Outline

1 Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2 Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Outline

1. Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2. Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Introduction

You came up with a mathematical model of a system, which contains some parameters (ex: mass, elasticity, specific heat,...).

⇒ Now you need to run experiments to identify the model parameters.

How to proceed?
Introduction

Least Squares Methods:

- **Classical LS methods** for *static* and *linear* systems \(\Rightarrow\) closed-form solutions available.
- **Nonlinear LS methods** for *dynamic* and *nonlinear* systems \(\Rightarrow\) only numerical (optimization) solutions available.

**Remark:** there are closed-form approaches for *linear* *dynamic* systems as well. See master-level courses (e.g. “Introduction to Recursive Filtering and Estimation”).
Planning Experiments

Planning experiments is about knowing:

- What excitation of the system: choice of correct input signals
- What to measure in the system (choice of sensors, their location, etc.)
- Measurements for linear or nonlinear model identification
- Frequency content of the excitation signals
- Noise level at input and output of the system
- Safety issues

are best to efficiently identify the system parameters.

Choose “signals such that all the relevant dynamics and static effects inside the plant are excited with the correct amount of input energy.” in p. 75 - 76 script.
The data obtained experimentally may be used for two purposes:

1. **To identify** unknown system structures and system parameters. Using a first set of data: $u_1, y_{r,1}$

2. **To validate** the results of the system modeling and parameter identification. Using a second set of data: $u_2, y_{r,2}$
A word of caution: It is of fundamental importance *not* to use the same data set for both purposes.

The real quality of a parameterized model may only be assessed by:

- comparing the prediction of that model
- with measurement data that have not been used in the model parametrization.

Remark: the model and its identification are validated if: for the same input signal $u_2$, the output signals $y_{r,2}$ and $y_m$ are sufficiently “similar”.
Outline

1 Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2 Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Least Squares estimation

is used to fit the parameters of a **linear** and **static** model (model that mathematically describes inputs/outputs of the system).

The model is never exact: ⇒ for the same input signals there will be a difference between the outputs of the model, and true system outputs ⇒ modeling errors.

**Remark:** These errors may be considered a deterministic or stochastic variables. Both formulations are equivalent, as long as these errors are completely unpredictable and not correlated with the inputs.
It is assumed that the output of the real system may be approximated by the output of the system’s model with some model error $e$ according to the linear equation:

$$y(k) = h^T(u(k)) \cdot \pi + e(k)$$

with: $k \in [1, \ldots, r]$: discrete-time instant
$y(k) = h^T(u(k)) \cdot \pi + e(k)$

$k$: represents the index of discrete time (discrete-time instant $k$)

$u(k) \in \mathbb{R}^m$ input vector

$y(k) \in \mathbb{R}$ is the output signal (measurement) (scalar).

$\pi \in \mathbb{R}^q$ is the vector of the $q$ unknown parameters (those we want to estimate).

$h(.) \in \mathbb{R}^q$ is the regressor, depends on $u$ in a nonlinear but algebraic way.

$e(k)$ is the error (scalar).

Typically, there are more measurements than unknown parameters: $(r \gg q)$. 
LS Objective:
Estimate $\pi \in \mathbb{R}^q$ is the vector of unknown parameters (those we want to estimate) such that the model error $e$ is minimized.

In order to do that, let’s formulate the problem into a matrix form (derived on the blackboard during the class).

+ Example.
Outline

1. Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2. Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Least-square solution and comments

\[ \pi_{LS} = \left[ H^T \cdot W \cdot H \right]^{-1} H^T \cdot W \cdot \tilde{y} \]

The regression matrix \( H \) must have full column rank, i.e., all \( q \) parameters \((\pi_1, \pi_2, \ldots, \pi_q)\) are required to explain the data.

Moore-Penrose inverse:

\[ M^\dagger = (M^T \cdot M)^{-1} \cdot M^T, \]
\[ M \in \mathbb{R}^{r \times q}, \ r > q, \ \text{rank}\{M\} = q \]
If the error $e$ is an uncorrelated white noise signal with:
- mean value 0
- and variance $\sigma$,

then

1. the expected value of the parameter estimation $\pi_{LS}$ is equal to its true value,\(^a\) $E(\pi_{LS}) = \pi_{true}$
2. covariance matrix: $\Sigma = \sigma^2 \cdot (H^T \cdot W \cdot H)^{-1}$.

