**TRAVAUX PRATIQUES n° 9**
**Outils graphiques**

# I. Se documenter sur une bibliothèque

Dans ce TP, nous aurons besoin d'utiliser deux bibliothèques importantes de Python : `numpy` et `matplotlib`

- ⋆ La bibliothèque `numpy` permet de faire du calcul scientifique et fourni également une nouveau type de données en Python appelé **tableau** (`array`).
- ⋆ La bibliothèque `matplotlib` permet de créer des graphiques ou des animations.

Pour utiliser ces bibliothèque, on écrira, **en début de TP uniquement et une seule et unique fois**, les lignes suivantes :

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

On ne peut pas connaitre par cœur toutes les fonctions utiles de ces deux bibliothèques. Il faut donc savoir se documenter pour

- ⋆ trouver une fonction inconnue qui permet de résoudre un problème ;
- ⋆ trouver à quoi sert une fonction dont on connait le nom ;
- ⋆ trouver la liste des paramètres (obligatoires et optionnels) d'une fonction et leurs rôles.

Exercice 1 (*Se documenter sur numpy*) :

1) Aller sur le site `https://numpy.org/doc/stable/` et taper dans la barre de recherche `linspace`. Cliquer sur le premier lien `numpy.linspace` et essayer de comprendre ce que fait cette fonction. En particulier
   a) À quoi servent les paramètres `start`, `stop`, `num` et `endpoint` ?
   b) Sont-ils obligatoires ou optionnels ?
   c) S'ils sont optionnels, quelle est leur valeur par défaut ?
2) Essayer la fonction `np.linspace` sur différents exemples (ceux de la documentation ou d'autres de votre choix).
3) Taper `help(np.linspace)` dans le shell Pyzo. Que constate-t-on ?

Exercice 2 (*Se documenter sur matplotlib*) : Aller sur le site `https://matplotlib.org/stable/` et cliquer sur *Reference* dans la barre de menus.

Dans la liste des modules, cliquer sur le module `matplotlib.pyplot`.

Dans la liste des fonctions, cliquer sur la fonction `plot`.

Pour cette fonction, il y a beaucoup trop d'informations et il est difficile de comprendre ce qu'elle fait et comment elle marche quand on débute mais on sait où trouver une information si besoin et on peut s'inspirer des exemples en fin de page.

Si vous voulez en apprendre plus sur ces bibliothèques, vous pouvez aussi consulter les tutoriels suivants :

- ⋆ `numpy` : `https://www.w3schools.com/python/numpy/default.asp`
- ⋆ `matplolib` : `https://www.w3schools.com/python/matplotlib_intro.asp`

## II. Deux exemples pour démarrer

On commence par définir une fonction :

```
def f(x):
    return x**3 + 3*x + 1
```

Puis on construit le graphique :

```
# Création d'une liste d'abscisses et des ordonnées correspondantes
abscisses = [-3, -2, 0, 1, 5, 6]
ordonnees = [f(x) for x in abscisses]

# fig = fenêtre qui sera affichée à l'écran avec un titre
fig = plt.figure("Exemple 1")

# On donne un titre au graphique et un nom aux axes
plt.title('Mon premier graphique')
plt.xlabel('x')
plt.ylabel('f(x)')

# On trace la courbe en reliant les points avec pleins d'options
plt.plot(abscisses, ordonnees, color = "red", linestyle = "dashed", linewidth = 2,
   marker = "*", markeredgecolor = "blue", markersize = 10)

# On ajoute une grille pour plus de lisibilité
plt.grid()

# On demande à Python d'ouvrir la fenêtre contenant le graphique
plt.show()
```

Pour avoir une courbe plus lisse, il faut ajouter plus d'abscisses ! On utilise la fonction `linspace` pour créer rapidement pleins d'abscisses entre deux valeurs :

```
# Création d'une liste d'abscisses et des ordonnées correspondantes
abscisses = np.linspace(-5, 5, num=100)
ordonnees = [f(x) for x in abscisses]   # ordonnees = f(abscisses) fonctionne aussi

# fig = fenêtre qui sera affichée à l'écran avec un titre
fig = plt.figure("Exemple 2")

# On donne un titre au graphique
plt.title('Mon deuxième graphique')

# On trace la courbe
plt.plot(abscisses, ordonnees, color='green', label="Fonction f")

# On impose une échelle pour les axes (Python peut les calculer automatiquement)
plt.xlim(-7, 7)
plt.ylim(-5, 85)

# On affiche la légende en haut à droite
plt.legend(loc='upper right')

# On ajoute une grille pour plus de lisibilité
plt.grid()

# On demande à Python d'ouvrir la fenêtre contenant le graphique
plt.show()
```

On pourra aussi regarder les autres options de la fonction `plt.plot`.

