Numerical optimization

Mines Nancy – Fall 2024

session 3 – least squares problems

Lecturer: Julien Flamant (CNRS, CRAN)

julien.flamant@univ-lorraine.fr

Office 425, FST 1er cycle

Course material:

arche.univ-lorraine.fr/course/view.php?id=74098

github.com/jflamant/mines-nancy-fall24-optimization

Outline

In this session

We'll study a particular optimization problem known as least squares.

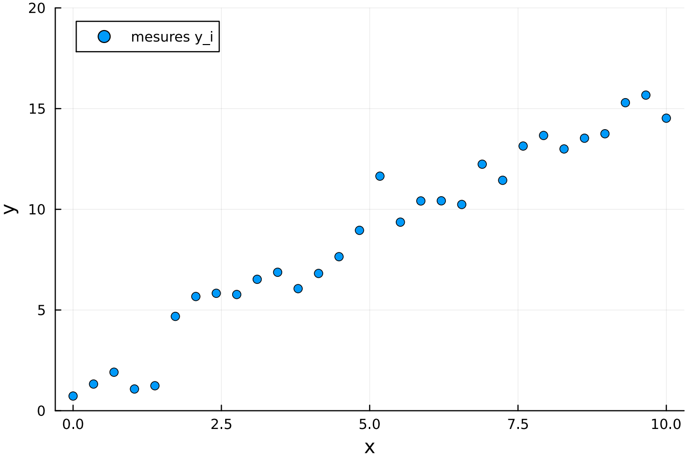
It is a very important problem because:

- it is often the first approach used, e.g., in inverse problems;
- it can be derived statistically from a Gaussian noise assumption;
- it lays the foundation for many more sophisticated procedures;
- under some conditions, it admits an explicit (closed-form) solution;

What we'll see:

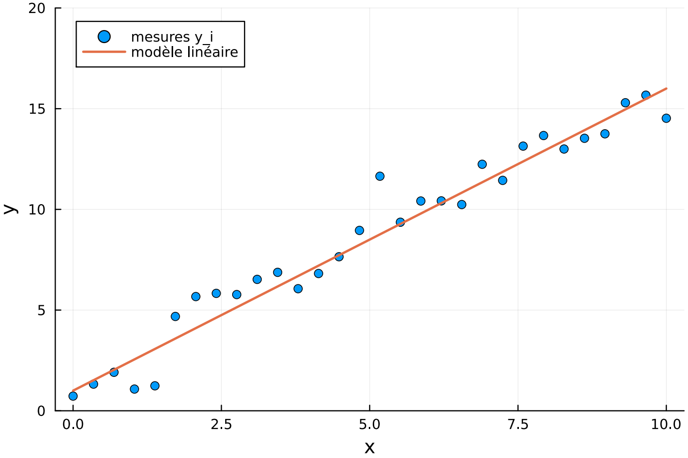
- several use-case of (linear) least squares problems;
- conditions for uniqueness of solutions;
- how to compute an explicit solution;
- **TP**: practical implementation of least-squares problems, with an introduction to the notion of regularization.

A first example: linear regression



we seek a simple relationship between
the input (explanatory) variables x_i and the data (or measurements) y_i

A first example: linear regression



linear model $y_i \approx \beta_0 + \beta_1 x_i$

where (β_0, β_1) are the model parameters to be estimated

Solution by least squares

Let's construct an objective function in the parameters (β_0, β_1)

Model error for fixed i: $\varepsilon_i(\beta_0, \beta_1) = (y_i - \beta_0 - \beta_1 x_i)^2$

Solution by least squares

Let's construct an objective function in the parameters (β_0, β_1)

Model error for fixed i: $\varepsilon_i(\beta_0, \beta_1) = (y_i - \beta_0 - \beta_1 x_i)^2$

Total model error: $\varepsilon(\beta_0, \beta_1) = \sum_{i=1}^{N}(y_i - \beta_0 - \beta_1 x_i)^2$

Solution by least squares

Let's construct an objective function in the parameters (β_0, β_1)

Model error for fixed i: $\varepsilon_i(\beta_0, \beta_1) = (y_i - \beta_0 - \beta_1 x_i)^2$