\(^a\)Of course, only if the model perfectly describes the true system.
Particular case: $q = 2$, $r = 3$

The result of the LS identification can be geometrically interpreted: the columns of $H$ define the directions (projection vectors) that define a plane (defined by the 2 vectors in this case: $\tilde{h}_1$, $\tilde{h}_2$) and therefore, $\tilde{e}_{LS}$ is perpendicular to that plane.

$$\tilde{y} = H \pi_{LS} + \tilde{e}_{LS}$$

$$\tilde{y} = [\tilde{h}_1 \quad \tilde{h}_2] \begin{bmatrix} \pi_{LS,1} \\ \pi_{LS,2} \end{bmatrix} + \tilde{e}_{LS}$$

$$\tilde{y} = \pi_{LS,1} \cdot \tilde{h}_1 + \pi_{LS,2} \cdot \tilde{h}_2 + \tilde{e}_{LS}$$
Model Parametrization
Iterative Least Squares
Planning Experiments
Least Squares Methods for Linear Systems
Solution of the Least Squares Problem

\[ \tilde{y} = \tilde{h}_1 + \tilde{h}_2 + \tilde{e}_{LS} \]

\[ \pi_{LS,1} \cdot \tilde{h}_1 \]

\[ \pi_{LS,2} \cdot \tilde{h}_2 \]
Outline

1 Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2 Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Iterative Least Squares: Problem Definition

Up to now, a **batch-like approach**\(^1\) has been assumed:

\[
\pi_{LS} = \left[ H^T \cdot W \cdot H \right]^{-1} H^T \cdot W \cdot \tilde{y}
\]

Problems:

1. The computation the **matrix inversion** part is the most time-consuming step.
2. Assuming that
   - \( r \) measurements have been taken,
   - a solution has been computed,

   \( \Rightarrow \) numerically very inefficient to repeat the full matrix inversion procedure when an additional measurement data becomes available.

---

\(^1\)Batch-like approach: 1. all measurement are made, 2. data are organized in the LS pb formulation, 3. the LS solution is computed once.
Instead, an *iterative* solution of the form

\[ \pi_{LS}(r + 1) = f(\pi_{LS}(r), y(r + 1)), \]

initialized by

\[ \pi_{LS}(0) = E\{\pi\} \]

would be much more efficient.

How do we build up a recursive Least-Squares algorithm?
Recursive LS Formulation

1. Start:

\[ \pi_{LS} = \left( H^T \cdot W \cdot H \right)^{-1} H^T \cdot W \cdot \tilde{y} \]
Recursive LS Formulation

1. **Start:**

   \[
   \pi_{LS} = [H^T \cdot W \cdot H]^{-1} H^T \cdot W \cdot \tilde{y}
   \]

2. **Simplification:** consider weighting matrix simply as \( W = I \) (extension with \( W \) easily possible).

   \[
   \pi_{LS} = [H^T \cdot H]^{-1} H^T \cdot \tilde{y}
   \]
Recursive LS Formulation

1. **Start:**
   \[ \pi_{LS} = [H^T \cdot W \cdot H]^{-1} H^T \cdot W \cdot \tilde{y} \]

2. **Simplification:** consider weighting matrix simply as \( W = I \) (extension with \( W \) easily possible).
   \[ \pi_{LS} = [H^T \cdot H]^{-1} H^T \cdot \tilde{y} \]

3. **Formulate matrix products as sums:**
   \[ \pi_{LS}(r) = \left[ \sum_{k=1}^{r} h(k) \cdot h^T(k) \right]^{-1} \cdot \sum_{k=1}^{r} h(k) \cdot y(k) \]
Recursive LS Formulation

1. Start:
\[ \pi_{LS} = \left[H^T \cdot W \cdot H\right]^{-1}H^T \cdot W \cdot \tilde{y} \]

2. Simplification: consider weighting matrix simply as \( W = I \)
   (extension with \( W \) easily possible).
\[ \pi_{LS} = \left[H^T \cdot H\right]^{-1}H^T \cdot \tilde{y} \]

3. Formulate matrix products as sums:
\[ \pi_{LS}(r) = \left[\sum_{k=1}^{r} h(k) \cdot h^T(k)\right]^{-1} \cdot \sum_{k=1}^{r} h(k) \cdot y(k) \]

4. Use matrix inversion Lemma
Suppose

- $M \in \mathbb{R}^{n \times n}$ is a regular matrix ($\det(M) \neq 0$),
- and $\mathbf{v} \in \mathbb{R}^n$ is a column vector,

which satisfies the condition: $1 + \mathbf{v}^T \cdot M^{-1} \cdot \mathbf{v} \neq 0$.

