Devoir maison 2 pour vendredi 10 octobre (ou avant)

L'interpolation est une technique fondamentale en analyse numérique, qui consiste à construire une fonction approchant au mieux un ensemble de données discrètes. La méthode la plus naturelle consiste à utiliser un polynôme unique passant par tous les points connus. Cependant, cette approche devient rapidement problématique : les polynômes de degré élevé sont coûteux à manipuler et souffrent de phénomènes d'instabilité numérique, comme les oscillations parasites (phénomène de Runge).
Pour pallier ces difficultés, on utilise les splines, c'est-à-dire des fonctions polynomiales définies par morceaux, assemblées de manière à garantir une certaine régularité (continuité de la fonction, mais aussi de ses dérivées). L'exemple le plus courant est la spline cubique, où chaque intervalle est interpolé par un polynôme de degré 3, raccordé de façon à produire une courbe lisse.
Les splines jouent un rôle central dans de nombreux domaines : traitement de données, conception assistée par ordinateur, robotique, animation, ou encore résolution numérique d'équations différentielles.
Dans ce travail, nous étudierons l'interpolation par splines sur des exemples et dans le cas général et nous la comparerons à l'interpolation polynomiale classique.

– –

Le but des questions **I)**, **II)** et **III)** est de trouver une fonction f simple et qui vérifie les propriétés suivantes :

$$f \text{ est de classe } C^1 \text{ sur } [-1;4] \text{ et : } \begin{cases} f(-1)=0 \\ f'(-1)=0 \end{cases}, \begin{cases} f(0)=1 \\ f'(0)=1 \end{cases}, \begin{cases} f(2)=2 \\ f'(2)=-1 \end{cases} \text{ et } \begin{cases} f(4)=0 \\ f'(4)=0 \end{cases}$$

On note $(*)$ l'ensemble de ces conditions.

Les deux dernières parties étudierons une généralisation de ce problème.

I) Une illustration graphique.

Donner l'allure de la courbe d'une fonction f vérifiant les conditions $(*)$

II) Une solution polynomiale. Polynôme de Hermite.

1) Montrer que si P est une fonction polynomiale réelle vérifiant les conditions $(*)$ alors $\deg(P) \geqslant 4$.
2) Un calcul a permis de trouver une fonction polynomiale de degré 7 vérifiant les conditions $(*)$
   Parmi les fonctions polynômiales suivantes, seule une correspond à la solution du problème. Déterminer laquelle est la bonne.
   (On pourra procéder par élimination sachant qu'il y en a une de correct)

$$P_1 : x \longmapsto \frac{1}{16}\left(\frac{119}{324}x^2 - \frac{1}{2}x + 1\right)(x+1)^2(x-4)^3$$

$$P_2 : x \longmapsto \frac{1}{16}\left(-\frac{29}{216}x^3 + \frac{53}{108}x^2 - \frac{1}{2}x + 1\right)(x+1)^2(x-4)^2$$

$$P_3 : x \longmapsto \frac{1}{16}\left(-\frac{47}{648}x^4 + \frac{119}{324}x^3 + 2x^2 - \frac{1}{2}x + 1\right)(x+1)(x-4)^2$$

$$P_4 : x \longmapsto \frac{1}{16}\left(x^3 + x^2 - \frac{1}{2}x + 1\right)(x+1)^2(x-4)^2$$

3) Montrer qu'il n'y a pas d'autre polynôme de degré inférieur ou égal à 7 vérifiant les conditions $(*)$.
   Indication : On prendra P et Q deux fonctions polynomiales de degré inférieur ou égal à 7 et vérifiant les conditions $(*)$, puis on démontrera que cela entraîne $P - Q = 0$.

III) Une solution polynomiale par morceaux.

1) Déterminer une fonction polynomiale f_1 de degré inférieur ou égal à 3 vérifiant :

$$f_1(-1) = 0, \quad f_1'(-1) = 0, \quad f_1(0) = 1 \text{ et } \quad f_1'(0) = 1.$$

2) Déterminer une fonction polynomiale f_2 de degré inférieur ou égal à 3 vérifiant :

$$f_2(0) = 1, \quad f_2'(0) = 1, \quad f_2(2) = 2 \text{ et } \quad f_2'(2) = -1.$$

3) Déterminer une fonction polynomiale f_3 de degré inférieur ou égal à 3 vérifiant :

$$f_3(2) = 2, \quad f_3'(2) = -1 \ , \quad f_3(4) = 0 \text{ et } f_3'(4) = 0.$$

On définit alors la fonction f définie sur $[-1, 4]$ par :

si $x \in [-1, 0]$, $f(x) = f_1(x)$, si $x \in]0, 2]$, $f(x) = f_2(x)$ et si $x \in]2, 4]$, $f(x) = f_3(x)$.

4) Montrer que cette fonction f vérifie les conditions $(*)$.

5) Montrer que cette fonction f n'est pas de classe C^2 sur $[-1, 4]$.

