Motivation

Numerical analysis is the study of algorithms for the problems of continuous mathematics

Lloyd N. Trefethen [1]

- study of algorithms :
  étude des algorithmes
- problems of continuous mathematics :
  problèmes à variables dans $\mathbb{R}$, $\mathbb{C}$
- Exemples :
  - résolution de systèmes d'équations linéaires
  - détermination de valeurs et vecteurs propres d'une matrice
  - résolution d'équations différentielles
  - calcul des zéros d'une fonction
  - évaluation d'intégrales
  - interpolation et approximation

1. metronu.ulb.ac.be/MATH-H-202/trefethen-def-na.pdf

Motivation

Numerical analysis is the study of algorithms for the problems of continuous mathematics

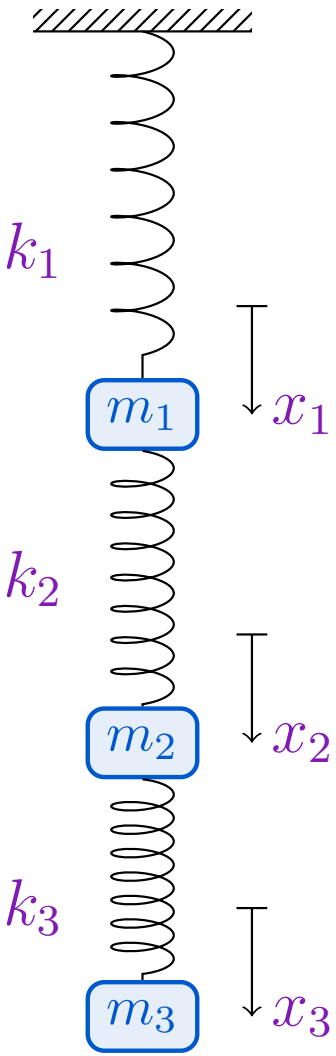
Lloyd N. Trefethen [1]

- study of algorithms :
  étude des algorithmes
- problems of continuous mathematics :
  problèmes à variables dans $\mathbb{R}$, $\mathbb{C}$
- Exemples :
  - résolution de systèmes d'équations linéaires
  - détermination de valeurs et vecteurs propres d'une matrice
  - résolution d'équations différentielles
  - calcul des zéros d'une fonction
  - évaluation d'intégrales
  - interpolation et approximation

1. metronu.ulb.ac.be/MATH-H-202/trefethen-def-na.pdf

Analyse numérique utile pour les ingénieurs ?

Exemple 1



x_1, $x_2 - x_1$, $x_3 - x_2$ – élongations des ressorts
k_1, k_2, k_3 – constantes de rappel

conditions d'équilibre :

$$\begin{aligned} k_2(x_2 - x_1) - k_1 x_1 + m_1 g &= 0 \\ k_3(x_3 - x_2) - k_2(x_2 - x_1) + m_2 g &= 0 \\ -k_3(x_3 - x_2) + m_3 g &= 0 \end{aligned}$$

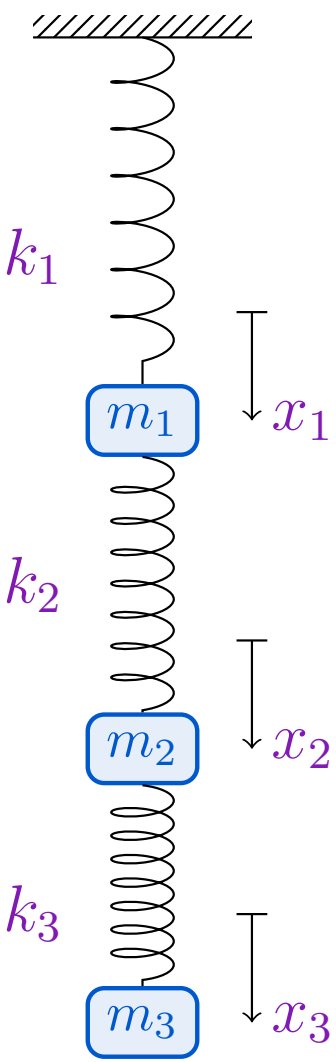
$$\Rightarrow \begin{pmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} m_1 g \\ m_2 g \\ m_3 g \end{pmatrix}$$

$$\Rightarrow A\mathbf{x} = \mathbf{b} \quad \text{système linéaire}$$

Le système pour $n = 3$ masses peut être résolu à la main.