Total model error: $\varepsilon(\beta_0, \beta_1) = \sum_{i=1}^N (y_i - \beta_0 - \beta_1 x_i)^2$

For a fixed data vector $\mathbf{y} = (y_1, \dots, y_N)^\top$ and known explanatory variables $\mathbf{x} = (x_1, \dots, x_N)$, we obtain the following unconstrained optimization problem:

$$\min_{\beta_0, \beta_1 \in \mathbb{R}} \sum_{i=1}^N (y_i - \beta_0 - \beta_1 x_i)^2$$

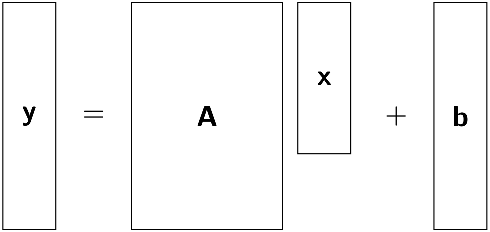
This is a linear least squares problem.

This type of problem is found in many applications, such as engineering, econometrics, image processing, etc.

Outline

General form: linear least squares

Hypothesis: linear relationship between data $\mathbf{y}$ and the variables $\mathbf{x}$.



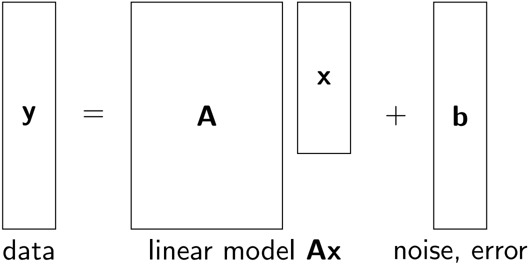
data linear combination $\mathbf{Ax}$ noise, error

Definitions and notations

- the vector $\mathbf{y} \in \mathbb{R}^M$ contains the data (which we want to model)
- the matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$ encodes the model (assumed to be known)
- the vector $\mathbf{x} \in \mathbb{R}^N$ represents the unknowns of the problem
- the vector $\mathbf{b} \in \mathbb{R}^M$ expresses the errors (due to the model, noise, etc.)

General form: linear least squares

Hypothesis: linear relationship between the data $\mathbf{y}$ and the variables $\mathbf{x}$



Linear least squares problem:

$$\min_{\mathbf{x} \in \mathbb{R}^N} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$$

where $\mathbf{y} \in \mathbb{R}^M$ and $\mathbf{A} \in \mathbb{R}^{M \times N}$. When a solution exists, we write $\hat{\mathbf{x}}$ s.t.

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x} \in \mathbb{R}^N} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$$

Least squares as an unconstrained quadratic program

The objective function $f(\mathbf{x}) = \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$ can be rewritten as

$$\begin{aligned} f(\mathbf{x}) &= (\mathbf{y} - \mathbf{A}\mathbf{x})^\top(\mathbf{y} - \mathbf{A}\mathbf{x}) \\ &= \mathbf{y}^\top\mathbf{y} - \mathbf{x}^\top\mathbf{A}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{A}\mathbf{x} + \mathbf{x}^\top\mathbf{A}^\top\mathbf{A}\mathbf{x} \end{aligned}$$

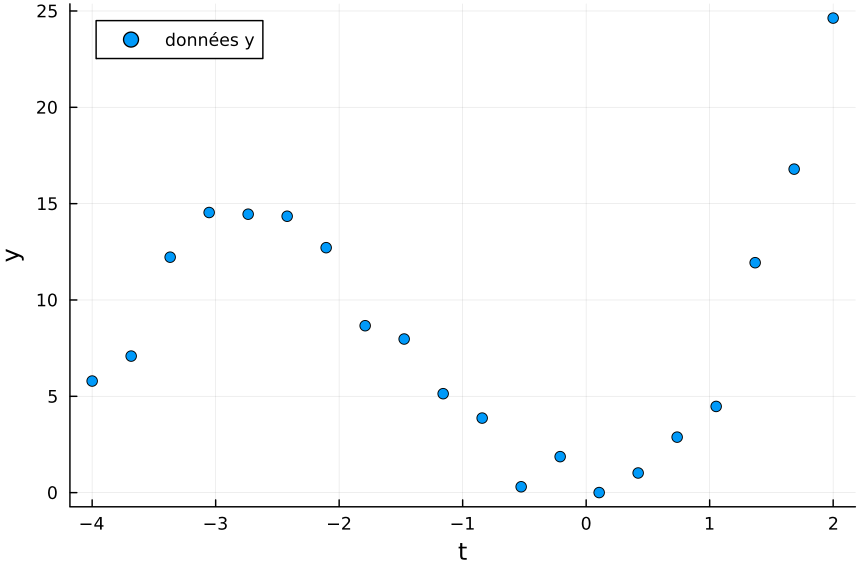
such that the least-squares optimization problem can be equivalently reformulated as the unconstrained quadratic program

