Série 5 (Corrigé)

Partiellement en classe vendredi

Exercice 1

Nous avons des mesures $y_0, y_1, \ldots, y_n$ correspondant à des points $x_0, x_1, \ldots x_n$. Montrer que la droite de régression $y = a_0 + a_1 x$, qui approche au sens des moindres carrés ces données, passe par le baricentre $(\bar{x}, \bar{y})$ du nuage de points (x_i, y_i) :

$$\bar{y} = a_0 + a_1 \bar{x}, \qquad \text{où } \bar{x} = \frac{1}{n+1}\sum_{i=0}^{n} x_i \text{ et } \bar{y} = \frac{1}{n+1}\sum_{i=0}^{n} y_i.$$

Sol. : *Les coefficients de la droite de régression $y = a_0 + a_1 x$ sont donnés par la solution du système linéaire $B^T B\mathbf{a} = B^T\mathbf{y}$, i.e.,*

$$\begin{pmatrix} n+1 & \sum_{i=0}^{n} x_i \\ \sum_{i=0}^{n} x_i & \sum_{i=0}^{n} x_i^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^{n} y_i \\ \sum_{i=0}^{n} y_i x_i \end{pmatrix}. \tag{1}$$

En particulier, si on considère la première équation de ce système on a

$$(n+1)a_0 + \left(\sum_{i=0}^{n} x_i\right) a_1 = \sum_{i=0}^{n} y_i.$$

Si on divise par $n+1$ les deux membres on trouve

$$a_0 + a_1 \left(\frac{1}{n+1}\sum_{i=0}^{n} x_i\right) = \left(\frac{1}{n+1}\sum_{i=0}^{n} y_i\right)$$

c'est-à-dire

$$a_0 + a_1 \bar{x} = \bar{y},$$

donc la droite de régression passe par le barycentre des données.

Exercice 2

On considères la rèaction chimique suivante :

$$OH_{(g)} + Cl\,C\,H_2\,C\,H_2\,Cl_{(g)} \to H_2O_{(g)} + Cl\,C\,H\,H\,C\,H_2\,Cl_{(g)}. \tag{2}$$

Pour ètudier la vitesse de cette rèaction chimique considèrer la loi de Arrhenius :

$$k = A\,e^{-\frac{E_a}{RT}}, \tag{3}$$

k [dm^3 mol^{-1} s^{-1}]	1.24	1.32	1.81	2.08	2.29	2.75
T [K]	292	296	321	333	343	363

TABLE 1 – Valeurs expèrimentales de T et k.

où $R = 8.3144621$ J K^{-1} mol^{-1} est la constante universelle des gaz parfaits. Soient les données suivantes (Tableau 1) qui correspondent aux valeurs expèrimentales de T et k :

Dèterminer les valeurs des constantes A (facteur prè-exponentiel) et E_a (l'ènergie d'activation) de la loi d'Arrhenius à l'aide de la mèthode des moindres carrès.

Conseil : Il faut d'abord prendre le logarithme de (3) *pour obtenir une relation linèaire entre* $\ln A$ *et* E_a*. Les calculs se font à l'aide de Python.* ***Sol. :*** *Il faut d'abord prendre le logarithme de l'équation d'Arrhenius pour obtenir une relation linéaire entre* $\ln A$ *et* E_a *:*

$$\ln k = \ln A - \frac{E_a}{R}\frac{1}{T} \tag{4}$$

Donc, l'équation d'Arrhenius peut être réécrite comme :

$$y = mx + q \tag{5}$$

où $y = \ln k$*,* $x = \frac{1}{T}$*,* $q = \ln A$ *et* $m = -\frac{E_a}{R}$*. Calculons les valeurs de* q *et* m *avec Matlab à l'aide de la méthode des moindres carrés.*

```
close all; clear all

T = np.array([292, 296, 321, 333, 343, 363])
k = np.array([1.24, 1.32, 1.81, 2.08, 2.29, 2.75])

Tinv = 1. / T;
lnk = np.log(k);

B = np.zeros((T.size,2))
# least squares.
for i in range(T.size) :
    B[i,:] = [1, Tinv[i]];

a = np.linalg.solve( B.T.dot(B), B.T.dot(lnk))

m = a[1]; q = a[0];

# plotting the data and the fitting
x = np.linspace(np.min(Tinv),np.max(Tinv),500)
y = m*x + q;
x = 1. / x;
y = np.exp(y);

plt.plot(T,k,'*',x,y,'r')
plt.xlabel('$T$'); plt.ylabel('$k$');
```

```
plt.legend([ 'data ', '$\hat p_n$ '])
plt.show()
```