Pour n grand (ex : $n = 10^6$) : ordinateur nécessaire !

Exemple 1 (suite)



Pour $k_i = m_i g = 1$ le système devient :

$$\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Résolution avec Octave :

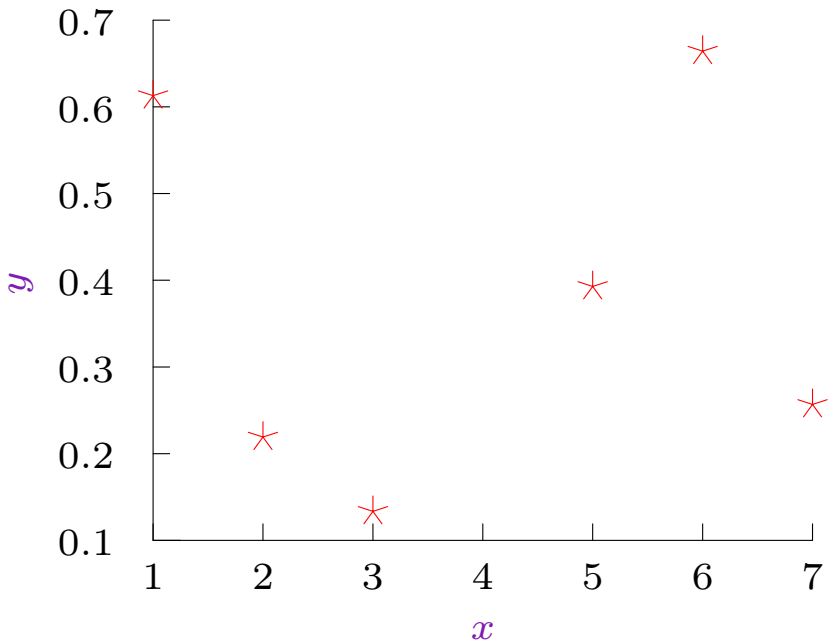
```
    % créer le second membre b
b = [1; 1; 1];
    % créer la matrice A
A = [2 -1 0; -1 2 -1; 0 -1 1];
    % résoudre Ax=b
x = A\b;
    % afficher x
x
```

Exemple 2

Approcher des données par un polynôme.

Choisir $x_1, \ldots, x_6$ et $y_1, \ldots, y_6$:

```
    % choisir les abscisses
x = [1; 2; 3; 5; 6; 7];
    % choisir les ordonnées
y = rand(6,1);
    % afficher le résultat
plot(x, y, '*r')
```

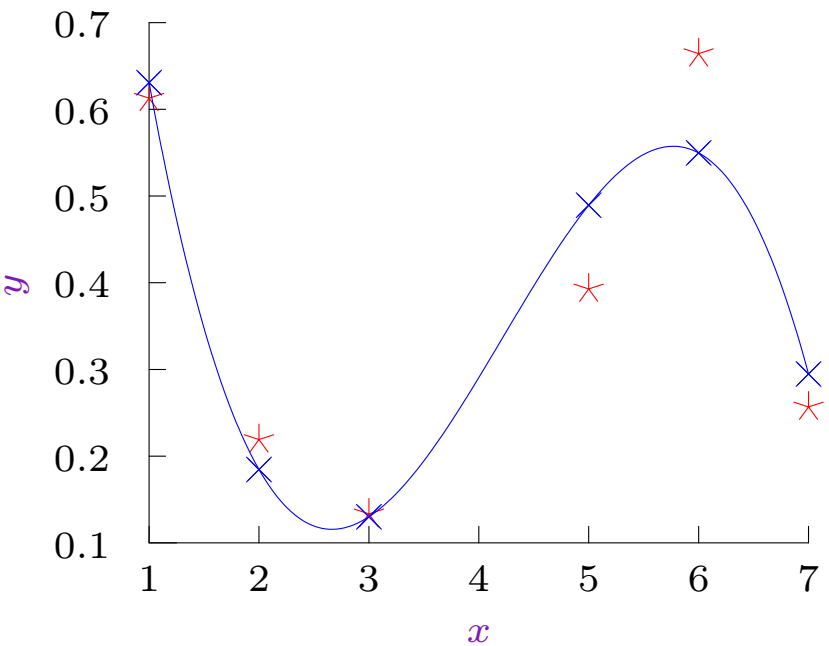


Exemple 2 (suite)