IV) Généralisation :

Soit n un entier naturel non nul, $(n \geqslant 2)$,
on considère trois listes de réels : $(x_1, x_2, \ldots, x_n)$, $(y_1, y_2, \ldots, y_n)$ et $(d_1, d_2, \ldots, d_n)$
et on suppose que $x_1 < x_2 < x_3 < \cdots < x_n$ (les x_i sont dans l'ordre strictement croissant).
On cherche une fonction f dérivable sur $\mathbb{R}$ et vérifiant : $\forall i \in [\![1; n]\!], \quad f(x_i) = y_i \quad \text{et} \quad f'(x_i) = d_i$

1) Faites une représentation graphique illustrant cette introduction.

2) Montrer que la matrice suivante est inversible : $M = \begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 0 & 1 & 2x_1 & 3x_1^2 \\ 0 & 1 & 2x_2 & 3x_2^2 \end{pmatrix}$

3) En déduire qu'il existe une unique fonction polynomiale f_1 de degré inférieur ou égal à 3 vérifiant :

$$f_1(x_1) = y_1, \qquad f_1'(x_1) = d_1, \qquad f_1(x_2) = y_2 \qquad \text{et} \qquad f_1'(x_2) = d_2$$

4) Décrire un moyen de construire une fonction f simple vérifiant les conditions données en introduction de cette partie.

V) Un programme Python

Vous trouverez à la page suivante un programme Python qui utilise la méthode décrite dans la question **IV)**.
Vous pouvez voir en annexe ce qu'affiche ce programme.

1) Que fait la fonction `fonction_polynomiale` ? Donner un exemple d'appel à cette fonction.

2) Que fait la fonction `spline` ? *(Une explication de la fonction `linalg.solve` est en page 4)*

3) Que fait la fonction `trace` ?

4) Quel est le type de la variable `L` ? Que représente les valeurs contenues dans `L` ?

5) Compléter le document donné en annexe :
- en traçant les axes,
- en plaçant les points importants et les tangentes en ces points,
- en indiquant les courbes représentatives des fonctions f, f_1, f_2 et f_3 :

$$C_f \qquad C_{f_1} \qquad C_{f_2} \qquad C_{f_3}$$

- en indiquant C_P la courbe représentative du polynôme de Hermite de la question **II)** 2)

```
import numpy as np
import matplotlib.pyplot as plt

couleur = ['r--','c--', 'b--']
a, b, N = -1, 4, 100
L = [(-1, 0, 0), (0, 1, 1), (2, 2, -1), (4, 0, 0)]

def spline(pt1, pt2):
    x1, y1, d1 = pt1
    x2, y2, d2 = pt2
    A = [[1, x1, x1**2, x1**3],
         [1, x2, x2**2, x2**3],
         [0, 1, 2*x1, 3*x1**2],
         [0, 1, 2*x2, 3*x2**2]]
    B = [y1, y2, d1, d2]
    return np.linalg.solve(A, B)

def fonction_polynomiale(p):
    def f(x):
        S = 0
        for k in range(len(p)):
            S += p[k]*x**k
        return S
    return f

def trace(f, a, b , c):
    x = [ a + k*(b-a)/N for k in range(N+1) ]
    y = [ f(t) for t in x]
    plt.plot(x, y, c)

def g(x):
    Q = -29*x**3/3456 + 53*x**2/1728 - x/32 + 1/16
    return (x + 1)**2 * (x - 4)**2 * Q

plt.subplot(221)
for i in range(len(L)-1):
    p = spline(L[i], L[i+1])
    f = fonction_polynomiale(p)
    trace(f, L[i][0], L[i+1][0], couleur[i%3])
trace(g, a, b , 'k')
plt.ylim(-0.5,2.5)

plt.subplot(222)
for i in range(len(L)-1):
    p = spline(L[i], L[i+1])
    f = fonction_polynomiale(p)
    trace(f, a, b, couleur[i%3])
plt.ylim(-0.5, 2.5)

plt.subplot(223)
for i in range(len(L)-1):
    p = spline(L[i], L[i+1])
    f = fonction_polynomiale(p)
    trace(f, L[i][0], L[i+1][0], 'k')
plt.ylim(-0.5, 2.5)

plt.subplot(224)
trace(g, a, b , 'k')
plt.ylim(-0.5,2.5)

plt.show()
```

Extrait du manuel de la bibliothèque `numpy` a propos de la fonction `linalg.solve` :

