# Si Euler était vivant, il ferait de l'analyse numérique

D. Doyen

"As a branch of mathematics, however, numerical analysis probably ranked the lowest, even below the theory of statistics, in terms of what most university mathematicians found interesting."

A. Hodges, *Alan Turing : The Enigma*

**Définition :** L'analyse numérique est l'étude des algorithmes permettant de résoudre les problèmes issus des mathématiques du continu.

**Exemples de problèmes :**

- Résoudre un système linéaire
- Résoudre une équation ou un système non-linéaire
- Minimiser une fonction d'une ou plusieurs variables
- Calculer l'intégrale d'une fonction d'une ou plusieurs variables
- Trouver les valeurs propres (ou les vecteurs propres) d'une matrice
- Résoudre une équation différentielle
- Résoudre une équation aux dérivées partielles

**Autres noms :** calcul scientifique, *computational mathematics*

## Pourquoi veut-on faire de tels calculs ?

Historiquement, les premiers domaines consommateurs de calculs numériques sont la balistique, l'astronomie, la géodésie.

**Balistique**

- Etablir des tables de tir pour les artilleurs

**Astronomie**

- Déterminer les orbites des corps célestes à partir d'observations

**Géodésie**

- Etablir des cartes par triangulation

Quelques noms associés à ces travaux : Euler, Gauss, Laplace, Lagrange, Legendre, ...

Aujourd'hui, quasiment tous les domaines des sciences et de l'industrie ont des besoins en calcul numérique.

**Météorologie**

- Prévision du temps

**Astronomie**

- Déterminer les positions des corps célestes par simulation

**Génie civil**

- Etude de la résistance des structures

**Traitement des données**

- Classement des pages web (PageRank)

**Et encore beaucoup d'autres applications...**

# Pourquoi a-t-on besoin d'algorithmes ?

- Résoudre **l'équation** $ax^2 + bx + c = 0$

  $\rightarrow x = \frac{-b-\sqrt{\Delta}}{2a}$ ou $x = \frac{-b+\sqrt{\Delta}}{2a}$

- Trouver **les valeurs propres** de la matrice $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

  $\rightarrow$ Ecrire le polynôme caractéristique (de degré 2), puis trouver ses racines.

- Résoudre **l'équation différentielle** $y'(t) = ay(t)$

  $\rightarrow y(t) = Ke^{at}$

- Résoudre **le système linéaire**

$$\begin{cases} a_{11}x + a_{12}y = b_1 \\ a_{21}x + a_{22}y = b_2 \end{cases}$$

  $\rightarrow$ Résolution par substitution ou combinaison

- Résoudre **l'équation** $ax^5 + bx^4 + cx^3 + dx^2 + ex + f = 0$

  $\rightarrow$ pas de formule

- Trouver **les valeurs propres** d'une matrice $A$ de taille $n \geq 5$

  $\rightarrow$ pas de formule

- Résoudre **l'équation différentielle** $y'(t) = y^2(t) + t^2$

  $\rightarrow$ pas de formule

- Résoudre **le système linéaire** $Ax = b$

  $\rightarrow$ Résolution par la méthode d'élimination de Gauss (nombre fini d'étapes)

## Qu'est-ce qu'un bon algorithme ?

- Un algorithme peu coûteux en nombre d'opérations.
- Un algorithme peu coûteux en occupation de la mémoire.
- Un algorithme qui n'amplifie pas les erreurs (erreurs sur les données, erreurs d'arrondi).

## Une première idée : itérer

# Algorithme de Newton-Raphson

**Objectif :** Trouver une solution de l'équation $f(x) = 0$ (où $f$ est une fonction de $\mathbb{R}$ dans $\mathbb{R}$).