Donc, on trouve $A = \exp(q)$ *et* $E_a = -m\,R$ *:*

```
print(f 'Energie d\'activation: {-m*8.3144621:.3f} ')
print(f 'Facteur pre-exponentiel: {np.exp(q):.3f} ')
```

$A = 73.7423$ *et* $E_a = 9903.6$.

Python

Exercice 3

Depuis https://hsso.ch/fr/2012/b/14 téléchargez les données de la population suisse dans le fichier `Data/PopulationSuisse.csv` :

Année	1860	1870	1880	1888	1900	1910	1920
Population	2506784	2654394	2924702	2917754	3315443	3753293	3880320
Année	1930	1941	1950	1960	1970	1980	1990
Population	4066400	4265703	4714992	5429061	6269783	6365960	6873687

Approximez l'évolution de la population avec un polynôme de degré $n = 1, 2, 3, 7$.

Ensuite faites l'hypothèse de croissance exponentielle de la population, c'est-à-dire $p(x) = Ce^{a_1 x}$ où C et a_1 sont des paramètres. Comment utiliser l'approximation polynômiale dans ce contexte ? Calculez les valeurs de C et a_1. *Indications : Ici la population est positive, donc* $C > 0$. *On peut donc remplacer* C *avec* e^{a_0} *dans* $p(x)$. *Un polynome* $a_0 + a_1 x$ *apparait. Comment le retrouver dans les données ?*

Voici quelques commandes utile en Python

```
import pandas as pd

# Read data from file 'PopulationSuisse.csv'
data = pd.read_csv("Data/PopulationSuisse.csv")
# Preview the first 5 lines of the loaded data
data.head()

# load the data into numpy arrays :
x = data['Annee'].to_numpy()
y = data['Population'].to_numpy()
```

Attention, remplacer `data['Annee']` *par* `data['Année']`. *LaTeX ne permet pas de le faire dans ce code ...*

Sol. : *Pour construire la matrice de Vandermonde, vous pouvez utiliser la fonction*

```
def VandermondeMatrix(x, m=0):
    # Input
    # x : +1 array with interpolation nodes
```

```
    # m : degree of the polynomial. If empty, chooses m=size(x)-1
    # Output
    # Matrix of Vandermonde of size m x n
```

que vous pouvez importer avec la commande

```
from InterpolationLib import VandermondeMatrix
```

```
# importing libraries used in this book
import numpy as np
import matplotlib.pyplot as plt

# In my case, the InterpolationLib is in the parent directory,
# therfore have have to add the aprent directory to path :
import sys
sys.path.append('..')

from InterpolationLib import VandermondeMatrix
```

Exercice *Depuis https ://hsso.ch/fr/2012/b/14 on a téléchargé les données de la population suisse dans le fichier* `Data/PopulationSuisse.csv`.

Approximez l'évolution de la population avec un polynôme de degré $n = 1, 2, 3, 7$.

Ensuite faites l'hypothèse de croissance exponentielle de la population, c'est-à-dire $p(x) = e^{a_1 x + a_0}$ *où* a_0 *et* a_1 *sont des paramètres. Comment utiliser l'approximation polynômiale dans ce contexte ?*

```
# Data of the population has been dowloaded from https://hsso.ch/fr/2012/b/14
# into the file Data/PopulationSuisse.csv
import pandas as pd

# Read data from file 'PopulationSuisse.csv'
data = pd.read_csv("../Data/PopulationSuisse.csv")

# To preview the first 5 lines of the loaded data execute
# data.head()
```

```
x = data['Annee'].to_numpy()
y = data['Population'].to_numpy()

m = 2
B = VandermondeMatrix(x,m)

# compute coefficients
a = np.linalg.solve( B.T.dot(B), B.T.dot(y))

# print the coefficients on screen
print('The coefficients a_0, ..., a_n are')
```