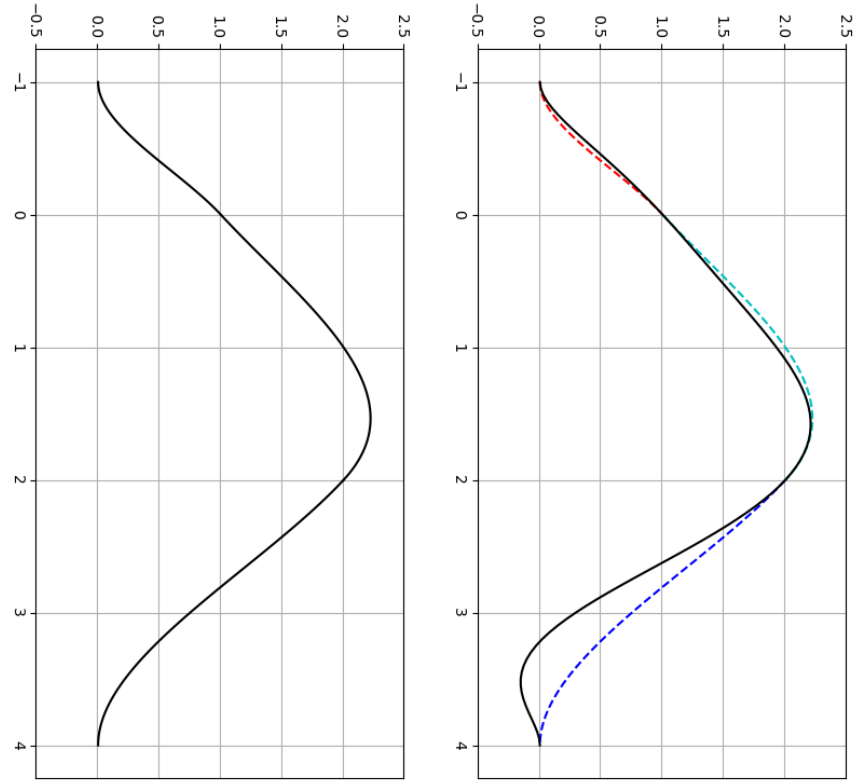
```
numpy.linalg.solve(a, b)
```

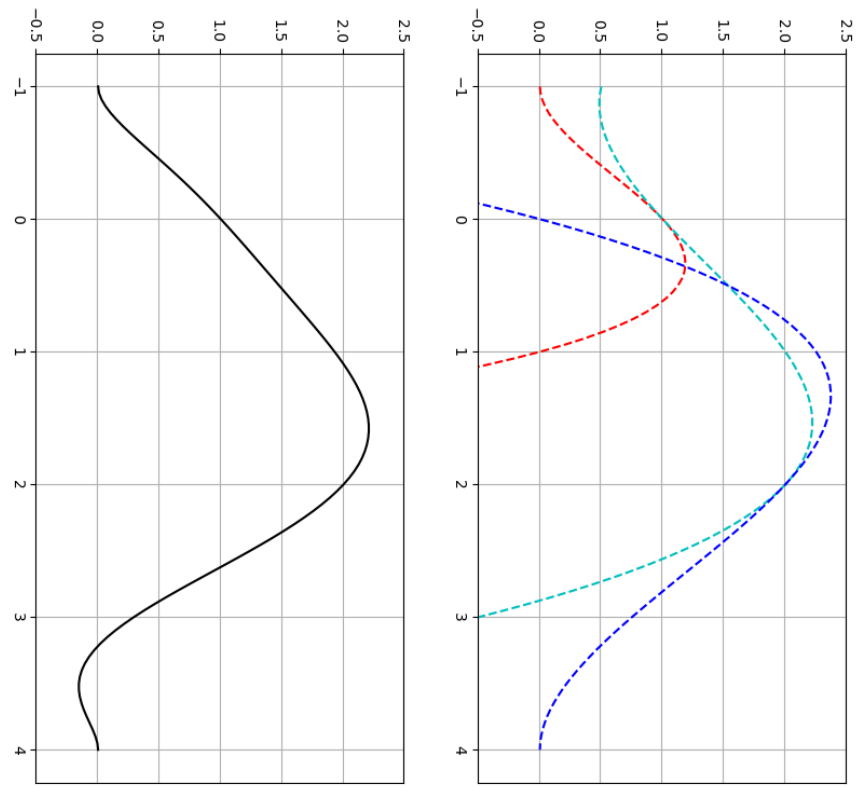
Solve a linear matrix equation, or system of linear scalar equations.

```
Parameters : a : (..., M, M) array like
                 Coefficient matrix.
             b : {(..., M,), (..., M, K)}, array like
                 Ordinate or "dependent variable" values.
Returns :    x : {(..., M,), (..., M, K)} ndarray
                 Solution to the system a x = b. Returned shape is identical to b.
Raises :     LinAlgError
                 If a is singular or not square.
```

Example : Solve the system of equations $3x_0 + x_1 = 9$ and $x_0 + 2x_1 = 8$:

```
>>> a = np.array([[3,1], [1,2]])
>>> b = np.array([9,8])
>>> x = np.linalg.solve(a, b)
>>> x
array([ 2.,  3.])
```


−0.5 0.0 0.5 1.0 1.5 2.0 2.5
−1 0 1 2 3 4


−0.5 0.0 0.5 1.0 1.5 2.0 2.5
−1 0 1 2 3 4