In this case:

$$[M + \mathbf{v} \cdot \mathbf{v}^T]^{-1} = M^{-1} - \frac{1}{1 + \mathbf{v}^T \cdot M^{-1} \cdot \mathbf{v}} \cdot M^{-1} \cdot \mathbf{v} \cdot \mathbf{v}^T \cdot M^{-1}$$

Remarks

- Proof by inspection: multiply from the left with $M + \mathbf{v} \cdot \mathbf{v}^T$.
- Main advantage of this lemma: no additional matrix inversion than $M^{-1}$ is needed.

Inversion of the new matrix $M + \mathbf{v} \cdot \mathbf{v}^T$ may be carried out very efficiently.
\[ \pi_{LS}(r) = \left[ \sum_{k=1}^{r} h(k) \cdot h^T(k) \right]^{-1} \cdot \sum_{k=1}^{r} h(k) \cdot y(k) \]

To simplify the notation, a matrix \( \Omega \) is defined as:

\[ \Omega(r) = \left[ \sum_{k=1}^{r} h(k) \cdot h^T(k) \right]^{-1} \]

Then compute \( \Omega(r + 1) \):

\[ \Omega(r + 1) = \left[ \sum_{k=1}^{r+1} h(k) \cdot h^T(k) \right]^{-1} = \left[ \sum_{k=1}^{r} h(k) \cdot h^T(k) + h(r + 1) \cdot h^T(r + 1) \right]^{-1} \]
\[ \Omega(r + 1) = \left[ \sum_{k=1}^{r} h(k) \cdot h^T(k) + h(r + 1) \cdot h^T(r + 1) \right]^{-1} \]

we use the Inversion Lemma:

\[ [M + v \cdot v^T]^{-1} = M^{-1} - \frac{1}{1 + v^T \cdot M^{-1} \cdot v} \cdot M^{-1} \cdot v \cdot v^T \cdot M^{-1} \]

Recursive formulation of the matrix inverse

\[ \Omega(r + 1) = \Omega(r) - \frac{1}{1 + c(r + 1)} \cdot \Omega(r) \cdot h(r + 1) \cdot h^T(r + 1) \cdot \Omega(r) \]

where \( c(r + 1) = h^T(r + 1) \cdot \Omega(r) \cdot h(r + 1) \) (scalar).
\[ \pi_{LS}(r) = \Omega(r) \cdot \sum_{k=1}^{r} h(k) \cdot y(k) \]

How to compute recursively the estimate?

\[ \pi_{LS}(r + 1) = \Omega(r + 1) \cdot \sum_{k=1}^{r+1} h(k) \cdot y(k) \]

\[ = \left[ \Omega(r) - \frac{1}{1 + c(r + 1)} \Omega(r) h(r + 1) h^T(r + 1) \Omega(r) \right] \]

\[ \cdot \left( \sum_{k=1}^{r} h(k) \cdot y(k) + h(r + 1) \cdot y(r + 1) \right) \]
\[ \pi_{LS}(r+1) = \Omega(r) \cdot \sum_{k=1}^{r} h(k) \cdot y(k) + \Omega(r) \cdot h(r+1) \cdot y(r+1) \]

\[ \pi_{LS}(r) \]

\[ - \frac{1}{1 + c(r+1)} \Omega(r) h(r+1) \cdot h^T(r+1) \Omega(r) \cdot \sum_{k=1}^{r} h(k) \cdot y(k) \]

\[ \pi_{LS}(r) \]

and

\[ - \frac{c(r+1)}{1 + c(r+1)} = \frac{1}{1 + c(r+1)} - 1 \]
\[
\pi_{LS}(r+1) = \pi_{LS}(r) + \Omega(r) \cdot h(r+1) \cdot y(r+1) \\
- \frac{1}{1 + c(r+1)} \Omega(r) h(r+1) h^T(r+1) \pi_{LS}(r) \\
+ \left( \frac{1}{1 + c(r+1)} - 1 \right) \Omega(r) h(r+1) \cdot y(r+1)
\]
Recursive computation of the parameter vector $\pi_{LS}(r)$

$$
\pi_{LS}(r+1) = \pi_{LS}(r) + \frac{1}{1 + c_{(r+1)}} \Omega(r) h_{(r+1)} \left( y_{(r+1)} - h_{r+1}^T \pi_{LS}(r) \right)
$$

with

Recursive update of the gain matrix $\Omega$

$$
\Omega_{(r+1)} = \Omega_{(r)} - \frac{1}{1 + c_{(r+1)}} \cdot \Omega_{(r)} \cdot h_{(r+1)} \cdot h_{(r+1)}^T \cdot \Omega_{(r)}
$$

where $c_{(r+1)} = h_{(r+1)}^T \cdot \Omega_{(r)} \cdot h_{(r+1)}$ (scalar).