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x} - \mathbf{p}^\top\mathbf{x}$$

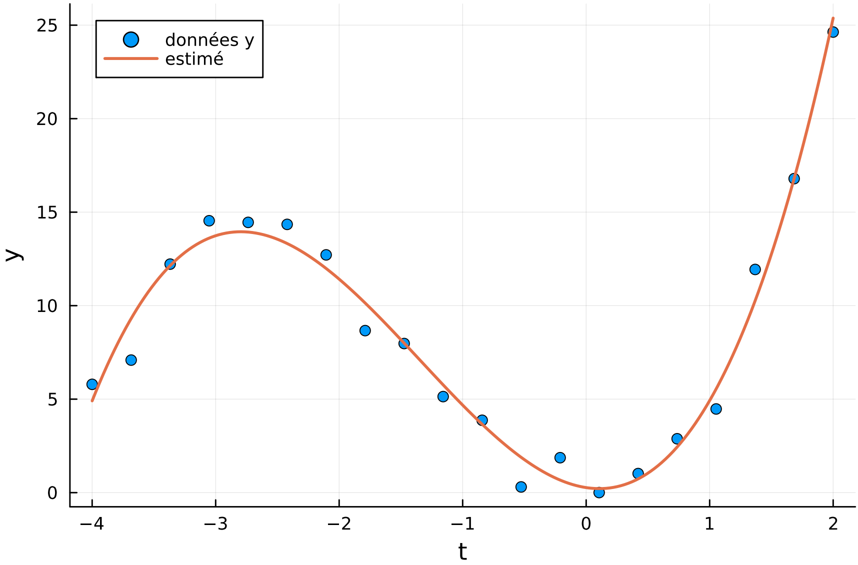
where $\mathbf{Q} := \mathbf{A}^\top\mathbf{A} \in \mathbb{R}^{N\times N}$ and $\mathbf{p} = \mathbf{A}^\top\mathbf{y} \in \mathbb{R}^N$.

the interpretation of LS as a QP will be useful later on to study the solutions of least square problems

TP example: polynomial regression



TP example: polynomial regression



Example 1: polynomial regression

The degree of the polynomial is fixed (or known) (here $d = 3$).

Polynomial model $$y_m(t) = x_3 t^3 + x_2 t^2 + x_1 t + x_0$$

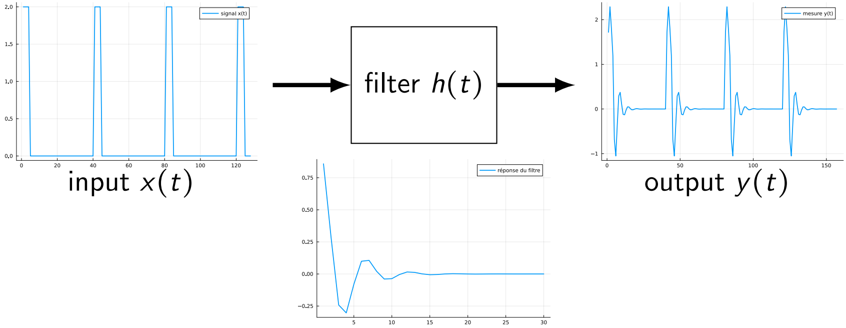
Measurements $$y_i = y_m(t_i) + b_i \text{ for known } t_1, \dots, t_M$$

Setting up the equations

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 1 & t_2 & t_2^2 & t_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_M & t_M^2 & t_M^3 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

The vector $\hat{\mathbf{x}}$ gives the least squares estimation of the coefficients of the polynomial $y_m(t)$.

