

# Analyse Numérique SV

Simone Deparis

EPFL Lausanne – MATH

Printemps 2025



# Organisation

## Vidéos

Pour chaque chapitre, une série de vidéos à apprendre [avant](#) le début du thème.

## Cours

Séances le [vendredi](#) de [10h15 à 12h00](#).

## Séances d'exercices

Séances le [mardi](#) de [8h15 à 10h00](#) en salle [CO 020-023](#) et [CO260](#).

## Moodle, Jupyter, Ed Discussions

[Moodle](#) est la plateforme officielle avec les ressources.

[Jupyter Lab](#) est le GUI pour l'utilisation de Python.

[Ed Discussions](#) est pour les séances d'exercices.

# Règles concernant les exercices

Les exercices (séances théoriques et séances Python) :

- 1 Ils sont **une préparation pour l'examen propédeutique**.
- 2 Chaque série présente des aspects théoriques et Python.
- 3 Vous êtes censés travailler sur chaque série pendant la séance d'exercices et la terminer chez vous si vous en avez la nécessité. Nous ne nous attendons pas à ce que tout le monde ait fini la série en une heure.
- 4 Les énoncés sont disponibles sur le site moodle :  
`https://moodle.epfl.ch/course/view.php?id=7131`
- 5 Les séries d'exercices ne seront pas corrigées.
- 6 Un **corrigé** sera disponible sur le site web.

# Tests intermédiaires et examen

- 1 Il y aura des devoirs à rendre par groupes de 3-4.
- 2 Ces tests font partie intégrante de la note finale. Chaque test donne lieu à  $b_i$  points. La note de l'ensemble des tests se calcule par

$$B = 1 + \sum_{i=1}^5 b_i \in [1, 6], \text{ où } b_i \in \left\{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\right\}, i = 1, \dots, 5.$$

Pour le plus mauvais test, les points  $b_i$  sont remplacés par le meilleur des résultats  $b_i$ . Une absence compte pour  $b_i = 0$ .

Soit  $N$  la note à l'examen. La note finale sera  $0.15B + 0.85N$ .

- 3 L'examen propédeutique concerne le cours et les exercices, Python compris.
- 4 Un formulaire sera disponible pendant l'examen, le cours introductif à Python sera à disposition.
- 5 Si l'examen ne se fait pas à l'ordinateur, vous pourrez utiliser uniquement les calculatrices scientifiques non-programmables suivantes :
  - TI-30X II (B ou S) + TI-30 ECO RS
  - HP 10S – HP 10S+
  - ou des calculatrices encore plus simples.

# Support de cours

- [Notes de cours](#) (disponibles sur moodle).
- Séries d'exercices avec corrigé (disponibles sur moodle chaque semaine)

## Matériel complémentaire

- A. Quarteroni, F. Saleri, P. Gervasio, [Calcul scientifique : Cours, exercices corrigés et illustrations en MATLAB et Octave](#), 2<sup>em</sup>, Springer Italia, 2010.
- J. Rappaz, M. Picasso, [Introduction à l'analyse numérique](#), PPUR, 2010.
- M. Picasso, [Analyse Numérique pour Ingénieurs](#), online Courseware de l'EPFL.ch, <https://courseware.epfl.ch/courses/course-v1:EPFL+analyse-numerique-SV+2020>

## Pour des approfondissements

- A. Quarteroni, R. Sacco, F. Saleri, [Méthodes Numériques - Algorithmes, analyse et applications](#), Springer Italia, 2007

# Objectifs d'apprentissage

- Dans ce cours on étudie plusieurs algorithmes qui nous permettent de résoudre des problèmes mathématiques de façon approchée à l'aide d'un ordinateur.

# Objectifs d'apprentissage

- Dans ce cours on étudie plusieurs algorithmes qui nous permettent de résoudre des problèmes mathématiques de façon approchée à l'aide d'un ordinateur.
- **Ce cours n'est pas un cours de programmation.** On ne vous demandera pas de coder des algorithmes compliqués.

# Objectifs d'apprentissage

- Dans ce cours on étudie plusieurs algorithmes qui nous permettent de résoudre des problèmes mathématiques de façon approchée à l'aide d'un ordinateur.
- **Ce cours n'est pas un cours de programmation.** On ne vous demandera pas de coder des algorithmes compliqués.
- Les objectifs d'apprentissage sont :
  - Avoir une idée de comment un algorithme fonctionne et en connaître la théorie.
  - Connaître les avantages et inconvénients de chaque algorithme.
  - Savoir utiliser les algorithmes en Python.
  - Apprendre à analyser les résultats de façon critique.
  - Apprendre à choisir le meilleur algorithme pour chaque problème.