**Idée :** Linéariser l'équation par un développement de Taylor au 1er ordre, puis itérer.

**Description de l'algorithme :**

- Initialisation. On part d'une valeur $x_0 \in \mathbb{R}$.
- Itération. On calcule $x_{k+1}$ à partir de $x_k$ en résolvant l'équation

$$f(x_k) + f'(x_k)(x - x_k) = 0,$$

  ce qui donne

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

- Critère d'arrêt. On arrête l'algorithme quand

$$|x_{k+1} - x_k| < \epsilon \qquad \text{ou} \qquad |f(x_{k+1})| < \epsilon$$

# Algorithme de Newton-Raphson

**Exemple :** $x^2 + 2 = 0$.

**Programme Scilab :**

```
N=4 // nombre d'itérations fixé
x=1
for n=1:N
  x=1/2*(x+2/x)
end
```

**Résultats :**

| $x_0$ | 1 |
|---|---|
| $x_1$ | 1.5 |
| $x_2$ | 1.4166667 |
| $x_3$ | 1.4142157 |
| $x_4$ | 1.4142136 |

| $x_0$ | -3 |
|---|---|
| $x_1$ | -1.8333333 |
| $x_2$ | -1.4621212 |
| $x_3$ | -1.4149984 |
| $x_4$ | -1.4142138 |

# Algorithme de Newton-Raphson

- Quel est le coût de cet algorithme ?
- A quelles conditions (sur le problème, sur $x_0$) la suite des solutions approchées $(x_k)_{k\in\mathbb{N}}$ converge-t-elle vers une solution ?
- A quelle vitesse cette suite converge-t-elle ? (ordre de convergence)

**Théorème**

Supposons que la fonction $f$ est $C^2$. Soit $x_*$ tel que $f(x_*) = 0$ et $f'(x_*) \neq 0$. Alors, si $x_0$ est choisi suffisamment proche de $x_*$, l'algorithme de Newton-Raphson converge vers $x_*$. De plus,

$$|x_{k+1} - x_*| \leq C|x_k - x_*|^2$$

- Peut-on trouver un algorithme avec de meilleures propriétés ?

# Algorithme de la puissance

Soit $A$ une matrice symétrique réelle de taille $n$. On suppose que les valeurs propres sont ordonnées de la façon suivante :

$$|\lambda_n| > |\lambda_{n-1}| \geq |\lambda_{n-2}| \geq \ldots \geq |\lambda_1|$$

**Objectif :** Déterminer $\lambda_n$.

**Idée :** Si $x = \alpha_1 v_1 + \ldots + \alpha_n v_n$, alors $Ax = \lambda_1 \alpha_1 v_1 + \ldots + \lambda_n \alpha_n v_n$.

**Description de l'algorithme :**

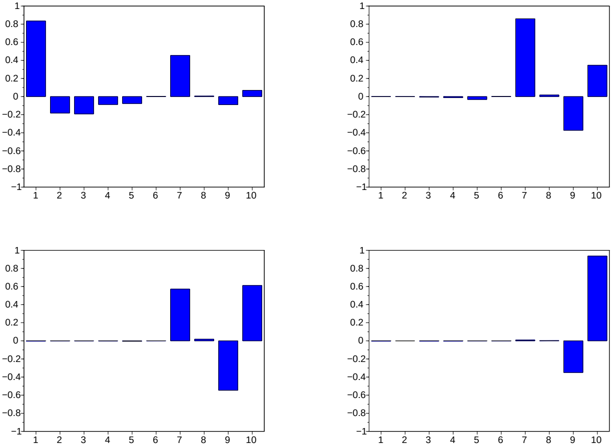
- On part d'un vecteur $x_0 \in \mathbb{R}^n$.

- On calcule $x_{k+1}$ à partir de $x_k$ en faisant

$$x_{k+1} = \frac{Ax_k}{\|Ax_k\|}.$$

# Algorithme de la puissance

**Exemple :** Matrice symétrique définie positive de taille 10. Vecteur initial choisi alétoirement. Les composantes dans la base des vecteurs propres des vecteurs $x_0$, $x_3$, $x_6$ et $x_{20}$ sont représentées ci-dessous.



# Algorithme de Gauss-Seidel

La méthode d'élimination de Gauss n'est pas efficace pour résoudre les systèmes linéaires dont la matrice est creuse.