Another example: deconvolution



Objective

given $h(t)$, recover $x(t)$ from $y(t) = (h * x)(t) + b(t)$

Example 2: deconvolution

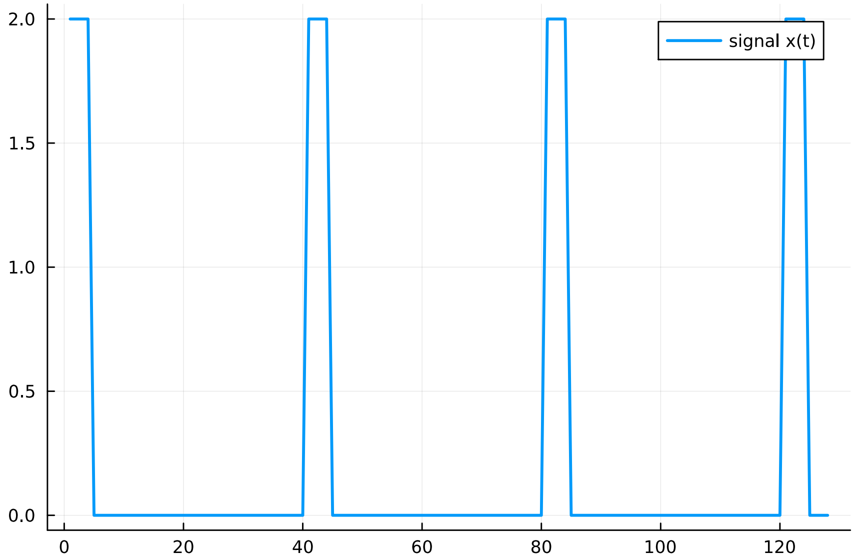
Discrete-time variables $\mathbf{y} \in \mathbb{R}^M, \mathbf{h} \in \mathbb{R}^L, \mathbf{x} \in \mathbb{R}^N$, $M = N + L - 1$.

Discrete-time convolution product
$$y_m = \sum_{\ell=1}^{L} h_\ell x_{m-\ell+1}$$

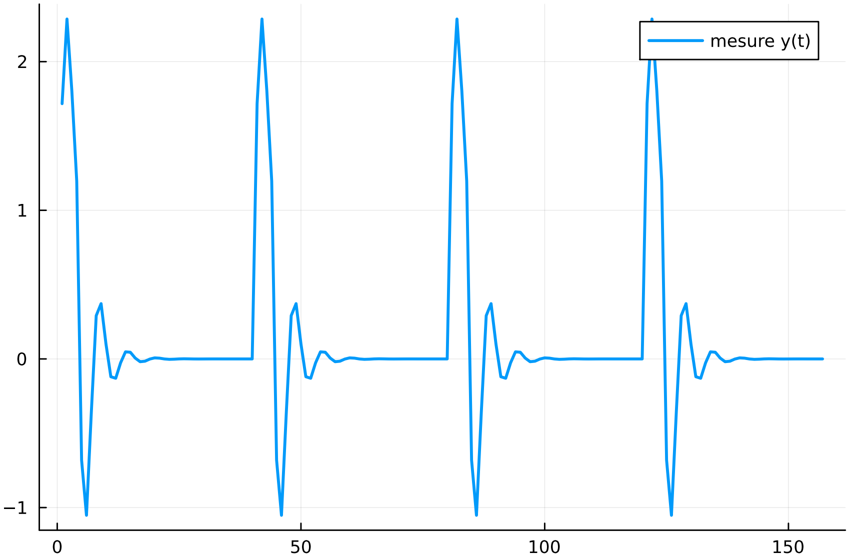
Setting up the equations

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{L-1} \\ \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} h_1 & 0 & \cdots & 0 & 0 \\ h_2 & h_1 & & \vdots & \vdots \\ h_3 & h_2 & \cdots & 0 & 0 \\ \vdots & h_3 & \cdots & h_1 & 0 \\ h_{L-1} & \vdots & \ddots & h_2 & h_1 \\ h_L & h_{L-1} & & \vdots & h_2 \\ 0 & h_L & \ddots & h_{L-2} & \vdots \\ 0 & 0 & \cdots & h_{L-1} & h_{L-2} \\ \vdots & \vdots & & h_L & h_{L-1} \\ 0 & 0 & 0 & \cdots & h_L \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