# Objectifs d'apprentissage

- Dans ce cours on étudie plusieurs algorithmes qui nous permettent de résoudre des problèmes mathématiques de façon approchée à l'aide d'un ordinateur.
- **Ce cours n'est pas un cours de programmation.** On ne vous demandera pas de coder des algorithmes compliqués.
- Les objectifs d'apprentissage sont :
  - Avoir une idée de comment un algorithme fonctionne et en connaître la théorie.
  - Connaître les avantages et inconvénients de chaque algorithme.
  - Savoir utiliser les algorithmes en Python.
  - Apprendre à analyser les résultats de façon critique.
  - Apprendre à choisir le meilleur algorithme pour chaque problème.

Concepts fondamentaux : on parlera beaucoup de

- **stabilité** d'un algorithme ;
- **précision** d'un algorithme ;
- **coût** d'un algorithme.

# Contenu du cours

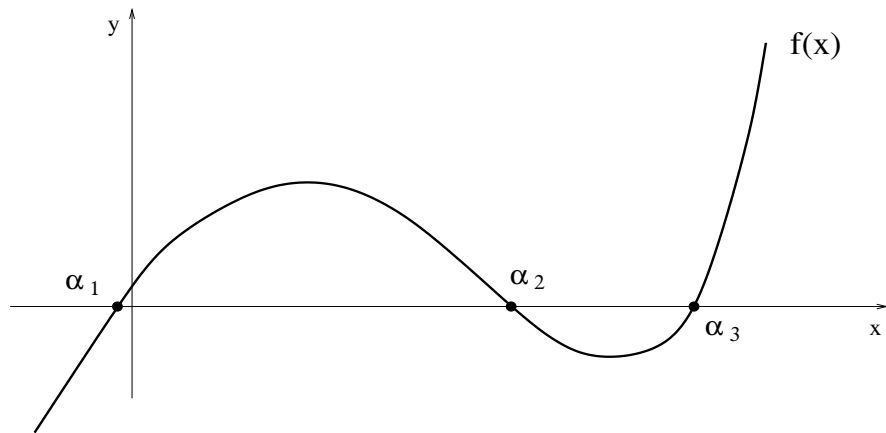
## Prérequis

Cours d'Analyse, Algèbre Linéaire, ICC et (nouveau) Python

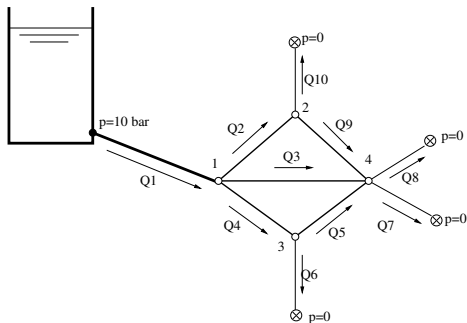
## Contenus du cours

- Introduction à Python et Jupyter.
- Équations non linéaires.
- Interpolation.
- Interpolation par morceaux.
- Approximation par moindres carrés (pas de MOOC).
- Intégration numérique.
- Dérivation numérique.
- Systèmes linéaires :
  - méthodes directes ;
  - méthodes itératives.
- Équations différentielles ordinaires.

# Équations non linéaires



# Systèmes linéaires



Exemple de réseau hydraulique

$$A\mathbf{p} = \mathbf{b},$$

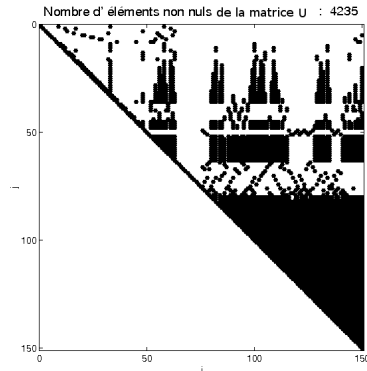
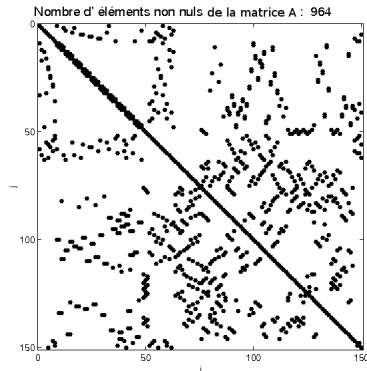
où  $\mathbf{p} = [p_1, p_2, p_3, p_4]^T$  est le vecteur des pressions (inconnues) aux 4 noeuds du réseau. On a la relation suivante :

$$Q_j = kL\Delta p_j,$$

où  $k$  est la résistance hydraulique et  $L$  la longueur de la conduite. Pour des paramètres donnés,  $A$  et  $b$  sont