**Exemple :**

$$A = \begin{pmatrix} 1 & 0.1 & \dots & \dots & 0.1 \\ 0.1 & \ddots & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0.1 & 0 & \dots & 0 & 1 \end{pmatrix}.$$

Lorsqu'on applique la méthode d'élimination de Gauss à cette matrice creuse, on obtient la matrice triangulaire pleine suivante :

$$U = \begin{pmatrix} 1 & 0.1 & \dots & \dots & 0.1 \\ 0 & 0.99 & -0.01 & \dots & -0.01 \\ 0 & 0 & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & -0.01 \\ 0 & 0 & 0 & 0 & 0.99 \end{pmatrix}.$$

# Algorithme de Gauss-Seidel

**Objectif :** Résoudre le système linéaire $Ax = b$.

**Idée :** Décomposer $A$ sous la forme $A = M + N$ où $M$ est une matrice facile à "inverser" (triangulaire inférieure).

**Description de l'algorithme :**

- On part d'un vecteur $x_0 \in \mathbb{R}^n$.
- On calcule $x_{k+1}$ à partir de $x_k$ en résolvant le système linéaire

$$Mx - k + 1 = -Nx - k + b$$

- On arrête l'algorithme quand

$$\|x_{k+1} - x_k\| < \epsilon \qquad \text{ou} \qquad \|Ax_{k+1} - b\| < \epsilon$$

# Une deuxième idée : discrétiser

# Schéma d'Euler

**Objectif :** Trouver la solution du problème

$$\begin{cases} y'(t) = f(t, y(t)), & t \in [0, T], \\ y(0) = y_{\text{ini}}, \end{cases}$$

**Idée :** Si l'on connaît la solution au temps $t$, une approximation de la solution au temps $t + h$ est

$$y(t+h) \approx y(t) + hy'(t) = y(t) + hf(t, y(t)).$$

**Description de l'algorithme :** On commence par choisir un nombre de pas $N$ ; le pas vaut alors $h = T/N$. On pose $t_n = nh$ pour tout $n \in \{0, \ldots, N\}$. La méthode d'Euler consiste à calculer les valeurs $y_0$, $y_1$, ..., $y_N$ définies par la formule de récurrence

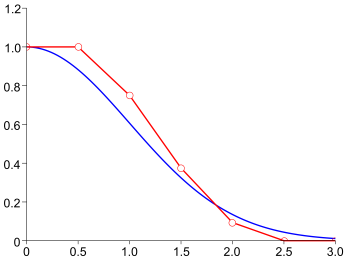
$$\begin{cases} y_0 = y_{\text{ini}} \\ y_{n+1} = y_n + hf(t_n, y_n) \end{cases}$$

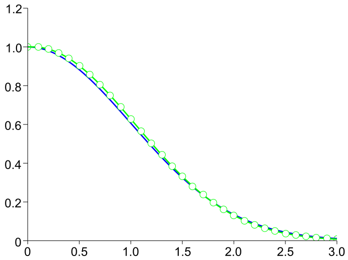
# Schéma d'Euler

**Exemple :** Résolution du problème

$$\begin{cases} y'(t) = -ty(t), & t \in [0,3], \\ y(0) = 1, \end{cases}$$

avec $h = 0.5$ et $h = 0.1$



# Schéma d'Euler

## Théorème

Supposons que la fonction $f$ est lipschitzienne et que la solution $y$ soit $C^2$. Alors,

$$\max_{0\leq n\leq N} |y_n - y(t_n)| \leq Ch.$$

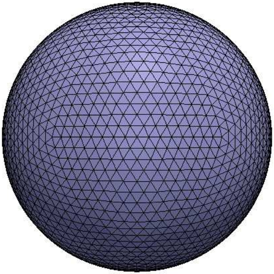
**Exemple :**

| $h$ | erreur |
|---|---|
| $10^{-1}$ | 0.0241574 |
| $10^{-2}$ | 0.0023122 |
| $10^{-3}$ | 0.0002301 |
| $10^{-4}$ | 0.0000230 |

