TP N°19 : Simulation d'expériences et traçage de courbes
Le module `Matplotlib`

Ce TP a pour objectif d'apprendre à utiliser le module `Matplotlib` de Python grâce à des exemples à saisir dans la console (sous IDLE) et quelques exercices d'applications.

Présentation

Le module `Matplotlib` est chargé de tracer les courbes. Dans ce TP on donne un bref aperçu des possibilités graphiques de `Matplotlib`. Il est possible de générer des graphiques à deux et à trois dimensions, et d'agir facilement sur les attributs du graphe (couleurs, type de ligne, axes, annotations...).
On importera la bibliothèque `Matplotlib` au moyen de la commande : `import matplotlib.pyplot as plt`

```
>>> import matplotlib.pyplot as plt
```

Les commandes présentées ci-dessous sont utiles notamment pour être introduites dans des scripts et automatiser la production de figure.

La fonction `plot()`

D'une manière générale les fonctions `plt.plot` attendent des tableaux de points du plan. Selon les options, ces points du plan sont reliés entre eux de façon ordonnée par des segments : le résultat est une courbe.
La fonction `plot(x,y)` permet de tracer une courbe reliant les points dont les **abscisses** sont données dans le vecteur **x** et les **ordonnées** dans le vecteur **y**. Commençons par la fonction *sinus* :

```
import matplotlib.pyplot as plt
import numpy as np
from math import pi
x= np.arange(0,2*pi,pi/16) # ou x= np.linspace(0,2*pi,100) :  échantillonner la variable x
plt.plot(x,np.sin(x),label="sin") # on utilise la fonction sinus de Numpy
plt.axis("equal") # Imposer une échelle identique sur les deux axes de coordonnées.
plt.xlim(-2,10) # Fixer l'intervalle de visualisation sur l'axe des abscisses.
plt.ylabel('fonction sinus')
plt.xlabel("l'axe des abcisses")
plt.title('Fonction sinus')
plt.savefig('ma_figure.png') # sauvegarder la figure (en .png ou .pdf)
plt.show() #Afficher l'image
plt.clf()
```

Si tout se passe bien, une fenêtre doit s'ouvrir avec la figure ci-dessus.

Exercice 1

1. Écrire un script qui demande à l'utilisateur d'introduire des nombres $x_1, y1, x_2, y_2, x_3, y_3$ et trace le triangle ABC avec $A(x_1, y_1), B(x_2, y_2)$ et $C(x_3, y_3)$.
2. Tracer la courbe représentative de $x \longmapsto cos(x)$ sur l'intervalle [-5, 5] à l'aide de la fonction `plot()`.
   Pour utiliser la fonction `plot()` de `matplotlib.pyplot`, on pourra exécuter les instructions :

   ```
   X = arange(-5,5,0.1)
   Y = cos(X)
   ```

Manipuler plusieurs graphiques

On peut utiliser plusieurs fenêtres graphiques en même temps, à condition de stoker les figures dans des variables :

```
t = np.linspace(0, 2*pi, 50)
fig1 = plt.figure()
plt.plot(t, np.cos(t), label="cos")
fig2 = plt.figure()
plt.plot(t, np.sin(t), label="sin")
plt.show() #Afficher toutes les figures
```

Une même fenêtre peut intégrer plusieurs graphes non superposés grâce à la commande `subplot(n, m, k)` qui permet de subdiviser la fenêtre en $n \times m$ cases, n lignes et m colonnes (comme pour les matrices).
Ces cases sont numérotées de gauche à droite et de haut en bas. La valeur de k permet de spécifier dans quelle case on désire faire un graphique.

```
plt.subplot(2, 1, 1)
plt.plot(t, np.cos(t))
plt.subplot(2, 1, 2)
plt.plot(t, np.sin(t))
plt.show() #Afficher toutes les figures
```

Il est possible de superposer deux courbes sur le même graphique :

```
plt.plot(t, np.cos(t), t,np.sin(t))
plt.show() #Afficher toutes les figures
```

Graphiques à 3 dimensions : Les courbes

Il est possible de tracer des courbes dans l'espace avec la librairie Axes3D. Par exemple une hélice :

```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.gca(projection='3d')
plt.plot(np.cos(t), np.sin(t), t, label="helice")
plt.show()
```

Autres options

Pour connaître toutes les options, le mieux est de se référer à la documentation de `Matplotlib`. Voyons ici quelques unes d'entre elles :

- **Ajouter un titre :** `plt.title("Mon titre")`
- **Légendes :** `plt.legend((g1, g2), ("ligne2","ligne 1"))`.
- **Épaisseur du trait :** `linewidth=flottant`
- **Style du trait :** se décide avec `linestyle` : '-' ligne continue, '–' tirets, '-.' points-tirets, ' :' pointillés, ...
- **Couleur du trait :** pour changer la couleur du tracé une lettre g vert (green), r rouge (red), k noir, b bleu, c cyan, m magenta, y jaune (yellow), w blanc (white).
- **Bornes :** spécifier un rectangle de représentation, ce qui permet un zoom, d'éviter les grandes valeurs des fonctions par exemple, se fait via la commande `plt.axis([xmin,xmax,ymin,ymax])`
- **Symboles :** mettre des symboles aux points tracés se fait via l'option marker. Les possibilités sont nombreuses parmi ['+' | '*' | ',' | '.' | '1' | '2' | '3' | '4' | '<' | '>' | 'D' | 'H' | 'ˆ' | '_' | 'd' | 'h' | 'o' | 'p' | 's' | 'v' | 'x' | '|' | TICKUP | TICKDOWN | TICKLEFT | TICKRIGHT | 'None' | ' '].

Exercice 2

1. Réaliser le graphe de la fonction $y(t) = v_0 t - \frac{1}{2}gt^2$ pour $v_0 = 10$, $g = 9.81$, et $t \in [0, 2\frac{v_0}{g}]$.
   Le label sur l'axe des x devra être "temps (s)" et le label sur l'axe des y est "hauteur (m)".
2. La fonction $f(x,t) = e^{-(x-3t)^2} sin(3\pi(x-t))$ décrit pour une valeur fixe de t une onde localisée en espace. Faire un programme qui visualise cette fonction comme une fonction de x dans l'intervalle $x \in [-4,4]$ pour $t = 0$.