEFAULT]
1: use steepest-descent minimisation
2: use Fletcher-Reeves conjugate gradient minimisation
3: use Polak-Ribiere conjugate gradient minimisation
NCYC > 0 number of steps before resetting the conjugate-gradient search direction
$= 0$ reset only if the energy grows in the search direction
DELE > 0.0 energy threshold for convergence
> 0.0 (conjugate-gradient) RMS force threshold for convergence
DX0 > 0.0 initial step size
DXM > 0.0 maximum step size
NMIN > 0 minimum number of minimisation steps
FLIM ≥ 0.0 limit force to maximum value
CGIM > 0 (conjugate-gradient only) maximum number of cubic interpolations per step
CGIC > 0.0 (conjugate-gradient only) RMSD threshold after interpolation

- DX0 $\leq$ DXM
- NTSD $\neq 0$ requires NTEM= 0
- NTRD $\neq 0$ requires NTEM= 0
- pressure or temperature coupling is not allowed with NTEM = 0
- NSCM $\neq 0$ (center-of-mass motion removal) requires NTEM = 0

EWARN

MAXENER

MAXENER Issues a warning if the total energy is larger than this value

FORCE*

NTF(1..6)
NEGR
NRE(1..NEGR)

NTF(1..6) 0,1 determines terms used in force calculation
NTF(I)= 0 do not include terms of type I
NTF(I)= 1 include terms of type I
NTF (1) bonds
NTF (2) bond angles
NTF (3) improper dihedrals
NTF (4) dihedrals
NTF (5) nonbonded electrostatic interactions
NTF (6) nonbonded van der Waals interactions
NEGR ≥ 0 number of energy groups
0: no energy groups

> 0: number of energy groups
NRE(1..NEGR) ≥1 last atom in each energy group

- If NEGR= 0, the specification of NRE(1..NEGR) is omitted
- NTF(5)=NTF(6)= 0 suppresses non-bonded interactions but does not affect the pairlist making
- NEGR≠ 0 requires NRE values in ascending order and NRE(NEGR) = NATTOT

INITIALISE

NTIVEL,NTISHK,NTINHT, NTINHB
NTISHI,NTIRTC,NTICOM
NTISTI
IG,TEMPI

NTIVEL 0,1 controls generation of initial velocities
0: read from startup file (if applicable) [DEFAULT]
1: generate from Maxwell distribution at temperature TEMPI
NTISHK 0..3 controls shaking of initial configuration
0: no initial SHAKE [DEFAULT]
1: initial SHAKE on coordinates only
2: initial SHAKE on velocities only (not allowed)
3: initial SHAKE on coordinates and velocities
NTINHT 0,1 controls generation of initial Nosé-Hoover (chain) thermostat variables
0: read from startup file (if applicable) [DEFAULT]
1: initialise variables to zero
NTINHB 0,1 controls generation of initial Nosé-Hoover (chain) barostat variables
0: read from startup file (if applicable) [DEFAULT]
1: reset variables to zero
NTISHI 0,1 controls initial setting of atomic shift vectors across infinite periodic system
0: read from startup file (if applicable) [DEFAULT]
1: reset shift vectors to zero
NTIRTC 0,1 controls initial setting of positions and orientations for roto-translational constraints
0: read from startup file (if applicable) [DEFAULT]
1: reset positions and orientations based on the initial configuration of startup file
NTICOM 0..2 controls initial removal of com motion
0: no initial system com motion removal [DEFAULT]
1: initial com translation is removed
2: initial com translation is removed and initial com rotation is set to constraint value
NTISTI 0,1 controls generation of stochastic integrals
0: read stochastic integrals and IG from startup file (if applicable) [DEFAULT]
1: set stochastic integrals to zero and use IG from INITIALISE input block
IG > 0 random number generator seed
TEMPI ≥ 0.0 initial temperature

- NTRD≠ 0 requires NTIVEL= 0 and NTISHK= 0, 1
- NTBTYP≠ 3 requires NTINHT= 0
- NTB= 0 requires NTISHI≠0
- NTSD= 0 requires NTISTI= 0
- initial coordinates (POSITION or POSITIONRED) read from @conf
- initial velocities (VELOCITY or VELOCITYRED) read from @conf if NTEM= 0 and NTIVEL= 0
- initial Nosé-Hoover chain thermostat variables (NHCVARIABLES) read from @conf if NTBTYP = 2 and NTINHT= 0

- initial shift vectors (LATTICESHIFTS) read from @conf if NTB≠ 0 and NTISHI= 0
- initial positions and orientations (ROTTRANSREFPOS) read from @conf if RTC=1 and NTIRTC= 0
- stochastic integrals and seed (STOCHINT) read from @conf if NTSD≠ 0 and NTISTI= 0
- If NTIVEL= 0 and NTISTI= 0, IG is irrelevant
- If NTIVEL= 0, TEMPI is irrelevant
- NTIVEL≠ 0 results in a warning if VELOCITY block is found in @conf
- NTISTI≠ 0 results in a warning if STOCHINT block is found in @conf
- NTIRTC≠ 0 results in a warning if ROTTRANSREF block is found in @conf

INNERLOOP

NTILM 0..4, acceleration method used
- 0: use standard solvent loops [DEFAULT]
- 1: use fast generic solvent loops
- 2: use solvent loops with hardcoded parameters
- 3: use solvent loops with tabulated forces and energies
- 4: use solvent loops with CUDA library

NTILS 0..1, solvent used
- 0: use topology [DEFAULT]
- 1: use SPC

NGPUS number of GPUs to use

NDEVG which GPU device number to use; if not given driver will determine

INTEGRATE

NINT

NINT $= 0, 1$ selects integration method
- 0: No integration takes place
- 1: Leap-frog integration scheme is used [DEFAULT]

JVALUERES

NTJVR,NTJVRA,CJVR,TAUJVR,NJVRTARS,
NJVRBIQW,LE,NGRID,DELTA,NTWJV

NTJVR -3...2
- -3: biquadratic using CJVR×WJVR
- -2: time-averaged using CJVR×WJVR
- -1: time-averaged using CJVR (WJVR ignored)
- 0: no ^3J-value restraining [DEFAULT]
- 1: instantaneous using CJVR (WJVR ignored)
- 2: instantaneous using CJVR×WJVR

NTJVRA 0,1 controls reading of averages from startup file
- 0: start from initial values of J0 [DEFAULT]
- 1: read time-averages from startup file (for continuation of time-averaged run)

CJVR ≥0 ^3J-value restraining force constant (weighted by individual WJVR)

TAUJVR >0 coupling time for time-averaging

NJVRTARS 0,1 controls scaling of force in time-averaging
- 0: omit factor $[1 - \exp(\Delta t/\tau_{Jr})]$, i.e. set it to one
- 1: scale force by $[1 - \exp(\Delta t/\tau_{Jr})]$

NJVRBIQW 0...2 controls weighting of contributions for biquadratic restraining
- 0: equal weights of $\mathbf{f}_i^{tav}$ and $\mathbf{f}_i^{inst}$
- 1: multiply $\mathbf{f}_i^{tav}$ with $[1 - \exp(\Delta t/\tau_{Jr})]$
- 2: multiply $\mathbf{f}_i^{tav}$ with zero

LE 0,1 local-elevation restraining [md++ only]

0: local-elevation off [DEFAULT]
1: local-elevation on

NGRID	>1	number of grid points in local-elevation restraining
DELTA	≥0.0	no elevation of potential if J is within DELTA of J0
NTWJV	≥0	write ^3J-value averages and LE grid to special trajectory

=0: do not write [DEFAULT]
>0: write every NTWJV-th step

- NTJVRA≠ 0 requires NTJVR< 0
- NTJVRA= 0 and NTJVR< 0 results in a warning

LAMBDAS

NTIL
NTLI(1..),NILG1(1..),NILG2(1..),ALI(1..),BLI(1..),CLI(1..),DLI(1..),ELI(1..)

NTIL	off, on, 0, 1
	off,0: no special treatment of interactions with individual λ−values
	on,1: interactions are treated with special individual λ−values
NTLI(1..)	interaction type to treat with individual λ: bond(1), angle(2), dihedral(3), improper(4), vdw(5), vdw_soft(6), crf(7), crf_soft(8), distanceres(9), dihedralres(10), mass(11)
NILG1, NILG2	energy groups of interactions that are treated with individual λ−values
ALI, BLI, CLI, DLI, ELI	Polynomial coefficients linking the individual λ−values to the overall λ−value

- Input for this block is read linewise, i.e. you need to specify each interaction type within one separate line.

LOCALELEV

NTLES,NLEPOT,NTLESA,NTWLE,NLEPID,NTLEFR

NTLES	0..2 controls application of local-elevation
	0: no local-elevation potential energy function[DEFAULT]
	1: local-elevation using linear build up
	automatic force-constant update
NLEPOT	Number of applied potential energy functions
NTLESA	1..2 controls reading of local-elevation potential energy functions
	1: read averages and parameters from startup file [DEFAULT]
	2: read averages and parameters from LEUS database file (@lud)
NTWLE	≥ 0 write potential energy to special trajectory
NLEPID(1..)	ID of potential energy function to read and apply
NTLEPFR(1.0),1	controls build up vs freezing of memory
	0: do memory build up (time-dependent potential energy function)
	1: freeze memory (no build up)

- List of local-elevation dihedrals (LOCALELEVSPEC) read from @led

MULTIBATH

NTBTYP (NUM)
NBATHS
TEMP0 (1..NBATHS) TAU(1..NBATHS)
DOFSET
LAST(1..DOFSET) COM-BATH(1..DOFSET) IR-BATH(1..DOFSET)

NTBTYP controls temperature coupling algorithm to use
weak-coupling(0) use weak coupling scheme
nose-hoover(1) use Nosé Hoover scheme
nose-hoover-chains(2) use Nosé Hoover chains scheme

NUM	≥ 0	number of chains in Nosé Hoover chains scheme [only specify when needed]
NBATHS	≥ 0	number of temperature baths to couple to
TEMP0()	≥ 0.0	bath reference temperature per bath
TAU()	≥ 0.0 or -1	coupling time per bath, -1 turns coupling off
DOFSET	≥ 0	number of distinguishable sets of degrees of freedom
LAST()	> 0	last atom for set of degrees of freedom
COM-BATH()	≥ 1	temperature bath to couple centre-of-mass motion of this set of d.o.f. to
IR-BATH()	≥ 1	temperature bath to couple internal and rotational degrees of freedom of this set to

- LAST should be ≤NATTOT
- COM-BATH and IR-BATH should be between 1 and NBATHS

MULTICELL

NTM,NCELLA,NCELLB,NCELLC,
TOLPX,TOLPV,TOLPF,TOLPFW

NTM	0,1	switch for multiple-unit-cell simulation 0: single-unit-cell simulation [DEFAULT] 1: multiple-unit-cell simulation
NCELLA	≥ 1	number of subdivisions along a-axis
NCELLB	≥ 1	number of subdivisions along b-axis
NCELLC	≥ 1	number of subdivisions along c-axis
TOLPX	> 0.0	relative tolerance for coordinate periodicity check (not supported)
TOLPV	> 0.0	absolute tolerance for velocity periodicity check (not supported)
TOLPF	> 0.0	absolute tolerance for force periodicity check (not supported)
TOLPFW	> 0.0	absolute tolerance for force periodicity fix and warning (not supported)

- The indexing of subcells goes along c (fastest index), then b, then a
- Solvent molecules are reset to initial subcell
- Solute molecules may drift across subcells
- NTB$\neq$ 1,2 requires NTM= 0

MULTIGRADIENT

NTMGRE, NTMGRP,
NTMGRN,
MGRVAR(1..NTMGRN), MGRFRM(1..NTMGRN), MGRNCP(1..NTMGRN),
MGRCPT(1..NTMGRN), MGRCPV(1..NTMGRN)

NTMGRE	0, 1	disables/enable multiple gradients 0: disable gradients 1: enable gradients
NTMGRP	0..3	printout of the gradient curves in the output file 0: don't print 1: plot the curves 2: print the values of the curves 3: plot and print the curves
NTMGRN	>=0	number of gradients
MGRVAR()		name of the variable to be affected, available are: TEMP0, CPIR, CDIR, RESO, CXR, COPR
MGRFRM()	0..3	functional form of the gradient

0: linear interpolation between control points
1: cubic spline interpolation between control points
2: Bezier curve
3: Oscillation: $A \sin\left[\frac{2\pi}{T}(d - dt)\right] + b$
Note: MGRNCP is 2, A =MGRCPT[1], T =MGRCPV[1], dt =MGRCPT[2], b =MGRCPV[2]
MGRCP() >=2 number of control points
MGRCPT() >=0 time of the control point
MGRCPV() value of the variable at the control point

MULTISTEP

STEPS,BOOST

STEPS ≥ 0 calculate non-bonded every STEPSth step
BOOST 0,1 switch to control the method:
0: stored forces of STEPSth step are added every step
1: stored forces of STEPSth setp are multiplied by STEPS and added every STEPSth step [DEFAULT]

NONBONDED*

NLRELE
APPAK,RCRF,EPSRF,NSLFEXCL
NSHAPE,ASHAPE,NA2CLC,TOLA2,EPSLS
NKX,NKY,NKZ,KCUT
NGX,NGY,NGZ,NASORD,NFDORD,NALIAS,NSPORD
NQEVAL,FACCUR,NRDGRD,NWRGRD
NLRLJ,SLVDNS

NLRELE -1..3 method to handle electrostatic interactions
-1: reaction-field method (LSERF compatibility mode)
0: no electrostatic interactions
1: reaction-field method
2: Ewald method
3: P^3M method
APPAK ≥ 0.0 reaction-field inverse Debye screening length
RCRF ≥ 0.0 reaction-field radius
0.0: set reaction-field radius to infinity
> 0.0: reaction-field radius
EPSRF = 0.0 or ≥ 1.0 controls reaction-field permittivity
0.0: set reaction-field permittivity to infinity
≥ 1.0: reaction-field permittivity
NSLFEXCL 0,1 contribution of excluded atoms to reaction field
0: contribution turned off
1: contribution considered [DEFAULT]
NSHAPE -1..10 lattice-sum charge-shaping function (-1: Gaussian)
ASHAPE > 0.0 width of the lattice-sum charge-shaping function
NA2CLC 0..4 controls evaluation of the lattice-sum A_2 term
0: $A_2 = \tilde{A}_2 = 0$
1: $\tilde{A}_2$ exact, $A_2 = \tilde{A}_2$
2: A_2 numerical, $\tilde{A}_2 = A_2$
3: $\tilde{A}_2$ exact from Ewald or from mesh and atom coords, A_2 numerical
4: $\tilde{A}_2$ averaged from mesh only, A_2 numerical
TOLA2 > 0.0 relative tolerance for numerical A_2 evaluation
EPSLS = 0.0 or ≥ 1.0 controls lattice-sum (external) permittivity
0.0: set lattice-sum permittivity to infinity (tinfoil)
≥ 1.0: lattice-sum permittivity

NKX,NKY,NKZ	> 0	maximum absolute Ewald k-vector components
KCUT	> 0.0	Ewald k-space cutoff
NGX,NGY,NGZ	> 0	P^3M number of grid points along the three box axes (even)
NASORD	1..5	order of the mesh charge-assignment function
NFDORD	0..5	order of the mesh finite-difference operator (0: ik-differentiation)
NALIAS	> 0	number of mesh alias vectors considered
NSPORD		order of the SPME B-spline function (not available)
NQEVAL	≥ 0	controls accuracy reevaluation
		0: do not reevaluate accuracy
		>0: reevaluate accuracy every NQEVAL steps
FACCUR	> 0.0	rms force error threshold to recompute influence function
NRDGRD	0,1	read initial influence function (and derivatives) from file (not implemented)
NWRGRD	0,1	write final influence function (and derivatives) to file (not implemented)
NLRLJ	0,1	controls long-range Lennard-Jones correction (not implemented)
SLVDNS	> 0.0	average solvent density for long-range Lennard-Jones correction (not implemented)

- Numerical A_2: by Ewald summation up to relative tolerance TOLA2
- Exact $\tilde{A}_2$: by Ewald(abs(NLRELE)= 2) or based on mesh and exact atom coordinates (abs(NLRELE= 3,4))
- Average $\tilde{A}_2$: based on mesh for atom coordinates averaged over box
- For a truncated octahedron box, NGA, NGB and NGC refer to the axes of the transformed triclinic cell.
- Choices for NSHAPE are found in Tab. 2-7.1
- NA2CLC= 1 requires abs(NLRELE)= 2
- NA2CLC= 4 requires abs(NLRELE)= 3,4
- NGX,NGY and NGZ must be even
- NTB= 0 requires NLRELE=-1,0,1
- NLRELE$\neq$ 0,1 and ASHAPE>RCUTP results in a warning
- NA2CLC= 0 and NLRELE= 2,3 results in a warning
- P3M and Ewald require atomistic cutoff scheme
- P3M and Ewald can not be used with multiple energy groups

ORDERPARAMRES

NTOPR,NTOPRA,COPR,TAUJVR,UPDOPR,NTOPW

NTOPR	-2...2	
		-2: time-averaged using COPR×WOPR
		-1: time-averaged using COPR (WOPR ignored)
		0: no S^2-order parameter restraining [DEFAULT]
		1: window-averaged using COPR (WOPR ignored)
		2: window-averaged using COPR×WOPR
NTOPRA	0,1	controls reading of averages from startup file
		0: start from initial values [DEFAULT]
		1: read time-averages from startup file (for continuation of time-averaged run)
COPR	≥ 0	S^2-order parameter restraining force constant (weighted by individual WOPR)
TAUOPR	≥ 0	coupling time for time-averaging, length of averaging window for window averaging
UPDOPR	>0	update order parameters only every UPDOPR steps (only relevant for window averaging)
NTWOP	≥ 0	write S^2-value averages to special trajectory
		=0: do not write [DEFAULT]
		>0: write every NTWOP-th step

PAIRLIST

algorithm NSNB RCUTP RCUTL SIZE TYPE

algorithm	standard, grid	method for generating pairlist

standard(0)	GROMOS96 like pairlist
grid(1)	md++ grid pairlist
grid_cell(2)	grid-based algorithm using a mask[6]
NSNB	>0 frequency (number of steps) a pairlist is constructed
RCUTP	>0.0 cut-off used in pairlist construction
RCUTL	>0.0 cut-off used in long range interaction
SIZE	>0.0, auto size of grid cell
auto:	0.5*RCUTP
TYPE	chargegroup (0), atomic (1) type of cut-off
chargegroup:	chargegroup based cut-off
atomic:	atom based cut-off

PERSCALE

RESTYPE
KDIH,KJ,T,DIFF,RATIO,READ

RESTYPE		Special energy term to which periodic scaling should be applied
0:		Do not apply periodic scaling.
1:		Apply periodic scaling to ^3J-value restraints
KDIH	≥ 0.0	Maximum scaling factor for dihedral angle potential
KJ	≥ 0.0	Maximum scaling factor for ^3J-value restraint potential
T	> 0	Period of cosine scaling function
DIFF	≥ 0.0	Minimum deviation from target value to start a scaling period
RATIO	> 0.0	Minimum fraction of T that needs to be passed before starting a new scaling period
READ	$= 0, 1$	Read scaling parameters from coordinate file for continuation simulation

- RESTYPE=1 requires NTBDN=1

PERTURBATION

NTG,NRDGL,RLAM,DLAMT
ALPHLJ,ALPHC,NLAM
NSCALE

NTG	0,1 controls use of free-energy calculation
0:	no free-energy calculation is performed [DEFAULT]
1:	calculate $\frac{\partial \mathcal{H}(\lambda,\mu)}{\partial \lambda}$
NRDGL	0,1 controls reading of initial values
0:	use initial λ parameter from PERTURBATION input block
1:	read initial λ value from startup file
RLAM	0.0..1.0 initial value for λ
DLAMT	≥ 0.0 rate of λ increase in time
ALPHLJ	≥ 0.0 Lennard-Jones soft-core parameter
ALPHC	≥ 0.0 Coulomb soft-core parameter
NLAM	> 0 power dependence of λ coupling
NSCALE	0,1,2 turn energy group scaling on
0:	no scaling [DEFAULT]
1:	scaling
2:	scaled interactions only

- NTWG$\neq$ 0 requires NTG$\neq$ 0
- Perturbation topology file read from @pttopo
- λ (PERTDATA) read from @conf if NRDGL=1

PRECALCLAM

NRLAM, MINLAM, MAXLAM

NRLAM ≥ 0 determines calculation of $\mathcal{H}$ and $\partial\mathcal{H}/\partial\lambda$ at alternative values of λ
0: off
> 0: precalculating energies and derivatives for NRLAM extra λ values
MINLAM 0.0 .. 1.0: minimum λ value to precalculate energies and derivatives
MAXLAM MINLAM .. 1.0: maximum λ value to precalculate energies and derivatives

POLARISE

COS,EFIELD,MINFIELD,DAMP,WRITE

COS 0,1,2 controls explicit inclusion of electronic polarisation effects
0: do not explicitly include electronic polarisation [DEFAULT]
1: use charge-on-spring model for dipolar polarisation
2: use charge-on-spring model for dipolar polarisation with off atom site
EFIELD 0,1 controls evaluation site for electric field
0: evaluate at atomic position of polarisable centres
1: evaluate at position of charges-on-spring
MINFIELD > 0.0 convergence criterium in iterative procedure to determine positions of charges-on-spring
DAMP 0,1 controls polarisability damping
0: use linear relationship between induced dipole moments and electric field
1: damp polarisability (with parameters from topology)
WRITE ≥ 0 write COS positions to special trajectory file
0: do not write COS positions
> 0 : write COS positions every WRITEth step

POSITIONRES

NTPOR,NTPORB,NTPORS,CPOR

NTPOR 0..3 controls atom position restraining or constraining
0: no position re(con)straining [DEFAULT]
1: restraining with force constant CPOR (no B-factor weighting)
2: restraining with force constant CPOR weighted by atomic B-factors
3: position constraining
NTPORB 0,1 controls reading of reference positions and B-factors
0: read reference positions from startup file (@conf) [DEFAULT]
1: read reference positions and B-factors (if required) from special file (@refpos)
NTPORS 0,1 controls scaling of reference positions upon pressure scaling
0: do not scale reference positions [DEFAULT]
1: scale reference positions together with box parameters
CPOR ≥ 0.0 position restraining force constant

- NTPOR= 2 requires NTPORB= 1
- List of re(con)strained atoms (POSRESSPEC) read from @posresspec
- Reference positions in REFPOSITION blocks
- Without pressure coupling, NTPORS has to be 0

PRESSURESCALE

COUPLE,SCALE,COMP,TAUP,VIRIAL
SEMI(1..3)
PRES0(1, 3, 1..3)

COUPLE off,calc,scale controls calculation and scaling of pressure
off(0) no pressure calculation or scaling
calc(1) calculate pressure but no scaling

	scale(2)	calculate and couple pressure to a pressure bath
SCALE	off,iso,aniso,full	controls isotropy of pressure scaling
	off(0)	no pressure scaling
	iso(1)	isotropic pressure scaling
	aniso(2)	anisotropic pressure scaling (x-, y-, z-axes, no angle deformation)
	full(3)	fully anisotropic pressure scaling
	semianiso(4)	semi-anisotropic pressure scaling
COMP		> 0.0 isothermal compressibility
TAUP		≥ 0.0 coupling relaxation time
VIRIAL	none,atomic,group	controls type of virial for pressure calculation
	none(0)	no pressure calculation
	atomic(1)	atomic virial
	group(2)	group-based virial according to PRESSUREGROUPS
SEMI		0..2,0..2,0..2 (semianisotropic couplings: x-, y-, and z-axes)
PRES0(,)		≥ 0.0 reference pressure in Tensor format

PRINTOUT

NTPR,NTPP

NTPR	≥ 0 controls printing of energies
	0: no printing out of energies [DEFAULT]
	> 0: print out energies every NTPR steps
NTPP	0,1 controls dihedral angle transition monitoring
	0: no dihedral angle transition monitoring [DEFAULT]
	1: perform dihedral angle transition monitoring

- Data is printed to standard output.
- Dihedral angle transitions are printed to @trs.

QMMM

NTQMMM,NTQMSW,RCUTQ,NTWQMMM

NTQMMM	0,1 controls application of QM/MM
	0: do not apply QM/MM [DEFAULT]
	1: perform QM/MM simulation
NTQMSW	0,1 software package to use for QM calculation
	0: MNDO
	1: TURBOMOLE
RCUTQ	≥ 0.0 cutoff for electrostatic QM/MM interactions, inclusion of MM charge groups in QM Hamiltonian
	0.0: include all MM atoms
	> 0.0: include only atoms of charge groups closer than RCUTQ
NTWQMMM	≥ 0 write QM/MM related data to special trajectory
	0: do not write [DEFAULT]
	> 0: write every NTWQMMMth step (not yet available)

- Note: QM/MM currently only applicable to systems with non-covalent interactions between QM and MM region.

RANDOMNUMBERS

NTRNG, NTGSL

NTRNG	0,1 random number generator
	0 use GROMOS 96 algorithm
	1 use GSL library (DEFAULT)
NTGSL	$\geq$ -1 GSL random number generation algorithm

-1 use default algorithm (mt 19937)
> = 0 run contrib/rng_gsl for a list of possible arguments

READTRAJ

NTRD,NTSTR,NTRB,NTSHK

NTRD 0,1 controls trajectory-reevaluation mode
0: do not use trajectory-reevaluation mode [DEFAULT]
1: use trajectory-reevaluation mode
NTSTR ¿0 stride: should be the NTWX used to produce the analyzed trajectory
NTRB 1 obsolete option to control reading of box parameters (must be 1)
NTSHK 0..2 controls application of constraints
0: apply constraints with respect to previous coordinates [default]
1: apply constraints with respect to current coordinates
2: do not apply constraints (neither solute nor solvent)

- For consistency, a pairlist should have been made every NTSTR steps (or a divisor thereof) in the generating run
- Velocities and dependent quantities are zeroed
- NTEM≠ 0 requires NTRD= 0
- NTSD≠ 0 requires NTRD= 0
- NTB= 0 requires NTRD= 0
- NSCM> 0 and RTC> 0 (centre-of-mass removeal and roto-translational constraints) are ignored when NTRD= 1.
- Coordinate trajectories (POSITIONRED and GENBOX) are read from @anatrj if NTRD≠ 0

REPLICA

NRET
RET(1..NRET)
LRESCALE
NRELAM
RELAM(1..NRELAM)
RETS(1..NRELAM)
NRETRIAL,NREQUIL,CONT

NRET	≥ 1	Number of replica exchange temperatures
RET()	≥ 0.0	Temperature for each replica
LRESCALE	$= 0, 1$	Scale temperatures after exchange trial
NRELAM	≥ 1	Number of replica exchange lambda values
RELAM()	≥ 0.0	Lambda value of each lambda-replica
RETS()	≥ 0.0	Timestep of each lambda-replica
NRETRIAL	≥ 0	Number of overall exchange trials
NREQUIL	≥ 0	Number of exchange periods to equilibrate (disallow switches)
CONT	$= 0, 1$	Continuation run
0		start from one configuration file
1		start from multiple configuration files

- if CONT=1, the name specified for @conf will be split before the last "." and replica numbers inserted, e.g. input.cnf will be expanded to input_1.cnf .. input_n.cnf where n is the number of replicas
- NRESCALE≠0 requires NRET>1

ROTTRANS

RTC,RTCLAST

RTC $= 0, 1$ Turn roto-translational constraints on (1)
RTCLAST > 0 Last atom of subset to be roto-translationally constraint

- Use either centre of mass removal or roto-translational constraints but not both!

SASA

NTSASA,NTVOL,P_12,P_13,P_1X,SIGMAV,RSOlV,AS1,AS2

NTSASA 0,1 controls use of SASA implicit solvent model
0: do not use SASA [DEFAULT]
1: use SASA
NTVOL 0,1 controls use of VOLUME correction to SASA implicit solvent model
0: do not use VOLUME correction [DEFAULT]
1: use VOLUME correction (requires NTSASA = 1)
P_12 $> 0, < 1$ pair parameter for SASA reduction for first neighbours
P_13 $> 0, < 1$ pair parameter for SASA reduction for second neighbours
P_1X $> 0, < 1$ pair parameter for SASA reduction for third and higher neighbours
SIGMAV > 0 scaling parameter for volume energy term (kJ.mol^{-1}.nm^{-3})
RSOLV > 0 radius of solvent molecule for SASA calculation (nm)
AS1 > 0 an atom with SASA below this contributes to the VOLUME correction (nm^2)
AS2 > 0 an atom with SASA above this is not considered for the VOLUME correction (nm^2)

- NTSASA$\neq$0 requires NTB=0
- NTVOL= 1 requires NTSASA= 1
- Suitable values of P_12, P_13 and P_1X for the SASA and SASA/VOL implicit solvent models are given in[2] and[3]
- SIGMAV is required if NTVOL= 1. Its parameterisation is discussed in[3]
- AS1 and AS2 are required if NTVOL= 1. Atoms with AS1<SASA<AS2 have a partial contribution determined by a switching function, thus AS1 and AS2 should in most cases be close to each other and close to zero.

STEP*

NSTLIM,T,DT

NSTLIM > 0 number of steps
T ≥ 0.0 time at beginning of simulation
DT > 0.0 timestep

- Final configuration (POSITION,VELOCITY,GENBOX) written to @fin
- If NTRD$\neq$ 0, NSTLIM is the total number of configurations, T the initial time of the first file, and DT the time interval between successive records on file
- If NTEM$\neq$ 0, T and DT are irrelevant

STOCHDYN

NTSD,NTFR,NSFR,NBREF,RCUTF,CFRIC,TEMPSD

NTSD 0,1 controls stochastic dynamics mode
0: do not do stochastic dynamics [DEFAULT]
1: do stochastic dynamics
NTFR 0..3 defines atomic friction coefficients γ
0: set γ to 0.0 [DEFAULT]
1: set γ to CFRIC
2: set γ to CFRIC*GAM0
3: set γ to CFRIC*ω_i from Eq. 2-13.30
NSFR > 0 recalculate γ every NSFR steps
NBREF > 0 threshold number of neighbour atoms for a buried atom
RCUTF ≥ 0.0 interatomic distance considered when calculating γ

CFRIC ≥ 0.0 global weighting for γ
TEMPSD ≥ 0.0 temperature of stochastic bath

- NTEM$\neq 0$ requires NTSD$= 0$
- NTRD$\neq 0$ requires NTSD$= 0$
- NTISTI$\neq 0$ requires NTSD$\neq 0$
- Atomic friction coefficients GAM0 (FRICTIONSPEC) read from @friction if NTSD$\neq 0$ and NTFR$= 2$
- If NTFR$= 0$, CFRIC is irrelevant
- If NTFR$\neq 3$, NSFR, NBREF and RCUTF are irrelevant

SYSTEM*

NPM,NSM
NPM 0,1 number of solute molecules
NSM ≥ 0 number of identical solvent molecules

- NPM$= 0$ and NSM$= 0$ are not allowed simultaneously
- Data on the system topology is read from @topo
- Note that MD++ as well as GROMOS++ do not accept NPM > 1 (solute molecules have to be explicitly replicated in the topology file)

WRITETRAJ

NTWX,NTWSE,NTWV,NTWF,NTWE,NTWG,NTWB

NTWX controls writing of coordinate trajectory
0: no coordinate trajectory is written [DEFAULT]
> 0: write solute and solvent coordinates every NTWX steps
< 0: write solute coordinates every abs(NTWX) steps
NTWSE ≥ 0 selection criteria for coordinate trajectory writing
0: write normal coordinate trajectory [DEFAULT]
> 0: write minimum-energy coordinate and energy trajectory (based on the energy entry selected by NTWSE and as blocks of length NTWX)
NTWV controls writing of velocity trajectory
0: no velocity trajectory is written [DEFAULT]
> 0: write solute and solvent velocities every NTWV steps
< 0: write solute velocities every abs(NTWV) steps
NTWF controls writing of force trajectory
0: no force trajectory is written [DEFAULT]
> 0: write solute and solvent forces every NTWF steps
< 0: write solute forces every abs(NTWF) steps
NTWE ≥ 0 controls writing of energy trajectory
0: no energy trajectory is written [DEFAULT]
> 0: write energy variables every NTWE steps
NTWG ≥ 0 controls writing of free energy trajectory
0: no free energy trajectory is written [DEFAULT]
> 0: write free energy variables every NTWG steps
NTWB ≥ 0 controls writing of block-averaged energy trajectory
0: no block-averaged energy trajectory is written [DEFAULT]
> 0: write block-averaged energies (and free energies if NTWG>0) every NTWB steps

- NTWSE$\neq 0$ requires NTWX$\neq 0$, NTWV$= 0$, NTWF$= 0$, NTWE$= 0$ or abs(NTWX), NTWG$= 0$, NTWB$= 0$
- NTWSE denotes a potential energy term (Sec. 4.17)
- NTG$= 0$ requires NTWG$= 0$
- NTEM$\neq 0$ requires NTWV$= 0$
- NTRD$\neq 0$ requires NTWV$= 0$
- Coordinates (POSITIONRED) written to @trc if NTWX$\neq 0$
- Velocities (VELOCITYRED) written to @trv if NTWV$\neq 0$

- Forces (FREEFORCERED,CONSFORCERED) written to @trf if NTWF≠ 0
- Energies (ENERGY03) written to @tre if NTWE≠ 0
- Volume and pressure quantities (VOLUMEPRESSURE03) written to @tre if NTWE≠ 0
- Free energy quantities (FREEENERDERIVS03) are written to @trg if NTWG≠ 0
- Block-averaged energies and fluctuations (BAENERGY03 and BAEFLUCT03) written to @bae if NTWB≠ 0
- Block-averaged volume and pressure quantities (BAVOLUMEPRESSURE03) written to @bae if NTWB≠0
- X(t),V(t-dt/2), and F_{uc}(t) are written at the beginning of a timestep, F_c(t) right after SHAKE
- If NTWSE≠ 0, a minimum-energy trajectory is written, i.e. only the configuration and energy corresponding to the lowest NTWSE component within a block of length abs(NTWX) steps is reported

Input data are described in Vol. 5 (Program Library Manual) and Vol. 7 (Tutorials, Benchmarks, Test Sets).

Examples of MD input files are named:

∗.imd

CHAPTER 9

Output files for MD++

The data structure of the *output file* of the (simulation) programs will not be given here.

Output of programs is discussed in Vol. 5 (Program Library Manual) and Vol. 7 (Tutorials, Benchmarks, Test Sets).

Examples of MD output files are named:

∗.omd

CHAPTER 10

Files accessed by MD++ for reading or writing

Indicated are: files for reading (R), files for write-up (W) and compulsory blocks (*).

@input Standard input (control) file (R; always)
- TITLE*
- SYSTEM*
- ENERGYMIN
- STOCHDYN
- READTRAJ
- STEP*
- REPLICA
- BOUNDCOND*
- MULTICELL
- MULTIBATH
- PRESSURESCALE
- MULTIGRADIENT
- FORCE*
- COVALENTFORM
- CONSTRAINT*
- POLARISE
- INTEGRATE
- CGRAIN
- ROTTRANS
- INNERLOOP
- MULTISTEP
- PAIRLIST*
- NONBONDED*
- INITIALISE
- RANDOMNUMBERS
- COMTRANSROT
- POSITIONRES
- DISTANCERES
- DIHEDRALRES
- JVALUERES
- ORDERPARAMRES
- DISTANCEFIELD
- QMMM
- LOCALELEV
- PERSCALE
- ELECTRIC
- SASA
- PERTURBATION
- LAMBDAS
- PRINTOUT
- WRITETRAJ
- EWARN
- EDS

@out Standard output file (W; always)
MD++ output

@fin Final configuration file (W; if NTRD= 0)
TITLE
POSITION (if no SHAKE failure)
SHAKEFAILPOSITION (if SHAKE failure)
SHAKEFAILPREVPOSITION (if SHAKE failure)
VELOCITY (if NTEM= 0 and NTRD= 0)
STOCHINT (if NTSD≠ 0)
GENBOX (BOX; if NTB≠ 0)
LATTICESHIFTS (if NTB= 0)
ROTTRANSREFPOS (if NTT≠ 0 and NTCNS(J)≠ 0 for at least one J)
REFPOSITION (if NTPOR≠ 0)
DISRESEXPAVE (if NTDIR= $-1, -2$)
JVALRESEXPAVE (if NTJVR= $-1, -2$)
ORDERPARAMRESEXPAVE (if NTOPR= $-1, -2$)
ORDERPARAMRESWINAVE (if NTOPR= $1, 2$)
LEMEMORY (if NTLES≠ 0)
PERTDATA (if NTG≠ 0)

@trc Coordinate trajectory (W; if NTWX≠ 0)
TITLE
TIMESTEP
POSITIONRED
GENBOX (if NTB≠ 0)

@trv Velocity trajectory (W; if NTWF≠ 0)
TITLE
TIMESTEP
VELOCITYRED

@trf Force trajectory (W; if NTWF≠ 0)
TITLE
TIMESTEP
FREEFORCERED
CONSFORCERED

@tre Energy trajectory (W; if NTWE≠ 0)
TITLE
TIMESTEP
ENERGY03
VOLUMEPRESSURE03

@trg IOTRJG Free-energy trajectory (W; if NTWG≠ 0 and NTG≠ 0)
TITLE
TIMESTEP
FREEENERGYDERIVS03

@bae Energy block-average trajectory (W; if NTWB≠ 0)
TITLE

TIMESTEP
BAENERGY03
BAEFLUCT03

@topo Topology file (R; always)
TITLE (compulsory, first)
PHYSICALCONSTANTS (compulsory, second)
TOPVERSION (compulsory, third)
ATOMTYPENAME (compulsory)
RESNAME
SOLUTEATOM (compulsory)
CGSOLUTE
BONDSTRETCHTYPE or BONDTYPE or HARMBONDTYPE (one of them is compulsory if BONDH or BOND)
BONDH
BOND
BONDDP
BONDANGLEBENDTYPE or BONDANGLETYPE or BONDANGLEBENDTYPE (one of them is compulsory if BONDANGLEH or BONDANGLE)
BONDANGLEH
BONDANGLE
IMPDIHEDRALTYPE
IMPDIHEDRALH
IMPDIHEDRAL
TORSDIHEDRALTYPE or DIHEDRALTYPE (either of the two; compulsory if DIHEDRALH or DIHEDRAL)
DIHEDRALH
DIHEDRAL
LJPARAMETERS
CGPARAMETERS
SOLUTEMOLECULES
TEMPERATUREGROUPS
PRESSUREGROUPS
SOLVENTATOM (compulsory)
SOLVENTCONSTR
SASAPARAMETERS (if NTSASA= 1)

@conf Initial configuration (startup) file (R; always, except if NTRD= 1 and NTRB= 1)
TITLE (compulsory, first)
TIMESTEP
POSITION or POSITIONRED (either of the two; compulsory)
VELOCITY or VELOCITYRED (either of the two; if NTEM= 0 and NTRD= 0 and NTIVEL= 0)
LATTICESHIFTS (if NTB$\neq$ 0 and NTISHI= 0)
STOCHINT (if NTSD$\neq$ 0 and NTISTI= 0)
GENBOX (if NTB$\neq$ 0)
ROTTRANSREFPOS (if NTT$\neq$ 0 and NTCNS(J)$\neq$ 0 for at least one J)
REFPOSITION (if NTTPOR$\neq$ 0 and NTPORB= 0)
PERTDATA (if NTG$\neq$ 0 and NRDGL$\neq$ 0)
DISRESEXPAVE (if NTDIR= $-1, -2$ and NTDIRA$\neq$ 0)
JVALRESEXPAVE(if NTJVR= $-1, -2$ and NTJVRA$\neq$ 0)
ORDERPARAMRESEXPAVE (if NTOPR= $-1, -2$)
ORDERPARAMRESWINAVE (if NTOPR= $1, 2$)
LEMEMORY (if NTLES$\neq$ 0 and NTLESA$\neq$ 0)

@refpos Reference coordinates for position re(con)straining (R; if NTPOR≠ 0 and NTPORB= 1)
TITLE (compulsory)
REFPOSITION (compulsory)

@posresspec Atom specification for position re(con)straining (R; if NTPOR≠ 0)
TITLE (compulsory)
POSRESSPEC (compulsory)

@distrest Distance specification for distance re(con)straining (R; if NTDIR≠ 0)
TITLE (compulsory)
DISTANCERESSPEC
PERTDISRESSPEC
MDISRESSPEC
DFRESSPEC
PERTDFRESSPEC

@dihrest Dihedral specification for dihedral-angle re(con)straining (R; if NTDLR≠ 0)
TITLE (compulsory)
DIHEDRALRESSPEC
PERTDIHRESSPEC

@jval ^3J-value specification for ^3J-value restraining (R; if NTJVR≠ 0)
TITLE (compulsory)
JVALRESSPEC (compulsory)

@order S^2-value specification for S^2-order parameter restraining (R; if NTOPR≠ 0)
TITLE (compulsory)
ORDERPARAMRESSPEC (compulsory)

@qmmm QM/MM specification file (R; if NTQMMM≠ 0)
TITLE (compulsory)
QMZONE (compulsory)
QMUNIT (compulsory)
MNDOFILES (if NTQMSW= 0)
MNDOHEADER (if NTQMSW= 0)
TURBOMOLEFILES (if NTQMSW= 1)
TURBOMOLETOOLCHAIN (if NTQMSW= 1)
TURBOMOLEELEMENTS (if NTQMSW= 1)

@led Coordinate specification for local-elevation (R; if NTLES≠ 0)
TITLE (compulsory)
LOCALELEVSPEC (compulsory)

@lud LEUS biasing potential database (R; if NTLES= 2)
TITLE (compulsory)
LEUSGRID (one or more)

@friction Atomic friction coefficients for stochastic dynamics (R; if NTSD≠ 0 and NTFR= 2)
TITLE (compulsory)
FRICTIONSPEC (compulsory)

@pttopo Data determining perturbation (R; if NTG≠ 0)
TITLE (compulsory)
PERTATOMPARAM
MPERTATOM PERTATOMPAIR
PERTATOMGROUPS
PERTPOLPARAM
PERTBONDSTRETCHH
PERTBONDSTRETCH
PERTCONSTRAINT03
PERTBONDANGLEH
PERTBONDANGLE
PERTIMPROPERDIHH
PERTIMPROPERDIH
PERTPROPERDIHH
PERTPROPERDIH

@anatrj Input coordinate trajectories (R; if NTRD≠ 0)
TITLE
series of
TIME
POSITIONRED
BOX (if variable box)

CHAPTER 11

Other non-GROMOS formats

Some GROMOS programs can read non-GROMOS data and formats, e.g. protein data bank data and formats, see Vol. 5 (Program Library Manual).

CHAPTER 12

List of GROMOS blocknames

Three *categories of blocks* are distinguished:

- data blocks,
- MD input blocks,
- molecular topology blocks.

The *current GROMOS blocknames* are listed below. In addition to the following reserved names, no block may be called 'END'.

Data blocks

ACCEPTORMASS
ANGLETYPECONV
ATOMNAMELIB
ATOMTYPECONV
BFACTOR
BFACTORANISO
BONDANGLEBENDTYPECODE
BONDSTRETCHTYPECODE
BONDTYPECONV
BOX
CONSFORCE
CONSFORCERED
DIFFSTAT
DIHEDRALTYPECODE
DIHEDRALTYPECONV
DIHRESSPEC
DIPMSTAT
DISRESEXPAVE
DISRESSPEC
DISTANCERESSPEC
ENERGIES
ENERGY
ENERGY03
ENERTRJ
FILENAMES
FORMAT
FOURDIMATOMSPEC
FREE3D4DDATA
FREEENERGY3D4
FREEENERGYDERIVS03
FREEENERGYLAMBDA
FREEFORCE
FREEFORCERED
FREELAMBDADATA
FRENERTRJ
FRICTIONSPEC
GENBOX
HYDROGENMASS
IMPDIHEDRALTYPECODE
IMPROPERTYPECONV
JOBSCRIPTS
JVALUERESEPS
JVALUERESEXPAVE
JVALUERESSPEC
LEDIHSPEC
LEMEMORY
LINKADDITION
MASSATOMTYPECODE
MISCELLANEOUS
MIXEDATOMLJPAIR
MPERTATOM
MTBUILDBLEND
MTBUILDSOLUTE
MTBUILDSOLVENT
MULTIPLICITY
NOECALCSPEC
NOEGORGROMOS
NOELIB
NOESPEC
ORDERPARAMRESEXPAVE
ORDERPARAMRESWINAVE
PERTATOM
PERTATOMPAIR
PERTBONDANGLE
PERTBONDANGLEH
PERTBONDSTRETCH
PERTBONDSTRETCHH
PERTDATA
PERTDIHRESSPEC
PERTDISRESSPEC
PERTIMPROPERDIH
PERTIMPROPERDIHH
PERTPROPERDIH
PERTPROPERDIHH

POSITION
POSITION4THD
POSITION4THDRED
POSITIONFOURTHM
POSITIONOF
POSITIONRED
POSITIONSECONDM
POSITIONSECONDMT
POSITIONTHIRDM
POSRES
POSRESSPEC
QUANDISTRIB
QUANENERAVER
QUANSUMENERAVER
QUANTIMECORR
QUANTIMECORRSPE
QUANTIMESERIES
QUANTITYAVER
REFPOSITION
RESIDUENAMELIB
RUNDATA
SASASPEC
SHAKEFAILPOSITION
SHAKEFAILPREVPOSITION
SINGLEATOMLJPAIR
SOLVSTAT
STOCHINT
STOCHINT4THD
TIMESTEP
TITLE
TRANSFORM
TRICLINICBOX
VARIABLES
VELOCITY
VELOCITY4THD
VELOCITY4THDRED
VELOCITYRED
VOLUMEPRESSURE
VOLUMEPRESSURE03

MD input blocks

BOUNDCOND
CGRAIN
COMTRANSROT
CONSTRAINT
COVALENTFORM
DIHEDRALRES
DISTANCEFIELD
DISTANCERES
EDS
ELECTRIC
ENERGYMIN
EWARN
FORCE
INITIALISE
INNERLOOP
INTEGRATE
JVALUERES
LAMBDAS
LOCALELEV
MULTIBATH
MULTICELL
MULTIGRADIENT
MULTISTEP
NONBONDED
ORDERPARAMRES
PAIRLIST
PERSCALE
PERTURBATION
POLARISE
POSITIONRES
PRESSURESCALE
PRINTOUT
QMMM
RANDOMNUMBERS
READTRAJ
REPLICA
ROTTRANS
SASA
STEP
STOCHDYN
SYSTEM
WRITETRAJ

Topology blocks

ATOMTYPENAME
BOND
BONDANGLE
BONDANGLEBENDTYPE
BONDANGLEH
BONDANGLETYPE
BONDDP
BONDH
BONDSTRETCHTYPE
BONDTYPE
CGSOLUTE
CONSTRAINT
DIHEDRAL
DIHEDRALH
DIHEDRALTYPE
HARMBONDANGLETYPE

HARMBONDTYPE
IMPDIHEDRAL
IMPDIHEDRALH
IMPDIHEDRALTYPE
LJPARAMETERS
CGPARAMETERS
PATHINTSPEC
PHYSICALCONSTANTS
RESNAME
SASAPARAMETERS
SOLUTEATOM
SOLVENTATOM
SOLVENTCONSTR
SUBMOLECULES
TITLE
TOPVERSION
TORSDIHEDRALTYPE

CHAPTER 13

Recommendations for standard input and output file names

molecular building blocks	*.mtb
interaction-function parameters	*.ifp
script to run the program	*.run
input file	*.imd
output file	*.omd
configuration	*.cnf
topology	*.top
perturbation topology	*.ptp
reference positions and possibly B-factors for position re(con)straining (if in a file separate from .cnf)	*.rpr
position restraints	*.por
distance restraints	*.dsr
dihedral restraints	*.dhr
^3J-value restraints	*.jvr
S^2-order parameter restraints	*.opr
crystallographic restraints	*.xrs
LEUS database file	*.lud
local-elevation dihedrals	*.led
atomic friction coefficients	*.frc
P3M optimal influence function (G_hat)	*.ght
gromos++- specific libraries	*.lib

trajectories:

coordinates	*.trc
velocities	*.trv
forces	*.trf
energies	*.tre
free energies	*.trg
special trajectories	*.trs
block average energies	*.bae
block average free energies	*.bag

Bibliography

[1] W. Hasel, T. F. Hendrickson, and W. C. Still. A rapid approximation to the solvent accessible surface areas of atoms. *Tetra. Comput. Method.*, 1:103–116, 1988.

[2] F. Fraternali and W.F. van Gunsteren. An Efficient Mean Solvation Force Model for Use in Molecular Dynamics Simulations of Proteins in Aqueous Solution. *J. Mol. Biol.*, 256:939–948, 1996.

[3] J.R. Allison, K. Boguslawski, F. Fraternali, and W.F. van Gunsteren. A refined, efficient mean solvation force model that includes the interior volume contribution. *J. Phys. Chem. B*, 115:4547–4557, 2011.

[4] G. Nagy and C. Oostenbrink. Dihedral-based segment identification and classification of biopolymers I: Proteins. *J. Chem. Inf. Model.*, 54:266 – 277, 2014.

[5] G. Nagy and C. Oostenbrink. Dihedral-based segment identification and classification of biopolymers II: Polynucleotides. *J. Chem. Inf. Model.*, 54:278 – 288, 2014.

[6] T. Heinz and P.H. Hünenberger. A fast pairlist construction algorithm for molecular simulations under periodic boundary conditions. *J. Comput. Chem.*, 25:1474, 2004.

The GROMOS Software for (Bio)Molecular Simulation

Volume 5: Program Library Manual

January 9, 2021

Contents

CHAPTER 1

Introduction

GROMOS, consisting of MD++ and GROMOS++, is a collection of programs developed to prepare, run and analyse a MD simulation. Most programs belong to GROMOS++ and may be used to set up a simulation or analyse the trajectories of a simulation, while MD++ is used to run the simulation.

This volume gives an overview over all the programs, listed either in Chap. 2 (setup of simulations), Chap. 3 (minimizers and simulators) and Chap. 4 (analysis of trajectories) or Chap. 5 with a program description together with required and optional input arguments as well as standard and additional outputs. The focus is on the use of the programs and not on the source code behind. The reader who wishes to change or add the source code of GROMOS is referred to Chap. 6-1 where an outline of the source code including libraries as well as predefined classes and namespaces is given in more datail.

Most common arguments used and needed by a majority of GROMOS programs are explained more extensively in Sec. 1.2 of this volume. The diverse use of (atom, property and vector) specifiers, a powerful tool to specify a group of atoms, properties as distances, angles and many others, is described in Sec. 1.3, accompanied by multiple examples.

1.1. Nomenclature of GROMOS files

GROMOS is very generous concerning the names and endings of input and output files. It leaves the user absolute freedom. Nevertheless, we strongly recommend a consistent pattern of file name endings which helps keeping the overview over different file types. A possible naming (recommendations) is given in Chap. 4-13.

1.2. Common arguments in GROMOS++

Several arguments appear in many GROMOS++ programs and their explanation is given here.

1. `@topo`
   Molecular topology files are read from the `@topo` argument. The file format of a topology is described in Sec. 4-3.2.
2. `@pbc arg1 [arg2] [arg..]`
   Periodic boundary type and gathering parameters are read from `@pbc`. The first argument is the boundary type which may take the following values:
   - `v` vacuum, non-periodic boundary conditions
   - `r` rectangular periodic boundary conditions
   - `c` triclinic periodic boundary conditions
   - `t` truncated octahedral periodic boundary conditions

   The second and following arguments determine the gathering method and additional gather options. The available gathering methods (`arg2`) are:

`nog` or `0`	do not gather
`glist` or `1`	(default) gathering, based on a list of pairs of atoms the atom pair should be in the sequence: A B, where A is an atom of the molecule to be gathered, and B is an atom of the reference molecule
`gtime` or `2`	gathering based on previous frame
`gref` or `3`	gathering based on a reference structure
`gltime` or `4`	gather first frame based on a list, next frames based on previous frame
`grtime` or `5`	gather first frame based on a reference structure, next frames based on previous frame
`gbond` or `6`	gathering based on bond connectivity
`cog` or `7`	gathering with respect to the centre of geometry of all atoms of the first molecule in the system
`gfit` or `8`	gather selected molecules based on a reference structure which has been superimposed on the first frame of the trajectory, gather remaining molecules to the cog of selected molecules

Further arguments are necessary or optional for specific gathering methods:

`list <atom pair list>`	e.g. `@pbc` r gref refg coord.cnf
`refg <ReferenceStructure>`	e.g. `@pbc` r gltime list 2:res(15:CA) 1:11 3:134 1:11
`molecules <molecule numbers>`	e.g. `@pbc` r gfit refg coord.cnf molecules 1-5

3. `@outformat`
   Some GROMOS++ programs can write the output coordinates in different formats. The following formats are supported:

`cnf`	Configuration format containing POSITION blocks (extension .cnf).
`trc`	Coordinate trajectory format containing POSITIONRED blocks (extension .trc).
`por`	Position restraints specification format (extension .por).
`pdb`	Protein Data Bank (PDB) format. An additional factor can be given to convert the length unit to Å, default 10.0 (nm to Å).
`vmdam`	VMD's „Amber Coordinates" format. An additional factor can be given to convert the length unit to Å, default 10.0 (nm to Å).

1.3. Atom, property and vector specifiers in GROMOS++

Analysis of the trajectories of a MD simulation is, besides the correct setup, of importance to a computational scientist. The more specific the questions, the more flexible the programs which answer those questions have to be. GROMOS++ makes use of three specifiers to keep its flexibility: atom specifiers, property specifiers and vector specifiers. Each of them is used as an input parameter (a string with a well defined format) for some GROMOS++ programs. Note that some shells may modify the brackets or other special characters. Therefore, quotes should be used when using atom specifiers as a command line argument.

This section introduces the reader to each of the three specifiers giving an overview of the different possibilities to use them.

1.3.1. Atom specifiers. Atom specifiers define a general way to access specific atoms of a system. It is even possible to access atoms which are not there (virtual atoms) or common properties (e.g. the center of geometry or center of mass) of multiple atoms. The atom specifier can be defined using four different formats:

- Molecules and Atoms:
  `<mol>[-<mol>]:<atom>[-<atom>]`
- Residues:
  `<mol>[-<mol>]:res(<residue>:<atom>[,<atom>...])`
- Virtual Atoms:
  `va(<type>, <atomspec>)`

- File:
  `file(<filename>)`

`<mol>` is the molecule number and `<atom>` either the atom number or the atom name. Instead of the atom or molecule number one can also specify all solute or solvent molecules or atoms using `a` or `s`, respectively. The solvent is only accessible if there is a topology and a coordinate file given. It is not possible to access the solvent from a topology only, since GROMOS does not know how many solvent molecules the system consists of. The `<type>` argument defines the virtual atom type the specifier accesses. The following types are known in GROMOS++:

0: explicit/real atom
1: aliphatic CH_1 group
2: aromatic CH_1 group
3: non-stereospecific aliphatic CH_2 group (pseudo atom)
4: stereospecific aliphatic CH_2 group
5: single CH_3 group (pseudo atom)
6: non-stereospecific CH_3 groups (isopropyl; pseudo atom)
7: non-stereospecific CH_3 groups (tert-butyl; pseudo atom)
-1: centre of geometry
-2: centre of mass

`<atomspec>` is a complete atom specifier of one of the four formats above and `<filename>` is the name of the file (output of the `atominfo` program) listing the atoms to consider.

Multiple atom specifiers must be separated by a semicolon while multiple molecules or atoms within the same atom specifier are separated by a comma.

Examples:

atoms 3 an 7 to 12 of molecule 2:

```
2:3,7-12
```

all atoms of molecule 1:

```
1:a
```

all CA, N or C atoms of all molecules

```
a:CA,N,C
```

atom 1 of residue 3 and 5 of molecule 1:

```
1:res(3,5:1)
```

all C, N or CA atoms of residues named SER or THR of molecule 1:

```
1:res(SER,THR:C,N,CA)
```

the centre of mass of molecule 1:

```
va(-2,1:a)
```

the alpha hydrogen of residue 1 in a protein:

```
va(1,1:res(1:CA,N,CB,C))
```

all CA atoms of the first molecule, accessed by a file from atominfo:

```
\$ atominfo @topo ex.topo @atomspec 1:CA > ca.spec
file(ca.spec)
```

In addition, there are the two keywords `not` and `minus` to exclude some atoms from an atom specifier. Note that the atoms specified with the keyword `not` are never included in the resulting atom specifier while the keyword `minus` allows to add the removed atoms within the same specifier later on again.

Examples:

all atoms of the first molecule but without the second residue:

```
1:a minus(1:res(2:a))
```

all atoms of the first molecule but without any C atoms:

```
1:a minus(1:res(2:a)) 1:res(2:C)
```

Finally there are programs that syntactically require an atom specifier although one does not want to specify any atoms for the computation (e.g. to disable rotational fits). For this purpose there is the keyword `no`. It can be used to specify an empty set of atoms.

Example:
no atoms:

```
no
```

1.3.2. Vector specifiers. Vector Specifiers may be used in a property specifier (see Sec. 1.3.3) to calculate some well defined properties using the help of vectors. There are three different formats to specify a vector:

- `cart(<x>,<y>,<z>)`
- `polar(<r>,<`α`>,<`β`>)`
- `atom(<atomspec>)`

with `<x>`, `<y>` and `<z>` being the three coordinates of a Cartesian 3D vector, `<r>` the length (norm) of a vector $\mathbf{x}$, `<`α`>` and `<`β`>` the polar angles in degree,

$$\mathbf{x} = (\mathbf{r}\cos(\alpha)\cos(\beta), \mathbf{r}\sin(\alpha), -\mathbf{r}\cos(\alpha)\sin(\beta)) \quad , \tag{1.1}$$

and `<atomspec>` is an atom specifier (see Sec. 1.3.1).

Examples:

the vector vector $(2, 5, 1)$:

```
cart(2,5,1)
```

the vector $(0, 2.5, 0)$

```
polar(2.5,45.0,90.0)
```

1.3.3. Property specifiers. Like the other two specifiers (see Sec. 1.3.1 and Sec. 1.3.2), property specifiers are used as an argument for some analysis programs of GROMOS++. They are used to specify the property one is interested in. The general format of a property specifier is:

- `<type>%<arg1>[%<arg2>...]`

with `<type>` defining the specific property and `<arg1>` the argument (an atom or vector specifier, compare Sec. 1.3.1 and Sec. 1.3.2, respectively) needed to calculate this property. It is clear that dependent on the property type the number of required arguments may change.

The following is a list of all property types implemented in GROMOS++:

d: distance
a: angle
t: torsion
tp: periodic torsion
ct: cross torsion
hb: hydrogen bond
st: stacking
o: order
op: order parameter
pr: pseudo rotation
pa: pucker amplitude
expr: expression property

Multiple calculations of the same property at different positions of the molecule or system are available via substitutions (see example 4 where it is shown to calculate multiple distances within the same molecule).

Some examples for all properties but the `expr` property follow. The latter is a bit more complex and will be discussed after the examples at the end of Sec. 1.3.1.

Examples:

the distance between atoms 1 and 2 of molecule 1:

```
d%1:1,2
```

the distance between the first atoms of molecule 1 and 2:

```
d%1:1;2:1
```

the distance between the centres of mass of molecules 1 and 2:

```
d%va(com,1:a);va(com,2:a)
```

the distances between the H and N atoms of residues 3 to 5 of molecule 1 (making use substitution):

```
d%1:res((x):H,N)|x=3-5
```

the angle defined by atoms 1, 2 and 3 of molecule 1:

```
a%1:1-3
```

the angle between the virtual alpha hydrogen of residue 1 and the atoms CA and C of residue 1 of molecule 1:

```
a%va(1,1:res(1:CA,N,CB,C));1:res(1:CA,C)
```

the torsion defined by the atoms H, N, CA and CB of residue 2 of molecule 1:

```
t%1:res(2:H,N,CA,CB)
```

the cross torsion defined by the atoms 2, 3, 4 and 5 as well as 3, 4, 5 and 6 of molecule 1:

```
t%1:1,2,3,4%1:3,4,5,6
```

the hyrdogen bond between the H atom of residue 3 and the O atom of residue 5 of the first molecule:

```
hb%1:res(3:N,H);1:res(5:O)
```

the stacking between the HISB ring of residue 44 of molecule 1 and and the pyrimidine ring of residue 2 of of molecule 2:

```
st%1:res(44:CG,CE1,CE2,CD1,CD2,CZ)%2:res(1:N1,C5,N3,C6,C2)
```

the order between the N-H bond of residue 2 and the z-axis:

```
o%atom(1:res(2:H,N))%cart(0,0,1)
```

the order parameter between the x-axis and the vector defined by atoms 1 and 2 of the first molecule:

```
op%cart(1,0,0)%atom(1:1,2)
```

the pseudo rotations of the atoms in the furanose ring of residues 1 to 6 of molecule 1:

```
pr%1:res((x):C1*,C2*,C3*,C4*,O4*)|x=1-6
```

the pucker amplitude of the atoms in the furanose ring of residue 1 of molecule 1:

```
pa%1:res(1:C1*,C2*,C3*,C4*,O4*)
```

The **expression property** allows the evaluation of a multitude of expressions over a trajectory. The general form is:

- `expr%<f1>(<args1>) <op> <f2>(<args2>)`

where `<op>` is one of the following arithmethic or logical operators:

`+`	addition
`-`	subtraction
`*`	multiplication
`/`	division
`!`	not
`==`	equals
`!=`	does not equal
`>`	bigger than
`<`	smaller than
`>=`	bigger or equal than
`<=`	smaller or equal than
`&&`	logical and

|| logical or

<f1> and <f2> are functions followed by their arguments. Depending on the type of function, the arguments are different. There are the following funcitons:

- functions where the argument is a scalar:
- `sin` the sine function
- `cos` the cosine function
- `tan` the tangens function
- `asin` the inverse sine function
- `acos` the inverse cosine function
- `atan` the inverse tangens function
- `exp` the exponential function
- `ln` the logarithm
- `abs` the absolute value
- `sqrt` the square root
- functions where the argument is a vector:
- `abs` the norm of a vector
- `abs2` the squared norm of a vector
- functions which need two vectors:
- `dot` the dot product of two vectors
- `cross` the cross product of two vectors
- `ni` the nearest image of vector 1 with respect to vector 2

Examples:

calculates the dot product between position of atom(1:1) and the vector (0,0,1); that is the z-component of the position of the first atom of the first molecule:

```
expr%dot(atom(1:1),cart(0,0,1))
```

calculates the distance between the first atom of the first and second molecule. First, the nearest image of atom(2:1) according to atom(1:1) is calculated. Second, this vector is substracted from atom(1:1) and the absolute value is taken:

```
expr%abs(atom(1:1) - ni(atom(2:1), atom(1:1)))
```

returns 1 if the the discussed distance is below 1.0 nm and 0 if not:

```
expr%abs(atom(1:1) - ni(atom(2:1), atom(1:1)))<1.0
```

calculates the order of the vector defined by atoms 1:1,2 and the z-axis. First, the cosine is calculated by the dot product of the vectors and division by their lengths (the second vector has length 1). Then the angle is calculated and converted to degrees:

```
expr%acos(dot(atom(1:1,2),cart(0,0,1)) / abs(atom(1:1,2)))*(180/3.1415)
```

CHAPTER 2

Setup of simulations (preprocessing)

2.1. `bin_box` (GROMOS++ program)

Program description:

When simulating a molecular liquid, a starting configuration for the solvent molecules has to be generated. To generate a starting configuration for the simulation of a binary mixture, the program `bin_box` can be used. A cubic box is filled with solvent molecules by randomly distributing them on an evenly spaced grid such that the total density of the box and the mole fractions of the solvent components match the specified values. Note that the program `ran_box` (see Sec. 2.25) can be used alternatively, which generates a starting configuration for the simulation of mixtures consisting of an unlimited number of components, in which the molecules are oriented randomly.

Required input arguments	
`@topo1`	⟨molecular topology file for a single molecule of the type of mixture component 1⟩
`@pos1`	⟨input coordinate file for a single molecule of the type of mixture component 1⟩
`@topo2`	⟨molecular topology file for a single molecule of the type of mixture component 2⟩
`@pos2`	⟨input coordinate file for a single molecule of the type of mixture component 2⟩
`@nsm`	⟨total number of molecules per dimension (NSM)⟩
`@densit`	⟨density of the binary mixture (kg/m^3)⟩
`@fraction`	⟨mole fraction of mixture component 1⟩

Optional input arguments

none

Standard output

coordinate file with NSM^3 molecules in a cubic box at the specified density.

Additional output

none

2.2. `build_box` (GROMOS++ program)

Program description:

When simulating a molecular liquid, a starting configuration for the solvent molecules has to be generated. Program `build_box` generates a cubic box filled with identical solvent molecules which are put on an evenly spaced grid such that the density of the box matches the specified value. Note that to generate a starting configuration for the simulation of a binary mixture, the program bin_box can be used (see Sec. 2.1). Alternatively, program `ran_box` (see Sec. 2.25) generates a starting configuration for the simulation of mixtures consisting of an unlimited number of components, in which the molecules are oriented randomly.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2) for a single molecule⟩
`@pos`	⟨input coordinate file for a single molecule⟩
`@nsm`	⟨number of molecules per dimension (NSM)⟩
`@dens`	⟨density of the liquid (kg/m^3)⟩

Optional input arguments

none

Standard output

coordinate file with NSM^3 solvent molecules in a cubic box at the specified density

Additional output

none

2.3. `check_box` (GROMOS++ program)

Program description:

To check for the distances between atoms and periodic copies of the other atoms in the system, program `check_box` can be used. `check_box` calculates and writes out the minimum distance between any atom in the central box of the system and any atom in the periodic copies (rectangular box and truncated octahedron are supported).

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2) of the system⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj` ⟨input coordinate (trajectory) file⟩

Optional input arguments

`@atoms` ⟨atoms to include in calculation (default: all solute)⟩

Standard output

time series of the shortest distance between periodic copies of the selected atoms, followed by the overall minimum over the simulation

Additional output

none

2.4. `check_top` (GROMOS++ program)

Program description:

Making a correct topology is one of the most important requirements for doing a successful simulation. `check_top` helps to remove often made errors from a topology in three ways. First, it runs some standard tests on the molecular topology and warns if something unexpected is observed in the topology. Second, it can calculate all bonded interaction energies for a given set of coordinates to determine the compatibility of the topology with the coordinates. Third, it can check for consistency in the force-field parameters by comparing it to a specified set of building blocks and force-field parameters.

In the first phase, `check_top` tests that:

1. there is maximally one bond defined between any pair of atoms
2. no atom appears twice in the definition of one given bond
3. only bond types are used that are defined in the topology
4. a bond angle is defined for the atoms involved in any two bonds sharing one atom
5. there is maximally one bond angle defined for a given set of three atoms
6. atoms involved in a bond angle definition are bound to the central atom
7. no atom appears twice in the definition of one given bond angle
8. only bond angle types are used that are defined in the topology
9. an improper dihedral angle is defined centered on every atom that is bound to exactly three other atoms
10. there is maximum one improper dihedral angle defined for any set of four atoms
11. atoms involved in an improper dihedral angle definition are bound
12. no atom appears twice in the definition of one given improper dihedral angle
13. only improper dihedral types are used that are defined in the topology
14. atoms involved in a proper dihedral angle are sequentially bound
15. no atom appears twice in the definition of one given dihedral angle
16. only dihedral angle types are used that are defined in the topology
17. only atom types are used that are defined in the topology
18. the sum of partial charges on atoms in one charge group is an integer value
19. excluded atoms are 1,2- or 1,3- or 1,4-neighbours
20. atoms only have atoms with a higher sequence number in their exclusion list
21. 1,2- or 1,3-neighbours are excluded
22. 1,4-exclusions are separated by 3 bonds (1,4-neighbours)
23. atoms only have atoms with a higher sequence number in their 1,4-exclusion list
24. 1,4-neighbours are in the exclusion list or in the 1,4-exclusion list
25. no exclusions or 1,4-exclusions are defined for the last atom in the topology
26. the charge group code of the last atom in the topology is 1

Additionally, for atoms that are 1,4 neighbours but are listed as excluded atoms a warning is printed. This is usually only the case if an aromatic group is involved. Note that a topology that passes all these tests is by no means guaranteed to be error-free. Conversely, some of these tests are merely meant as warnings for the user which may point at errors in the majority of cases. In some cases, the user may very well want to use a topology that does not fulfill all tests.

In the second phase, potential energies of all bonds, bond angles, improper dihedral angles and proper dihedral angles are calculated and written out. Abnormally large energies or deviations from ideal values may indicate an error in the topology, or an inconsistent set of coordinates with the topology. See program `shake_analysis` (see Sec. 5-5.6) for a similar check on the non-bonded interaction energies.

In the third phase `check_top` can compare the topology with other building blocks in a specified molecular topology building block file and the corresponding interaction function parameter file. It checks if in the molecular topology building block file we observe one of the following:

1. other atoms with the same name and the same integer atom code (IAC)
2. other atoms with the specified IAC
3. other atoms with the same IAC and mass
4. other atoms with the same IAC and charge
5. other bonds between atoms of the same IAC with the same bond type
6. other bond angles between atoms of the same IAC with the same bond-angle type

7. other improper dihedral angles between atoms of the same IAC with the same improper dihedral type
8. other dihedral angles between atoms of the same IAC with the same dihedral-angle type

In cases where the parameters specified in the program are not observed anywhere else, or when they are not the most common parameter, the program prints out a list of possible alternatives. Again, we stress that `check_top` only points at possible inconsistencies and does not necessarily indicate errors in your topology.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩

Optional input arguments

`@coord` ⟨coordinate file for energy calculation⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@build` ⟨building block file for consistency check⟩
`@param` ⟨parameter file for consistency check⟩

Standard output

list of inconsistencies

Additional output

none

2.5. com_top (GROMOS++ program)

Program description:

To generate molecular topology files for the use in simulations of e.g. (macro) molecular complexes, or mixtures containing several solutes and/or (co)solvents, it is usually convenient to merge existing molecular topology files. Program com_top combines multiple topologies into one new topology.

The user has to specify which molecular topologies are to be merged, and from which file the force-field parameters and the solvent have to be taken. The resulting molecular topology file is written out to the standard output.

The program can also be used for topology file format conversion. The argument @inG96 converts GROMOS96 topologies to the current format. On the other hand @outG96 converts topologies in the current format to GROMOS96 format.