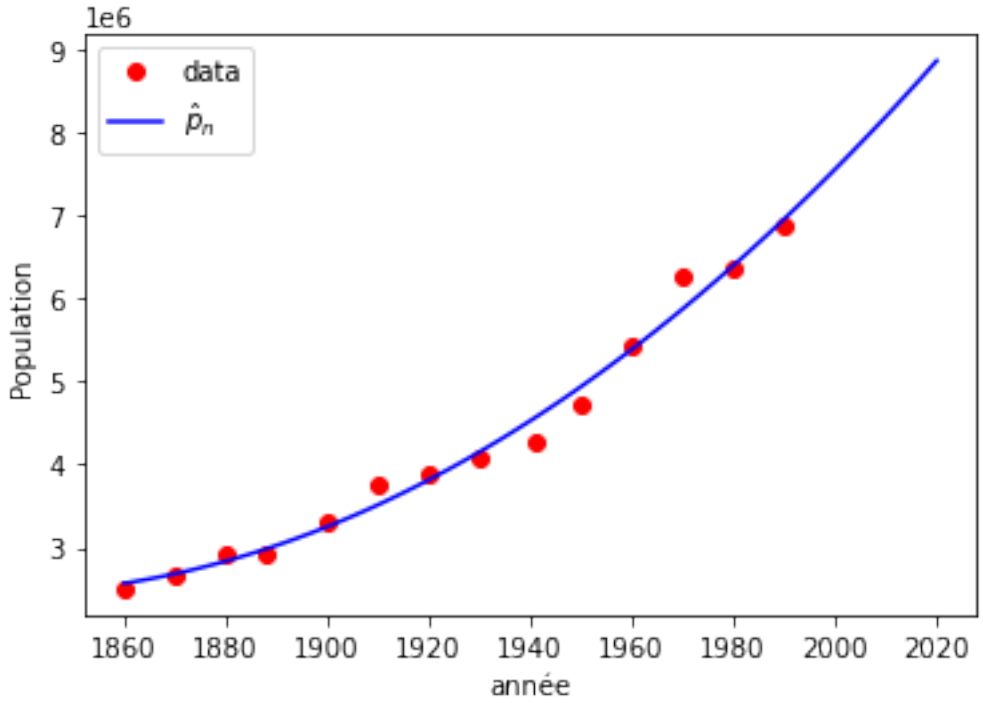

FIGURE 1 – png

```
print(a)
```

```
The coefficients a_0, ..., a_n are
[ 6.26809805e+08 -6.80846809e+05  1.85609737e+02]
```

```
def polynomial(a,x):
    m = a.size-1
    # \hat p = a_0 + a_1 x + ... + a_n x^m
    # is equal to the scalar product between the vectors a and (1, x, ...,
       x^m) :
    return np.power( np.tile(x, (m+1, 1)).T , np.linspace(0,m,m+1)).dot(a)

# points used to plot the graph, slightly larger than data
z = np.linspace(x[0], 2020, 100)

plt.plot(x, y, 'ro', z, polynomial(a,z), 'b')
plt.xlabel('annee'); plt.ylabel('Population');
plt.legend(['data', '$\hat p_n$'])
plt.show()
```

On assume une croissance exponentielle : $population(x) = C * e^{a_1 x} = e^{a_o + a_1 x}$
En d'autres termes : $\log(population)(x) = a_o + a_1 x$

```
#
```

```
x = data['Annee'].to_numpy()
y = np.log(data['Population'].to_numpy())


m = 1
B = VandermondeMatrix(x,m)

# compute coefficients
a = np.linalg.solve( B.T.dot(B), B.T.dot(y))

# print the coefficients on screen
print('The coefficients a_0, ..., a_n are')
print(a)
```

```
The coefficients a_0, ..., a_n are
[-0.01356599  0.00791249]
```

```
def expPolynomial(a,x):
    m = a.size-1
    # \hat p = a_0 + a_1 x + ... + a_n x^m
    # is equal to the scalar product between the vectors a and (1, x, ...,
       x^m) :
    return np.exp(np.power( np.tile(x, (m+1, 1)).T ,
       np.linspace(0,m,m+1)).dot(a))

# points used to plot the graph, slightly larger than data
z = np.linspace(x[0], 2020, 100)

plt.plot(x, np.exp(y), 'ro', z, expPolynomial(a,z), 'b')
plt.xlabel('annee'); plt.ylabel('Population');
plt.legend(['data', '$\hat p_n$'])
plt.show()
```