Approcher des données par un polynôme.

Appliquer la méthode des moindres carrés :

```
    % approcher par polynôme
    % de degré <= 3 au sens
    % des moindres carrés
pol = polyfit(x,y,3);
    % garder graphe précédent
hold on
    % afficher polynôme en x
plot(x, polyval(pol,x), 'x')
    % raffiner x & afficher
x = [1:0.01:7];
plot(x, polyval(pol,x), '-')
```



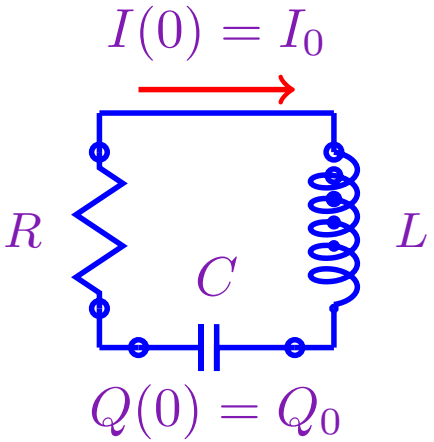
`polyfit(·,·,3)` détermine un polynôme $p(x)$ de degré au plus 3 qui minimise

$$\sum_{k=1}^{6} (y_i - p(x_i))^2$$

Introduction : Exemple 3

Transitoire dans un circuit électrique :

loi des mailles + définition du courant



$$\begin{cases} L\frac{dI}{dt} + RI + \frac{Q}{C} &= 0 \\ \frac{dQ}{dt} &= I \\ I(0) &= I_0\,, \quad Q(0) = Q_0 \end{cases}$$

$$\Rightarrow \begin{cases} \frac{d}{dt}\begin{pmatrix} I \\ Q \end{pmatrix} &= \begin{pmatrix} -RL^{-1} & -C^{-1}L^{-1} \\ 1 & \end{pmatrix}\begin{pmatrix} I \\ Q \end{pmatrix} \\ \begin{pmatrix} I \\ Q \end{pmatrix}\Big|_{t=0} &= \begin{pmatrix} I_0 \\ Q_0 \end{pmatrix} \end{cases}$$

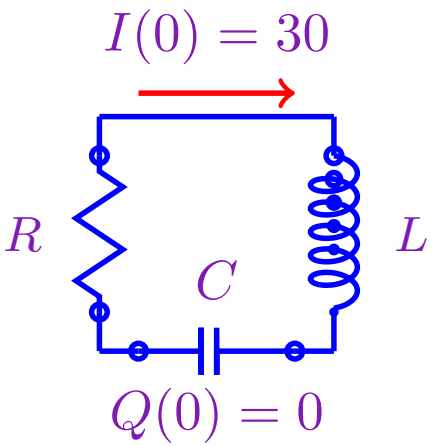
$$\Rightarrow \begin{cases} \frac{d}{dt}\mathbf{u} &= \mathbf{f}(\mathbf{u}) \\ \mathbf{u}(0) &= \mathbf{u}_0 \end{cases}$$

(système d') équations différentielles

Exemple 3 (suite)

Transitoire dans un circuit électrique :