Required input arguments

@topo	⟨molecular topology files (see Sec. 1.2)⟩
@param	⟨index number of molecular topology file to take parameters from⟩
@solv	⟨index number of molecular topology file to take solvent from⟩

Optional input arguments

@inG96	⟨reads in a topology file in GROMOS96 format⟩
@outG96	⟨the output topology is written in the GROMOS96 format⟩

Standard output

combined topology file

Additional output

none

2.6. con_top (GROMOS++ program)

Program description:

A molecular topology file in which a system is described by a specific version of a force-field parameter set can be converted into a molecular topology file with interaction parameters from a different force-field version, using the program con_top.

An interaction function parameter file has to be specified that corresponds to the force-field version into which the molecular topology should be converted. con_top checks whether the topology is not referring to atom, bond, bond angle or (improper) dihedral types that are not defined in the parameter file. If type numbers of atoms, bonds, bond angles, etc. change with the force-field parameter set, a renumbering file (see Sec. 4-7.2 for its formats) can be given to specify these changes.

Required input arguments

@topo	⟨molecular topology file (see Sec. 1.2) to be converted⟩
@param	⟨interaction function parameter file⟩
@renum	⟨renumbering file⟩

Optional input arguments

none

Standard output

converted molecular topology file

Additional output

none

2.7. copy_box (GROMOS++ program)

Program description:

Program `copy_box` can be used to duplicate the size of a system in the x, y or z direction (or **a**, **b**, **c** for triclinic boxes). This is especially convenient if one wants to double the size of a system under periodic boundary conditions in which the central box has a rectangular or triclinic shape. If one wants to perform more elaborate transformations, the program `cry` might be of use (see Sec. 2.8). Note that program `com_top` (see Sec. 2.5) can be useful to additionally duplicate the solute block in the topology. The `@pbc` flag is optional. Only if this flag is given, gathering of the molecules will be performed before copying the box.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2) of the system⟩
`@pos` ⟨input coordinate file⟩
`@dir` ⟨direction in which the coordinates are to be duplicated (x,y,z)⟩

Optional input arguments

`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩

Standard output

coordinate file containing the multiplied system

Additional output

none

2.8. `cry` (GROMOS++ program)

Program description:

When using periodic boundary conditions, the computational box containing the molecular system is treated as being translationally invariant. So, periodic boundary conditions can also be used when simulating a crystal as long as the unit cell, or a number of adjacent unit cells is used as computational box. Unless the asymmetric unit is translationally invariant, it cannot be used as computational box. Since crystallographic coordinates of molecular systems are generally only provided for the molecules in one asymmetric unit, the coordinates of the other molecules in the unit cell or cells are to be generated by crystallographic symmetry transformations.

The program `cry` can rotate and translate copies of a system to create a crystal unit cell. Based on a topology and initial (gathered) coordinates, as well as a specification file for the symmetry transformations. A conversion factor can also be given if the translation vector is specified in different units than the coordinates. A coordinate file is generated that contains coordinates for as many systems as there are transformations defined in the specification file. The corresponding topology can easily be generated using the program `com_top` (Sec. 2.5).

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pos`	⟨input coordinate file for the molecules⟩

Optional input arguments

`@spec`	⟨specification file for the symmetry transformations⟩
`@factor`	⟨conversion factor for distances⟩
`@spacegroup`	⟨space group symbol⟩
`@cell`	⟨cell edge lengths [nm] and angles [degrees]⟩
`@keepbox`	⟨no expansion of the initial box⟩

Standard output

coordinates for a crystal unit cell

Additional output

none

2.9. `duplicate` (GROMOS++ program)

Program description:

Program duplicate searches for duplicated atoms, i.e. atoms having the same coordinates as another atoms. If requested, a coordinate file with the duplicated molecules removed is written out.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pos` ⟨input coordinate file for the molecules⟩

Optional input arguments

`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@write` ⟨write out filtered coordinates⟩

Standard output

A list of molecules containing duplicated atoms. The coordinates filtered for duplicated if requested.

Additional output

none

2.10. `explode` (GROMOS++ program)

Program description:

Program `explode` takes a box with `nsm` molecules and puts them on a grid with distance `dist` between the grid points. This tool is useful in case a vacuum simulation has to be performed by simulating `nsm` molecules at a large intermolecular distance. The input topology and coordinate files must contain at least `nsm` molecules. If there are less molecules than grid positions a message is printed to inform the user. As input coordinates, one can for instance take a liquid box generated by program `build_box` (see Sec. 2.2).

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2) of the system⟩
`@pos` ⟨input coordinate file⟩
`@nsm` ⟨number of solute molecules⟩
`@dist` ⟨distance to put between molecules ⟩

Optional input arguments

none

Standard output

coordinate file containing the coordinates of molecules that were put at larger intermolecular distances

Additional output

none

2.11. gca (GROMOS++ program)

Program description:

Sometimes, one may want to modify a specified molecular configuration such as to obtain specified values of bond lengths, bond angles or dihedral angles. Program `gca` allows the user to do this. In addition, series of configurations can be generated in which the molecular properties of choice are modified stepwise. If more than one property to be changed has been specified, configurations for all combinations of values will be generated, allowing for a systematic search of the property space.

In order to fulfill the requested property values, program `gca` will

- for a bond length between atoms i and j, shift all atoms connected to j (not i) and onwards;
- for an bond angle defined by atoms i,j and k rotate all atoms connected to k (not j) and onwards around the axis through atom k and perpendicular to the i,j,k-plane;
- for a dihedral angle defined by atoms i,j,k and l rotate all atoms connected to k and l (not j) around the axis through atoms j and k.

This procedure may lead to distortions elsewhere in the molecule if the atom count is not roughly linear along the molecular structure, or if the specified properties are part of a cyclic structure. The program does not check for steric clashes resulting from the modifications.

The properties to be modified are specified through a property specifier (see Sec. 1.3.3), followed by either one additional argument (single value to be specified) or three additional arguments (to generate a range of values).

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@prop` ⟨property specifier (see Sec. 1.3.3): properties to change⟩
`@traj` ⟨input coordinate file⟩

Optional input arguments

`@outformat` ⟨output coordinates format, see Sec. 1.2⟩

Standard output

molecular configurations in the requested format

Additional output

none

2.12. gch (GROMOS++ program)

Program description:

In the standard GROMOS force fields, part of the hydrogen atoms (polar, aromatic) are explicitly treated, whereas other hydrogen atoms (aliphatic, some aromatic) are implicitly treated by incorporation into the (carbon)atom to which they are attached. Depending on the presence or absence of hydrogen atom coordinates in a molecular configuration file, hydrogen atom coordinates may have to be recalculated.

Program `gch` calculates optimal positions for hydrogen atoms for which the connecting bond shows a relative deviation from the zero-energy value larger than a user specified threshold. Coordinates for all hydrogen atoms that are explicitly listed in the topology should already be contained in the coordinate file. Program `pdb2g96` (see Sec. 2.19) e.g. will include atoms for which no coordinates were present in the pdb file with coordinates set to zero. If defined, `gch` uses topological information on bonds, bond angles and dihedral angles to place hydrogen atoms at the optimal location. In cases where the necessary angular parameters are not provided in the topology, `gch` uses 109.5° for tetrahedral centers and 120° for planar centers.

Eight types of geometries can be handled when generating hydrogen atom coordinates:

1. An atom (a) is bonded to one hydrogen (H) and one other heavy atom (nh). A fourth atom (f) is searched for which is bound to nh and preferably used to define the dihedral around the nh-a bond. The coordinates of H are generated in such a way that the dihedral f-nh-a-H is trans and that the angle nh-a-H and bond length a-H correspond to their minimum energy values.
2. An atom (a) is bonded to one hydrogen (H) and two other heavy atoms (nh1 and nh2). The coordinates of H are generated to be in the plane through nh1, nh2 and a, on the line bisecting the nh1-a-nh2 angle and with an a-H bond length corresponding to the minimum energy value in the topology, such that the nh1-a-H and nh2-a-H angles are larger than 90 degrees.
3. An atom (a) is bonded to two hydrogens (H1 and H2) and one other heavy atom (nh). A fourth atom (f) is searched for which is bound to nh and preferably is used to define the dihedral around the nh-a bond. The coordinates of H1 are generated in such a way that the dihedral f-nh-a-H1 is trans and that the angle nh-a-H1 and bond length a-H1 correspond to their minimum energy values. The coordinates of H2 are generated to have the angles nh-a-H2 and H1-a-H2 as well as the bond length a-H2 at their minimum energy values. If this does not result in a planar configuration around a, the improper dihedral a-nh-H1-H2 will be positive.
4. An atom (a) is bonded to three hydrogens (H1, H2 and H3) and one other heavy atom (nh). A fourth atom (f) is searched for which is bound to nh and preferably is used to define the dihedral around the nh-a bond. The coordinates of H1 are generated in such a way that the dihedral f-nh-a-H1 is trans and that the angle nh-a-H1 and bond length a-H1 correspond to their minimum energy values. The coordinates of H2 are such that the angles nh-a-H2 and H1-a-H2 and the bond length a-H2 are at their minimum energy values, and the improper dihedral a-nh-H1-H2 is positive. The coordinates of H3 are such that the angles nh-a-H3 and H1-a-H3 and the bond length a-H3 are at their minimum energy values and the improper dihedral a-nh-H1-H3 has a negative value.
5. An atom (a) is bonded to one hydrogen atom (H) and three other heavy atoms (nh1, nh2, nh3). The coordinates of H are generated along the line going through atom a and a point corresponding to the average position of nh1, nh2 and nh3, such that the bond length a-H is at its minimum energy value and the angles nh1-a-H, nh2-a-H and nh3-a-H are larger than 90 degrees.
6. An atom (a) is bonded to two hydrogen atoms (H1 and H2) and two other heavy atoms (nh1 and nh2). The coordinates of H1 and H2 are placed above and below the plane going through atoms nh1, nh2 and a, in such a way that the a-H1 and a-H2 bond lengths and the H1-a-H2 bond angle are at their minimum energy values. The improper dihedral angle a-nh1-nh2-H1 will be positive.
7. An atom (a) is bonded to two hydrogen atoms (H1 and H2), but to no heavy atoms. This is likely to be a (crystallographic) water molecule. First a molecule is generated having the a-H1 aligned in the z-direction and the a-H2 in the z-y plane with the angle H1-a-H2 and bond lengths a-H1 and a-H2 according to their minimum energy values. This molecule is then rotated around x, y and z by three random angles.
8. An atom (a) is bonded to four hydrogen atoms (H1, H2, H3 and H4), but to no heavy atoms. A molecule is generated with all bond lengths at their minimum energy values, the a-H1 aligned in the z-direction, H2 in the x-z plane and H3 such that the improper a-H1-H2-H3 is positive and H4 such that the improper a-H1-H2-H4 is negative. The complete molecule is then rotated by three random angles around x, y and z.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩

`@pos` ⟨input coordinate file⟩

Optional input arguments

`@tol` ⟨tolerance (default 0.1 %)⟩

`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩

Standard output

coordinates of the molecular system in which the coordinates of all hydrogen atoms that displayed a relative deviation larger than the specified tolerance have been repositioned

Additional output

none

2.13. `ion` (GROMOS++ program)

Program description:

When simulating a charged solute in solution, one may wish to include counter-ions in the molecular system in order to obtain a neutral system, or a system with a specific ionic strength. The program `ion` can replace solvent molecules by atomic ions by placing the ion at the position of the first atom of a solvent molecule. Substitution of solvent molecules by positive or negative ions can be performed by selecting solvent positions with the lowest or highest Coulomb potential, respectively, or by random selection. In order to prevent two ions being placed too close together, a sphere around each inserted ion can be specified from which no solvent molecules will be substituted by additional ions. In addition, the user can specify specific water molecules that should not be considered for replacement.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@pos`	⟨input coordinate file⟩

Optional input arguments

`@positive`	⟨number⟩ ⟨ionname⟩ ⟨residue name (optional)⟩
`@negative`	⟨number⟩ ⟨ionname⟩ ⟨residue name (optional)⟩
`@potential`	⟨cutoff for potential calculation⟩
`@random`	⟨random seed⟩
`@exclude`	⟨atom specifier (see Sec. 1.3.1): solvent molecules to be excluded⟩
`@mindist`	⟨minimum distance between ions⟩

Standard output

Coordinates of the molecular system in which the specified number of ions replace solvent molecules

Additional output

none

2.14. `link_top` (GROMOS++ program)

Program description:

For branched systems, it may be very cumbersome to create crosslinks in a topology. Program link_top allows the user to apply a pre-defined link to the topology. The link is defined in a special building block file, which contains (after a TITLE block), MTBUILDBLLINK blocks. This block has the following layout:

```
MTBUILDBLLINK
# RNME
XYZ
# number of atoms
7
#ATOM RES ANM IACM MASS CGMICGM MAE MSAE
1 1 CA 14 13.018 0.00000 1 2 2 6
2 1 CB 15 14.027 0.16000 1 2 5 6
3 1 OG 0 0 0.00000 1 0
4 1 HG 0 0 0.00000 1 0
5 2 CB 15 14.027 0.16000 0 1 6
6 2 OG 4 15.994 -0.32000 1 0
7 2 HG 0 0 0.00000 1 0
# bonds
# NB
2
# IB JB MCB
1 2 1
2 6 12
# bond angles
# NBA
2
# IB JB KB MCB
1 2 6 12
2 6 5 12
# improper dihedrals
# NIDA
0
# IB JB KB LB MCB
# dihedrals
# NDA
3
# IB JB KB LB MCB
0 1 2 6 1
1 2 6 5 1
2 6 5 0 1
END
```

The atoms section of the building block contains all atoms that are involved in the link. The second column specifies that these atoms are to be found in the first or second residue of the link. The atoms are identified in the original topology by the residue sequence number indicated in the input (`@linking`) and the name of the atom according to the MTBUILDBLLINK.

In a first step, link_top, removes all atoms for which the IAC is 0. All references to these atoms (exclusions, bonds, angles, etc.) are removed from the topology. Next, the remaining atoms in the link defition get updated: the values in the topology of IAC, mass, charge, and charge group get replaced by whatever is indicated in the MTBUILDBLLINK block. The exclusions of the original topology (without the removed atoms) remain, and the exclusions that are specified in the MTBUILDBLLINK block are added.

Covalent interactions that need to be changed are also specified in the MTBUILDBLLINK block. The program only allows the user to specify bonds, angles and improper dihedral angles that are referring to

atoms that are all part of the link specification. Any bonds, angles and improper dihedral angles that were present in the topology for these atoms will be removed and the newly defined interactions are added to the topology. For dihedral angles, the program allows the user to refer to the first and/or last atom to be represented by a number 0. For these atoms, the program will search in the topology for an atom that is bound to the second or third atom, respectively and assign the dihedral angle to this atom. Any dihedral angles that were already defined for this group is replaced. Multiple dihedral angles for the same set of four atoms may be added.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@links`	⟨file containing the MTBUILDBLLINK blocks⟩
`@linking`	⟨residue sequence number⟩ ⟨residue sequence number⟩ ⟨name of link⟩

Optional input arguments

none

Standard output

molecular topology file

Additional output

none

2.15. `make_pt_top` (GROMOS++ program)

Program description:

Program `make_pt_top` takes two or more molecular topologies and writes the differences in the perturbation topology format. Both topologies must contain the same number of atoms. The softness parameters α_{LJ} and α_C can be specified by an input parameter.

Required input arguments

`@topo` ⟨multiple molecular topology files (see Sec. 1.2)⟩

`@softpar` ⟨softness parameters (α_{LJ} and α_C)⟩

Optional input arguments

none

Standard output

perturbation topology, containing the parameter differences.

Additional output

none

2.16. `make_sasa_top` (GROMOS++ program)

Program description:

Program `make_sasa_top` adds the atom-specific information required to use the SASA/VOL implicit solvent model to the molecular topology file. It reads in an existing molecular topology file created using `make_top` (see Sec. 2.17), along with a SASA/VOL specification library file, which contains the atom-specific SASA/VOL parameters. The specification library file must be for the same force field as was used to create the molecular topology file. The inclusion of hydrogen atoms in the calculation of the SASA during the simulation may also be specified.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@sasaspec` ⟨SASA/VOL specification library file⟩

Optional input arguments

`@noH` ⟨do not include hydrogen atoms (default: include)⟩

Standard output

molecular topology file with appended SASA/VOL information

Additional output

none

2.17. `make_top` (GROMOS++ program)

Program description:

Program `make_top` builds a molecular topology from a building block sequence. `make_top` reads in a molecular topology building-block file (e.g. `mtb53a6.dat`) and an interaction function parameter file (e.g. `ifp53a6.dat`), and gathers the specified building blocks to create a topology. Cysteine residues involved in disulfide bridges as well as heme and coordinating residues involved in covalent bonds to the iron atom have to be explicitly specified. Topologies for cyclic sequences of building blocks can be generated using `@cyclic`.

Required input arguments	
`@build`	⟨molecular topology building block file⟩
`@param`	⟨interaction function parameter file⟩
`@seq`	⟨sequence of building blocks in the solute⟩
`@solv`	⟨building block for the solvent⟩

Optional input arguments	
`@cys`	⟨cys1⟩–⟨cys2⟩ . . . ⟨cys1⟩–⟨cys2⟩
`@heme`	⟨residue sequence number⟩ ⟨heme sequence number⟩

Standard output

molecular topology file

Additional output

none

2.18. mk_script (GROMOS++ program)

Program description:

A MD simulation is usually performed by executing a small script that combines all the necessary files and redirects the output to the appropriate places. When simulations are performed on a queue, such scripts become indispensable. Additionally, in many simulation projects the user prepares similar input files and scripts over and over again. Program mk_script can either create a series of similar scripts that run a sequential list of simulations (@script) or it can create scripts for a more complex set of simulations that perform a specific task (start-up, perturbation; @joblist). Scripts for special cases such as REMD simulations (@remd) can also be written.

GROMOS does not require specific filenames for specific types of files. However, most users find it useful to retain some order in their filenames. mk_script has a standard way of constructing filenames that depends on the script number and the system name. The user can specify another set of rules to create filenames through the mk_script library file (@template). In this file, machine-dependent modifications to the scripts that are to be written can also be specified, such as the job submission command, the MPI command, the stopcommand (to delete all subsequent jobs from the queue in case the current job fails) and which queue to use (@queue). A standard location of the mk_script library file can be specified through the environment variable MK_SCRIPT_TEMPLATE.

Program mk_script can write input files for MD++ (@version). The MD++ input file (@files->input) should also be of the correct format: mk_script cannot convert program-specific MD++ input blocks into the analogous blocks for the other version of GROMOS.

In addition to write out scripts and input files, mk_script performs a small number of tests on the given input files to prevent the user from submitting a simulation that will fail within the first few seconds. In the messages produced by these tests, a distinction is made between warnings and errors. A warning is given for inconsistencies in the inputs that may lead to an erroneous simulation, but could also be intentional. An error is produced by inconsistencies that will definitely result in the program crashing. Note that passing the tests carried out by mk_script does not guarantee that a simulation will work, as these checks are not exhaustive. All performed tests are listed below (Warnings, Errors).

The mentioned tests are done for every script since some variables may change due to a joblist file. If there are no errors, the input file and script will be written to disc. If there are errors, the script and input file will not be written, unless the user forces this (@force).

Warnings:

1. the GROMOS binary specified in the mk_script input file cannot be found
2. the highest LAST atom number in the MULTIBATH block in the MD++ input file is not equal to the total number of atoms calculated from the topology file and SYSTEM block
3. DT in the STEP block is too large in combinations with the geometric constraints in the CONSTRAINT block (MD++). Suggested step sizes are:
   - 0.0005 ps: no constraints on solvent or bonds involving hydrogens
   - 0.001 ps: no constraints on bonds not involving hydrogens
   - 0.002 ps: all bonds constrained
4. the FORCE for bonds that are SHAKEn is calculated
5. the FORCE for bonds that are not SHAKEn is not calculated
6. smallest box dimension (length) of the periodic box is less than twice the long-range cut-off RCUTL in the PAIRLIST block of the MD++ input file
7. the reaction field cut-off distance RCRF in the NONBONDED block of the MD++ input file is not equal to the long-range cut-off RCUTL in the PAIRLIST block (MD++)
8. a perturbation topology was specified in the mk_script input file but no perturbation was requested in the MD++ input file
9. the combination of RLAM and DLAMT in the PERTURBATION block and the number of steps from the STEP block in the MD++ input file will lead to a lambda value larger than 1

Errors:

1. one of the essential blocks is missing (MD++): STEP, BOUNDCOND, INITIALISE, FORCE, CONSTRAINT, PAIRLIST, NONBONDED

2. there is no `VELOCITY` block in the coordinate file, but `NTIVEL` in the `INITIALISE` block of the MD++ input file specifies that the velocities should be read from file
3. non-zero `NTISHI` in the `INITIALISE` block of the MD++ input file specifies that the lattice shifts should be initialised, but zero `NTB` in the `BOUNDCOND` block specifies a vacuum simulation
4. there is no `LATTICESHIFT` block in the coordinate file, but `NTISHI` in the `INITIALISE` block of the MD++ input file specifies that the lattce shifts should be read from file
5. there is no `GENBOX` block in the coordinate file, but non-zero `NTB` in the `BOUNDCOND` block specifies a non-vacuum simulation
6. the number of the last atom given in the `FORCE` block of the MD++ input file is not equal to the total number of atoms calculated from the topology and `SYSTEM` block
7. the number of atoms calculated from the topology and `SYSTEM` block of the MD++ input file is not equal to the number of atoms in the `POSITION` block of the coordinate file
8. in the `PAIRLIST` block, the short-range cutoff `RCUTP` is larger than the long-range cutoff `RCUTL` (MD++)
9. no position restraints specification file is specified in the `mk_script` input file, but position restraining is switched on in the MD++ input file
10. no perturbation topology file is specified in the `mk_script` input file, but perturbation is switched on in the MD++ input file

Required Input Arguments

`@sys`	⟨system name⟩	
`@bin`	⟨GROMOS binary to use⟩	
`@dir`	⟨where the files should be (all filenames to be given relative to this)⟩	
`@version`	⟨md++ / promd⟩	
`@files`		
	`topo`	⟨molecular topology file (see Sec. 1.2)⟩
	`input`	⟨input file⟩
	`coord`	⟨initial coordinates⟩
	`refpos`	⟨reference positions⟩
	`posresspec`	⟨position restraint specifications⟩
	`disres`	⟨distance restraint specifications⟩
	`dihres`	⟨dihedral restraint specifications⟩
	`jvalue`	⟨j-value restraint specifications⟩
	`order`	⟨order parameter restraint specifications⟩
	`ledih`	⟨local elevation dihedral specifications⟩
	`pttopo`	⟨perturbation topology⟩

Optional input arguments

`@script`	⟨first script⟩ ⟨number of scripts⟩ (default: 1 1)
`@joblist`	⟨joblist file⟩
`@template`	⟨`mk_script` library file or filename template⟩
`@queue`	⟨which queue to use in `mk_script` library file⟩
`@remd`	⟨master / slave hostname port⟩ (replica exchange MD)
`@cmd`	⟨overwrite last command in `mk_script` library file⟩
`@force`	(write script regardless of errors)

Standard output

warnings and errors concerning the consistency checks

Additional output

scripts and input files for the requested MD simulations

2.19. `pdb2g96` (GROMOS++ program)

Program description:

Converts coordinates of a pdb file (Protein Data Bank) to coordinates in GROMOS format. The unit of the coordinates is converted from Å to nm. The order of the atoms in the pdb file does not necessarily correspond to the order of the atoms in the topology, but the residues should come in the proper order. The program identifies atoms and residues based on their names. Alternatives to the atom and residue names in the topology can be specified in a library file (see Sec. 4-7.3). The only requirement on residue numbers in the pdb file is that the residue number should change when going from one residue to the next. Mismatches between the topology and the pdb file are treated as follows:

1. If the expected residue according to the topology is not found, a warning is written out and the next residue in the pdb file is read in until a match with the topology is found.
2. Atoms that are expected according to the topology, but that are not found in the pdb file are written out in the coordinate file with coordinates (0.0, 0.0, 0.0). A warning is written to the standard error.
3. Atoms that are present in the pdb file, but not expected according to the topology are ignored, a warning is written to standard error.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pdb` ⟨pdb coordinates⟩
`@lib` ⟨library for atom and residue names⟩

Optional input arguments

`@out` ⟨resulting GROMOS coordinates⟩ (optional, defaults to stdout)
`@outbf` ⟨write B factors and occupancies to an additional file⟩

Standard output

GROMOS coordinates for the atoms of the system

Additional output

none

2.20. `pert_top` (GROMOS++ program)

Program description:

Creates a perturbation topology to perturb specified atoms. A perturbation topology is written that defines a perturbation to alter the specified atoms into a specified atom types, charges and masses. Each of the arguments `@types`, `@masses` and `@charges` can be omitted. In this case the values from the topology are taken. If not sufficient values are given, the last given value is taken for all the remaining atoms.

Use program `pt_top` to convert the resulting perturbation topology to a different format or to a regular molecular topology.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to be modified⟩
`@types`	⟨IACs of the perturbed atoms⟩
`@charges`	⟨charges of the perturbed atoms⟩
`@masses`	⟨masses of the perturbed atoms⟩

Optional input arguments

none

Standard output

perturbation topology

Additional output

none

2.21. prep_eds (GROMOS++ program)

Program description:

The topology file for EDS (dual topology!) is generated from $\mathcal{N}^{(s)}$'normal' topologies, where $\mathcal{N}^{(s)}$ is the number of end states. An end state is in EDS a molecule, e.g. a ligand. In the EDS topology, all states or molecules, respectively, are combined and excluded from another. The resulting molecular topology file is written out to a file called `com_eds.top`.

In the EDS perturbation topology, a molecule is 'visible' in one state and in all other states it consists of dummy atoms. For this the `MPERTATOM` block is used. The resulting perturbation topology file is written out to a file called `pert_eds.ptp`.

The argument `@inG96` converts GROMOS96 topologies to the current formats. On the other hand `outG96` converts topologies in the current format to the GROMOS96 format.

Required input arguments	
`@topo`	⟨molecular topology files of end states⟩
`@numstat`	⟨number of end states $\mathcal{N}^{(s)}$⟩
`@param`	⟨index number of molecular topology file to take parameters from⟩
`@solv`	⟨index number of molecular topology file to take solvent from⟩

Optional input arguments	
`@update_tree`	⟨switch for max. spanning tree update (required if `@form`=3)⟩
`@tree`	⟨file with old max. spanning tree (required if `@update_tree` is specified)⟩

Standard output

none

Additional output

combined molecular topology file (written to a file with name `com_eds.top`) and perturbation topology file (written to a file with name `pert_eds.ptp`) for EDS simulation (dual topology)

2.22. `prep_xray` (GROMOS++ program)

Program description:

Program prep_xray converts a crystallographic information file (CIF) containing reflection data into a GROMOS X-ray restraints specification file. Using a mapping file (map) it writes out the element names of the solute and solvent atoms according to their integer atom codes. The atoms' B-factors and occupancies are read from a special file (@bfactor) if requested or defaulted to 0.01nm^2 and 100%. The reflection list can be filtered according to the given resolution range. If no resolution is given, it is determined automatically. A random set of amplitudes is created for the computation of the free R factor.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@cif`	⟨crystallographic information file⟩
`@map`	⟨file with IAC-to-element name mapping⟩
`@bfactor`	⟨file with the B-factors and occupancies⟩
`@spacegroup`	⟨space group in Hermann-Mauguin format⟩
`@cell`	⟨cell descriptor: a, b, c, α, β, γ⟩
`@resolution`	⟨resolution range: minimum, maximum⟩
`@rfree`	⟨percentage taken for the free R factor⟩

Optional input arguments	
`@filter`	⟨filter amplitudes smaller than multiple of RMSD⟩
`@symmetrise`	⟨apply symmetry operations to reflections⟩
`@factor`	⟨factor to convert the length unit to Angstrom⟩

Standard output

crystallographic restraints specification data

Additional output

none

2.23. `prep_xray_le` (GROMOS++ program)

Program description:

Program prep_xray_le creates a X-ray local elevation file. It takes the side chains of the residues contained in the specified atoms. The side chains are defined in a special file (`@library`). It should contain the following block:

```
LESIDECHAIN
# name dim atom names
ARG 4 N CA CB CG CA CB CG CD CB CG CD NE CG CD NE CZ
ASN 2 N CA CB CG CA CB CG OD1
END
```

As the atom names define (dim) dihedral angles they have to be a multiple of four. The local elevation parameters (force constant, the number of bins of the grid, the functional form switch, the width of the potential energy function and its cutoff) are specified using `@leparam`. The X-ray parameters R^0 threshold and cutoff for R_{real} calculation are specified using `@xrayparam`.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@atoms`	⟨atoms to consider⟩
`@library`	⟨library file⟩

Standard output

crystallographic local elevation specification data

Additional output

none

2.24. pt_top (GROMOS++ program)

Program description:

Combines topologies with perturbation topologies to produce new topologies or perturbation topologies. Reads a topology and a perturbation topology to produce a new (perturbation) topology. The perturbation topology can contain a PERTATOMPARAM, or MPERTATOM block (see Sec. 4-3.3). The atom numbers in the perturbation topology do not need to match the numbers in the topology exactly. If the topology and perturbation topology do not match in their atom numbering, a shift can be applied using the @firstatom option.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pttopo	⟨perturbation topology with PERTATOMPARAM or MPERTATOM block⟩
@type	⟨output format: TOPO, PERTTOPO or PERTTOPO03⟩
@npt	⟨sequence number of the perturbation in a MPERTATOM block to apply (0 = stateA)⟩
@firstatom	⟨atom specifier (see Sec. 1.3.1): first atom to which the perturbation will be applied⟩

Optional input arguments

none

Standard output

combined (perturbation) topology

Additional output

none

2.25. ran_box (GROMOS++ program)

Program description:

When simulating a molecular liquid, a starting configuration for the solvent molecules has to be generated. Program `ran_box` generates a starting configuration for the simulation of mixtures consisting of an unlimited number of components. The molecules are randomly placed in a cubic or a truncated octahedron box, in a random orientation. Note that for the generation of a starting configuration for the simulation of pure liquids and binary mixtures, the programs `build_box` and `bin_box` can alternatively be used (see Secs. 2.2 and 2.1, respectively).

Required input arguments	
`@topo`	⟨topologies of single molecule for each molecule type: topo1 topo2 ...⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@pos`	⟨coordinates of single molecule for each molecule type: pos1 pos2 ...⟩
`@nsm`	⟨number of molecules for each molecule type: nsm1 nsm2 ...⟩
`@dens`	⟨density of liquid (kg/m^3)⟩

Optional input arguments	
`@thresh`	⟨threshold distance in overlap check (nm) [default: 0.20 nm]⟩
`@layer`	⟨create molecules in layers (along z-axis)⟩
`@boxsize`	⟨boxsize⟩
`@fixfirst`	⟨do not rotate / shift first molecule⟩
`@seed`	⟨random number generator seed⟩

Standard output

coordinate file (cubic or truncated octahedron box, specified density)

Additional output

progress report (written to the standard error)

2.26. `ran_solvation` (GROMOS++ program)

Program description:

When simulating a molecule in solution or in a crystal containing solvent molecules, the atomic coordinates of the solvent molecules are to be generated if they are not available from experiment. Alternatively to `sim_box` (which solvates a solute in a pre-equilibrated box of a molecular liquid, see Sec. 2.28), `ran_solvation` can solvate a solute in a mixture consisting of an unlimited number of components. The program places the solute in the center of a box (rectangular or truncated octahedron) and generates a random distribution of solvent molecules around it, which are placed in a random orientation. The total number of solvent molecules is calculated based on the specified solvent density and the molar fractions. When calculating the solvent density, the excluded volume of the solute is taken into account as specified by the user. `ran_solvation` checks separately for solute-solvent and solvent-solvent overlap after every insertion. If any solute-solvent or solvent-solvent interatomic distance is smaller than the respective threshold distance specified by the user, the insertion trial is rejected. Note that for the optional input flags, either the box-size or the minimum solute-to-wall distance should be specified.

Required input arguments

`@topo_u`	⟨molecular topology file (see Sec. 1.2) for the solute⟩
`@pos_u`	⟨input coordinate file for the solute to be solvated⟩
`@sev`	⟨solvent-excluded volume of the solute (nm^3) ⟩
`@topo_v`	⟨list of (single molecule) molecular topology files of solvents: topo_solv1 topo_solv2 . . . ⟩
`@pos_v`	⟨list of (single molecule) coordinate files of solvents: pos_solv1 pos_solv2 . . . ⟩
`@molf_v`	⟨mole fraction of each solvent: molf_solv1 molf_solv2 . . . ⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@dens`	⟨mass density of solvent mixture (kg/m^3)⟩

Optional input arguments

`@minwall`	⟨minimum solute-to-wall distance(s)⟩
`@boxsize`	⟨length of box-edge(s)⟩
`@thresh_u`	⟨threshold interatomic distance in overlap check (solute - solvent); default: 0.40 nm⟩
`@thresh_v`	⟨threshold interatomic distance in overlap check (solvent - solvent); default: 0.20 nm⟩

Standard output

coordinate file for the solvated solute

Additional output

none

2.27. `red_top` (GROMOS++ program)

Program description:

For large molecular complexes, one would sometimes like to consider only a part of the many atoms, thereby reducing the computational effort required by a simulation. Programs `tstrip` and `filter` can filter an atomic coordinate file (see Secs. 4.64 and 4.29, respectively). Accordingly, program `red_top` can cut out parts of a molecular topology.

The user has to list atoms of the molecular topology to be reduced. All atoms, exclusions, bonds, bond angles etc. that involve atoms that are not in this list are removed. Note that to cut out all atoms within a sphere around a part of the system, one could first generate a list of the corresponding atoms using the program `pairlist` (see Sec. 5.5).

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2) to be reduced⟩

`@atoms` ⟨atoms in the system to keep⟩

Optional input arguments

none

Standard output

reduced topology file

Additional output

none

2.28. sim_box (GROMOS++ program)

Program description:

When simulating a molecule in solution or in a crystal containing solvent molecules, the atomic coordinates of the solvent molecules are to be generated, if they are not available from experiment. Program `sim_box` can solvate a solute in a pre-equilibrated box of solvent molecules. The file specifying the solvent configuration should contain a `GENBOX` block with the dimensions corresponding to the pre-equilibrated density. The solvent topology is read from the `SOLVENTATOM` block of the specified topology.

To prevent overlap between solute and solvent molecules, only solvent molecules for which the centre of geometry is at a minimum distance from any solute atom (which can be defined via the `@thresh` flag) are put into the box. Before solvating the solute molecules, the solute can be rotated such that the largest distance between any two solute atoms is directed along the z-axis, and the largest atom-atom distance in the xy-plane lies in the y-direction, by giving the `@rotate` flag. The resulting box containing solute and solvent molecules can be either rectangular or a truncated octahedron. Its dimensions can be specified via the `@boxsize` flag. If this flag is given, the box dimensions are read in from the `GENBOX` block in the solute coordinate file. Alternatively, when the `@minwall` flag is given, the solute is put into a box filled with solvent molecules with box dimensions guaranteeing a minimum distance between any solute molecule and the box edges. Either one value can be specified for the `@minwall` flag, resulting in a cubic or truncated octahedron box, or three values can be specified, to generate a rectangular box. In the latter case, the solute molecules can be gathered on request (by specifying the `@gather` flag) and the `@rotate` flag must be given. Note that to solvate a solute in a triclinic box, one can use `sim_box` to generate a rectangular box and subsequently apply the appropriate symmetry transformations on the generated box using the program `cry` or use `sim_box` to generate a truncated octahedral box and convert it to a triclinic box using the program `unify_box`.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2) of the solute⟩
`@pbc`	⟨periodic boundary condition (r or t)⟩
`@pos`	⟨input coordinate file for the solute⟩
`@solvent`	⟨input coordinate file for the pre-equilibrated solvent⟩

Optional input arguments	
`@boxsize`	⟨use boxsize specified in input file⟩
`@minwall`	⟨minimum solute to wall distance⟩
`@thresh`	⟨minimum solvent to solute distance (nm) [default: 0.23 nm]⟩
`@gather`	⟨gather solute⟩
`@rotate`	⟨rotate solute: biggest axis along z, second along y⟩

Standard output

coordinate file of the solvated solute.

Additional output

none

CHAPTER 3

Minimizers and simulators

When performing an energy minimization (EM) or a molecular dynamics (MD) or stochastic dynamics (SD) simulation, the minimum requirement is the availability of a starting configuration and a molecular topology file containing the atomic masses and physical force-field data with respect to the molecular system (Chap. 2-5). The non-physical atomic interaction function $\mathcal{V}^{(spec)}$ (Chap. 2-9) needs not be specified, since it only serves special purposes. The different terms in $\mathcal{V}^{(spec)}$ represent different ways in which the motion of the atoms can be restrained or influenced:

- atom position restraining or fixing
- atom-atom distance restraining
- dihedral-angle restraining
- ${}^3J$-coupling constant restraining
- S^2 order-parameter restraining
- X-ray structure factor amplitude, electron density or symmetry restraining
- distancefield distance restraining
- local-elevation interaction

Since application of these special forces is optional, the different types of special force data are kept in different files. When performing a SD simulation, atomic friction coefficients γ_i must be available. These can be given in an atomic friction coefficients file. Finally, when performing a free energy calculation a perturbation molecular topology is required specifying the change in Hamiltonian from state A to state B.

3.1. `md` (MD++ program)

Program description:

Program `md` carries out an EM, MD or SD simulation for a molecular system consisting of solute and solvent (molecules Chaps. 2-11, 2-12 and 2-13). Periodic boundary conditions can be applied Sec. 2-4.1. Bond lengths and in solvent molecules also bond angles can be treated as holonomic constraints (Chap. 2-10). Temperature and pressure can be maintained by weak coupling of different degrees of freedom (solute internal plus rotational, solute translational, solvent) to different temperature baths and of the box dimensions (isotropic or along x-, y- and z-axis) to different pressure baths. Translation of and rotation around the centre of mass of the molecular system can be monitored and halted. Atomic coordinates, velocities, energies, pressure, volume and free energy data can be written to various trajectory files for later analysis.

Required input arguments:

`@topo`	R	⟨molecular topology file (see Sec. 1.2)⟩
`@conf`	R	⟨coordinates and restart data⟩
`@input`	R	⟨input parameters⟩
`@fin`	W	⟨final configuration⟩
`@trc`	W	⟨coordinate trajectory⟩

Optional input arguments:

`@pttopo`	R	⟨perturbation topology file⟩
`@trc`	W	⟨coordinate trajectory⟩
`@trv`	W	⟨velocity trajectory⟩
`@trf`	W	⟨force trajectory⟩
`@trs`	W	⟨special trajectory⟩
`@tramd`	W	⟨RAMD trajectory⟩
`@tre`	W	⟨energy trajectory⟩
`@bae`	W	⟨block averaged energy trajectory⟩
`@trg`	W	⟨free energy trajectory⟩
`@bag`	W	⟨block averaged free energy trajectory⟩
`@posresspec`	R	⟨position restraints specification⟩
`@refpos`	R	⟨position restraints⟩
`@distrest`	R	⟨distance restraints specification⟩
`@dihtrest`	R	⟨dihedral restraints specification⟩
`@jval`	R	⟨J-value restraints specification⟩
`@order`	R	⟨S2-value restraints specification⟩
`@xray`	R	⟨Xray restraints specification⟩
`@lud`	R	⟨local elevation umbrella database⟩
`@led`	R	⟨local elevation coordinate specification⟩
`@friction`	R	⟨atomic friction coefficients⟩
`@print`		⟨print additional information⟩
`@anatraj`	R	⟨re-analyze trajectory⟩
`@verb`		⟨control verbosity⟩
`@version`		⟨print version information⟩

Standard output

general information about the running simulations, printed to the standard output

Additional output

different files (trajectories) depending on the input flags specified in the input file

3.2. `repex_mpi` (MD++ program)

Program description:

This program is used to run replica exchange simulations. See `md` (Sec. 3.1) for the documentation of all command line arguments. Additional command line arguments are reported below.

Required input arguments:

`@repout` W ⟨output file for replicas⟩

`@repdat` W ⟨replica data file⟩

Optional input arguments:

none

Standard output

information from master about timings, printed to the standard output

Additional output

general information about the running simulations is printed to the output file specified under `@repout`, different files (trajectories) depending on the input flags specified in the input file

3.3. `eds_2box` (MD++ program)

Program description:

This program is used to run twin-system EDS simulations. The command line arguments are reported below.

Required input arguments:

`@topo1`	R	⟨molecular topology file for box 1 (see Sec. 1.2)⟩
`@topo2`	R	⟨molecular topology file for box 2 (see Sec. 1.2)⟩
`@conf1`	R	⟨coordinates and restart data for box 1⟩
`@conf2`	R	⟨coordinates and restart data for box 2⟩
`@input1`	R	⟨input parameters for box 1⟩
`@input2`	R	⟨input parameters for box 2⟩
`@pttopo1`	R	⟨perturbation topology file for box 1⟩
`@pttopo2`	R	⟨perturbation topology file for box 2⟩
`@tre1`	W	⟨energy trajectory for box 1⟩
`@tre2`	W	⟨energy trajectory for box 2⟩
`@fin1`	W	⟨final configuration for box 1⟩
`@fin2`	W	⟨final configuration for box 2⟩

Optional input arguments:

`@trc1`	W	⟨coordinate trajectory for box 1⟩
`@trc2`	W	⟨coordinate trajectory for box 2⟩
`@trv1`	W	⟨velocity trajectory for box 1⟩
`@trv2`	W	⟨velocity trajectory for box 2⟩
`@trf1`	W	⟨force trajectory for box 1⟩
`@trf2`	W	⟨force trajectory for box 2⟩

Standard output

general information about the running simulations, printed to the standard output

Additional output

different files (trajectories) depending on the input flags specified in the input file

CHAPTER 4

Analysis of trajectories (postprocessing)

4.1. `bar` (GROMOS++ program)

Program description:

Program `bar` calculates free energy differences between (at least) two states using Bennett's Acceptance Ratio method.[1] The free energy between two states, i and j, is given by

$$\Delta G(\lambda_i \rightarrow \lambda_j) = k_B T \ln \frac{\langle f(E(\lambda_i) - E(\lambda_j) + C)\rangle_{\lambda_j}}{\langle f(E(\lambda_j) - E(\lambda_i) - C)\rangle_{\lambda_i}} + C \tag{4.1}$$

where $f(x)$ denotes the Fermi function

$$f(x) = \frac{1}{1 + exp(x/k_B T)} \tag{4.2}$$

and

$$C = k_B T ln \frac{N_j}{N_i} + \Delta G(\lambda_i \rightarrow \lambda_j). \tag{4.3}$$

These equations are solved self-consistently, using a numerically stable implementation.[2] The convergence criterion (relative change in free energy between iterations; `@convergence`) and maximum number of iterations (`@maxiterations`) can be specified.

Time series of the energies (with lenght N_i and N_j, respectively) are read in from one file per simulated state, which also contains the energies of the neighbouring states. These files may be generated using program `ext_ti_ana` with option `@bar_data` (see Sec. 4.27).

Error estimates are standardly determined from the ensemble averages in the equations above. Optionally, a bootstrap error can be computed, where the calculation is repeated the indicated number of times (option `@bootstrap`), with random samples of the original time series. The standard deviation of the bootstrap estimates is reported.

The program also computes the overlap integral from distributions of the energy differences $P_i(\Delta E)$ and $P_j(\Delta E)$, using

$$OI = 2 \sum_{\Delta E} \frac{P_i(\Delta E) P_j(\Delta E)}{P_i(\Delta E) + P_j(\Delta E)} \tag{4.4}$$

with the sum running over all bins of the distribution. The distributions may be written out to separate files using option `@printdist`.

Required input arguments	
`@files`	⟨energy files from `ext_ti_ana` (see Sec. 4.27)⟩
`@temp`	⟨absolute temperature⟩

Optional input arguments	
`@maxiterations`	⟨maximum number of iteractions (default: 500)⟩
`@convergence`	⟨relative change in free energy for convergence (default: 1E-5)⟩
`@printdist`	⟨write out distributions⟩
`@bootstrap`	⟨number of bootstrap estimates for error estimates (default: 0)⟩

Standard output

Overview of free energy calculation

Additional output

Distributions of the individual energy differences

4.2. `bilayer_dist` (GROMOS++ program)

Program description:

Distributions along the bilayer normal are useful to characterise membrane systems. This program calculates the distribution of a given set of atoms with respect to the center of mass of another given set of atoms (usually, this set will include all bilayer atoms to give a distribution of atoms with respect to the bilayer center of mass).

By default, all distributions are normalised to unity. However, the user may be interested in density distributions. In this case, the flag `@density` must be included. The user can also obtain mass or electron densities with the help of the `@mult` argument. This will multiply the distribution by a given number. If the user wants to calculate electron density profiles the `@mult` and `@density` arguments must be given. In some cases, the bilayer is not centered in the periodic box and the center of mass will not be in between the two layers but in the bulk of the solvent instead. The argument `@translate` can be used to circumvent this problem.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms for c.o.m calculation⟩
`@selection`	⟨atom specifier (see Sec. 1.3.1): atoms to consider⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@grid`	⟨integer (default: 100)⟩
`@translate`	⟨translate box⟩
`@mult`	⟨double (default: 1)⟩
`@density`	⟨calculate density distribution⟩

Standard output

distribution of selected atoms

Additional output

none

4.3. `bilayer_oparam` (GROMOS++ program)

Program description:

Deuterium order parameters (S_{CD}) can be derived from deuterium quadrupole splitting experiments and have used to study biological membranes. The corresponding carbon-hydrogen order parameters (S_{CH}) can be calculated by computing the correlation functions describing the reorientation of the carbon-hydrogen vectors. More precisely, for each methylene group along the chain, an order parameter tensor $\underline{\mathbf{S}}$ can be defined as

$$S_{ij} = \frac{1}{2}\langle 3cos\theta_i \; cos\theta_j - \delta_{ij}\rangle, \tag{4.5}$$

where θ_i is the angle between the i^{th} local molecular axis (x', y' or z') and the bilayer normal (z-axis), δ_{ij} is the Kronecker delta symbol and $\langle ... \rangle$ stands for trajectory averaging. As a convention, for the n^{th} methylene group C_n, the direction of the vector $C_{n-1} - C_{n+1}$ is taken as z', the direction of the vector normal to z' in the plane C_{n-1}, C_n, and C_{n+1} defines y', while x' is the direction of the vector perpendicular both to z' and y'. The quantity $S_{CH} = -(2/3S_{xx} + 1/3S_{yy})$ is the value to be compared with the experimental S_{CD} value.

The order parameters are calculated with respect to a fixed orientational vector (corresponding to the direction of the experimental magnetic field; usually taken along the bilayer normal) given by the flag `@refvec`.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1) for which order parameters will be calculated⟩
`@refvec`	⟨reference orientational vector⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩

Standard output

order parameter tensor components and carbon-hydrogen order parameter for selected atoms

Additional output

none

4.4. `cluster` (GROMOS++ program)

Program description:

Program `cluster` performs a conformational clustering based on a similarity matrix, such as calculated by the program `rmsdmat` (see Sec. 4.54). The program uses the clustering algorithm of Daura.[3] Structures with RMSD values smaller than a user specified cutoff are considered to be structural neighbours. The structure with the highest number of neighbours is considered to be the central member of the cluster of similar structures forming a conformation. After removing all structures belonging to this first cluster, the procedure is repeated to find the second, third etc. most populated clusters.

One specific structure can be forced to be the central member structure of the first cluster, this can also be the reference structure, by specifying structure number 0. The clustering can be performed on a subset of the matrix, by specifying the maximum number of structures to consider. This allows for an assessment of the development of the number of clusters over time.

Depending on the settings used for program `rmsdmat`, the flags `@human` and `@big` may need to be specified to ensure proper reading of the matrix.

Clusters may be further analysed using program `postcluster` (see Sec. 4.43).

Required input arguments	
`@rmsdmat`	⟨RMSD matrix file name⟩
`@cutoff`	⟨cutoff⟩
`@time`	⟨t0⟩ ⟨dt⟩

Optional input arguments	
`@maxstruct`	⟨maximum number of structures to consider⟩
`@human`	(use a human readable matrix)
`@force`	⟨structure⟩ (force clustering on the indicated structure, 0 is the reference)
`@big`	(when clustering more than 50'000 structures)

Standard output

listing of the size of every cluster that was found

Additional output

Two additional files are written to disk: `cluster_structures.dat` and `cluster_ts.dat`. `cluster_structures.dat` contains for every cluster its size, the central member and a listing of all structures in the cluster. `cluster_ts.dat` contains a time series of the clusters. At every point in time the structure number and cluster number are given.

4.5. `cog` (GROMOS++ program)

Program description:

Program `cog` calculates the centre of geometry (cog) or centre of mass (com) of the solute molecule(s) in the specified frames of the input trajectory file(s), and writes out a single trajectory file in which the position of the cog or com either replaces the atomic coordinates of the solute molecule(s) or are appended directly after the coordinates of the last atom of the solute molecule(s).

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨input trajectory files⟩

Optional input arguments

`@nthframe`	⟨write every nth frame (default: 1)⟩
`@cog_com`	⟨calculate center of geometry (cog) or centre of mass (com) of solute molecules (default: cog)⟩
`@add_repl`	⟨add (add) the position of the cog/com or replace (repl) the solute coordinates with the position of the cog/com (default: repl)⟩

Standard output

single trajectory file

Additional output

none

4.6. `cos_dipole` (GROMOS++ program)

Program description:

Program `cos_dipole` calculates the average dipole moments over a selected set of molecules, taking into account also polarizable sites. Standardly it outputs the magnitude of the average total, fixed and induced molecular dipoles, but if needed, the x-, y- and z-components can be written to a file by specifying the `@xyz` flag.

Note that the dipole moment is only well-defined for systems consisting of neutral molecules. If the system carries a net-charge, the dipole moment will depend on the position of the origin. In cases where the overall system is neutral but contains ions, the dipole moment will depend on which periodic copy of the ions is taken. In these cases, the program issues a warning that results will critically depend on the choice of gathering method.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨trajectory files⟩
`@trs`	⟨special trajectories with COS displacements⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@molecules`	⟨solute molecules to average over, e.g. 1-5,10,17⟩
`@fac`	⟨conversion factor for the unit of the dipole, default: 1; use 48.032045 to convert from e∗nm to Debye⟩
`@xyz`	⟨filename for writing out dipole x-,y-,z-components, if no name given: Mxyz.out⟩
`@solv`	⟨include solvent⟩

Standard output

time series of the magnitude of the average total, fixed and induced molecular dipoles

Additional output

output file `Mxyz.out` contains average x-, y- and z-components of the average total, fixed and induced molecular dipoles

4.7. `cos_epsilon` (GROMOS++ program)

Program description:

Program `cos_epsilon` calculates the dielectric permittivity, $\epsilon(0)$, of a simulation box from a Kirkwood-Fröhlich type of equation as derived by Neumann,[4] taking into account also the polarizable centers.

$$(\epsilon(0)-1)\frac{2\epsilon_{rf}+1}{2\epsilon_{rf}+\epsilon(0)} = \frac{\langle \boldsymbol{M}^2\rangle - \langle \boldsymbol{M}\rangle^2}{3\epsilon_0 \mathcal{V} k_B T} \quad , \tag{4.6}$$

where $\boldsymbol{M}$ is the total dipole moment of the system, ϵ_0 the dielectric permittivity of vacuum, ϵ_{rf} is a reaction-field epsilon value, $\mathcal{V}$ is the volume and $k_B T$ is the absolute temperature multiplied by the Boltzmann constant.

If the `@autocorr` flag is given, the program also writes out the normalized autocorrelation function $\theta(\tau)$,

$$\theta(\tau) = \frac{< \vec{M}(t)\cdot\vec{M}(t+\tau) >_t}{< M(t)^2 >_t} = exp(-\frac{t}{\tau_\phi}) \tag{4.7}$$

from which the Debye relaxation time τ_D can be calculated:[4]

$$\tau_D = \frac{2\epsilon_{rf}+\epsilon(0)}{2\epsilon_{rf}+1}\tau_\phi \tag{4.8}$$

Using the `@truncate` flag, the autocorrelation output can be truncated when the number of independent contributing frames drops below a given threshold.

Note that the total dipole moment of the system is only well-defined for systems consisting of neutral molecules. If the system carries a net-charge, the dipole moment will depend on the position of the origin. In cases where the overall system is neutral but contains ions, the dipole moment will depend on which periodic copy of the ions is taken. In these cases, cos_epsilon issues a warning that results will critically depend on the choice of gathering method.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@e_rf`	⟨reaction field epsilon⟩
`@temp`	⟨temperature⟩
`@traj`	⟨trajectory files⟩
`@trs`	⟨special trajectories with COS displacements⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@fac`	⟨conversion factor for the unit of the dipole, default: 1; use 48.032045 to convert from e∗nm to Debye⟩
`@autocorr`	⟨filename for storing time autocorrelation⟩
`@truncate`	⟨minimum number of independent contributing frames after which to truncate the correlation function⟩

Standard output

time series of the box dipole, average molecular dipole and epsilon

Additional output

output file `Mcorr.out` contains the box dipole autocorrelation data

4.8. cry_rms (GROMOS++ program)

Program description:

Program cry_rms is used to compute atom positional RMSDs and RMSFs between the asymmetric units within a unit cell of a crystalline system. The symmetry operations are either specified using a special file (@spec, @factor) or by the space group (@spacegroup). In order to identify the individual asymmetric units (ASUs) an @ref AtomSpecifier AtomSpecifier to the first atom of every ASU have to be given (@asuspec). If an RMSD is requested (@atomsrmsd), the atom positional RMSD between all the asymmetric units is printed in separate columns. The RMSF is calculated for the requested atoms (@atomsrmsf) while taking also the fluctuations of the symmetry related copies into account.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@traj	⟨input trajectory files⟩
@asuspec	⟨AtomSpecifier to the first atom of every asymetric unit⟩

Optional input arguments	
@spec	⟨specification file for the symmetry transformations⟩
@factor	⟨conversion factor for distances⟩
@spacegroup	⟨space group in Hermann-Mauguin format⟩
@atomsrmsd	⟨AtomSpecifier used for RMSD calculation⟩
@atomsrmsf	⟨AtomSpecifier used for RMSF calculation⟩

Standard output

Time-series of the atom-positional RMSD between of specified atoms with respect to all asymmetric units. Atom-positional RMSF of the asymmetric unit taking all asymmetric units of the unit cell into account.

Additional output

none

4.9. `dfgrid` (GROMOS++ program)

Program description:

Program `dfgrid` calculates a distancefield grid analogously to the way it is done in md++, where distancefield (DF) distances can be used as restraints and reaction coordinates in path pulling methods[5]. The DF method is based on the mapping of distances from a target point on a grid, where grid points that overlap with the protein are penalized. As a result the shortest path for a ligand to the target point will never go through the protein, making it less likely to get stuck in a dead end.

Program `dfgrid` calculates the grid for any given snapshot, facilitating the visualization and analysis. It writes out a coordinate file which contains both the input coordinates and the distancefield grid, in addition to the target (zero-distance) point which will often be a virtual atom. When choosing pdb as output format, the distances on the grid will be written to the b-factor column for easy visualization.

The use of either `@stride` or `@frames` to reduce the analysis to a few snapshots is recommended.

The following flags are defined in the same way as in the md++ parameter and distance restraints files: `@gridspacing`, `@proteinoffset`, `@proteincutoff` and `@smooth` (see Sec. 2-9.12). `@proteinatoms` has the same function as the corresponding md++ parameter, but here the atom selection is specified in the form of an atomspecifier.

`@max` allows to specify a maximum distance, grid points to which higher distances are mapped will not be written to the coordinate file. `@protect` protects grid points within a certain radius from the target point from being flagged as protein.

With the `@distatoms` flag you can specify (virtual) atoms for which the df distance will be printed in standard output and for which the shortest df path will be added to the output coordinates for visual inspection. If one is only interested in the distances, `@nogrid` will prevent writing of the coordinate file.

Required input arguments	
`@topo`	⟨molecular topology file ⟩
`@pbc`	⟨boundary type [gather method]⟩
`@atom`	⟨(virtual) atom specifier for the target (zero-distance) point⟩
`@gridspacing`	⟨grid spacing ⟩
`@proteinoffset`	⟨penalty for being in the protein ⟩
`@proteincutoff`	⟨cutoff to determine gridpoints within the protein ⟩
`@proteinatoms`	⟨last atom considered as protein⟩
`@traj`	⟨input trajectory files⟩

Optional input arguments	
`@max`	⟨maximum distance: do not write out grid points with higher distances (default: 1)] ⟩
`@smooth`	⟨number of rounds to smoothen the forces at the edge of the protein ⟩
`@protect`	⟨radius around the target atom that will not be flagged as protein ⟩
`@outformat`	⟨output coordinates format ⟩
`@notimeblock`	⟨do not write timestep block ⟩
`@time`	⟨time and dt⟩
`@stride`	⟨write every nth frame (default: 1)⟩
`@frames`	⟨select frames to write out, starts at 0 (default: 0) ⟩
`@distatoms`	⟨(virtual) atom specifier for atoms for which to output the df distance⟩
`@nogrid`	⟨do not write out grid coordinate file⟩

Standard output

none

Additional output

coordinates for the input system, DF grid and target point in the specified coordinate file format

4.10. `dfmult` (GROMOS++ program)

Program description:

Calculates free energy differences between multiple states A and B from an EDS simulation of a reference state R according to

$$\begin{aligned}\Delta\mathcal{F}_{BA} &= \mathcal{F}_B - \mathcal{F}_A = \mathcal{F}_B - \mathcal{F}_R - (\mathcal{F}_A - \mathcal{F}_R) = \Delta\mathcal{F}_{BR} - \mathcal{F}_{AR} \\ &= -\beta^{-1} ln \frac{\langle exp[-\beta(\mathcal{V}_B - \mathcal{V}_R)]\rangle_R}{\langle exp[-\beta(\mathcal{V}_A - \mathcal{V}_R)]\rangle_R}\end{aligned} \tag{4.9}$$

The program reads in energy time series generated by `ene_ana`. It provides an error estimate (err) which is based on calculation of the (co)variances and the statistical inefficiency as described in[6]. The implementation closely follows the Python implementation provided by the authors. When calculating averages and uncertainties special care is taken in order to avoid overflow (see[7]).

Required input arguments	
`@temp`	⟨temperature of the system⟩
`@stateR`	⟨energy time series of the reference state R⟩
`@endstates`	⟨energy time series of the end states⟩

Optional input arguments

none

Standard output

free energy differences between end states and reference state, and between end states

Additional output

none

4.11. `disicl` (GROMOS++ program)

Program description:

Program `disicl` classifies secondary structure elements in proteins and nucleic acids based on dihedral angles.[8,9] Angle, region and class definitions are read from a user-specified library file (see Sec. 4-7.10). The program will warn about overlapping regions and region limits that are outside the chosen periodic range and will abort if two classes have the same definition.

The program writes out classification statistics per residue and averaged over all residues (stat_disicl.out). Timeseries are written for each class (class_XXX.dat) and for the dihedral angles (ts_disicl.dat).

In the output files the determined class will be assigned to the residue 0 (as defined in the DSCLANG block) of the first region contributing to the classification.

The program provides an option (pdbstride) to write out pdb files containing class information in the b-factor column for visualization using the "color by b-factor"-function in your favorite visualization software. An additional pdb is created (colorlegend.pdb), which can be used as a kind of legend for the class color code when loaded into the visualization software together with the output pdbs.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@lib`	⟨library file⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@atoms`	⟨atoms to include (default: all atoms)⟩
`@skip`	⟨skip n first frames⟩
`@stride`	⟨take every n-th frame⟩
`@periodic`	⟨dihedral angle periodic range (default: -180 180)⟩
`@nots`	(do not write time series)
`@pdbstride`	⟨write out pdb coordinates every n steps (has to be >= stride)⟩

Standard output

timeseries of the dihedrals; timeseries and statistics of the classification

Additional output

pdb files with class information in the b-factor column

4.12. dg_ener (GROMOS++ program)

Program description:

Program dg_ener applies the perturbation formula to calculate the free energy difference between two states A and B. It reads in the output of program ener (section Sec. 4.22), which can be calculated for the same trajectory using two different Hamiltonians. The free energy difference is calculated as

$$\Delta G_{BA} = -k_B T \ln\langle e^{-(\hat{\mathcal{H}}_B - \hat{\mathcal{H}}_A)/k_B T}\rangle_A \tag{4.10}$$

where the average is over all entries of the energy files that are specified and the Hamiltonians are taken from the last column of these files.

Required input arguments

@temp	⟨temperature for perturbation ⟩
@stateA	⟨energy files for state A ⟩
@stateB	⟨energy files for state B ⟩

Optional input arguments

@col	⟨numbers of the columns to use from file A and B [default: last] ⟩

Standard output

For every line in the energy files, the program writes out the energy difference and the Boltzmann probability for that particular frame. The last column contains the current estimate of the free-energy difference.

Additional output

none

4.13. `dGslv_pbsolv` (GROMOS++ program)

Program description:

Progam `dGslv_pbsolv` will compute two electrostatic components of the solvation free energy, namely one for Coulombic interactions under non-periodic boundary conditions (CB/NPBC) and one for the user-specified electrostatics scheme (lattice sum, LS or reaction field, RF), under periodic boundary conditions (PBC). The solute will be centered in the computational box, with its center of geometry.

In the following, the abbreviation ΔG_{chg} will be used to denote an electrostatic component of the solvation free energy. ΔG_{chg} will be computed for a user-specified group of atoms. Note that all atoms of the solute topology block will be nonpolarizable, i.e. they will be assigned a relative dielectric permittivity of one.

$\Delta G_{chg}^{CB/NPBC}$ and $\Delta G_{chg}^{LS/PBC}$ will both be computed from two calculations employing finite difference (FD) algorithms.[10] One calculation is carried out with a permittivity appropriate for the solvent (ϵ_{sol}), the other calculation is carried out under vacuum conditions (ϵ_0). The differences between both is the solvation free energy. The resulting correction for the LS scheme is:

$$\Delta G_{corr}^{LS/PBC} = \Delta G_{chg}^{CB/NPBC(FD)} - \Delta G_{chg}^{LS/PBC(FD)} \tag{4.11}$$

with

$$\Delta G_{chg}^{env} = \Delta G_{chg;\epsilon_{sol}}^{env} - \Delta G_{chg;\epsilon_0}^{env} \tag{4.12}$$

where env can be $CB/NPBC(FD)$ or $LS/PBC(FD)$.

$\Delta G_{chg}^{RF/PBC}$ will be computed from a FFT algorithm.[11,12] For the RF scheme, the user should use the corresponding LS calculation to compute a correction to cancel possible grid discretization errors. That is, the resulting correction is:

$$\Delta G_{corr}^{RF/PBC} = \Delta G_{chg}^{CB/NPBC(FD)} - \Delta G_{chg}^{LS/PBC(FD)} + \Delta G_{chg}^{LS/PBC(FFT)} - \Delta G_{chg}^{RF/PBC(FFT)} \tag{4.13}$$

In the LS-scheme, tinfoil boundary conditions are used and a hat charge shaping function will be used. In the RF-scheme, a user-specified relative dielectric permittivity is used. Note that a relative dielectric permittivity of one implies no application of a reaction-field correction.

As an alternative, PQR files can be used instead of GROMOS coordinate files and topologies. PQR files are PDB files with the temperature and occupancy columns replaced by columns containing the per-atom charge and radius.

Required input arguments

if used with a GROMOS coordinate file and topology:

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@coord`	⟨g96 coordinates⟩
`@probeIAC`	⟨integer atom code to take for radius calculation (for water, it would be 4 or 5 depending on the ff)⟩
`@atoms`	⟨atoms to include⟩
`@atomsTOcharge`	⟨atoms to charge⟩
`@rminORsigma`	⟨which radii to use: rmin (0) or sigma (1); default 0⟩

if used with a PQR file:

`@pqr`	⟨pqr file⟩
`@coordinates`	⟨box coordinates in X,Y,Z direction (in nm) that were used in the simulation⟩
`@atoms`	⟨atoms to include; the molecule (as used in gromos-standard format) can be scipped (e.g. simply 'a' is enough to include all atoms)⟩
`@atomsTOcharge`	⟨atoms to charge; the molecule (as used in gromos-standard format) can be scipped (e.g. simply 1-5,7 is enough to include atoms 1 to 5 and 7)⟩

general input:

`@schemeELEC`	⟨electrostatics scheme: LS or RF⟩
`@rcut`	⟨cutoff distance in nm (ONLY USED IF scheme==RF)⟩
`@epsRF`	⟨reaction field relative dielectric permittivity (ONLY USED IF scheme==RF)⟩
`@epsSOLV`	⟨solvent relative dielectric permittivity of the employed solvent model⟩
`@gridspacing`	⟨grid spacing in nm⟩

Optional input arguments

`@epsNPBC`	⟨relative dielectric permittivity for NPBC calculation; default: 78.4⟩
`@maxiter`	⟨maximum number of iteration steps; default: 600⟩
`@cubesFFT`	⟨number of cubes in the fast Fourier transformation for boundary smoothing; default: 4⟩
`@probeRAD`	⟨probe radius in nm; default 0.14 (for water)⟩
`@radH`	⟨your desired hydrogen radius in nm; default 0.05⟩
`@radscal`	⟨scale non-H radii with this factor (in case you want to play with radii); default 1.0⟩
`@verbose`	⟨path to log file to document status and errors⟩

Standard output

Additional output

none

4.14. `diffus` (GROMOS++ program)

Program description:

Program `diffus` calculates the diffusion of the centre-of-geometry of a specified set of atoms. Firstly, the mean square displacements ($\Delta(t)$) are calculated over all considered molecules and over multiple time averages.

$$\Delta(t) = \frac{1}{N_m} \sum_{i=1}^{N_m} \langle [\mathbf{r}_i(t+\tau) - \vec{\mathbf{r}_i}(\tau)]^2 \rangle_{\tau \leq t_{av}-t} \tag{4.14}$$

where N_m is the total number of molecules (or atoms) considered in the analysis, and t_{av} is the duration of the averaging block.

According to the Einstein expression, the function $\Delta(t)$ should be approximately linear and in practice, the diffusion could be obtained from the slope of the $\Delta(t)$ devided by $2N_d t$:

$$D = \lim_{t \to \infty} \frac{\Delta(t)}{2N_d t} \tag{4.15}$$

where N_d is the number of considered dimensions (3 for 3D vectors $\mathbf{r}_i$). The slope of the $\Delta(t)$ is obtained from linear least-square fit (LSF). The diffus program makes an automatic LSF considering the whole time range of $\Delta(t)$, which might not be a reasonable approach due to the poor statistics for bigger values of t. It is recommended that the user analyzes the shape of $\Delta(t)$ and performs the LSF considering only the region of linearity.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@dim`	⟨dimensions to consider⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to follow⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@time`	⟨time and dt⟩

Standard output

The value for the diffusion D obtained from LSF and the corresponding R^2 of the LSF are printed to the standart output. The outputfile diffusdp.out contains the time series of the mean square displacement $\Delta(t)$.

Additional output

none

4.15. `dipole` (GROMOS++ program)

Program description:

Program `dipole` will calculate and print the dipole moment of molecules. By default, the program will take all solute atoms into account, but the user can also specify a set of atoms. The dipole moment of a set of atoms carrying a net-charge is ill-defined and depends on the position of the origin. For these cases, the program allows the user to move the centre of geometry of the atoms to the origin.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj` ⟨trajectory files⟩

Optional input arguments

`@time` ⟨time and dt⟩
`@atoms` ⟨atom specifier (see Sec. 1.3.1) atoms to include⟩ (default: all solute)
`@cog` (move molecule to centre of geometry)

Standard output

time series and averages of the magnitude of the dipole moment and its components

Additional output

none

4.16. `ditrans` (GROMOS++ program)

Program description:

Dihedral angle transitions can be monitored during the course of a simulation using MD++. Even though in many cases a molecular trajectory file will not contain every structure of the simulation, dihedral angle transitions can also be determined *a posteriori* from such a trajectory using program `ditrans`. This program can also write the time series of dihedral angles without taking the inherent periodicity of a dihedral angle into account, but rather allow for dihedral angle values below 0° or above 360°.

The program determines the position of maxima and minima in the dihedral angle potential energy function based on the phase shift and multiplicity given in the topology. Energy barriers arising from alternative terms, such as non-bonded interactions, which may in theory shift the position of energy minima and maxima of the dihedral angle are not taken into account.

Two different criteria can be used to count dihedral angle transitions, as described in the manual. A strict criterion only counts a transition once a dihedral angle passes beyond the minimum of an adjacent energy well to prevent counting of short lived transitions of the maximum dividing the two energy wells. Because of a possibly sparse sampling of data in a molecular trajectory, this criterion may be too restrictive. As an alternative a transition can also be counted as soon as a dihedral angle is seen to cross the maximum separating two energy wells.

The (standard) output can be restricted to the number of observed dihedral angle transitions for every dihedral angle that was specified or can be extended to information on every transition encountered.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@prop`	⟨property specifier (see Sec. 1.3.3)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@time`	⟨time and dt⟩
`@strict`	(use GROMOS96 transition criterion)
`@verbose`	(print out every encountered transition)
`@tser`	⟨file name⟩ (extended time series)

Standard output

number of dihedral angle transitions for every specified dihedral angle

Additional output

time series of dihedral angles without periodicity restrictions (optional)

4.17. `dssp` (GROMOS++ program)

Program description:

Program `dssp` monitors secondary structure elements for protein structures over a molecular trajectory. The definitions are according to the DSSP rules defined by Kabsch and Sander[13]. Within these rules it may occur that one residue is defined as being part of two different secondary-structure elements. In order to avoid duplicates in the output, the following priority rules are applied: Beta Sheet/Bridge > 4-helix > 5-helix > 3-helix > H-bonded turn > Bend. As a consequence, there may be, for instance, helices that are shorter than their minimal length.

The program summarizes the observed occurrences of the secondary structure elements and averages the different properties over the protein. In addition time series for every type of secondary structure element are written to file.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1) for the protein⟩
`@time`	⟨time and dt⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@nthframe`	⟨write every nth frame⟩	(default is 1)

Standard output

summary of the occurrences of all selected residues within different secondary structure elements

Additional output

time series of the occurrences of all structural elements are written to different files: `Bend.out`, `Beta-Bridge.out`, `Beta-Strand.out`, `3-Helix.out`, `4-Helix.out`, `5-Helix.out` and `Turn.out`

4.18. eds_update_1 (GROMOS++ program)

Program description:

Calculates iteratively the EDS parameters E^R and s from energy time series of a number (@numstat) of end states (@vy) and the reference state R (@vr). The form of the used Hamiltonian (single $s = 1$, multiple $s = 2$, maximum spanning tree $s = 3$) has to be specified (@form), as the number of s parameters depends on the functional form. For @form = 1, only a single s parameter is calculated (only recommended for $\mathcal{N}^{(s)} = 2$), for @form = 2 $\mathcal{N}^{(s)}$ ($\mathcal{N}^{(s)}$-1)/2 s parameters are calculated and for @form = 3 ($\mathcal{N}^{(s)}$-1) s parameters, respectively. There are always $\mathcal{N}^{(s)}$ energy offset parameters E^R, independent of the functional form. The same number of old parameters have to be given (s and E^R).
If a maximum spanning tree is used as functional form (@form = 3), an initial tree must be specified (@tree) and if this tree shall be updated along with the parameters (@update_tree = 1) or not (update_tree = 0).