Exercices supplémentaires

Exercice 4

On considère la fonction

$$f(x) = 1 + \frac{1}{2}x + \sin(x).$$

Remarquez que cette fonction est la somme d'une fonction linéaire $f_0(x) = 1 + \frac{1}{2}x$ et d'une perturbation périodique $f_p(x) = \sin(x)$. Calculer la droite de régression qui approche f lorsque on prend les échantillons $x_i = 2i\pi/3$, $y_i = f(x_i)$, pour $i = 0, 1, 2, 3$. Tracer un graphe qualitatif de la fonction f, de la droite f_0 et de la droite de régression (en utilisant éventuellement le résultat du point a). Est-ce que cette droite coïncide avec f_0 ? Discutez brièvement les différences.

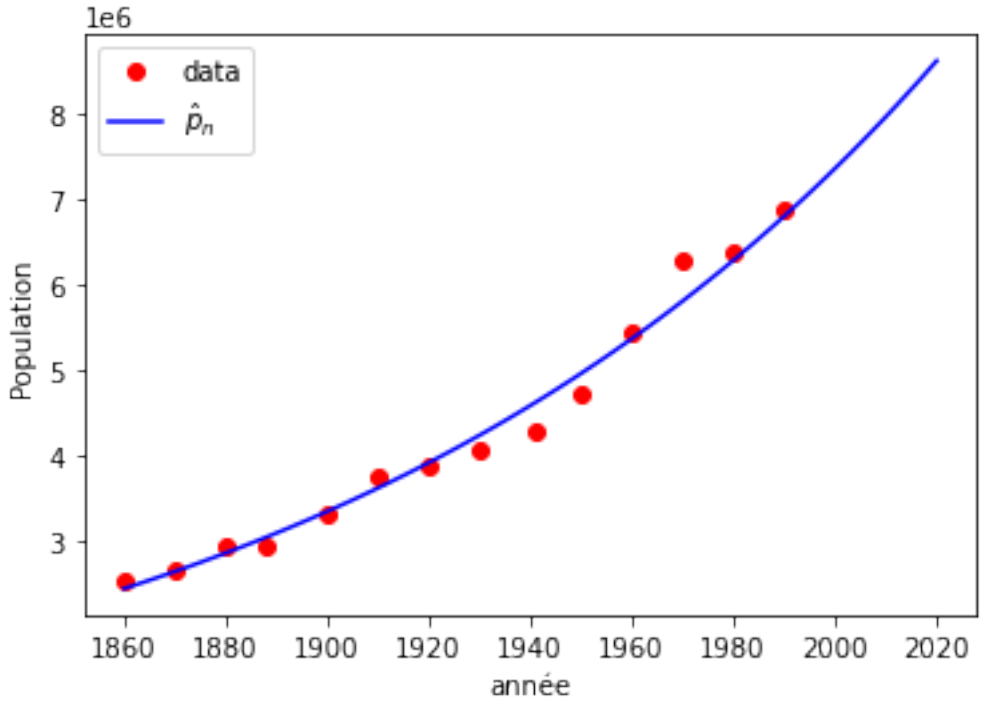

FIGURE 2 – png

Sol. : *On calcule :*

$$y_0 = 1, \quad y_1 = 1 + \frac{2\pi}{6} + \frac{\sqrt{3}}{2}, \quad y_2 = 1 + \frac{4\pi}{6} - \frac{\sqrt{3}}{2}, \quad y_3 = 1 + \pi.$$

En plus :

$$\sum_{i=0}^{n} x_i = 4\pi, \quad \sum_{i=0}^{n} x_i^2 = \frac{56}{9}\pi^2,$$

et

$$\sum_{i=0}^{n} y_i = 4 + 2\pi, \quad \sum_{i=0}^{n} x_i y_i = 4\pi + \frac{28\pi^2}{9} - \frac{\pi}{\sqrt{3}}.$$

L'équation normale devient

$$\begin{pmatrix} 4 & 4\pi \\ 4\pi & \frac{56}{9}\pi^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} 4 + 2\pi \\ 4\pi + \frac{28\pi^2}{9} - \frac{\pi}{\sqrt{3}} \end{pmatrix}$$