Soient $RL^{-1} = C^{-1}L^{-1} = I_0 = 30$, $Q_0 = 0$; on a alors



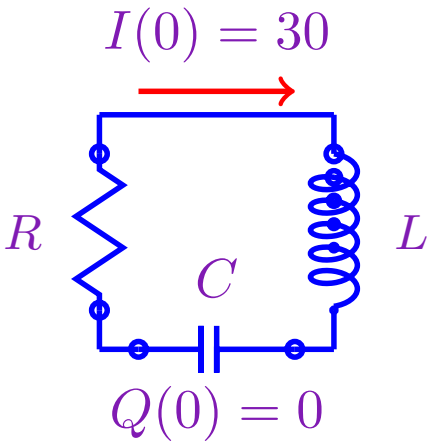
$$\begin{cases} \frac{dI}{dt} &= -30I - 30Q \\ \frac{dQ}{dt} &= I \\ I(0) &= 30, \quad Q(0) = 0 \end{cases}$$

```
    % définir F(u,t)
F = @(u,t)( [-30*u(1)-30*u(2); u(1)]);
    % conditions initiales
u0 = [30; 0];
    %  t = 0, 0.01, ..., 1
T = 0:0.01:1;
    % résoudre
sol = lsode(F, u0, T);
    % graphe de I & Q
plotyy(T, sol(:,1), T, sol(:,2))
```

Exemple 3 (suite)

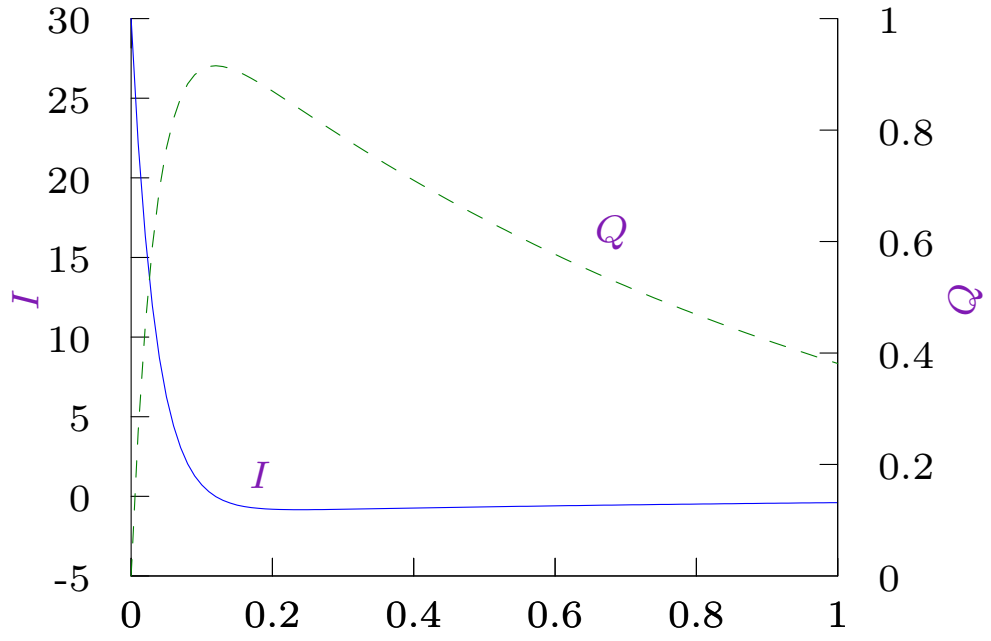
Transitoire dans un circuit électrique :

Soient $RL^{-1} = C^{-1}L^{-1} = I_0 = 30$, $Q_0 = 0$; on a alors



$$\begin{cases} \frac{dI}{dt} &= -30I - 30Q \\ \frac{dQ}{dt} &= I \\ I(0) &= 30\,, \quad Q(0) = 0 \end{cases}$$

La solution numérique est



Conclusion intermédiaire : Octave semble utile aux ingénieurs.

Mais a-t-on vraiment besoin d'étudier et de comprendre les algorithmes correspondants ?

Les quelques raisons en faveur du «oui » ...