The s parameters are calculated using

$$ln \sum_{j=1,j\neq i}^{M} \left[\left(\left\langle e^{-\beta(|\Delta\mathcal{V}_{ji}|-\Delta {E^R}_{ji})} \right\rangle_i \right)^s \right] = ln(M-1) - 1 \tag{4.16}$$

where M = $\mathcal{N}^{(s)}$ for @form = 1, and M = 2 for @form = 2,3, respectively.

The energy offset parameters are calculated using

$${E^R}_i(new) = -\beta^{-1} \cdot ln \left\langle \left(1 + \sum_{j=1,j\neq i}^{\mathcal{N}^{(s)}} e^{-\beta(\Delta\mathcal{V}_{ji}-\Delta {E^R}_{ji})} \right)^{-1} \right\rangle_{R_{new}} + {E^R}_i(old) \tag{4.17}$$

As the formulae are correlated, they are solved iteratively until both parameters are converged.

At the end of the program, the number of iterations are written out together with the final parameters. For @form = 3 and @update_tree = 1, the new maximum spanning tree is written to a separate file called tree.dat. When calculating averages and distributions special care is taken in order to avoid overflow (see[7]).

Required input arguments	
@temp	⟨temperature of the system⟩
@numstat	⟨number of end states $\mathcal{N}^{(s)}$⟩
@form	⟨functional form of the Hamiltonian⟩
@vr	⟨energy time series of the reference state R⟩
@vy	⟨energy time series of the end states⟩
@s	⟨ list of old s parameters⟩
@EiR	⟨ list of old energy offset parameters (${E^R}_i$)⟩

Optional input arguments	
@update_tree	⟨switch for max. spanning tree update (required if form=3)⟩
@tree	⟨file with old max. spanning tree (required if update_tree is specified)⟩

Standard output

new s and ${E^R}_i$ parameters

Additional output

if @form=3, the maximum spanning tree is written to a file with name tree.dat

4.19. eds_update_2 (GROMOS++ program)

Program description:

Calculates the EDS parameters E^R and s from energy time series of two endstates (`@vy`) and the reference state R (`@vr`). Two update schemes are implemented: Scheme 1 calculates the new parameters according to the procedure described in Ref.[14] In that case the parameter `@eunder` corresponds to the energy threshold. Scheme 2 calculates new parameters according to the procedure described in Ref.[15] In that case the parameter `@eunder` corresponds to the energy separating sampling from state A from sampling of state B while the parameters `@etrans` specifies the width of the transition region.

Required input arguments	
`@temp`	⟨temperature of the system⟩
`@vr`	⟨energy time series of the reference state R⟩
`@vy`	⟨energy time series of the end states⟩
`@s`	⟨current s parameter⟩
`@s_old`	⟨old s parameter⟩
`@EiR`	⟨old energy offset parameters ($E^R{}_i$)⟩
`@update`	⟨choice of update scheme⟩
`@eunder`	⟨energy threshold if update=1; separation energy if update=2⟩

Optional input arguments	
`@etrans`	⟨ignored if update=1; size of transition region if update=2 (required)⟩
`@scale`	⟨scaling factor to modify default factors⟩

Standard output

new s and $E^R{}_i$ parameters

Additional output

none

4.20. edyn (GROMOS++ program)

Program description:

Program edyn performs an essential dynamics analysis over a trajectory. The covariance matrix is calculated for the specified atoms and diagonalised. The eigenvalues and eigenvectors are written to file, as well as information about selected eigenvalues.

The trajectory is subsequently analysed as projections along the eigenvalues. For all of the selected eigenvalues, the atomic components of the eigenvalues and the time series of the projection along the eigenvalue are written to file. In addition, pdb files are written with coordinates of the specified atoms at the extreme values of the projection along the eigenvalue. With the @skip flag, the usually time-consuming projections can be skipped and, in this case, only the covariance matrix, the eigenvalues and the eigenvectors will be printed to file.

Required input arguments

@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@atoms	⟨atom specifier (see Sec. 1.3.1): atoms to be considered⟩
@ref	⟨reference coordinates⟩
@traj	⟨trajectory files⟩

Optional input arguments

@eigenvalues	⟨list of eigenvalues for which data is written⟩
@skip	(skip the (time-consuming) projections)

Standard output

none

Additional output

Most of the output of this program is written to a selection of files:

AVE.pdb	contains the average position of the specified atoms
EIVAL.out	contains the eigenvalues of the covariance matrix
EIVEC.out	contains the eigenvectors of the covariance matrix
EIFLUC.out	contains the fluctuation along the eigenvectors
ESSDYN.out	contains the averages, fluctuations, minimum and maximum values of the projections along the eigenvectors

In addition, several files are written out for each selected eigenvalue, x:

EVCOMP_x.out	contains the atomic contributions to the eigenvector
EVPRJ_x.out	contains the time series of the projection of the trajectory along the eigenvector
PRJMAX_x.pdb	contains coordinates of the selected atoms, displaced from the average positions along the eigenvector to the maximum value of the observed projection
PRJMIN_x.pdb	contains coordinates of the selected atoms, displaced from the average positions along the eigenvector to the minimum value of the observed projection

4.21. ene_ana (GROMOS++ program)

Program description:

GROMOS can write energies, free-energy derivatives and block averages of them to separate trajectory files for later analysis. Program `ene_ana` extracts individual values from such files and can perform simple mathematical operations on them. The format for (free) energy trajectory files as written by MD++ is known to the program. In addition, the user can define custom made formats of any trajectory file that comes in a block-format through a library file. `ene_ana` is able to read and interpret series of two types of such files simultaneously, typically referred to as the "energy file" and the "free energy file".

Using the same library file one can define properties to be calculated from the values that are listed in them. For the selected properties, `ene_ana` will calculate the time series, averages, root-mean-square fluctuations and a statistical error estimate. The error estimate is calculated from block averages of different sizes. The time for the time series is taken from the trajectory files, unless a different time interval between blocks is specified through an input parameter. If a topology is supplied, the `ene_ana` uses this to define the total solute mass (`MASS`) and the total number of solute molecules (`NUMMOL`).

Required input arguments	
`@en_files`	⟨energy files⟩
`@fr_files`	⟨free energy files⟩
`@prop`	⟨properties to monitor⟩

Optional input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩ (for `MASS` and `NUMMOL`)
`@time`	⟨t and dt⟩ (overwrites `TIME` in the trajectory files)
`@library`	⟨library for property names⟩ [print]

Standard output

averages, root-mean-square fluctuations and error estimates for the requested properties over the supplied trajectories

Additional output

time series of every property will be written to a separate file with name ⟨property⟩.dat.

4.22. `ener` (GROMOS++ program)

Program description:

Program `ener` can recalculate interaction energies over molecular trajectory files using the interaction parameters specified in the molecular topology file.

Non-bonded interactions are calculated for all selected atoms with all other atoms in the system. Some atoms can be specified as being soft, indicating that interactions involving any of these atoms have a specified softness parameter, for all other atoms in the system, the softness parameter $\alpha = 0$. Van der Waals interactions between particles i and j are calculated as

$$\mathcal{V}^{(vdw)}{}_{ij} = \left[\frac{C_{12}(i,j)}{(r_{ij}^6 + \alpha_{LJ}\lambda^2 C_{126})} - C_6(i,j)\right] \frac{1}{(r_{ij}^6 + \alpha_{LJ}\lambda^2 C_{126})} \tag{4.18}$$

with $C_{126} = C_{12}/C_6$ for C_{12} and C_6 unequal 0, $C_{126} = 0$ otherwise. C_{12} and C_6 are the interaction parameters taken from the topology, λ and α_{LJ} are specified by the user. Similarly, the electrostatic interaction, including reaction field contribution for a homogeneous medium outside the cutoff sphere is calculated as

$$\mathcal{V}^{(ele)}{}_{ij} = \frac{q_i q_j}{4\pi\epsilon_0}\left[\frac{1}{(r_{ij}^2 + \alpha_C\lambda^2)^{1/2}} - \frac{\frac{1}{2}C_{RF}\, r_{ij}^2}{({R_{RF}}^2 + \alpha_C\lambda^2)^{3/2}} - \frac{(1 - \frac{1}{2}C_{RF})}{R_{RF}}\right] \tag{4.19}$$

where ϵ_0 is the dielectric permittivity of vacuum and q_i and q_j are the atomic partial charges. R_{RF} is the reaction field cutoff distance, here assumed to be the same as the interaction cutoff. α_C and λ are again user specified. C_{RF} is calculated from the reaction field dielectric constant ϵ_{RF} and κ_{RF} (user specified) as

$$C_{RF} = \frac{(2 - 2\epsilon_{RF})(1 + \kappa_{RF}R_{RF}) - \epsilon_{RF}(\kappa_{RF}R_{RF})^2}{(1 + 2\epsilon_{RF})(1 + \kappa_{RF}R_{RF}) + \epsilon_{RF}(\kappa_{RF}R_{RF})^2} \tag{4.20}$$

The bonded interactions are calculated for all specified properties using the following interaction functions. For bonds we use the quartic bond stretching interaction form $V^{(b)} = V^{(b,q)}$:

$$V^{(b,q)}(b_n; k_n^{(b,q)}, b_n^0) = {}^1\!/_4 k_n^{(b,q)}({b_n}^2 - {b_n^0}^2)^2 \tag{4.21}$$

with b_n the actual bond length, $k_n^{(b,q)}$ and b_n^0 the force constant and optimal bond length, respectively. For angles we use the cosine-harmonic bond-angle bending interaction form $V^{(\theta)} = V^{(\theta,c)}$:

$$V^{(\theta,c)}(\theta_n; k_n^{(\theta,c)}, \theta_n^0) = {}^1\!/_2 k_n^{(\theta,c)}(\cos(\theta_n) - \cos(\theta_n^0))^2 \tag{4.22}$$

with θ_n the actual bond angle, $k_n^{(\theta,c)}$ and θ_n^0 the force constant and optimal bond angle respectively. For proper torsional dihedral angle terms we use:

$$V^{(\varphi)}(\varphi_n; k_n^{(\varphi)}, \varphi_n^0, m_n^{(\varphi)}) = k_n^{(\varphi)}(1 + \cos(\varphi_n^0)\cos(m_n^{(\varphi)}\varphi_n)) \quad \text{with} \quad \varphi_n^0 = 0, \pi\ . \tag{4.23}$$

with φ_n the actual dihedral angle value, $k_n^{(\varphi)}$ the force constant and φ_n^0 and $m_n^{(\varphi)}$ the phase shift and multiplicity, respectively. Improper dihedral energy contributions are calculated from the function $V^{(\xi)}$:

$$V^{(\xi)}(\xi_n; k_n^{(\xi)}, \xi_n^0) = {}^1\!/_2 k_n^{(\xi)}(\xi_n - \xi_n^0)^2 \tag{4.24}$$

ξ_n^0 are the force constant and optimal improper dihedral angle value.

The program can print out various energies in separate columns, e.g. bonded energies, non-bonded energies with the solute only, with the solvent only or the total energies.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1) : atoms for non-bonded interaction⟩
`@energies`	⟨energy specifier for the interaction energies to be calculated (1: covalent; 2: elec with solute; 3: elec with solvent; 4: elec total; 5: vdW with solute; 6: vdW with solvent; 7: vdW total; 8: nonbonded total; 9: total)⟩
`@props`	⟨property specifier (see Sec. 1.3.3): bonded properties to be calculated⟩
`@cut`	⟨cut-off distance⟩
`@eps`	⟨epsilon for reaction field contribution⟩
`@kap`	⟨kappa for reaction field contribution⟩
`@RFex`	⟨switch the self term for excluded atoms in the reaction field polarization on or off ⟩
`@soft`	⟨atom specifier (see Sec. 1.3.1) for soft atoms⟩
`@softpar`	⟨lam⟩ ⟨a_lj⟩ ⟨a_c⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@time` ⟨time⟩ ⟨dt⟩

Standard output

time series of calculated energies (averages).

Additional output

none

4.23. `epath` (GROMOS++ program)

Program description:

Program `epath` finds electron-tunneling pathways in proteins. It uses Dijkstra's graph search algorithm[16] to find the pathway with the highest product of the decay factors, corresponding to the "shortest path". The decay factor ϵ_{ij} for the electron transfer between atoms i and j is calculated according to:

$$\epsilon_{ij} = Ae^{B(r_{ij}^2 - R)} \tag{4.25}$$

where r_{ij} is the distance between the atoms and the different parameters A, B and R are specified for jumps through covalent bonds, hydrogen bonds and space as described by Beratan.[17]

For every atom of the system, the neighbouring atoms within a user-specified cutoff are determined. For every neighbouring atom its connectivity is classified as covalent, H-bonded or through space. The decay factor for the neighbouring atoms is calculated using the appropriate parameters and stored if it is higher than the decay factor that was already stored from a previous cycle. Additionally the jump type and the atom from where this jump occured are stored. When all atoms have been visited, the "shortest path" is backtraced from the acceptor to the donor.

The parameters A, B and R are configurable via a parameter file, as well as the parameters for the Hbond detection. All filenames are configurable.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@donor`	⟨atom specifier (see Sec. 1.3.1): electron donor⟩
`@acceptor`	⟨atom specifier (see Sec. 1.3.1): electron acceptor⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@cutoff`	⟨cut-off distance (default 0.6)⟩
`@param`	⟨parameter file⟩
`@outfile`	⟨name of the output file⟩
`@details`	(detailed output)
`@detailsfile`	⟨name of the output file of the details⟩
`@timeseries`	(print time series)
`@timeseriesfile`	⟨filename of the time series file⟩
`@verbose`	(produce a verbose output to stderr giving runtime details)

Standard output

A pdb file containing the coordinates of all atoms that have been part of a path throughout the different frames and how often they have been part of a path as percentage in the B-factor column

Additional output

Detailed output consiting off a summary per frame and a pdb file containing the coordinates of the system as well as the coordinates of the path corresponding to that frame linked together in order to make the path visualisable. Also the program can output to a file a timeseries of the product of the decay factor of the different frames as well as its log, and the avarages of both at the end.

4.24. `eps_field` (GROMOS++ program)

Program description:

Program `eps_field` estimates the relative static dielectric permittivity, $\epsilon(0)$, of a liquid when an external electric field was applied during the simulation. The permittivity for a specific external field is given by

$$\epsilon(0) = 1 + 4\pi \frac{< \mathbf{P} >_t}{\mathbf{E}^{ext}} \tag{4.26}$$

where $\mathbf{E}^{ext}$ is the external electric field, ϵ_0 is the dielectric permittivity of vacuum, and $\mathbf{P}$ is the polarisation of the system defined as

$$\mathbf{P}(t) = \mathcal{V}(t)^{-1} \boldsymbol{M}(t) \tag{4.27}$$

where $\boldsymbol{M}$ is the total dipole moment of the system and $\mathcal{V}$ is the volume.
Note, to get a linear response of the polarisation, the electric field should be small enough to avoid saturation, which is the case if

$$\frac{< \boldsymbol{\mu}_i \mathbf{E}^{ext} >}{3k_B} << T \tag{4.28}$$

with $\boldsymbol{\mu}_i$ the dipole moment of molecule i, k_B the Boltzmann constant and T the temperature, is fulfilled.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@E_ex` ⟨external electric field strength⟩
`@trs` ⟨special trajectory files (with box dipole moment)⟩

Optional input arguments

`@time` ⟨time and dt⟩

Standard output

time series of polarisation in z-direction. Average of polarisation in x-, y-, and z-direction, and $\epsilon(0)$

Additional output

none

4.25. `epsilon` (GROMOS++ program)

Program description:

Program `epsilon` calculates the dielectric properties of the system. For systems containing only neutral molecules, it estimates the relative dielectric permittivity, $\epsilon(0)$, of a simulation box from a Kirkwood-Fröhlich type of equation, as derived by Neumann,[4]

$$(\epsilon(0)-1)\frac{2\epsilon_{rf}+1}{2\epsilon_{rf}+\epsilon(0)} = \frac{\langle \boldsymbol{M}^2\rangle - \langle \boldsymbol{M}\rangle^2}{3\epsilon_0 \mathcal{V} k_B T} \quad , \tag{4.29}$$

where $\boldsymbol{M}$ is the total dipole moment of the system, ϵ_0 the dielectric permittivity of vacuum, ϵ_{rf} is a reaction-field epsilon value, $\mathcal{V}$ is the volume and $k_B T$ is the absolute temperature multiplied by the Boltzmann constant.

For systems containing ionic species, the total dipole moment is split into a rotational part, $\boldsymbol{M_d}$, and a translational part, $\boldsymbol{M_j}$:

$$\boldsymbol{M_d} = \sum_{m=1}^{N_m}\sum_{a=1}^{N_{m,a}} q_{m,a}(\mathbf{r}_{m,a} - \mathbf{r}_{cm,m}) \tag{4.30}$$

$$\boldsymbol{M_j} = \sum_{m=1}^{N_m} q_m \mathbf{r}_{cm,m} \tag{4.31}$$

where N_m is the total number of molecules, $N_{m,a}$ is the number of atoms of the molecule m and cm stands for the center of mass.

The generalized frequency-dependent dielectric constant can be decomposed into the following contributions:

$$\langle \boldsymbol{M_d}^2\rangle \tag{4.32}$$

and the autocorrelation functions

$$\langle \boldsymbol{M_d}(\tau)\boldsymbol{M_d}(\tau+t)\rangle \tag{4.33}$$

$$\langle \boldsymbol{J}(\tau)\boldsymbol{J}(\tau+t)\rangle \tag{4.34}$$

as well as the cross term

$$\langle \boldsymbol{M_d}(\tau)\boldsymbol{J}(\tau+t)\rangle \tag{4.35}$$

where $\boldsymbol{J} = \frac{d\boldsymbol{M_j}}{dt} = \sum_{m=1}^{N_m} q_m \mathbf{v}_{cm,m}$.

Using fit functions one can calculate both the frequency-dependent dielectric response and the static dielectric constant of the system. For more details see Ref.[18].

Practically, to calculate the static dielectric constant the contribution of the cross term can be neglected, while the contribution from Eq. 4.34 can be calculated from $\langle \Delta \boldsymbol{M_j}^2\rangle$ (the mean square displacement of $\boldsymbol{M_j}$) using the Einstein relation. For more details see Ref.[19].

Note that $\langle \Delta \boldsymbol{M_j}^2\rangle$ has to be calculated from an unfolded trajectory. For that reason, the first frame of the trajectory is gathered using the gbond (to avoid broken molecules) and the rest with the gtime method.

The program outputs the relative permittivity (epsilon) calculated exclusively from the $\langle \boldsymbol{M_d}^2\rangle$ contribution. For systems containing ionic species, $\langle \Delta \boldsymbol{M_j}^2\rangle$ is calculated and written in file `Mj2.out`, from which the translational contribution to the relative permittivity can be calculated. See the `fit_Mj2.py` script in the `gromos++/examples/` directory.

Optionally, the program can calculate the $\langle \boldsymbol{M_d}(\tau)\boldsymbol{M_d}(\tau+t)\rangle$ and $\langle \boldsymbol{J}(\tau)\boldsymbol{J}(\tau+t)\rangle$ autocorrelation functions as well as the $\langle \boldsymbol{M_d}(\tau)\boldsymbol{J}(\tau+t)\rangle$ cross term (files `MdMd.out`, `JJ.out` and `MdJ.out`). Note that velocity trajectory files have to be provided to calculate the contributions due to Eq. 4.34 and Eq. 4.35. The translational contribution to the relative permittivity can be calculated from file `JJ.out`. See the `fit_JJ.py` script in the `gromos++/examples/` directory.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@temp` ⟨temperature⟩
`@traj` ⟨trajectory files⟩

Optional input arguments

`@e_rf` ⟨reaction field epsilon⟩
`@time` ⟨time and dt⟩
`@traj_vel` ⟨velocity trajectory files⟩
`@omega` ⟨enable omega-dependent calculation⟩
`@MdJ` ⟨enable cross-term MdJ calculation (usually neglected)⟩

Standard output

time series of the current estimate of $\epsilon(0)$ calculated exclusively from the $\langle \boldsymbol{M_d}^2 \rangle$ contribution

Additional output

output file `Mj2.out` contains the time series of the mean square displacement of $\boldsymbol{M_j}$
output file `MdMd.out` contains the autocorrelation function of $\boldsymbol{M_d}$
output file `JJ.out` contains the autocorrelation function of $\boldsymbol{J}$
output file `MdJ.out` contains the cross correlation function of $\boldsymbol{M_d}$ and $\boldsymbol{J}$

4.26. `espmap` (GROMOS++ program)

Program description:

Program `espmap` calculates the vacuum electrostatic potential around a user specified group of atoms. It uses the atomic partial charges as defined in the topology and calculates the potential on a grid. The results are written to a `.pl` file that can be converted to a `.plt` file which can be read in by Gopenmol.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to consider⟩
`@grspace`	⟨grid spacing (default: 0.2 nm)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

none

Standard output

a `.pl` and `.plt` file to be read in by Gopenmol

Additional output

none

4.27. ext_ti_ana (GROMOS++ program)

Program description:

From a simulation at a single coupling parameter λ program `ext_ti_ana` predicts free energy derivatives $\frac{\partial \mathcal{H}}{\partial \lambda}$ over a range of λ values. To do this it requires that the terms described in ref[20] have been precalculated during the simulation and written to the energy trajectories (using gromos md++ block PRECALCLAM).

The program reconstructs the free energy derivatives at the requested λ_p values and performs a reweighing to obtain the ensemble averages at λ_p, from the simulations at the simulated λ_s, using

$$\left\langle \frac{\partial \mathcal{H}}{\partial \lambda} \right\rangle_p = \frac{\left\langle \frac{\partial \mathcal{H}}{\partial \lambda}\big|_p \, e^{-\beta[\mathcal{H}(\lambda_p)-\mathcal{H}(\lambda_s)]} \right\rangle_s}{\left\langle e^{-\beta[\mathcal{H}(\lambda_p)-\mathcal{H}(\lambda_s)]} \right\rangle_s} \tag{4.36}$$

The predictions from multiple simulations at λ_s can be merged into a single TI profile using program `ext_ti_merge` (see Sec. 4.28).

In GROMOS the coupling parameter of different interaction types x, Λ_x, can be set individually from a fourth order polynomial of the global coupling parameter λ (`md++` block LAMBDAS, see Sec. 2-14.4):

$$\Lambda_x = a_x\lambda^4 + b_x\lambda^3 + c_x\lambda^2 + d_x\lambda + e_x. \tag{4.37}$$

`ext_ti_ana` can make predictions for other combinations of the coefficients (a_x,b_x,c_x,d_x,and e_x) than the ones used in the simulation. The interaction properties (x) that can be given individual λ dependencies are:
- slj: lennard jones softness
- scrf: coulomb reaction-field softness
- lj: lennard jones
- crf: coulomb reaction-field
- bond: bond
- ang: angle
- impr: improper dihedral angle
- dih: dihedral angle
- kin: kinetic (not implemented in the LAMBDAS block of `md++`)

To facilitate the evaluation of many sets of coefficients for slj and scrf, these can be given in an input file of the following format:

```
TITLE
..
END
SLJ
# uniquelabel a b c d e
1 -0.4 0.4 -0.3 1.3 0.0
2 -0.4 0.4 -0.2 1.2 0.0
3 -0.4 0.4 -0.1 1.1 0.0
END
SCRF
# uniquelabel a b c d e
1 0.0 0.0 0.7 0.3 0.0
2 0.0 0.0 -0.1 0.3 0.0
END
```

Predictions will be made for any combination of every given slj with every given scrf.

The coefficients used in the simulation are specified by the `@lamX_sim` flags or read from the LAMBDAS block in a gromos input parameter file specified by `@imd`. The coefficients we want to predict for are specified by the `@lamX` flags. Also the temperature of the simulation, the simulated λ (`@slam`) and its exponent (`@NLAMs`) and the parameters of the PRECALCLAM block (`@nrlambdas`, `@minlam`, `@maxlam`) can be read from this parameter file. If parameters are specified explicitly as input flags they will always overwrite the corresponding values read from the imd file.

If the flag `@countframes` is set, the number of contributing frames for each predicted λ is appended as an additional column to the output. This number is evaluated as the number of snapshots for which the difference in the predicted energy and the simulated energy is less than the free energy difference between the two states, as calculated using the perturbation formula.

Error estimates can be calculated using bootstrapping. A random set of data points of the size of the original set will be chosen and the predictions made. This is repeated for as many bootstrap replicates as requested. The standard deviation over the bootstrap replicates is reported as a bootstrap error.

The predicted TI curves from several simulations at different λ values can be combined using program `ext_ti_merge` (see Sec. 4.28).

Finally, program `ext_ti_ana` can write out time series of energies at alternative value of λ to be used for free-energy estimates with Bennett's acceptance ratio (BAR). If option `@bar_data` is used without further input parameters, this data is written out for all predicted λ values, or if further input parameters are given it is only written for the selected values of λ. Program `bar` (see Sec. 4.1) can be used to estimate free-energy differences from these files.

Required input arguments

`@en_files`	⟨energy trajectory files⟩
`@fr_files`	⟨free-energy trajectory files⟩
`@library`	⟨library for block information⟩
`@temp`	⟨simulation temperature⟩

Optional input arguments

`@nrlambdas`	⟨number of precalculated lambdas (can also be read from `@imd`)⟩
`@minlam`	⟨minimum precalculated lambda (can also be read from `@imd`)⟩
`@maxlam`	⟨maximum precalculated lambda (can also be read from `@imd`)⟩
`@slam`	⟨lambda value of simulation (can also be read grom `@imd`)⟩
`@NLAMs`	⟨lambda exponent of simulation (can also be read grom `@imd`)⟩
`@lam<X>_sim`	⟨a⟩ ⟨b⟩ ⟨c⟩ ⟨d⟩ ⟨e⟩ coefficients for individual lambda dependence (default: 0 0 0 1 0) where ⟨X⟩ is one of: slj, scrf, lj, crf, bond, ang, impr, dih, kin
`@imd`	⟨gromos input parameter file⟩
`@NLAMp`	⟨lambda exponent value to predict for, default: `@NLAMs`⟩
`@lam<X>`	⟨a⟩ ⟨b⟩ ⟨c⟩ ⟨d⟩ ⟨e⟩ coefficients for individual lambda dependence (default: `@lam<X>_sim`) where ⟨X⟩ is one of: slj, scrf, lj, crf, bond, ang, impr, dih, kin
`@slj_scrf_file`	⟨file with sets of slj and scrf lambda coefficients⟩
`@no_<X>`	exclude ⟨X⟩; free energy derivative contribution, where ⟨X⟩ is one of: slj, scrf, lj, crf, bond, ang, impr, dih, kin
`@countframes`	⟨count nr of contributing frames for each plam⟩
`@pmin`	⟨min index of prediction⟩
`@pmax`	⟨max index of prediction⟩
`@bootstrap`	⟨number of bootstrap cycles⟩
`@outdir`	⟨directory to write output to⟩
`@lam_precision`	⟨lambda value precision in outfiles (default: 2)⟩
`@bar_data`	⟨print energies to be used for BAR (not reweighted)⟩
`@verbose`	⟨print used parameters to file header⟩
`@cpus`	⟨number of omp threads (default: 1)⟩

Standard output

details of the calculation

Additional output

- files with predicted free-energy derivatives using the specified parameters these can be merged with program `ext_ti_merge` (see Sec. 4.28
- files with data to be used by program `bar` (see Sec. 4.1)

4.28. `ext_ti_merge` (GROMOS++ program)

Program description:

Program `ext_ti_merge` combines TI curves predicted by program `ext_ti_ana` (Sec. 4.27) from several simulations at different λ_s points by a linear weighting scheme. Two weights are defined for any value of λ_P, which lies between two simulated points λ_{S1} and λ_{S2}:

$$w_{s1} = \frac{\lambda_p - \lambda_{s2}}{\lambda_{s1} - \lambda_{s2}} \qquad\qquad w_{s2} = \frac{\lambda_p - \lambda_{s1}}{\lambda_{s2} - \lambda_{s1}}. \tag{4.38}$$

The appropriate ensembler average of the free energy derivatives $\frac{\partial \mathcal{H}}{\partial \lambda}$ are then computed as,

$$\left\langle \frac{\partial \mathcal{H}(\lambda_p)}{\partial \lambda} \right\rangle_{\lambda_p} = w_{s1} \left\langle \frac{\partial \mathcal{H}(\lambda_p)}{\partial \lambda} \right\rangle_{\lambda_{s1}} + w_{s2} \left\langle \frac{\partial \mathcal{H}(\lambda_p)}{\partial \lambda} \right\rangle_{\lambda_{s2}} \tag{4.39}$$

The integral (using the trapezoidal rule) of the final TI curve (and of its error values) is appended to the output file.

Required input arguments

`@files` ⟨data files⟩ generated by `ext_ti_ana`, see Sec. 4.27

Optional input arguments

`@slam` ⟨lambda values of the simulation⟩ optional if found in the header of the data files after #SLAM

`@noerrors` ⟨do not read and use the error column which might be in the files⟩

Standard output

single merged free energy derivative profile

Additional output

none

4.29. `filter` (GROMOS++ program)

Program description:

Program `filter` reduces coordinate trajectory files and writes out a trajectory file (in GROMOS or pdb format) in which for every frame, the coordinates are only kept for atoms that are within a specific distance of a specified part of the system. To determine if interatomic distances are within the specified cut-off, either an atomic or a charge-group based cut-off scheme can be employed. Additionally, parts of the system can be specified for which in all cases the atomic coordinates should either be kept or rejected.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨input trajectory files⟩

Optional input arguments	
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to consider as reference part of the system⟩
`@cutoff`	⟨cut-off distance (nm, default: 0.0)⟩
`@pairlist`	⟨cut-off scheme (ATOMIC (default) or CHARGEGROUP)⟩
`@select`	⟨atom specifier (see Sec. 1.3.1): atoms to keep⟩
`@reject`	⟨atom specifier (see Sec. 1.3.1): atoms not to keep⟩
`@time`	⟨time and dt⟩ (overwrites `TIME` in the trajectory files)
`@outformat`	⟨output coordinates format, see Sec. 1.2⟩

Standard output

filtered trajectory file

Additional output

none

4.30. `follow` (GROMOS++ program)

Program description:

Program `follow` can create a 3D trace of selected atoms through time. The program always takes the nearest image with respect to the previous position of the particle.

Required input arguments	
`@topo`	⟨topology⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@dim`	⟨dimensions to consider⟩
`@atoms`	⟨atoms to follow⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt ⟩

Standard output

none

Additional output

For every atom that is selected, a pdb file is written out (`FOLLOW_x.pdb`) in which the trajectory is indicated in the `CONECT` entries.

4.31. `gathtraj` (GROMOS++ program)

Program description:

Program `gathtraj` applies the periodic boundary conditions to a coordinate trajectory and writes the gathered trajectory.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj` ⟨input trajectory files⟩

Optional input arguments

none

Standard output

a single trajectory file

Additional output

none

4.32. `hbond` (GROMOS++ program)

Program description:

Program `hbond` monitors the occurrence of hydrogen bonds over a molecular trajectory file. It can monitor conventional hydrogen bonds, as well as three-centered hydrogen bonds through geometric criteria.

A hydrogen bond is considered to be present if the distance between a hydrogen atom, H, connected to a donor atom D, is within a user specified distance (typically 0.25 nm) from an acceptor atom A and the D-H-A angle is larger than another user specified value (typically 135°). Occurrences of three centered hydrogen bonds are defined for a donor atom D, hydrogen atom H and two acceptor atoms A1 and A2 if:

(i) the distances H-A1 and H-A2 are within a user specified value (typically 0.27 nm)

(ii) the angles D-H-A1 and D-H-A2 are larger than a second user specified value (typically 90°)

(iii) the sum of the angles D-H-A1, D-H-A2 and A1-H-A2 is larger than a third user specified value (typically 340°)

(iv) the dihedral angle defined by the planes through the atoms D-A1-A2 and H-A1-A2 is smaller than a fourth user specified value (typically 15°).

The user can specify two groups of atoms (A and B) between which the hydrogen bonds are to be monitored. If hydrogen bond donors and acceptors are not explicitly specified, these can be filtered based on their masses, as can be specified in a so-called "massfile". If a reference structure is given, only hydrogen bonds that are observed in the reference structure will be monitored.

The program calculates average angles, distances and occurrences for all observed hydrogen bonds over the trajectories and prints out a time series of the observed hydrogen bonds.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@DonorAtomsA`	⟨atom specifier (see Sec. 1.3.1)⟩
`@AcceptorAtomsA`	⟨atom specifier (see Sec. 1.3.1)⟩
`@DonorAtomsB`	⟨atom specifier (see Sec. 1.3.1)⟩
`@AcceptorAtomsB`	⟨atom specifier (see Sec. 1.3.1)⟩
`@Hbparas`	⟨distance [nm] and angle [degrees]; default: 0.25, 135⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@threecenter`	⟨distances [nm]⟩ ⟨angles [degrees]⟩ ⟨sum⟩ ⟨dihedral⟩
`@ref`	⟨reference coordinates for native H-bonds⟩
`@massfile`	⟨massfile⟩
`@time`	⟨time and dt⟩

Standard output

Statistics on all monitored hydrogen bonds, consisting of average distances and angles, number of occurrences and percentage of occurrence over the trajectory.

Additional output

Time series of the number of hydrogen bonds and the number of three-centered hydrogen bonds are written to files `Hbnumts.out` and `Hb3cnumts.out`, respectively. Time series for every observed hydrogen bond and three-centered hydrogen bonds are written to files `Hbts.out` and `Hb3cts.out`, respectively.

4.33. `int_ener` (GROMOS++ program)

Program description:

Program `int_ener` recalculates the nonbonded interaction energy between two non-overlapping sets of solute atoms using the interaction parameters specified in the molecular topology file. It can also compute the interaction energy between a specified group of solute atoms and the solvent. If a time series is requested, the total nonbonded interaction is printed at each time point, along with the van der Waals and electrostatic contributions.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atomsA`	⟨atom specifier (see Sec. 1.3.1) for the first group of atoms⟩
`@traj`	⟨position trajectory file(s)⟩

Optional input arguments	
`@atomsB`	⟨atom specifier (see Sec. 1.3.1) for the second group of atoms⟩
`@solvent`	⟨compute energy between `atomsA` and solvent⟩
`@time`	⟨time and dt⟩
`@timeseries`	⟨print time series⟩
`@timespec`	⟨time points at which to compute the energy: ALL (default), EVERY or SPEC (if time series)⟩
`@timepts`	⟨time points at which to compute the energy (if timesseries and timespec EVERY or SPEC)⟩
`@cut`	⟨cut-off distance (default: 1.4)⟩
`@eps`	⟨epsilon for reaction field contribution (default: 1.0)⟩
`@kap`	⟨kappa for reaction field contribution (default: 0.0)⟩

Standard output

average total nonbonded interaction energy and the van der Waals and electrostatic contributions, preceded by time series if requested

Additional output

none

4.34. `iondens` (GROMOS++ program)

Program description:

Program `iondens` calculates the average density of ions (or other particles) over a trajectory file. A rotational fit of the system onto the solute can be performed, to correct for rotations of the complete simulation box. The density will be calculated on a grid of points. Two sets of densities can be written out, containing 1) occupancies on the grid points, relative to the maximally occupied gridpoint, or 2) occupancies as a percentage of the number of frames. User specified cutoffs determine which gridpoints will be written out.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@grspace`	⟨grid spacing (default: 0.2 nm)⟩
`@ions`	⟨atom specifier (see Sec. 1.3.1): ions to monitor⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to use for fit⟩
`@ref`	⟨reference coordinates⟩
`@thresholds`	⟨threshold values for occupancy percentages (default: 20 and 5)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

none

Standard output

none

Additional output

Four files will be written out, which are all in the same coordinate frame:

1) `ref.pdb` the reference structure used for fitting
2) `aver.pdb` the average structure over the trajectory
3) `grid.pdb` the ion density relative to the most occupied grid point
4) `gridnf.pdb` the ion density as percentage of the total number of frames

4.35. jepot (GROMOS++ program)

Program description:

Program jepot computes the ^3J-value local elevation (LE) potential energy term from a LE ^3J-value restrained simulation. The LE potential can be calculated for all values ($0 - 360°$) of all restrained angles at the end of the simulation only (@fin) or for selected angles (@angles) as a time series throughout the simulation (requires @topo, @pbc, @postraj and @restraj). The @timespec, @timepts and @restraj arguments control the time series. The time series can be of the LE potential for all values of the selected angle (ALL; default) or for only the current value of the selected angle (CURR) at each point in time, giving only the current contribution of the LE potential to the overall potential energy of the selected angle. With CURR, the @jval file must contain the ^3J-value specifications for the selected angle only.

@K is the force constant given in the MD input file. Note that this is multiplied by WJVR, the weight factor in the @jval file, during the calculation of the LE potential energy to give $k^{(Jr)}$.

Required input arguments	
@jval	⟨jvalue restraint specifications⟩
@K	⟨force constant⟩
@ngrid	⟨number of grid points⟩

Optional input arguments	
@angles	⟨angle⟩ values over which to compute the LE potential energy: ALL (default) or CURR
@fin	⟨file containing final coordinates (if not time series)⟩
@time	⟨time dt (optional and only if time series)⟩
@timespec	⟨time points at which to compute the LE potential energy: ALL (default), EVERY or SPEC (if time series)⟩
@timepts	⟨time points at which to compute the LE potential (if time series and @timespec is EVERY or SPEC)⟩
@topo	⟨molecular topology file (see Sec. 1.2) (if CURR)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2), if CURR⟩
@postraj	⟨position trajectory files (if CURR)⟩
@restraj	⟨restraint trajectory files (if time series)⟩

Standard output

With @angles ALL and @fin: ^3J-value LE potential energy over 360° for each of the ^3J-value restraints specified in @jval and @fin at the end of the simulation.
With @angles CURR, @topo, @pbc, @postraj and @restraj: the current value of the restrained angle and the ^3J-value LE potential energy for this angle value. A time series will be written unless only one frame is selected (using @timespec, @timepts). The time values printed can be manipulated using @time and the time points for which the LE potential energy is calculated and printed can be controlled using @time, @timespec and @timepts.

Additional output

none

4.36. `jval` (GROMOS++ program)

Program description:

Program `jval` computes the ^3J-values from a single conformation or from a trajectory. It can write out the values of all ^3J-couplings specified in the file specified by `@jval` or the total RMSD over all couplings from the reference values at each point in time. The final part of the output is always a summary of the ^3J-value specification parameters, the averages over the entire trajectory and other statistics. Note that the dihedral angle is computed in a non-periodic manner, possibly going beyond the range [0, 360]. This allows the user to distinguish very dynamic from more restricted dihedral angles through the average and root-mean-square fluctuations of the dihedral angle.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@jval`	⟨^3J-value specification file⟩
`@traj`	⟨position trajectory file(s)⟩

Optional input arguments	
`@timeseries`	⟨write time series of ^3J-values⟩
`@rmsd`	⟨write the RMSD over all ^3J-values as a time series⟩
`@time`	⟨time dt (optional and only if time series)⟩
`@timespec`	⟨time points at which to compute the ^3J-values: ALL (default), EVERY or SPEC (if time series)⟩
`@timepts`	⟨time points at which to compute the ^3J-values (if time series and `@timespec` is `EVERY` or `SPEC`)⟩

Standard output

Information required to specify each ^3J-value, averaged ^3J-values and other statistics. If `@timeseries`, the value of each ^3J-coupling is printed at each point in time. If `@rmsd`, the overall RMSD from the reference ^3J-values is printed at each point in time.

Additional output

none

4.37. m_widom (GROMOS++ program)

Program description:

Program m_widom can calculate the free energy of inserting a test particle into configurations of a molecular system. For every configuration in the given trajectory file, the program places the particle at a user specified number of random positions and evaluates the nonbonded interaction energy, $\mathcal{V}^{(nbd)}$. The free energy is calculated as

$$\Delta G_S = -k_B\mathcal{T}\ln\frac{<\mathcal{V}e^{-\mathcal{V}^{(nbd)}/k_B\mathcal{T}}>}{<\mathcal{V}>} \tag{4.40}$$

with k_B the Boltzmann constant and $\mathcal{T}$ and $\mathcal{V}$ the temperature and volume of the system. The program will also calculate the solute-solvent energies according to

$$\Delta\mathcal{U}_{uv} = \frac{<\mathcal{V}^{(nbd)}\mathcal{V}e^{-\mathcal{V}^{(nbd)}/k_B\mathcal{T}}>}{\mathcal{V}e^{-\mathcal{V}^{(nbd)}/k_B\mathcal{T}}} \tag{4.41}$$

which equals the solute-solvent enthalpy, H_{uv}, as no volume change upon solvation is taking place. The solute-solvent entropy is subsequently calculated from

$$\mathcal{T}\Delta\mathcal{S}_{uv} = \Delta G - \Delta H_{uv} \quad . \tag{4.42}$$

For a more complete description of these free energies, see e.g.[21] In addition to the energetics of the system, the program can also calculate radial distribution functions for all inserted species, with respect to user-specified atoms in the original system. Each group of atoms to include in the rdf calculations is preceded by the keyword new in the input string. The radial distribution function is calculated as in the program rdf (Sec. 4.48), where all averages are weighted with the Boltzmann probability of every insertion attempt.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@intopo	⟨topology of the inserted particle⟩
@inpos	⟨coordinates of the inserted particle⟩
@cut	⟨cut-off distance⟩
@temp	⟨temperature⟩
@ntry	⟨number of insertion tries per frame⟩
@traj	⟨trajectory files⟩

Optional input arguments	
@time	⟨time⟩ ⟨dt⟩
@stride	⟨take every n-th frame (default: 1)⟩
@eps	⟨epsilon for reaction field (default: 1)⟩
@kap	⟨kappa for reaction field (default: 0)⟩
@rdf	⟨rdf with atom types⟩
@rdfparam	⟨rdf-cutoff⟩ ⟨grid⟩

Standard output

time series and summaries of free energies and solute-solvent enthalpies and entropies for all species that are inserted

Additional output

for every species, a file called rdf_widom_⟨n⟩.out is written out, where n represents the sequence number of the species in the perturbation topology (the files contain radial distribution functions of the inserted species with user-specified atoms)

4.38. `matrix_overlap` (GROMOS++ program)

Program description:

This program makes use of the GSL library to calculate the overlap between two matrices. Considering the following equation for the difference between matrices $\underline{\mathbf{M}}_1$ and $\underline{\mathbf{M}}_2$

$$\underline{\mathbf{D}} = \sqrt{\mathrm{tr}((\sqrt{\underline{\mathbf{M}}_1} - \sqrt{\underline{\mathbf{M}}_2})^2)} \tag{4.43}$$

where tr is the trace and the square root operator corresponds to the matrix-square root and is calculated according to the following steps. Consider a matrix $\underline{\mathbf{A}}$ that can be diagonalized by

$$\underline{\mathbf{T}} = \underline{\mathbf{V}}^{-1}\underline{\mathbf{A}}\underline{\mathbf{V}} \tag{4.44}$$

The square root of the elements of the diagonal matrix is taken. This procedure corresponds to the application of a normal scalar square root operator to all the elements of the diagonal matrix. Third, the square root of the diagonal matrix is used to calculate the square root of the matrix as

$$\underline{\mathbf{A}}^{1/2} = \underline{\mathbf{V}}\underline{\mathbf{T}}^{1/2}\underline{\mathbf{V}}^{-1} \tag{4.45}$$

The (normalized) overlap $\underline{\mathbf{O}}$ is given by 1 minus the difference $\underline{\mathbf{D}}$ of the matrices divided by the normalization factor as shown below,

$$\underline{\mathbf{O}} = 1 - \frac{\underline{\mathbf{D}}}{\sqrt{tr(\underline{\mathbf{M}}_1) + tr(\underline{\mathbf{M}}_2)}} \quad . \tag{4.46}$$

Required input arguments

`@m1`	⟨matrix 1⟩
`@m2`	⟨matrix 2⟩
`@dimension`	⟨dimension⟩

Optional input arguments

Standard output

trace, difference and normalised overlap of the two matrices

Additional output

none

4.39. `mdf` (GROMOS++ program)

Program description:

Program `mdf` calculates and lists, for a given set of atoms, the distance to the nearest atom belonging to a second set of atoms. For every selected atom, an output file is written with the minimum distance to, and an atom specifier (see Sec. 1.3.1) for, the nearest atom. This program also works for virtual atoms.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@centre`	⟨atom specifier (see Sec. 1.3.1): central group of atoms⟩
`@with`	⟨atom specifier (see Sec. 1.3.1): group of atoms from which to find the nearest atom⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩ (overwrites `TIME` in the trajectory files)

Standard output

none

Additional output

for every centre-atom, a file (`MIN_`⟨`atomspec`⟩`.dat`) is written out with a time series of the distance to the nearest with-atom and an atom-specification of that atom

4.40. `nhoparam` (GROMOS++ program)

Program description:

Program `nhoparam` calculates order parameters for a given set of nitrogen atoms. In a first step, the program determines the N-H bonds (of which μ is the unit vector) by the atomic masses of nitrogen and hydrogen. For secondary and tertiary amides the different N-H bonds are averaged. Then,

$$S^2 = \frac{1}{2}\left[3\sum_{i=1}^{3}\sum_{j=1}^{3}\langle\mu_i(t)\mu_j(t)\rangle_t^2 - 1\right] \tag{4.47}$$

is applied in order to calculate the order parameter of the N-H bond after performing a least-square rotational fit. Fitting can be controlled using the `@ref` and `@atomsfit` arguments. If `@ref` is absent, the first frame of the trajectory is taken as reference. `@atomsfit` are the atoms used for fitting. If omitted, the nitrogen atoms are used. The fit can be disabled by giving an empty set of atoms.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@winframe`	⟨averarging window (number of frames)⟩
`@atoms`	⟨nitrogen atoms for order parameter calculation⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@atomsfit`	⟨atoms to consider for fit⟩
`@ref`	⟨reference coordinate (if absent, the first frame of @traj is reference)⟩

Standard output

final results and statistics are written to the standard output

Additional output

running averaged and window averaged (using a window size of `@winframe`) order parameters are written to two seperate time series files (`OPts.out`, `OPwints.out`).

4.41. `noe` (GROMOS++ program)

Program description:

Program `noe` calculates and averages atom-atom restraint distances for specified NOE distances over a molecular trajectory. The NOE distances are to be specified in a NOE specification file that can be prepared with e.g. program `prep_noe` (see Sec. 4.45). Program `noe` will calculate the average distance according to $\langle \boldsymbol{r}^{-p} \rangle^{-1/p}$ for values of $p = 1, 3, 6$. It will also calculate the deviations of these distances from the specified reference distances, $\boldsymbol{r}_0$. These violations can be written to a time series file. The average violation is calculated as the sum of positive violations divided by the total number of NOE distances considered in the analysis.

The output of the program can be further analysed using program `post_noe`.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@noe` ⟨NOE specification file⟩
`@traj` ⟨trajectory files⟩

Optional input arguments

`@time` ⟨time and dt⟩

Standard output

average distances, violations and average violations of all NOE distances

Additional output

none

4.42. post_noe (GROMOS++ program)

Program description:

Program post_noe allows the user to re-analyse the data that was generated by program noe (see Sec. 4.41). It reads in the NOE specification file, and the filter-file, which were generated by program prep_noe (Sec. 4.45), as well as the output of program noe.

In cases where a stereospecific hydrogen from a CH2-group, without an explicit assignment was given in the library file of program prep_noe (type 4 without subtype), the NOE distance according to both virtual atoms will have been calculated. Program post_noe can be used to select from these distances the proton that shows either the largest or the smallest violation with the experimental data. Additionally, the user can choose to disregard specific NOEs either by specifying a 0 in the filter field of the filter file that was generated by prep_noe, or by giving a cutoff distance.

The user may want to regenerate the filter file using a different kind of pseudo-atom and multiplicity correction by running the prep_noe program a second time. When using post_noe, the reference distances can then be read from this filter file allowing for a quick assessment of the effect of the corrections. Furthermore, the user can tell post_noe to change the exponential in the averaging method.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@noe	⟨NOE specification file⟩
@noeoutput	⟨output of noe-program⟩
@filter	⟨NOE filter file⟩
@averaging	⟨1/3/6⟩

Optional input arguments	
@distance	⟨additional filter distance⟩
@ref	⟨noeoutput/filter⟩
@minmax	⟨min/max⟩
@distribution	⟨binsize⟩

Standard output

overview of NOE violations for the remaining NOE's after the filtering process, average NOE violations and if requested a distribution of the NOE violations

Additional output

none

4.43. `postcluster` (GROMOS++ program)

Program description:

Program `postcluster` can do additional analyses on the output of cluster (section Sec. 4.43). Three different kinds of analyses are currently possible on specified clusters:

1. `postcluster` can perform a lifetime analysis. A lifetime limit can be specified. This is the number of subsequent structures in the time series need to have switched to a different cluster before a true transition to the new conformation is taken into account. This allows the user to disregard single events from being counted as a double transition to and from a new conformation. The program will write out the number of times a certain cluster is observed, its average lifetime. In addition it prints for every cluster the number of transitions to and from the other clusters.
2. `postcluster` can also be used to analyse combined clusterings, in which the original structures come from different sources (e.g. different trajectories). This can be used to assess the overlap in the sampled conformational space between two simulations. This option is called `@rgb`. By specifying the number of frames from every individual source, the program will write out a file that can easily be used to produce a bar-plot in which the height of the bar indicates the size of the cluster and individual colors represent the portions of that cluster coming from the different sources.
3. `postcluster` can be used to write out trajectory files and single structure files containing the central member structures of the clusters. The trajectories can subsequently be used in any other analysis program to monitor properties over all structures belonging to one cluster.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@cluster_struct`	⟨structures file from cluster⟩
`@cluster_ts`	⟨time series file from cluster⟩
`@clusters`	⟨StructureSpecifier⟩

Optional input arguments	
`@lifetime`	⟨lifetime limit⟩
`@rgb`	⟨red⟩ ⟨green⟩ ⟨blue⟩ ...
`@traj`	⟨trajectory files⟩

Standard output

lifetime analyses of the specified clusters

Additional output

If requested a file `cluster_rgb.dat` will be written out, containing the split up of clusters to their origins. If the original trajectory files were specified, central member structures and complete clusters will be written to files `cluster_`⟨`X`⟩`.cms` and `cluster_`⟨`X`⟩`.trj`, respectively, where ⟨`X`⟩ represents the cluster number.

4.44. predict_noe (GROMOS++ program)

Program description:

Program predict_noe is used to predict possible NOE pairs from distance averages. The program calculates and averages all possible NOE pairs from a trajectory. The nuclei are taken from the atom specifier provided if those are found in the NOE library file as well.

The averaging is carried out as

$$\overline{r} = \left\langle r^{-p} \right\rangle^{-\frac{1}{p}} \tag{4.48}$$

where p can be either 1, 3 or 6 (@averaging). Distances above a threshold level (@filter) are discarded in the final output.

Required input arguments	
@topo	⟨molecular topology file⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@atoms	⟨atom specifier (see Sec. 1.3.1)⟩
@lib	⟨NOE specification library⟩
@traj	⟨trajectory files⟩

Optional input arguments	
@dish	⟨carbon-hydrogen distance (default: 0.1 nm)⟩
@disc	⟨carbon-carbon distance (default: 0.153 nm)⟩
@averaging	⟨averaging power: 1, 3 or 6 (default 6)⟩
@filter	⟨distance above which NOE's should be discareded [nm]⟩

Standard output

list of expected NOEs

Additional output

none

4.45. prep_noe (GROMOS++ program)

Program description:

Program prep_noe converts NOE data from an X-plor like format to GROMOS format, determining the proper choice of pseudo- or virtual atoms based on the topology and a library file. The output can be used to apply distance restraints during a simulation using program MD++, or to analyse a molecular trajectory using the program noe (Sec. 4.41). For a definition of the different types of pseudo- and virtual atoms see Sec. 2-9.4. In cases where the library file specifies a stereospecific CH2 atom (type 4), but does not indicate which of the two protons is specified, NOE upper bounds are created for both protons. Program post_noe can process the output of an NOE analysis to determine the best assignment.

The experimentally determined upper bounds are generally listed in a three column format, with distances in Å. prep_noe has three types of parsing these three columns, specified by @parsetype:

1. take the first value as the upper bound;
2. take the sum of the first and third values as the upper bound (default);
3. take the difference between the first and second values (commonly the lower bound).

The experimentally determined upper bounds can be corrected for pseudo-atom distances (addition of a geometric constant) or multiplicity factors (typically multiplication with $N^{1/p}$, where N is the multiplicity of indistinguishable protons involved and p is the averaging power). Such corrections can either be applied to the distances or can be taken out of a set of distances.

The NOE specification file (@noe) should contain the NOESPEC block (see section Sec. 4-7.8), which contains the ambiguous and unambiguous NOEs. For unambiguous NOEs, only the first eight columns of this file are to be specified. For ambiguous restraints, the 9th column repeats the number of the NOE (first column), the 10th column contains the number of NOEs this NOE may be linked to and the remaining columns lists the numbers of the NOEs to which it is linked.

Ambiguous NOEs can be assigned by program post_noe (see Sec. 4.42) by removing all but one of the ambiguous NOE distances through the @minmax flag. prep_noe writes a filter file which can be used to re-evaluate a given analysis over a specific trajectory, without recalculating all distances (also through program post_noe).

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@title	⟨NOE title for output⟩
@noe	⟨X-plor like NOE specification file⟩
@lib	⟨NOE specification library⟩

Optional input arguments	
@dish	⟨carbon-hydrogen distance; default: 0.1 nm⟩
@disc	⟨carbon-carbon distance; default: 0.153 nm⟩
@parsetype	⟨Upper bound parse type: 1, 2 or 3⟩ Choices are: 1: Upper bound = first number 2: Upper bound = first + third number (most common, default) 3: Upper bound = first - second number (commonly the lower bound)
@correction	⟨correction file⟩ [⟨correction type⟩]
@action	⟨add⟩ or ⟨sub⟩ correction from upper bound; default: add
@filter	⟨discard NOEs above a certain distance [nm]; default 10000 nm⟩
@factor	⟨conversion factor Angstrom to nm; default is 10⟩

Standard output
distance restraints specification file to be used with the analysis tool noe

Additional output

noe.filter: NOE filter file, containing upper bounds and information on automatically generated NOE distances in case of unassigned stereospecific protons

noe.dsr: distance restraint file to be used as `@disres` input for MD++.

4.46. r_factor (GROMOS++ program)

Program description:

Program r_factor calculates crystallographic structure-factor amplitudes and phases from a given trajectory and compares them to experimental values. Only the atoms given by the AtomSpecifier @atomssf are considered for the calculation. The atoms' IAC are mapped to their element names according to the rules given in the @map file. The atoms' B-factors and occupancies are read from a special file (@bfactor) if requested or defaulted to 0.01nm^2 and 100%. Structure factors are calculated to the given resolution (@resultion) while the cell information is calculated from the system's box. Symmetry operations are taken into account by specifying a space group (@spacegroup). Make sure you only give asymmetric unit when using spacegroup. The program can write the electron density (a $2 \mid F^0 \mid - \mid F \mid$ map) to special files (FRAME_DENSITY_#.ccp4), if requested (density flag).

A bulk solvent correction can be applied if @solvent is given. In a first step a solvent mask is determined. Therefore the parameters r_{vdW}, r_{ion}, r_{shrink} and the IAC of the water oxygen have to be provided. The occupied space is determined by the van-der-Waals radius of the atoms plus a probe radius. The van-der-Waals radius of an atom is calculated as half of the distance where the Lennard Jones potential energy of the atom/water-oxygen interaction reaches its minimum. The probe radius is either taken as r_{vdW} (for neutral atoms) or r_{ion} (for charged atoms). The occupied space is shrinked by r_{shrink}. The structure factor is calculated as

$$F = F_{\mathrm{model}} + \rho \exp(-B \sin(\theta)^2/\lambda^2)\mathcal{F}(\mathbf{M}). \tag{4.49}$$

The parameters ρ and B are determined by least-square fitting. Initial values have to be provided. For numerical stability the reflections are split in a high and low resolution set in the fitting procedure. Therefore a resolution cutoff has to be given. Finally the maximum iterations have to be given.

Required input arguments

@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@traj	⟨trajectory files⟩
@time	⟨time and dt⟩
@atomssf	⟨atoms considered in the structure factor calculation⟩
@cif	⟨crystallographic information file⟩
@map	⟨file with IAC-to-element name mapping⟩
@bfactor	⟨file with the B-factors and occupancies⟩
@resolution	⟨resolution range: minimum, maximum⟩

Optional input arguments

@spacegroup	⟨space group in Hermann-Mauguin format, default: P 1⟩
@density	⟨write electron density maps⟩
@factor	⟨factor to convert length unit to Angstrom⟩
@bins	⟨number of resolution bins for computation of the R-factor⟩
@solvent	⟨solvent parameters: RVDW RION RSHRINK IACW IRHO IB RESCUT MAXIT. RVDW: van-der-Waals radius of the probe, RION: van-der-Waals radius of an atom, RSHRINK: radius for shrinking of the solvent mask, IACW: integer atom code of the solvent van-der-Waals atom (e.g OW for SPC), IRHO: initial value for ρ, IB: initial value for B, RESCUT: resolution cutoff, MAXIT: maximum iterations⟩

Standard output

R-factor for all resolution bins

Additional output

calculated and ,,observed" electron density maps for every frame.

4.47. r_real_factor (GROMOS++ program)

Program description:

Program r_real_factor calculates two electron densities. One (ρ) from the atomic positions and a second (ρ^0) from the structure factor amplitudes and calculated phases. Only the atoms given by the AtomSpecifier `@atomssf` are considered for the structure factor calculation.

The real space residual

$$R = \frac{\sum \alpha\rho^0 + \beta - \rho}{\sum \alpha\rho^0 + \beta + \rho} \tag{4.50}$$

is calculated for every residue. Summation is only carried out over the extent of the atoms contained in the AtomSpecifier `@atomsr`

For the documentation of the other arguments see Sec. 4.46.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨trajectory files⟩
`@time`	⟨time and dt⟩
`@atomssf`	⟨atoms considered in the structure factor calculation⟩
`@atomsr`	⟨atoms considered in the R factor calculation⟩
`@cif`	⟨crystallographic information file⟩
`@map`	⟨file with IAC-to-element name mapping⟩
`@bfactor`	⟨file with the B-factors and occupancies⟩
`@resolution`	⟨resolution range: minimum, maximum⟩

Optional input arguments	
`@spacegroup`	⟨space group in Hermann-Mauguin format, default: P 1⟩
`@factor`	⟨factor to convert length unit to Angstrom⟩

Standard output

A time-series of the real-space R-factor.

Additional output

none

4.48. rdf (GROMOS++ program)

Program description:

Program rdf calculates radial distribution functions over structure files or trajectories. The radial distribution function, $g(r)$, is defined here as the probability of finding a particle of type J at distance $\boldsymbol{r}$ from a central particle I relative to the same probability for a homogeneous distribution of particles J around I. Program rdf calculates $g(r)$ for a number of discreet distances $r_{r(k)}$, separated by distance dr as

$$g(r) = \frac{\mathcal{N}_{a\,J(k)}}{4\pi {r_{r(k)}}^2 dr \rho_J} \tag{4.51}$$

where $\mathcal{N}_{a\,J(k)}$ is the number of particles of type J found at a distance between $r_{r(k)}$ - 1/2 dr and $r_{r(k)}$ + 1/2 dr and ρ_J is the number density of particles J. If particles I and J are of the same type, ρ_J is corrected for that. At long distances, $g(r)$ will generally tend to 1.

Both atoms of type I and J can be solute atoms, solvent atoms as well as virtual atoms. If more than one particle of type I is specified, rdf calculates the average radial distribution function for all specified atoms.

Required input arguments

@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@centre	⟨atom specifier (see Sec. 1.3.1): atoms to take as centre⟩
@with	⟨atom specifier (see Sec. 1.3.1): atoms to calculate distances for⟩
@cut	⟨maximum distance⟩
@grid	⟨number of points⟩
@traj	⟨trajectory files⟩

Optional input arguments

@nointra	⟨exclude intramolecular atoms ⟩

Standard output

radial distribution function of particles J around I

Additional output

none

4.49. rep_ana (GROMOS++ program)

Program description:

Program rep_ana extracts the information stored in the replica exchange molecular dynamics (REMD) output file replica.dat. It produces seven multi column output files:

temperature.dat	run number vs. temperature of replica (column 2 corresponds to replica 1, column 3 to replica 2 etc.)
lambda.dat	run number vs. lambda value of replica
epot.dat	run number vs. potential energy of replica
probability.dat	run number vs. switching probability of replica
switches.dat	run number vs. switching data of replica (0 = no switch in this run, 1 = switch in this run)
prob_T.dat	switching probabilities per temperature
prob_l.dat	switching probabilities per lambda

Furthermore it calculates an optimized temperature or lambda set based on the fraction of replicas diffusing from the lowest to the highest temperature (lambda value). This algorithm is based on[22].

Optional input arguments

@repdata ⟨REMD output file, replica.dat⟩

Optional input arguments

none

Standard output

none

Additional output

temperature.dat, lambda.dat, epot.dat, probability.dat, switches.dat, prob_T.dat and prob_l.dat

4.50. `rep_reweight` (GROMOS++ program)

Program description:

Program `rep_rewrite` sorts replica exchange trajectories according to the lambda values or the temperature and writes them to individual files.

Required input arguments

`@input` ⟨input file⟩
`@trj` ⟨cordinate trajectories⟩
`@name` ⟨prefix and postfix of output trajectories⟩

Optional input arguments

none

Standard output

none

Additional output

output file named ⟨`prefix`⟩_⟨`temperature`⟩_⟨`lambda`⟩.⟨`postfix`⟩

4.51. `reweight` (GROMOS++ program)

Program description:

Reweights a time series of observed values of X sampled during a simulation at state R (i.e. using the Hamiltonian $\hat{\mathcal{H}}_R = \hat{\mathcal{K}}_R(\vec{p}) + \hat{\mathcal{V}}_R(\vec{r})$) to another state Y (neglecting kinetic contributions for simplicity):

$$\langle X \rangle_Y = \frac{\langle X \exp\left[-\beta\left(\mathcal{V}_Y - \mathcal{V}_R\right)\right]\rangle_R}{\langle \exp\left[-\beta\left(\mathcal{V}_Y - \mathcal{V}_R\right)\right]\rangle_R} = \langle X \exp\left[-\beta\left(\mathcal{V}_Y - \mathcal{V}_R - \Delta\mathcal{F}_{YR}\right)\right]\rangle_R \tag{4.52}$$

with $\Delta\mathcal{F}_{YR} = \mathcal{F}_Y - \mathcal{F}_R$. The observed quantitiy X can be a structural quantity (e.g. the time series of an angle) or an energetic quantity (e.g. the time series of the ligand-protein interaction energy). Note that the reweighting will only give useful results if during the simulation at state R all configurations that are important to Y are sampled. The program reads three time series corresponding to the quantitiy X, the energy of state R, and the energy of state Y. All time series must have been calculated from the same ensemble R. The time series files consist of a time column and a column containing the quantity (i.e. X, $\mathcal{V}_R$, or $\mathcal{V}_Y$). The time series are obtained e.g. by `ene_ana` or `tser`. If the bounds flag is given a normalized distribution of X in the Y ensemble will be written out. When calculating averages and distributions special care is taken in order to avoid overflow (see[7]).

Required input arguments

`@temp`	⟨temperature of the system⟩
`@x`	⟨time series of quantity X⟩
`@vr`	⟨energy time series of the reference state R⟩
`@vy`	⟨energy time series of the end states⟩

Optional input arguments

`@bounds`	⟨lower bound⟩ ⟨upper bound⟩ ⟨grid points⟩

Standard output

average of quantity X in ensemble of state Y

Additional output

if bounds are given, a histogram (distribution) of the reweighted quantity X is given

4.52. rgyr (GROMOS++ program)

Program description:

Program `rgyr` calculates the radius of gyration, R_{gyr}, for a selected set of atoms over the trajectory according to

$$R_{gyr} = \sqrt{\frac{1}{N\sum_{i=1}^{N}}\sum(\mathbf{r}_i - \mathbf{r}_{com})^2} \tag{4.53}$$

where N is the number of specified atoms, $\mathbf{r}_i$ is the position of particle i and $\mathbf{r}_{com}$ is the centre-of-mass of the N atoms.

Alternatively, the radius of gyration can be calculated in a mass-weighted manner,

$$R_{gyr} = \sqrt{\frac{1}{M}\sum_{i=1}^{N} m_i(\mathbf{r}_i - \mathbf{r}_{com})^2} \tag{4.54}$$

where M is the total mass of the specified atoms and m_i is the mass of particle i.

Please note that in case atoms from more than one molecule have been chosen, care should be taken in the choice of gathering method to ensure a proper calculation of the centre-of-mass.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@time`	⟨time and dt⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1) for the atoms to consider⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@massweighted` (use massweighted formula)

Standard output

time series of the radius of gyration for the specified atoms

Additional output

none

4.53. `rmsd` (GROMOS++ program)

Program description:

The structural deformation of a molecule with respect to a reference structure can be expressed in terms of a root-mean-square deviation (RMSD) of the position of selected atoms. Program `rmsd` calculates the RMSD over a molecular trajectory after superimposing the centres of mass and performing a least-squares rotational fit. The fit can be performed using a different set of atoms than the calculation of the RMSD. The fit can be disabled by giving an empty set of atoms (compare Sec. 1.3.1).

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@time`	⟨time and dt⟩
`@atomsrmsd`	⟨atom specifier: atoms to consider for RMSD⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@ref`	⟨reference coordinates (if absent, the first frame of `@traj` is used as reference coordinates)⟩
`@atomsfit`	⟨atom specifier (see Sec. 1.3.1): atoms to consider for fit⟩

Standard output

time series of the root-mean-square deviation from the reference structure

Additional output

none

4.54. `rmsdmat` (GROMOS++ program)

Program description:

Program `rmsdmat` calculates the atom-positional root-mean-square deviation between all pairs of structures in a given trajectory file. This matrix of RMSDs can subsequently be used by program `cluster` to perform a conformational clustering. The matrix can be written out in human readable form, or – to save disk space – in binary format. For efficiency reasons, the RMSD values are written in an integer format. The user can specify the required precision of the RMSD values that are stored, if the precision is less or equal to 4, the values are stored as unsigned short int, otherwise as unsigned int.

Different sets of atoms can be selected to perform a rotational least-squares-fit and to calculate the RMS deviation from. The RMSD matrix can also be calculated from deviations in internal coordinates defined by a set of properties (e.g. torsional angles or hydrogen bonds). A selection of structures in the trajectory file to consider can be made using the options `@skip` and `@stride`. Structure pairs may occur for which the least-squares rotational fit fails for numerical reasons. In these cases both structures are fit to the reference structure. If no user specified reference structure is available, the first structure in the trajectory is taken as such. Specifying a reference structure allows the program `cluster` (section Sec. 4.4) to perform a forced clustering as well, requiring that the first cluster contains the reference structure, regardless of the cluster size.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@atomsfit`	⟨atom specifier (see Sec. 1.3.1): atoms to consider for fit⟩
`@atomsrmsd`	⟨atom specifier (see Sec. 1.3.1): atoms to consider for RMSD⟩
`@prop`	⟨property specifier (see Sec. 1.3.3): properties to consider for RMSD⟩
`@ref`	⟨reference coordinates⟩
`@skip`	⟨skip frames at beginning⟩
`@stride`	⟨use only every step frame⟩
`@human`	(write the matrix in human readable form)
`@precision`	⟨number of digits in the matrix (default 4)⟩
`@big`	(when clustering more than 50'000 structures)

Standard output

some information about the clustering process

Additional output

RMSD matrix in binary form (`RMSDMAT.bin`) or human readable form (`RMSDMAT.dat`) depending on the user specifications

4.55. `rmsf` (GROMOS++ program)

Program description:

Program `rmsf` calculates atom-positional root-mean-square fluctuations (RMSF) around average positions for selected atoms over a trajectory. A superposition of centres of mass and a rotational fit to a reference structure is performed for every structure in the trajectory. Different sets of atoms can be specified for the fitting procedure and for the calculation of the RMSF. The fit can be disabled by giving an empty set of atoms (see Sec. 1.3.1).