C'est un système de taille deux et donc on peut le résoudre à la main + calculatrice. Si on multiplie la première équation fois π et ensuite on la soustrait à la deuxième pour éliminer a_0, on calcule

$$a_1 \simeq 0.4173.$$

Puis, on utilise la première équation pour calculer a_0 :

$$a_0 \simeq 1.2598.$$

On voit donc que la droite de régression $y = 1.2598 + 0.4173x$ ne coïncide pas avec la partie linéaire $f_0(x) = 1 + 0.5x$. Par exemple, la pente de la droite de régression est un peu plus

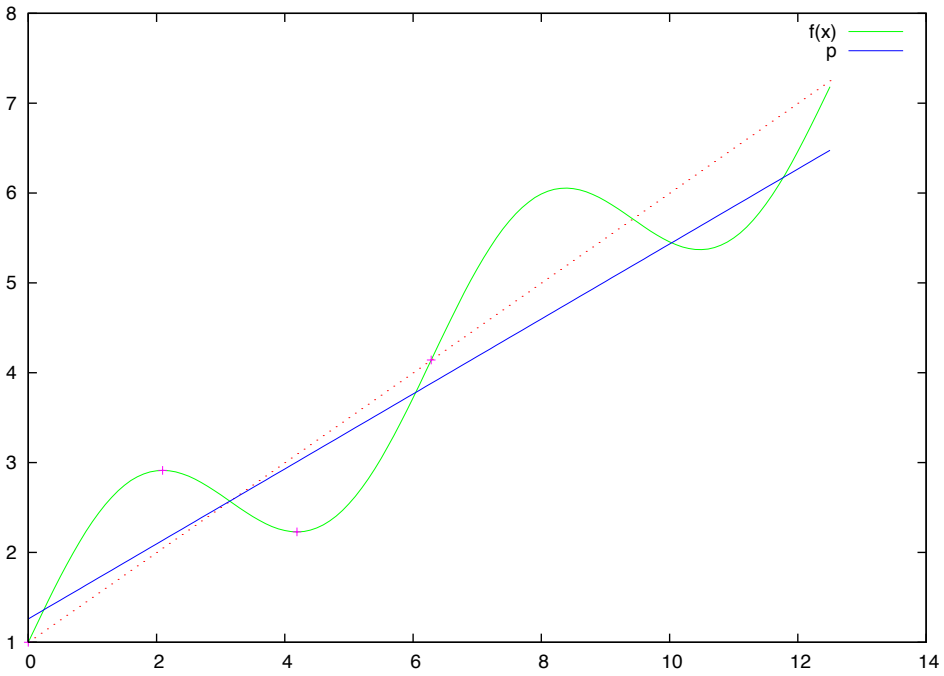

FIGURE 3 – Fonction et droite de regression

grande que 0.4 *alors que celle de* f_0 *est* 0.5. *Les graphes sont montrés en figure (exercice : essayez de les obtenir en Octave).*

On remarque que la droite de régression (ligne continue, en bleu) passe par le barycentre $(\bar{x}, \bar{y}) = (\pi, 1 + \pi/2)$ *des quatre points donnés dans le plan (marqués avec des "+"), qui est aussi un point du graphe de* f *et de la partie linéaire* f_0 *(ligne pointillée, en rouge).*

Le fait que la droite de régression ne coïncide pas avec la partie linéaire f_0*, qui est censée être la "droite qui approche mieux la fonction* $f(x)$*", est dû à la forme du fonctionnel dont le minimum correspond au coefficients de la droite de régression : ce fonctionnel est en fait*

$$\Phi(a_0, a_1) = \sum_{i=0}^{n} (a_0 + a_1 x_i - f(x_i))^2,$$

et dépend que des $n+1$ *points* $(x_i, f(x_i))$ *donnés. On a vu au cours que les coefficients* (a_0, a_1) *de la droite de régression sont les valeurs où* Φ *atteint son minimum. Notre droite de régression est la droite qui approche au mieux l'ensemble fini des (quatre) points* $\{(x_i, f(x_i))\}_{i=0,\dots,n}$*, non pas l'ensemble infini* $\{(x, f(x))\}_{x\in[0,\pi]}$*. Si par contre on minimise par rapport à* (a_0, a_1) *le fonctionnel*

$$\tilde{\Phi}(a_0, a_1) = \int_0^{2\pi} (a_0 + a_1 x - f(x))^2 dx,$$

on trouve que le minimum est donné par les coefficients $(a_0, a_1) = (1, 1/2)$*, correspondant à la droite* f_0*. En effet,* $\tilde{\Phi}$ *tient en compte de* tous *les points* $\{(x, f(x))\}_{x\in[0,\pi]}$*.*