Exemple 4

Raison 1 : Une méthode est rarement la meilleure dans tous les cas

Exemple 4 : La fonction `solvetime` mesure le temps de résolution d'un système (`x = A\b`) représenté en format

- dense (par défaut)
- creux (`sparse`)

Le format creux ne prend en compte que les éléments non nuls de la matrice.

```
function solvetime(A,b)
    % solveur dense
tic
   x = A\b;
toc
    % solveur creux
A = sparse(A);
tic
   x = A\b;
toc
```

Exemple 4 (suite)

Cas 1 : matrice de l'Exemple 1 avec $n = 1000$ masses ($k_i = m_i g = 1$) :

$$\begin{pmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

```
% matrice plutôt creuse
n=1000;
A = -diag(ones(n-1,1),1) + ...
    diag(ones(n,1));
A = A + A';
A(n,n) = 1;
b = ones(n,1);
solvetime(A,b)
```

résultat :

```
Elapsed time is 0.14 seconds.
Elapsed time is 0.00051 second
```

fonction pour matrices denses
275× plus lente !

Exemple 4 (suite)

Cas 2 : matrice 1000×1000 avec les éléments aléatoires :

```
% matrice plutôt dense
n=1000;
A=rand(n);
b = ones(n,1);
solvetime(A,b)
```

résultat :

```
Elapsed time is 0.16 seconds.
Elapsed time is 1.2 seconds.
```

fonction pour matrices denses
$7.5\times$ plus rapide !

Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.

```
function resol_Ex5(Tmax)
% Tmax - durée de l'experience
f = @(u,t)([2*u(2)-1*u(1); u(1)]);
u0 = [-2; 1];
T = 0:0.001:Tmax;
sol = lsode(f, u0, T);
plotyy(T, sol(:,1), T, sol(:,2))
```

Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

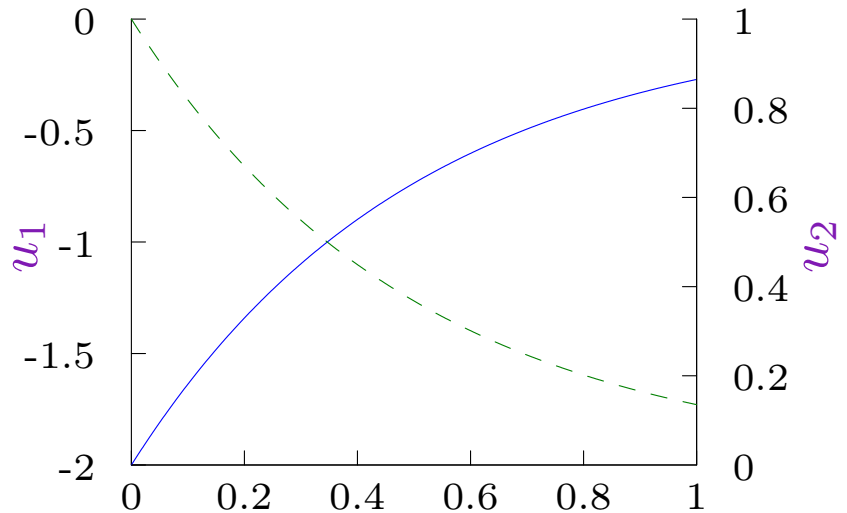
Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.



Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

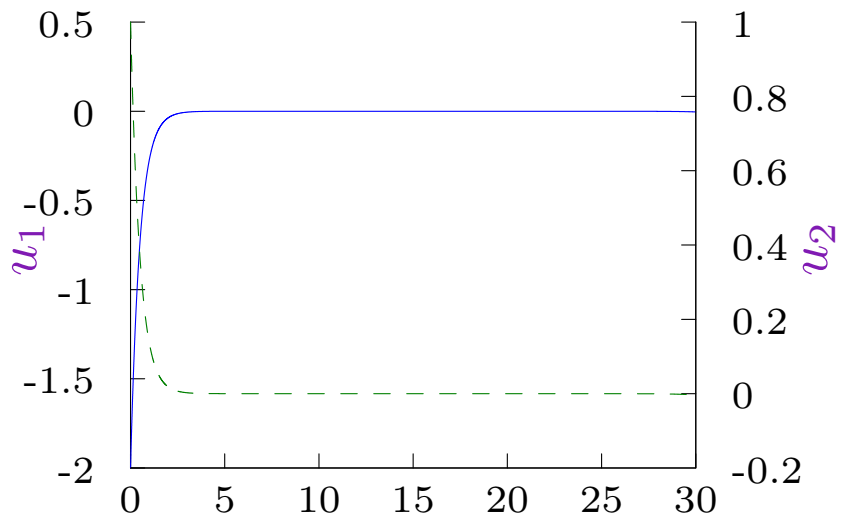
Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.



Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

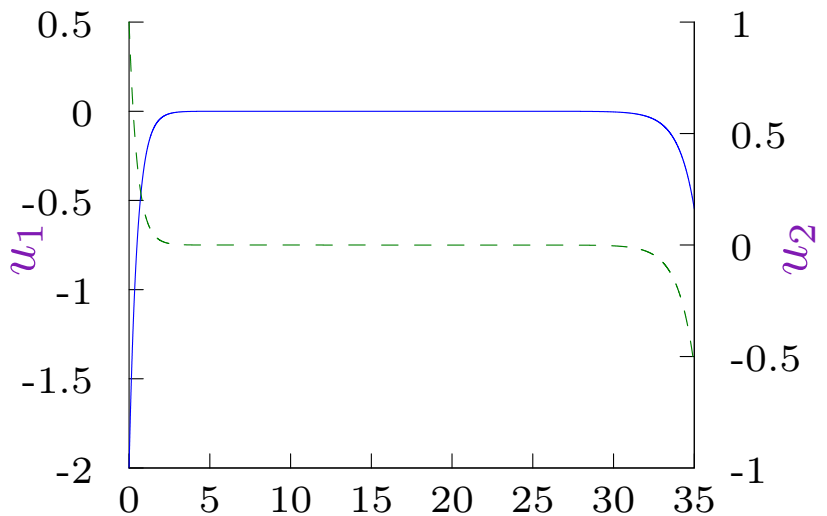
Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.



Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

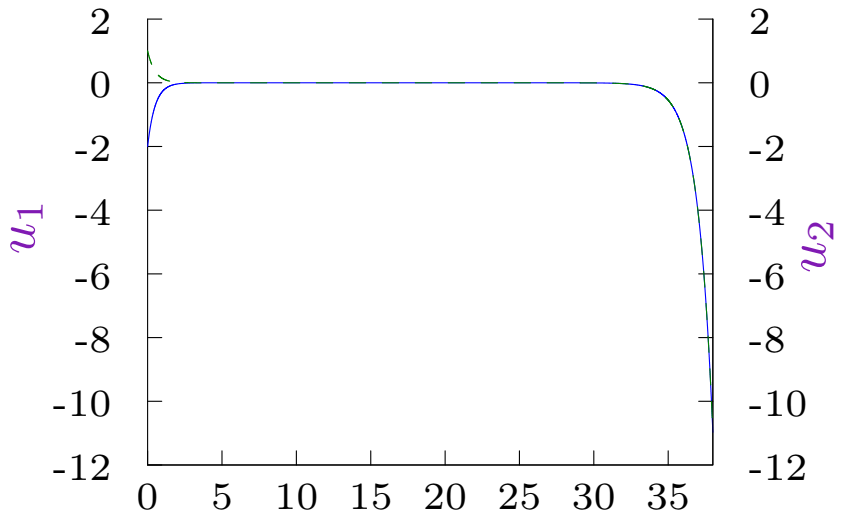
Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.



Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

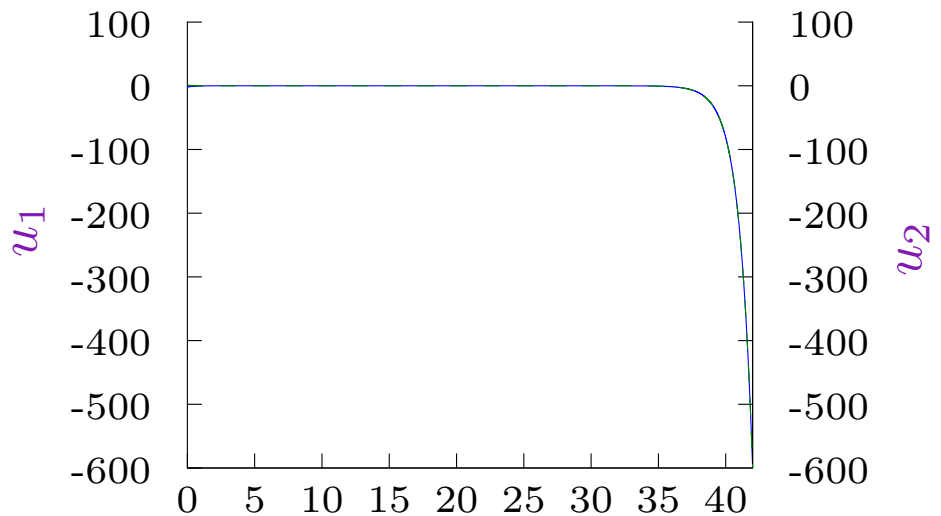
Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.