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atomsrmsf`	⟨atom specifier (see Sec. 1.3.1): atoms to consider for RMSF⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@ref`	⟨reference coordinates (if absent, the first frame of `@traj` is reference)⟩
`@atomsfit`	⟨atom specifier: atoms to consider for fit⟩

Standard output

a list containing the atom-positional root-mean-square fluctuation for each of the atoms specified by `atomsrmsf`

Additional output

none

4.56. `sasa` (GROMOS++ program)

Program description:

Program `sasa` calculates and prints the solvent-accessible surface area (SASA) of all heavy atoms in the solute part of the molecular system. It also calculates the contribution made by a specified set of heavy atoms. The program uses the algorithm of Lee and Richards[23]. A spherical probe of given radius is rolled over the surface of the molecule (the size of the probe is typically 0.14 nm for water). The path traced out by its centre gives the accessible surface. In GROMOS, the radii of the heavy atoms are obtained by calculating the minimum energy distance of the interaction between the heavy atom and the first solvent atom. This value is reduced by the specified probe radius to account for the radius of the solvent atom.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨boundary type⟩ [⟨gather method⟩]
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms to consider for sasa⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@zslice`	⟨distance between the Z-slices through the molecule (default: 0.005 nm)⟩
`@probe`	⟨probe IAC and radius (default: 4 0.14 nm)⟩
`@time`	⟨time and dt⟩
`@verbose`	(print summaries)

Standard output

time series of the solvent-accessible surface area for the selected heavy atoms and for all heavy atoms and, if requested, the atomic contributions for the selected atoms are also printed

Additional output

none

4.57. sasa_hasel (GROMOS++ program)

Program description:

Program sasa_hasel computes the solvent-accessible surface area (SASA) of all atoms in the solute part of the molecular system according to the method of Hasel et al.[24]. This is the method implemented in the SASA/VOL implicit solvent model. If a single conformation is given, either the atomic SASA values or the total SASA, along with the hydrophilic and hydrophobic contributions (defined by the sign of the sigma values given in the sasaspec file) may be printed. If multiple conformations are given, the averaged totals, the averaged atomic SASA values, or a time series of the total SASA values may be printed.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@sasaspec	⟨sasa specification library file⟩
@probe	⟨IAC of central atom of solvent and radius of solvent molecule⟩
@traj	⟨trajectory file(s)⟩

Optional input arguments	
@time	⟨time and dt (optional and only if time series)⟩
@timeseries	⟨if you want the time series as well as the average⟩
@timespec	⟨time points at which to compute the sasa: ALL (default), EVERY or SPEC (if time series)⟩
@timepts	⟨time points at which to compute the sasa (if time series and timespec EVERY or SPEC)⟩
@atomic	⟨print atomic sasa (only if not time series)⟩
@noH	⟨do not include hydrogen atoms in the sasa calculation (default: include)⟩
@p_12	⟨overlap parameter for bonded atoms (default: 0.8875)⟩
@p_13	⟨overlap parameter for atoms separated by two bonds (default: 0.8875)⟩
@p_1x	⟨overlap parameter for atoms separated by more than one bond (default: 0.3516)⟩

Standard output

single input conformation:

total sasa and contributions made by hydrophilic and hydrophobic atoms

single input conformation and @atomic:

atomic sasa values followed by total sasa and contributions made by hydrophilic and hydrophobic atoms

time series of input conformations:

average total sasa and contributions made by hydrophilic and hydrophobic atoms

time series of input conformations and @atomic:

average atomic sasa values followed by total sasa and contributions made by hydrophilic and hydrophobic atoms

time series of input conformations and @timeseries:

time series of total sasa and contributions made by hydrophilic and hydrophobic atoms, followed by averages

Additional output

none

4.58. `solute_entropy` (GROMOS++ program)

Program description:

Program `solute_entropy` takes a coordinate trajectory and calculates the configurational entropy using the Schlitter and the quasiharmonic analysis methods (see[25]) for a given set of atoms. The entropy can be averaged over a window of a given size. If requested, a superposition of centres of mass and a rotational fit prior to the entropy calculation is carried out.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atomsentropy`	⟨atoms to consider for entropy calculation⟩
`@temp`	⟨temperature⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@time`	⟨time and dt⟩
`@ref`	⟨reference structure for fit⟩
`@ref_pbc`	⟨reference boundary type⟩
`@atomsfit`	⟨atoms to consider for fit⟩
`@method`	⟨method to use: schlitter or quasiharm⟩
`@n`	⟨entropy is calculated every nth step⟩

Standard output

time series of the calculated configurational entropy

Additional output

none

4.59. `structure_factor` (GROMOS++ program)

Program description:

Program structure_factor calculates crystallographic structure-factor amplitudes and phases from a given trajectory. Only the atoms given by the AtomSpecifier `@atomssf` are considered for the calculation. The atoms' IAC are mapped to their element names according to the rules given in the `@map` file. The atoms' B-factors and occupancies are read from a special file (`@bfactor`) if requested or defaulted to 0.01nm^2 and 100%. Structure-factor amplitudes are calculated to the given resolution (`@resultion`) while the cell information is calculated from the system's box. Symmetry operations are taken into account by specifying a space group (`@spacegroup`). When using `@spacegroup`, make sure only the asymmetric unit is given.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨trajectory files⟩
`@time`	⟨time and dt⟩
`@atomssf`	⟨atoms considered in the structure factor calculation⟩
`@map`	⟨file with IAC-to-element name mapping⟩
`@bfactor`	⟨file with the B-factors and occupancies⟩
`@resolution`	⟨resolution range: minimum, maximum⟩

Optional input arguments	
`@spacegroup`	⟨space group in Hermann-Mauguin format, default: P 1⟩
`@factor`	⟨factor to convert length unit to Angstrom⟩

Standard output

averaged structure factor amplitudes

Additional output

none

4.60. `temperature` (GROMOS++ program)

Program description:

Program `temperature` will calculate the temperature for different sets of atoms, as specified by the atom-specifier(s). Multiple sets of atoms can be specified by white-space separated Atomspecifiers (see Sec. 1.3.1). For each of the sets one dof value is expected.

You can find the number of degree of freedoms for a temperature group in the md++ output file under "DEGREES OF FREEDOM" → "DOF"

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@atoms` ⟨atom specifier⟩
`@dofs` ⟨degrees of freedom⟩
`@traj` ⟨velocity trajectory files⟩

Optional input arguments

`@time` ⟨time and dt⟩

Standard output

timeseries of temperature for each set of atoms

Additional output

none

4.61. `tcf` (GROMOS++ program)

Program description:

Program `tcf` performs simple statistical analyses, calculates distributions and time-correlation functions for any series of data points. It takes files with data listed in any number of columns as input, but no further formatting, e.g. the output files of programs `tser` (see Sec. 4.63) or `ene_ana` (compare Sec. 4.21). Lines starting with the character # are ignored.

For data in the specified columns, the program writes out the number of data points, the average value, root-mean-square fluctuation, a statistical error estimate as well as the minimal and maximal value observed. The error estimate is calculated from block averages of different sizes. In addition, the program can calculate distributions and time-correlation functions. The program does not read time from the data file, but the time interval between data points can be specified by the user. Otherwise it is taken to be 1.

Distributions can be calculated for data in specified columns and can be normalized.

Time correlation functions of the general form

$$C(t) = \langle f(A(\tau), B(\tau + t)) \rangle_\tau \tag{4.55}$$

can be calculated, where $A(\tau)$ and $B(\tau + t))$ represent the data points at different time points and the user can specify any function $f(A, B)$. The program can calculate both auto-correlation functions ($B = A$) and cross correlation functions ($B! = A$) for time series of scalars or vectors. If A and B are represented by scalars and $f(A(\tau), B(\tau+t)) = A(\tau)*B(\tau+t)$, the program makes use of fast Fourier transforms to calculate $C(t)$. In other cases a direct summation algorithm is used, which may be considerably slower.

In cases where one is interested in the correlation function of the fluctuations around the average, this average value can be subtracted from the data points before calculating the correlation function. A power spectrum can also be calculated. Because the correlation function is usually very noisy at larger times, the noise level can be specified as the fraction of the correlation function that should be considered. This part of the correlation function is then smoothened by multiplication by a cosine to make sure that it is zero at the end. It is then mirrored: all data points are repeated in reverse order at the end. From this the Fourier transform is taken, which is the resulting spectrum.

Required input arguments

`@time`	⟨time⟩ ⟨time step⟩
`@files`	⟨data files⟩

Optional input arguments

`@distribution`	⟨data columns to consider⟩
`@bounds`	⟨lower bound⟩ ⟨upper bound⟩ ⟨grid points⟩
`@normalize`	(normalize the distributions)
`@tcf`	⟨data columns to consider⟩
`@expression`	⟨expression for correlation function⟩
`@spectrum`	⟨noise level⟩
`@subtract_average`	(take difference with respect to average value for `tcf`)

Standard output

statistical analyses for all specified data columns, distributions and / or time-correlation functions

Additional output

none

4.62. trs_ana (GROMOS++ program)

Program description:

Program `trs_ana` extracts individual values from gromos trajectory files and can perform simple mathematical operations on them.

The program is based on `ene_ana` (see Sec. 4.21). It uses the same library format to define blocks which can be read from any trajectory file that comes in the Gromos block-format. In contrast to `ene_ana` it does not require that all the blocks defined in the library are present in the trajectory file or in the specified order. It can handle trajectories where not all timesteps contain the same number of blocks, e.g. when different properties were written to the trajectory at different intervals. The time in the output timeseries will always correspond to the time in the previous TIMESTEP block if no time is given by the user, else the time will be increased by the given timestep at every occurrence of a TIMESTEP block. If multiple blocks of the same name occur between two TIMESTEPs, only the last one will be used.

In the library file one can also define properties to be calculated from the defined entries. For the selected properties, trs_ana will calculate the time series, averages, root-mean-square fluctuations and a statistical error estimate. The error estimate is calculated from block averages of different sizes, as described in Allen and Tildesley: "Computer Simulation of Liquids", 1987. If a topology is supplied, the trs_ana uses this to define the total solute mass (MASS) and the total number of solute molecules (NUMMOL).

Required input arguments	
`@trs`	⟨trajectory files⟩
`@prop`	⟨properties to monitor⟩
`@library`	⟨library for property names⟩

Optional input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩ (for `MASS` and `NUMMOL`)
`@time`	⟨t and dt⟩ (overwrites `TIME` in the trajectory files)

Standard output

averages, root-mean-square fluctuations and error estimates for the requested properties over the supplied trajectories

Additional output

time series of every property will be written to a separate file with name ⟨property⟩.dat.

4.63. `tser` (GROMOS++ program)

Program description:

Program `tser` can calculate structural quantities from a trajectory file and print the time series and/or a distribution of the value associated with the requested property. The quantity to be calculated is specified through a property specifier (compare Sec. 1.3.3) and can be any of the structural properties described in Sec. 1.3, which can be calculated from atomic positions in the trajectory file. Time series can later be analysed further with e.g. the program `tcf`.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@time`	⟨time and dt⟩
`@prop`	⟨property specifier (see Sec. 1.3.3)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments	
`@nots`	(do not write time series)
`@dist`	⟨steps [min max]⟩
`@norm`	(normalise distribution)
`@solv`	(read in solvent)
`@skip`	⟨skip n first frames⟩
`@stride`	⟨take every n-th frame⟩

Standard output

time series and/or distributions of the specified properties

Additional output

none

4.64. `tstrip` (GROMOS++ program)

Program description:

Program `tstrip` removes all solvent coordinates from a (list of) trajectory files for ease of later analysis. Note that program `filter` (see Sec. 4.29) captures the functionality of `tstrip` as well.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@traj`	⟨input trajectory file(s)⟩

Optional input arguments	
`@nthframe`	⟨write every nth frame (default: 1)⟩
`@time`	⟨time and dt⟩ (overwrites `TIME` in the trajectory files)
`@notimeblock`	(suppresses reading and writing of TIMESTEP block)

Standard output

trajectory file for the solute atoms only

Additional output

none

4.65. `visco` (GROMOS++ program)

Program description:

Program `visco` calculates the bulk and shear viscosities from the elements of the pressure tensor that are written to MD++ energy trajectory files. In order to access this data, `visco` makes use of the `ene_ana` library. To obtain more accurate results from the simulation, the Einstein relation is used instead of the direct evaluation of the autocorrelation function (Green-Kubo formulation). Consider $\mathcal{P}_{\alpha\beta}$ as being an element of the pressure tensor. Consider that $\mathcal{G}_{\alpha\beta}(t)$ is the integral of $\mathcal{P}_{\alpha\beta}dt$:

$$\mathcal{G}_{\alpha\beta}(t) = \int_0^t \mathcal{P}_{\alpha\beta}(t)dt \tag{4.56}$$

We define $\eta_{\alpha\beta}$ as a viscosity term calculated in terms of the integral of the pressure component ($\mathcal{G}_{\alpha\beta}$). It will be proportional to the mean-square "displacements" of $\mathcal{G}_{\alpha\beta}(t)$ in the limit of infinit time.

$$\eta_{\alpha\beta} = \frac{\mathcal{V}}{2k_BT} \lim_{t\to\infty} \langle \frac{[\mathcal{G}_{\alpha\beta}(t+\tau) - \mathcal{G}_{\alpha\beta}(t)]^2}{\tau} \rangle \tag{4.57}$$

where $\mathcal{V}$ is the volume of the periodic box, k_B is the Boltzmann constant and T is the absolute temperature of the system. For isotropic systems, the estimation of the bulk viscosities can be obtained from the average of the viscosity terms obtained from the diagonal components of the pressure tensor:

$$\eta_{bulk} = (\eta_{xx} + \eta_{yy} + \eta_{zz})/3 \tag{4.58}$$

The shear viscosities of an isotropic system can be estimated by averaging the viscosity terms obtained from the off-diagonal elements of the pressure tensor:

$$\eta_{shear} = (\eta_{xy} + \eta_{xz} + \eta_{yz})/3 \quad . \tag{4.59}$$

The time series of the mean square "displacements" of $\mathcal{G}_{\alpha\beta}(t)$ are printed to separate files (`Gxx_msd.dat`, `Gyy_msd.dat`, `Gzz_msd.dat`, `Gxy_msd.dat`, `Gxz_msd.dat`, `Gyz_msd.dat`). In view of the poor statistics for long times, it is up to the user to decide the interval for which the least-squares-fitting should be performed. For convenience, program `visco` also prints the constant $\frac{\mathcal{V}}{2k_BT}$ and the conversion factors with respect to the commonly used units.

Required input arguments

`@en_files`	⟨energy files⟩
`@temp`	⟨temperature⟩
`@library`	⟨library file (same as for `ene_ana`)⟩

Optional input arguments

`@time`	⟨time and dt⟩

Standard output

none

Standard output

`Gxx_msd.dat`, `Gyy_msd.dat`, `Gzz_msd.dat`, `Gxy_msd.dat`, `Gxz_msd.dat`, `Gyz_msd.dat`

4.66. `xrayts` (GROMOS++ program)

Program description:

Program **xrayts** extracts the crystallographic restraints information form a special trajectory. In addition it calculated the minimal normal and free R factors.

Required input arguments

`@restraj` ⟨special trajectory files⟩

Optional input arguments

`@time` ⟨time and dt⟩

Standard output

Time-series of R factors.

Standard output

none

CHAPTER 5

Miscellaneous

5.1. `atominfo` (GROMOS++ program)

Program description:

Internally the GROMOS preparation and analysis tools determine which atoms belong to one molecule based on bonds specified in the topology. These programs can make use of the convenient atom specifier to select atoms, molecular properties etc. For efficiency reasons, the MD engine `md` numbers all atoms in the molecular system sequentially. Program `atominfo` can read both atom specifiers and sequential numbers (GROMOS-numbers) and will list the properties of the selected atoms.

The atom list can be sorted, according to the following priority: solute atom < virtual atom < solvent molecule. All programs that make use of atom specifiers (see Sec. 1.3.1) can also read in a file containing the output of atominfo, by specifying a file (⟨`atominfo` output file⟩). This allows the user to store complicated selections in a file for future use.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@gromosnum`	⟨GROMOS atom number⟩
`@atomspec`	⟨atom specifier (see Sec. 1.3.1)⟩

Optional input arguments	
`@sort`	(sort the atoms)
`@redun`	⟨1 for redundancy check (default), 0 for not (important when generating an atom list for gathering with redundant presence of one or more atoms)⟩

Standard output

list with atomic information of all specified atoms

Additional output

none

5.2. `close_pair` (GROMOS++ program)

Program description:

Program close_pair is to find the closest atom pairs between two molecules in a system with multiple molecules. It is mainly used to propose an atom list for the gathering of a complicated system, and can also be used to analyze the close contacts of two or more molecules in a system.

Periodicity and time series are supported in the program close_pair.

Note: the atom list proposed by the program close_pair represents the atom pairs that are closest to each other in two (specified) molecules, but not necessarily the best choice for gathering since in order to obtain an ideal picture of the unit cell, one should also consider the assembly of the molecules. For this purpose, the closest pairs beween one molecule and all the other molecules are also calculated, and if the closest molecule to the molecule that is to be gathered is not a good reference, one may choose other molecules as the reference, depending on the symmetry of the system.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨boundary type⟩
`@groupA`	⟨atoms to be analyzed that are from the molecule group to be gathered⟩
`@groupB`	⟨atoms to be analyzed that are from the reference molecule group for the gathering of group A⟩
`@traj`	⟨coordinate file⟩

Optional input arguments	
`@dist`	⟨lower limit of distance for searching the closest pair. Default: 0.3 nm⟩
`@time`	⟨t0 and dt⟩

Standard output

The standard output contains two parts:

1. the close atom pairs between each molecule in group A and all molecules that are specified in group B;

2. the Summary part: contains the close atom pairs between two molecules that are closest to each other.

Additional output

A file called atominfo.atomspec is generated with an atom list which is from the Summary part of the standard output. The atom list is formatted and can be directly used by the program atominfo to generate an atominfo atom list file.

5.3. `frameout` (GROMOS++ program)

Program description:

Program frameout can be used to extract individual configurations from a molecular trajectory file. Three different formats are supported: the GROMOS96 format, the PDB format and an VMD-Amber format which can be read by program VMD. The user determines which frames should be written out and if solvent should be included or not. Atom positions can be corrected for periodicity by taking the nearest image to connected atoms, or to the corresponding atom in a reference structure. A centres of mass superposition and least-squares rotational fit to a reference structure can be performed based on selected atoms.

Required input arguments

`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj`	⟨trajectory files⟩

Optional input arguments

`@spec`	⟨specification for writing out frames: `ALL` (default), `EVERY` or `SPEC`⟩
`@frames`	⟨frames to be written out⟩
`@outformat`	⟨output coordinates format, see Sec. 1.2⟩
`@include`	⟨`SOLUTE` (default), `SOLVENT` or `ALL`⟩
`@ref`	⟨reference structure to fit to or to gather with respect to⟩
`@atomsfit`	⟨atom specifier (see Sec. 1.3.1): atoms to fit to⟩
`@single`	⟨write to a single file⟩
`@time`	⟨time and dt⟩ (overwrites `TIME` in the trajectory files)
`@notimeblock`	(suppresses reading and writing of TIMESTEP block)

Standard output

none

Additional output

selected frames are written to files `FRAME_xxxxx.ext`, where `xxxxx` is the frame number of the individual frame and `ext` is determined by the file format

5.4. `inbox` (GROMOS++ program)

Program description:

Even though all GROMOS programs correct for periodic boundary conditions whenever necessary, it can sometimes be quite cumbersome to create a simulation box for display that contains all molecules. For simulations containing one or a few solute molecules, program `frameout` in combination with the proper gathering method will be sufficient, but for molecular systems consisting of many solute molecules, it may be that none of the gather settings works correctly.

Program `inbox` puts the atoms into the positive quadrant of the computational box according to the periodic boundary conditions. It can be used to visualize the computational box in a crystal simulation. The connectivity and gathering of charge groups is ignored, thus the charge groups (and solvent molecules) will not be gathered after application of this program.

One can specify the atoms which are put into the box. All other atoms are not affected by the program. By default all atoms are put into the box.

Required input arguments

`@topo` ⟨molecular topology file (see Sec. 1.2)⟩
`@pbc` ⟨periodic boundary and gathering (Sec. 1.2)⟩
`@traj` ⟨trajectory files⟩

Optional input arguments

`@atoms` ⟨atom specifier (see Sec. 1.3.1): atoms to put in the box⟩

Standard output

shifted coordinates, with all atoms forced to be within the simulation box, in pdb format

Additional output

none

5.5. `pairlist` (GROMOS++ program)

Program description:

Program `pairlist` determines all particles within user specified cutoffs from a given reference point. The reference point can either be an atom specifier (see Sec. 1.3.1) to a single atom or a set of three Cartesian coordinates. The output can be written in the same style as the output of the program `atominfo` to allow usage as an atom specifier (see Sec. 1.3.1) itself.

The program can produce two pairlists at the time, one short-range and one long-range. It will also print out a list of particles that occur in the long-range pairlist only. The pairlist determination can be done on an atomic basis or based on charge groups.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@coord`	⟨coordinates to base the list on⟩
`@refpos`	⟨atom specifier (see Sec. 1.3.1)⟩ or ⟨vector⟩

Optional input arguments	
`@cutp`	⟨small cutoff⟩
`@cutl`	⟨large cutoff⟩
`@type`	⟨ATOMIC (default) or CHARGEGROUP⟩
`@atominfo`	(write in atominfo style)

Standard output

lists of particles within the short-range and long-range cutoff, or in the shell between short- and long-range cutoffs

Additional output

none

5.6. `shake_analysis` (GROMOS++ program)

Program description:

A SHAKE failure in one of the MD engines is one of the few indications that something is going wrong in your simulation. Most often, there is a mismatch between the topology and the set of coordinates, or an extremely strong force between particles is built up otherwise. Program `shake_analysis` is a diagnostic tool that can be used to evaluate all interaction energies for selected atoms, on a coordinate file right before or after a SHAKE failure. The output can be limited by specifying the number of interactions that should be displayed, or by giving an energy cutoff above which interactions should be listed.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@atoms`	⟨atom specifier (see Sec. 1.3.1): atoms for which shake fails⟩
`@cut`	⟨cut-off distance⟩
`@coord`	⟨coordinate file⟩

Optional input arguments	
`@eps`	⟨epsilon for reaction field correction⟩
`@kap`	⟨kappa for reaction field correction⟩
`@top`	⟨number of non-bonded interactions per atom to print⟩
`@higher`	⟨print energies higher than specified value⟩
`@nocov`	(do not print covalent interactions)

Standard output

tables with interaction energies involving the specified atoms

Additional output

none

5.7. `unify_box` (GROMOS++ program)

Program description:

Program `unify_box` can convert different box shapes. All periodic boxes can be described as a triclinic box, which is defined by vectors **a**, **b** and **c**. The program is mostly used to convert a truncated octahedral box into a triclinic box or vice versa, according to[26]. The user can also specify a rotation matrix and **a**, **b** and **c** vectors directly.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩
`@to_pbc`	⟨target boundary condition⟩
`@pos`	⟨coordinate file⟩

Optional input arguments	
`@from_pbc`	⟨original boundary condition⟩
`@rot`	⟨rotation matrix⟩
`@KLM`	⟨**a**, **b** and **c**⟩

Standard output

coordinates in which the molecular system has been rotated to fit the target box shape

Additional output

none

5.8. rot_rel (GROMOS++ program)

Program description:

The rotational relaxation time of molecules can be estimated from the autocorrelation function of the Legendre polynomials of molecular axes $\mathbf{r}_i, \mathbf{r}_j$ and $\mathbf{r}_k$.

$$C_1(t) = \langle \mathbf{r}_i(\tau) \cdot \mathbf{r}_i(\tau + t) \rangle_\tau \quad (5.1)$$

$$C_2(t) = \frac{1}{2}(3\langle \mathbf{r}_i(\tau) \cdot \mathbf{r}_i(\tau + t) \rangle_\tau^2 - 1) \quad (5.2)$$

Program rot_rel calculates the first and second order Legendre polynomials and calculates the time correlation functions. The user specifies two of the molecular axes, the third is defined as the cross product of the first two. The program can average the correlation functions over multiple molecules in the system. Note that the output of this program can also be produced by a combination of programs tser and tcf (see Secs. 4.63 and 4.61, respectively).

Required input arguments

@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pbc	⟨periodic boundary and gathering (Sec. 1.2)⟩
@ax1	⟨specify molecular axis 1⟩
@ax2	⟨specify molecular axis 2⟩
@traj	⟨trajectory files⟩

Optional input arguments

@average	⟨average over all molecules⟩
@time	⟨time and dt⟩

Standard output

autocorrelation functions of the first and second Legendre polynomials of the molecular axes

Additional output

none

5.9. `VMD plugin` (GROMOS++ program)

Program description:

This program is not an individual program but a plugin library which runs in the VMD (Visual Molecular Dynamics) program. It is used to open GROMOS configuration files directly in VMD.

Once a file is opened using one of the GROMOS plugins VMD will prompt for the arguments in the VMD console. Because the topological data in GROMOS is separated from the configurational data the plugin can only roughly guess the data from a configuration file. For this reason it is recommended to give information about the topology and periodic boundary conditions using the arguments. The arguments can also be read from a file (`@f`). Gathering and superpositioning and rotational fitting (to a reference structure or the first structure in the trajectory) can be carried out directly in the plugin.

Required input arguments	
`@topo`	⟨molecular topology file (see Sec. 1.2)⟩

Optional input arguments	
`@pbc`	⟨periodic boundary and gathering (Sec. 1.2)⟩
`@include`	⟨`SOLUTE` (default), `SOLVENT` or `ALL`⟩
`@ref`	⟨reference structure to fit to⟩
`@atomsfit`	⟨atom specifier (see Sec. 1.3.1): atoms to fit to⟩
`@time`	⟨time and dt⟩
`@factor`	⟨factor to convert length unit to Angstrom, 10.0⟩

5.10. xray_map (GROMOS++ program)

Program description:

Program xray_map is used to transform and/or filter crystallographic maps. It reads given CCP4 map files (@map) and atomic coordinates (@pos) and writes the final result to a CCP4 map file (@out) and/or prints some statistics (@stat) to the standard output.

If requested by @expression an expression is evaluated to calculate every grid points value from the maps, which are available in the expression via the symbols rho1, rho2, etc. By default the expression rho1 is evaluated which corresponds of the value of the first map provided. A difference map, for example, can be calculated by giving rho1 - rho2.

The final map can be filtered by a simple cutoff criterion. All grid points closer than a given distance (@cutoff) to given atom centres (@centre) are included in the final map. All other grid points are set to zero.

If @symmetrise is given, the symmetry operations of the space group are applied in oder to create a P 1 map of the while unit cell.

Required input arguments	
@topo	⟨molecular topology file (see Sec. 1.2)⟩
@pos	⟨coordinate file for filtering and expressions⟩
@map	⟨map files⟩
@out	⟨output filename⟩

Optional input arguments	
@stat	⟨print map statistics⟩
@expression	⟨expression to evaluate at every grid point⟩
@centre	⟨AtomSpecifier of centre atoms⟩
@cutoff	⟨grid cell cutoff⟩
@symmetrise	⟨apply symmetry operations to create a P 1 map⟩
@factor	⟨factor to convert length unit to Angstrom⟩

Standard output

Map statistics

Additional output

A CCP4 crystallographic map

Bibliography

[1] C. H. Bennett. Efficient estimation of free-energy differences from Monte-Carlo data. *J. Comput. Phys.*, 22(2):245–268, 1976.

[2] M.R. Shirts, E. Blair, G. Hooker, and V.S. Pande. Equilibrium free energies from nonequilibrium measurements using maximum-likelihood methods. *Phys. Rev. Lett.*, 91:140601, 2003.

[3] X. Daura, W.F. van Gunsteren, and A.E. Mark. Folding-Unfolding Thermodynamics of a beta-Heptapeptide From Equilibrium Simulations. *Proteins*, 34:269–280, 1999.

[4] M. Neumann. Dipole-Moment Fluctiation Formulas in Computer-Simulations of Polar Systems. *Mol. Phys.*, 50(4):841–858, 1983.

[5] A. de Ruiter and C. Oostenbrink. Protein-ligand binding from distancefield distances and Hamiltonian replica exchange simulations. *J. Chem. Theor. Comput.*, 9:883 – 892, 2012.

[6] J. D. Chodera, W. C. Swope, J. W. Pitera, C. Seok, and K. A. Dill. Use of the weighted histogram analysis method for the analysis of simulated and parallel tempering simulations. *J. Chem. Theory Comput.*, 3(1):26–41, 2007.

[7] B.A. Berg. Multicanonical simulations step by step. *Comput. Phys. Commun.*, 153(3):397–406, 2003.

[8] G. Nagy and C. Oostenbrink. Dihedral-based segment identification and classification of biopolymers I: Proteins. *J. Chem. Inf. Model.*, 54:266 – 277, 2014.

[9] G. Nagy and C. Oostenbrink. Dihedral-based segment identification and classification of biopolymers II: Polynucleotides. *J. Chem. Inf. Model.*, 54:278 – 288, 2014.

[10] M.E. Davis, J.D. Madura, B.A. Luty, and J.A. McCammon. Electrostatics and diffusion of molecules in solution: simulations with the university of houston brownian dynamics program. *Comput. Phys. Commun.*, 62:187 – 197, 1991.

[11] C. Peter, W.F. van Gunsteren, and P.H. Hünenberger. Solving the Poisson equation for solute-solvent systems using fast Fourier transforms. *J. Chem. Phys.*, 116:7434–7451, 2002.

[12] C. Peter, W.F. van Gunsteren, and P.H. Hünenberger. A fast-Fourier-transform method to solve continuum-electrostatics problems with truncated electrostatic interactions: algorithm and application to ionic solvation and ion-ion interaction. *J. Chem. Phys.*, 119:12205–12223, 2003.

[13] W. Kabsch and C. Sander. Dictionary of protein secondary structure - Pattern-recognition of hydrigen-bonden and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.

[14] S. Riniker, C.D. Christ, N. Hansen, A.E. Mark, P.C. Nair, and W.F. van Gunsteren. Comparison of enveloping distribution sampling and thermodynamic integration to calculate binding free energies of phenylethanolamine N-methyltransferase inhibitors. *J. Chem. Phys.*, 135:024105, 2011.

[15] N. Hansen, J. Dolenc, M. Knecht, S. Riniker, and W.F. van Gunsteren. Assessment of enveloping distribution sampling to calculate relative free enthalpies of binding for eight netropsin-DNA duplex complexes in aqueous solution. *J. Comput. Chem.*, 33:640–651, 2012.

[16] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1:269–271, 1959.

[17] D. N. Beratan. Electron-Tunneling Pathways in Ruthenated Proteins. *J. Am. Chem. Soc.*, 112:7915–7921, 1990.

[18] C. Schroeder and O. Steinhauser. Using fit functions in computational dielectric spectroscopy. *J. Chem. Phys.*, 132:244109, 2010.

[19] C. Schroeder. Collective translational motions and cage relaxations in molecular ionic liquids. *J. Chem. Phys.*, 135:024502, 2011.

[20] A. de Ruiter and C. Oostenbrink. Extended thermodynamic integration: efficient prediction of lambda derivatives at non-simulated points. *J. Chem. Theory Comput.*, 12:4476 – 4486, 2016.

[21] H. Yu and W.F. van Gunsteren. Charge-on-spring polarizable water models revisited: From water clusters to liquid water to ice. *J. Chem. Phys.*, 121:9549–9564, 2004.

[22] H. G. Katzgraber, S. Trebst, D. A. Huse, and M. Troyer. *Feedback-Optimized Parallel Tempering Monte Carlo Journal of Statistical Mechanics-Theory and Experiment*. Iop Publishing Ltd, 2006.

[23] B. Lee and F. M. Richards. The interpretation of protein structures: estimation of static accesibility. *J. Mol. Biol.*, 55:379–400, 1971.

[24] W. Hasel, T. F. Hendrickson, and W. C. Still. A rapid approximation to the solvent accessible surface areas of atoms. *Tetra. Comput. Method.*, 1:103–116, 1988.

[25] R. Baron, W.F. van Gunsteren, and P.H. Hünenberger. Estimating the configurational entropy from molecular dynamics simulations: anharmonicity and correlation corrections to the quasi-harmonic approximation. *Trends Phys. Chem.*, 11:87–122, 2006.

[26] H. Bekker. Unification of box shapes in molecular simulations. *J. Comput. Chem.*, 18:1930–1942, 1997.

Index

The GROMOS Software for (Bio)Molecular Simulation

Volume 6: Technical Details

January 9, 2021

Contents

CHAPTER 1

Outline of the GROMOS Code

1.1. MD++ outline

The code is split into two parts, the first one being an MD library containing basic functions necessary to run an MD simulation, the second one being the actual MD program. This second part is very small. It is therefore easy to write other specialised MD programs that make use of a subset of the functions provided in the library or apply them in a different order. The source code of the library is in turn split up into nine different parts: *math*, *simulation*, *topology*, *configuration*, *algorithm*, *interaction*, *io*, *util* and *check* (represented as C++ *namespaces*).

- *math* contains classes for vectors, matrices and vector arrays, mathematical operations, physical constants and periodic boundary treatment.
- *simulation* contains the simulation parameters supplied to run an MD or SD simulation or an EM.
- *topology* contains the topology of the simulated system, possibly also including a perturbation topology.
- *configuration* contains the state of a system: its coordinates, velocities, forces, restraints data and so on.
- *algorithm* contains classes that use information from *simulation* and *topology* to act upon a *configuration*. All steps during an MD or SD simulation or EM can be carried out using an *algorithm*.
- *interaction* contains the largest algorithm: the energy, forces and virial evaluation. Here, all interaction terms and their parameters are defined. Because of its size, *interaction* is a separate part, though it formally belongs to *algorithm*. The *interaction* part is further split into *bonded*, *nonbonded* and *special* interactions.
- *io* contains classes to read in or write out information. All file access is block oriented and human readable.
- *util* contains a few extra classes that are necessary to set up a simulation but which do not exactly belong to it. Parsing of command line arguments, generation of initial velocities or setting of debug levels are examples of classes found herein.
- *check* contains test routines. Testing includes the automatic calculation of energies under different conditions as well as the calculation of forces, virial tensor and energy λ-derivatives and their comparison to values obtained by finite difference calculations.

One step of an MD or SD simulation or EM consists of several `Algorithm`s (List. 1.1) applied to the `Configuration` in the right order.

```
1  class Algorithm {
2    public:
3    Algorithm(string name) : name(name) {}
4    ~Algorithm() {}
5    virtual int init(Topology & topo,
6                         Configuration & conf,
7                     Simulation & sim) = 0;
8
9    virtual int apply(Topology & topo,
10                     Configuration & conf,
11                     Simulation & sim) = 0;
12
13   string name;
14 };
```

Listing 1.1. Interface of the `Algorithm` class

The `Algorithm_Sequence` class (List. 1.2) is a container for all these algorithms. When a simulation is set up, they are inserted in the correct order into the `Algorithm_Sequence`. Before the start of a simulation, all algorithms will be initialised (by calling the `init()` function). During an MD step (`Algorithm_Sequence::run()`), the algorithms are applied (by calling `Algorithm::apply()`).

```
1  class Algorithm_Sequence : public vector<Algorithm *> {
2    public:
3    Algorithm_Sequence();
4    ~Algorithm();
5
6    int init(Topology & topo,
7             Configuration & conf,
8             Simulation & sim);
9
10   int run(Topology & topo,
11           Configuration & conf,
12           Simulation & sim);
13
14   Algorithm * algorithm(string name);
15 };
```

LISTING 1.2. Interface of the `Algorithm_Sequence` class. It is a container for `Algorithm` objects which provides methods to initialise and run the contained algorithms. It further provides access by name.

The force-field itself is also an `algorithm`, which, when applied, calculates the energies, forces and virial contribution of all force-field terms for the complete system. The force-field terms themselves are `Interaction` classes. The `Forcefield` is therefore a container to store the different `Interaction` objects (in analogy to the `Algorithm_Sequence` and `Algorithm` classes). When the force-field is applied, it calls `calculate_interactions()` on all interaction objects. There are distinct interaction objects for the covalent interactions (bond-length, bond-angle, improper-dihedral and torsional-dihedral interactions), the non-bonded interactions (pairlist construction, long-range interactions and short-range interactions) and the non-physical interactions (atom-position, atom-distance, dihedral-angle, NOE, ^3J-value or S^2 order-parameter restraints). It is very easy to add a custom `Interaction` class to calculate a non-standard interaction. An overview of the (non-bonded) interaction classes is given in Fig. 1.1. The `Nonbonded_Sets` contain independent subsets of the non-bonded interactions. Their `calculate_interactions()` method may be called in parallel (using either *shared* or *distributed* memory parallelisation). The `Nonbonded_Sets` share (through the `Nonbonded_Interaction`) a pairlist construction algorithm, which they call to create the part of the complete pairlist relevant to them. These different parts of the pairlist stay together with the `Nonbonded_Set` and need never be assembled into the complete pairlist. To gain flexibility, the calculation of the individual atom - atom pair interaction is further split up into a `Nonbonded_Outerloop` (loops over the atom - atom pairs), a `Nonbonded_Innerloop` (prepares the parameters necessary to calculate the interaction) and a `Nonbonded_Term` (calculates the atom - atom pair interaction energy, force and virial contribution). The `Storage` class provides directly accessible (local) memory for each `Nonbonded_Set`.

1.1.1. Efficiency. The main goal for writing a new C++ MD engine was to further improve on modularity (using some object-oriented features) and extendability (using clear and common interfaces between the modules). Nevertheless, a simulation code has to be reasonably efficient to be of practical use. The complete code is written in standard C++[1] with no language extensions or machine-specific parts, resulting in a highly portable program. This means that the compiler has to do all machine specific optimisations. We believe that the absence of any machine specific parts of code, which require duplication to be able to run on different machines, facilitates future modification. Furthermore, current compilers are getting ever better at producing fast programs, making use of the specific features available on the machine. In the inner loops of the interaction calculation, templates are used to generate specialised code. There are, for instance, specialised periodicity classes for the different implemented types of periodic boundary conditions (vacuum, rectangular, truncated octahedral and triclinic). The `Innerloop` methods are called with the boundary type as a template argument. Thus the compiler will generate a different specialised version of the inner loops for different boundary conditions automatically. In the same manner, the interaction function term of the non-bonded interaction can also be chosen (*e.g.* with or without switching function for non-bonded interactions) without any *if* statement required in the compiled inner loop. Example code fragments are shown

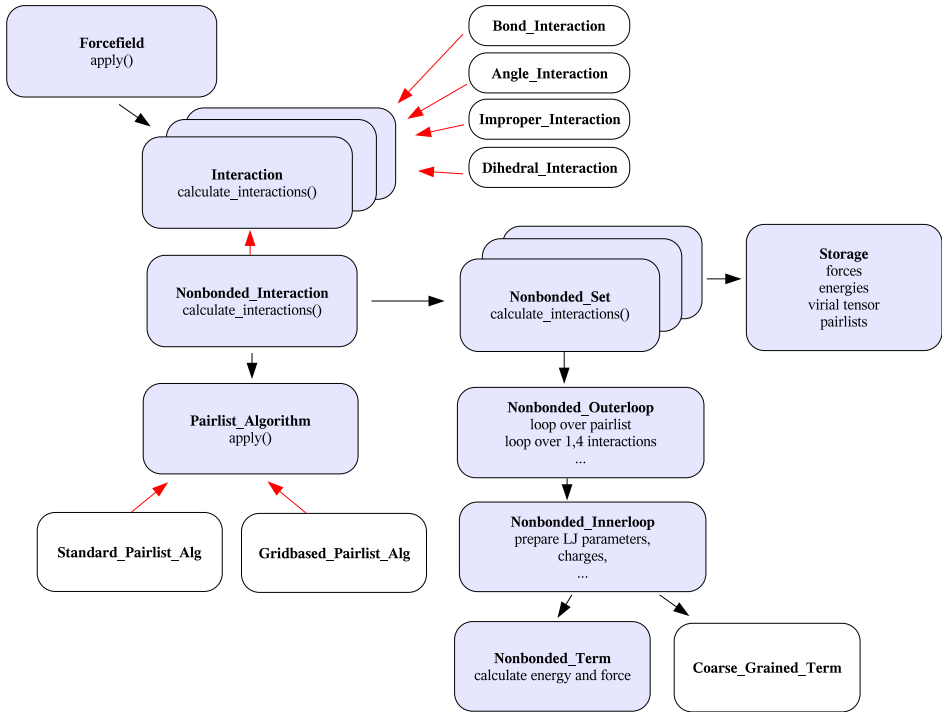


FIGURE 1.1. Illustration of the `Interaction` classes in MD++. The red arrows denote a *is-a* relationship, the black arrows *has-a*. All `Interaction` classes inherit from `Interaction` and, therefore, can be stored in the `Forcefield`, which is a `vector` of `Interaction` classes. The `Nonbonded_Interaction` consists of a `Pairlist_Algorithm` (either a `Standard_Pairlist_Algorithm` or a `Grid_Pairlist_Algorithm`) and (depending on parallelisation) one or more `Nonbonded_Set`s. Those, in turn, consist of `Storage` (to locally store forces, energies, virial tensor and pair lists) and an `Outerloop` (to calculate the interactions). The `Outerloop` relies on the `Innerloop` and on `Term` to calculate the interactions.

in List. 1.3 and List. 1.4. The same technique is used to implement perturbation simulations and different definitions of the virial tensor.

Some algorithms do rely on information from the previous integration step. To help implementing those kinds of algorithms, the complete current and old state (positions, velocities, forces, energies, restraint and constraint data, averages, and so on) of the simulation are stored. During the leap-frog algorithm, the current state becomes the old state and the updated information is stored in the new current state. This transfer is done by a simple and fast pointer exchange. This slightly increases memory usage, but the required space is still small compared to that used to store the pairlists.

1.1.2. Debugging information. It is often difficult to figure out what is going on during an MD or SD simulation or an EM and users tend to use the program as a *black box*. MD++ tries to improve this situation by enabling the user to select a tuneable amount of information to be printed out during the simulation. Every output or debugging message is associated with a debugging level, and the message is printed only if the requested debugging level is high enough. Additionally, every code section belongs to a *module* and a *submodule*. Different debug levels can be specified for all combinations of *modules* and *submodules*. In that way, fine grained control is achieved on how much information from which part of the MD++ code should be printed. For example, running MD++ like this

```
1 ~/> md @f md.args @verb interaction:special:4
```

will print all debug messages in the interaction/special part of the code with a level lower than four. Additional information on debugging can be found in the `doxygen` documentation.

1.1.3. In-code documentation. All classes, structures and enumerations are documented *in-code* using the `doxygen` documentation tool. This documentation contains descriptions of the classes of MD++ and their usage. Inheritance diagrams, function call relationships and interactive links to other classes

```
enum boundary_type {vacuum, rectangular, triclinic};
template<boundary_type boundary>
class Periodicity;

template<>
class Periodicity<vacuum>{
public:
  void nearest_image(Vec const & ri, Vec const & rj, Vec & rij);
};

template<>
class Periodicity<rectangular>{
public:
  void nearest_image(Vec const & ri, Vec const & rj, Vec & rij);
};

template<>
class Periodicity<triclinic>{
public:
  void nearest_image(Vec const & ri, Vec const & rj, Vec & rij);
};

template<boundary_type boundary>
class Interaction{
public:
  virtual int calculate_interactions(Topology const & topology,
                                     Configuration & configuration,
                                     Simulation const & simulation) {

    Vec r;
    Periodicity<boundary> periodicity(configuration.current().box);

    periodicity.nearest_image(
            configuration.current().pos(0),
            configuration.current().pos(1),
            r);
    const double r2 = math::abs2(r);
    // and so on
    return 0;
  }
};
```

LISTING 1.3. Specialized code generation using templates.

```
int main(int argc, char **argv) {
  Interaction<triclinic> interaction;
  interaction.calculate_interactions(
           topology, configuration, simulation);
  return 0;
}
```

LISTING 1.4. The usage of periodic boundary condition specific templates demonstrated on the `Interaction` class.

are automatically generated by the tool. The documentation further contains a brief description of the current input formats used in the given version of MD++. See Sec. 8-3.1 on how to generate the `doxygen` documentation during the compilation procedure of MD++.

1.2. GROMOS++ outline

GROMOS++ is a software package providing the user with tools to prepare all the needed input files for a standard simulation using MD++, e.g. the generation of the molecular topology, initial coordinates of randomly distributed molecules (solvent) or initial coordinates derived from a pdb file (solute), the solvation of a solute in the solvent and the split up of a simulation in multiple jobs with constant or changing simulation parameters over the job sequence. Furthermore, there are multiple programs to analyse the simulations performed. The following is a list of the most important GROMOS++ programs and the corresponding tasks. A complete list is available via the documentation tool, see Sec. 8-3.1 for more information:

- *com_top* combines multiple topology files into one file.

- *dssp* monitors secondary structure elements of a protein, based on the rules defined by Kabsch and Sander[2].
- *ene_ana* analyses (energy) trajectories.
- *frameout* writes out individual configurations or movies from a molecular trajectory file.
- *hbond* monitors the occurrence of hydrogen bonds.
- *ion* replaces water molecules by ions (to get an overall neutral box).
- *make_top* creates molecular topologies from building block and force-field parameter files.
- *mk_script* generates (multiple) script files to run simulations.
- *noe* analyses NOE distances over a trajectory.
- *pdb2g96* converts coordinate files from pdb to the GROMOS file format.
- *ran_box* creates a condensed phase system of any composition (randomly distributed molecules).
- *sim_box* solvates a solute in a box of pre-equilibrated solvent.
- *tser* calculates time series of properties which may be specified flexibly by the user (distances, angles, dihedral angles, intersection angles with planes, ...).

As mentioned before, this list is not complete and a lot of more specific analyses can be done using multiple programs in the right order.

Besides all the programs listed above there is a *contrib* collection of programs, a folder containing some GROMOS++ programs which are not of general use but treat a very specific topic or programs which were replaced by newer versions. A list and short explanation of these programs is available via the documentation tool, see Sec. 8-3.1 for more information.

1.2.1. GROMOS++ source code and in-code documentation. The GROMOS++ source code is divided into two major parts, one containing the programs and *contrib* programs, the other one collecting the tools (classes, structures and enumerations) used within the programs. The second part is in turn split up into eight different parts: *gromos*, *gcore*, *gmath*, *gio*, *bound*, *fit*, *args* and *utils* (represented as C++ *namespaces*):

- *gromos* handles the gromos exceptions (error messages).
- *gcore* contains all the classes that store the information about the molecular system, e.g. angles, bonds, atom properties, Lennard-Jones parameters, information and coordinates of the solvent and many more.
- *gmath* contains the tools of the basic vector and matrix algebra, handles time correlation functions and distributions for a series of values as well. There is also a class to handle a kind of pocket calculations read from a string (useful to mathematically interpret a program input parameter defining some specific properties or calculations).
- *gio* contains the tools to read in data or write them out. The read or written data may for example be a topology, coordinates, building block or input parameter files or any kind of trajectory.
- *bound* contains the classes to handle periodic boundary conditions (rectangular, triclinic, truncated octahedron and vacuum).
- *fit* is the namespace that contains code for translational superpositioning and rotational fitting of configurations.
- *args* contains classes to handle the different command line arguments needed by the programs.
- *utils* is the biggest and most manifold namespace. It contains a class which may perform some basic tests on a molecular topology, classes which provide the tools to observe hydrogen bonds or define secondary structure elements within the backbone of a protein using the rules defined by Kabsch and Sander[2], and many other classes. One of the most used classes within this namespace is probably the class AtomSpecifier: it defines and implements a general form to access atoms in a system. It is used to look over a specific set of atoms, possibly spanning different molecules. An AtomSpecifier is basically a string defining one or a group of atoms, used as an input parameter of a program. More detailed information about the exact format is given in the documentation tool (see Sec. 8-3.1).

All the classes, structures and enumerations of the eight namespaces used in GROMOS++ are documented *in-code* and available via the `doxygen` documentation tool. This also contains a description of all programs together with some example input parameters. Interactive links to other classes are automatically generated and help to understand the specific parts and functions of the code.

CHAPTER 2

Error Messages

Error checking is done in GROMOS with respect to three *types of inconsistencies.*

1. The *array sizes* defined in the header files may not be sufficiently large to cope with the *size of the molecular system* (solute, solvent, restraints, etc.) as specified in the input files. This type of inconsistency is signalled by an error message indicating the subroutine producing the error and that the value of an (input) variable is larger than the array size parameter MAX.... to be found in the header files. So, either the former should be reduced or the latter enlarged.

2. The *files* from which data are read by a program may contain data or data types that are *incompatible with* the *expectations of the program.* This type of inconsistency is signalled by an error message as described under Pt. 1.

3. The *control switches* governing the action of a program may be set such that *incompatible options* or *program actions* are selected. This type of inconsistency is signalled by an error message that specifies the incompatible conditions that have been selected.

The philosophy with respect to error checking in GROMOS is that the *user should be allowed to do silly things*, since what is silly in one case, may be useful in another. This means that only inconsistencies of the first type mentioned above are rigorously checked. GROMOS *error messages state the inconsistency*, so what's wrong, *not what's to be done* to remove the inconsistency. It is up to the user to think of and select the appropriate action to avoid the error message.

With respect to the inconsistencies of the types mentioned above, the error message indicates the line of the file where the error occurs and the name of the program or subroutine. This allows the user to identify and analyse the inconsistency in case the printed error message is not sufficiently informative.

CHAPTER 3

Machine Compatibility

The GROMOS programs, class libraries and subroutines have been written in *standard C++*[1]. This means that GROMOS should compile and run on any machine for which standard C++ compilers are available.

MD++ and GROMOS++ require a set of libraries to carry out numerical calculations. These libraries are written in the C programming language and can be compiled with the same compilers as MD++ and GROMOS++ themselves, See Chap. 8-2. To maximize operating system and compiler compatibility configuration is carried out by a *GNU Autotools* generated configuration script which generates the *Makefile*s and takes care of correct linking of the libraries. All calculations are carried out in 64 bit (double) precision only. Single precision may be available (by some compilers through their options) but is not recommended.

CHAPTER 4

Numerical and Mathematical Functions

The MD++ source code contains a set of mathematical classes and functions to carry out mathematical operations.

4.1. Numerical functions

MD++ contains two *random number generators*.

1. Function *math::RandomGeneratorG96::get()* generates (a series of) *uniformly distributed* random numbers between 0 and 1, using a linear congruential method.

2. Function *math::RandomGeneratorG96::get_gauss()* generates (a series of) normally or *Gaussian distributed* random numbers x with mean $<x>$ and standard deviation σ, using the Box-Müller method. The probability distribution is

$$p(x) \quad = \quad [2\pi\,\sigma^2]^{-1/2}\exp[-(x- < x >)^2/(2\,\sigma^2)]. \tag{4.1}$$

Alternatively, the random number generators as available in the GSL-libraries may be used, which are interfaced via the functions *math::RandomGeneratorGSL::get()* and *math::RandomGeneratorGSL::get_gaussian()*, respectively. The choice of random number generator is determined via the block RANDOMNUMBERS in the MD++ input file.

The Gaussian random number generator is used in program MD++ for the following purposes:

1. Generation of random initial atomic velocities, see Chap. 12.

2. Sampling of the stochastic integrals in a stochastic dynamics simulation, see Chap. 2-13.

4.2. Mathematical functions

4.2.1. MD++. MD++ source code contains a set of mathematical classes and functions to carry out mathematical operations. These classes and functions are grouped into the `math` namespace. Most of the classes are implemented as templates and can be used for either integer and floating point numbers.

1. Class `math::GenericVec<t>` is a generic three-dimensional vector type class. The standard arithmetic operators are overloaded. `math::Vec` is a widely used typedef to `math::GenericVec<double>`.
2. Class `math::GenericMatrix<t>` is a generic 3×3 matrix type class. The standard arithmetic operators are overloaded. `math::Matrix` is a widely used typedef to `math::GenericMatrix<double>`.
3. Class `math::GenericSymmetricMatrix<t>` is a generic 3×3 matrix type class. The only difference to the standard matrix is that it is symmetric.

There are several manipulation functions for vectors and matrices:

1. `product` either calculates the product of a matrix and a vector or the product of a two matrices.
2. `dot` and `cross` are used to calculate the scalar product and the vector product of two vectors.
3. `square` and `pow` are used to calculate the square or power of a matrix.
4. `inverse` calculates the inverse of a matrix.
5. `det` and `trace` compute the determinant and the trace of a matrix.
6. `transpose` gives the transposed of a matrix.

7. `tensor_product` and `dyade` calculate the tensor product of two vectors resulting in a matrix.
8. `symmetric_tensor_product` calculates the tensor product of two vectors assuming the result is a symmetric matrix.
9. `v2s` and `m2s` can be used to convert vectors and matrices to formatted strings.
10. `abs` and `abs2` calculate the length and the squared length of a vector.
11. `norm` normalises a vector to a length of 1.

The standard arithmetic and overloaded functions for vectors `math::GenericVec<t>` and matrices `math::GenericMatrix<t>` are

1. `operator+`/`operator-` either calculates the vector sum/difference or the sum/difference of two matrices/vectors.
2. `operator*`/`operator/` calculates the multiplication/division of a vector with/by a scalar.

The type `math::Box` is very similar to the matrix and is used to represent the box. There are constructors to convert the matrix and the box into each other. `product` can be used the calculate the product of a box with a vector. There are accessor functions to the column vectors of the box matrix.

The type `configuration::GenericMesh<t>` is a generic mesh class which can be used for gridded quantities. It is cuboid and constructed from the number of grid points. There are accessor functions to get and set the value of a grid point and transformation functions to carry out a fast Fourier transform (FFT) using an underlying FFT library (FFTW).

A more detailed description and a list of all available classes and functions can be found in the MD++ `doxygen` under Modules in math.

4.2.2. GROMOS++. The GROMOS++ source code contains a namespace `gmath` including a set of mathematical classes and functions to carry out mathematical operations for vectors and matrices and the calculation of (weighted) distributions and correlation functions. The namespace consists of the following classes:

1. Class `gmath::correlation` is a class to calculate time correlation functions. It calculates almost any kind of correlation function between two time series of scalars or vectors. The data should be provided by either two (or one) vectors of `double`, statistic classes or vectors `gmath:Vec` for vector correlation functions.
2. Class `gmath::Distribution` calculates a distribution for a series of values. The user has to specify an upper and lower bound and number of grid points. After adding all the values a distribution can be written out.
3. Class template `gmath::Stat<t>` is a class template to perform some basic statistics on a series of numbers (`double`, `float`). This class allows one to store a series of numbers and calculates the average, rmsd and an error estimate.
4. Class template `gmath::StatDisk<t>` is similar to the class template `gmath::Stat<t>` but stores the data in a scratch file and not in the memory.
5. Class `gmath::WDistribution` is similar to the class `gmath::Distribution` but calculates a distribution for a series of values with different weights.
6. Class `gmath::Vec` is a three-dimensional vector type class. The standard arithmetic operators are overloaded.
7. Class `gmath::Matrix` is a 3×3 matrix type class. The standard arithmetic operators are overloaded.

The standard arithmetic and overloaded functions for vectors `gmath::Vec` and matrices `gmath::Matrix` are:

- `operator+`/`operator-` either calculates the vector sum/difference or the sum/difference of two matrices.
- `operator*`/`operator/` calculates the multiplication/division of a vector with/by a scalar. The class `gmath::Matrix` is overloaded for multiplications (with a scalar) only.
- `dot` and `cross` are used to calculate the scalar product and the vector product of two vectors.
- `normalize` stretches a vector to a length of 1.
- `abs` and `abs2` compute the length and the squared length of a vector.
- `luDecomp` performs a single value decomposition of a matrix.
- `diagonaliseSymmetric` is used to diagonalise a symmetric matrix. The corresponding eigenvalues are returned.
- `det` and `fastdet3X3Matrix` calculate the determinant of a matrix.

- `transpose` returns the corresponding transposed matrix of a matrix.

CHAPTER 5

Nomenclature

In Vol. 4 the basic principles of storage and *identification of topological and configurational data* concerning a molecular system were discussed. Data or quantities related to e.g. atoms or atom-atom distance restraints, etc. are identified by their *position in the sequence* of such data or quantities, and *not* by their *names*, e.g. atom names or names of restraints, etc. This choice of the sequence number in a list as the key to identifying data has been made to keep GROMOS independent of naming conventions, e.g. for atoms, which may vary through the various fields of application of computer simulation.

For the convenience of the user, however, atoms can be given *atom names* and (sequential) groups of atoms can be given amino acid *residue* or nucleotide or glucose unit, etc. *names*. In principle, such names are only used for writing to file or printing, not as parameters in an algorithm. However, in GROMOS++ programs, atom names or residue names can be used to define a particular selection, as outlined below.

1. In *program make_top* the argument `@seq` is used to define the sequence of building blocks in order to make a topology. In this case, residue names (e.g. CYS1, CYS2, HIS1, HEME, ...) play a role when building a molecular topology from molecular topology building blocks.

2. In various *analysis programs*, atom names can be used as AtomSpecifiers, see Sec. 5-1.3, to select atoms for which a translational superposition and rotational positional *least-squares fit* is to be performed (e.g. in program rmsd, the argument of type atom specifier `@atomsfit` can be set to 1:CA, which means that all CA atoms of molecule 1 are selected). In various *analysis programs*, atom names and residue names can be or are used to define sets of atoms or quantities over which *averages* are to be calculated.

As long as no ambiguity is introduced, GROMOS data files contain atom names and residue or nucleotide names as defined by the *IUPAC-IUB convention*[3].

We note that for a *bond i-j* or an improper *dihedral angle i-j-k-l* the residue name that is associated with the bond or improper dihedral is the *residue name of the first atom, i, in the definition of the bond or improper dihedral*. For a *bond angle i-j-k* or a *torsional dihedral angle i-j-k-l* the residue name that is associated with the bond angle or torsional dihedral is the *residue name of the second atom, j, in the definition of the bond angle or torsional dihedral*.

CHAPTER 6

Units

Different sets of units are used in molecular simulations. In simulations of model systems, such as Lennard-Jones liquids, it is often advantageous to work with dimensionless quantities (*reduced units*) and apply the appropriate scaling to the required units afterwards. When treating realistic molecular systems the use of *Standard International (SI) units* is recommended. Apart from restrictions when storing or printing data in non-exponential format, the GROMOS programs are independent of the chosen units. The units are defined by the ones used for physical constants and atomic or molecular quantities in the (GROMOS) data files.

When choosing the *SI* system it is *recommended* to use the following *basic units.*

- *length*:	r : nm	$= 10^{-9}$m $= 10$ Å
- *mass*:	m: u	= atomic mass unit
		= 1/12 of the mass of a ^{12}C atom
		$= 10^{-3}/N_{\mathrm{Av}}$ kg
		$= 1.6605655\ 10^{-27}$ kg
- *time*:	t: ps	$= 10^{-12}$ s
- *temperature*:	T: K	
- *charge*:	q : e	= electronic charge
		$= 1.6021892\ 10^{-19}$ C

The basic units determine the units for other quantities, e.g. the quoted basic units yield

- *energy*:	E :	kJ mol^{-1}	$= 0.2390$ kcal mol^{-1}
			$= 10^3/\mathrm{N}_{Av}$
			$= 1.6605655\ 10^{-21}$ J
- *force*:	f :	kJ mol^{-1} nm^{-1}	$= 1.6605655\ 10^{-12}$ N
- *pressure*:	P :	kJ mol^{-1} nm^{-3}	$= 10^{30}/N_{\mathrm{Av}}$ Pa
			$= 1.6605655$ MPa
			$= 16.6057$ Bar
			$= 16.3885$ atm
- *velocity*:	v :	nm ps^{-1}	
- *Boltzmann's constant*:	k_B:		$= 8.31441\ 10^{-3}$ kJ mol^{-1} K^{-1}
			(= gas constant)
- *electric field*:	E :	kJ mol^{-1} e^{-1} nm^{-1}	
- *electric dipole*:	μ:	e nm	= 48.032424 D
- *polarisability*:	α:	e^2 nm^2 kJ^{-1} mol	$= 138.9354\ (4\pi\epsilon_0)$ nm^3

We have used the following *physical constants.*

- $N_{\rm Av}$ = Avogadro's number = 6.022045 10^{23} mol^{-1}
- R = gas constant = 8.31441 10^{-3} kJ mol^{-1} K^{-1}
- k_B = Boltzmann's constant = $R/N_{\rm Av}$
  = 1.380662 10^{-26} kJ K^{-1}

Other physical constants required by GROMOS are, again using the basic units quoted above,

- ϵ_0 = permittivity of vacuum
  = 5.727659 10^{-4} kJ^{-1} mol e^2 nm^{-1}
- $(4\pi\epsilon_0)^{-1}$ = 138.9354 kJ mol^{-1} e^{-2} nm
- h = Planck's constant
  = 0.3990313 kJ mol^{-1} ps
- $\hbar$ = $h/\ 2\pi$
  = 0.06350780 kJ mol^{-1} ps
- c = speed of light
  = 2.99792458 10^5 nm ps^{-1}

We note that only a restricted set of units can be chosen independently. For example, if the energy unit is kcal mol^{-1}, the length unit is Angstrom and the mass unit is atomic mass unit, the time unit is a derived unit equal to 0.0488882 ps, which makes the use of these units in simulation a nuisance.

We note that it is possible to use for a quantity in a particular part of a calculation a unit that differs from the unit generally used in GROMOS for the quantity involved. For example, it is general custom to use the unit Hz = 10^{-12} ps^{-1} for ^3J-coupling constants, see Sec. 2-9.7. The recommended SI unit mentioned before would be ps^{-1}. However, one may choose to use the unit Hz for ^3J-coupling constants and the parameters a, b and c in (Eq. 2-9.62) as long as the energy units used for the parameters ${\mathcal{V}^{(Jr)}}_n$ (energy (time)$^{-2}$) are consistent with those of the other interaction terms. The GROMOS data files (Sec. 4-4.11) use kJ mol^{-1} Hz^{-2} for ${\mathcal{V}^{(Jr)}}_n$.

In the *(perturbation) molecular topology file* the atomic *charges* q_i are *stored as such* in the chosen units. In MD++ and GROMOS++, the charges are stored as in the molecular topology file.

A number of *quantities* or interaction function *parameters* in GROMOS are either angles or *dependent on angle units* through their definition.

1. Bond-angle bending interaction Sec. 2-5.2 and Sec. 2-17.2:
   bond angles θ_n (angle)
   parameters θ_n^0 (angle)
   parameters $k_n^{(\theta,h)}$ (energy (angle)$^{-2}$)

2. Improper dihedral-angle bending interaction Sec. 2-5.3 and Sec. 2-17.3:
   improper dihedral angles ξ_n (angle)
   parameters ξ_n^0 (angle)
   parameters $k_n^{(\xi)}$ (energy (angle)$^{-2}$)

3. Trigonometric dihedral-angle torsion interaction Sec. 2-5.4 and Sec. 2-17.4:
   dihedral angles φ_n (angle)

4. Dihedral-angle restraining interaction Sec. 2-9.6:
   dihedral angles φ_n (angle)
   parameters φ_n^0 (angle)
   parameters $k^{(tr)}$ (energy (angle)$^{-2}$)

5. ^3J-coupling constant restraining interaction Sec. 2-9.7:
   dihedral angles η_n (angle)

dihedral angles ζ_n (angle)
parameters δ_n (angle)

6. Local-elevation interaction Sec. 2-9.13.1:

dihedral angles φ_n(angle)
parameters $\varphi_n^{m'}$(angle)
parameters $\Delta\varphi_n^0$(angle)

For convenience of the user these quantities are kept in the *GROMOS data files* using *degrees as angle units*. However, when angles are used in calculations involving mathematical functions such as *sin*, *cos*, etc. they should be expressed in radians. Therefore, upon reading GROMOS data files the values of quantities and parameters that depend on angle units are converted from degrees to radians. So, in the *programs and functions* these quantities and parameters are stored using *radians as angle units.*

Finally, we note that the GROMOS programs can also be used using so-called *reduced units*, which are denoted by *:

- *length*:	r *	=	$r\ /\sigma$
- *mass*:	m *	=	m/M
- *time*:	t *	=	$t/[\sigma\ (M/\epsilon)^{1/2}]$
- *temperature*:	T *	=	$T/[\epsilon/k_B\ ]$
- *charge*:	q *	=	$q/[(4\pi\epsilon_0\sigma\epsilon)^{1/2}]$
- *energy*:	E *	=	$E\ /\epsilon$
- *force*:	f *	=	$f\ /[\epsilon/\sigma\ ]$
- *pressure*:	P *	=	$P\ /[\epsilon/\sigma^{\ 3}]$
- *velocity*:	v *	=	$v\ /[(\epsilon/M)^{1/2}]$

where the parameters ϵ and σ are defined by the Lennard-Jones interaction

$V(r_{ij}) = 4\epsilon\left[(\sigma/r_{ij})^{12} - (\sigma/r_{ij})^{6}\right]$

and M is the mass unit.

CHAPTER 7

Charge Group Codes

The concept of a charge group of atoms has been defined and discussed in Sec. 3-2.6.2. In the GROMOS non-bonded interaction subroutine NONBML the non-bonded energy and forces are only calculated between charge groups: for two (different) charge groups the interaction is, apart from excluded neighbours, either calculated between all or none (if the distance between the charge groups is longer than the cut-off radius) of the atoms forming the charge groups.

The *selection* of atoms belonging to a particular *charge group* is subject to the following *restrictions*.

1. *Atoms belonging to one charge group must have sequential atom sequence numbers.*
2. The solute may contain more than one charge group, but *atoms of different molecules may not belong to one charge group.*
3. *Each solvent molecule consists of one charge group.*

When *defining solute charge groups*, two considerations should be kept in mind.

1. The larger the charge groups, the smaller the total number of charge groups and the faster the charge group pair list can be constructed.
2. The larger the spatial size (radius R^{cg} of a charge group, the larger cut-off radius R_c must be used in order to avoid that the atoms of spatially adjacent charge groups do not interact with each other (see also Sec. 2-4.4):

   $R_c \gg 2 * largest R^{cg}$

The charge group definitions of the GROMOS force-field are given in Tabs. 3-3.12-3-3.16 for the 45A4 and 45B4 GROMOS force fields, Tabs. 3-3.27-3-3.31 for the 54A7 and 54B7 GROMOS force fields and in the figures of Chap. 3-5.

In GROMOS the identification of which atoms of the solute belong to which charge group is done in two different ways.

1. In the GROMOS *data files*, the molecular topology building block file (Sec. 4-5.2) and the molecular topology file (Sec. 4-3.2), *each solute atom i* has a so-called *charge group code* ICGM or ICG: the last atom of a charge group has ICG[i] = 1, whereas the other member atoms of a charge group have ICG[i] = 0.

2. In the *programs* and *functions* a *solute charge group pointer list* INC[1..NCAG] is used to indicate the NCAG charge groups of a solute. The atom sequence number of the last atom of the I -th charge group is stored in INC[I].

Upon reading the GROMOS topology files, the solute charge group codes ICGM[i] or ICG[i] are used to calculate the solute charge group pointer list INC[1..NCAG] (see Sec. 4-3.2 and Sec. 4-5.2).

CHAPTER 8

Pair List Generation

An essential part of the calculation of short-range nonbonded interactions is the construction of a pair list, containing all atom pairs separated by a distance less than the given cutoff distance. Various algorithms for the generation of pair lists are available in MD++.

8.1. Double loop pair list

A pair list is generated by a double loop, looping over $\frac{N_{CG}(N_{CG}-1)}{2}$ possible charge group pairs and checking their respective separation distance against the given cutoff R_c . A double loop pair list algorithm can be applied by setting algorithm=standard(0) in the PAIRLIST block of MD++. The double loop pair list algortihm is slow compared to grid-based pair list algorithms and is not recommended for the simulation of large systems.

8.2. Grid pair list (Heinz and Hünenberger)

A pair list is generated by sorting the charge groups into grid cells with a specified grid index, defined as

$$I_{grid} = N_y N_z i_x + N_z i_y + i_z \ , \tag{8.1}$$

where N gives the number of grid cells in each direction and i_x is defined as $i_x = mod(x, (x_{box}/N_x)$ (similarily for y and z).

For each grid cell, the neighbouring grid cell indices are found by the application of a mask array. This mask array contains the relative grid cell indices to all cells that may possibly be within the cutoff distance of the central cell, and the possibly neighbouring grid cells are found by adding the indices of the mask array to the index of the central cells. Due to the definition of the grid cell index Eq. 8.1, cells are sequential in the z-direction. This allows the mask array to be constructed as stripes, giving the grid cell range of each stripe in the z-direction. The sorted charge groups are stored in an array sorted by the same grid index. The algorithm applies one array of the size $\mathcal{O}(N_{cell}^3)$ pointing to the position in the charge group array of the first charge group of the previous non-empty grid cell. This enables the mask ranges to be converted to ranges in the sorted charge group array. This striping makes the overall algorithm less dependent on the grid size.

The Heinz and Hünenberger grid pair list algorithm can be applied by setting ALGORITHM=2 in the PAIRLIST block of MD++. For further information see[4].

8.3. Grid pair list with expanded coordinates

The original Heinz and Hünenberger pair list algorithm creates the mask such that the periodicity is implicit. This is done by adding the possible box-shifted stripes to the mask. Another approach is to use a non-periodic mask, but expanding the system by duplicating atom coordinates close to the box edge to its periodicly shifted boxes. This requires some overhead in terms of computational speed and memory but prevents the consideration of periodic shifts in the nonbonded calculation routines, and improves data locality.

This modified grid pair list algorithm can be applied by setting ALGORITHM=1 in the PAIRLIST block of MD++.

CHAPTER 9

Boundary Conditions and Periodicity

The concept of periodic boundary conditions has been discussed in Chap. 2-4, where also the formulae for the *three types of periodic boundary conditions* that can be selected in GROMOS were given.

1. *Periodic rectangular box.*

   switch NTB = 1, angle BETA = 90.0
   box lengths BOX[1..3] = a, b, c

2. *Periodic triclinic box.*

   switch NTB = 2, angle BETA[1..3] = α, β, γ
   box lengths BOX [1..3] = a, b, c

3. *Periodic truncated octahedral box.*

   switch NTB = -1, angle BETA = 90.0
   box lengths BOX [1..3] = a, a, a

The use of pressure coupling in a simulation under periodic boundary conditions (Sec. 2-12.5-Sec. 2-12.4) enables variations of the box parameters. The various options for these variations are:

1. no variations of the box parameters (NTP=0);
2. isotropic scaling, *i.e.* identical relative variations of the box-edge lengths only (NTP>0, NPCPL {1,1,1,0,0,0});
3. partially-anisotropic scaling, *i.e.* independent relative variations of the box-edge lengths only (NTP>0, NPCPL {i, j, k, 0, 0, 0} with i,j,k = 0,1,2 or 3 and i≠j or i≠k);
4. fully-anisotropic scaling, *i.e.* independent variations of all box parameters (box-edge lengths, box angles and Euler angles)(NTP>0, NPCPL {1,1,1,1,1,1}).

For a system under vacuum boundary conditions (VBC), only the first option is allowed. For a truncated-octahedral box, only the first two options are allowed. For a rectangular box, only the first three options are allowed. For a triclinic box, all options are allowed.

In GROMOS, the periodic boundary transformations are generally not performed for single atoms, but for all atoms of a charge group, that is, charge groups are (periodically) translated as one entity.

When considering a time series of configurations $\boldsymbol{r}(t_n)$ (n=1,2,3,...$\mathcal{N}_{tot}$), e.g. to obtain averages or time correlation functions, the function $\boldsymbol{r}(t_n)$ should be continuous in the time, that is, the atom positions $\mathbf{r}_i(t_{n-1})$ and $\mathbf{r}_i(t_n)$ at consecutive time points should satisfy the nearest image condition. This is enforced in the analysis programs and will only give correct results if the time difference $t_n - t_{n-1}$ is smaller than the time period that an atom needs to travel over a distance d equal to half the box size (i.e. d should satisfy the relations Eq. 2-4.55 or Eq. 2-4.60 or Eq. 2-4.59 with R_c replaced by d).