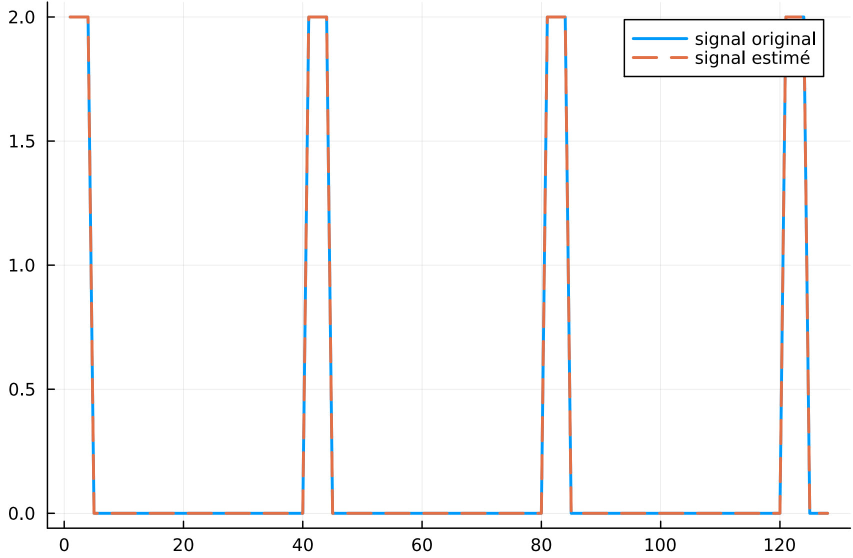
Example 2: deconvolution



Example 2: deconvolution



Example 2: deconvolution



Outline

Unconstrained quadratic programs

Let us study the unconstrained quadratic program (QP)

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{p}^\top \mathbf{x}$$

where $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is symmetric and $\mathbf{p} \in \mathbb{R}^N$ is arbitrary.

Questions

- Existence of solutions?
- Uniqueness of solutions?
- Expression of solutions?

Review: vector derivatives formulas

Recall that for a real-valued function $f : \mathbb{R}^N \to \mathbb{R}$, we have the following

$$\nabla f(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \qquad \nabla^2 f(\mathbf{x}) = \frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x} \partial \mathbf{x}^\top}$$

Key identities to remember [matrixcookbook]

$$\frac{\partial \mathbf{x}^\top \mathbf{a}}{\partial \mathbf{x}} = \frac{\partial \mathbf{a}^\top \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a} \qquad \frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{A}$$

$$\frac{\partial \mathbf{x}^\top \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \left(\mathbf{A} + \mathbf{A}^\top\right)\mathbf{x} \qquad \frac{\partial^2 \mathbf{x}^\top \mathbf{A}\mathbf{x}}{\partial \mathbf{x} \partial \mathbf{x}^\top} = \mathbf{A} + \mathbf{A}^\top$$

it is better to know how to prove these results!

Back to unconstrained quadratic programming

Unconstrained QP

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x} - \mathbf{p}^\top\mathbf{x}$$

where $\mathbf{Q}\in\mathbb{R}^{N\times N}$ is symmetric and $\mathbf{p}\in\mathbb{R}^N$ is arbitrary.