# Méthode des volumes finis

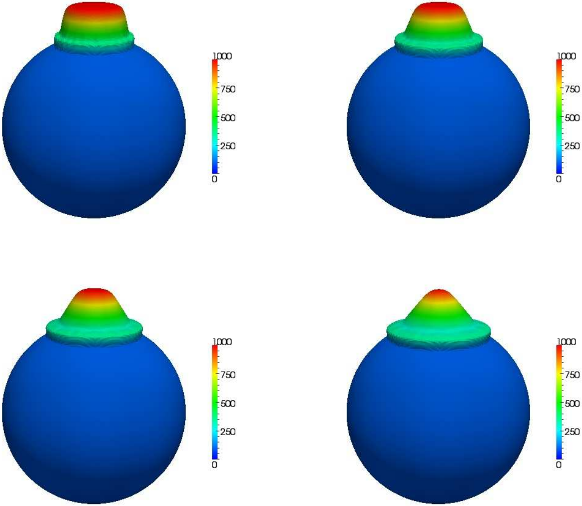
**Objectif :** Résoudre les équations (aux dérivées partielles) décrivant le mouvement d'un fluide à la surface d'une sphère.

**Idée :** Mailler la sphère. Approcher les fonctions inconnues par un ensemble fini de valeurs (une par maille). Ecrire des bilans de conservation entre les mailles.



# Méthode des volumes finis

**Exemple de simulation :**



# Représentation des nombres réels sur un ordinateur

## Représentation à virgule flottante

$$1.2345 = \underbrace{12345}_{\text{mantisse}} \times \underbrace{10}_{\text{base}}^{\overbrace{-4}^{\text{exposant}}}$$

## Formats standards

| format | taille | base | signe | mantisse | exposant |
|---|---|---|---|---|---|
| simple précision | 32 bits | 2 | 1 bit | 23 bits | 8 bits |
| double précision | 64 bits | 2 | 1 bit | 52 bits | 11 bits |

**On ne peut pas représenter les réels trop grands, ni les réels trop petits.**

En double précision, $r_{min} \approx 2.2 \cdot 10^{-308}$ et $r_{max} \approx 1.8 \cdot 10^{308}$

**La plupart des réels de taille intermédiaires ne peuvent pas être représentés exactement.**

```
-->0.1
 ans  =
0.10000000000000000555111
```

**Mais l'erreur d'arrondi relative est bornée :**

$$|fl(x) - x| \leq \mathtt{u}|x|$$

où $\mathtt{u}$ est l'unité d'arrondi ($\mathtt{u} \approx 1.11 \cdot 10^{-16}$ en double précision).

**A chaque opération, on fait généralement aussi une erreur d'arrondi.**

# Attention aux erreurs d'arrondi !

## Un premier exemple

```
-->((1+1e-15)-1)/1e-15
 ans  =
    1.1102230246251565404236
```

Pourquoi le résultat est-il grossièrement faux ?

```
-->1+1e-15
 ans  =
    1.000000000000001110
```

```
-->(1+1e-15)-1
 ans  =
    1.110223024625156540e-15
```

## Un deuxième exemple

Calculer $\left(1+\dfrac{1}{n}\right)^{n}$.

| $n$ | résultat |
|---|---|
| $10^{11}$ | 2.7182821 |
| $10^{12}$ | 2.7185235 |
| $10^{14}$ | 2.7185235 |
| $10^{13}$ | 2.71611 |
| $10^{14}$ | 2.71611 |
| $10^{15}$ | 3.0350352 |
| $10^{16}$ | 1. |

## Un troisième exemple (tiré d'un sujet de baccalauréat récent)

Calculer l'intégrale $I_n = \int_0^1 x^n e^{x^2}\, dx$. en utilisant la récurrence

$$I_{n+2} = \frac{1}{2}e - \frac{n+1}{2} I_n.$$

| $n$ | $I_n$ |
|---|---|
| 25 | 0.0975500 |
| 27 | 0.0909915 |
| 29 | 0.0852595 |
| 31 | 0.0802479 |
| 33 | 0.0751741 |
| 35 | 0.0811815 |
| 37 | - 0.1021268 |
| 39 | 3.2995497 |
| 41 | - 64.631854 |