and

Initialization

$\pi_{LS}(0), \Omega(0)$
Iterative Least Squares

Problem Definition
Least Squares with Exponential Forgetting
Simplified Recursive LS Algorithm

Recursive computation of the parameter vector $\pi_{LS}(r)$

$$
\pi_{LS}(r+1) = \pi_{LS}(r) + \frac{1}{1 + c(r+1)} \Omega(r) h(r+1) \left( y(r+1) - h^T(r+1) \pi_{LS}(r) \right)
$$

can be rewritten as:

$$
\pi_{LS}(r+1) = \pi_{LS}(r) + \delta(r + 1) \left( y(r+1) - h^T(r+1) \pi_{LS}(r) \right)
$$

Comments on the recursive formulation:

- The blue term is a vector indicating the *correction direction*: $\delta(r + 1)$ applied by the *innovation term* (or *prediction error*).
- Interesting to note that the *correction direction* is not dependent on the magnitude of the *prediction error*. 
Outline

1 Model Parametrization
   • Planning Experiments
   • Least Squares Methods for Linear Systems
   • Solution of the Least Squares Problem

2 Iterative Least Squares
   • Problem Definition
   • Least Squares with Exponential Forgetting
   • Simplified Recursive LS Algorithm
Exponential Forgetting

New error weighting for the recursive case

\[ \epsilon(r) = \sum_{k=1}^{r} \lambda^{r-k} \cdot [y(k) - h^T(k) \cdot \pi_{LS}(k)]^2, \quad \lambda < 1 \]

- This introduces an “exponential forgetting” process: older errors have a smaller influence on the result of the parameter estimation.
- Can cope with slowly varying parameters.

Update equations

\[
\begin{align*}
\pi_{LS}(r+1) &= \pi_{LS}(r) + \frac{1}{\lambda + c(r+1)} \Omega(r) h(r+1) \left[ y(r+1) - h^T(r+1) \pi_{LS}(r) \right] \\
\Omega(r+1) &= \frac{1}{\lambda} \Omega(r) \left[ I - \frac{1}{\lambda + c(r+1)} h(r+1) h^T(r+1) \Omega(r) \right]
\end{align*}
\]
Outline

1 Model Parametrization
   - Planning Experiments
   - Least Squares Methods for Linear Systems
   - Solution of the Least Squares Problem

2 Iterative Least Squares
   - Problem Definition
   - Least Squares with Exponential Forgetting
   - Simplified Recursive LS Algorithm
Kaczmarz’s projection algorithm

Each new prediction error: \( e_{(r+1)} = y_{(r+1)} - h_{(r+1)}^T \cdot \pi_{(r)} \) contains new information on the parameters \( \pi \) only in the direction of \( h_{(r+1)} \). Therefore, \( \pi_{(r+1)} \) is sought, which requires the smallest possible change \( \pi_{(r+1)} - \pi_{(r)} \) to explain the new observation.

Cost function to minimize:

\[
J(\pi) = \frac{1}{2} \cdot \left[ \pi_{(r+1)} - \pi_{(r)} \right]^T \cdot \left( \pi_{(r+1)} - \pi_{(r)} \right) + \mu \cdot \left[ y_{(r+1)} - h_{(r+1)}^T \cdot \pi_{(r+1)} \right]
\]

Necessary conditions for the minimum:

\[
\frac{\partial J}{\partial \pi_{(r+1)}} = 0 \quad \frac{\partial J}{\partial \mu} = 0
\]
Simplified Recursive LS Algorithm

Solve this linear equations for $\pi(r+1)$ and $\mu$

$$\pi(r+1) = \pi(r) + \frac{h(r+1)}{h^T(r+1) \cdot h(r+1)} \cdot [y(r+1) - h^T_{(r+1)} \cdot \pi(r)]$$

Usually this solution is modified as

$$\pi(r+1) = \pi(r) + \frac{\gamma \cdot h(r+1)}{\lambda + h^T_{(r+1)} \cdot h(r+1)} \cdot [y(r+1) - h^T_{(r+1)} \cdot \pi(r)]$$

$0 < \gamma < 2$, $0 < \lambda < 1$ to achieve desired convergence and forgetting.

Discussions

- Kaczmarz’ projection algorithm requires less computational efforts than regular LS
- It converges much slower than regular LS algorithm.
Next lecture + Upcoming Exercise

Next lecture
- Stability Analysis
- Properties of Linear Systems

Next exercises:
- Least squares
- Parameter identification