When using the GROMOS analysis package GROMOS++, the periodic boundary transformations are controlled by the switch

```
@pbc <boundary type> [<gathermethod>]
     [list <list of atom pairs>]
     [refg <reference structure>]
```

where *boundary type* is the periodic boundary condition used in generating the trajectory:

v - vacuum

r - rectangular

t - truncated octahedral

c - triclinic

The *gathermethod* controls which method is to be used for gathering. During a simulation the molecules in the system can cross the boundaries of the central periodic box. If one would depict the system in such a case, using a molecular visualization program, the molecules would appear as if they were 'broken'. Parts of the molecules that are outside the central box would appear inside of it, but on the opposite edge of the box. The gathering procedure puts the atoms comprising the molecular system into a single central box without 'breaking' the molecules. The available gathering options are:

`nog or 0`	no gathering;
`glist or 1`	gathering of each configuration based on a single list of pairs of atoms that are close to each other; the atom pair should be in the sequence: A B, where A is an atom of the molecule to be gathered, and B is an atom of the reference molecule; if the list argument is not given by the user, gathering is done based on the first atom of the previous molecule; the list argument should be in the atom specifier format;
`gtime or 2`	gathering based on the previous configuration; the first configuration is not gathered;
`gref or 3`	gathering of each configuration based on a reference structure; refg argument required;
`gltime or 4`	gathering based on the previous configuration; the first configuration is gathered based on a single list file; list argument required;
`grtime or 5`	gathering based on the previous configuration; the first configuration is gathered based on a reference structure; refg argument required;
`gbond or 6`	gathering of each configuration based on bond connectivity.
`cog or 7`	gathering based on the centre of geometry of the first molecule.
`gfit or 8`	gather selected molecules based on a reference structure which has been superimposed on the first frame of the trajectory, gather remaining molecules to the cog of selected molecules. Depends on correctly gathered reference ("refg" argument). Molecules to be selecte are specified with the "molecules" argument.

The default option in GROMOS is *glist*. For a single molecule in solution it is often not neccessary to gather the system. If gathering is required, then the methods gbond or gref are the most suitable ones. For systems with more than one molecule and for crystals gathering is usually an essential step in the analysis of trajectories. The correct gathering strongly influences the results of the calculation of atom-positional RMSD, RMSF, and other quantities. Not every method is appriopriate for every case, though. For instance, the *glist* method is not suitable for e.g. lipid bilayers, since the lipids can flip from one layer to the other. The lipid which moved to another layer will be close to completely different lipid molecules than before the flip.

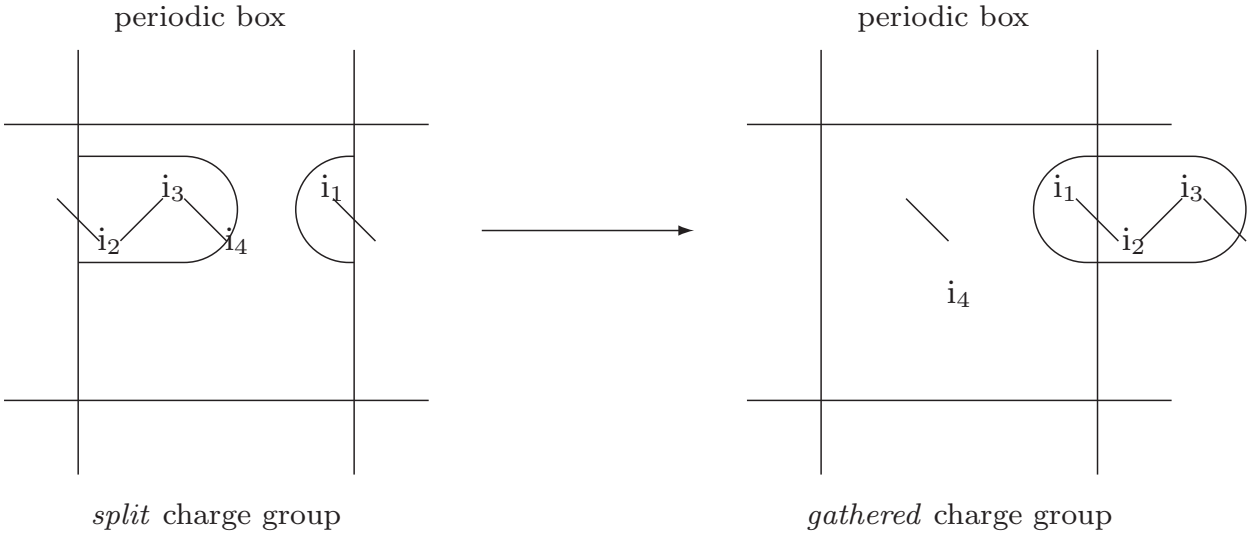


FIGURE 9.1. The atoms with atom sequence numbers i_1, i_2 and i_3 belong to a solute charge group and are gathered such that the spatial closeness of the charge group is not broken by the periodic boundary condition.

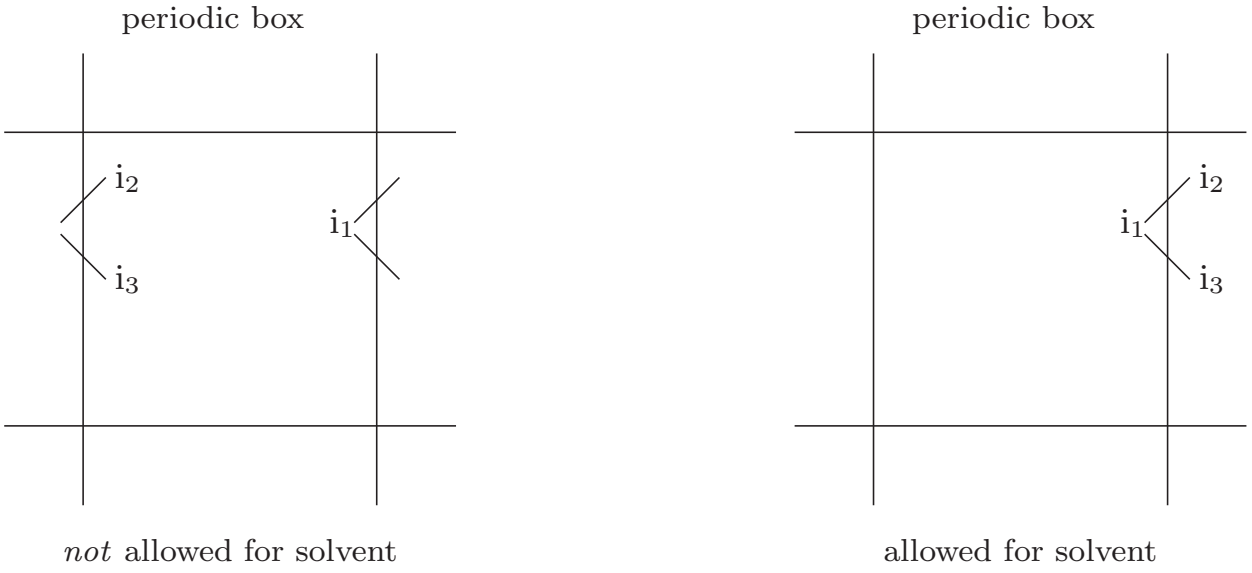


FIGURE 9.2. For the three atoms of a solvent molecule the simulation in the left figure must never occur in GROMOS, that is, the covalent bonds must not be split by the periodic boundary condition. All solvent configurations must be as in the right figure.

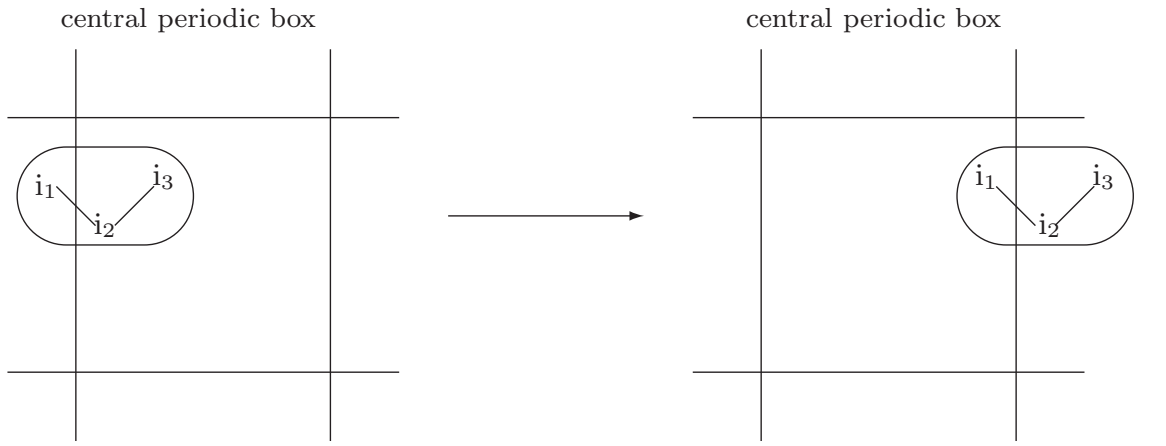


FIGURE 9.3. For the three atoms of a solvent molecule the simulation in the left figure must never occur in GROMOS, that is, the covalent bonds must not be split by the periodic boundary condition. All solvent configurations must be as in the right figure.

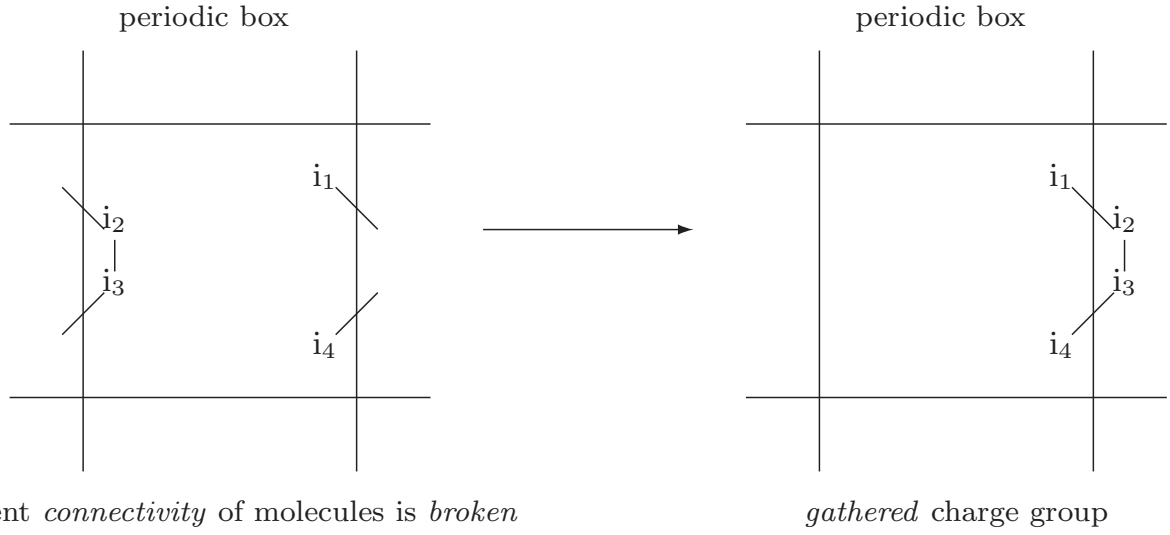


FIGURE 9.4. The covalent connectivity of a molecule with atom sequence numbers i_1, i_2, i_3, i_4 that is broken by the periodic boundary condition, is restored by making the atomic coordinates of atoms for which the sequence numbers differ only by one satisfy the nearest image condition.

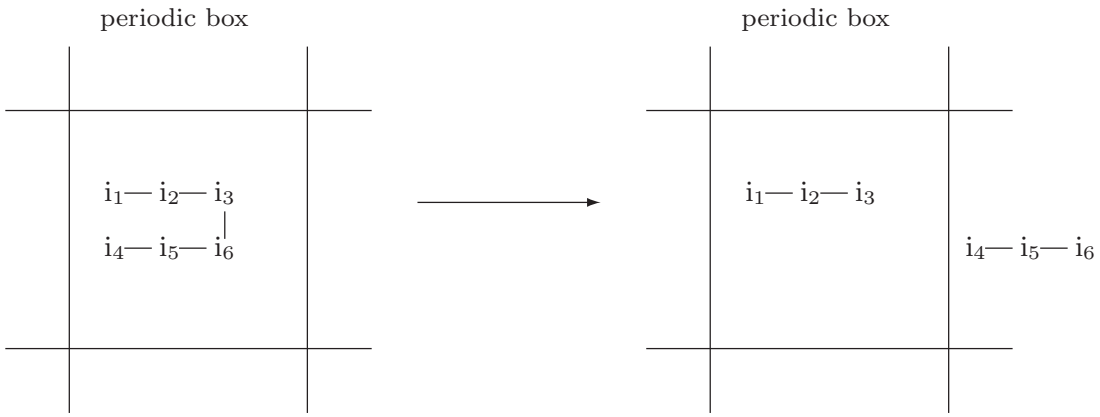


FIGURE 9.5. *Undesired* action of gathering routines.
The atoms are put into the nearest neighbour position with respect to the previous atom in the atom sequence. Since after atom i_3 the numbering of the atoms does not follow more or less (i.e. within half a box length) the covalent connectivity of the chain, the latter is broken between atoms i_3 and i_6 by the gather method. Selecting the proper gather method can be a challenging task.

CHAPTER 10

Generation of Cartesian Coordinates from Internal Coordinates

The configuration of a molecule can be characterized by *different types of coordinates*.

1. *Cartesian coordinates* for all the atoms of the molecule.

2. *Internal coordinates* such as bond lengths, bond angles and dihedral angles, one of each per atom plus the spatial positions of three atoms not lying on a line.

GROMOS++ contains programs and functions performing the transformations from Cartesian to internal coordinates and backwards.

1. Program tser (see Sec. 5-4.63) *calculates bond-length or bond-angle or dihedral-angle values* from sets of Cartesian coordinates.
2. Program gca (see Sec. 5-2.11) *generates Cartesian coordinates* from a set of bond-length, bond-angle and dihedral-angle values.

Using the IUPAC-IUB convention for dihedral angle values the position $\boldsymbol{r}_{l'}$ of atom l after rotation over a dihedral angle $\Delta\varphi$ around the axis defined by the line connecting atoms j and k and starting from a position $\boldsymbol{r}_l$ before the rotation is

$$\mathbf{r}_{l'} = \mathbf{r}_l + [\cos(\Delta\varphi) - 1]\frac{\mathbf{r}_{nk}}{r_{kj}^2} + \sin(\Delta\varphi)\frac{r_{nk}}{r_{kj}^2}\frac{\mathbf{r}_{mk}}{r_{mk}}, \tag{10.1}$$

where

$$\begin{aligned} \mathbf{r}_{kj} &= \mathbf{r}_k - \mathbf{r}_j \\ \mathbf{r}_{lk} &= \mathbf{r}_l - \mathbf{r}_k \\ \mathbf{r}_{mk} &= \mathbf{r}_{kj} \times \mathbf{r}_{lk} \\ \mathbf{r}_{nk} &= \mathbf{r}_{mk} \times \mathbf{r}_{kj} \end{aligned} \tag{10.2}$$

and $\Delta\varphi = \varphi$(i-j-k-l') - φ(i-j-k-l) for example.

Resetting the dihedral angles along a linear covalently bound chain of atoms, as in Fig. 10.1, is straightforward. When rotating atom i_n (n = 4,5,..., N) around bond i_{n-2} - i_{n-1} over $\Delta\varphi$, all atoms i_m with $n < m \leqslant N$ should be rotated over the same angle $\Delta\varphi$ in order to avoid deformation of the molecule. When the chain is branched, as in Fig. 10.2, it must be possible to specify an upper bound M for the atom sequence numbers m so that only the atoms m with $n < m \leqslant M$ of a side chain are rotated with atom n if n denotes a side chain atom.

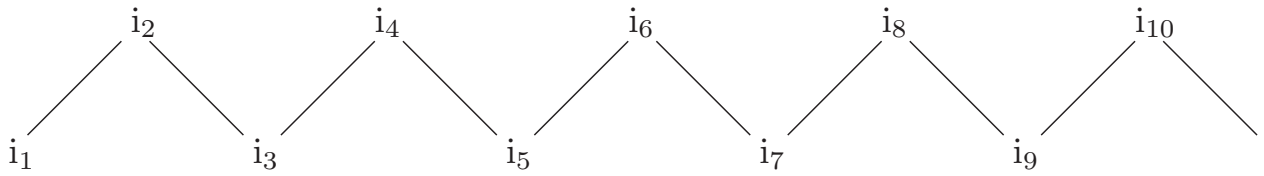


FIGURE 10.1. *Unbranched* covalently bound *chain*. Rotating atom i_4 around $i_2 - i_3$ over an angle $\Delta\varphi$, atoms i_5 to i_{10} should also be rotated over $\Delta\varphi$.

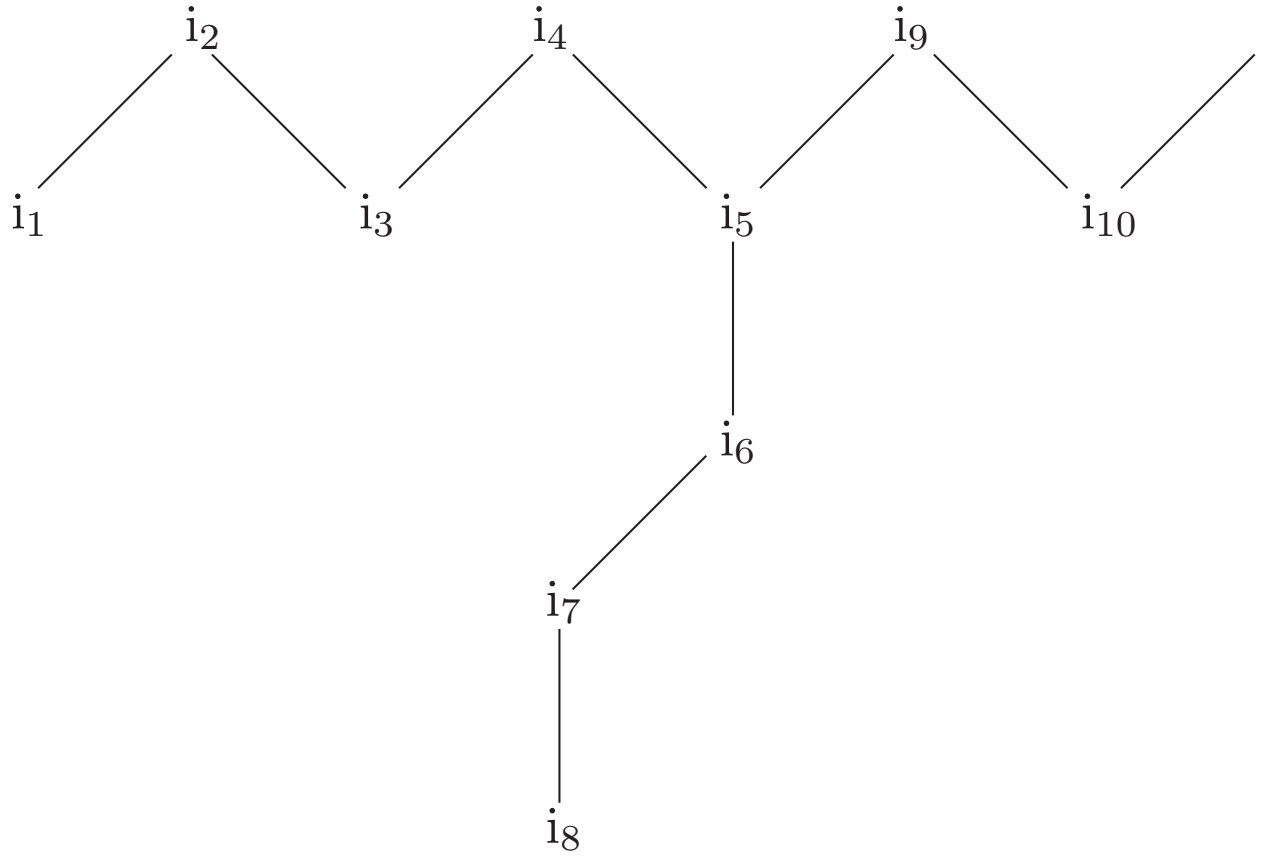


FIGURE 10.2. *Branched* covalently bound *chain*. Rotating atom i_7 around $i_5 - i_6$ over an angle $\Delta\varphi$, atom i_8 should also be rotated over $\Delta\varphi$.

CHAPTER 11

Generation of Hydrogen Atom Coordinates

Generally, molecular configurations as obtained by X-ray diffraction experiments do not contain coordinates for hydrogen atoms. If the molecular model and force field include an explicit representation of hydrogen atoms, their coordinates must be generated using the coordinates of the non-hydrogen atoms of the molecule.

When generating a GROMOS coordinate file from a PDB file, the GROMOS++ program pdb2g96 generates enries for all hydrogen atoms for which no coordinates were present in the PDB file and sets the corresponding Cartesian coordinates to zero (see Sec. 5-2.19).

The GROMOS++ program gch (Sec. 5-2.12) can be used to generate the Cartesian coordinates for the hydrogen atoms. With gch, it is possible to use topological information on bonds, bond angles and dihedral angles to place hydrogen atoms at the optimal location. In cases where the necessary angular parameters are not provided in the topology, gch uses 109.5 degrees for tetrahedral centers and 120 degrees for planar centers.

Eight types of geometry can be handled by gch:

1. An atom (i) is bonded to one hydrogen (H) and one other heavy atom (j). A fourth atom (k) is searched for which is bonded to j and preferably is used to define the dihedral around the j-i bond. The coordinates of H are generated in such a way that the dihedral k-j-i-H is trans and that the angle j-i-H and bond length i-H correspond to their minimum energy values. Considering that b is the bond length i-H and α is the angle j-i-H, the position $\mathbf{r}_H$ of the hydrogen atom is given by

$$\mathbf{r}_H = \mathbf{r}_i + b \cdot \frac{\mathbf{t}}{t}, \tag{11.1}$$

with

$$\mathbf{t} = \cos\alpha \cdot \frac{\mathbf{r}_{ji}}{r_{ji}} - \sin\alpha \cdot \left[\left(\frac{\mathbf{r}_{ji}}{r_{ji}} \right) \times \left(\frac{\mathbf{t}_a}{t_a} \right) \right], \tag{11.2}$$

in which

$$\mathbf{t}_a = \mathbf{r}_{ji} \times \mathbf{r}_{kj}. \tag{11.3}$$

2. An atom (i) is bonded to one hydrogen (H) and two other heavy atoms (j and k). The coordinates of H are generated to be in the plane through j, k and i, on the line bisecting the j-i-k angle and with an i-H bond length corresponding to the minimum energy value in the topology, such that the j-i-H and k-i-H angles are larger than 90 degrees. Considering that b is the bond length i-H, the position $\mathbf{r}_H$ of the hydrogen atom is given by

$$\mathbf{r}_H = \mathbf{r}_i - b \cdot \frac{\mathbf{t}}{t}, \tag{11.4}$$

with

$$\mathbf{t} = \frac{\mathbf{r}_{ji}}{r_{ji}} + \frac{\mathbf{r}_{ki}}{r_{ki}}. \tag{11.5}$$

3. An atom (i) is bonded to two hydrogens (H1 and H2) and one other heavy atom (j). A fourth atom (k) is searched for which is bonded to nh and preferably is used to define the dihedral around the j-i bond. The coordinates of H1 are generated in such a way that the dihedral k-j-i-H1 is trans and that the angle j-i-H1 (α_1) and bond length i-H1 (b_1) correspond to their minimum energy values. The coordinates of H2 are generated to have the angles j-i-H2 (α_2) and H1-i-H2 (α_3) as well as the bond length i-H2 (b_2) at their minimum energy values. If this does not result in a planar configuration around i, the improper dihedral i-j-H1-H2 will be positive.

$$\mathbf{r}_{H1} = \mathbf{r}_i + \mathbf{t}_1, \tag{11.6}$$

with

$$\mathbf{t}_1 = b_1 \left[\cos\alpha_1 \cdot \frac{\mathbf{r}_{ji}}{r_{ji}} - \sin\alpha_1 \cdot \frac{\mathbf{t}_a}{t_a} \right], \tag{11.7}$$

and with

$$\mathbf{t}_a = \frac{\mathbf{r}_{ji}}{r_{ji}} \times \left[\frac{\mathbf{r}_{kj}}{r_{kj}} \times \frac{\mathbf{r}_{ji}}{r_{ji}} \right]; \tag{11.8}$$

and

$$\mathbf{r}_{H2} = \mathbf{r}_i + \mathbf{t}_2, \tag{11.9}$$

with

$$\mathbf{t}_2 = c_1 \frac{\mathbf{r}_{ji}}{r_{ji}} + c_2 \frac{\mathbf{t}_a}{t_a} + c_3 \left[\frac{\mathbf{r}_{kj}}{r_{kj}} \times \frac{\mathbf{r}_{ji}}{r_{ji}} \right], \tag{11.10}$$

in which the scalar coeficients c_1, c_2 and c_3 are given by

$$c_1 = b_2 \cdot \cos\alpha_2 \tag{11.11}$$

$$c_2 = \frac{b_1 \cdot b_2 \cdot \cos\alpha_3 - c_1 \left(\frac{\mathbf{r}_{ji}}{r_{ji}} \cdot \mathbf{t}_1 \right)}{\frac{\mathbf{t}_a}{t_a} \cdot \mathbf{t}_1} \tag{11.12}$$

$$c_3 = \sqrt{b_2^2 - c_1^2 - c_2^2}. \tag{11.13}$$

4. An atom (i) is bonded to three hydrogens (H1, H2 and H3) and one other heavy atom (j). A fourth atom (k) is searched for which is bonded to j and preferably is used to define the dihedral around the j-i bond. The coordinates of H1 are generated in such a way that the dihedral k-j-i-H1 is trans and that the angle j-i-H1 (α_1) and bond length i-H1 (b_1) correspond to their minimum energy values. The coordinates of H2 are such that the angles j-i-H2 (α_2) and H1-i-H2 (α_4) and the bond length i-H2 (b_2) are at their minimum energy values, and the improper dihedral i-j-H1-H2 is positive. The

coordinates of H3 are such that the angles j-i-H3 (α_3) and H1-i-H3 (α_5) and the bond length i-H3 (b_3) are at their minimum energy values and the improper dihedral i-j-H1-H3 has a negative value.

$$\mathbf{r}_{H1} = \mathbf{r}_i + \mathbf{t}_1, \tag{11.14}$$

with

$$\mathbf{t}_1 = b_1 \left[\cos\alpha_1 \cdot \frac{\mathbf{r}_{ji}}{r_{ji}} - \sin\alpha_1 \cdot \frac{\mathbf{t}_a}{t_a} \right], \tag{11.15}$$

and with

$$\mathbf{t}_a = \frac{\mathbf{r}_{ji}}{r_{ji}} \times \left[\frac{\mathbf{r}_{kj}}{r_{kj}} \times \frac{\mathbf{r}_{ji}}{r_{ji}} \right]; \tag{11.16}$$

now for the second hydrogen atom

$$\mathbf{r}_{H2} = \mathbf{r}_i + \mathbf{t}_2, \tag{11.17}$$

with

$$\mathbf{t}_2 = c_1 \frac{\mathbf{r}_{ji}}{r_{ji}} + c_2 \frac{\mathbf{t}_a}{t_a} + c_3 \left[\frac{\mathbf{r}_{kj}}{r_{kj}} \times \frac{\mathbf{r}_{ji}}{r_{ji}} \right], \tag{11.18}$$

in which the scalar coeficients c_1, c_2 and c_3 are given by

$$c_1 = b_2 \cdot \cos\alpha_2 \tag{11.19}$$

$$c_2 = \frac{b_1 \cdot b_2 \cdot \cos\alpha_4 - c_1 \left(\frac{\mathbf{r}_{ji}}{r_{ji}} \cdot \mathbf{t}_1 \right)}{\frac{\mathbf{t}_a}{t_a} \cdot \mathbf{t}_1} \tag{11.20}$$

$$c_3 = \sqrt{b_2^2 - c_1^2 - c_2^2}. \tag{11.21}$$

finally, for the third hydrogen atom

$$\mathbf{r}_{H3} = \mathbf{r}_i + \mathbf{t}_3, \tag{11.22}$$

with

$$\mathbf{t}_3 = c_4 \frac{\mathbf{r}_{ji}}{r_{ji}} + c_5 \frac{\mathbf{t}_a}{t_a} + c_6 \left[\frac{\mathbf{r}_{kj}}{r_{kj}} \times \frac{\mathbf{r}_{ji}}{r_{ji}} \right], \tag{11.23}$$

in which the scalar coeficients c_4, c_5 and c_6 are given by

$$c_4 = b_3 \cdot \cos\alpha_3 \tag{11.24}$$

$$c_5 = \frac{b_1 \cdot b_3 \cdot \cos\alpha_5 - c_4\left(\frac{\mathbf{r}_{ji}}{r_{ji}} \cdot \mathbf{t}_1\right)}{\frac{\mathbf{t}_a}{t_a} \cdot \mathbf{t}_1} \tag{11.25}$$

$$c_6 = \sqrt{b_3^2 - c_4^2 - c_5^2}. \tag{11.26}$$

5. An atom (i) is bonded to one hydrogen atom (H) and three other heavy atoms (j, k, l). The coordinates of H are generated along the line going through atom i and a point corresponding to the average position of j, k and l, such that the bond length i-H (b) is at its minimum energy value and the angles j-i-H, k-i-H and l-i-H are larger than 90 degrees.

$$\mathbf{r}_H = \mathbf{r}_i - b\frac{\mathbf{t}_a}{t_a}, \tag{11.27}$$

with

$$\mathbf{t}_a = \mathbf{r}_{ji} + \mathbf{r}_{ki} + \mathbf{r}_{li} \tag{11.28}$$

6. An atom (i) is bonded to two hydrogen atoms (H1 and H2) and two other heavy atoms (j and k). The coordinates of H1 and H2 are placed above and below the plane going through atoms j, k and i, in such a way that the i-H1 (b_1) and i-H2 (b_2) bond lengths and the angle H1-i-H2 (α) are at their minimum energy values. The improper dihedral angle i-j-k-H1 will be positive.

$$\mathbf{r}_{H1} = \mathbf{r}_i + b_1\left[\sin\left(\frac{\alpha}{2}\right) \cdot \frac{\mathbf{t}_b}{t_b} + \cos\left(\frac{\alpha}{2}\right) \cdot \frac{\mathbf{t}_a}{t_a}\right], \tag{11.29}$$

and

$$\mathbf{r}_{H2} = \mathbf{r}_i - b_2\left[\sin\left(\frac{\alpha}{2}\right) \cdot \frac{\mathbf{t}_b}{t_b} + \cos\left(\frac{\alpha}{2}\right) \cdot \frac{\mathbf{t}_a}{t_a}\right], \tag{11.30}$$

with

$$\mathbf{t}_a = -(\mathbf{r}_{ji} + \mathbf{r}_{ki}), \tag{11.31}$$

and

$$\mathbf{t}_b = \mathbf{r}_{ji} \times \mathbf{r}_{ki}. \tag{11.32}$$

7. An atom (i) is bonded to two hydrogen atoms (H1 and H2), but to no heavy atoms. This is likely to be a (crystallographic) water molecule. First a molecule is generated having the i-H1 aligned in the z-direction and the i-H2 in the z-y plane with the angle H1-i-H2 (α) and bond lengths i-H1 (b_1) and i-H2 (b_2) according to their minimum energy values. This molecule is then rotated around x, y and z by three random angles.

$$\mathbf{r}_{H1} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_1, \tag{11.33}$$

$$\mathbf{r}_{H2} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_2, \tag{11.34}$$

in which $\mathbf{t}_1$ has the coordinates $(0.0, 0.0, b_1)$ and $\mathbf{t}_2$ has the coordinates $(0.0, b_2 \sin\alpha, b_2 \cos\alpha)$ and $\mathbf{R}$ is a matrix that corresponds to rotations around x, y and z by three random angles (ϕ, ψ and θ). $\mathbf{R}$ can be written as:

$$\mathbf{R} = \begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & \cos\phi & -\sin\phi \\ 0.0 & \sin\phi & \cos\phi \end{pmatrix} \times \begin{pmatrix} \cos\psi & 0.0 & \sin\psi \\ 0.0 & 1.0 & 0.0 \\ -\sin\psi & 0.0 & \cos\psi \end{pmatrix} \times \begin{pmatrix} \cos\theta & -\sin\theta & 0.0 \\ \sin\theta & \cos\theta & 0.0 \\ 0.0 & 0.0 & 1.0 \end{pmatrix}. \tag{11.35}$$

8. An atom (i) is bonded to four hydrogen atoms (H1, H2, H3 and H4), but to no heavy atoms. A molecule is generated with all bond lengths (b_1, b_2, b_3 and b_4) at their minimum energy value, the i-H1 aligned in the z-direction, H2 in the x-z plane and H3 such that the improper i-H1-H2-H3 is positive and H4 such that the improper i-H1-H2-H4 is negative. In addition, all H_n-i-H_m angles (α) are set to 109.5°. The complete molecule is then rotated by three random angles around x, y and z. Here we also make use of the rotational matrix ($\mathbf{R}$) defined by Eq. 11.35. Similarly to the above case, we have

$$\mathbf{r}_{H1} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_1, \tag{11.36}$$

$$\mathbf{r}_{H2} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_2, \tag{11.37}$$

$$\mathbf{r}_{H3} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_3, \tag{11.38}$$

$$\mathbf{r}_{H4} = \mathbf{r}_i + \mathbf{R}\mathbf{t}_4, \tag{11.39}$$

in which $\mathbf{t}_1$ has the coordinates $(0.0, 0.0, b_1)$, $\mathbf{t}_2$ has the coordinates $(b_2 \sin\alpha,\ 0.0,\ b_2 \cos\alpha)$, $\mathbf{t}_3$ has the coordinates $(b_3 \sin\alpha \cos 120°,\ b_3 \sin\alpha \sin 120°,\ b_3 \cos\alpha)$, and $\mathbf{t}_4$ has the coordinates $(b_4 \sin\alpha \cos 240°,\ b_4 \sin\alpha \sin 240°,\ b_4 \cos\alpha)$.

CHAPTER 12

Generation of Atomic Velocities

The *atomic velocities* $\mathbf{v}_j$ of a molecular system in equilibrium will obey a *Maxwell distribution* at a given temperature T, that is, the probability that the atomic velocity lies between $\mathbf{v}_j$ and $\mathbf{v}_j + \mathbf{dv}_j$ is

$$P(\mathbf{v}_j)\mathbf{dv}_j \quad = \quad [2\pi k_B T/m_j]^{-\frac{3}{2}} \exp[m_j \mathbf{v}_j^2/(2k_B T)]\mathbf{dv}_j \tag{12.1}$$

where k_B is Boltzmann's constant and m_j the mass of atom j. If no constraints are applied, the 3D velocity distribution has the form of a product of three Gaussian distributions Eq. 4.1,

$$\mathbf{p}(x)dx \quad = \quad [2\pi\sigma^2]^{-\frac{1}{2}} \exp[-(x- < x >)^2/(2\sigma^2)]dx \tag{12.2}$$

for the Cartesian velocity components $\mathbf{v}_{jx}$, $\mathbf{v}_{jy}$ and $\mathbf{v}_{jz}$, each with $< x > = 0$ and Eq. 2-12.58

$$\sigma = [k_B\, T/\, m_j\,]^{\frac{1}{2}}\,. \tag{12.3}$$

If constraints are applied, the velocity components that will induce a violation of the constraints, have to be eliminated. For solute or solvent distance constraints this is done using the procedure to obtain so-called shaken or constrained velocities (Sec. 2-10.3.7). For position constrained or fixed atoms the velocities are simply set equal to zero (Sec. 2-10.2). These operations imply a modification of the sampled unconstrained ($\mathbf{v}_j^{uc}$) velocity distribution to a constrained ($\mathbf{v}_j$) velocity distribution, which may involve a change of properties. For example, the solute or solvent temperatures as calculated from the velocities via (Eq. 2-10.49 and Eq. 2-10.50) or (Eq. 2-10.52 and Eq. 2-10.53), before or after shaking may be different.

$$T(\mathbf{v}_j^{uc}) \neq T(\mathbf{v}_j). \tag{12.4}$$

The constrained velocity distribution can be brought to the desired temperature T by coupling to a temperature bath in a simulation (Sec. 2-12.2).

CHAPTER 13

What to Do when SHAKE Fails

When something goes wrong in a simulation that involves constraints handled by the SHAKE method, it often shows up as a SHAKE error. This means that the atomic coordinate resetting, i.e. from the unconstrained atomic positions $\mathbf{r}^{uc}$ to the constrained atomic positions $\mathbf{r}$ in Eq. 2-10.14, cannot be accomplished within the limit of $N_{sh} = 1000$ iterations over the solute or solvent constraints, or it cannot be accomplished since the deviation between ${\mathbf{r}^{uc}}_{k_2}$ and $\mathbf{r}_{k_2}$ is or has become too large, as illustrated in Fig. 2-10.2. This situation can easily occur when something is wrong with

1. the constraint lengths ${d^0}_{k_1 k_2}$ in Eq. 2-10.11,
2. the reference atomic positions $\mathbf{r}_{k_1 k_2}(t)$ in Eq. 2-10.11,
3. the unconstrained atomic positions ${\mathbf{r}^{uc}}_{k_1 k_2}(t + \Delta t)$ in Eq. 2-10.11,
4. the constrained atomic velocities $\mathbf{v}_i(t - \Delta t/2)$ determining ${\mathbf{r}^{uc}}_i(t + \Delta t)$ through Eq. 2-10.6,
5. the unconstrained atomic forces ${\mathbf{f}^{uc}}_i(t)$ determining ${\mathbf{r}^{uc}}_i(t + \Delta t)$ through Eq. 2-10.6.

So, any anomalously large atomic force or arbitrary modification of the quoted quantities may induce SHAKE to fail. However, due to this sensitivity of SHAKE to incorrect forces, velocities or coordinates, a failure of SHAKE often signals an error for which the cause must be sought elsewhere. Here, we list a few *possible causes of errors showing up in SHAKE*, and what could be done *to identify their cause*.

1. The *length unit* in the *molecular topology* does not match that of the *atomic coordinates*, e.g. nm versus Å. This will result in the energies of the bonds being much too large, see Sec. 2-10.3.
2. The *sequence of the atoms* in the *molecular topology* does not match that of the *atomic coordinates*. This error usually shows up in the bond-angle energies.
3. The *chirality* of the *atomic coordinates* does not correspond to the definition of the *improper dihedral angles* in the *molecular topology* (building blocks). This error will show up in the improper dihedral angle energies.
4. The (initial) *molecular configuration* (atomic coordinates) has a *very high energy* in terms of the force-field used. This error will show up in the output of an energy minimization without constraints: the energies in the zero-th EM step will be large.
5. During a simulation, the *forces* may become *too large*, for example, when positively and negatively charged atoms come too close to each other, SHAKE may not be able to maintain the bonds to these atoms. This error will show up in the output of an MD run without constraints: the energy of the bonds will become large.
6. During a simulation, some forces may act largely perpendicular to an *extended*, *constrained planar group of atoms*, for example the side chain atoms of Arg residues. In such a situation SHAKE is not very efficient, since it attempts to compensate the unconstrained step which is *perpendicular* to the plane of the planar group by modifying iteratively the atomic coordinates *within* the plane of the planar group, see Fig. 13.1. This error shows up in the SHAKE error message. The coordinates of the atoms of the planar group or nearby atoms cannot be reset. Use of a smaller time step Δt (so that the positional changes per step are smaller) or switching off (part of) the solute constraints (Sec. 2-10.4) may help to overcome this situation.

We note that the contribution of the various terms in the force field to the (in)stability of a simulation can be analyzed by switching on and off the various force-field terms. The switches NTF[1..10] (forces), NTPOR

(position restraining), NTDIR (distance restraining), NTDLR (dihedral angle restraining), NTJVR (^3J-value restraining) and NTLES (local elevation biasing) can be used to this end (Secs. 2-12.7 and 2-10.3). The solute constraints can be switched on and off using the switch NTC (Sec. 2-10.4), whereas the solvent constraints cannot be switched off (Sec. 2-10.5).

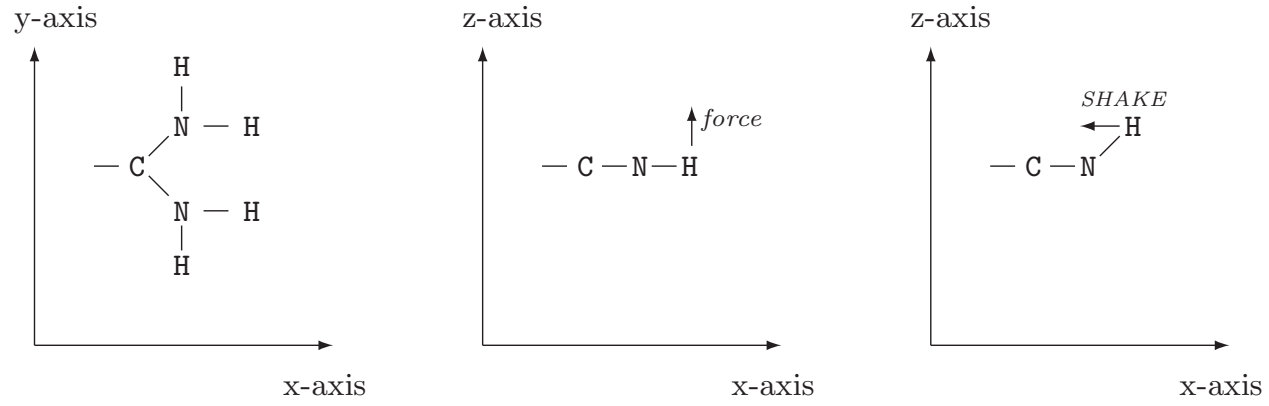


FIGURE 13.1. The inefficiency of SHAKE when forces perpendicular to constrained, extended planar groups of atoms, e.g. in an Arg side chain (left panel), are present. The force (in the z-direction) induces a change of the H atom position out of the (x, y) plane (middle panel), which SHAKE attempts to compensate for by coordinate modification within the (x, y) plane (right panel).

CHAPTER 14

Removal of Centre of Mass Motion

When simulating a system in vacuo, the total translational momentum and the total angular momentum are conserved quantities. When periodic boundary conditions are applied the total angular momentum is not conserved, but the total translational momentum still is. When parts of the system are positionally restrained (e.g. by position restraining, Sec. 2-9.2) neither of these quantities are conserved. In the case of in vacuo or periodic boundary condition simulations, it is common practice to stop the translational motion of the centre of mass and the rotational motion around the centre of mass of the entire molecular system at the start of such a simulation (NTICOM = 1 see Chap. 4-8 block *INITIALISE* for more details) and at regular time intervals afterwards (NSCM $\approx 10^4$) in order to counteract a numerical build-up of centre of mass motion (Sec. 2-4.2, Sec. 2-4.4, and Sec. 2-12.7). Here we present the *algorithm* for*removal of motion of and around the system centre of mass*. It consists of the following steps.

1. Determine the *centre of mass coordinates*

$$\mathbf{r}_{cm} \;=\; M^{-1}\sum_{i=1}^{N} m_i\,\mathbf{r}_i \tag{14.1}$$

of the molecular *system* containing N atoms with masses m_i and positions $\mathbf{r}_i$, where

$$M \;=\; \sum_{i=1}^{N} m_i\,. \tag{14.2}$$

2. Determine the *coordinates* of the atoms *relative* to the system *centre of mass*

$$\mathbf{r'}_i \;=\; \mathbf{r}_i - \mathbf{r}_{cm}\,. \tag{14.3}$$

3. Determine the *system centre of mass velocity*

$$\mathbf{v}_{cm} \;=\; M^{-1}\sum_{i=1}^{N} m_i\,\mathbf{v}_i, \tag{14.4}$$

and calculate the system centre of mass translational kinetic energy

$$\mathcal{K}_{cm,tr} \;=\; \tfrac{1}{2} M\,\mathbf{v}_{cm}^2\,. \tag{14.5}$$

4. Determine the *velocities* of the atoms *relative* to the system *centre of mass translation*

$$\mathbf{v'}_i \;=\; \mathbf{v}_i - \mathbf{v}_{cm}\,. \tag{14.6}$$

5. Determine the system *angular momentum around* the system *centre of mass*

$$\mathbf{L}_{cm} \;=\; \sum_{i=1}^{N} m_i\,\mathbf{r'}_i \times \mathbf{v'}_i\,. \tag{14.7}$$

6. Determine the *inertia tensor* of the system with respect to the centre of mass

$$\underline{\mathbf{I}}_{cm} = \sum_{i=1}^{N} m_i \begin{pmatrix} (r'_i)^2 - (x'_i)^2 & -x'_i\,y'_i & -x'_i\,z'_i \\ -y'_i\,x'_i & (r'_i)^2 - (y'_i)^2 & -y'_i\,z'_i \\ -z'_i\,x'_i & -z'_i\,y'_i & (r'_i)^2 - (z'_i)^2 \end{pmatrix}. \tag{14.8}$$

7. Determine the *angular velocity around* the system *centre of mass* using the inverted inertia tensor

$$\mathbf{O}_{cm} \;=\; \underline{\mathbf{I}}_{cm}^{-1}\,\mathbf{L}_{cm}, \tag{14.9}$$

and calculate the rotational kinetic energy around the system centre of mass

$$\mathcal{K}_{cm,rot} \;=\; \tfrac{1}{2}\,\mathbf{O}_{cm}\cdot\mathbf{L}_{cm} \tag{14.10}$$

8. Determine the *velocities* of the atoms *relative* to the system *centre of mass translation* and to its *rotation* around its centre of mass

$$\mathbf{v''}_i \quad = \quad \mathbf{v'}_i - \mathbf{O}_{cm} \times \mathbf{r'}_i \,. \tag{14.11}$$

CHAPTER 15

Saving Trajectories

The molecular information which is generated by a simulation can be stored in different ways in order to enable the user to analyze the various molecular or system properties after completion of the simulation using specialized analysis programs or so-called post-MD programs (see Vol. 5). Saving all information that is generated in a simulation, i.e. atomic coordinates, velocities, forces, pair energies, etc. at each time point $t_{n'}$ would require an excessive amount of disc space. Moreover, the information stored would be redundant, since the values of the mentioned quantities are highly correlated between subsequent simulation time points t_n, $t_{n+1} = t_n + \Delta t$, etc., since Δt is generally small ≈ 0.002 ps. The information may also be redundant due to dependence between different physical quantities. For example, molecular potential energies and atomic forces can be (re)calculated from atomic coordinates (if no velocity dependent forces are present).

The amount and type of molecular or system information that can be stored during a simulation can be controlled in program MD++ within the WRITETRAJ input block (described in Chap. 4-8) in the following manner:

1. In every simulation the final configuration and velocities and other quantities that are needed to continue the simulation are saved (Sec. 5-1.1) in a so-called *single-configuration file*. These configurations, which generally lie far apart in time, typically 10-100 ps, can be used to obtain a low time-resolution picture of the properties of the molecular system.

2. During a simulation different groups of atomic or system quantities can be saved at different time intervals in different *trajectory files*.
   a. *Atomic coordinates* (Sec. 4-4.2) and possibly corresponding energies (Sec. 4-4.17) and time step data (Sec. 4-4.16) in a *coordinates trajectory file*. The switches NTWX and NTWSE control the saving of configurations:
      (i) If $NTWX \neq 0$ and $NTWSE = 0$, atomic coordinates (POSITION block) are saved at *constant time intervals* $|NTWX| \times \Delta t$, where Δt is the simulation time step. If $NTWX > 0$, *solute plus solvent* atomic coordinates are saved, whereas if $NTWX < 0$, *only solute* atomic coordinates are saved.
      (ii) If $NTWX \neq 0$ and $NTWSE > 0$, the *lowest energy* configuration of each sequential block of |NTWX| simulation time points is saved, while the value of $NTWSE$ specify the type of energy used for the selection, $ENER[NTWSE]$ (refer to Sec. 4-4.17 for the description of the energy types). In this case, the *time intervals* between saved configurations are *variable*, minimally Δt and maximally $|NTWX| \times \Delta t$, so the time and step number and the energy are saved together with the configuration (POSITION block, TIMESTEP block, ENERGY03 block). If $NTWX > 0$, *solute plus solvent* atomic coordinates are saved, whereas if $NTWX < 0$, *only solute* atomic coordinates are saved.
   b. *Atomic velocities* (Sec. 4-4.3) in a *velocity trajectory file*. The switch NTWV controls the saving of velocities. If $NTWV \neq 0$, atomic velocities (VELOCITY block) are saved at *constant time intervals* $|NTWV| \times \Delta t$. If $NTWV > 0$, *solute plus solvent* atomic velocities are saved, whereas if $NTWV < 0$, *only solute* atomic velocities are saved.
   c. *Energies, temperature scaling factors, virial, pressure and computational box size* in a so-called *energy trajectory file*. The switch NTWE controls the saving of these data. If $NTWE \neq 0$, the ENERGY03 block and the VOLUMEPRESSURE03 block, as described in Sec. 4-4.17, are saved at *constant time intervals* $|NTWE| \times \Delta t$.
   d. Data to compute relative *free energies* with respect to changing the coupling parameter λ are saved in a so-called *free-energy trajectory file*. The switch NTWG controls the saving of free

energy data. If $NTWG \neq 0$, the FREEENERDERIVS03 block is saved at *constant time intervals* $|NTWG| \times \Delta t$.

e. *Forces* are saved to a *force trajectory file*. If $NTWF \neq 0$, the FORCE block is saved at *constant time intervals* $|NTWF| \times \Delta t$. If $NTWF > 0$, *solute plus solvent* atomic forces are saved, whereas if $NTWF < 0$, *only solute* atomic forces are saved.

f. *Block-averaged energies* are written to a *block average energies file*. If $NTWB \neq 0$, the block averaged energies (and free energies if $NTWG > 0$) are saved at *constant time intervals* $|NTWB| \times \Delta t$.

g. A *special trajectory file* is written for some specific applications (*e.g.* polarisation, NMR data, X-ray data, ...). The writting of this trajectory is not controlled by the WRITETRAJ input block, but within the specific input blocks. For example, the POLARISE block in MD++ contains a $WRITE$ flag which determines the frequency with which the distances Δr between the charges (of the charge on spring model, see Sec. 2-7.5) will be written to the special trajectory (see Chap. 4-8).

When *choosing* the *time interval* between *saved configurations*, velocities, energies or free energy data, the following points should be considered.

1. Sufficient data points (*i.e.* > 1000) should be available to secure sufficient precision of averages, fluctuations.
2. The larger the accessible part of phase space of the molecular system, the more configurations, etc. may be needed for satisfactory averaging.
3. The longer the relaxation time of the property of interest, the more time points should be saved.
4. The wider the time scales that determine a molecular or system property of interest, the more dense the time points for saving configurations or velocities should be chosen.

For example, configurations are typically stored every 0.1-10 ps. In simulations of molecular liquids comprising a few hundred identical molecules, simulation periods of 10-100 ps are sufficient to obtain precise values for quantities, such as the density, heat of vaporization, diffusion coefficient, rotational correlation times, thermal expansion coefficient, isothermal compressibility, specific heat, excess free energy[5,6]. *Dielectric properties* require much longer (> 1 nsec) simulations in which long-range electrostatic interactions are taken into account. Calculation of the shear *viscosity* requires saving the pressure at every time step for more than a nanosecond[5,6].

GROMOS++ analysis programs usually require one or more trajectory files as input. This is usually performed with specifying the flag @traj (although, other types are also possible: @en_files, @fr_files, ...). See Vol. 5 for more details.

If necessary, GROMOS++ program *tstrip* can be used to remove solvent coordinates from the trajectory file (Sec. 5-4.64). In addition, GROMOS++ program *filter* can be used to filter a coordinate trajectory for a limited set of atoms.

CHAPTER 16

Performing a Translational Superposition and a Rotational Least-Squares Fit

When analyzing or averaging quantities that depend on the atomic position vectors $\mathbf{r}_i(t)$ of the atoms of a (solute) molecule generated in a simulation, it is often desired to separate the internal motions or fluctuations from the centre of mass translation of the molecule and the rotation around the centre of mass of the molecule. The translation of and the rotation around the centre of mass of a molecule can be eliminated from the configurations of a trajectory by superimposing the centres of mass of the sequential configurations and subsequently performing a rotational least-squares fit of the positions of corresponding atoms in the configurations of the trajectory and in the first configuration.

Elimination of the translational centre of mass motion from a trajectory of solute configurations $\mathbf{r}_i(t_n)$ with $n = 1, 2, \ldots, \mathcal{N}_a^{solu}$ is achieved by conversion of the atomic coordinates to atomic coordinates relative to the solute centre of mass for each time frame $t_{n'}$,

$$\mathbf{r}'_i(t_n) = \mathbf{r}_i(t_n) - \mathbf{r}_{COM}(t_n) \tag{16.1}$$

with

$$\mathbf{r}_{COM}(t_n) = \frac{1}{m_{solute}} \sum_{i=1}^{\mathcal{N}_a^{solu}} m_i \mathbf{r}_i(t_n) \tag{16.2}$$

and

$$m_{solute} = \sum_{i=1}^{\mathcal{N}_a^{solu}} m_i \quad . \tag{16.3}$$

Eq. 16.2 is calculated in GROMOS++ using the class `fit::PositionUtils`.

Elimination of the rotational motion of the solute around its centre of mass is achieved by performing a least-squares fit of the position vectors $\mathbf{r}'_i(t_n)$ and $\mathbf{r}'_i(t_m)$ of two different configurations at times t_n and t_m. The atoms $i = 1, 2, \ldots, \mathcal{N}$ for which the superposition is to be carried out, should be chosen from the relatively rigid parts of the solute in order to minimize the effect of internal molecular deformations on the rotational fit. For example, one may use all solute atoms, or only the atoms bearing the name CA or the atoms specified in a list for the rotational fit. The problem is to find an orthonormal 3x3 matrix $\underline{Q}$ which represents a solute rotation

$$\mathbf{r}''_i(t_n) = \underline{Q}\mathbf{r}'_i(t_n) \quad , \tag{16.4}$$

and which minimizes the function

$$\begin{aligned} E(\underline{Q}) &= \sum_{i=1}^{\mathcal{N}} w_i \left[\underline{Q}\mathbf{r}'_i(t_n) - \mathbf{r}'_i(t_m)\right]^2 \\ &= \sum_{i=1}^{\mathcal{N}} w_i \left[\mathbf{r}''_i(t_n) - \mathbf{r}'_i(t_m)\right]^2 \end{aligned} \tag{16.5}$$

where the w_i are weights given to the atoms of the solute. All w_i are zero, except for the atoms for which the rotational least-squares fit is to be performed.

Two procedures to obtain $\underline{Q}$ have been implemented in GROMOS++, proposed by McLachlan[7] and Kabsch[8].

1. The *method of McLachlan*, J. Mol.Biol. **128** (1979) 74-77 in subroutine LSQSTR, file `lsqstr.f`.
2. The *method of Kabsch*, Acta Cryst. **A32** (1976) 922-933 in subroutine LSQSTR, file `lsqstrk.f`.

Given the reference coordinates $\mathbf{r}'_i(t_m)$, the coordinates $\mathbf{r}'_i(t_n)$ are rotated around the origin such that $\mathbf{r}''_i(t_n)$ of Eq. 16.4 are obtained and the function Eq. 16.5 is minimal.

Elimination of solute translation and rotation can be selected in a number of analysis programs (frameout, nhoparam, rmsd, rmsdmat, rmsf and solute_entropy, see Vol. 5) and using the input parameter flags `@atomsfit`.

CHAPTER 17

Transformation between Coordinates

17.1. Cartesian and Oblique Contravariant Crystallographic Coordinates

The result of a crystal structure determination of a molecule using X-ray or neutron diffraction is an electron density distribution function $\rho(\mathbf{r}')$, the number of electrons per unit volume at position $\mathbf{r}'$. It is derived from the measured diffracted beam intensities $I_{obs}(\mathbf{h})$, which are proportional to the square of the amplitude of the crystallographic structure factor

$$F(\mathbf{h}) = \int\limits_{V_c} \rho(\mathbf{r}')\, e^{+2\pi i \mathbf{h}\cdot\mathbf{r}'}\, \mathbf{dr}' \quad , \tag{17.1}$$

i.e. the 3-dimensional Fourier transform of $\rho(\mathbf{r}')$ over a crystal unit cell with volume V_c. So, we have

$$I_{obs}(\mathbf{h}) \alpha |F(\mathbf{h})|^2 \quad . \tag{17.2}$$

The electron density $\rho(\mathbf{r}')$ could be derived from the structure factors $F(\mathbf{h})$ through the inverse of Eq. 17.1,

$$\rho(\mathbf{r}') = V_c^{-1} \sum_h \sum_k \sum_l F(\mathbf{h})\, e^{-2\pi i(hx'+ky'+lz')} \quad , \tag{17.3}$$

where $\mathbf{h}$ is the reciprocal space vector[9]. Since only the amplitudes $|F(\mathbf{h})|$ of the complex structure factors $F(\mathbf{h})$ can be obtained from experiment, it is common practice to postulate an analytical form for the function $\rho(\mathbf{r}')$.

In *isotropic crystallographic refinement* it is assumed that the electron density is distributed as an isotropic Gaussian function around the positions $\mathbf{r}'_j$ of the atoms,

$$P_j(\mathbf{r}') = \left[(2\pi)^{3/2} u_j^3\right]^{-1} e^{-(x'^2+y'^2+z'^2)/(2\,u_j^2)} \quad . \tag{17.4}$$

Fourier transforming Eq. 17.4 yields again a Gaussian distribution in $\mathbf{h}$ space

$$\begin{aligned} \hat{P}_j(\mathbf{h}) &= e^{-2\pi^2 u_j^2 (h^2+k^2+l^2)} \\ &= e^{-2\pi^2 u_j^2 (2\sin\theta/\lambda)^2} \\ &= e^{-B_j (\sin\theta/\lambda)^2} \quad , \end{aligned} \tag{17.5}$$

where we have expressed it in terms of the isotropic atomic B-factor or temperature factor

$$B_j = 8\,\pi^2\, u_j^2 \quad , \tag{17.6}$$

the diffracted beam scattering angle θ and wave length λ.

In *anisotropic crystallographic refinement*, the atomic electron density distribution is assumed to be an ellipsoid Gaussian, so

$$\begin{aligned} \hat{P}_j(\mathbf{h}) &= e^{-(b_{11}h^2+b_{22}k^2+b_{33}l^2+b_{12}hk+b_{13}hl+b_{23}kl)} \\ &= e^{-2\pi^2[U'_{11}(ha^*)^2+(U'_{12}+U'_{21})(ha^*kb^*)+\ldots]} \end{aligned} \tag{17.7}$$

in $\mathbf{h}$-space. The anisotropic temperature factors are usually given in the form of the symmetric 3×3 matrix $\underline{b}$ or $\underline{U}'$, where the prime in the latter is used to indicate that a crystallographic, possibly oblique coordinate system has been used in Eq. 17.7. According to crystallographic convention the real space lengths of the edges of the unit cell are indicated by a, b and c and the corresponding lengths of the reciprocal lattice cell by a^*, b^* and c^*.

Since Newton's equations of motion are not valid in oblique coordinates, simulations of molecular crystal unit cells are performed using (orthogonal) Cartesian coordinates. When the crystallographically refined atomic coordinates $\mathbf{r}'_j$ and anisotropic temperature factors $\underline{U}'_j$ are given in an oblique crystallographic coordinate system, indicated by the prime on the symbols, the quantities have to be transformed to an orthogonal (Cartesian) coordinate system. This can be done in the following way:

The *standard orthogonal coordinate system* with coordinates indicated by x, y and z is defined by

1. The x-axis is the projection of the crystallographic x'-axis on the plane orthogonal to the crystallographic y'-axis.
2. The y-axis coincides with the crystallographic y′-axis.
3. The z-axis is orthogonal to the x-axis and y-axis and defined as in a right-handed Cartesian system.

The *transformation* of the *atomic coordinates* (x', y', z') in the *oblique crystallographic* coordinate system to the atomic coordinates (x, y, z) in the *orthogonal (Cartesian)* coordinate system then reads

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \quad , \tag{17.8}$$

with

$$\begin{aligned} T_{11} &= \sin\gamma \\ T_{22} &= 1 \\ T_{33} &= [\sin^2\alpha - ((\cos\beta - \cos\alpha\cos\gamma)/\sin\gamma)^2]^{\frac{1}{2}} \\ T_{13} &= (\cos\beta - \cos\alpha\cos\gamma)/\sin\gamma \\ T_{21} &= \cos\gamma \\ T_{23} &= \cos\alpha \\ T_{12} &= T_{31} = T_{32} = 0 \end{aligned} \quad , \tag{17.9}$$

where α is the angle between the positive y'- and z'-axes, β is the angle between the positive z'- and x'-axes and γ is the angle between the positive x'- and y'-axes.

The *transformation* of the matrix $\underline{U}'$ of *atomic temperature factors* in the *oblique crystallographic* coordinate system to the matrix $\underline{U}$ of temperature factors in the *(orthogonal) Cartesian* system reads

$$\underline{U} = (\underline{T}\,\underline{D})\;\underline{U'}\;(\underline{T}\,\underline{D})^{\tau} \tag{17.10}$$

where

$$\underline{D} = \begin{pmatrix} d^{-1}\sin\alpha & 0 & 0 \\ 0 & d^{-1}\sin\beta & 0 \\ 0 & 0 & d^{-1}\sin\gamma \end{pmatrix} \quad , \tag{17.11}$$

with

$$d = [1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\,\cos\beta\,\cos\gamma]^{\frac{1}{2}} \quad , \tag{17.12}$$

and the transpose of a matrix is indicated by the superscript τ. So we find

$$\begin{aligned} (\underline{T}\,\underline{D})_{11} &= d^{-1}\sin\alpha\sin\gamma \\ (\underline{T}\,\underline{D})_{22} &= d^{-1}\sin\beta \\ (\underline{T}\,\underline{D})_{33} &= d^{-1}\sin\gamma[\sin^2\alpha - ((\cos\beta - \cos\alpha\cos\gamma)/\sin\gamma)^2]^{\frac{1}{2}} \\ (\underline{T}\,\underline{D})_{13} &= d^{-1}(\cos\beta - \cos\alpha\cos\gamma) \\ (\underline{T}\,\underline{D})_{21} &= d^{-1}\sin\alpha\cos\gamma \\ (\underline{T}\,\underline{D})_{23} &= d^{-1}\cos\alpha\sin\gamma \\ (\underline{T}\,\underline{D})_{12} &= (\underline{T}\,\underline{D})_{31} = (\underline{T}\,\underline{D})_{32} = 0 \quad . \end{aligned} \tag{17.13}$$

For a *monoclinic crystallographic unit cell* (2^{nd} setting, y'-axis unique) we have $\alpha = \gamma = 90^o$, so $d = \sin\beta$ and

$$\underline{T} = \begin{pmatrix} 1 & 0 & \cos\beta \\ 0 & 1 & 0 \\ 0 & 0 & \sin\beta \end{pmatrix} \tag{17.14}$$

and

$$\underline{T}^{-1} = \begin{pmatrix} 1 & 0 & -\cot\beta \\ 0 & 1 & 0 \\ 0 & 0 & 1/\sin\beta \end{pmatrix} \tag{17.15}$$

and

$$\underline{TD} = \begin{pmatrix} 1/\sin\beta & 0 & \cot\beta \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad . \tag{17.16}$$

CHAPTER 18

Distributions, Averages and Root-Mean-Square Fluctuations

The ensemble or trajectory that is produced in a simulation of a molecular system can be analyzed in a variety of ways.

1. Different *scalar quantities* Q , such as energies, atom-atom distances, ^3J-coupling constants, internal coordinates, etc., or *vector quantities* $\vec{Q}$, such as atomic positions, velocities, molecular dipole moments, etc. can be considered.

2. The static equilibrium properties of quantities Q or $\vec{Q}$ can be analyzed in terms of their *probability distribution* or *frequencies of occurrence* of the various Q or $\vec{Q}$ values, $P(Q)$ or $P(\vec{Q})$, in a trajectory.

3. The probability distributions $P(Q)$ or $P(\vec{Q})$ can also be characterized by calculating the *various moments* (p = 1,2,3...)

$$< Q^p > \equiv \frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [Q(t)]^p \tag{18.1}$$

or

$$< \vec{Q}^p > \equiv \frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [\vec{Q}(t)]^p \tag{18.2}$$

of the distributions. The $\mathcal{N}_{tot}$ being the total number of data points (time frames) of the trajectory, and the product of two vectors is to be interpreted as their scalar or dot product. From these moments of the distributions the *mean* or *average*

$$< Q > \equiv \frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} Q(t) \tag{18.3}$$

and

$$< \vec{Q} > \equiv \frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} \vec{Q}(t) \tag{18.4}$$

can be determined. If we assume that the instantaneous value of a given property Q is statistically independent of the other values in the time series of this property, then the variance can be simply taken as

$$\sigma^2(Q) = \frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} (Q(t) - < Q >_{\mathcal{N}_{tot}})^2 \tag{18.5}$$

where $< Q >_{\mathcal{N}_{tot}}$ represents an average of $Q(t)$ over the entire run.

If the configurations are correlated, the block averaging method should be used. The *time series* $Q(t)$ is divided into blocks of length t_b, the average of each block being

$$< Q >_b = \frac{1}{t_b} \sum_{t=1}^{t_b} Q(t) \tag{18.6}$$

The variance can then be estimated with the average values for all blocks of this kind as

$$\sigma^2(< Q >_b) = \frac{1}{n_b} \sum_{b=1}^{n_b} (< Q >_b - < Q >_{\mathcal{N}_{tot}})^2 \tag{18.7}$$

where n_b is the number of blocks.

The quantity $\sigma^2(< Q >_b)$ is expected to be inversely proportional to the block size (t_b), for increasing values of t_b. The goal is now to find the proportionality constant that will allow for an estimation of $\sigma^2(< Q >_b)$ for the single large block that characterizes the entire trajectory ($t_b = \mathcal{N}_{tot}$), for which the statistical inefficiency $\mathcal{S}_{inef}$ is defined as

$$\mathcal{S}_{inef} = \lim_{t_b \to \infty} \frac{t_b \sigma^2(< Q >_b)}{\sigma^2(Q)} \tag{18.8}$$

with the error estimation given by

$$\sigma(< Q >_{\mathcal{N}_{tot}}) = \sqrt{\frac{\mathcal{S}_{inef}}{\mathcal{N}_{tot}}} \times \mathrm{RMSD}(Q) \tag{18.9}$$

where RMSD(Q) is the root mean square deviation of Q . This blocking averaging method is used by the GROMOS++ program ene_ana.

The *root-mean-square fluctuations* can be calculated as

$$\begin{aligned} \Delta Q &\equiv \sqrt{< [Q - < Q >]^2 >} \\ &= \sqrt{\frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [Q(t) - < Q >]^2} \\ &= \sqrt{\frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [Q(t)]^2 - < Q >^2} \end{aligned} \tag{18.10}$$

or for the vector quantity $\vec{Q}$

$$\begin{aligned} \Delta \vec{Q} &\equiv \sqrt{< [\vec{Q} - < \vec{Q} >]^2 >} \\ &= \sqrt{\frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [\vec{Q}(t) - < \vec{Q} >]^2} \\ &= \sqrt{\frac{1}{\mathcal{N}_{tot}} \sum_{t=1}^{\mathcal{N}_{tot}} [\vec{Q}(t)]^2 - < \vec{Q} >^2} \end{aligned} \tag{18.11}$$

When the quantity $\vec{Q}$ is an atomic position, one finds the following relation between the 3-dimensional mean square displacement of an atom j,

$$\begin{aligned} (\Delta \mathbf{r}_j)^2 &= < [\mathbf{r}_j - < \mathbf{r}_j >]^2 > \\ &= 3u_j^2 \\ &= 3B_j/(8\pi^2) \end{aligned} \tag{18.12}$$

and the isotropic atomic B-factor or temperature factor B as described in Chap. 17.

4. The *dynamic properties* of the quantities Q or $\vec{Q}$ can be analyzed in terms of *time series* and *time correlation functions* using the GROMOS++ programs tser and tcf, respectively.

CHAPTER 19

Dihedral-Angle Conventions, Names and Transitions

In the literature *different conventions* for the definition of the value and sign of a *dihedral angle* φ(i-j-k-l) defined by the planes through atoms i, j and k and through j, k and l are in use, see Fig. 19.1

1. *IUPAC-IUB convention*[3].
   The dihedral angle φ_I is considered positive or negative according as, when the system is viewed along the central bond j-k in the direction from j to k (or k to j), the bond i-j to the front atom j (or the bond l-k to the front atom k) requires rotation to the right or to the left, respectively, in order that it may eclipse the bond l-k to the rear atom k (or the bond i-j to the rear atom j).
2. *Polymer convention.*
   The *trans* conformation has $\varphi_p = 0°$. When the system is viewed along the central bond j-k in the direction from j to k (or k to j), a counterclockwise rotation of the bond i-j (or l-k) around the central bond j-k is defined to be positive.

In GROMOS the IUPAC-IUB convention is standardly used.

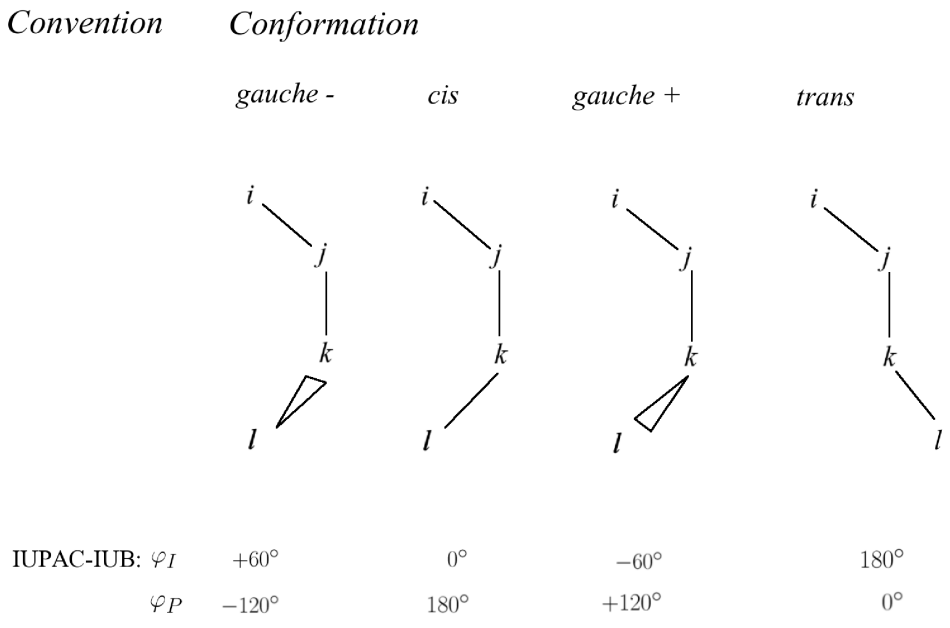


FIGURE 19.1. The relation between a dihedral angle value φ_I according to the IUPAC-IUB convention and the corresponding value φ_p in the polymer convention is $\varphi_p = \varphi_I \pm 180°$. For both conventions the rotation of the angle φ is positive giong from the right to the left through the 4 pictures using the shortest rotation pathway.