En d'autres mots, on obtient des approximations différentes avec des différents choix du fonctionnel à minimiser, il est donc naturel que la droite de régression ne corresponde pas à f_0*. Pourtant, on a vu que la différence n'est pas énorme : en plus, on peut vérifier (en*

Octave) que si l'on augmente le nombre de points, l'écart tend vers zéro : cela est raisonnable car $\frac{1}{n}\Phi(a_0, a_1)$ *tend vers le fonctionnel intégral* $\tilde{\Phi}(a_0, a_1)$.

Exercice 5

On considère la fonction $f(x) = \sin(x)$ définie sur l'intervalle $[0, 3\pi]$.

1. En utilisant Python, calculer le polynôme d'interpolation $\Pi_n f$ de la fonction $f(x) = \sin(x)$ pour une distribution de nœuds uniforme dans $[0, 3\pi]$, dans les cas $n = 1, \ldots, 5$ (n étant le degré du polynôme d'interpolation). Comparer graphiquement le résultat avec la fonction donnée [1].
2. Evaluer l'erreur $\max_{x\in[0,3\pi]} |\sin(x) - \Pi_n \sin(x)|$ et visualiser le graphe de E_n en fonction de n.
3. En observant que $|\sin^{(n)}(x)| \leq 1$, $\forall x \in [0, 3\pi]$, $\forall n$, comparer l'erreur obtenue au point b) avec l'estimation théorique donnée au cours :

$$\max_{x\in I} |E_n f(x)| \leq \frac{1}{4(n+1)} \left(\frac{b-a}{n}\right)^{n+1} \max_{x\in[a,b]} |f^{(n+1)}(x)| \, .$$

4. En utilisant la fonction `PiecewiseLinearInterpolation` calculer le polynôme linéaire par intervalles $\Pi_1^H \sin(x)$, sur N sous-intervalles de longueur $H = 3\pi/N$. Considérer $N = 2, 4, 6, 8, 10$ et comparer graphiquement les résultats obtenus avec la fonction donnée[1].
5. Evaluer l'erreur $\max_{x\in[0,3\pi]} |E_n f(x)| = \max_{x\in[0,3\pi]} |\sin(x) - \Pi_1^H \sin(x)|$ et visualiser le graphe de $\max_{x\in[0,3\pi]} |E_n f(x)|$ en fonction du nombre d'intervalles N.

Sol. : *(La correction n'est pas à jour, elle est écrite en Matlab)*

1. *Il faut se rappeler que dans Matlab/Octave, n'importe quel polynôme* $p(x) = a_0 + a_1 x + \ldots + a_n x^n$ *de degré* n *est representé par le vecteur* `p = [an, ..., a1, a0]` *des* $n+1$ *coefficients de* p *(attention : le premier élément du vecteur correspond au terme de plus haut degré). En outre,*
   - *la commande* `polyfit` *sert à calculer le vecteur des coefficients* a_i *d'un polynôme* $p(x)$ *interpolant des données (fitting) ;*
   - *la commande* `polyval` *permet d'évaluer n'importe quel polynôme* $p(x)$ *en un point (ou un vecteur de points)* x*, à partir du vecteur des coefficients* a_i *(par example, le vecteur* `p = [1, 2, 3]` *définit le polynôme* $p(x) = x^2 + 2x + 3$*, donc la valeur retournée par* `polyval(p,1)` *est 6).*

On définit la fonction $\sin(x)$ *et les vecteurs suivants :*

```
fun = lambda x: np.sin(x)

poly = []
for i in range(1,6): # 2 to 6 points
    xp = np.linspace(0, 3*np.pi, i+1)
    an = np.polyfit(xp,fun(xp),i)
```