- Compute the gradient and Hessian of the objective function:

$$\nabla f(\mathbf{x}) = \mathbf{Q}\mathbf{x} - \mathbf{p}, \qquad \nabla^2 f(\mathbf{x}) = \mathbf{Q}$$

- Since $\mathbf{Q}$ is symmetric, the spectral theorem shows $\mathbf{Q}$ is diagonalizable by an orthogonal matrix U such that

$$\mathbf{Q} = \mathbf{U}^\top\mathbf{D}\mathbf{U}, \text{ where } \mathbf{D} = \begin{bmatrix} \lambda_1 & & (0) \\ & \ddots & \\ (0) & & \lambda_N \end{bmatrix} \quad \text{with } \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$$

Solutions of unconstrained QP (I)

Unconstrained QP

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x} - \mathbf{p}^\top\mathbf{x}$$

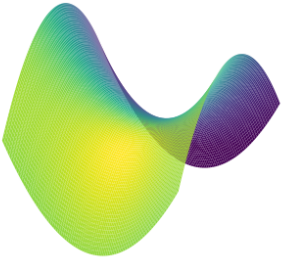
where $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is symmetric and $\mathbf{p} \in \mathbb{R}^N$ is arbitrary.

First case: $\lambda_1 = \lambda_{\min}(\mathbf{Q}) < 0$

Let $\mathbf{u}_1 \in \mathbb{R}^N$ be the eigenvector associated to λ_1. Then for $z \in \mathbb{R}$,

$$f(z\mathbf{u}_1) = \frac{\lambda_1}{2}z^2 - z\mathbf{p}^\top\mathbf{u}_1 \xrightarrow[|z|\to+\infty]{} -\infty$$

which shows that f is unbounded below.



for $\lambda_{\min}(\mathbf{Q}) < 0$, the unconstrained QP has no solution

Solutions of unconstrained QP (II)

Unconstrained QP

$$\min_{\mathbf{x} \in \mathbb{R}^N} \frac{1}{2} \mathbf{x}^\top \mathbf{Q} \mathbf{x} - \mathbf{p}^\top \mathbf{x}$$

where $\mathbf{Q} \in \mathbb{R}^{N \times N}$ is symmetric and $\mathbf{p} \in \mathbb{R}^N$ is arbitrary.

Second case: $\lambda_1 = \lambda_{\min}(\mathbf{Q}) = 0$ (i.e. $\mathbf{Q} \succeq 0$)

- if $\mathbf{p} \notin \operatorname{Im} \mathbf{Q}$, the equation $\nabla f(\mathbf{x}) = 0$ has no solution, hence there is no stationary point. Since f is convex ($\nabla^2 f \succeq 0$), there is no solution.
- if $\mathbf{p} \in \operatorname{Im} \mathbf{Q}$, the equation $\nabla f(\mathbf{x}) = 0$ has an infinite number of solutions of the form $\mathbf{x}_0 + \mathbf{n}$, $\mathbf{n} \in \ker \mathbf{Q}$

Solutions of unconstrained QP (III)

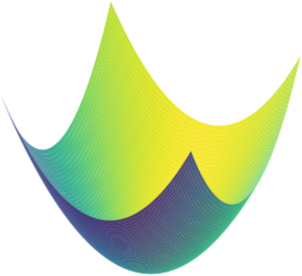
Unconstrained QP

$$\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{p}^\top \mathbf{x}$$

where $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is symmetric and $\mathbf{p} \in \mathbb{R}^N$ is arbitrary.

Third case: $\lambda_1 = \lambda_{\min}(\mathbf{Q}) > 0$ (i.e. $\mathbf{Q} \succ 0$)

The matrix $\mathbf{Q}$ is invertible since all its eigenvalues are strictly positive. Moreover f is strictly convex ($\nabla^2 f \succ 0$) and therefore there is a unique solution, given by $\hat{\mathbf{x}} = \mathbf{Q}^{-1}\mathbf{p}$



Outline

Back to linear least squares

$$\min_{\mathbf{x}\in\mathbb{R}^N} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$$

where $\mathbf{y} \in \mathbb{R}^M$ and $\mathbf{A} \in \mathbb{R}^{M\times N}$. We have already seen that it can be viewed as an unconstrained QP with $\mathbf{Q} = \mathbf{A}^\top\mathbf{A}$ and $\mathbf{p} = \mathbf{A}^\top\mathbf{y}$.