The direction of positive rotation is the same in both conventions, only the zero point is shifted by 180°,

$$\varphi_p = \varphi_I \pm 180° \tag{19.1}$$

or

$$\varphi_I = \varphi_p \pm 180° \,. \tag{19.2}$$

The residue name that is associated with a torsional dihedral angle φ(i-j-k-l) is the residue name of the second atom j in the definition of the dihedral angle.

When calculating a dihedral angle value φ using the definition (Eq. 2-5.19) or (Eq. 2-5.14) its value will lie in the range

$$-\pi \leqslant \varphi \leqslant \pi. \tag{19.3}$$

This means that the φ value shows a discontinuity of 2π when passing the point $\varphi = \pm\pi$. When *analyzing trajectories*, that is, dihedral angles as a function of time, the *function $\varphi(t)$ should be made continuous*, i.e. the restriction (Eq. 19.3) should not be applied. This can be achieved by applying the transformation

$$\varphi(t_n) = \varphi(t_n) - NINT((\varphi(t_n) - \varphi(t_{n-1}))/(2\pi)) * 2\pi \tag{19.4}$$

as long as the dihedral angle has not rotated over more than 180° from time point t_{n-1} to time point t_n.

During a simulation the extent of the conformational changes of a solute can be measured by *monitoring the transitions of the torsional dihedral angles* between adjacent minima of the dihedral angle torsional potential energy term (Eq. 2-5.18). The simplest way to define a dihedral angle transition would be to use the passing by the maximum in the torsional dihedral angle energy function $V^{(\varphi)}$, see Fig. 19.2. However, if the dihedral angle φ immediately returns backwards over the barrier, one would not consider such crossing events as two transitions. Therefore, in GROMOS a *dihedral angle transition is only considered to be completed if the dihedral angle passes the bottom of an adjacent well* in the dihedral angle energy function, see Fig. Fig. 19.3.

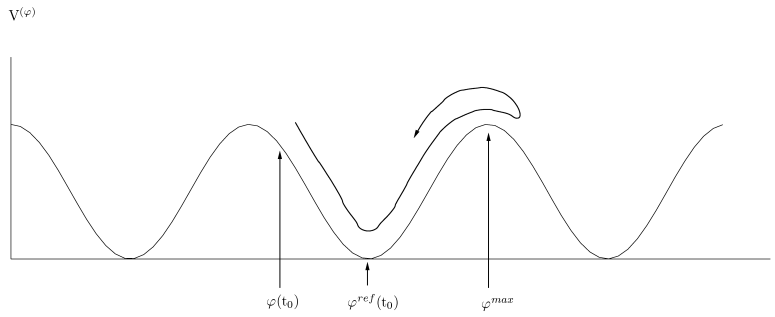


FIGURE 19.2. Monitoring of torsional dihedral angle transitions. A barrier crossing should *not* be counted as a transition.

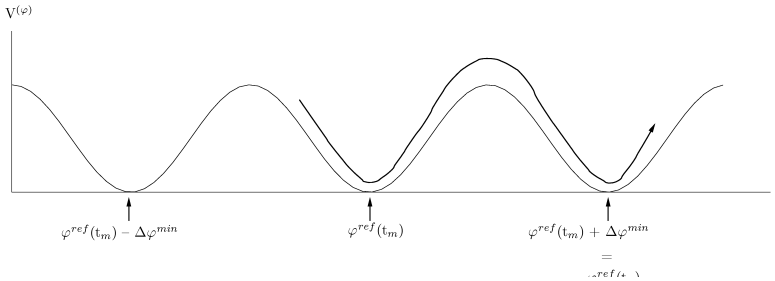


FIGURE 19.3. Monitoring of torsional dihedral angle transitions. A passing by the minimum of an adjacent energy well should be counted as a transition.

The procedure to monitor torsional dihedral angle transitions is the following:

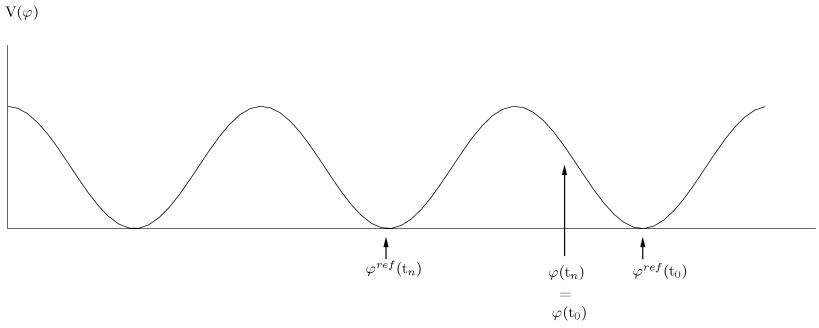


FIGURE 19.4. Monitoring of torsional dihedral angle transitions. Between separate parts of a simulation a transition may be missed.

1. Using the first (t=t_0) molecular configuration, it is determined in which valley of $V^{(\varphi)}$, characterized by the minimum energy dihedral angle $\varphi^{ref}(t_0)$, the dihedral angle $\varphi(t_0)$ is found (Fig. 19.2). The distance between the minima is given by

   $$\Delta\,\varphi^{\min} = 2\pi/m \tag{19.5}$$

   where $m_n^{(\varphi)}$ (=1,2,3,4,5,6) is the multiplicity of the dihedral energy function (Eq. 2-5.18). One of the minima is given by

   $$\varphi^{ref} = \pi[3 - \cos(\varphi_n^0)]/(2m_n^{(\varphi)}) \tag{19.6}$$

   where $\cos(\varphi_n^0) = \pm 1$, see (2.5.5.2). A transformation such as (Eq. 19.7) can be used to bring φ^{ref} within $\pm \quad \pi$ from $\varphi(t_0)$,

   $$\varphi^{ref} = \varphi^{ref} - NINT((\varphi^{ref} - \varphi(t_0))/(2\pi)) * 2\pi, \tag{19.7}$$

   and the minimum $\varphi^{ref}(t_0)$ that is closest to $\varphi(t_0)$ is then found by the transformation

   $$\varphi^{ref}(t_0) = \varphi^{ref} - NINT((\varphi^{ref} - \varphi(t_0))/\Delta\,\varphi^{\min}) * \Delta\,\varphi^{\min}\,. \tag{19.8}$$

   The relation between $\varphi^{ref}(t_0)$ and $\varphi(t_0)$ is illustrated in Fig. Fig. 19.2.
2. At each time point t_n the occurrence of a dihedral angle transition since the last transition at time point t_m is checked. If

   $$|\varphi(t_n) - \varphi^{ref}(t_m)| > \Delta\,\varphi^{\min} \tag{19.9}$$

   a transition is registered and the reference angle is set equal to the minimum of the well that has just been passed by (Fig. Fig. 19.3)

   $$\varphi^{ref}(t_n) = \varphi^{ref}(t_m) + \Delta\,\varphi^{\min}\,\mathrm{sign}(\varphi(t_n) - \varphi^{ref}(t_m)). \tag{19.10}$$

This procedure delivers all transitions that have occurred since the start of a simulation at t=t_0. However, if the monitoring of dihedral transitions is not continuous, but restarted, for example for each separate part (job) of a simulation, a dihedral transition may be missed. This is illustrated in Fig. 19.4. If the first part of a simulation ends with $\varphi(t_n)$ and $\varphi^{ref}(t_n)$ as indicated in Fig. 19.4, the first configuration of the second part, i.e. $\varphi(t_n)$, is used to determine the new $\varphi^{ref}(t_0)$, which will differ (by $\Delta\varphi^{min}$) from the $\varphi^{ref}(t_n)$ at the end of the first part.

CHAPTER 20

Definition of Hydrogen Bonds

Hydrogen bonds may be defined by specifying donor and acceptor atoms and an energetic or geometric criterion. GROMOS++ contains a program called *hbond* (Sec. 5-4.32) which determines the occurrence of hydrogen bonds in a molecular system containing solute and solvent molecules using a geometric criterion.

The *geometry of a hydrogen bond* (Fig. 20.1) is defined by

1. a maximum distance d(H-A) between the hydrogen (H) atom and the acceptor (A) atom, and
2. a minimum angle θ(D-H-A) between the donor (D) atom, the hydrogen (H) and the acceptor (A) atom.

A so-called *three centre hydrogen bond* is defined by the additional specification of

3. a minimum value for the sum of the three angles θ(D-H-A1), θ(D-H-A2) and θ(A1-H-A2), and
4. a maximum value for the (improper) dihedral angle φ(H-A2-A1-D).

FIGURE 20.1. Left: definition of a hydrogen bond using a maximum distance d(H-A) and a minimum angle θ(D-H-A). Right: a three-centre hydrogen bond is defined by the additional specification of a minimum value for the sum of the three angles θ(D-H-A1), θ(D-H-A2) and θ(A1-H-A2), and a maximum value for the (improper) dihedral angle φ(H-A2-A1-D).

In the program, two groups of atoms (A and B) can be specified between which the hydrogen bonds are to be monitored. The hydrogen bond *donor* and *acceptor* atoms are either *identified* using the *atom specifier*, by a *mass file* or a reference structure. In case of the *mass file*, the donor and acceptors are filtered based on their masses which are given in the file (HYDROGENMASS and ACCEPTORMASS block). If a reference structure is provided, only the hydrogen bonds observed in the reference structure are monitored.

In addition, time series of specific hydrogen bonds can be generated using the GROMOS++ program *tser* (Sec. 5-4.63). Thereby is the *property specifier* set to *hb* and the involved atoms (with *atom specifiers*), the distance and angle are to be specified. The default distance is 0.25 nm and the default angle 135 degrees.
General: hb%<atomspec>%<dist_upper>%<angle_lower>
Example: hb%1:res(3:N,H);1:res(5:O)
means the hydrogen bond between the H atom of residue 3 and the O atom of residue 5 of the first molecule.

For *three centre hydrogen bonds*, the default lower angle is 90 degrees, the default sum of angles is 340 degrees and the default angle of the plane is 15 degrees.
General: hb%<atomspec>%<dist_upper>%<angle_lower>%<angle_sum>% <angle_plane>
Example: hb%1:res(3:N,H);1:res(5:O);1:res(6:O)
means the *three centre hydrogen bond* between atom H of residue 3 and atom O of residues 5 and 6 of molecule 1.

CHAPTER 21

Time Correlation Functions and Spectral Densities

As was mentioned in Chap. 18, the dynamic properties of a scalar quantity Q or a vector quantity $\vec{Q}$ can be analyzed in terms of

1. *time series* $Q(t)$ or $\vec{Q}(t)$, and

2. *time correlation functions* $C_Q(t)$ or $C_{\vec{Q}}(t)$.

The time correlation function $C_Q(t)$ of a quantity $Q_i(t)$, or two quantities $Q_i(t)$ and $Q_j(t)$ is defined by

$$\begin{aligned} C_Q(t) &= < Q_i(t') * Q_j(t'+t) >_{t'} \\ &= [t_{\rm MD} - t]^{-1} \int_0^{t_{\rm MD}-t} Q_i(t') Q_j(t'+t) dt' \end{aligned} \tag{21.1}$$

where $t_{\rm MD}$ is the length of the simulation. From a trajectory file the quantities $Q_i(t)$ and $Q_j(t)$ can only be calculated at $\mathcal{N}_t$ discrete, equally spaced time points $n\Delta t$ with n = 0,1,..., $\mathcal{N}_t - 1$. The discrete equivalent of (Eq. 21.1) is then

$$C_Q(n\Delta t) = [\mathcal{N}_t - n]^{-1} \sum_{k=0}^{\mathcal{N}_t - n - 1} Q_i(k\Delta t) Q_j((k+n)\Delta t). \tag{21.2}$$

This formula (Eq. 21.2) requires computation time proportional to ${\mathcal{N}_t}^2$. A much faster method based on the convolution theorem combined with a fast Fourier transform (FFT) algorithm is the following. The discrete Fourier transform of the quantity $Q(t)$ with respect to time t is

$$\hat{Q} = \sum_{k=0}^{\mathcal{N}_t - 1} Q(k\Delta t) e^{+im\Delta\omega k\Delta t} \tag{21.3}$$

where m = 0,1,..., $\mathcal{N}_t - 1$ and

$$\Delta\omega = \frac{2\pi}{\mathcal{N}_t \Delta t}. \tag{21.4}$$

By taking the Fourier transform of (Eq. 21.2) and using the convolution theorem the summation (integral) reduces to a product of the Fourier transformed function $\hat{Q}$ and the complex conjugate $\hat{Q}^*$, which may be subsequently inversely transformed to obtain the time correlation function

$$C_Q(n\Delta t) = [\mathcal{N}_t(\mathcal{N}_t - n)]^{-1} \sum_{m=0}^{N_t - 1} \hat{Q}_i(m\Delta\omega) \hat{Q}_j(m\Delta\omega)^* e^{-im\Delta\omega n\Delta t} \tag{21.5}$$

where $C_Q(t)$ is assumed to be periodic with period $\mathcal{N}_t\Delta t$. This assumption introduces spurious correlations in $C_Q(t)$, which can be avoided by simply adding a series of $\mathcal{N}_t$ zeros to the $n = 0, 1, \ldots, \mathcal{N}_t - 1$ known values $Q_i(n\Delta t)$ and $Q_j(n\Delta t)$. The summation in (Eq. 21.3) and (Eq. 21.5) then contains $2\mathcal{N}_t$ terms and the *FFT expression* for the time correlation function becomes

$$C_Q(n\Delta t) = [2\mathcal{N}_t(\mathcal{N}_t - n)]^{-1} \sum_{m=0}^{2N_t - 1} \hat{Q}_i(m\Delta\omega) \hat{Q}_j(m\Delta\omega)^* e^{-im\Delta\omega n\Delta t} \tag{21.6}$$

with

$$.\Delta\omega = \frac{2\pi}{2\mathcal{N}_t \Delta t} \tag{21.7}$$

This expression for the time correlation function requires computation time proportional to $\mathcal{N}_t lg \mathcal{N}_t$.

The *spectral density* of the *time correlation function* $C_Q(t)$ is its Fourier transform $\hat{C}_Q(\omega)$, or in discrete form

$$\hat{C}_Q(m\Delta\omega) = \sum_{k=0}^{\mathcal{N}_t-1} C_Q(k\Delta t)e^{+im\Delta\omega k\Delta t}. \tag{21.8}$$

Introduction of spurious components by the Fourier transform (Eq. 21.8) can be avoided by making the function to be transformed periodic (with period $2\mathcal{N}_t$), which can be achieved by adding the inverse sequence of $C_Q(t)$ values after $C_Q((\mathcal{N}_t-1)\Delta t)$. Then we find

$$\hat{C}_Q(m\Delta\omega) = \sum_{k=0}^{\mathcal{N}_t-1} C_Q(k\Delta t)e^{+im\Delta\omega k\Delta t} + \sum_{k=\mathcal{N}_t}^{2\mathcal{N}_t-1} C_Q((2\mathcal{N}_t-k-1)\Delta t)e^{+im\Delta\omega k\Delta t}. \tag{21.9}$$

The GROMOS++ program tcl (see Vol. 5) is able to calculate distributions and time-correlation functions form any series of data points. The time correlation functions of the general form

$$C_Q(t) =< f(Q_i(\tau), Q_j(\tau+t)) >_\tau \tag{21.10}$$

can be calculated, where $Q_i(\tau)$ and $Q_j(\tau+t)$ represent the data points at different time points and the user can specify any function $f(Q_i, Q_j)$ which is then inserted into (Eq. 21.2). The program can calculate both auto-correlation functions ($Q_i = Q_j$) and cross correlation functions ($Q_i \neq Q_j$) for time series of scalars or vectors. If Q_i and Q_j are represented by scalars and $f(Q_i, Q_j) = Q_i(\tau) * Q_j(\tau+t)$, the program makes use of fast Fourier transform to calculate $C_Q(t)$. In other cases the direct summing algorithm is used.

21.1. Use of fast Fourier transform (FFT) routines in GROMOS

Application of the PPPM algorithm in MD++ requires the availability of functions performing fast Fourier transforms (FFT). Such functions are collected in FFT libraries. MD++ is using the FFTW library by default and no other library can be selected. In GROMOS++ either the FFT routines from the Gnu Scientific Library (GSL) or the FFTW library are used depending on the individual program.

CHAPTER 22

Coarse Graining in GROMOS

For coarse graining, two different models are implemented in MD++: the MARTINI model[10] and the GROMOS model[11] (for details of both models see the corresponding literature). To use the MARTINI model ($NTCGRAN = 1$ or 2), the Lennard-Jones parameters have to be specified in the special topology block CGPARAMETERS.
The GROMOS model on the other hand ($NTCGRAN = 3$ or 4) makes use of the normal LJPARAMETERS block for the Lennard-Jones parameters and the BONDSTRETCHTYPE block for the bonds, although the range of particles which are coarse grained has to be specified in the topology block CGSOLUTE. As the coarse grained bonds in the GROMOS model are unconstrained, bonds involving coarse grained particles need to be given in the topology block CGBOND.

CHAPTER 23

Parallelisation in GROMOS

The most time-consuming parts of the GROMOS code are parallelised in order to run on shared- or distributed-memory architectures. The details of the parallelisation employed depend on the part of the GROMOS code.

23.1. Parallelisation in MD++

Computationally, the interaction calculation is by far the most expensive part of an MD or SD simulation or an EM, while the non-bonded interactions constitute the bulk of the effort. Again, MD++ is focused on achieving parallelisation without complicating the code. The non-bonded interaction is split up into `Nonbonded_Set`s, each containing its own storage space for a pairlist, energies, forces, and virials. In this way, the standard code is ready for shared and distributed memory parallelisation without any need for code duplication. If the system is using distributed memory, the (updated) positions and box parameters have to be copied from the master to all other processes before the next interaction calculation. While composing the pairlist in parallel, only a subset of atoms is considered per process, so that each processor creates its own partial and local pairlist. The interactions are calculated from this partial pairlist and stored in local arrays. This ensures synchronisation for shared memory machines and replicated data parallelisation for distributed memory systems. For the PPPM method, additional parallelisation steps are required. The mesh used for the long-range interaction evalulation is split and distributed over the individual compute nodes in a slice manner. Every compute node only evaulates the interactions for the atoms mapped on the slice of the grid. After the charge assignment the bordering cells of the slices are shared with the neighbouring nodes. The longe-range energy and virial are calculated on the individual slices of the mesh and summed in the final reduction step. Before the force calculation, bordering cells of the electrostatic potential are again shared with the neighbouring nodes in order to allow the evalulation of the force. All FFT calculations are carried out in a parallel way using the FFTW MPI library. The computation of the expensive $\tilde{A}_2$ term is also parallelised using MPI. After the partial interaction calculations have finished, the energies, forces, and virials of all non-bonded sets are summed up and stored in the `Configuration` of the master process. The SHAKE algorithm of the solvent molecules is parallelised using MPI. The old and new positions of the atoms are broadcasted to the compute nodes and every node applies the SHAKE algorithm on a subset of the solvent molecules. The resulting positions are then send back to the master node. MD++ can use `OpenMP`[12] for shared memory and `MPI`[13] for distributed memory parallelisation. For best performance the use of MPI is recommended. Reasonable parallelisation (using a small number of parallel processes) can be achieved with only a few lines of code (almost) completely separate from the non-bonded routines.

23.2. Parallelisation in GROMOS++

Some analysis programs in GROMOS++ carry out very intensive computation. These programs can be executed in parallel on shared memory architectures using OpenMP[12]. The programs including some form of parallelisation:

1. `filter`: Parallel filtering using a cutoff criterion.
2. `rdf`: Parallel radial distribution function evaluation.
3. `rot_rel`: Parallel rotational correlation function.
4. `utils::Energy`: Parallel interaction function evaluation in programs using the `utils::Energy` class like the program `ener`.

CHAPTER 24

Fast Solvent Interaction Function Evaluation

In a biomolecular simulation, using an explicit representation of the solvent, the solvent-solvent interaction function evaluation is the most time consuming computational step. Using roughly 75% of the computation time, this step is a good candidate for further optimizations. The following features of solvent molecules in the GROMOS software can be used to speed up the solvent-solvent interaction evaluation:

1. A solvent molecule is also a charge group, all atoms in one solvent molecule interact with all atoms in another solvent molecule.
2. A solvent molecule is rigid (i.e. does only have distance constraints and no harmonic bonds, bond angles or dihedral angles)
3. The atoms within a molecule do not interact with each other (no intra-molecular interaction).
4. Inter-molecular interaction cutoff truncation is applied. The first atom in the solvent topology is used to calculate the molecule-molecule distance.

Solvents which do violate these assumptions cannot be simulated using the GROMOS solvent loops but have to be technically treated as solute molecules. These assumptions, with the additional condition that a charge group is always gathered, i.e. no bonds between atoms of a charge group are broken by the periodic boundary condition, allow us to implement more efficient solvent-solvent loops.

24.1. Solvent innerloops in MD++

MD++ contains four additonal innerloops for solvent-solvent interactions. These loops can be controlled using the `INNERLOOP` block in the input file (see 4-95).

1. The first innerloop (`NTILM`=1) is a generic fast version. It takes advantage of the simplified representation of the solvent in order to speed up the interaction evalulation. The pairlist is evalulated in a molecule against molecule instead of an atom against atom way. Because the molecules are gathered, the nearest image calculation is only applied once. The nonbonded parameters are stored in a small atom against atom matrix which is efficiently cached. Application of this loop does not affect the accuracy and results in a speedup factor of about 1.5 for SPC water.
2. The second innerloop (`NTILM`=2) is a solvent specific version. The solvent has to be specified using the `NTILS` switch. In addition to the first method the molecule against molecule loop is manually unrolled and and the nonbonded parameters are hardcoded. The computation steps are aligned in a special way to help the compiler to use efficient (SSE) optimisation and automatic vectorisation. In the initalisation period, checks are made to ensure that the topological parameters are in agreement with the hardcoded ones. This loop does not affect the accuracy and results in a speedup factor of about 1.6 for SPC water. It is recommended to use this loop. Usage of the first method is only recommended if the solvent of interest is not implemented.
3. The third innerloop (`NTILM`=3) is a solvent specific version. It makes use of tables for the interaction evaluation. For every atom-type pair a hardcoded table holding the Lennard-Jones- and Coulomb-Reaction-Field-energies and forces are used. The energies and forces are tabulated for $\mathbf{r}_{ij}^2$ in order to avoid the expensive square-root and inverse computations. In the initalisation period, checks are made to ensure that the topological and input file parameters are in agreement with the hardcoded ones. Between the individual tabulated data points linear interpolation is used. The tables have to be generated and provided as a header file. The program `tabulate_spc` is used to generate the table for SPC water. By default a table size of 5000 for shortrange- and of 2000 for longrange-interactions is used. The resulting energies and forces are approximate and a speedup by a factor of 2 can be expected. Due to the $\mathbf{r}_{ij}^2$ nature of the table, it contains less grid points for small distances than for long distances. For this reason this method should be used with caution in high temperature or pressure simulations.

4. The fourth innerloop (`NTILM`=4) makes use of graphics processing units (GPUs) as acceleration hardware. The solvent-solvent iteractions are not run on the CPU but are executed on the CUDA enabled devices with the device number `NGPUS`. In a first step the positions and box parameters are transfered to the GPU. The pairlist for the solvent-solvent interaction is generated on the GPU. The pairlist evaluation on the GPU is executed in a parallel way[14]. The resulting energies, forces and virials are transfered to the main memory and summed in double precision. The computation is carried out in mixed precision resulting in numerical differences in comparison to the standard loops. Nevertheless, energetic, structural and dynamic quantities are not affected by this[14]. Depending on the GPUs, CPU and solvent used an overall speedup of a factor of 6 to 9 can be expected. In order to use this acceleration technique MD++ has to be compiled using special compiler options (see Sec. 8-3.1.3).

CHAPTER 25

Replica Exchange Simulation

In MD++, temperature and/or Hamiltonian replica exchange simulation can be performed using the program `repex_mpi`. Note that for this MD++ has to be compiled with `MPI`[13]. The replicas are controlled by the `REPLICA` block in the input file (see 4-103). In case of Hamiltonian replica exchange, the `PERTURBATION` block is additionally required (see 4-100). The number of replicas is given by `NRET*NRELAM`. Each replica with its specific combination of T and λ is assigned to a `MPI` process and remains with this process throughout the simulation. The master process is always replica 1. If a switching occurs, the configuration data is exchanged between two replicas.

Each replica writes into its own trajectory files and output file which are distinguished automatically by `_X` in the file name, where `X` is the number of replica (starting at 1). The output file for the replicas is given with the flag `@repout`. In contrast to normal MD++ simulations, only some information from the master about timings are printed to the standard output.

A continuation run is started by setting the parameter `CONT` in the `REPLICA` block to 1. Thus, a separate input coordinate file, distinguished by `_X` in the file name, where `X` is the number of replica, is read in for each replica. With the flag `@conf` simply the root of the file name has to be given and the program will search automatically for files with this root file name containing `_X`.

Information about the switching behaviour with probabilities and energies is printed to the replica data file specified with the flag `@repdat`. This data can be further analyzed using the GROMOS++ program `rep_ana`.

For optimal performance, it is advised to use the same number of `MPI`[13] processes as replicas. Additional speed-up can be obtained by compiling MD++ as `OpenMP`[12]/`MPI`[13] hybrid where each replica is parallelized further by `OpenMP`[12].

Bibliography

[1] ISO 14882:2003. *Programming languages – C++*. ISO, Geneva, Switzerland, 2003.

[2] W. Kabsch and C. Sander. Dictionary of protein secondary structure - Pattern-recognition of hydrigen-bonden and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.

[3] IUPAC-IUB commission on biochemical nomenclature. Abbreviations and symbols for the description of the conformation of polypeptide chains. Tentative rules (1969). *Biochemistry*, 9:3471–3479, 1970.

[4] T. Heinz and P.H. Hünenberger. A fast pairlist construction algorithm for molecular simulations under periodic boundary conditions. *J. Comput. Chem.*, 25:1474, 2004.

[5] I.G. Tironi and W.F. van Gunsteren. A molecular dynamics simulation study of chloroform. *Mol. Phys.*, 83:381–403, 1994.

[6] H. Liu, F. Müller-Plathe, and W.F. van Gunsteren. A Force Field for Liquid Dimethyl Sulfoxide and Physical Properties of Liquid Dimethyl Sulfoxide Calculated Using Molecular Dynamics Simulation. *J. Am. Chem. Soc.*, 117:4363–4366, 1995.

[7] A.D. McLachlan. Gene duplications in the structural evolution of chymotrypsin. *J. Mol. Biol.*, 128:44–77, 1979.

[8] W Kabsch. Solution for best rotation to relate 2 sets of vectors. *Acta Crystallogr.*, A32:922–923, 1976.

[9] G.H. Stout and L.H. Jensen. *X-ray structure determination*. Wiley, New York, USA, 1989.

[10] S.J. Marrink, A.H. de Vries, and A.E. Mark. Coarse Grained model for Semiquantitative Lipid Simulations. *J. Phys. Chem. B*, 108:750, 2004.

[11] S. Riniker and W.F. van Gunsteren. A simple, efficient polarisable coarse-grained water model for molecular dynamics simulations. *J. Chem. Phys.*, 134:084110, 2011.

[12] OpenMP.

[13] The Message Passing Interface.

[14] N. Schmid, M. Bötschi, and W.F. van Gunsteren. A GPU solvent-solvent interaction calculation accelerator for biomolecular simulations using the GROMOS software. *J. Comput. Chem.*, 31:1636–1643, 2010.

Index

Symbols

Symbol	Meaning
	Common names and abbreviations
GROMOS	The GROMOS software package
MD++	The MD++ simulation engine in C++
GROMOS++	The GROMOS++ analysis package in C++
GROMOS96	The GROMOS96 simulation package (1996)
3D	abbreviation for three dimensions
AA	Atomistic (All Atom) models
BD	Brownian Dynamics simulation
B&S – LEUS	Ball and stick local elevation umbrella sampling
CG	Coarse Grained models
CGEM	Conjugate gradient method for energy minimization
FRCG	Fletcher-Reeves conjugate gradient method for energy minimization
PRCG	Polak-Ribiére conjugate gradient method for energy minimization
COG	Center of geometry
COS	Charge On Spring approach
CP	Car Parrinello approach
DF	Distancefield
DOF	Degrees of freedom (abbreviation)
DPD	Diffusive Particle Dynamics simulation
`doxygen`	Documentation platform
EM	Energy minimisation
EDS	Enveloping distribution sampling
FBC	Fixed boundary conditions
HBC	Hyper-spherical boundary conditions
LE	Local elevation
LEUS	Local elevation umbrella sampling
LS	Lattice-sum method
MC	Monte Carlo sampling
MD	Molecular Dynamics simulation
NOE	Nuclear Overhauser Effect
PBC	Periodic boundary conditions
PPPM	Particle-particle–particle-mesh (P^3M) method
QM	Quantum Mechanical models
QMD	Quantum Molecular Dynamics simulation
RDF	Radial distribution function
RE	Replica Exchange
REMD	Replica Exchange Molecular Dynamics simulation
RF	Reaction-field method
RMSD	Root-mean-square difference
RMSF	Root-mean-square fluctuation
SD	Stochastic Dynamics simulation
SDEM	Steepest descent method for energy minimization
TI	Thermodynamic integration
US	Umbrella sampling

Symbol	Meaning
VBC	Vacuum boundary conditions
Physical constants	
h	Planck's constant [0.3990313 kJ mol^{-1} ps]
$\hbar$	Planck's constant divided by 2π [0.06350780 kJ mol^{-1} ps]
$N_{\rm Av}$	Avogadro's number [6.02214 $\times 10^{23}$]
k_B	Boltzmann's constant [1.380662 $\times 10^{-26}$ kJ K^{-1})]
R	Ideal gas constant ($N_{\rm Av} \times k_B$)
c	Speed of light [2.99792458 $\times 10^5$ nm ps^{-1}]
Degrees of freedom and system configuration	
$\mathcal{N}_d$	Number of degrees of freedom of a system
$\mathcal{N}_a$	Number of particles in a system of particles ($\mathcal{N}_d$ =3$\mathcal{N}_a$)
$\mathcal{N}_a^{solu}$	Number of particles the solute consists of
$\boldsymbol{q}$	$3\mathcal{N}_a$-dimensional generalized coordinate vector of a system of particles
$\boldsymbol{p_q}$	$3\mathcal{N}_a$-dimensional generalized momentum vector of a system of particles
$\boldsymbol{r}$	$3\mathcal{N}_a$-dimensional Cartesian coordinate vector of a system of particles
$\boldsymbol{p}$	$3\mathcal{N}_a$-dimensional Cartesian momentum vector of a system of particles
$\boldsymbol{f}$	$3\mathcal{N}_a$-dimensional Cartesian force vector of a system of particles
$\overline{\boldsymbol{f}}$	$3\mathcal{N}_a$-dimensional Cartesian mean force vector of a system of particles
$\boldsymbol{f}^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\mathbf{f}_i^{st}$	$3\mathcal{N}_a$-dimensional Cartesian stochastic force vector of a system of particles
$\boldsymbol{v}$	$3\mathcal{N}_a$-dimensional Cartesian velocity vector of a system of particles
$\mathbf{r}$	3-dimensional Cartesian coordinate vector of a particle
$\mathbf{p}$	3-dimensional Cartesian momentum vector of a particle
$\mathbf{f}$	3-dimensional Cartesian force vector of a particle
$\mathbf{v}$	3-dimensional Cartesian velocity vector of a particle
Ψ $[\Psi(\boldsymbol{r})]$	Wavefunction (position representation; configuration of a quantum-mechanical system of $\mathcal{N}_a$ particles)
$\{$ $\boldsymbol{r}$,$\boldsymbol{p}$ $\}$	Phase-space point (Cartesian coordinates; configuration of a classical system of $\mathcal{N}_a$ particles)
(Statistical) thermodynamics	
$\mathcal{F}$	Free energy
G	Gibbs free energy
H	Enthalpy
$\mathcal{U}$	Energy of a system
$\mathcal{S}$	Entropy of a system
$\mathcal{Z}$	Partition function
$\mathcal{T}$	Instantaneous temperature
$\mathcal{T}_o$	Reference temperature
$\mathcal{K}$	Instantaneous kinetic energy of a system
$\mathcal{K}_{tr}$	Instantaneous translational kinetic energy
$\mathcal{K}_{ir}$	Instantaneous internal+rotational kinetic energy
$\mathcal{U}$	Instantaneous total potential energy of a system
$\mathcal{W}$	Instantaneous virial of a system
$\mathcal{P}$	Instantaneous pressure of a system
$\mathcal{V}$	Instantaneous volume of a system
ρ_J	Number particle density of particles J
Miscellaneous	

Symbol	Meaning
t	Time
Δt	discrete time step
$\mathcal{N}_t$	Number of MD steps
P	Probability
m	Mass of a particle
M	Mass of the whole system
$\underline{\mathbf{m}}$	Diagonal mass matrix of a system of $\mathcal{N}_a$ particles
γ	Friction coefficient of a particle
$\underline{\gamma}$	Diagonal friction coefficient matrix of a system of $\mathcal{N}_a$ particles
T	Absolute temperature
β	prefactor: $1/k_BT$
τ_T	relaxation time for the coupling to a temperature bath
$\mathbf{s}$	Vector denoting the collection of all force-field parameters
λ	Coupling parameter Lambda for a lambda dependent Hamiltonian
$\mathcal{N}_\lambda$	Number of λ-values in a TI simulation
H	Heaviside function defined as $H(x) = 0 \; \forall \; x < 0$ and $H(x) = 1 \; \forall \; x > 0$
sign	Sign function: $\text{sign}(x) = 1 \; \forall \; x > 0$ and $\text{sign}(x) = -1 \; \forall \; x < 0$
i	imaginary number, $i^2 = -1$
δ_{ij}	general Kronecker delta
σ	Standard deviation
σ^2	Variance
$\mathcal{N}_{conf}$	Number of configurations in an ensemble
D	Diffusion constant
R_{gyr}	radius of gyration
η	the viscosity of a system
$g(r)$	radial distribution function
s	Smoothness parameter in EDS simulations
E^R	Energy offset parameter in EDS simulations
$\mathcal{N}^{(s)}$	Number of states in EDS simulations
Spatial boundary conditions	
$\underline{\mathcal{B}}$	3×3-matrix of the box-edge vectors (columns) in the reference Cartesian coordinate system (PBC)
$\hat{\mathbf{e}}$	Unit vector
$\mathbf{a}$	First edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{b}$	Second edge vector of a (triclinic) box (in the reference coordinate system)
$\mathbf{c}$	Third edge vector of a (triclinic) box (in the reference coordinate system)
a	length of first edge of a (triclinic) box
b	length of second edge of a (triclinic) box
c	length of third edge of a (triclinic) box
$\mathbf{T}$	Position vector of the reference corner of a triclinic box (components in the reference coordinate system and vector relative to the origin of this system)
$\underline{\mathbf{L}}$	Computational box matrix (columns defined by the components of edge vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ in the reference coordinate system)
$\underline{\mathbf{B}}$	Edge length matrix (diagonal, elements a, b and c)
α	First edge angle a triclinic box (between $\mathbf{b}$ and $\mathbf{c}$)
β	Second edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{c}$)
γ	Third edge angle a triclinic box (between $\mathbf{a}$ and $\mathbf{b}$)
ϕ	First Euler angle of a triclinic box

Symbol	Meaning
θ	Second Euler angle of a triclinic box
ψ	Third Euler angle of a triclinic box
$\check{\mathbf{r}}$	Oblique coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{r}}$	Oblique fractional coordinates of a real-space vector (with reference to the box-edge vectors)
$\check{\mathbf{k}}$	Oblique coordinates of a reciprocal-space vector
$\check{\mathbf{k}}$	Oblique fractional coordinates of a reciprocal-space vector
$\mathbf{l}$	Lattice vector (three-dimensional vector with integer components)
$\mathbf{k}$	Reciprocal-lattice vector ($\mathbf{k} = 2\pi\underline{\mathbf{L}}^{-1}\mathbf{l}$)
$\underline{\mathbf{S}}$	Transformation matrix
$\underline{\mathbf{R}}$	Transformation matrix
$\underline{\mathbf{T}}$	Transformation matrix
Representation of the interaction	
$\hat{\mathcal{H}}$	Hamiltonian operator describing the interaction for quantum-mechanical degrees of freedom
$\hat{\mathcal{K}}$	Kinetic energy operator (kinetic energy contribution to the quantum-mechanical Hamiltonian operator)
$\hat{\mathcal{V}}$	Potential energy operator (potential energy contribution to the quantum-mechanical Hamiltonian operator)
$\mathcal{H}$ $[\mathcal{H}(\boldsymbol{r},\boldsymbol{p})]$	Hamiltonian function describing the interaction for classical degrees of freedom
$\mathcal{K}$ $[\mathcal{K}(\boldsymbol{p})]$	Kinetic energy contribution to the classical Hamiltonian function
$\mathcal{V}$ $[\mathcal{V}(\boldsymbol{r})]$	Potential energy contribution to the classical Hamiltonian function
$\overline{\mathcal{V}}$ $[\overline{\mathcal{V}}(\boldsymbol{r})]$	Potential of mean force contribution to the classical Hamiltonian function
Physical interactions	
φ [Proper dihedral-angle term]	
$\mathcal{V}^{(phys)}$ $[\mathcal{V}^{(phys)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Physical potential energy contribution to $\mathcal{V}$
$\mathcal{V}^{(cov)}$ $[\mathcal{V}^{(cov)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Covalent potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(nbd)}$ $[\mathcal{V}^{(nbd)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Non-bonded potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(b)}$ $[\mathcal{V}^{(b)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond stretching potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\theta)}$ $[\mathcal{V}^{(\theta)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Bond-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\xi)}$ $[\mathcal{V}^{(\xi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Improper dihedral-angle bending potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(\varphi)}$ $[\mathcal{V}^{(\varphi)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Proper dihedral-angle torsion potential energy contribution to $\mathcal{V}^{(cov)}$
$\mathcal{V}^{(vdw)}$ $[\mathcal{V}^{(vdw)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Van der Waals potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(ele)}$ $[\mathcal{V}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic potential energy contribution to $\mathcal{V}^{(nbd)}$
$\mathcal{V}^{(LJCRF)}$	Sum of the non-bonded potentials $\mathcal{V}^{(vdw)}$ and $\mathcal{V}^{(ele)}$
Physical force-field terms	
$V^{(b)}$ $[V^{(b)}(b;k^{(b)},b^0)]$	Potential energy function associated with the stretching of a single covalent bond (quartic: $V^{(b,q)}$; harmonic: $V^{(b,h)}$; soft harmonic: $V^{(bs,h)}$)
$V_n^{(b)}$ $[V^{(b)}(b_n;k_n^{(b)},b_n^0)]$	Potential energy function associated with the stretching of the nth single covalent bond (quartic: $V_n^{(b,q)}$; harmonic: $V_n^{(b,h)}$; soft harmonic: $V_n^{(bs,h)}$)
$\mathbf{f}^{(b,q)}$	Force due to the bond stretching potential (quartic)
$\mathbf{f}^{(b,h)}$	Force due to the bond stretching potential (harmonic)
$\mathbf{f}^{(bs,h)}$	Force due to the bond stretching potential (soft harmonic)
$N^{(b)}$	Number of covalent bonds in the molecular system
$N^{(bs)}$	Number of soft covalent bonds in the molecular system
$M_n^{(b)}$	Bond type code associated with covalent bond term n

Symbol	Meaning
b_n $[b_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Length of covalent bond n in the considered configuration
b_n^0 $[b^0(M_n^{(b)}, \mathbf{s})]$	Reference length of covalent bond term n
$k_n^{(b,q)}$	Force constant of stretching for covalent bond term n (quartic potential)
$k_n^{(b,h)}$	Force constant of stretching for covalent bond term n (harmonic potential)
$V^{(\theta)}$ $[V^{(\theta)}(\theta; k^{(\theta)}, \theta^0)]$	Potential energy function associated with the bending of a single covalent bond angle (cosine-harmonic: $V^{(\theta,c)}$; soft cosine-harmonic: $V^{(\theta s,c)}$; angle-harmonic: $V^{(\theta,h)}$)
$V_n^{(\theta)}$ $[V_n^{(\theta)}(\theta_n; k_n^{(\theta)}, \theta_n^0)]$	Potential energy function associated with the bending of the nth covalent bond angle (cosine-harmonic: $V_n^{(\theta,c)}$; soft cosine-harmonic: $V_n^{(\theta s,c)}$; angle-harmonic: $V_n^{(\theta,h)}$)
$\mathbf{f}^{(\theta,c)}$	Force due to the bond angle potential (cosine-harmonic)
$\mathbf{f}^{(\theta s,c)}$	Force due to the bond angle potential (soft cosine-harmonic)
$\mathbf{f}^{(\theta,h)}$	Force due to the bond angle potential (angle-harmonic)
$N^{(\theta)}$	Number of covalent bond angles in the molecular system
$M_n^{(\theta)}$	Bond-angle type code associated with covalent bond-angle term n
θ_n $[\theta_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent bond angle n in the considered configuration
θ_n^0 $[\theta^0(M_n^{(\theta)}, \mathbf{s})]$	Reference angle of covalent bond-angle term n
$k_n^{(\theta,c)}$	Force constant of bending for covalent bond-angle term n (cosine-harmonic potential)
$k_n^{(\theta s,c)}$	Force constant of bending for covalent bond-angle term n (soft cosine-harmonic potential)
$k_n^{(\theta,h)}$	Force constant of bending for covalent bond-angle term n (angle-harmonic potential)
$V^{(\xi)}$ $[V^{(\xi)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$V^{(\xi s)}$ $[V^{(\xi s)}(\xi; k^{(\xi)}, \xi^0)]$	Potential energy function associated with the bending of a single covalent improper dihedral angle
$\mathbf{f}^{(\xi)}$	Force due to the improper dihedral-angle potential
$\mathbf{f}^{(\xi s)}$	Force due to the soft improper dihedral-angle potential
$N^{(\xi)}$	Number of covalent improper dihedral angles in the molecular system
$N^{(\xi s)}$	Number of covalent improper dihedral angles in the molecular system
$M_n^{(\xi)}$	Improper dihedral-angle type code associated with covalent improper dihedral-angle term n
ξ_n $[\xi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent improper dihedral angle n in the considered configuration
ξ_n^0 $[\xi^0(M_n^{(\xi)}, \mathbf{s})]$	Reference angle of covalent improper dihedral-angle term n
$k_n^{(\xi)}$	Force constant of bending for covalent improper dihedral-angle term n
$V^{(\varphi)}$ $[V^{(\varphi)}(\varphi; k^{(\varphi)}, \varphi^0)]$	Potential energy function associated with the torsion of a single covalent proper dihedral angle (symmetric potential: $V^{(\varphi,s)}$; generalized: $V^{(\varphi,g)}$)
$\mathbf{f}^{(\varphi,s)}$	Force due to the symmetric proper dihedral-angle potential
$\mathbf{f}^{(\varphi,g)}$	Force due to the generalized proper dihedral-angle potential
$N^{(\varphi)}$	Number of covalent proper dihedral angles in the molecular system
$M_n^{(\varphi)}$	Proper dihedral-angle type code associated with covalent proper dihedral-angle term n
φ_n $[\varphi_n(\boldsymbol{r}, \underline{\mathcal{B}})]$	Value of covalent proper dihedral angle n in the considered configuration
φ_n^0 $[\varphi^0(M_n^{(\varphi)}, \mathbf{s})]$	Reference angle (phase shift) of covalent proper dihedral-angle term n
$m_n^{(\varphi)}$ $[m_n^{(\varphi)}(M_n^{(\varphi)}, \mathbf{s})]$	Multiplicity of covalent proper dihedral-angle term n
$k_n^{(\varphi,s)}$	Force constant of torsion for covalent proper dihedral-angle term n (symmetric potential; $\varphi_n^0 = 0, \pi$; $m_n^{(\varphi)} \leq 6$)

Symbol	Meaning
$k_n^{(\varphi,g)}$	Force constant of torsion for covalent proper dihedral-angle term n (generalized potential; $\varphi_n^0 \in [0, 2\pi[$)
q	Partial charge of an atom or site
C_{12}	Van der Waals (Pauli) repulsion coefficient of an atom or site (Lennard-Jones function)
C_6	Van der Waals (London) dispersion coefficient of an atom or site (Lennard-Jones function)
C_{126}	Ratio of Van der Waals coefficients $\frac{C_{12}}{C_6}$ (Lennard-Jones function)
α_{LJ}	Lennard-Jones soft-core switching parameter
α_C	Coulomb soft-core switching parameter
$\mathcal{V}^{(ele,pws)}$ $[\mathcal{V}^{(ele,pws)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Pairwise potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,slf)}$ $[\mathcal{V}^{(ele,slf)}(\underline{\mathcal{B}};\mathbf{s})]$	Self potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathcal{V}^{(ele,srf)}$ $[\mathcal{V}^{(ele,srf)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Surface potential energy contribution to $\mathcal{V}^{(ele)}$
$\mathbf{f}^{(nbd)}$	Force due to the non-bonded forces
$\Psi_{ij}^{(ele)}$ $[\Psi_{ij}^{(ele)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Electrostatic influence function associated with the particle pair $i-j$
$\delta_{ij}^{(exc)}$ $[\delta_{ij}^{(exc)}(\mathbf{s})]$	Indicator of non-bonded exclusion for the particle pair $i-j$
$\Psi^{(ele,slf)}$ $[\Psi^{(ele,slf)}(\underline{\mathcal{B}})]$	Electrostatic self influence function
$\psi^{(RF)}$ $[\psi^{(RF)}(\mathrm{x})]$	Influence function at distance x of a particle in RF electrostatics
H $[H(\mathrm{x})]$	Heaviside step function (one if x is positive, zero otherwise)
R_C	Cutoff distance (truncation)
R_{cp}	Short-range cut-off
R_{cl}	Long-range cut-off
R^{cg}	radius of a charge group
N_{cg}	number of atoms belonging to a charge group
R_{RF}	Cutoff distance (onset of the RF continuum; usually set equal to R_C)
ϵ_{RF}	Relative dielectric permittivity of the RF continuum (usually set equal to that of the solvent)
κ_{RF}	Inverse Debye screening length of the RF continuum (usually set to zero)
C_{RF}	Constant characterizing the effect of the RF continuum
$\overline{\mathbf{R}}_{ij}$ $[\overline{\mathbf{R}}_{ij}(\boldsymbol{r})]$	Vector (FBC) or minimum-image vector (PBC) connecting the center of the CG containing particle j to the center of the CG containing particle i (norm $\overline{R}_{ij}$)
$\mathcal{V}^{(ele,pws,RF-CB)}$ $[\mathcal{V}^{(ele,pws,RF-CB)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Coulombic pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RF)}$ $[\mathcal{V}^{(ele,pws,RF-RF)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-dependent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\mathcal{V}^{(ele,pws,RF-RC)}$ $[\mathcal{V}^{(ele,pws,RF-RC)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Distance-independent pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (RF electrostatics)
$\Psi_{ij}^{(ele,LS-RS)}$ $[\Psi_{ij}^{(ele,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space component of electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\Psi_{ij}^{(ele,LS-KS)}$ $[\Psi_{ij}^{(ele,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space component of the electrostatic influence function $\Psi_{ij}^{(ele)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-RS)}$ $[\mathcal{V}^{(ele,pws,LS-RS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Real-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\mathcal{V}^{(ele,pws,LS-KS)}$ $[\mathcal{V}^{(ele,pws,LS-KS)}(\boldsymbol{r};\underline{\mathcal{B}};\mathbf{s})]$	Reciprocal-space pairwise potential energy contribution to $\mathcal{V}^{(ele,pws)}$ (LS electrostatics)
$\psi^{(LS)}$ $[\psi^{(LS)}(\mathbf{x})]$	Influence function at position $\mathbf{x}$ relative to a particle in LS electrostatics
a	Width of the charge-shaping function
γ $[\gamma(\mathrm{x})]$	Charge-shaping function

Symbol	Meaning
$\hat{\gamma}$ $[\hat{\gamma}$ (x)]	Fourier transformed charge-shaping function
$\mathbf{E}$	Electric field
$\boldsymbol{\mu}$	Dipole
$\boldsymbol{J}$	
α	Electronic polarisability
$\mathbf{P}$	Polarisation
ϵ	Dielectric permittivity
γ^{pol}	γ to calculate position of off site charge
k^{ho}	harmonic force constant in the COS model
ϕ	Electrostatic potential
Unphysical force-field terms	
$\mathcal{V}^{(spec)}$	Unphysical potential energy
$\mathcal{V}^{(res)}$	Restraint energy
$\mathcal{V}^{(pr)}$	Position restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(c)}$	Force due to the position constraints
$k^{(pr)}$	Force constant of an unphysical position-restraining term
$\mathcal{N}^{(pr)}$	number of positionally restrained atoms
l	Lagrange multiplier for position constraints
$\mathcal{V}^{(dr)}$	Distance restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(dir)}$	Force due to the atom-atom distance restraints
$k^{(dr)}$	Force constant of an unphysical distance-restraining term
$\mathbf{r}^0$	Equilibrium distance of distance restraint
$N^{(dir)}$	Number of atom-atom distance restraints
d_{CH}	carbon-hydrogen distance
d_{CC}	carbon-carbon distance
τ_{dr}	decay time for time-averaged distance restraining
$\mathcal{V}^{(tr)}$	Dihedral-angle restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(tr)}$	Force constant of an unphysical dihedral-angle restraining term
$\mathcal{N}^{(tr)}$	number of restrained dihedral angles
$\mathcal{V}^{(Jr)}$	^3J-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Jr)}$	Force constant of an unphysical ^3J-value restraining term
^3J	J-value or J-coupling constant
$^3J^0$	experimental J-value
J	general representation of a J-value
J^0	experimental J-value
ΔJ^0	width of flat-bottom for J-value restraining
a	a in Karplus relation
b	b in Karplus relation
c	c in Karplus relation
τ^s_{Jr}	period of scaling in periodically-scaled J-value restraining
Δt_ω	time period for which scaling is suspended in periodically-scaled J-value restraining
N_{le}	number of bins in J-value local elevation biasing
$w_{\zeta ni}$	weight of gaussian in J-value LE
$\mathcal{V}^{(Fxr)}$	$\mid F \mid$-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(exr)}$	ρ-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathcal{V}^{(sxr)}$	symmetry restraining potential energy contribution to $\mathcal{V}^{(phys)}$

Symbol	Meaning
k^{xr}	(harmonic) force constant for the crystallographic restraining
k^{sym}	harmonic force constant for the crystallographic symmetry restraining
F	Structure factor amplitude
ρ	Electron density
S	space group of a crystal
N_{sym}	Number of symmetry operations of a space group
$\mathbb{S}$	Symmetry operator $\mathbb{S} = \mathbf{R}\mathbf{r} + \mathbf{t}$
$\mathbf{R}$	Rotation matrix of a symmetry operator
$\mathbf{t}$	Translation vector of a symmetry operator
$\mathcal{V}^{(Sr)}$	S^2-restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$k^{(Sr)}$	Force constant of an unphysical S^2-value restraining term
S^2	S^2-order parameter
$S^{2,0}$	experimental S^2-value
S	general representation of a S^2-value
S^0	experimental S^2-value
$\mathcal{V}^{(df)}$	Distancefield restraining potential energy contribution to $\mathcal{V}^{(phys)}$
$\mathbf{f}^{(df)}$	Force due to the atom-atom distance restraints
$k^{(df)}$	Force constant of an unphysical distance-restraining term
l^0	Equilibrium distance of distance restraint
g_s	Distancefield grid distance
$\mathcal{V}^{(le)}$	Local elevation (LE) energy
$\mathcal{V}^{(bias)}$	bias energy
γ	LE basis function
$k^{(le)}$	LE force constant
$\mathbf{r}^{uc}$	unconstrained atomic positions
N_c	Number of constraints
N_{sh}	number of iterations of the SHAKE algorithm
d^0	constraint length
$\mathbf{f}^{uc}$	unconstrained atomic forces

The GROMOS Software for (Bio)Molecular Simulation

Volume 7: Tutorial with Examples

January 9, 2021

Contents

CHAPTER 1

Introduction

1.1. Simulation using GROMOS

GROMOS is a software package used for computer simulations of molecular systems like proteins, inorganic and organic chemical compounds, DNA, etc. In order to perform a molecular dynamics simulation using GROMOS one should:

1. create a topology of the system (`*.top` file)
2. have spacial coordinates of the system converted into GROMOS format (`*.cnf` file)
3. prepare an input file (`*.imd`) with parameters for the MD simulation

1.1.1. Units. Different sets of units are used in MD simulations. In general the use of *Standard International (SI)* units is recommended. In MD simulation of model systems, like Lennard-Jones liquids, it is often advantageous to work with dimensionless quantities (*reduced units*) and apply the appropriate scaling afterwards.

When choosing the SI system it is recommended to use the basic units shown in Tab. 1.1. From this basic set of units you can derive the units for all other quantities used in GROMOS. Units for some important quantities are given in Tab. 1.2. Apart from restrictions when storing or printing data in non-exponential format, the GROMOS programs are independent of the chosen units. The units are defined by the ones used for physical constants (Tab. 1.3) and atomic or molecular quantities in the (GROMOS) data files. Note that a number of *quantities* or interaction function *parameters* in GROMOS are either angles or *dependent on angle units* through their definition (see Chap. 6-6). For convenience of the user these quantities are kept in the *GROMOS data files* using *degrees as angle units*. However, when angles are used in calculations involving mathematical functions such as *sin*, *cos*, etc. they should be expressed in radians. Therefore, upon reading GROMOS data files the values of quantities and parameters that depend on angle units are converted from degrees to radians. So, in the *programs and subroutines* these quantities and parameters are stored using *radians as angle units.* For more information on units see Chap. 6-6.

Quantity		Unit	
length:	r :	nm	10^{-9} m
mass:	m:	u	atomic mass unit
			1/12 of the mass of a ^{12}C atom
			$10^{-3}/N_{\mathrm{Av}}$ kg
			$1.6605655 \cdot 10^{-27}$ kg
time:	t:	ps	10^{-12} s
temperature:	T:	K	
charge:	q:	e	elementary charge
			$1.6021892 \cdot 10^{-19}$ C

TABLE 1.1. Recommended units

1.1.2. File and software organisation. GROMOS knows different types of data and data files. Two types of information concerning a molecular system can be distinguished.

Quantity		Unit	
energy:	$\mathcal{U}$,$\mathcal{K}$:	kJ mol^{-1}	0.2390 kcal mol^{-1}
force:	$\boldsymbol{f}$:	kJ mol^{-1} nm^{-1}	
pressure:	$\mathcal{P}$:	kJ mol^{-1} nm^{-3}	$10^{30}/N_{\mathrm{Av}}$ Pa
			1.6605655 MPa
			16.6057 Bar
			16.3885 atm
velocity:	$\boldsymbol{v}$:	nm ps^{-1}	

TABLE 1.2. Derived units

Constant		Value
N_{Av}	Avogadro's number	$6.022045 \cdot 10^{23}$ mol^{-1}
R	gas constant	$8.31441 \cdot 10^{-3}$ kJ mol^{-1} K^{-1}
k_B	Boltzmann's constant	$R/N_{\mathrm{Av}} = 1.380662 \cdot 10^{-26}$ kJ K^{-1}

TABLE 1.3. Physical constants used

1. *Topological information*: data on the covalent structure, atomic masses, charges, van der Waals parameters, atom-atom distance restraints specification, ^3J-value restraints specification, local-elevation dihedral angles specification, etc.
2. *Configurational information*: atomic coordinates and atomic coordinate dependent or related quantities, such as velocities and forces, atom-atom distances, dihedral angles, ^3J-values, energies, size of the computational box, etc.

These two types of information are generally stored in separate files, since configurations change continuously during a simulation, whereas molecular topological data generally do not change. The naming convention for these files can be found in Chap. 4-13. Both types of files, *topological files* and *configurational files,* for a specific molecular system are related through the requirement that in both the sequence of the quantities is the same, e.g.

1. sequence of atoms
2. sequence of atom-atom distance restraints
3. sequence of dihedral angle restraints
4. sequence of ^3J-value restraints

This identity of sequence could be checked e.g. by comparing atom names occurring in topological files with those from the configurational files. However, in order to avoid dependence on naming conventions and to maintain maximum flexibility, this is not done in the GROMOS programs. When molecular information, such as residue numbers and names or atom sequence numbers or names, is present both in a topological file and in a configurational file of a molecular system, the program generally uses the data from the topological file and ignores the corresponding data on the configurational file.

In GROMOS files all the information is contained in *blocks*. For example, the molecular topology building block (`*.mtb`) file and the interaction function parameter (`*.ifp`) file, which are used for creating topologies, consist of blocks describing e.g. the physical constants used in GROMOS (`PHYSICALCONSTANTS` block), names of atom types recognized by GROMOS (`ATOMTYPENAME` block), force constants and lengths of various bonds in molecules (`BONDSTRETCHTYPECODE` block) and other atomic details and force-field parameters.

1.1.3. Summary of the exercise. All topological data mentioned above is used to construct topological building blocks. Many building blocks are chained together to form a final topology. If the system under consideration is charged, *counter ions* may be added, for which parameters are also included in the topological and force-field files (Sec. 2.1).

The second task is to obtain atomic cartesian coordinates of the system in GROMOS format. One can convert e.g. Protein Data Bank (PDB) files into GROMOS `*.cnf` files using programs available in the package (Sec. 2.2).

Further steps of an MD simulation involve minimizing the obtained spacial structure in vacuum, adding solvent molecules (Sec. 2.2.3) and counter ions (Sec. 2.2.4), minimizing the energy of the system and finally equlibration and thermalisation simulations (Sec. 2.3). After performing all these steps the actual MD simulation can be carried out. Input files used for minimisation, equilibration, thermalisation and molecular dynamics simulations also consist of blocks, which contain information about the simulation. For example, one has to specify the system by writing in the `SYSTEM` block how many solvent molecules there are. Simulation time, as well as integration time step, temperature and frequency of writing output files are all defined in the GROMOS `*.imd` files.

1.1.4. Calling the GROMOS programs. GROMOS programs are used with certain arguments, for example argument `@topo` would define the name of the topology file that should be used by the program called. One can write argument (`*.arg`) files which contain each argument and can be read by a program in the following way:

```
~> a_program @f file.arg
```

This saves a lot of time and makes it easier to repeat certain steps of your simulation, if needed.

1.2. Practical information

In this volume practical exercises are presented. You will perform a simulation of a penta-peptide in simple point charge (SPC) water with two chloride counter ions using the molecular dynamics software package GROMOS. Furthermore, you will analyse your results using GROMOS++ programs. Basic knowledge of the *UNIX* operating system is required in order to carry out the exercises. Further, you are going to use a graphical data plotting software called `xmgrace`. You can find an online *xmgrace* manual on http://linux.die.net/man/1/xmgrace.

After downloading the tutorial files from www.gromos.net to your `local` directory, type

```
~> tar zxf tutorial_files.tar.gz
```

Now you should have a new directory called `peptide` in your `local` directory and be ready to start doing the tutorial.

CHAPTER 2

A practical exercise

2.1. Building a topology

When a simulation study of a particular system or process is to be carried out, a number of choices have to be made with respect to the set-up of the simulation. The first task is to generate a molecular topology file containing the topological and force-field data concerning the molecular system under consideration. Specifying a complete molecular topology for a large molecule, however, is a tedious task. Therefore, in GROMOS a molecular topology is generated from molecular topology building blocks which carry the topological information of molecules like amino acid residues, nucleotides, etc., see Vol. 3. The molecular topology building blocks can be linked together into a complete molecular topology file using the GROMOS++ program `make_top`. Many molecular topology building blocks are available in the molecular topology building block files (`*.mtb`), which are standard data files that come together with the GROMOS package. In case the needed molecular topology building blocks are not part of the standard distribution, they must be constructed. Constructing a new topology building block may require estimation of additional force-field parameters, which have to be added to the interaction function parameter file (`*.ifp`). When generating a molecular topology for the system of interest one should also address the protonation state of the molecular groups according to the pH and of the solvent and counter ions that need to be included in the simulation box. In case of a molecular complex, e.g. a DNA-protein complex, the two separately generated molecular topologies for the protein and the DNA can be merged using the GROMOS++ program `com_top`.

2.1.1. Creating the topology for the penta-peptide. In this section you should build a molecular topology of a linear charged penta-peptide (Fig. 2.1) with water as a solvent, including two Cl^- counter ions.

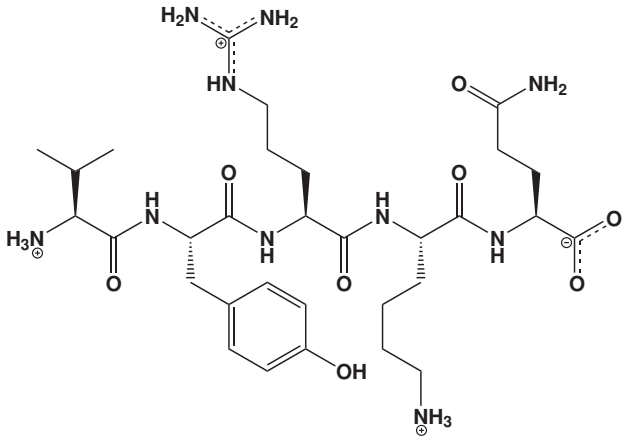


FIGURE 2.1. The topology of the penta-peptide

You need the following programs from the GROMOS simulation package:

GROMOS++: `make_top com_top check_top`

You need the following input files:

`54a7.mtb 54a7.ifp make_top_peptide.arg make_top_Cl.arg com_top_peptide_2Cl.arg`

The procedure will create the following output files:

`peptide_54a7.top Cl_54a7.top peptide_2Cl_54a7.top`

Go into the subdirectory `topo` of the directory `peptide` in your home.

```
~> cd ~/peptide/topo
```

You will build the molecular topology file of the linear charged penta-peptide Val-Tyr-Arg-Lys-Gln by using the GROMOS++ program `make_top`. The input file `make_top_peptide.arg` is already prepared and contains the following data: under the argument `@build` the molecular topology building block file is specified. The argument `@param` specifies the interaction function parameter file. Under the argument `@seq` the sequence of the building blocks for the amino acid residues, including the amino and carboxy terminus is specified (NH3+ VAL TYR ARG LYSH GLN COO-). Notice that both termini are charged.

Hint: If CA is a CH2 united atom (for natural amimo acids, this is only glycine), the N-terminal patches GH3+ or GH2 should be used instead of NH3+ or NH2. For beta-peptides, the N-terminal patches AH3+ or AH2 should be used to precede a residue where CB is a CH2 united atom, while BH3+ or BH2 should be used where CB is a CH1 united atom.

Hint: The ARG and LYSH building blocks correspond to the protonated amino acids. Always check carefully the name of the charged amino acids, to make sure you have the correctly protonated species. See Chap. 3-4.

The argument `@solv` specifies the solvent. The molecular topology file for the penta-peptide, `peptide_54a7.top`, with water as a solvent can then be generated as

```
~/peptide/topo> make_top @f make_top_peptide.arg > peptide_54a7.top
```

The next step is to build a molecular topology file for a chloride ion using the GROMOS++ program `make_top` and the input file `make_top_Cl.arg`:

```
~/peptide/topo> make_top @f make_top_Cl.arg > Cl_54a7.top
```

Now we will combine the generated molecular topology files (`peptide_54a7.top` and `Cl_54a7.top`) into the molecular topology file `peptide_2Cl_54a7.top` using the GROMOS++ program `com_top`. The input file `com_top_peptide_2Cl.arg` is already prepared and it contains the following data: under the argument `@topo` the molecular topology files that you would like to combine are specified. Since two chloride ions are needed to neutralize the charge of the penta-peptide, the molecular topology file of the chloride ion is specified as `2:Cl_54a7.top`. The arguments `@param` and `@solv` specify from which molecular topology file the parameters for the solute and solvent should be taken. Since we use the same molecular topology building block and interaction function files for both topologies, this is not important for us and both numbers are set to 1. To run `com_top` type:

```
~/peptide/topo> com_top @f com_top_peptide_2Cl.arg > peptide_2Cl_54a7.top
```

The file `peptide_2Cl_54a7.top` contains the complete molecular topology of the linear charged penta-peptide including $2Cl^-$ counter ions with water as a solvent. Have a look at it and check whether it makes sense! To be sure that your topology is correct, you should always check your topology using the `check_top` program.

Hint: Use `check_top` with the additional arguments `@build` and `@param` for a more careful check of your topology against the force field. (see the `check_top_peptide.arg` file for an example).

Warning: Be aware that although `check_top` can find many mistakes in your topology it is always important to carefully check your topology and building block files manually. This is especially important if you had to modify an existing or create a new building block.

2.2. Generating atom Cartesian coordinates for the solute, solvent and counter ions

Coordinates for biomolecules are often available from X-ray or NMR experiments and can be obtained in Protein Data Bank (PDB) format, which can be converted to GROMOS format using the GROMOS++ program `pdb2g96`. However, the conversion is not always straightforward since the naming and numbering of the atoms in the PDB format usually do not match the GROMOS format. Moreover, the coordinates for hydrogen atoms are not present in the PDB files (when the structure was determined using X-ray diffraction data) and have to be generated using the GROMOS++ program `gch`. When the structure is determined using NMR data, the PDB structure often contains more hydrogen atoms than are needed for GROMOS, as in the GROMOS force field only polar and aromatic hydrogens are explicitly represented. Aliphatic hydrogens are non-existing due to the use of so-called united atoms. The aliphatic hydrogen and carbon atoms are merged to form united atoms which have their own parameters. If no atomic coordinates for the solute are available from experimental data, the coordinates have to be generated using molecular modeling software. Often parts of the structure (e.g. flexible loops) are not resolved in the experiment and therefore not available in the PDB and have to be modeled as well. When a simulation of a solute in solution is to be carried out, a (periodic) box (be it rectangular, triclinic or truncated octahedral) is put around the solute and filled with solvent molecules up to the required density. The solvent coordinates can e.g. be generated using the GROMOS++ program `sim_box`. The generated box should be sufficiently large to allow the use of a reasonable non-bonded interaction cut-off radius. Putting the solute in a box of solvent using the `sim_box` program will result in several high-energy atom-atom contacts at the solute-solvent interface and at the box edges. In order to relax the generated configuration the solvent configuration should be energy minimized while positionally restraining the solute. Counter-ion atomic coordinates can then be generated using the GROMOS++ program `ion`, which can replace a number of solvent molecules by ions.

2.2.1. Generating atomic Cartesian coordinates for the linear charged penta-peptide. You need the following programs from the GROMOS simulation package:

GROMOS++: `pdb2g96 gch frameout`

You need the following input files:

`peptide.pdb pdb2g96_peptide.arg gch_peptide.arg frameout_peptide.arg`

The procedure will create the following output files:

`pdb2g96_peptide.cnf gch_peptide.cnf gch_peptide.pdb`

Go into the subdirectory `coord`. Open the file `peptide.pdb` and check if the atom names match the names in the molecular topology file `peptide_2Cl_54a7.top`.

In the pdb file `peptide.pdb` the coordinates for hydrogen atoms are not given and have to be generated. Convert the PDB file `peptide.pdb` into the GROMOS format using the GROMOS++ program `pdb2g96`. The hydrogen atoms will be added to the coordinate file according to the topological requirements. The input arguments of the `pdb2g96` program are self explanatory.

```
~/peptide/coord> pdb2g96 @f pdb2g96_peptide.arg > pdb2g96_peptide.cnf
```

Warning: When converting coordinate files from the Protein Data Bank to GROMOS format many difficulties may emerge. If you encounter problems using the `pdb2g96` program, have a look at Sec. 4-7.3. There you can find further documentation on the advanced usage of this program. Especially the use of a library that matches residue and atom names might be useful in many cases. `pdb2g96.lib` which you can find in the directory is an example of the PDB library file.

Have a look at the `pdb2g96_peptide.cnf` file. You will notice that the hydrogen atoms have been added to the coordinate file with the Cartesian coordinates being set to zero. In order to generate meaningful coordinates for the hydrogen atoms run the GROMOS++ program `gch`. It will generate the coordinates for hydrogen atoms by geometric means using the information from the molecular topology file. Therefore, the molecular topology file, `peptide_2Cl_54a7.top`, and the coordinate file, `pdb2g96_peptide.cnf`, have to be specified in the `gch` input file. The argument `@tol` sets the tolerance that is used for keeping the coordinates of hydrogens that are already present in the coordinate file.

To run gch type:

```
~/peptide/coord> gch @f gch_peptide.arg > gch_peptide.cnf
```

Using the GROMOS++ program frameout you can convert the coordinate file gch_peptide.cnf back to the PDB format and look at the structure using the molecular visualization program *VMD* (Visual Molecular Dynamics).

```
~/peptide/coord> frameout @f frameout_peptide.arg
~/peptide/coord> mv FRAME_00001.pdb gch_peptide.pdb
~/peptide/coord> vmd gch_peptide.pdb
```

2.2.2. Energy minimisation of the penta-peptide. You need the following programs from the GROMOS simulation package:

MD++: md

You need the following input files:

```
peptide_54a7.top gch_peptide.cnf em_peptide.imd em_peptide.run
```

The procedure will create the following output files:

```
em_peptide.omd peptide_min.cnf
```

Before putting the penta-peptide into a box of solvent, its configuration has to be relaxed by energy minimisation. Go into the subdirectory called min.

The MD++ input file em_peptide.imd contains the following blocks:

```
TITLE
steepest descent energy minimisation of the peptide in vacuum.
END
```

In the TITLE block you specify what is done with following input file.

```
ENERGYMIN
#     NTEM    NCYC    DELE    DX0     DXM   NMIN   FLIM
         1       0     0.1   0.01    0.05   2000    0.0
END
```

The existence of the ENERGYMIN block means that the MD++ program will perform an energy minimisation (EM) run. The NTEM switch indicates which minimisation algorithm to be used. With NTEM = 1 we indicate that the steepest-descent algorithm (Sec. 2-11.2) is used. NCYC gives the number of steps before resetting of conjugate-gradient search direction in case we would use the conjugate gradient method (NTEM = 2). Using DELE the energy threshold (the difference in energy between two energy minimisation steps) for stopping the minimisation process (convergence) is specified. The initial step size and maximum step size is given in DX0 and DXM, respectively. Using FLIM the absolute value of the forces can be limited to a maximum value before the algorithm is applied (see also 4-93).

```
SYSTEM
#      NPM      NSM
         1        0
END
```

In the SYSTEM block you specify the number of solutes (NPM) and solvent (NSM) molecules. You only have one solute NPM = 1 and no solvent molecules NSM = 0 because you still did not add any solvent molecules to the configuration file and the peptide is still in vacuum. Otherwise you would have to tell MD++ how many solvent molecules you are using.