1. Utiliser au moins 100 points pour les représentations.

```
    poly.append( an )

x = np.linspace(0, 3*np.pi, 100)
plt.plot(x, fun(x) , label = '$f$' , linewidth = 3.0)

for an in poly:
    plt.plot(x, np.polyval(an,x) , label = 'n=' + str(len(an)))

plt.xlabel('x'); plt.legend()
```

2. *Pour évaluer l'erreur et en tracer le graphe, on utilise les commandes suivantes (prochain point)*
3. *On peut calculer les estimations théoriques grâce aux commandes suivantes :*

```
Error = [np.max(np.abs(np.polyval(p,x) - fun(x))) for p in poly]
Estimation = [ (3*np.pi/n)**(n+1)/(4*(n+1)) for n in range(1,6) ]

print(Error)
plt.plot(np.arange(1,6), Error, 'o-', label = 'error')
plt.plot(np.arange(1,6), Estimation, '--', label = 'estimation')
plt.xlabel('n')
plt.ylabel('error')
```

On voit bien que E_n est toujours plus petite que son estimation théorique.

4.

```
# importing PiecewiseInterpolation
from InterpolationLib import PiecewiseLinearInterpolation as  PiH1

# interval and function
a = 0; b= 3*np.pi

# Values of N to use
Nrange = [2,4,6,8,10]
# plotting points
z = np.linspace(a, b, 100)

for N in Nrange :
    plt.plot(z, PiH1(a,b,N,fun,z), ':')

plt.plot(z, fun(z), 'b')
plt.xlabel('t'); plt.ylabel('y'); #plt.title('data')
plt.legend(Nrange)
plt.show()
```

5. *On peut calculer l'erreur grâce aux commandes :*

```
# erreur calculee en 100 points
z = np.linspace(a, b, 100)
```

```
Error = [np.max(np.abs(PiH1(a,b,N,fun,z) - fun(x))) for N in Nrange]

print(Error)
plt.plot(Nrange, Error, 'o-', label = 'error')
plt.xlabel('n')
plt.ylabel('error')
plt.legend()
#plt.xscale('log')
plt.yscale('log')
```

Exercice 6

Soit $f(x) = e^{-x^2/2}$. On divise l'intervalle $[0, 5]$ en M sous-intervalles de longueur constante $H = 5/M$. Soit $\Pi_1^H f(x)$ le polynôme par morceaux de degré 1 sur chaque sous-intervalle interpolant f.

a) Majorer l'erreur $E_H = \max_x |f(x) - \Pi_1^H f(x)|$ en fonction de H.

b) Trouver le nombre M de sous-intervalles nécessaires pour que $E_H \leq 0.5 \cdot 10^{-6}$.

Sol. :

a) On a l'estimation suivante de l'erreur d'interpolation :

$$\max_{x\in[a,b]} |f(x) - \Pi_1^H f(x)| \leq \frac{1}{8} H^2 \max_{x\in[a,b]} |f''(x)|.$$

On calcule

$$f'(x) = -xe^{-x^2/2}, \quad f''(x) = (-1 + x^2)e^{-x^2/2}, \quad f'''(x) = x(3 - x^2)e^{-x^2/2},$$

Étudions $g = f''$: $g(x) = 0$ si et seulement si $x = 0, -\sqrt{3}, \sqrt{3}$. On a que $\lim_{x\to\infty} g(x) = 0$ et que $g(0) = -1$ et $g(1) = 0$. On en déduit que g est croissante entre 0 et $\sqrt{3}$ et décroissante ensuite, mais toujours positive. Les extrema de g sont en $x = 0$ et $x = \sqrt{3}$.

$$\max_{x\in[0,5]} |f''(x)| = \max\{|g(0)|, |g(\sqrt{3})|\} = \max\{1, 2e^{-3/2}\} = 1,$$

puisque $e^{3/2} > e > 2.7$.
L'estimation de l'erreur sur $[0, 5]$ devient

$$\max_{x\in[0,5]} |f(x) - \Pi_1^H f(x)| \leq \frac{1}{8} H^2 = \frac{5^2}{8M^2},$$

b) et donc pour avoir $5^2/(8M^2) \leq 10^{-6}/2$ il faut $4M^2 \geq 5^2 \cdot 10^6$ et donc

$$M \geq 2500,$$

c'est-à-dire le nombre de sous-intervalles nécessaires est 2500.