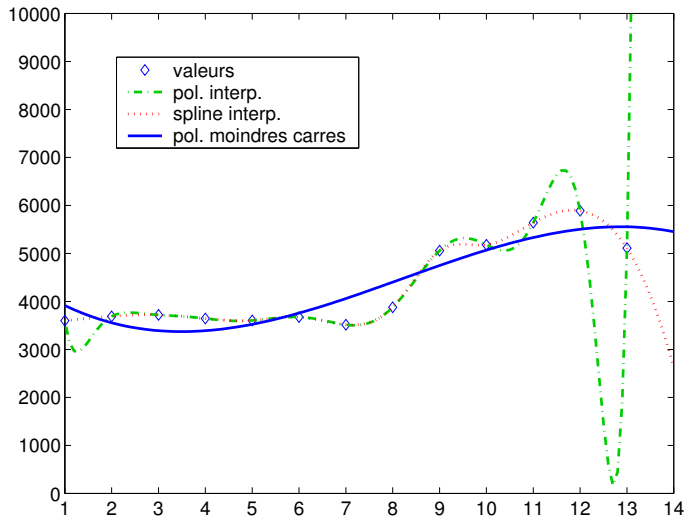
$$A = \begin{bmatrix} -0.360 & 0.050 & 0.050 & 0.060 \\ 0.050 & -0.116 & 0.000 & 0.050 \\ 0.050 & 0.000 & -0.116 & 0.050 \\ 0.060 & 0.050 & 0.050 & -0.192 \end{bmatrix}, \quad b = \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

# Systèmes linéaires



Disposition des éléments non nuls de la matrice creuse  $K$  (gauche) et de la matrice  $H^T$  de la factorisation de Cholesky de  $K$  (droite)

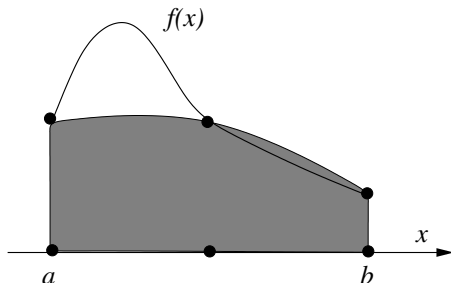
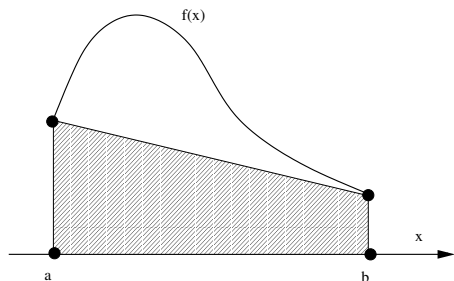
# Interpolation



Comparaison entre le polynôme interpolant, la spline interpolante et le polynôme aux moindres carrés de degré 3.

# Intégration

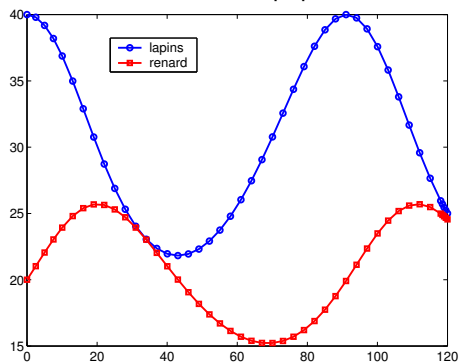
$$\int_0^1 e^{-x^2} dx \quad \int_0^{\frac{\pi}{2}} \sqrt{1 + \cos^2 x} dx \quad \int_0^1 \cos x^2 dx \quad \int_0^1 \frac{\log x}{\sqrt{1+x^2}} dx \quad \int_0^1 \frac{e^x}{\sqrt{1+x^2}}$$



Formules du trapèze (à gauche) et de Simpson (à droite)

# Équations différentielles

On considère deux populations composées de  $y_1$  et  $y_2$  individus, où  $y_1$  est le nombre de proies et  $y_2$  le nombre de prédateurs. L'évolution des deux populations est alors décrite par le système



d'équations différentielles

Évolution des populations de lapins et de renards sur 10 ans.

$$\begin{cases} y_1'(t) = C_1 y_1(t) [1 - b_1 y_1(t) - d_2 y_2(t)], \\ y_2'(t) = -C_2 y_2(t) [1 - b_2 y_2(t) - d_1 y_1(t)], \end{cases}$$



# What can go wrong ?

Nous allons découvrir des méthodes d'**approximation numérique**. Il y a (au moins) deux composantes importantes à comprendre avec cette approximation :

- **La discrétisation du problème**, qui permet de transcrire un problème mathématique continu en un problème de dimension finie. À l'aide de l'ordinateur, nous pouvons résoudre le problème discret.
- **L'erreur d'approximation**, qui est toujours présente et que nous devons apprendre à **maîtriser**. Autrement, la solution du problème discret ne nous apporte aucune information utile.