Hint: Note that the solute and solvent topologies contained in the topology file are multiplied by these factors to match the sequence of atoms in the configuration file. If you want to simulate a peptide with two counter ions you still have to specify NPM = 1. The peptide and the counter ions form *one solute*. As NPM > 1 is not supported by MD++ you should use the GROMOS++ program com_top to multiply the topology of a solute if required.

```
INITIALISE
#  NTIVEL  NTISHK  NTINHT  NTINHB  NTISHI  NTIRTC  NTICOM  NTISTI     IG    TEMPI
       0       0       0       0       1       0       0       0  210185  300.0
END
```

This block will be explained in more detail in section Sec. 2.3.

```
STEP
#   NSTLIM        T        DT
     2000      0.0    0.002
END
```

In the `STEP` block you specify the maximum number of steps (`NSTLIM`) for the energy minimisation. GROMOS will stop as soon as the energy changes less than `DELE` or this step number is reached. For an MD simulation `T` is the initial time and `DT` is the integration time step used.

```
BOUNDCOND
#      NTB    NDFMIN
         0         1
END
```

In the `BOUNDCOND` block you specify which periodic boundary conditions (PBC) you are going to use in the EM procedure. `NTB` = 0 defines a vacuum simulation: PBC are not applied. To indicate the truncated octahedron (t) PBC, `NTB` is set to -1, for rectangular (r) PBC `NTB` is 1, and for the triclinic (c) PBC `NTB` is 2. `NDFMIN` defines the number of degrees of freedom subtracted from the total number of degrees of freedom for the calculation of the temperature.

```
PRINTOUT
#     NTPR           NTPP
        10              0
END
```

With the `PRINTOUT` block you can specify how often (every `NTPR`th step) you are printing out the energies to the output file.

```
CONSTRAINT
#      NTC    NTCP    NTCP0(1)    NTCS    NTCS0(1)
         3       1      0.0001       1      0.0001
END
```

Bonds vibrate at high frequencies ($h\nu \gg k_B T$). Therefore, these vibrations are of quantum-mechanical nature. So constraining the bond lengths is a better approximation than treating them as classical harmonic oscillators. Constraining all bond lengths of the solute and solvent (`NTC=3`) allows the use of a rather large time step of 2 fs. In this example the constraints are imposed by the `SHAKE` algorithm for both solute (`NTCP=1`) and solvent (`NTCS=1`) with a tolerance of 0.0001. See 4-90 for more information.

```
 FORCE
# NTF(1..6): 0,1 determines terms used in force calculation
#            0: do not include terms
#            1: include terms
# NEGR: ABS(NEGR): number of energy groups
#            > 0: use energy groups
#            < 0: use energy and force groups
# NRE(1..NEGR): >= 1.0 last atom in each energy group
# NTF(1) NTF(2) NTF(3) NTF(4) NTF(5)    NTF(6)
# bonds    angles   improper dihedral electrostatic vdW
  0        1        1        1        1            1
# NEGR   NRE(1)   NRE(2)   ...     NRE(NEGR)
     1      71
END
```

In the `FORCE` block you tell MD++ which terms it should use for the energy and force calculation. For bond angles, improper dihedrals, torsional dihedrals and the non-bonded interactions the standard terms of the GROMOS force field are switched on (1). Because we are using bond-length constraints and the SHAKE algorithm, we have to switch off (0) the bond-stretching terms for the bonds involving hydrogen atoms and not involving hydrogen atoms..

In the last line of this block, the energy groups are defined. In general, we define one or more energy groups for every molecule, and one comprising all the solvent molecules. The first integer is the number of energy groups we want to use (in the present case we only have one energy group). The following numbers are the atom sequence numbers of the last atom of each energy group. By defining these energy groups we

tell MD++ to sum up the energies between the atoms within these groups and calculate the inter-group energies, which can be very useful.

Warning: Think very carefully about the definition of energy groups before running the simulation. Energies of energy groups can not be calculated from the trajectories in an efficient way. So, changing an energy-group definition will result in rerunning the simulation.

```
PAIRLIST
#      algorithm: standard (0) (gromos96 like pairlist)
#                 grid (1) (XX grid pairlist)
#      SIZE:     grid cell size (or auto = 0.5 * RCUTP)
#      TYPE:     chargegoup (0)(chargegroup based cutoff)
#                atomic (1)(atom based cutoff)
#
#   ALGORITHM  NSNB    RCUTP    RCUTL  SIZE  TYPE
            0     5      0.8      1.4   0.4     0
END
```

In the `PAIRLIST` block you specify which algorithm you will use for the pairlist generation. The cut-off used in the short-range pairlist construction is given by `RCUTP` and for GROMOS it is usually 0.8 nm. The cut-off used in the long-range interactions is given by `RCUTL` and for GROMOS it is usually 1.4 nm. The pairlist is generated every 5th (`NSNB`) step. `TYPE` specifies the type of the cut-off, whether it is based on the distance between charge-groups (0) or on the distance between atoms (1).

```
NONBONDED
# NLRELE   APPAK     RCRF    EPSRF    NSLFEXCL
       1     0.0      1.4        1         1
# NSHAPE  ASHAPE   NA2CLC  TOLA2  EPSLS
     -1      1.4       2  1e-10      0
# NKX   NKY   NKZ   KCUT
   10    10    10    100
# NGX  NGY   NGZ  NASORD NFDORD  NALIAS NSPORD
   32    32    32       3       2       3       4
# NQEVAL  FACCUR  NRDGRD  NWRGRD  NLRLJ   SLVDNS
  100000     1.6       0       0       0     33.3
END
```

In the `NONBONDED` block you specify using `NLRELE` which method for the evaluation of long-range electrostatic interactions is used. Since you will use the reaction-field method, the value of `NLRELE` should be equal to 1. The long-range electrostatic interactions are truncated beyond a certain cutoff (`RCUTL` in the `PAIRLIST` block). Beyond the reaction-field cut-off radius (`RCRF`) the electrostatic interactions are replaced by a static reaction field with a dielectric permittivity of `EPSRF`. `RCRF` and `RCUTL` should be identical. Because we are doing the energy minimisation in vacuo `EPSRF` is set to 1. With `NSLFEXCL` equal to 1, you include the contributions of excluded atoms to the electrostatic energy. The ionic strength of the continuum is set to 0 (`APPAK`). All other switches are not used for the reaction-field method. See 4-98 for more information.

In order to run the MD++ program, a shell script needs to be prepared. Open the shell script `em_peptide.run` and adapt the paths and the names of the files according to your system. The energy minimisation of the solute in vacuo is very fast and you can run it interactively by typing:

```
~/peptide/min> ./em_peptide.run
```

Once the energy minimisation is finished, the file with the minimized coordinates, `peptide_min.cnf`, and the general output file, `em_peptide.omd`, that reports the progress of the minimisation, will be written out. Have a look at both files and check if the minimisation has finished successfully. Using the GROMOS++ program `frameout` you can again convert the coordinate file `peptide_min.cnf` into PDB format and view the new configuration using the *VMD* program.

2.2.3. Solvating the penta-peptide in a water box. You need the following programs from the GROMOS simulation package:

GROMOS++:	`sim_box`
MD++:	`md`

You need the following input files:

`peptide_54a7.top spc.cnf em_solvent.imd sim_box_peptide.arg sim_box_peptide.rpr sim_box_peptide.por em_solvent.run`

The procedure will create the following output files:

```
sim_box_peptide.cnf em_solvent.omd peptide_h2o.cnf
```

Now you can put the energy minimized penta-peptide in a box of solvent using the GROMOS++ program `sim_box` which can solvate the solute in a pre-equilibrated box of solvent molecules. Go to the subdirectory `box`. In the input file for the program `sim_box` you have to specify the following input arguments: the molecular topology file under the argument `@topo`, the resulting box shape under the argument `@pbc` (r-rectangular, t-truncated octahedron, c-triclinic), the coordinate file of the solute under the argument `@pos`, the coordinate file of the pre-equilibrated box of solvent molecules under the argument `@solvent`, minimum solute-to-wall distance under the argument `@minwall`, and the minimum solute-to-solvent distance under the argument `@thresh`. If you are using a rectangular box (`@pbc` r) it is recommended to use an additional argument, `@rotate`. With this additional argument the solute can be rotated (before solvating) such that the largest distance between any two solute atoms is directed along the z-axis, and the largest atom-atom distance in the xy-plane lies in the y-direction. An input file `sim_box_peptide.arg` is already prepared. To put the solvent box around the penta-peptide type:

```
~/peptide/box> sim_box @f sim_box_peptide.arg > sim_box_peptide.cnf
```

In order to relax the unfavorable atom-atom contacts between the solute and the solvent, energy minimisation of the solvent should be performed while keeping the solute positionally restrained (i.e. connecting the atom to a reference position by a spring). In order to do that two additional files, in which the positionally restrained atoms and the reference coordinates are specified, have to be generated from the coordinate file `sim_box_peptide.cnf`. Copy the coordinate file `sim_box_peptide.cnf` to `sim_box_peptide.por` and to `sim_box_peptide.rpr`. Open the new file `sim_box_peptide.por` with a text editor, write in the `TITLE` block the text "solute atoms to be positionally restrained", and delete all the coordinates of the solvent such that only the coordinates of the solute atoms are left. Then change the keyword `POSITION` at the beginning of the atom coordinate block of `sim_box_peptide.por` to `POSRESSPEC`. You now have a file containing a `POSRESSPEC` block which only contains the atoms of the peptide. With these changes the position restraints specification file `sim_box_peptide.por` should look like this:

```
TITLE
solute atoms to be positionally restrained
END
POSRESSPEC
    1 VAL  H1        1   0.508983407 -0.108220645 -0.056160171
    1 VAL  H2        2   0.478246738 -0.131824700  0.103013982
...
    5 GLN  O1       70  -0.389334812 -0.481578304 -0.248788696
    5 GLN  O2       71  -0.408991389 -0.269160831 -0.187628610
END
```

The preparation of the reference position file is even simpler as it is very similar to the coordinate file and only the `TITLE` block has to be changed and one block has to be renamed. In order to prepare the reference position file, open the new file `sim_box_peptide.rpr` in a text editor, write in the `TITLE` block the text "reference positions for restraining solute atoms", and change the block name `POSITION` to `REFPOSITION`.

Warning: When reading in a coordinate file with a `POSITION` block the first 24 characters (atom specification) are ignored. However, in a position restraints file each line of the `POSRESSPEC` block must have seven columns as the fourth to seventh column are read in by the program.

The MD++ input files for minimisation of the solvent around the penta-peptide, `em_solvent.imd`, and the script to run the minimisation, `em_solvent.run`, are already prepared. Nevertheless, open the `em_solvent.imd` and compare it with `em_peptide.imd`. You will notice that `em_solvent.imd` contains one input block more, the `POSITIONRES` block.

```
POSITIONRES
#    values for NTPOR
#    0: no position re(con)straining
#    1: use CPOR
#    2: use CPOR/ ATOMIC B-FACTORS
#    3: position constraining
#     NTPOR    NTPORB   NTPORS   CPOR
      1           1       0     25000
END
```

Using this block you are specifying that you want to restrain the positions of your solute molecule. The restraining is achieved by a harmonic force term with a force constant `CPOR`. The `NTPORB` indicates that the reference positions are read from a separate file, which is in our case `sim_box_peptide.rpr`. `NTPORS` equal to 0 prevents the program from changing the reference positions upon pressure scaling.

Hint: Notice and understand the differences in SYSTEM, BOUNDCOND and FORCE block between `em_peptide.imd` and `em_solvent.imd`.

Have a look at `em_solvent.run` and put in the correct paths to your files. Run the energy minimisation of the solvent interactively by typing

```
~/peptide/box> ./em_solvent.run
```

This will take a few minutes. Once the minimisation is finished, the new coordinate file, `peptide_h2o.cnf`, and the general output file `em_solvent.omd` will be written out.

2.2.4. Adding counter ions to the simulation box. You need the following programs from the GROMOS simulation package:

GROMOS++: `ion`

You need the following input files:

`peptide_54a7.top peptide_h2o.cnf ion_peptide.arg`

The procedure will create the following output files:

`peptide_2Cl_h2o.cnf`

In the next step, two chloride ions should be added to the box. Go to the subdirectory `ion`. The two chloride ions are added to the simulation box by using the GROMOS++ program `ion` such that they replace the water molecules which have the highest electrostatic potential. You can run `ion` by typing

```
~/peptide/ion> ion @f ion_peptide.arg > peptide_2Cl_h2o.cnf
```

Convert the file `peptide_2Cl_h2o.cnf` into the PDB format using the GROMOS++ program `frameout` and check where the chloride ions have been placed. In Fig. 2.2 you can see how it should look like.

Hint: Have a look at the file `frameout_1.arg` in `~/peptide/ana/frameout/`. Two arguments have been added:

```
@pbc   r
@include  ALL
```

The argument `@pbc` specifies which periodic boundary condition was used (in our case rectangular - `r`) and that we want to gather the frames. The argument `include` specifies whether `frameout` should convert only the solute, which is the default option, only the solvent (`@include SOLVENT`), or all the atoms (`@include ALL`).

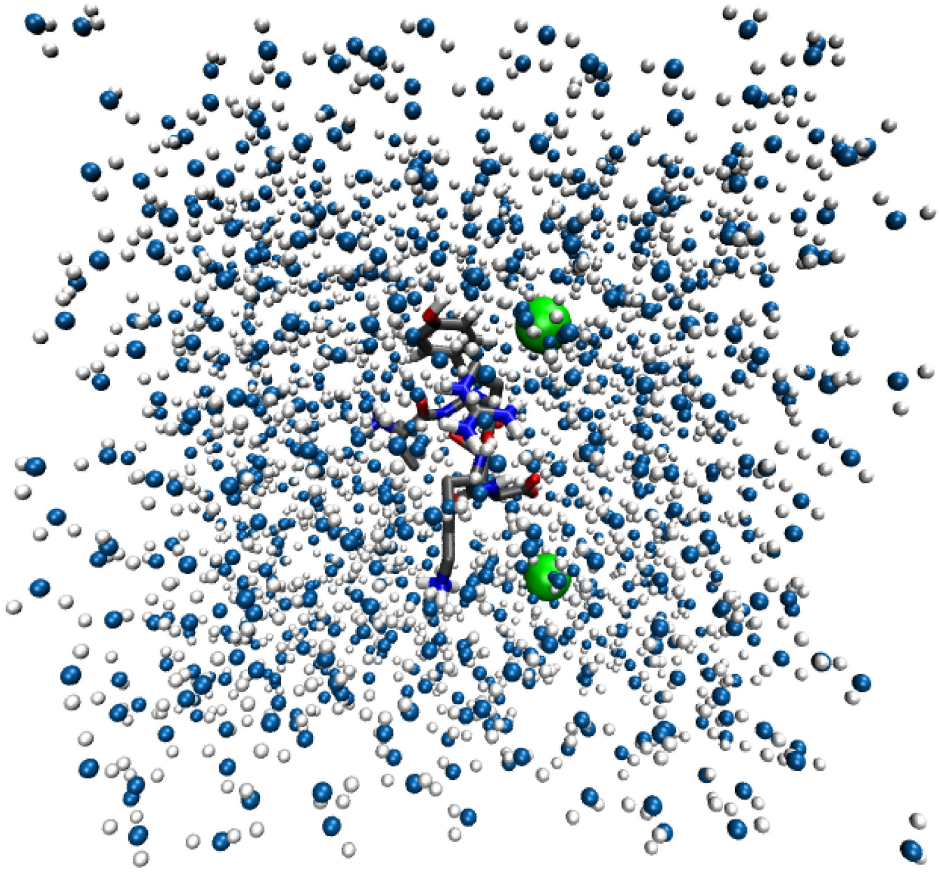


FIGURE 2.2. The penta-peptide and the two counter ions in a box of water.

2.3. Set-up and production simulation of the penta-peptide

2.3.1. Thermalisation and equilibration. You need the following programs from the GROMOS simulation package:

GROMOS++:	`mk_script ene_ana`
MD++:	`md`

You need the following input files:

`eq_mk_script.arg equilibration.imd equilibration.jobs mk_script.lib ene_ana.arg ene_ana.md++.lib`

The procedure will create the following output files:

`eq_peptide_*.run eq_peptide_*.imd eq_peptide_*.omd eq_peptide_*.cnf eq_peptide_*.tre.gz eq_peptide_*.trc.gz totkin.dat`

In the previous steps you have generated a topology and initial coordinates of your system. At this point, you have to generate initial velocities. In the process of thermalisation and equilibration, initial velocities are sampled from a Maxwell-Boltzmann distribution at a low temperature and the system is slowly heated up to the final production simulation temperature. The atoms of the solute are positionally restrained and these restraints are loosened while heating up. With the help of these restraints you make sure that the random initial velocities do not disrupt the initial conformation too much.

You already know a couple of things about job scripts. Because the set-up of a job script can be a labor-intensive undertaking, there is a little but powerful helper called `mk_script`. This GROMOS++ program is able to automatically generate a job script from a given input file and a series of arguments.

Before we have a detailed look at `mk_script`, let's see which MD++ input file we need for the thermalisation and equilibration period. Go to the subdirectory `eq`. Open the `equilibration.imd` file. This file contains a series of input blocks some of which you have already seen at the energy minimisation step. Here only new or changed input blocks are explained.

```
INITIALISE
#   NTIVEL  NTISHK  NTINHT   NTINHB   NTISHI NTIRTC    NTICOM  NTISTI     IG    TEMPI
        1       0       0        0        1      0         1       0  210185    300.0
END
```

In the INITIALISE block the NTIVEL tells GROMOS whether it should generate the initial velocities or read them from the configuration file. NTISHK is used to restore bond length constraints (SHAKE). NTINHT and NTINHB are only used for Nose-Hoover thermo- and barostats and can be ignored in our case. Every time an atom is leaving the periodic box and entering it from the opposite site this incident is recorded in the so-called lattice shift vectors. Using NTISHI we want to make sure that these vectors are initialised to zero. As you don't want to use roto-translational constraints NTIRTC can be ignored. NTICOM is used for initial removal of centre of mass motion. NTISTI is used to reset the stochastic integrals used in stochastic dynamics (SD) simulations. IG is the random number generator seed and TEMPI the initial temperature used to generate the Maxwell-Boltzmann distribution for generation of initial velocities. See also 4-94 for more information.

In the SYSTEM block you need to replace NSM with the number of solvent molecules in your system (you will find it in peptide_2Cl_h2o.cnf file).

```
STEP
#   NSTLIM        T       DT
    10000      0.0    0.002
END
```

In the STEP block you specify how many steps you want to simulate (NSTLIM), at what time your simulation starts (T) and how big the integration time step (DT) is. In this case you want to start at time 0 and you want to carry out a 20 ps simulation, because the time unit happens to be ps.

```
BOUNDCOND
#     NTB   NDFMIN
        1        3
END
```

As previously described, with the BOUNDCOND block you specify which PBC you will use. With NTB=1 you specify rectangular PBC.

```
MULTIBATH
# ALGORITHM:
#     weak-coupling(0):    use weak-coupling scheme
#     nose-hoover(1):      use Nose Hoover scheme
#     nose-hoover-chains(2): use Nose Hoover chains scheme
# NUM: number of chains in Nose Hoover chains scheme
#     !! only specify NUM when needed !!
# NBATHS: number of temperature baths to couple to
#         ALGORITHM
                0
#  NBATHS
        2
# TEMP0(1 ... NBATHS) TAU(1 ... NBATHS)
       60      0.1   60     0.1
#   DOFSET: number of distinguishable sets of d.o.f.
        2
# LAST(1 ... DOFSET) COMBATH(1 ... DOFSET) IRBATH(1 ... DOFSET)
      73        1        1     LSTATM       2        2
END
```

In our case we want to run the simulation at constant temperature. For this purpose, we have to add the MULTIBATH input block (see 4-96). First we specify which algorithm we will use. In this case we will use the weak-coupling scheme (ALGORITHM=0). How many temperature baths we want to couple to the system is specified by NBATHS. You can specify the temperature using the TEMP0 parameter. TAUT is the coupling time used in the weak-coupling method for this bath. DOFSET specifies the number of distiguishable sets of degrees of freedom. LAST is pointing to the last atom for the set of degrees of freedom; thus, you put the number of last atom of your system instead of LSTATM. COMBATH is the temperature bath to which we want to couple the centre of mass motion of this set of degrees of freedom IRBATH is the temperature bath to which the internal and rotational degrees of freedom of this set of degrees of freedom are coupled. The temperatures in this block are modified by mk_script.

```
COMTRANSROT
#   NSCM
    1000
```

```
END
```

This block is needed to remove the centre of mass motion (translation and rotation). Without this block it can happen that all the kinetic energy is converted to centre of mass translation (flying ice cube problem). With NSCM we specify how often the center-of-mass (COM) motion is removed. If NSCM is < 0 translation and rotation motion are removed every $|$NSCM$|$th step. If NSCM is > 0 only translation motion is removed every NSCMth step.

```
COVALENTFORM
# NTBBH: 0,1 controls bond-stretching potential energy term
#       0: quartic form (default)
#       1: harmonic form
# NTBAH: 0,1 controls bond-angle bending potential energy term
#       0: cosine-harmonic (default)
#       1: harmonic
# NTBDN: 0,1 controls torsional dihedral angle potential energy term
#       0: arbitrary phase shifts (default)
#       1: phase shifts limited to 0 and 180 degrees.
#   NTBBH    NTBAH    NTBDN
       0        0        0
END
```

This block is needed to define which functional form we will use for bond-stretching (NTBBH), bond-angle bending (NTBAH) and for torsional dihedral (NTBDN). We just use the default options for all functional forms.

```
WRITETRAJ
# NTWSE = configuration selection parameter
# =0: write normal trajectory
# >0: chose mininimum energy for writing configurations
#     NTWX     NTWSE      NTWV      NTWF     NTWE      NTWG   NTWB
       100         0         0         0      100         0      0
END
```

MD++ produces a massive amount of data and it is impossible to store all the data it produces. The WRITETRAJ block meets this demand: Here you specify how often the coordinate trajectory (NTWX), the velocity trajectory (NTWV), the force trajectory (NTWF), the energy trajectory (NTWE), the free energy trajectory (NTWG) and the block averaged energy trajectory (NTWB) are written out. In the present case, we are only interested in the coordinates (NTWX) and energies (NTWE) and we write them every 100^{th} step. The second switch (NTWSE) defines selection criterion for trajectories: if NTWSE $= 0$ the normal coordinate trajectory will be written, or if NTWSE > 0 a minimum energy trajectory will be written.

Warning: It makes no sense to write out configurations too often. First, it needs a lot of disk space. Second, the data is highly correlated and so no additional information is gained from it.

```
PRINTOUT
#NTPR: print out energies, etc. every NTPR steps
#NTPP: =1 perform dihedral angle transition monitoring
#     NTPR        NTPP
       100           0
END
```

This block is very similar to the WRITETRAJ block but the information about the energies (NTPR) is printed to the output file. By giving NTPP, dihedral angle transitions are written to the special trajectory.

In the FORCE block you need to replace the LSTATM with the number of last atom of your system.

```
PAIRLIST
#      ALGORITHM: standard(0) (gromos96 like pairlist)
#                 grid(1) (MD++ grid pairlist)
#      SIZE:      grid cell size (or auto = 0.5 * RCUTP)
#      TYPE:      chargegroup(0) (chargegroup based cutoff)
#                     atomic(1) (atom based cutoff)
#
#      ALGORITHM      NSNB    RCUTP   RCUTL   SIZE    TYPE
               1         5      0.8     1.4    0.4    0
END
```

MD++ knows different algorithms for the generation of the pairlist, a list containing the atoms interacting with each other. Here, we use a grid based pairlist generation: the space is discretized into grid cells and only the neighboring cells are searched for interacting partners. The use of this algorithm results in a

significant speed increase because the scaling of the algorithm is changed from $\mathcal{O}\left({\mathcal{N}_a}^2\right)$ to $\mathcal{O}\left(\mathcal{N}_a\right)$. The pairlist is generated every 5th (NSNB) step. RCUTP and RCUTL are the cutoffs for the pairlist construction for the short-range and the long-range interactions.

```
POSITIONRES
#     values for NTR
#     0: no position re(con)straining
#     1: use CHO
#     2: use CHO/ ATOMIC B-FACTORS
#     3: position constraining
#      NTPOR    NTPORB    NTPORS    CPOR
       1            1         0   25000
END
```

Finally, we want to restrain the position of our solute. The restraining is achieved by a harmonic special force term with a force constant of CPOR. This force constant is also modified by mk_script.

Now you should understand the main blocks of the MD++ input files.

Hint: You can find further information on the GROMOS input file in Chap. 4-8.

Now it is time to have a look at mk_script. Open the input file eq_mk_script.arg. Here choose a system name @sys, describing your simulation. @bin points to the MD++ executable md and @dir points to the directory where the simulation files are. Using the @files argument you specify all the files needed by your simulation (topology, MD++ input file, initial coordinate file, position restraints and reference position file). @template respectively mk_script.lib is a configuration file for mk_script. Therein you can adapt mk_script to your local system or cluster. Because we are using MD++ you have to give the @version md++ argument. Finally, you tell mk_script what to do, using a joblist file that you specify with the @joblist argument.

Hint: All file paths that you give in a mk_script input file must be relative to the path @dir.

The joblist is already prepared for you. Therein you can change parameters in the MD++ input file for a certain job. Have a look into the file:

```
~/peptide/eq> cat equilibration.jobs
TITLE
General startup protocol.
heating while loosening the position restraints.
END
JOBSCRIPTS
job_id NTIVEL TEMPI TEMP0[1] TEMP0[2] COUPLE NTPOR CPOR subdir run_after
1      1      60.0     60.0    60.0      1     1  2.5E4   .      0
2      0      0.0     120.0   120.0      1     1  2.5E3   .      1
3      0      0.0     180.0   180.0      1     1  2.5E2   .      2
4      0      0.0     240.0   240.0      1     1  2.5E1   .      3
5      0      0.0     300.0   300.0      1     0  0.000   .      4
END
```

All the jobs have a certain ID which you can find in the first column. The following columns specify the parameters of the MD++ input file that you want to change. In the first job, we have to generate the initial velocities. Thus, we give an initial temperature (TEMPI) and set the NTIVEL parameter to one. In the further jobs we will read the velocities from a file, hence we set NTIVEL to 0. You can see that we are increasing the temperature for both baths by 60K at every new job (TEMP0[1..2]). Simultaneously, the force constant (CPOR) for the position restraints is decreased by an order of magnitude at every new job. Finally, in the last columns you specify in which subdirectory and in what order you want to run the jobs. The only thing that is missing to start the equilibration are the files containing the position restraints of the solute. To prepare it, copy the coordinate file peptide_2Cl_h2o.cnf from the ion directory to the local directory and open it with a text editor and prepare the files peptide_2Cl_h2o.por and peptide_2Cl_h2o.rpr as you prepared the files sim_box_peptide.por and sim_box_peptide.rpr for the energy minimisation of the solvent (Sec. 2.2.3).

Now, run mk_script:

```
~/peptide/eq> mk_script @f eq_mk_script.arg
```

This creates five `eq_peptide_*.run` job scripts and the corresponding input files (`eq_peptide_*.imd`).

Warning: `mk_script` will not complain if the refpos file is not found! And the MD++ will crash if the `POSRESSPEC` block contains an empty line (only whitespace) at the end of the file!

You are now ready to start the thermalisation and equilibration. Run the first job script and the others will be automatically executed as soon as the preceding script has finished.

```
~/peptide/eq> ./eq_peptide_1.run
```

Warning: Depending on your system's speed this will take up to five hours. As we are using the simulation directory as the working directory there will be an error message after every job which can be ignored.

Hint: Have a look at all the output files `eq_peptide_*.omd`. If anything goes wrong, a message will be printed to the output file.

After the equilibration has finished, carry out some basic checks in the `eq/ana` directory.

```
~/peptide/eq/ana> ene_ana @f ene_ana.arg
~/peptide/eq/ana> xmgrace totkin.dat
```

There you can see that the kinetic energy is increasing at every new job (Fig. 2.3).

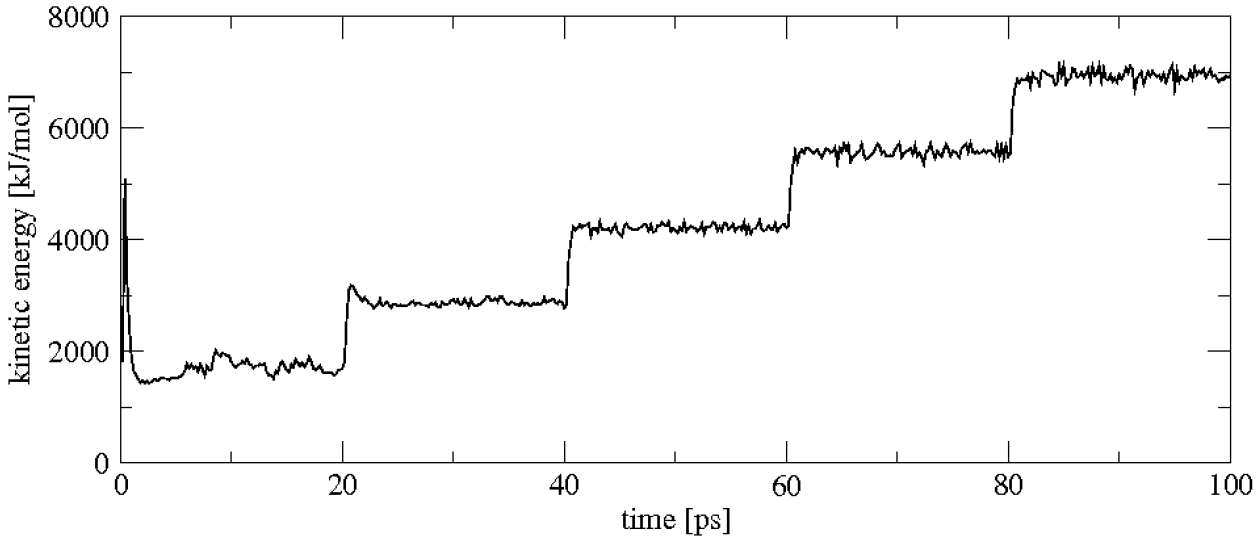


FIGURE 2.3. The kinetic energy during the equilibration.

2.3.2. Molecular dynamics sampling simulation. You need the following programs from the GROMOS simulation package:

GROMOS++:	`mk_script`
MD++:	`md`

You need the following input files:

`md_mk_script.arg md.imd mk_script.lib`

The procedure will create the following output files:

`md_peptide_*.run md_peptide_*.imd md_peptide_*.omd md_peptide_*.cnf md_peptide_*.trc.gz md_peptide_*.tre.gz`

The equilibration period already produced a short simulation at constant temperature and volume. At this point we want to elongate the simulation to a nanosecond under constant temperature and pressure. Go to the directory `md` and use the `mk_script` program to create the job scripts and the input files.

Have a look at the input file `md.imd`. Compared to the file used for the equilibration there are only a few differences: First, we don't use position restraining anymore and so, the `POSITIONRES` block was removed. We want to simulate under constant pressure rather than constant volume. For this purpose we have to add an additional block:

```
PRESSURESCALE
# COUPLE  SCALE     COMP     TAUP   VIRIAL
      2      1  0.0004575     0.5       2
#   SEMI (semianisotropic couplings: X, Y, Z)
    1 1 1
# PRES0(1...3,1...3)
 0.06102   0    0    0    0.06102  0   0   0   0.06102
END
```

In the `PRESSURESCALE` block we tell MD++ to calculate and scale the pressure by setting `COUPLE` to 2. As the box should be isotropically scaled we set `SCALE` equal to 1. The weak-coupling method (Sec. 2-12.2.2) uses two additional parameters: `COMP` is the isothermal compressibility and `TAUP` is the coupling time. We are calculating the molecular virial (`VIRIAL` is equal to 2), so intramolecular forces don't contribute to the pressure. The next line is only used for semi-anisotropic pressure coupling and can be ignored in our case. Finally, we have to specify the reference pressure in a tensor form.

In the other blocks only minor things have changed: the temperature was set to 300K and the trajectories are written out less often (every 250th step only). In the `mk_script` input file (`md_mk_script.arg`) the specifications of the position restraints file and the joblist file are not needed any more. Instead of the joblist we use the `@script` argument. Here, we tell `mk_script` to create 10 consecutive scripts, beginning with the first (1). Run `mk_script` to generate the job scripts.

```
~/peptide/md> mk_script @f md_mk_script.arg
```

This command creates 10 `md_peptide_*.run` job scripts and corresponding input files (`md_peptide_*.imd`). Now, you can submit the first script to the job control system (queue) or run it interactively on the command line.

```
~/peptide/md> ./md_peptide_1.run
```

After all the jobs are finished, you should start to analyse the trajectories.

2.4. Analysis of the penta-peptide trajectories

2.4.1. Analysis of the energy trajectory. GROMOS can write energies, free-energy λ-derivatives and block averages of these to separate trajectory files for later analysis. Program `ene_ana` extracts individual values from such files and can perform simple mathematical operations on them. In this tutorial we have only written energy trajectories (`*.tre.gz`) and in the following you will learn how to extract time series and averages from these files.

You need the following programs from the GROMOS simulation package:

GROMOS++:	`ene_ana`
others:	`xmgrace`

You need the following input files:

```
ene_ana_peptide.arg
```

The procedure will create the following output files:

```
ene_ana_peptide.out totene.dat totpot.dat totkin.dat pressu.dat solutemp2.dat solvtemp2.dat
```

Go to the direcory `ana`. In the subdirectory `ene_ana` have a look at the input file:

```
~/peptide/ana/ene_ana> cat ene_ana_peptide.arg
@en_files ../../md/md_peptide_1.tre.gz
          ../../md/md_peptide_2.tre.gz
...
@prop    totene totpot totkin solutemp2 solvtemp2 pressu
@topo    ../../topo/peptide_2Cl_54a7.top
@library ene_ana.md++.lib
```

With `@en_files` you tell `ene_ana` which energy trajectories should be read in. The `@prop` argument specifies for which properties the time series should be extracted from the energy trajectory. The topology is specified with `@topo`. One can specify an `ene_ana` library with the `@library` argument. Now you can run `ene_ana`:

```
~/peptide/ana/ene_ana> ene_ana @f ene_ana_peptide.arg > ene_ana_peptide.out
```

Have a look at the output:

```
~/peptide/ana/ene_ana> cat ene_ana_peptide.out
  property      average          rmsd    error est.
    totene    -33001.249    108.441729    4.49116286
    totpot    -40082.2818   138.689821    4.23048814
    totkin     7081.03283    95.494846    1.17739268
 solutemp2     301.387955    27.0028093   0.357744616
 solvtemp2     303.959409    4.13812131   0.051210902
    pressu      1.9180672    305.759605    7.91642893
```

The program calculates the average of the specified properties as well as the root-mean-square deviations (`rmsd`) and a statistical error estimate (`error est.`). The error estimate is calculated from block averages of growing sizes extrapolating to infinite block size[1].

Warning: Sometimes the error estimates are `NaN` (not a number), which is due to the fact that we do not have enough values to calculate a meaningful error estimate.

Program `ene_ana` also produced a couple of time series files

```
~/peptide/ana/ene_ana> ls *.dat
pressu.dat    solutemp2.dat totene.dat
totpot.dat    solvtemp2.dat totkin.dat
```

Exercise: Have a look at these time series with `xmgrace`. Annotate the plots (axes, legends etc.). Print the plots and hand them in.

In the following you should learn how to use the `ene_ana` library. In the input file we specified as a first property `totene`. How did `ene_ana` know which numbers should be extracted from the energy trajectory in order to calculate the total energy? Have a look at one of the energy trajectories

```
~/peptide/ana/ene_ana>less ../../md/md_peptide_2.tre.gz
TITLE
   GromosXX
   Automatically generated input file
```

```
   energy trajectory
END
TIMESTEP
            0  100.000000000
END
ENERGY03
# totals
  -3.304678446e+04
   7.179524549e+03
  -4.022630901e+04
   2.125140821e+02
   0.000000000e+00
   1.301350362e+02
   2.765269150e+01
   5.472635430e+01
   0.000000000e+00
  -4.043882310e+04
   6.127461522e+03
  -4.656628462e+04
   0.000000000e+00
   0.000000000e+00
   0.000000000e+00
   0.000000000e+00
```

As you can see there is a number of blocks, but you will not find `totene` anywhere. The `ene_ana.md++.lib` helps `ene_ana` to interpret the (free) energy trajectory files. This library file defines which value `ene_ana` should take from the trajectory if you ask for property `totene`. It can also calculate additional properties by performing simple mathematical operations on the values in the trajectories. [1]

Warning: The `ene_ana.md++.lib` file is very tightly coupled to the exact version of MD++ you use. The program checks if the file you specify matches the version of MD++ with which the energy trajectory was generated. If these do not match, you can find an updated version of `ene_ana.md++.lib` in the `data` directory of your MD++ installation.

The `ene_ana.md++.lib` file will look something like this:

```
~/peptide/ana/ene_ana>cat ene_ana.md++.lib | grep "totene = "
totene = ENER[1]
~/peptide/ana/ene_ana>more ene_ana.md++.lib
TITLE
  XX Library file for ene_ana
END
ENERTRJ
# block definition for the energy trajectory file.
# which is specified by the input flag en_files of program ene_ana.
#
# Use keyword 'block' to specify the blocks
#            'subblock' to specify name and dimensions of a set of data
#            'size' to specify a size that should be read in from the file
#                   this size can be used as dimension specification
#                   in a subblock definition. Using the prefix 'matrix_'
#                   with such a definition will expand the size N to
#                   N*(N+1)/2
#
# Following is the definition for a gromosXX energy trajectory
#
  block TIMESTEP
    subblock TIME 2 1
  block ENERGY03
    subblock ENER 35 1
    size NUM_BATHS
    subblock KINENER NUM_BATHS 3
    size NUM_ENERGY_GROUPS
    subblock BONDED NUM_ENERGY_GROUPS 5
    subblock NONBONDED matrix_NUM_ENERGY_GROUPS 4
    subblock SPECIAL NUM_ENERGY_GROUPS 9
    size NUM_EDS_STATES
```

[1] some properties can also be calculated without specifying a library file as some part of the library is implemented already in the program itself. However, understanding the library syntax is important as it allows you to calculate any property you wish from the energy trajectory.

```
    subblock EDS NUM_EDS_STATES 3
  block VOLUMEPRESSURE03
    subblock MASS 1 1
    size NUM_BATHS
    subblock TEMPERATURE NUM_BATHS 4
    subblock VOLUME 10 1
    subblock PRESSURE 30 1
END
FRENERTRJ
# block definition for the free energy trajectory file.
# which is specified by the input flag fr_files of program ene_ana.
#
# syntax as for the ENERTRJ definition
#
# Following is the definition for a gromosXX free energy trajectory.
#
  block TIMESTEP
    subblock TIME 2 1
  block FREEENERDERIVS03
    subblock RLAM 1 1
    subblock FREEENER 35 1
    size NUM_BATHS
    subblock FREEKINENER NUM_BATHS 3
    size NUM_ENERGY_GROUPS
    subblock FREEBONDED NUM_ENERGY_GROUPS 5
    subblock FREENONBONDED matrix_NUM_ENERGY_GROUPS 4
    subblock FREESPECIAL NUM_ENERGY_GROUPS 9
    size NUM_EDS_STATES
    subblock FREEEDS NUM_EDS_STATES 3
END
VARIABLES
# Here you can define variables to be calculated by the program ene_ana
# In principal the program refers to the blocknames you have defined above,
# accessing individual element using array indices (one- or two-dimensional)
#
# Predefined as well is the Boltzmann constant (as BOLTZ = 0.00831441) and
# the MASS which (if not present in the energy trajectory) will be calculated
# from the topology (if inputflag @topo is given).
#
# Additional properties can be defined here as a direct mapping of a known
# property or as an expression of such properties. Make sure that variables
# and operators are always seperated by spaces. Multi-line expressions are
# allowed.
#
# Examples that work with the standard gromos96 definition are
#         given below and are actually standardly define if no library
#         file is specified.
time   = TIME[2]
dvdl   = FREEENER[3]
totene = ENER[1]
totkin = ENER[2]
totpot = ENER[3]
...
```

As you can see `totene` is defined as the first entry of the `ENER` array. The `ENER` array is defined as a subblock of the `ENERGY03` block. This subblock has 1 column with 38 lines (`subblock ENER 38 1`).

Exercise: Use `ene_ana` to calculate the density of your system. You will have to specify a proper variable for the density behind the `@prop` argument. You can find out which variable from the `ene_ana.md++.lib` library file it should be. Plot the resulting density trajectory with `xmgrace`, annotate and hand in the plot.

Exercise: This is a rather advanced exercise. We want to extract the time series of the total nonbonded interactions of the peptide with itself, the chloride ions and the water. The energies we need for that are stored in the energy trajectory in the `# nonbonded` block. In section Sec. 2.2.2 we defined four energy groups: the peptide, the first chloride ion, the second chloride ion and the water molecules. The `# nonbonded` matrix contains the following information:

```
# nonbonded
#   van der Waals      Coulomb  LSR LSK
# nonbonded
```

```
-6.777453864e+01 -3.617661756e+02 0.000000000e+00 0.000000000e+00 # 1 - 1 peptide with peptide
-2.605382239e-02 -8.326755384e+00 0.000000000e+00 0.000000000e+00 # 1 - 2 peptide with first Cl-
-4.098207266e-01 -8.853119150e+01 0.000000000e+00 0.000000000e+00 # 1 - 3 peptide with second Cl-
-3.822977361e+01 -2.549027033e+03 0.000000000e+00 0.000000000e+00 # 1 - 4 peptide with water
 0.000000000e+00 -7.382455923e+01 0.000000000e+00 0.000000000e+00 # 2 - 2 first Cl- with itself
-2.759122936e-03 1.126804790e+00 0.000000000e+00 0.000000000e+00 # 2 - 3 first Cl- with second Cl-
 3.690831600e+01 -6.603813595e+02 0.000000000e+00 0.000000000e+00 # 2 - 4 first Cl- with water
 0.000000000e+00 -7.382455923e+01 0.000000000e+00 0.000000000e+00 # 3 - 3 second Cl- with itself
 3.924154608e+01 -5.281465864e+02 0.000000000e+00 0.000000000e+00 # 3 - 4 second Cl- with water
 6.258839244e+03 -4.214886612e+04 0.000000000e+00 0.000000000e+00 # 4 - 4 water with water
```

Copy the `ene_ana` library to your current directory and change its name (e.g. add a 'mod'). You will have to define four new variables. Add them at the end of the file (but before the last `END`). The variable for the peptide-peptide interactions could e.g. be called `e_pp`. It would be defined as:

```
e_pp = NONBONDED[1][1] + NONBONDED[1][2]
```

i.e. we add up the van der Waals and Coulomb energies[2] of the peptide-peptide interactions. Now define three more variables which you could call, e.g. `e_pCl1` (peptide with first chloride), `e_pCl2` (peptide with second chloride), `e_pwater` (peptide with water). Now you can specify these newly defined variables as properties (`@prop`) in the `ene_ana_peptide.arg` file. `ene_ana` will then produce the following four files:

```
e_pCl1.dat   e_pCl2.dat   e_pp.dat     e_pwater.dat
```

Plot these time series with `xmgrace`, annotate and hand in the plot. The plot should look like Fig. 2.4.

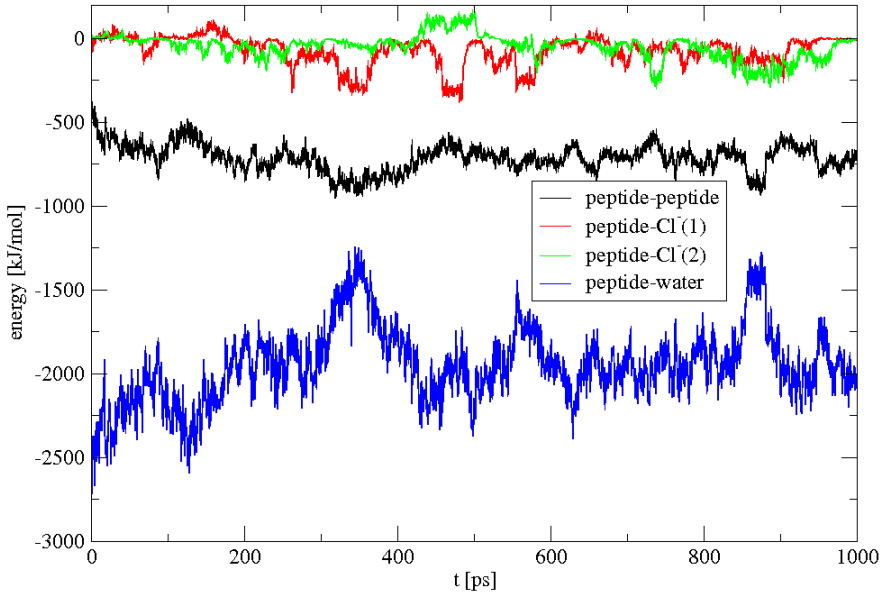


FIGURE 2.4. Time-series of energies.

2.4.2. Analysis of the coordinate trajectory.

2.4.2.1. *visual analysis.* You can generate PDB snapshots from your trajectory (or any other) coordinate files using the program `frameout`. Go to the directory `frameout`.

You need the following programs from the GROMOS simulation package:

GROMOS++:	`frameout`
others:	`vmd`

You need the following input files:

```
frameout_1.arg frameout_2.arg
```

The procedure will create the following output files:

```
FRAME_00001.pdb FRAME_00035.pdb FRAME_00049.pdb
```

[2] In LS simulations make sure to also add the third and forth column as these contain the real- and reciprocal-space contributions to the electrostatic energy.

Like in most GROMOS++ programs you have to specify some basic information about your system using the `@topo` and `@pbc` arguments. Let's look at an example:

```
~/peptide/ana/frameout> cat frameout_1.arg
@topo          ../../topo/peptide_2Cl_54a7.top
@pbc           r
@include       ALL
@outformat     pdb
@traj          ../../ion/peptide_2Cl_h2o.cnf
```

With the `@include` argument you tell frameout which atoms of the topology it should include in the output file. Here, we want to write all the atom coordinates (including the solvent) to the output file. Other options are `SOLVENT` which selects the solvent and `SOLUTE` which selects the solute. The latter is the default and so the `@include` argument can be omitted in this case.

The format of the output file is specified with the `@outformat` argument. Here, we use the PDB (Protein Data Bank) format. It is worth mentioning that other formats, including `cnf` the GROMOS coordinate format, are supported.

Hint: You can use frameout to create a gathered GROMOS frame by selecting `cnf` as output format.

Finally, you have to tell frameout where it can find the coordinate files with the `@traj` argument. Of course you can give multiple files as input.

Now try to run frameout:

```
~/peptide/ana/frameout> frameout @f frameout_1.arg
```

This command has generated a `FRAME_00001.pdb` file which can be opened with a molecular visualisation software (PDB viewer) like *VMD* or *pyMOL*.

Warning: Rename the PDB files generated by frameout or you might accidentally overwrite them.

It happens often that you see a certain effect (like a RMSD increase) at a certain time step and you want to have a look at this frame to see what happened. For this case, you have to add some arguments to the frameout input file:

```
~/peptide/ana/frameout> cat frameout_2.arg
@topo          ../../topo/peptide_2Cl_54a7.top
@pbc           r
@include       SOLUTE
@outformat     pdb
@traj
../../md/md_peptide_1.trc.gz
../../md/md_peptide_2.trc.gz
...
@spec          SPEC
@frames        35 49
```

First, you need to add all the MD trajectory coordinate files to the `@traj` argument. Because we want to extract a certain frame we tell frameout to be `SPEC`ific using the `@spec` argument. Last, the frames have to be given (`@frames`). This will create two PDB files containing the given frames. Open these files using *VMD* as well as the previous frame and compare the structures.

Hint: Use the option RMSD *Trajectory Tool* from the *VMD-Extensions-Analysis* menu. Align the structures and calculate the deviation from the initial structure. You can plot all three structures at the same time by going to *Graphics-Representation-Trajectory* and giving *1-3* for *Draw Multiple Frames*

Hint: You can use frameout to produce a movie that you can watch using *VMD* or *pymol*. Have a look at the files `frameout_movie.arg`. This will produce a single PDB file that contains all the configurations of the trajectory, after a roto-translational least-square fit based on the C_α atoms. Your favourite visualisation program can play this as a movie.

2.4.2.2. *Radial distribution function.* The radial distribution function (RDF) gives you the number density of atoms of a specified type around an atom (which can be of the same or of another type). The theoretical description is given in Sec. 5-4.48. GROMOS++ offers a tool that calculates the radial distribution function called `rdf`.

You need the following programs from the GROMOS simulation package:

GROMOS++:	`rdf`
others:	`xmgrace`

You need the following input files:

`rdf_peptide_2CL.arg`

The procedure will create the following output files:

`rdf_peptide_2CL.out`

Go to the directory `rdf` and have a look at the input file:

```
~/peptide/ana/rdf> cat rdf_peptide_2CL.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@pbc r
@centre a:CL
@with  s:OW
@cut   1.4
@grid  100
@traj  ../../md/md_peptide_1.trc.gz
       ../../md/md_peptide_2.trc.gz
...
```

The topology file is specified by `@topo`. With the option `@centre` you specify which atoms are supposed to be at the centre of your RDF and by `@with` which atoms should be the surrounding ones.

Hint: **The AtomSpecifier**

`a:CL` looks in all molecules for the atom named `CL` (chlorine ion)

`s:OW` looks in all solvent molecules for the atom named `OW` (water oxygen)

Have a look at the doxygen documentation of the *AtomSpecifier* and the *PropertySpecifier*.

With the parameter `@cut` you specify up to which distance you want to look at[3] and with `@grid` how many bins from 0 to `@cut` you want to have[4]. Finally you specify with `@traj` which trajectories you want to look at. The more trajectories, the better the statistics.

Now you can run `rdf`

```
~/peptide/ana/rdf> rdf @f rdf_peptide_2CL.arg > rdf_peptide_2CL.out
```

Have a look at the output. It should look similar to Fig. 2.5 where the first peak representing the first hydration shell is clearly visible meaning that the water molecules are organised around the chlorine ion. Note that there is no water molecule very close to the chloride ion as two atoms cannot be at the same place.

Exercise: Have a look at the difference between the RDF around the first and the second chlorine atom with `xmgrace`. Label the plot (axes, legends etc.). Print the plot and hand it in.

Hint: All you have to do to solve the exercise is to change the `@centre` in your input file. To find the atom specifier for the first chlorine atom, open the topology and see which `ATNM` (atom number) it has (should be 72). Then use

```
atominfo @topo ../../topo/peptide_2Cl_54a7.top @gromosnum 72
```

The atomspecifier is then `2:1` (first atom of the second molecule). Proceed in the same way for the other chlorine atom.

---

[3] do not go over half the box size

[4] bigger number means finer resolution, but less statistics

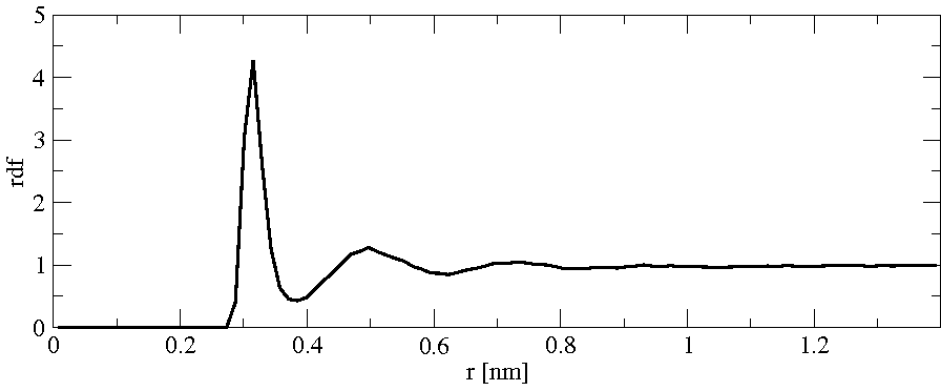


FIGURE 2.5. Cl-OW radial distribution function of chlorine ions in water

2.4.2.3. *Root-mean-square difference.* Atom-positional root-mean-square difference (RMSD) is a measurement of structural difference between two given conformations. For two conformations with the coordinates $\boldsymbol{r}$ and $\boldsymbol{r}^{\mathrm{ref}}$, the RMSD is given by

$$\mathrm{RMSD}(\boldsymbol{r}, \boldsymbol{r}^{\mathrm{ref}}) = \left(\frac{1}{\mathcal{N}_a} \sum_{i=1}^{\mathcal{N}_a} \left(\boldsymbol{r}_i - \boldsymbol{r}_i^{\mathrm{ref}} \right)^2 \right)^{\frac{1}{2}} \tag{2.1}$$

where $\boldsymbol{r}_i$ is the position of the i-th particle in the one configuration and $\boldsymbol{r}_i^{\mathrm{ref}}$ in the other, here called its reference position.

You need the following programs from the GROMOS simulation package:

GROMOS++:	`rmsd`
others:	`xmgrace`

You need the following input files:

`rmsd_peptide.arg`

The procedure will create the following output files:

`rmsd_peptide.out`

Go to the directory `rmsd` and have a look at the input file:

```
~/peptide/ana/rmsd> cat rmsd_peptide.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@pbc   r
@atomsrmsd 1:CA,N,C
@ref   ../../eq/eq_peptide_5.cnf
@traj  ../../md/md_peptide_1.trc.gz
       ../../md/md_peptide_2.trc.gz
...
```

Again the topology is given by `@topo` and `@pbc` defines the periodic boundary condition and gathers the frames. In `@atomsrmsd` one gives the atomspecifier of the atoms of which one wants to calculate the RMSD compared to a reference structure `@ref`. Here we want to analyse the backbone of the peptide (C_α, N, C) as a function of time. With `@traj` the coordinate trajectories are specified.

Now calculate the RMSD using the GROMOS++ program `rmsd`

```
~/peptide/ana/rmsd> rmsd @f rmsd_peptide.arg > rmsd_peptide.out
```

The output should look like Fig. 2.6, where you can see that at the beginning (t=0) the RMSD is zero as the structure in the beginning of the simulation is identical to the reference structure. The next few structures are nearly identical with the reference structure as well. Afterwards the RMSD increases as the structure is evolving away from the reference structure.

Exercise: Compare the RMSD of the backbone with the RMSD of the whole protein using `xmgrace`. Do you see more deviation from the original structure in the RMSD from the backbone or from the whole protein?

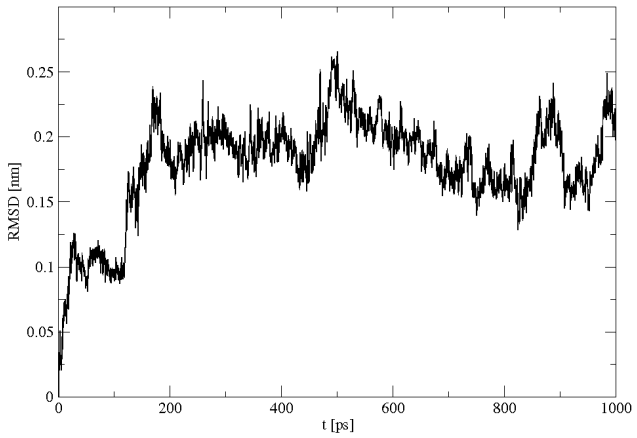


FIGURE 2.6. Atom-positional root-mean-square deviation of the backbone atoms of the penta-peptide from the starting (reference) structure.

Is that what you expected? Annotate the plots (axes, legends, etc.). Print the plot, answer the questions and hand them in.

Hint: The atom specifier for the whole protein is `1:a` (all atoms of the first molecule).

2.4.2.4. *Root-mean-square fluctuation.* Atom-positional root-mean-square fluctuation RMSF gives fluctuations of an atom coordinate $\boldsymbol{r}_i$ around its mean and is given by :

$$\mathrm{RMSF}(\boldsymbol{r}_i) = \left(\frac{1}{N_t} \sum_{n=1}^{N_t} \left(\boldsymbol{r}_i(t_n) - \langle \boldsymbol{r}_i \rangle \right)^2 \right)^{\frac{1}{2}} \tag{2.2}$$

where $\boldsymbol{r}_i(t_n)$ is the position of atom i at time $t_n = n\Delta t$ and $\langle \boldsymbol{r}_i \rangle$ is its mean position.

You need the following programs from the GROMOS simulation package:

GROMOS++:	`rmsf`
others:	`xmgrace`

You need the following input files:

`rmsf_peptide.arg`

The procedure will create the following output files:

`rmsf_peptide.out`

Go to the directory `rmsf` and have a look at the input file:

```
~/peptide/ana/rmsf> cat rmsf_peptide.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@pbc   r
@atomsfit 1:CA
@atomsrmsf 1:CA,N,C
@ref   ../../eq/eq_peptide_5.cnf
@traj  ../../md/md_peptide_1.trc.gz
       ../../md/md_peptide_2.trc.gz
...
```

Topology and periodic boundary conditions are given by `@topo` and `@pbc`, respectively. With `@atomsfit` one specifies which atoms are used to superimpose the structures (in order to remove translational and rotational changes in $\boldsymbol{r}_i$; just look at the intramolecular structural changes). `@ref` specifies the reference structure. The atoms for which the RMSFs are calculated are given by `@atomsrmsf` (here it is the backbone of the protein), and `@traj` indicates which coordinate trajectories are looked at.

Now you can run `rmsf`:

```
~/peptide/ana/rmsf> rmsf @f rmsf_peptide.arg > rmsf_peptide.out
```

Exercise: Look at the RMSF of the backbone to see that the ends of the peptide are more flexible than the middle. Look at the RMSF of the whole protein to see that the side chains move more than the backbone. Plot your output with `xmgrace`, label the graph and hand it in.

2.4.2.5. *Time series of properties.* Often you are interested in the time change of a certain property. You can monitor the properties of your system using time series. In addition, you may want to compare a property of your simulation with an experimental value. In this case a time-average is calculated which can be compared to experimental data. You need the following programs from the GROMOS simulation package:

GROMOS++:	`tser`
others:	`xmgrace`

You need the following input files:

`tser_peptide.arg`

The procedure will create the following output files:

`tser_peptide.out`

Such kind of analysis is carried out with the `tser` GROMOS++ program. `tser` is a very powerful program and only its basic function is explained here. Go to the directory `tser` and have a look into the example file:

```
~/peptide/ana/tser> cat tser_peptide.arg
@topo     ../../topo/peptide_2Cl_54a7.top
@pbc      r
@traj
 ../../md/md_peptide_1.trc.gz
...
@prop
d%1:3,69
d%1:37;2:1
a%1:70,69,71
t%1:47,46,48,49
```

First, you have to tell `tser` were the topology (`@topo`) resides and which boundary conditions (`@pbc`) you are using. With the `@traj` argument, tell `tser` where it can find the trajectory coordinates files. Second, tell `@tser` using `@prop` which properties it should calculate and print out.

1. In our penta-peptide system an interesting property is the head to tail distance. Its fluctuations over time give you an indication on the stiffness of the secondary structure: a stable α-helix, for example, has a rather constant head to tail distance. With `tser`, the calculation of distances is very easy: the `d` defines a distance between the two atoms of the atom specifier after the `%` sign. `d%1:3,69` thus calculates the distance between atoms 3 and 69 in molecule 1 (our penta-peptide). `1:3` is the nitrogen atom at the N-terminus, and `69` is the carbon atom at the C-terminus. Because atom `69` also belongs to the first molecule, `1:` can be omitted.
2. In the second example (`d%1:37;2:1`) the distance between the arginine residue (`1:37` is the `CZ` atom) and the first chloride ion is calculated. Because in the topology the chlorine atom is a molecule on it's own, you have to specify this with a molecule indicator (`2:` for the first chloride ion). Thus `2:1` denotes the first atom of the second molecule which is the first chloride ion.
3. The third property analysed is an angle (`a`). The angle is defined by the 3 atoms in the atom specifier after the `%` sign. In this case we monitor the atoms of the C-terminal carboxy group (`O1`, `C` and `O2`). You may also use an equivalent form of specifying the angle: `a%1:70-71`.
4. Finally, a torsional angle (`t`) is calculated. The torsional angle is defined by four atoms in the atom specifier after the `%` sign. Torsional angles are an important property of protein backbones (Ramachandran map) and are called ϕ- and ψ-angles. Here we look at the torsional angle between the H-N bond of lysine (atoms `47,46`) and the C_α-C_β (`48,49`) bond of the lysine residue.

Call `tser` and redirect its output:

```
~/peptide/ana/tser> tser @f tser_peptide.arg > tser_peptide.out
```

Now you can open the file in `xmgrace` and plot the time series. E.g. plot the time (1, first column in output) versus the arginine-chloride distance (`3` - third column in output).

```
~/peptide/ana/tser> xmgrace -block tser_peptide.out -bxy 1:3
```

Hint: The last line of the output file contains the averages of the properties.

Hint: Have a look at the doxygen documentation of *Property Specifier*. There you will find that you can specify many more properties (`@prop`) than shown in this simple example.

Hint: If you are simulating an elongated molecule, you should first gather the solute with `frameout` and then analyse the results using `tser` and vacuum boundary conditions (`@pbc v`). This way you avoid flawed results due to the nearest image distance between the specified atoms.

If everything worked out, your arginine-chloride distance should look like shown in Fig. 2.7.

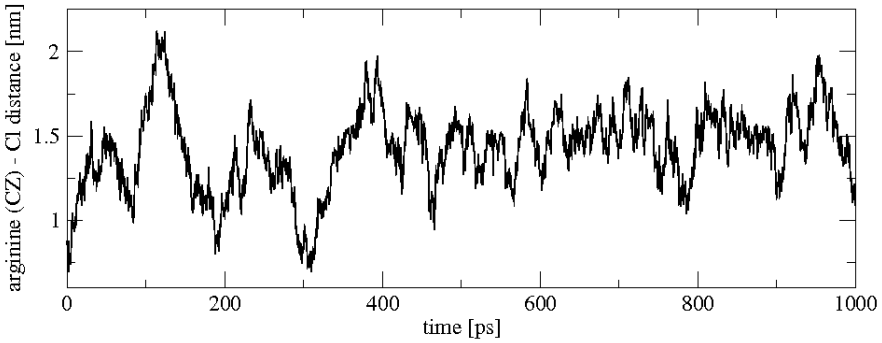


FIGURE 2.7. Time-series of the arginine (CZ) - chloride (1) distance.

Exercise: The distance between the two chloride atoms is an interesting property of your system. Create a time series and plot it with `xmgrace`. Annotate and hand in the graph.

2.4.2.6. *Hydrogen-bond analysis.* Hydrogen bonds are important intra- and intermolecular interactions. For example, the conservation of the genetic information of a cell is based on hydrogen bonding between bases in nucleic acids (Watson-Crick base pairing). The secondary structure of proteins is formed by hydrogen bonds between backbone amide hydrogens and oxygens. Thus, it is useful to analyse the hydrogen bonds of the penta-peptide in detail. You need the following programs from the GROMOS simulation package:

GROMOS++:	`hbond`
others:	`xmgrace`

You need the following input files:

`hbond_peptide.arg`

The procedure will create the following output files:

`hbond_peptide.out Hbond_2c_time_index.out Hbond_2c_time_numHb.out`

The GROMOS++ program `hbond` is a convenient tool for hydrogen bonds analysis. Go to the directory `hbond` and have a look at the input file:

```
~/peptide/ana/hbond> cat hbond_peptide.arg
@topo           ../../topo/peptide_2Cl_54a7.top
@pbc           r
@Hbparas       0.25 135
@massfile      mass.file
@AcceptorAtomsA 1:a
@AcceptorAtomsB 1:a
@DonorAtomsA   1:a
@DonorAtomsB   1:a
@traj
../../md/md_peptide_1.trc.gz
../../md/md_peptide_2.trc.gz
...
```

With the first two arguments you are already familiar from previous analyses. The `hbond` program determines the hydrogen bonds between donors (the atoms carrying a hydrogen atom) and acceptors (the atoms to which the bonds are formed) by geometrical criteria. `@Hbparas` takes two parameters for a hydrogen-bond calculation: a maximum distance between the hydrogen and the acceptor and a minimum angle between donor, hydrogen and acceptor atoms. The parameters in this example are used very often and can be considered as standard. `hbond` has to know which atoms it may consider as acceptors or donors. In the `@massfile` the acceptors and donors (and the hydrogen itself) are identified by their masses. You can define two groups (A and B) between which the hydrogen bonds are calculated. In this case we are interested in intramolecular hydrogen bonds, so both of the groups consist of atoms of the peptide only. We thus have to set the `@AcceptorAtomsA` and the `@DonorAtomsB` arguments both to an atom specifier of the whole peptide (`1:a`). The execution of the `hbond` program

```
~/peptide/ana/hbond> hbond @f hbond_peptide.arg > hbond_peptide.out
```

produces three output files: `hbond_peptide.out` (the redirected standard output), `Hbond_2c_time_index.out` and `Hbond_2c_time_numHb.out`

1. The `hbond_peptide.out` file contains a summary table. In this table all the hydrogen bonds are numbered (first column). In the second column you can find the molecule and residue numbers of the donor and acceptor, followed by the numbers and names of the atom involved in the hydrogen bond. In the next two columns the geometric properties are listed (average hydrogen-acceptor distance, average donor-hydrogen-acceptor angle). Finally, the last two columns show the occurrence (absolute and relative) of the hydrogen bonds.
2. The total number of hydrogen bonds at a certain time is listed in the `Hbond_2c_time_numHb.out` file.
3. The `Hbond_2c_time_index.out` file contains times series of all hydrogen bonds found in the summary table. The first column is the time, and in the second column is the ID number found in the first column of the summary file of the hydrogen bond occurring at this time. `Hbts.out` is usually filtered before plotting. A plot of the unfiltered file is shown in Fig. 2.8

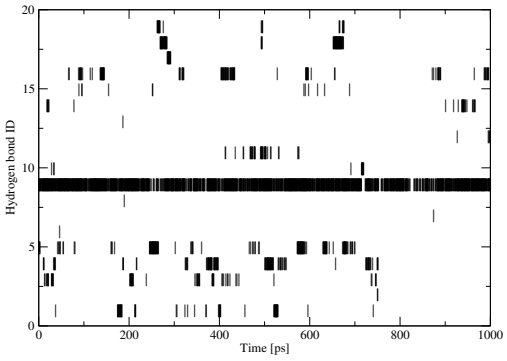


FIGURE 2.8. Time series of hydrogen bonds.

Exercise: Backbone hydrogen bonds are crucial for secondary structure. Can you tell which of them are present at $t = 640$ ps?

2.4.2.7. *Conformational clustering.* To map the structures sampled during the simulations onto a set of generic conformations, we can perform a clustering analysis of the MD trajectories.

You need the following programs from the GROMOS simulation package:

GROMOS++: `rmsdmat cluster postcluster`

You need the following input files:

`rmsdmat.arg cluster.arg postcluster.arg`

The procedure will create the following output files:

`RMSDMAT.dat cluster_structures.dat cluster_ts.dat postcluster.out cluster*.trc cluster*.cms cluster*.out`

The first program that we will use is the GROMOS++ program `rmsdmat`. This program calculates the atom-positional root-mean-square deviation between all pairs of structures in a given trajectory file. The RMSD matrix can be written out in a readable form, or in a binary format[5] in order to save disk space.

Go to the directory `cluster` and have a look at the input file `rmsdmat.arg`.

```
~/peptide/ana/cluster> cat rmsdmat.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@pbc   r
@atomsfit 1:CA,N,C
@stride 1
@human
@precision 6
@traj  ../../md/md_peptide_1.trc.gz
       ../../md/md_peptide_2.trc.gz
...
```

Topology and periodic boundary conditions are given by `@topo` and `@pbc`, respectively. With `@atomsfit` one specifies atom names which are to be used for least-squares fitting of the translational and rotational positions for the calculation of the RMSD. Here we only took the backbone atoms (N, C and C_α). With `@stride` one can specify a selection of structures in the trajectory file to be considered in this analysis. As we don't want to skip frames `@stride` is set to one. With `@human` the output file will be written in a readable form, otherwise it will be written in a binary form. With the argument `@precision` we specify the number of digits in the output matrix. With the argument `@traj` we specify the coordinate trajectories which will be analysed.

In the input for the `rmsdmat` program one can optionally specify the reference structure with respect to which the structures are fitted (`@ref ../../eq/eq_peptide_5.cnf`). Furthermore, in case the atoms differ from those on which the fit is performed this can be additionally specified with the argument `@atomsrmsd` 1:CA,N,C . If the reference structure is not provided in the input file, the first structure from the trajectory file is taken as the reference structure. By specifying a reference structure, one allows the program `cluster` (see below) to perform a forced clustering, where the first cluster contains the reference structure.

```
~/peptide/ana/cluster> rmsdmat @f rmsdmat.arg
```

As an output file you will get a file `RMSDMAT.dat`. This output file is used as an input file for the `cluster` program.

Now we will use the GROMOS++ program `cluster`. This program performs a conformational clustering based on a similarity matrix, such as calculated in program `rmsdmat`. The clustering algorithm is described in[2]. In this program a cut-off can be specified such that the structure pairs with RMSD values smaller than this cutoff are considered as structural neighbors. The structure with the highest number of neighbors is considered as the central member of the cluster of similar structures.

Have a look at the input file `cluster.arg`:

```
~/peptide/ana/cluster> cat cluster.arg
@rmsdmat RMSDMAT.dat
@human
@precision 6
@cutoff 0.06
@time 0 0.5
#@maxstruct
```

As previously mentioned `RMSDMAT.dat`, the output file of `rmsdmat`, is used here as an input file. With `@human` the program can use the readable matrix file. With the argument `@precision` we specify the number of digits in the input matrix. With the argument `@cutoff` we can specify the similarity criterion for two structures. In this case we are not reading a trajectory and thus we have to tell the program time information using the `@time` argument. The first argument (0) is the starting time and the second argument is the increment in time per element contained in the RMSD matrix. With `@maxstruct` we can specify how many structures are to be considered. Here we take all of them. If we have to perform a forced clustering to a specific structure, an extra argument has to be specified in the input file together with the number of the structure to which the clustering will be forced. For example, if one performs the calculation of the RMSD matrix with a reference structure, the reference structure used in this program is the first structure. This additional argument should look as follows: `@force 0`.

```
~/peptide/ana/cluster> cluster @f cluster.arg > cluster.out
```

[5] This is curcial for large trajectories.