Exemple 5

Raison 2 : Les méthodes *numériques* ont des limites et certains problèmes (problèmes mal conditionnés) sont difficiles pour toutes les méthodes

Exemple 5 : le système d'équations différentielles

$$\begin{cases} \frac{du_1}{dt} &= -u_1 + 2u_2 \\ \frac{du_2}{dt} &= u_1 \end{cases}, \text{ avec } \begin{cases} u_1(0) = -2 \\ u_2(0) = 1 \end{cases}$$

a comme solution exacte

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} e^{-2t};$$

en particulier $u_1(t)$ et $u_2(t)$ tendent vers 0 pour $t \to \infty$.

Explication : la solution pour $u_1(0)$, $u_2(0)$ quelconques est

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix} \frac{u_2(0) - u_1(0)}{3} e^{-2t} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \frac{2u_2(0) + u_1(0)}{3} e^{t};$$

les erreurs d'arrondi «révèlent» la composante instable,
absente dans la solution exacte.

Exemple 6

Raison 3 : Vous serez amenés à développer vos propres outils numériques

Exemple 6 : déterminer la variance d'un échantillon de taille n. Deux formules sont à votre disposition :

$$\mathrm{var}(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \overline{x})^2 \tag{1}$$

$$= \frac{1}{n} \sum_{i=1}^{n} x_i^2 - \overline{x}^2 \tag{2}$$

```
function var = var_frm1(x,n)
x_moy = sum(x)/n;
var = sum((x-x_moy).^2)/n;
```

```
function var = var_frm2(x,n)
x_moy = sum(x)/n;
var = sum(x.^2)/n - x_moy^2;
```

pour `x=[200000000 200000001 200000002]` les résultats sont

```
ans = 0.66667
```

```
ans = 0
```

pour `x=[100000000 100000001 100000002]` les résultats sont

```
ans = 0.66667
```

```
ans = 2
```

Exemple 6

Raison 3 : Vous serez amenés à développer vos propres outils numériques

Exemple 6 : déterminer la variance d'un échantillon de taille n. Deux formules sont à votre disposition :

$$\mathrm{var}(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \overline{x})^2 \tag{1}$$

$$= \frac{1}{n} \sum_{i=1}^{n} x_i^2 - \overline{x}^2 \tag{2}$$

```
function var = var_frm1(x,n)
x_moy = sum(x)/n;
var = sum((x-x_moy).^2)/n;
```

```
function var = var_frm2(x,n)
x_moy = sum(x)/n;
var = sum(x.^2)/n - x_moy^2;
```

pour x= [20000000 20000001 20000002] les résultats sont

ans = 0.66667 ans = 0

pour x= [10000000 10000001 10000002] les résultats sont

ans = 0.66667 ans = 2

La deuxième formule est clairement problématique. Et la première ?

En résumé

Message 1 : Octave (et bien d'autres logiciels) contient des commandes basées sur des algorithmes de qualité.

et malgré cela ...

Message 2 : Les outils en boites noires n'existent pas pour les problèmes d'analyse numérique ; tout utilisateur doit avoir une compréhension des algorithmes utilisés, leurs avantages, inconvénients et limitations.

Ce cours n'est donc pas (uniquement) ...

une introduction à Octave (Matlab, etc.),

mais (surtout) ...

une introduction à l'analyse numérique et au calcul numérique.