Normal equations

$$\nabla f(\mathbf{x}) = 0 \iff \mathbf{A}^\top\mathbf{A}\mathbf{x} = \mathbf{A}^\top\mathbf{y}$$

Existence and uniqueness

From the results on unconstrained QPs, we get

- Since $\mathbf{Q} = \mathbf{A}^\top\mathbf{A}$, $\mathbf{Q} \succeq 0 \rightarrow$ only cases II and III are relevant.
- Moreover, $\operatorname{Im}(\mathbf{A}^\top\mathbf{A}) = \operatorname{Im}\mathbf{A}^\top$ (since $\ker\mathbf{A} = \ker\mathbf{A}^\top\mathbf{A}$ and $\ker\mathbf{A} = (\operatorname{Im}\mathbf{A}^\top)^\perp$). Thus $\mathbf{p} \in \operatorname{Im}\mathbf{Q}$. Thus, under the conditions of case II, there is an infinite number of solutions.

The linear least squares problem admits at least one solution.

Typology of solutions

$$\min_{\mathbf{x}\in\mathbb{R}^N} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \quad \text{with } \mathbf{y} \in \mathbb{R}^M \text{ and } \mathbf{A} \in \mathbb{R}^{M\times N}$$

The rank of $\mathbf{A}$ rules the number of solutions.

- if $\operatorname{rank}\mathbf{A} = N$ ($\mathbf{A}$ has N linearly independent columns), then $\operatorname{rank}\mathbf{A}^\top\mathbf{A} = N$ and thus $\mathbf{A}^\top\mathbf{A}$ is invertible. The solution is unique, given by

$$\hat{\mathbf{x}} = \left(\mathbf{A}^\top\mathbf{A}\right)^{-1}\mathbf{A}^\top\mathbf{y} \quad \text{when } \operatorname{rank}\mathbf{A} = N$$

  (note that this case implies that $M \geq N$, and this case is sometimes referred to as the *overdetermined case*).

- if $\operatorname{rank}\mathbf{A} < N$, then $\mathbf{A}^\top\mathbf{A}$ has at least one zero eigenvalue and there is an infinite number of solutions of the form

$$\hat{\mathbf{x}} = \mathbf{x}_0 + \mathbf{n}, \text{ where } \mathbf{A}\mathbf{x}_0 = \mathbf{y}, \mathbf{n} \in \ker\mathbf{A} \quad \text{when } \operatorname{rank}\mathbf{A} < N$$

  this is also known as *underdetermined* case.

Moore-Penrose pseudoinverse

Often, the solution to the system $\mathbf{y} = \mathbf{A}\mathbf{x}$ is written as

$$\hat{\mathbf{x}} = \mathbf{A}^{\dagger}\mathbf{y}$$

where $\mathbf{A}^{\dagger}$ is called the Moore-Penrose pseudo-inverse of $\mathbf{A}$.

- when $\mathrm{rank}(\mathbf{A}) = N$ (and thus $M \geq N$), the pseudo-inverse corresponds to the matrix in the solution of the LS problem, i.e.,

$$\mathbf{A}^{\dagger} = \left[\mathbf{A}^{\top}\mathbf{A}\right]^{-1}\mathbf{A}^{\top}$$

- when $\mathrm{rank}(\mathbf{A}) = M < N$ it corresponds to a peculiar solution of the LS problem – that of minimal norm. It has a different expression. More on that in future lectures.

Properties (when $\mathrm{rank}(\mathbf{A}) = N$)

- $\mathbf{A}^{\dagger}$ is a left inverse of $\mathbf{A}$, i.e., $\mathbf{A}^{\dagger}\mathbf{A} = \mathbf{I}_N$
- If $M = N$, $\mathbf{A}$ is invertible and then $\mathbf{A}^{\dagger} = \mathbf{A}^{-1}$.

Outline

TP1: solving least squares problems

github.com/jflamant/mines-nancy-fall24-optimization

Running the notebook: two options

- Google Colab: requires a Google account. The notebook will be run in the cloud. No installation of Python needed.
- Locally: download the notebook and run it on your computer, e.g. using JupyterLab or any other software. Requires a working local install of Python (NumPy and Matplotlib).

Instructions for TPs

- write your answers directly in the notebook. Make use of Markdown language!
- comment your code, as much as possible
- don't know how a function works? don't forget to check the documentation of NumPy and Matplotlib.