# III. Exercices

**Exercice 3** (*Des graphes de fonctions*) :

1) Définir les fonctions suivantes :
   a) $f : x \mapsto \dfrac{e^x + e^{-x}}{2}$
   b) $g : x \mapsto x + \sin x$
   c) $h : x \mapsto x \sin \dfrac{1}{x}$
2) a) Tracer le graphe de $f$ sur le segment $[-2, 2]$.
   b) Conjecturer les variations de $f$ et ses limites en $\pm\infty$.
   c) Ajouter l'instruction `plt.axis("equal")` pour que Python conserve la même échelle pour les deux axes.
3) a) Tracer les graphes de $f$ et de $g$ sur le segment $[-5, 5]$ dans une même fenêtre : celui de $f$ en bleu et en trait plein et celui de $g$ en rouge et en pointillés.
   b) Essayer d'ajuster les échelles en abscisses et en ordonnées grâce aux fonctions `plt.xlim` et `plt.ylim`.
   c) Ajouter une légende qui montre quelle courbe correspond à quelle fonction.
4) a) Tracer le graphe de $h$ sur le segment $[-2, 2]$ en utilisant 2000 points.
   b) Zoomer le graphique grâce aux fonctions `plt.xlim(-0.5, 0.5)` et `plt.ylim(-0.25, 0.25)`.

**Exercice 4** (*Des courbes paramétrées*) : On peut aussi tracer des courbes qui ne sont pas des graphes de fonctions en prenant une fonction pour les abscisses et une (autre) fonction pour les ordonnées.

Exécuter les commandes suivantes :

1)
```
T = np.linspace(0, 100, 500)
X = T * np.cos(T)
Y = T * np.sin(T)
plt.figure()
plt.plot(X,Y)
plt.show()
```

2)
```
T = np.linspace(0, 2 * np.pi, 500)
X = np.cos(T) * (1 - np.cos(T))
Y = np.sin(T) * (1 - np.cos(T))
plt.figure()
plt.plot(X,Y)
plt.show()
```

La courbe obtenue s'appelle la cardioïde.

**Exercice 5** (*Conjecture graphique*) : Le but de cet exercice est de déterminer la plus grande valeur de $\sqrt[n]{n}$ pour $n \in \mathbb{N}^*$.

1) Tracer le graphe de la fonction $x \mapsto x^{\frac{1}{x}}$ sur le segment $[1, 10]$.
2) Conjecturer la valeur de $n \in \mathbb{N}^*$ tel que $\sqrt[n]{n}$ est maximal.

**Exercice 6** (*Une fonction et sa réciproque*) :

1) Tracer sur le même graphique les courbes représentatives des fonctions tan sur l'intervalle $\left]-\dfrac{\pi}{2}, \dfrac{\pi}{2}\right[$ et arctan sur $[-5, 5]$ en utilisant deux couleurs différentes.
2) Ajouter la première bissectrice (d'équation $y = x$) en pointillés et d'une troisième couleur et ajouter une légende.

**Exercice 7** (*L'escalier ou l'escargot*) : Le but de cet exercice est d'étudier graphiquement la convergence de la suite $(u_n)_{n\geqslant 0}$ définie par $u_0 = a$, avec $a \geqslant -1$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = \sqrt{1+u_n}$.

1) Créer une nouvelle figure, appellée `"Suite récurrente"` et tracer en noir les deux droites d'équations $y = 0$ et $x = 0$ sur l'intervalle $[-1, 3]$ (les axes).
2) Tracer en pointillés et en vert la droite d'équation $y = x$.
3) Tracer en bleu le graphe de la fonction $f : x \mapsto \sqrt{1+x}$.
4) Écrire une fonction prenant en arguments un entier $n$ et un réel $a \geqslant -1$ qui construit deux listes de la manière suivante :
   - ⋆ Initialement : `X = [a]` et `Y = [0]`
   - ⋆ Pour `k` allant de `0` à `n-1` :
     - ajouter le dernier élément de `X` (c'est-à-dire `X[-1]`) à la fin de `X`
     - ajouter l'image du dernier élément de `X` à la fin de `Y`
     - ajouter le dernier élément de `Y` à la fin de `X`
     - ajouter le dernier élément de `Y` à la fin de `Y`
   - ⋆ Afficher enfin la ligne brisée d'abscisses `X` et d'ordonnées `Y` en rouge.
5) Conjecturer alors la monotonie de $(u_n)_{n\geqslant 0}$ et son éventuelle limite. Ces résultats dépendent-ils de la valeur de $a$ ?
6) Mêmes questions pour la suite $(v_n)_{n\geqslant 0}$ définie par $v_0 = a$, avec $a > -1$, et pour tout $n \in \mathbb{N}$, $v_{n+1} = \dfrac{1}{1+v_n}$.
   *On utilisera l'intervalle* $[-0.5, 2.5]$ *pour tracer les courbes dans ce cas.*