The output consists of two files: `cluster_structures.dat` and `cluster_ts.dat`.

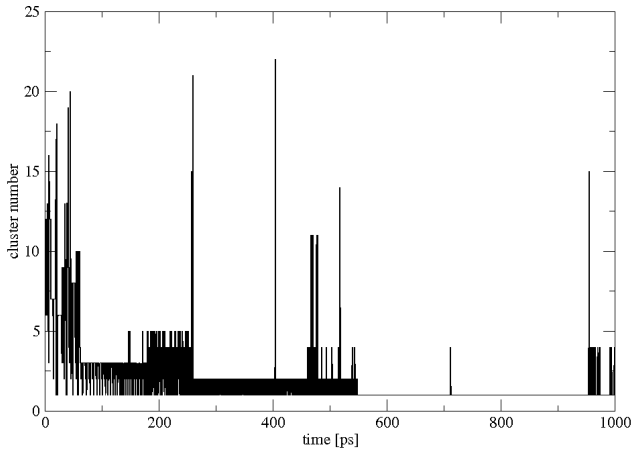


FIGURE 2.9. Time-series of the clusters.

File `cluster_structures.dat` contains information about the clustering procedure, information about each cluster such as the size of the cluster (i.e. how many structures are in each cluster), averaged lifetime of a given cluster and the number of the structure which is the central member structure of a specific cluster.

In `cluster_ts.dat` occurrences of specific clusters at specific times are listed (the time series of each cluster).

Next we will use the GROMOS++ program `postcluster`. This program can do additional analysis and data extraction on the output of the `cluster` program.

Have a look at the input file `postcluster.arg`:

```
~/peptide/ana/cluster> cat postcluster.arg
@topo          ../../topo/peptide_2Cl_54a7.top
@cluster_struct cluster_structures.dat
@cluster_ts    cluster_ts.dat
@clusters      1-8
@lifetime      2
@traj          ../../md/md_peptide_1.trc.gz
               ../../md/md_peptide_2.trc.gz
...
```

Among the analyses which one can do with the `postcluster` program is the lifetime-analysis. To perform it one has to specify the lifetime limit of a certain cluster with the argument `@lifetime`. It defines a number of subsequent structures that are significantly different from the structures of the cluster of interest. If this limit is reached, a switch to another cluster will occur. For example, if we specify a lifetime limit equal to 2, this means that at least two subsequent structures of a cluster that differ from the cluster of interest have to occur in order to detect a cluster transition.

```
~/peptide/ana/cluster> postcluster @f postcluster.arg > postcluster.out
```

The output of the `postcluster` program consists of trajectory files `*.trj` as well as `*.cms` files for each cluster. Thus, `postcluster` can also be used to write out the trajectory files and single structure files containing the central member structure of the cluster. With the argument `@clusters` we can specify for how many clusters we want to write out the trajectories and the corresponding central member structures. The resulting trajectories can further be used in any other analysis programs.[6]

2.4.2.8. *NMR analysis.* To compare simulated with experimental NMR data one may calculate NOE (Nuclear Overhauser Effect) distance upper bound violations and ^3J-coupling constant values and compare them with the corresponding experimental values.

You need the following programs from the GROMOS simulation package:

GROMOS++: `prep_noe noe post_noe jval`

You need the following input files:

`prep_noe.arg noe.arg post_noe.arg jval.arg jval.jvr prep.noe noecor.wuthrich noelib.54a7`

[6] You should compress them using `gzip` first.

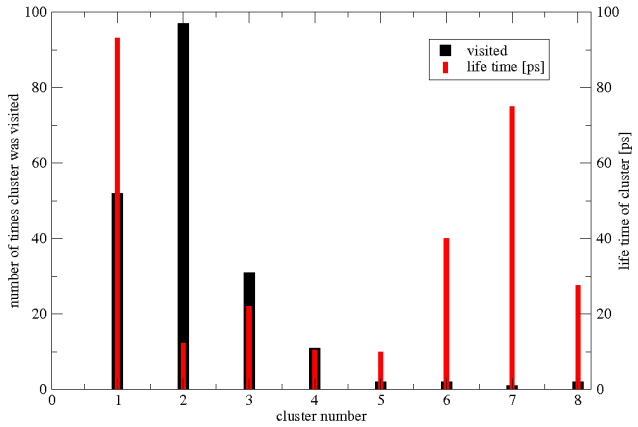


FIGURE 2.10. The number of visits and lifetimes of the clusters.

The procedure will create the following output files:

```
noe.filter prep_noe.out noe.dsr noe_peptide.out noets.out post_noe.out jval.out
```

Go into the subdirectory called `noe`. You will perform three steps in order to analyse the NOE distance upper bound violations. First you need to prepare the bonds derived from experiment such that GROMOS will understand them. Take a look at the argument file `prep_noe.arg`:

```
~/peptide/ana/noe> cat prep_noe.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@title NOE_specification_file
@filter 1000
@factor 10
@noe prep.noe
@lib noelib.54a7
@parsetype 1
@correction noecor.wuthrich
```

The topology file is given by the argument `@topo`. With `@title` one specifies the title in the `prep_noe` output. The `@filter` argument determines the upper limit of the NOE distance bounds which should be considered. By setting `@filter` to large value (1000 nm) we make sure that all distances will be taken into account. Most of the NMR experimental data is reported in Angstrom units [Å], whereas using the GROMOS force field the standard length unit is nanometer [nm]. Thus we have to give the program a `@factor` by which it has to scale all the NOE distance bounds prior to any other calculation. The input file in an *X-PLOR* like format (and units) with all NOE distances between the atoms of interest is specified by the argument `@noe`. Take a look at the `prep.noe` file in a text editor. It contains the atom number and the name of each hydrogen as it is used in structure determination and refinement using *X-PLOR*. In GROMOS the names used for the hydrogens might differ, thus we should specify a library file which converts the *X-PLOR* names into ones that the `noe` program can read. This is done with the argument `@lib`. The last three columns (6, 7 and 8) in the *X-PLOR* file stand for the upper bounds derived from the NOE intensities. There are three ways of parsing these columns. For that purpose program `prep_noe` uses `@parsetype` argument:

`@parsetype=1` : the first of the three columns is taken as the upper bound
`@parsetype=2` : (default value) the upper bound will be the sum of the first and third column
`@parsetype=3` : the upper bound is equal to the difference between the first and second column

If you set `@parsetype` to either `2` or `3`, you have to also use an argument called `@action`, which applies a correction to the upper bound value. The argument `@correction` specifies a file containing the information about types of correction that should be taken into account by `prep_noe`. NOE bound corrections are used in case of non-specific assignment of individual hydrogen atoms. Several methods have been proposed to define pseudo-atoms. Here we use the 'centre average' approach. The correction is two-fold: a pseudo-atom correction and a multiplicity correction. The former aims to solve the dilemma when several hydrogen atoms of group I interact with a second hydrogen atom, labelled S, and the interactions between the various pairs

of hydrogen atoms are practically indistinguishable. The solution to this problem is to define a pseudo atom, labelled Q, at the mean position of the hydrogen atoms of group I.

A sample of the correction file is shown below:

```
~/peptide/ana/noe> cat noecor.wuthrich
TITLE
NOE correction file containing the multiplicity corrections and pseudo
atom corrections in nm as described in
* Wuethrich, K.; Billeter, M. and Braun, W.: J. Mol. Biol. 169:949-961
  (1983)
* Wuethrich, K.: NMR of Protein and Nucleic Acids. John Wiley, New York
  (1986)
END
NOECORGROMOS
# NTPAC: NOE type to which the pseudo-atom correction applies
# NSPAC: NOE suptype to which the pseudo-atom correction applies
#       (set to 0 if no subtype defined)
# FTPAC: Distance of the pseudo-atom correction
#
# Possible combinations of NOE types/subtypes:
# NTPAC NSPAC NOE type
#    -1     1  flipping aromatic ring
#    -1     2  unassigned NH2 group
#     3     0  non-stereospecific aliphatic CH2 group
#     5     0  single CH3 group
#     6     0  non-stereospecific (CH3)2 group (isopropyl)
#     7     0  non-stereospecific (CH3)3 group (tert-butyl)
# NTPAC NSPAC FTPAC
    3    0    0.10000000
    5    0    0.15000000
    6    0    0.29000000
   -1    1    0.20000000
   -1    2    0.10000000
END
MULTIPLICITY
# NTMPC: NOE type to which the multiplicity correction applies
# NSMPC: NOE suptype to which the multiplicity correction applies
# FTMPC: Factor for the multiplicity correction
#
# Possible combinations of NOE types/subtypes:
# NTPAC NSPAC NOE type
#    -1     1  flipping aromatic ring
#    -1     2  unassigned NH2 group
#     3     0  non-stereospecific aliphatic CH2 group
#     5     0  single CH3 group
#     6     0  non-stereospecific (CH3)2 group (isopropyl)
#     7     0  non-stereospecific (CH3)3 group (tert-butyl)
# NTMPC NSMPC NTMPC
    3    0    1.00000000
    5    0    1.00000000
    6    0    1.00000000
   -1    1    1.00000000
   -1    2    1.00000000
END
```

The execution of the `prep_noe` program:

```
~/peptide/ana/noe> prep_noe @f prep_noe.arg > prep_noe.out
```

produces three output files: `prep_noe.out` (the redirected standard output), `noe.filter` and `noe.dsr`. Now look at the argument file `noe.arg` for the `noe` program:

```
~/peptide/ana/noe> cat noe.arg
@topo  ../../topo/peptide_2Cl_54a7.top
@pbc   r
@noe   prep_noe.out
@traj  ../../md/md_peptide_1.trc.gz
       ../../md/md_peptide_2.trc.gz
...
```

Topology and periodic boundary conditions are given by `@topo` and `@pbc`, respectively. Output from the previous program `prep_noe.out` is used here as the input file. Before executing the `noe` program open the `prep_noe.out` file and compare it with the `prep.noe`. With the argument `@traj` we specify the trajectories which will be analysed.

The execution of the `noe` program:

```
~/peptide/ana/noe> noe @f noe.arg > noe.out
```

produces the file `noe.out`. This file already contains the NOE violation information in a computer readable format. To make it more human readable you have to process it using the `post_noe` program. Now open the argument file for the `post_noe` program:

```
~/peptide/ana/noe> cat post_noe.arg
@topo        ../../topo/peptide_2Cl_54a7.top
@noe         prep_noe.out
@noeoutput   noe.out
@filter      noe.filter
@averaging   6
@distribution 0.05
```

The topology is given by `@topo`. The three following arguments `@noe`, `@noeoutput`, and `@filter` are output files of the `prep_noe` and `noe` programs. The `@noe` and `@noeoutput` arguments serve here as input files, whereas the `@filter` filters are the NOE distances which we will not use. With `@averaging 6` one specifies which averaging should be used. Here we are using the r^{-6} averaging. The argument `@distribution` indicates the binsize in which the final data will be separated.

The execution of the `post_noe` program:

```
~/peptide/ana/noe> post_noe @f post_noe.arg > post_noe.out
```

produces one output file: `post_noe.out`

The next analysis is ^3J-coupling constant analysis. In GROMOS, the ^3J-coupling constant can be calculated using program `jval`. It uses the so-called Karplus relation to relate the ^3J-coupling constant to the local molecular structure or torsional angle.

$$^3J(H_N, H_{\alpha/\beta}) = a\cos^2\zeta_n + b\cos\zeta_n + c \tag{2.3}$$

where ζ_n is the dihedral angle between the planes defined by the atoms (H, N, $C_{\alpha/\beta}$) and the atoms (N, $C_{\alpha/\beta}$, $H_{\alpha/\beta}$). In our calculations, we use parameters a, b, c equal to 6.4 Hz, -1.4 Hz and 1.9 Hz, respectively, which were optimized by Wüthrich et al.[3]

We will use the `jval` program to calculate the ^3J-coupling constant. Go into a subdirectory called `jval`.

Before we can calculate the ^3J-coupling constants, we need to define the torsional angle(s) that will be used for the calculation. For this we need a ^3J-coupling constant restraints file. In this example, as there are no experimental ^3J-values reported for this peptide, we will use hypothetical ^3J-coupling constants. The ^3J-coupling constant restraints file `jval.jvr` is already prepared for you and is shown below.

```
TITLE
penta-peptide, Val-Tyr-Arg-Lys-Gln, linear,
J-coupling constant restraints, hypothetical
END
JVALRESSPEC
# IPJV, JPJV, KPJV, LPJV:
#      atom sequence numbers of the real atoms defining the dihedral
#      angle that is related to the restrained J-value
# WJVR: individual 3J-value restraint weight factor by which
#      the restraining term for each 3J-value may be multiplied
# PJR0: the experimental or reference 3J-value. In case of a full-harmonic
#      3J-value restraint (NHJV = 0), it is the minimum-energy 3J-value;
#      in the case of an attractive or repulsive half-harmonic
#      3J-value restraint (NHJV = +- 1), it is the upper or lower bound,
#      respectively, beyond which the restraining force becomes non-zero.
# PSJR: phase shift or difference between the dihedral angle formed
#      by the possibly non-existing H-atoms defining the J-coupling
#      and the dihedral angle i-j-k-l formed by the real atoms
#      present in the simulation (in degrees)
# AJV, BJV, CJV:
#      Karplus parameters a, b and c for the J-coupling constant
#      expressed as a function of the dihedral angle
# NHJV: type of J-value restraint
#       0: full harmonic [recommended]
#      -1: half-harmonic repulsive
#      +1: half-harmonic attractive
#
#IPJV JPJV KPJV LPJV  WJVR  PJR0   PSJR    AJV    BJV    CJV   NHJV
    9   11   13   27    1.0   7.3  -60.0   6.4   -1.4   1.9    0.0
   27   29   31   44    1.0   8.6  -60.0   6.4   -1.4   1.9    0.0
```

```
   29   31   32   33   1.0  7.4 -120.0  9.5   -1.6  1.8    0.0
   29   31   32   33   1.0  5.7    0.0  9.5   -1.6  1.8    0.0
   44   46   48   57   1.0  7.4  -60.0  6.4   -1.4  1.9    0.0
END
```

The first four columns describe the dihedral angle η_n from which the ^3J-value is derived using the Karplus relation. The weight factor in the fifth column is only used for restraining and is ignored in this case. The sixth column is used as reference ^3J-value in order to calculate the deviation from it. As the measured ^3J-coupling constant may arise from a torsional angle ζ_n whose atoms, such as aliphatic hydrogens, are not explicilty represented in the GROMOS force field, the torsional angle ζ_n has to be related to the torsional angle η_n by a phase shift $\delta_n = \zeta_n - \eta_n$. This phase shift is read from the seventh column. In the following three columns you can specify different parameters for the Karplus relation for every individual ^3J-coupling constant. Finally in the last column the type of the ^3J-value restraining is specified. For more detailed information you can refer to Sec. 4-3.6.

Have a look at the input file `jval.arg`:

```
~/peptide/ana/jval> cat jval.arg
@topo          ../../topo/peptide_2Cl_54a7.top
@pbc           r
@jval          jval.jvr
@traj
../../md/md_peptide_1.trc.gz
../../md/md_peptide_2.trc.gz
...
```

You should be already familiar with the first three arguments from previous analyses. The argument `@jval` determines the file which contains the torsional angle specification corresponding to the specific ^3J-coupling constant.

The execution of the `jval` program:

```
~/peptide/ana/jval> jval @f jval.arg > jval.out
```

produces the output file `jval.out`. In the first columns it contains the information that we gave in the `jval.jvr` file, and the last five columns give the information about our calculated ^3J-coupling constants.

Hint: You can plot the observed vs. the calculate ^3J-values by `xmgrace -block jval.out -settype xydy -bxy 17:20:21`.

2.5. Enhancing sampling using Local Elevation

In a normal MD simulation like the one presented in this tutorial, not all minima of the potential energy surface are visited. This is called limited sampling and is a result of the rugged potential energy surface of the solvated peptide. There are many large ($> k_BT$) potential energy barriers present which prevent the peptide from changing from one minimal energy conformation to another. There are many sampling enhancement techniques available and the Local Elevation (LE) method[4] is discussed now.

Instead of looking at all degrees of freedom of the system, a set of possibly collective variables $\boldsymbol{Q}$ ($\boldsymbol{Q}_i$, $i = 1, 2, ..., N_{LE}$) such as dihedral angles can be used to describe the conformation of a peptide. For example the well-known Φ-Ψ dihedral angles can be used to describe the peptide's backbone conformation. In the Local Elevation method conformations visited in this (chosen) variable space are penalized by a multi-dimensional Gaussian-like potential energy term (see Sec. 2-9.13.1) centered at $\boldsymbol{Q}_i^0$

$$\mathcal{V}^{(le)}(\boldsymbol{Q}; n_k(t)) \quad = \quad \sum_{k=1}^{N_G} n_k(t) \prod_{i=1}^{N_{LE}} \gamma_i(\boldsymbol{Q}_i - \boldsymbol{Q}_{k,i}^0) \text{ with k} = 1, 2, ..., \text{N}_\text{G}. \tag{2.4}$$

For practical reasons the visited conformations are stored on grids. Everytime a grid point $\boldsymbol{Q}_{k,i}^0$ is visited, $n_k(t)$ is increased by one and thus the local-elevation energy function is pushing the conformation away from the grid point. For the potential energy function a grid-point centered γ_i is used. Usually this function is of Gaussian or Gaussian-like truncated polynomial functional form.

You need the following programs from the GROMOS simulation package:

MD++:	`md`
GROMOS++:	`mk_script tser`
others:	`xmgrace`

You need the following input files:

```
md_le.imd tyrosine.led tyrosine.lud mk_script.arg tser_md.arg tser_le.arg
```

The procedure will create the following output files:

```
le_peptide_1.* tser_md.out tser_le.out
```

Because the hydroxy-phenyl sidechain of the tyrosine residue is bulky, it is rather immobile in the first 100 ps of the simulation. In order to demonstrate the LE procedure we would like to enhance sampling on the N-CA-CB-CG dihdral angle of this residue. Change into the `le` folder and open the file `tyrosine.led`

```
~/peptide/le> cat tyrosine.led
TITLE
Dihedral angle definition file for local elevation.
END
LOCALELEVSPEC
# here we only define one dihedral angle on the
# tyrosine residue: N-CA-CB-CG
# NLEPID TYPE  IPLE  JPLE  KPLE  LPLE
      1    1    11    13    14    15
END
```

This file contains the `LOCALELEVSPEC` block which is used to specify the dihedral angles which are used as collective variables for local elevation. The first column is the number of the local elevation potential function we want to attach this dihedral to. The second column indicates the type of the internal coordinate. In the next four columns the atom numbers of the four atoms defining the dihdral angle are given. As we choose the N-CA-CB-CG dihedral angle we have to give the numbers 11, 13, 14 and 15.

Hint: The command `atominfo @topo ../topo/peptide_2Cl_54a7.top @atomspec '1:res(TYR:N,CA,CB,CG)'` yields this information.

Multiple collective variables now have to be grouped together to form the effective LEUS potential. Have a look at the `LEUSBIAS` block in the `tyrosine.lud` file.

```
~/peptide/le> cat tyrosine.lud
TITLE
Local elevation umbrella defintion file
END
LEUSBIAS
# NRUMB
  1
# NLEPID NDIM CLES
      1    1  0.1
# VARTYPE(N) NTLEFU(N) WLES(N) RLES(N) NGRID(N) GRIDMIN(N) GRIDMAX(N)
          1       0     1.0    1.0      36       0.0       0.0
# NCONLE
  0
# NVISLE ICONF
END
```

In this example we are using just one LEUS potential (`NRUMB`=1). Every LEUS potential gets its own identifier number (`NLEPID`). Note that this is the number we used in the `LOCALELEVSPEC` block. As we just use one dihdral angle in this example the dimensionality is one (`NDIM`=1). The force constant can be specified by `CLES`. The next line has to be given to characterize the (collective) variable attached to the LEUS potential function. `VARTYPE`=1 is used to specify a dihedral angle. The potential function in this variable γ_i is of truncated polynomial functional form (`NTLEFU`=0) with a width of `WLES` given in grid coordinates. It is truncated after a distance of `RLES` in grid coordinates. Finally, the grid has to be defined. It consists of `NGRID` grid cells and the boundary values are `GRIDMIN` and `GRIDMAX`. As we are using a cyclic dihedral angle coordinate the minimum and the maximum angle (in degree) are the same. Because we're just beginning the local elevation run we haven't penalized any conformation yet (`NCONLE`=0). Now the only thing still left to adjust is the MD++ input file `md_le.imd`. It is exactly the same file as the one in the `md` folder used for the main run but it contains an additional block to turn on local elevation.

```
LOCALELEV
# NTLES NLEPOT NTLESA NTWLE
     1      1      2   100
# NLEPID[1..NLEPOT]
     1
# NLEPFR[1..NLEPOT]
```

```
      0
END
```

NTLES is used to turn on LE. The number of LEUS potentials is given by NLEPOT. We want to read the LEUS definition from an external file (tyrosine.lud) and thus have to specifiy NTLESA=2. In order to monitor the LEUS potential built we write the LEUSBIAS block to the special trajectory every NTWLEth step. Finally we have to specify the ID number of the LEUS potential function (NLEPID). NLEPFR can be used to control whether the potential function is to be built or kept fixed and used for equillibrium sampling. We want to build and thus do not freeze the potential function.

The two additional files for LE (tyrosine.led and tyrosine.lud) have to be added to @files section in mk_script.arg. They are called ledih and leumb. Adjust the paths, run mk_script and the simulation.

```
~/peptide/le> mk_script @f mk_script.arg
~/peptide/le> ./le_peptide_1.run
```

Once the simulation has finished, change to the ana subdirectory and run the tser program to produce two time series of the dihedral angle. The first time series is obtained from the first 100 ps of the normal MD simulation, the second time series from the local elevation simulation.

```
~/peptide/le/ana> tser @f tser_md.arg > tser_md.out
~/peptide/le/ana> tser @f tser_le.arg > tser_le.out
```

Use xmgrace to visualise the angles and the distributions obtained from the two simulations. It should look as shown in Fig. 2.11. It can be seen that in the normal MD simulation the sidechain is moving only very slightly. The angle is fluctuating around the average value. In the LE simulation sampling of the rotation is enhanced by a large extent resulting in a uniform distribution of the dihedral angle.

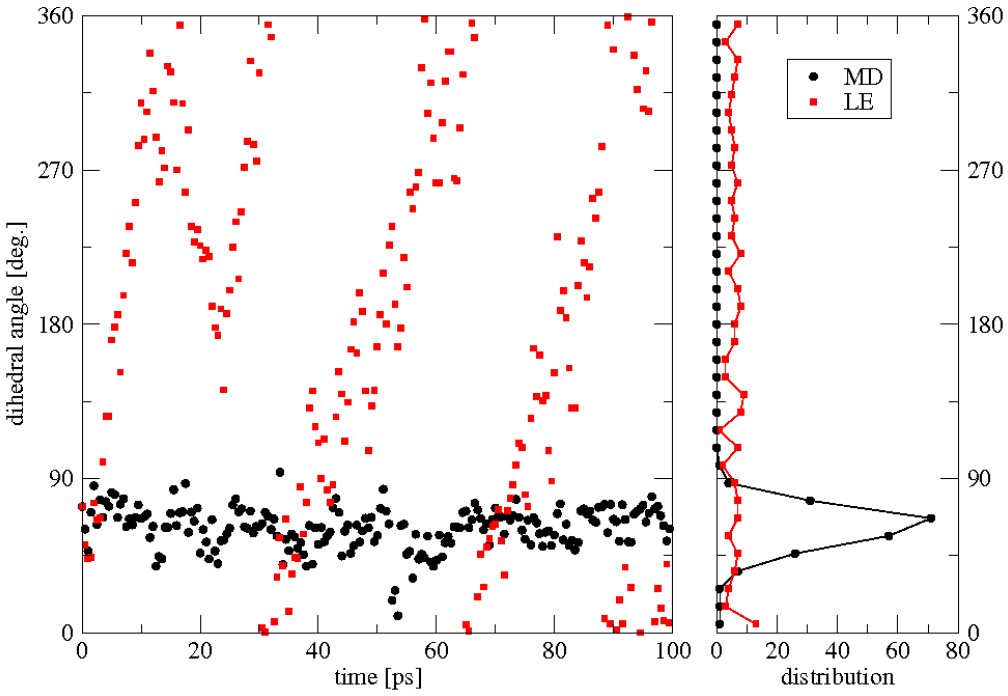


FIGURE 2.11. Tyr. N-CA-CB-CG dihedral angle time-series and distribution.

An application in ligand binding to the protein factor Xa involving LE-search for the χ_1 and χ_2 dihedral angles of Trp and Tyr side chains in the binding pockets can be found in[5]. An application of LE search to search possible conformations of a 10-residue loop of the protein Ribonuclease A can be found in[6].

As an exercise, do perform a LE search for the χ_1 and χ_2 side chain dihedral angles of the Tyr residue in the penta-peptide.

2.6. Free energy calculations

In GROMOS the free energy difference between two states A and B of a molecular system or between two molecular systems A and B can be calculated using i) thermodynamic integration (TI) (Sec. 2-14.6), ii) thermodynamic perturbation and extrapolation (Sec. 2-14.7), iii) umbrella sampling (Sec. 2-14.8), and iv) enveloping distribution sampling (Sec. 2-14.9).

2.6.1. Thermodynamic integration. In this exercise you will calculate two solvation free energies, the solvation free energy of a simple point charge (SPC) water molecule in a box of 999 SPC water molecules, and the solvation free energy of a methane molecule in a box of 999 SPC water molecules. Since the absolute value for free energy of solvation is difficult to calculate directly, we use the fact that the free energy is a state function. From the thermodynamic cycle presented in Fig. 2.12 it follows that

$$\Delta\mathcal{F}^{solvation}_{solute} = \Delta\mathcal{F}^{vacuo}_{dummies,solute} + \Delta\mathcal{F}^{solvation}_{dummies} - \Delta\mathcal{F}^{solvent}_{dummies,solute}. \tag{2.5}$$

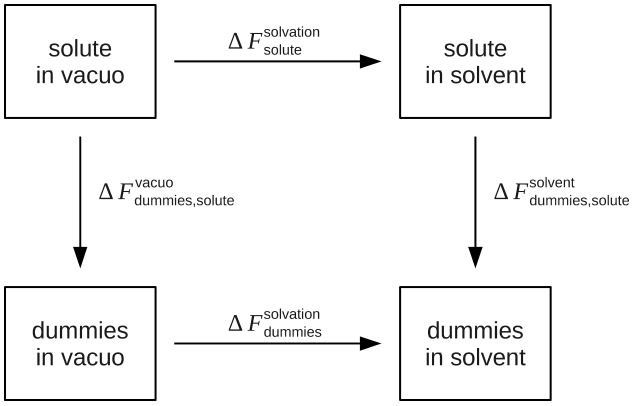


FIGURE 2.12. Thermodynamic cycle for the determination of solvation free energies

The solvation free energy $\Delta\mathcal{F}^{solvation}_{solute}$ is the work required to transfer a molecule from the gas phase into solution. $\Delta\mathcal{F}^{vacuo}_{dummies,solute}$ is the work required to remove all the internal nonbonded interactions in the solute in vacuo, which is achieved by gradually mutating all atoms in a solute molecule into dummy atoms. A dummy atom is an atom for which the nonbonded interactions, i.e. Lennard-Jones and electrostatic interactions, with all other atoms are set to zero while the bonded interactions within the molecule and the masses of individual atoms are kept unchanged. In the case of an SPC water molecule or a methane molecule this work is equal to zero. $\Delta\mathcal{F}^{solvation}_{dummies}$ is the work required to transfer the dummy solute molecule from vacuum to the solvated phase. As the dummy molecule does not interact with the rest of the system this term is also equal to zero. In order to determine the solvation free energy of an SPC water or a methane molecule in a box of SPC water only the free energy $\Delta\mathcal{F}^{solvent}_{dummies,solute}$ must therefore be calculated. Since $\Delta\mathcal{F}^{solvent}_{dummies,solute}$ is the work required to remove the solute-solvent and solute-intramolecular interactions. This value can be calculated by gradually mutating all atoms in the solute into dummy atoms. In this exercise you will do that by using the thermodynamic integration method (Sec. 2-14.6).

2.6.1.1. *Topology and perturbation topology files.* In order to perturb one SPC water molecule or one methane molecule in a water box into dummies, molecular topology files, which contain one water or one methane molecule as a solute, have to be prepared using the GROMOS++ program `make_top` as described in Sec. 7-2.1.1. The necessary input files `make_top_spc.arg` and `make_top_ch4.arg` are already prepared and you can find them in the subdirectory `topo` in the directory `TI`. Besides the molecular topology file, molecular perturbation topology file that specifies all the changes that occur to your solute in the perturbation, has to be prepared as well. In the case of the perturbation of a single SPC water molecule this file is called `spc_dummy.ptp` and you can find it in the subdirectory `topo` in the directory `TI`. If you have a look at it you will see that it lists the following perturbations: atom no. 1 of residue 1 `(OW)` will be changed from an oxygen atom with a integer atom code `(IAC)` 5, mass 15.99940 and a charge of -0.82 in the starting state to a dummy atom with `IAC` 22, mass 15.99940 and charge 0.00 in the final state. Besides that, atoms

no. 2 and 3 of residue 1 (HW1 and HW2) will be changed from hydrogen atoms with an IAC 21, mass 1.008 and charge 0.41 in the starting state to dummy atoms with IAC 22, mass 1.0080 and charge 0.00 in the end state. Since the Lennard-Jones as well as electrostatic interactions will be perturbed, the parameters ALJ and ACRF in the molecular perturbation topology file are set to 1.0. A similar file also has to be prepared for the perturbation of the methane molecule. Try to prepare this file yourself.

2.6.1.2. *Coordinate files.* In the subdirectory coord in the directory TI you will find two coordinate files, spc_1000.cnf and ch4_spc.cnf. The file spc_1000.cnf contains a rectangular box of 1000 SPC water molecules. Note that the first water molecule of the box will be treated as a solute in your simulation. The file ch4_spc.cnf contains a rectangular box with 1 methane molecule, which will be treated as a solute, and 999 SPC water molecules, which will be treated as solvent in your simulations.

2.6.1.3. *Thermalisation and equilibration.* As in the case of the peptide simulation the two systems considered in this exercise have to be slowly heated up from 60 K to 300 K, which is the temperature of your thermodynamic integration simulation. The input files for thermalisation of the SPC water box are prepared in the subdirectory eq_spc and the input files for equilibration of the SPC water box including methane are prepared in the subdirectory eq_ch4 of the directory TI. The process of thermalisation is for both systems the same as explained in Sec. 7-2.3.1, and it is specified in the joblist file called equilibration.jobs. Have a look at the files that have been prepared for you, generate the job scripts and the input files using the GROMOS++ program mk_script and then run the thermalisation of the two boxes as described in chapter Sec. 7-2.3.1. Each thermalisation will be 100 ps long. Note that you do not have to restrain the solute this time. After the thermalisation is finished you have to simulate the two boxes for another 100 ps at constant temperature and pressure. The necessary input files for the simulations are prepared in the subdirectories md_spc and md_ch4 of the directory TI. Again generate the job scripts and the input files using the GROMOS++ program mk_script and start the simulations as described in Sec. 7-2.3.2. Once the simulations are finished you are ready to start the thermodynamic integration simulations. Note that TI simulations can take up to ten hours, it is therefore recommended to run them over night.

2.6.1.4. *Thermodynamic integration simulations.* In this exercise you will run two thermodynamic integration simulations, one in which a SPC water molecule will be changed into dummies and one in which a methane molecule will be changed into a dummy atom. If your computer allows it, you can run the two TI simulations in parallel. In both cases you will run the thermodynamic integration simulations at 21 equally distributed λ points. At each λ point you will simulate your system for 60 ps. The first 20 ps will be taken as an equilibration period and will not be used in the subsequent free energy calculation. You can find the necessary input files for both TI simulations in the subdirectories TI_spc and TI_ch4 of the directory TI. If you take a look at the startup files for TI simulations, TI_spc_dummy.imd and TI_ch4_dummy.imd, you will notice that the main difference with respect to the input files used in equilibration simulations is an extra block called PERTURBATION in which the perturbation parameters are specified:

```
PERTURBATION
#   NTG: 0..1 controls use of free-energy calculation.
#        0: no free-energy calculation (default)
#        1: calculate dH/dRLAM
# NRDGL: 0,1 controls reading of initial value for RLAM.
#        0: use initial RLAM parameter from PERTURBATION block
#        1: read from configuration
#  RLAM: 0.0..1.0 initial value for lambda
# DLAMT: >= 0.0 rate of lambda increase in time.
# ALPHLJ: >= 0.0 Lennard-Jones soft-core parameter
# ALPHC: >= 0.0 Coulomb-RF soft-core parameter
#  NLAM: > 0 power dependence of lambda coupling
# NSCALE: 0..2 controls use of interaction scaling
#        0: no interaction scaling
#        1: interaction scaling
#        2: perturbation for all atom pairs with scaled
#           interactions. No perturbation for others.
#
#    NTG   NRDGL   RLAM  DLAMT
       1      0    0.0    0.0
#  ALPHLJ  ALPHC   NLAM NSCALE
     0.5    0.5      2      0
END
```

With setting the switch `NTG` to 1 you specify that you would like to do a free energy calculation. The parameter `NRGDL` controls reading of the initial value for the coupling parameter λ. In the `PERTURBATION` block this value is specified with the parameter `RLAM` and can be read from the jobscript file when the TI simulation is set up. If the coupling parameter λ is a function of time, the rate of λ increase in time has to be specified with parameter `DLAMT`. The parameters `ALPHLJ` and `ALPHC` control the softness of the Lennard-Jones and electrostatic interactions, the parameter `NLAM` defines the power dependence of λ coupling and the parameter `NSCALE` allows scaling of interactions between the energy groups. Besides the `PERTURBATION` block which has been added to the startup file for the TI simulation, the difference between the startup file for the TI simulation and the input file used in the equilibration simulation is that writing of the free energy trajectory is now specified using the switch `NTWG` in the `WRITETRAJ` block. By running the GROMOS++ program `mk_script` using a joblist `TI_joblist.dat`, 21 directories with the corresponding input files and job scripts for running the simulations will automatically be generated. You can start the TI simulation by running the first script, i.e. `TI_spc_dummy_1.run` or `TI_ch4_dummy_1.run` in the directory `L_0.0`. After the last simulation at a certain λ point is finished, a simulation at the next λ point will start using the coordinates generated in the last simulation at a previous λ point. After the simulations at all λ points have been finished, run the script `perform_analysis.sh`, which you can find in the directories `TI_spc` and `TI_ch4`. This script will create an input file `ene_ana.inp` in each of the λ directories, which will be used to calculate the $\langle \partial \mathcal{H}/\partial \lambda \rangle_{\lambda_n}$ (see Sec. 2-14.6) using the GROMOS++ program `ene_ana` (Sec. 7-2.4.1). Furthermore the `perform_analysis.sh` script runs the `ene_ana` program, copies the `dvdl.dat` files back in the λ directories and extracts the ensemble averages $\langle \partial \mathcal{H}/\partial \lambda \rangle_{\lambda_n}$ from the `ene_ana.out` files which were generated in each of the λ directories by the `ene_ana` program. In order to calculate the free energy difference of perturbing the SPC water molecule or the methane molecule to dummies, the calculated ensemble averages have to be integrated over all λ points. You can do that by plotting $\langle \partial \mathcal{H}/\partial \lambda \rangle_{\lambda_n}$ as a function of λ using the `xmgrace` program and calculating the respective integral using the option /Data/Transformations/Integration in the menu bar (see Fig. 2.13).

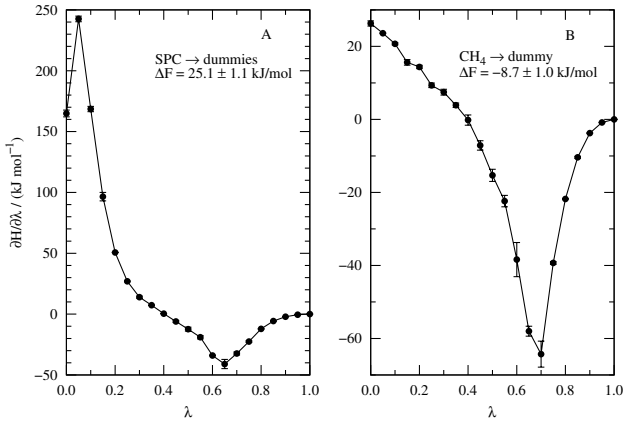


FIGURE 2.13. Thermodynamic integration perturbing (A) an SPC water molecule into dummy atoms (B) a methane molecule into a dummy atom[7].

2.6.2. Enveloping distribution sampling. In this exercise you will calculate the difference in solvation free energies between a simple point charge (SPC) water molecule and a methane molecule in a box of 999 SPC water molecules. In Sec. 7-2.6.1 we have calculated both solvation free energies, $\Delta G^{\text{solvation}}_{\text{SPC}}$ and $\Delta G^{\text{solvation}}_{\text{CH}_4}$, separately. Now we will determine the relative free energy $\Delta G^{\text{solvent}}_{\text{SPC}\rightarrow\text{CH}_4}$ directly via Enveloping Distribution Sampling (EDS) and compare the result to the difference of the two former values by making use of the thermodynamic cycle shown in figure Fig. 2.14.

$$\Delta G^{\text{solvent}}_{\text{SPC}\rightarrow\text{CH}_4} = \Delta G^{\text{solvent}}_{\text{dummies}\rightarrow\text{CH}_4} - \Delta G^{\text{solvent}}_{\text{dummies}\rightarrow\text{SPC}} \tag{2.6}$$

The relative free energy $\Delta G^{\text{solvent}}_{\text{SPC}\rightarrow\text{CH}_4}$ is the free energy change associated with the perturbation of a SPC molecule to a methane molecule in the solvated phase. This free energy change is estimated from simulation in a reference state with the corresponding potential energy $\mathcal{V}_R$. The reference state is designed in such a way

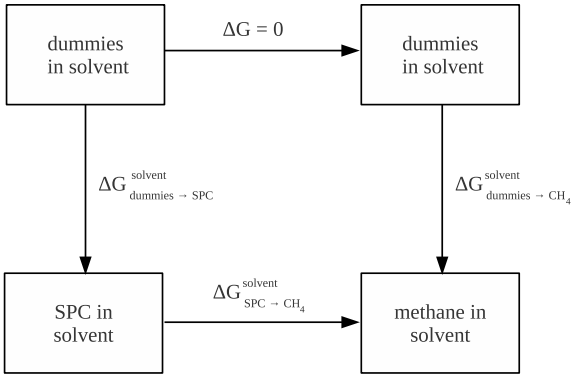


FIGURE 2.14. Thermodynamic cycle for the determination of relative solvation free energies

that the important phase spaces of both endstates, in our case the solvated SPC molecule and the solvated methane molecule, are sampled in the reference state simulation. The free energy difference $\Delta G^{\text{solvent}}_{\text{SPC}\rightarrow\text{CH}_4}$ can then be evaluated from the reference state simulation, see Sec. 2-14.9.

2.6.2.1. *Topology and perturbation topology files.* In order to perturb one SPC water molecule in a water box into a methane molecule a single or dual topology approach can be applied. In the single topology approach only the topology of one of the end states is used. The perturbation is specified in the molecular perturbation topology file based on this topology. You can find it in the subdirectory `topo` of the directory `EDS`. If you have a look at it you will see that the oxygen atom `OW` of the SPC molecule (state A) is perturbed to a carbon atom with an integer atom code 17. Besides that the hydrogen atoms of state A are perturbed to dummy atoms.
In the dual topology approach a combined topology of both end states is used. You can prepare the combined topology file with the GROMOS++ program `prep_eds`. Give the topology files of both end states as arguments:

```
~> prep_eds @topo <molecular topology files> @numstat 2 @param 1 @solv 1 > com_eds_H2O_ch4.top
```

You can find the corresponding molecular perturbation topology in the subdirectory `topo`. When you are finished with the single topology approach, which is described in the following, try to set up the dual topology approach, accordingly. The subdirectory `prodrun_dual` of the directory `EDS` is already prepared.

2.6.2.2. *Coordinate files.* For the EDS simulations the same coordinate files as in the TI chapter can be used. As you have already thermalised and equilibrated the system prior to the TI simulations you can use the equilibrated simulation boxes as input to the EDS simulations. For the dual topology approach the coordinates of both end states in one simulation box are needed. They are already prepared in the subdirectory `prodrun_dual` of the directory `EDS`. In the production simulations the two end state molecules are held together through a distance restraint between the oxygen atom of water and the united atom representing methane.

2.6.2.3. *Parameter update scheme simulations.* Prior to the EDS simulations suitable EDS parameters have to be determined. This can be done using an automatic iterative procedure which is described in[8]. You can find a bash script in the subdirectory `update_new` of the directory `EDS` which calculates new EDS parameters after evaluating the previous simulation run. It is named `submit_jobs_eds_update_new.sh`. The last command of this script is the submission of the next simulation run. You have to adapt the submission command to the infrastructure of your cluster. At the beginning of the script the variables needed to start the parameter update scheme are defined. Most of them are already initialized with the correct values. You have to specify the correct paths to the gromos directory and the working directory. If you take a look at the template input file for EDS simulations, `eds_template.imd`, you will notice that the main modification is an extra block called `EDS`, in which the EDS parameters are specified:

```
EDS
# EDS
1
# ALPHLJ ALPHC
  0.0    0.0
# FUNCTIONAL FORM
```

```
1
# NUMSTATES
2
# S
 0.08
# EIR
 0.0
25.0
END
```

With setting the switch `EDS` to 1 you specify that you would like to do a EDS simulation. The parameters `ALPHLJ` and `ALPHC` should be ignored, i.e. set to zero. They will be removed in the future. The parameter `FUNCTIONAL FORM` chooses the functional form of the reference state Hamiltonian and the parameter `NUMSTATES` defines the number of end states. At the end of the block starting values for the EDS parameters are set. The smoothness parameter should be set to a rather low value, the initial energy offset could be estimated from the energy difference between the two end states sampled in a short EDS simulation using a very low smoothness parameter and an energy offset of zero. The resulting value is then also a reasonable estimate for the parameter `Esep` in the bash script `submit_jobs_eds_update_new.sh`. You can perform the parameter update scheme by running the bash script `submit_jobs_eds_update_new.sh`. It will take some time, so make sure to run it with `nohup` in the background. Only the first job script needs to be submitted by hand. You can create the first job script with the `mk_script` file. This procedure generates also an input file which is essentially the same as the template input file. However, the template input file is formatted in a way that is compatible to the bash script `submit_jobs_eds_update_new.sh` and should therefore be used in the following. Thus, overwrite the just generated input file by

```
~> mv eds_template.imd eds_spc_ch4_1.imd
```

When the update scheme simulations are finished, have a look at the convergence of the EDS parameters. They will be written out by the bash script `extract.sh` to the output files `s_series.dat` and `eir_series.dat`. In this relatively simple example the smoothness parameter increases up to a value of 1 (see Fig. 2.15), which is usually not the case for more complex perturbations.

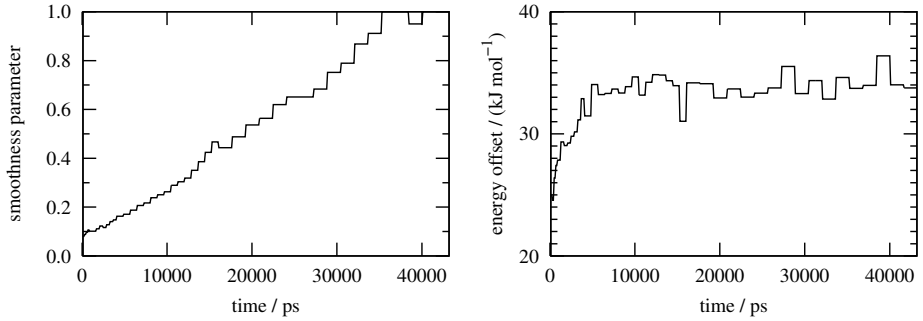


FIGURE 2.15. Convergence of the smoothness parameter (left) and the energy offset (right) for the parameter update scheme.

2.6.2.4. *Enveloping distribution sampling production simulations.* Now you will run an EDS simulation with fixed parameters in which the reference state envelopes the end states defined in the perturbation topology. You can find the necessary input files in the subdirectory `prodrun` of the directory `EDS`. In the input file the smoothness parameter is set to 0.8 and the energy offset to 33.0. In the file `eds.jobs` you can see the joblist for the simulation. The simulation time is 1 ns of equilibration and 20 ns of production and is split in 21 jobs. By running the GROMOS++ program `mk_script` using the argument file `md_mk_script.arg` the 21 input files and job scripts for running the simulation will automatically be generated. Make sure that you have correctly adapted the paths to the GROMOS program `md`, the working directory and the `mk_script` library to your system. You can start the EDS simulation by running the first script, when a job is finished it will submit the next one. After the last simulation is finished you can start with the analysis, which is prepared in the subdirectory `ana/ene` of the directory `prodrun`. Take a look at the file `ene_ana.md++.lib` and search for the section `EDS variables`. Here, the end state energies and the energy difference between them are defined. First run the energy analysis `ene_ana` with the corresponding argument file. It will write out the energy time series for the reference state `eR.dat` as well as of the two endstates,

`e1.dat` and `e2.dat`, and the energy difference `diff21.dat` between the two endstates. The GROMOS++ program `dfmult` calculates the resulting free energy difference from these energy time series. Use the given argument file to run it and write the output in a file `dfmult.out`. You can visualize if the sampling of both endstates in the reference state simulation is sufficient with some standard plots. One is the energy difference time series in Fig. 2.16 on the left hand side. To plot the distributions of the energy difference on the right hand side you have to reweight the energy difference from the reference state to the endstates. There is a GROMOS++ program for this which is called `reweight`. Have a look at the script `reweight.sh`, run it and plot the resulting distributions. To check if the important phase spaces of both end states were sampled in the EDS simulation, the potential energy distributions from the reference state simulation are compared to the potential energy distributions from independent MD simulations of the end states. With the script `reweight_endst.sh` you can reweight the potential energy distributions of the end states from the reference state to the end states. In the subdirectory `distri_solute_nonb` you can evaluate the potential energy distributions of the end state simulations carried out in Sec. 7-2.6.1. An energy analysis file is already prepared to evaluate the nonbonded potential energy of water. After the energy analysis is finished execute the bash script to obtain the distribution of the energy from the time series. This distribution should look very similar to the one obtained from the EDS simulation, see Fig. 2.17. If the distributions are too noisy you should extend the end state simulations in Sec. 7-2.6.1.

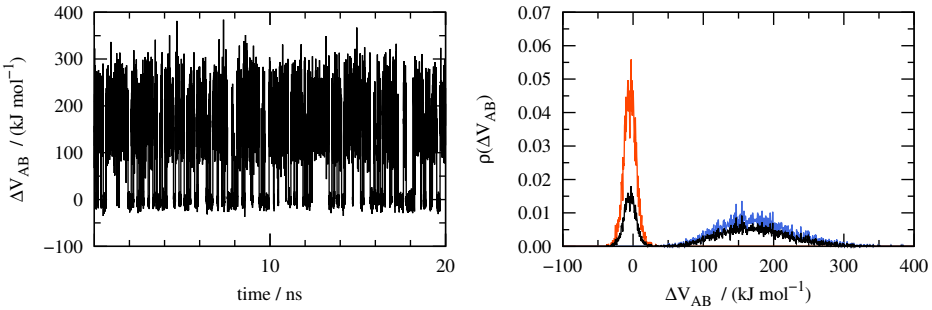


FIGURE 2.16. Energy difference time series in the reference state simulation (left) and energy difference distributions in the reference state as well as reweighted to the endstates (right). The blue curve corresponds to state A (SPC) the red curve to state B (CH_4).

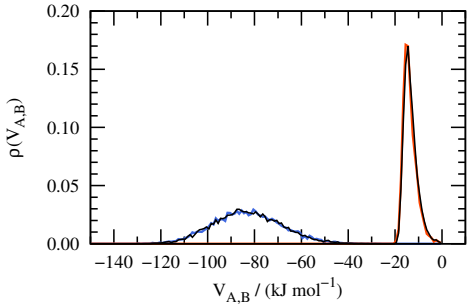


FIGURE 2.17. Comparison of the potential energy distributions of the EDS end states reweighted from the reference state simulation (red and blue) with the independent MD simulations (black).

2.7. Constructing a new building block

As mentioned in Chap. 7-1 there might be cases where the needed molecular topology building block is not available in the standard distribution. The following task is to construct a new building block.

You need the following programs from the GROMOS simulation package:

GROMOS++: `prep_bb`

You need the following input files:

`54a7.mtb 54a7.ifp prep_bb_DTT.arg spec_DTT.dat`

The procedure will create the following output files:

`BUILDING.out`

When creating a new building block it is highly recommended to first prepare pictures containing all the topological information. Draw your molecule on paper taking the united atom approach into account. Number the atoms sequentially, thinking about how the charge groups will be defined. By analogy to the existing GROMOS building blocks in Chap. 3-4 define the integer atom codes, mass codes and charges of all the atoms as they are described in Chap. 3-3. Define the charge groups making sure they are neutral unless the molecule itself is charged. By analogy to the existing GROMOS building blocks define also bond types, bond angle types, dihedral angle types and improper dihedral angle types. Once you have the picture ready use the GROMOS++ program `prep_bb`, which will help you create the building block.

In this exercise you will generate a building block for 2,3-dihydroxy-1,4-dithiobutane (DTT), a small detergent molecule. The pictures with the parameters for DTT are already prepared (Fig. 2.18).

Go into the subdirectory `DTT_bb` of the directory `peptide` in your home.

```
~> cd ~/peptide/DTT_bb
```

Have a look at the input file `prep_bb_DTT.arg`. Next to the standard force field files specified under the flags `@build` and `@param` a special file which lists sequentially all the atoms in the molecule and all the bonds between them has to be specified. For the case of DTT this file is already prepared and is called `spec_DTT.dat`. Since we will run the program `prep_bb` interactively the flag `@interact` is also needed. Try to run the program `prep_bb`

```
~/peptide/DTT_bb> prep_bb @f prep_bb_DTT.arg
```

The program will first check whether there are any aromatc rings present in the molecule. Then it will go through the list of atoms specified in the file `spec_DTT.dat` and for each atom will ask you to input the respective integer atom type (IAC), mass code, charge and charge group specifier. It will also give you suggestions for all these values as you go along. You can compare these suggestions with the parameters in Fig. 2.18. All the parameters are given in Chap. 3-3. Let's do this for the atom (CA in Fig. 2.18). First you have to give IAC (integer atom code). As CA is a united atom consiting of a carbon with two hydrogens, its IAC is 15, in agreement with Fig. 2.18 A. This means its mass code is 4. Next the program asks for the charge. We give a 0.150. The next thing the program wants to know is the charge group code. GROMOS topologies usually have charge groups with an integer charge, preferably zero. As the CA is charged, we should give a 0 and add the next atoms until we have an uncharged or integer charge charge group. Then we would give a charge group code 1 to the last atom. Otherwise if CA is uncharged, the charge group can be closed with a 1. In the GROMOS building block figures charge groups are specified by color. Now do the same for the other atoms using Fig. 2.18 A for support.

The same procedure will follow for the bonds, bond angles, dihedral angles and improper dihedral angles, respectively. The program will first ask you for the bonds. The first bond is between atom 1 (CA) and 1 (SA). In Fig. 2.18 B you see that we want this bond to be of type 32. Continue for the other bonds. The bond angles follow, with the first angle being between atoms 2 (SA) - 1 (CA) -4 (CB) being of type 16. Again, continue for all the bond angles. Then the improper dihedrals are set. For the first one 4 (CB) - 1 (CA) -5 (OB) - 7 (CG) the type is given as 2 (Fig. 2.18 C). Note that DTT has two chiral centers (CB and CG). Therefore there are four different combinations of the two correspoding improper dihedrals. In this exercise you decide for one of them. Fig. 2.18 shows (2R,3R)-dihydroxy-1,4-dithiobutan. Finally the program asks you about the proper dihedrals. The first proper dihedral is 4 (CB) - 1 (CA) - 2 (SA) - 3(HA) and has type 26. Define the rest of them yourself.

At the end the building block for DTT will be written to a file called BUILDING.out. Open it and verify if it is correct.

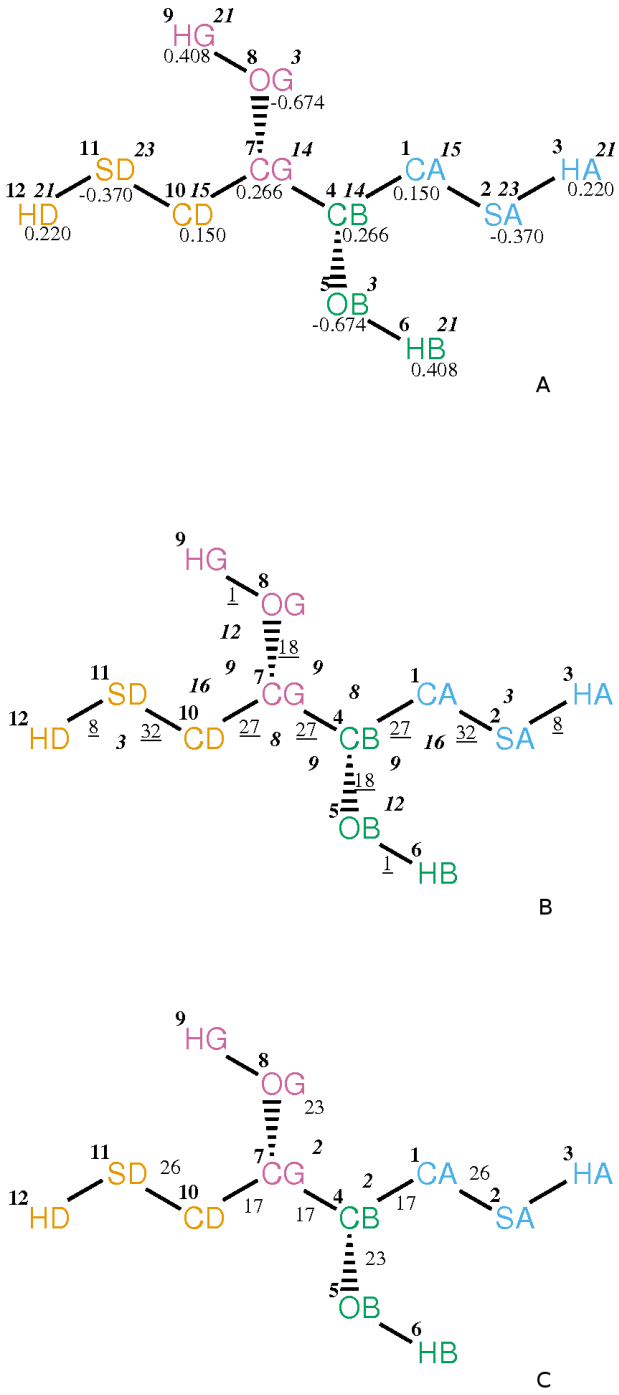


FIGURE 2.18. Force-field parameters for DTT. A) atom number (plain bold), integer atom code (italic bold), IAC charge (plain); B) atom number (plain bold), bond type (plain underlined), bond angle type (italic bold); C) atom number (plain bold), dihedral angle type (plain), improper dihedral angle type (italic bold). The charge groups are depicted in yellow, blue, green and red color.

Bibliography

[1] M.P. Allen and D.J. Tildesley. *Computer simulation of liquids.* Oxford University Press, New York, USA, 1987.

[2] X. Daura, W.F. van Gunsteren, and A.E. Mark. Folding-Unfolding Thermodynamics of a beta-Heptapeptide From Equilibrium Simulations. *Proteins*, 34:269–280, 1999.

[3] K Wuethrich, M Billeter, and W. Braun. Pseudo-structures for the 20 common amino acids for use in studies of protein conformations by measurements of intramolecular proton-proton distance constraints with nuclear magnetic resonance. *J. Mol. Biol.*, 169:949–961, 1983.

[4] T. Huber, A.E. Torda, and W.F. van Gunsteren. Local elevation: A method for improving the searching properties of molecular dynamics simulation. *J. Comput. Aided Mol. Des.*, 8:695–708, 1994.

[5] X. Daura, E. Haaksma, and W.F. van Gunsteren. Factor Xa: Simulation studies with an eye to inhibitor design. *J. Comput. Aided Mol. Des.*, 14:507–529, 2000.

[6] W.R.P. Scott, P.H. Hünenberger, I.G. Tironi, A.E. Mark, S.R. Billeter, J. Fennen, A.E. Torda, T. Huber, P. Krüger, and W.F. van Gunsteren. The GROMOS Biomolecular Simulation Program Package. *J. Phys. Chem. A*, 103:3596–3607, 1999.

[7] C. Peter, C. Oostenbrink, A. van Dorp, and W.F. van Gunsteren. Estimating entropies from molecular dynamics simulations. *J. Chem. Phys.*, 120:2652–2661, 2004.

[8] N. Hansen, J. Dolenc, M. Knecht, S. Riniker, and W.F. van Gunsteren. Assessment of enveloping distribution sampling to calculate relative free enthalpies of binding for eight netropsin-DNA duplex complexes in aqueous solution. *J. Comput. Chem.*, 33:640–651, 2012.

Index

The GROMOS Software for (Bio)Molecular Simulation

Volume 8: Installation Guide

January 9, 2021

Contents

CHAPTER 1

System requirements

GROMOS can be compiled on almost any operating system compatible with the POSIX standard[1]. Make sure that your system is powerful enough, that you have sufficient disk space, and that all required libraries are installed.

The hardware requirements are rather high for simulations, but nowadays most of the setup and analysis can be carried out on personal computers or even laptops.

Architecture: Intel or AMD x86, SUN SPARC or IBM PowerPC CPU.

Memory: This depends on the simulation you are running and the analysis you want to carry out. For most simulations and analyses you need just a few 100 MB of RAM but there are exceptions.

Diskspace: This also depends on the simulation. Typical are up to 5 GB for a big simulation. For analysis, up to 1 GB is sufficient.

Please have a look at the software requirements because these are usually not installed on an out-of-box operating system.

Operating System: POSIX compatible UNIX like Linux.

Build-Essentials: make, binutils (usually installed by default).

Compiler(C++): In principle it is possible to use any ISO C++ compiler but it is recommended to use the GNU Compiler Collection (GCC) `g++`.

Libraries(C++): There are a few of libraries that have to be installed on your system. Make sure the header files (-devel, -dev packages) are also installed. The libraries needed are:

- zlib compression library
- gsl GNU Scientific Library (see Appendix)
- for MD++: socket, nsl, fftw (3.3)

CHAPTER 2

Installation of required libraries

Some of the libraries required by GROMOS++ and MD++ are not available on standard operating systems and have to be installed manually. The installation of these libraries is discussed in this chapter. Make sure you only install the libraries that you really need for the subpart of GROMOS you want to install.

2.1. GNU scientific library

The GNU Scientific Library[2] is a C library for scientific calculations like complex number arithmetic, fast Fourier transformations, integration of functions etc. It is needed by MD++ and GROMOS++. Usually it can be installed via your operating systems package manager.

On Debian or Ubuntu Linux you can install it by typing `sudo apt-get install libgsl0 libgsl0-dev`. On Windows, install it via the CYGWIN setup.

If it is not distributed with your operating system or you are not super user you have to compile it from the source code. Fortunately this is rather easy and straightforward.

2.1.1. Installation from source. In your home create a directory where you want to install the library `/home/user/lib/gsl` and a working directory `/home/user/tmp`. Go to the GSL web page ftp://ftp.gnu.org/gnu/gsl/ and download the latest version into your working directory[1]. In a command line shell `cd /home/user/tmp` to the working directory and untar the package:

```
~/tmp> tar zxf gsl-2.6.tar.gz
~/tmp> cd gsl-2.6
```

Now you need to `./configure` for the compilation and installation. As a prefix give the directory where you want to install the library. After successful configuration make and install the GSL.

```
~/tmp/gsl-2.6> ./configure --prefix=/home/user/lib/gsl
~/tmp/gsl-2.6> make
~/tmp/gsl-2.6> make install
```

After a successful installation you can delete the working directory

```
~/tmp/gsl-2.6> cd ..
~/tmp> rm -rf gsl-2.6*
```

The GSL is now successfully installed in your home.

2.2. FFTW 3

The Fastest Fourier Transform in the West library[3] is a C library used to carry out fast Fourier transformations. It is needed by MD++. Usually the version available from package managers is too old to be of any use. Make sure that you install version 3. MD++ uses MPI parallelization and thus it is important that an MPI version of FFTW is installed.

On Debian or Ubuntu Linux you can install it by typing `sudo apt-get install libfftw3-3 libfftw3-dev`. However, this will not install the MPI version of the library.

If it is not distributed with your operating system or you are not super user you have to compile it from the source code.

[1] In the example below the version 2.6 was used but make sure the most recent version is used

2.2.1. Installation from source. In your home create a directory where you want to install the library `/home/user/lib/fftw3` and a working directory `/home/user/tmp`. Go to the FFTW web page http://www.fftw.org and download the latest version into your working directory. In a command line shell go to your working directory and untar the package:

```
~/tmp> tar zxf fftw-3.3.8.tar.gz
~/tmp> cd fftw-3.3.8
```

Then configure FFTW. In this case we want to build an MPI version of the library. Thus one has to use the correct MPI compiler wrappers and enable MPI.

```
~/tmp/fftw-3.3.8> ./configure --prefix=/home/user/lib/fftw3 \
                              --enable-moi CC=mpicc CXX=mpiCC F77=mpif77 \
                              --enable-fortran --disabled-shared
~/tmp/fftw-3.3.8> make
~/tmp/fftw-3.3.8> make install
```

Like this the normal FFTW library, an MPI version, a Fortran binding and shared libraries are installed. On clusters with multiple MPI implemenations installed it is important to use the same compiler wrappers for FFTW and for MD++.

CHAPTER 3

Installation of GROMOS

3.1. Installing MD++

MD++ is a C++ batch program for molecular simulation jobs. Because its execution time is critical you have to compile it on the machine you would like to use it to make sure that the maximum performance is obtained. Open a command line shell and find out where your home is. Then create a working and an installation directory (md++).

```
~> pwd
/home/user
~> mkdir temp
~> mkdir md++
```

Change the path `/home/user` to your home (the result of the pwd command).

Copy md++.tar.gz into your working directory. Change into the `temp` directory and unpack MD++.

```
~> cd temp
~/temp> tar zxf md++.tar.gz
~/temp> cd md++
```

In this directory now lies the source code of MD++ which is ready for compilation. The configuration is a bit tricky, because there are many options which are explained below:

1. You can specify the installation path using the `--prefix` directive.
2. If the GNU Scientific Library (gsl) was installed by the superuser `configure` will find it. If it is installed locally in your home you must specify this using the `--with-gsl` directive.
3. The same applies for the FFTW library. If it is installed locally in your home you must specify this using the `--with-fftw` directive.
4. On clusters where setup of library paths is not guaranteed, it is a good idea to link it statically. `--disable-shared` will disable the shared library and create statically linked executables.

Now you know what the options mean and you are ready to configure and run the compilation of MD++. On multicore CPU machines add the `-j` flag to `make` to boost the compilation.

```
~/temp/md++> ./configure --prefix=/path/to/install/md++ \
                         --with-gsl=/path/to/gsl \
                         --with-fftw=/path/to/fftw3 \
                         --disable-shared
~/temp/md++> make
~/temp/md++> make install
~/temp/md++> make check
~/temp/md++> touch doc/doxygen.conf.in
~/temp/md++> make doc
~/temp/md++> cp -r doc /path/to/install/md++
```

MD++ is now installed. You can test it by typing

```
~/temp/gromosxx> /path/to/install/md++/bin/md
```

If it prints the usage everything is fine.

Clean up the working directory because the static builds need a lot of diskspace and are not needed anymore.

```
~/temp/md++> cd ..
~/temp> rm -rf md++
```

3.1.1. Debug version of MD++. MD++ allows you to compile a special version for debugging. This version has additional debug statements and the code is not optimised by the compiler. You can enable debugging by giving `--enable-debug` as an argument to `configure`. Using this special version you can specify level of debug information you are interested in. See Sec. 6-1.1.2 for details.

3.1.2. Parallel version of MD++. In order to enable MD++ to use the full power of your computer's hardware you have to compile a parallel version. MD++ knows two kinds of parallelization:

OpenMP: This parallelization is straightforward and enables MD++ to run on multiple core CPUs, like all recent x86 CPUs are. No additional software is required. You can enable this by adding `--enable-openmp` to configure.

MPI: MPI is used for parallelization when no shared memory is available: the different CPUs you want to use for the calculations are located in different machines. This is the case in computer clusters.

To compile the MPI version you have to use a special set of compilers wrappers, which know which MPI version and implementation you have on the cluster. In general these compilers are called `mpicc` for the C compiler and `mpiCC` for the C++ compiler. You have to tell `configure` to use these compilers and to `--enable-mpi`:

```
~/temp/md++> ./configure CC=mpicc CXX=mpiCC \
                         --enable-mpi \
                         --disable-shared \
                         --with-gsl=/path/to/gsl \
                         --with-fftw=/path/to/fftw3 \
                         --prefix=/home/user/md++
```

You have to make sure that the binary of the FFTW library was compiled using the same compiler wrappers and is linked to the same MPI libraries. This can be achieved by compiling an own version of FFTW with MPI enabled.

```
~/temp/fftw-3.3.8> ./configure --enable-mpi CC=mpicc CXX=mpiCC F77=mpif77 \
                               --enable-fortran \
                               --prefix=/path/to/install/fftw3_mpi
~/temp/fftw-3.3.8> make
~/temp/fftw-3.3.8> make install
```

After successful configuration just `make` and `make install` it as usual. If the test call

```
~/temp/md++> /home/user/md++/md/md_mpi
```

does not tell you to enable MPI, everything is fine.

3.1.3. Compiling MD++ using the CUDA solvent-solvent interaction evaluation acceleration. In order to make use of the CUDA solvent-solvent interaction evaluation acceleration library (*cukernel*) MD++ has to be compiled using an additional path pointing to the directory containing the CUDA libraries and header file.

```
~/temp/md++> ./configure --disable-shared \
                         --with-gsl=/path/to/gsl \
                         --with-fftw=/path/to/fftw3 \
                         --with-cuda=/path/to/cuda \
                         --prefix=/home/user/md++
```

If necessary the appropriate compiler and flags can be set by adding the appropriate variables, e.g.:

```
NVCC=nvcc
NVCCFLAGS='-arch sm_30'
NVCC_CFLAGS='-O2 -D DNDEBUG -lcuda -lcudart '
```

3.1.4. What is installed. After successful compilation,

1. the program binaries are in `bin/`. If you used `--prefix` option, you will find `bin/` there. Note that for MPI support you should use the binary `md_mpi` (or `repex_mpi` for replica exchange).
2. in the `include/` and `lib/` subdirectories are the files needed for programming with MD++.

3.2. Installing GROMOS++

GROMOS++ is a collection of command line programs needed to setup a simulation or to analyze the results of a simulation. In order to use these programs you have to compile and install them on your machine. Open a command line shell and find out where your home is.

```
~> pwd
/home/nschmid
```

Change the path `/home/user` to your home (the result of the pwd command).

Unpack GROMOS++.

```
~> tar xfz gromos++.tar.gz
~> cd gromos++
```

You are now in the GROMOS++ source directory and can start to configure and make GROMOS++. Tell configure where it has to look for the GNU Scientific Library (gsl) and the Fastest Fourier Transform in the West library (fftw) using the `--with-gsl` and `--with-fftw` directives. Usually these libraries are installed in /usr/local and configure will find it without telling, but if you have installed them in your home you have to tell configure using the `--with-gsl` and `--with-fftw`directives. Some programs in GROMOS++ make use of algorithms of MD++. In order to use these programs one has to specify the location of the MD++ libraries and header files using the `--with-mdpp` directive. Debugging should be disabled in order to make use of compiler optimizations. Some operating systems require static linking which can be controlled by the `--disable-shared` directive. A few computationally demanding programs can be run in parallel on shared memory machines using OpenMP. OpenMP can be enabled by the `--enable-openmp` directive. On multicore CPU machines add the `-j` flag to `make` to speed up the compilation.

```
~/gromos++> ./configure --with-gsl=/path/to/gsl --with-fftw=/path/to/fftw
~/gromos++> make
~/gromos++> make install
```

3.2.1. Generating the documentation. If doxygen is installed on your machine you can generate documentation directly from the source code. Go to the gromos++ directory and type

```
~/gromos++> touch doc/doxygen.conf.in
~/gromos++> make doc
```

In the `doc` directory you will find html documentation. Open a web browser and open `file:///home/<username>/<path to gromos++>/gromos++/doc/html/index.html`. Under `available` you find the documentation of the various GROMOS++ programs.

After the successful installation you should clean up the working directory.

```
~/gromos++> make clean
```

3.2.2. Adding it to the path. In order to use the programs without specifying the full path you can add them to your PATH veriable. Add the following two lines to your `~/.bashrc`:

```
export PATH="${PATH}:/path/to/gromos++/bin"
export LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:/path/to/gromos++/lib"
```

3.2.3. What is installed. After successful compilation,

1. the program binaries are in `bin/`. If you used `--prefix` option, you will find `bin/` there.
2. in the `include/` and `lib/` subdirectories are the files needed for programming with GROMOS++
3. `share/` is home of useful files (called libraries) and examples.

Bibliography

[1] IEEE Standards Department. *IEEE 1003.1-2008, Standard for Information Technology - Portable Operating System Interface (POSIX®)*. Institute of Electrical and Electronics Engineers (IEEE), 2008.

[2] M. Galassi, J. Daviesm, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi. *Gnu Scientific Library: Reference Manual*. Network Theory Ltd., 2003.

[3] F. Matteo and S. G. Johnson. The Design and Implementation of FFTW3. *Proceedings of the IEEE*, 93:216–231, 2005.

Index

The GROMOS Software for (Bio)Molecular Simulation

Volume 9: Index

January 9, 2021

Contents

VOLUME 1

VOLUME 2

VOLUME 3

VOLUME 4

VOLUME 5

VOLUME 6

VOLUME 7

VOLUME 8

Index