MATPLOTLIB ET LECTURE DE FICHIERS CSV

Tracés de base

Importer les modules `numpy` et `matplotlib` de la manière suivante

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

La fonction `plt.plot(X, Y)` permet de tracer une ligne polygonale des points (x_i, y_i) où `X=[`x_i`]` et `X=[`y_i`]`.
On peut préciser la couleur, si on veut seulement les points et sous quelle forme...
Par exemple `plt.plot(X,Y,'g-*')`...
On affiche alors le résultat avec `plt.show()`
Si on veut effacer la figure : `plt.clf()`
On peut préciser le titre, la légende des axes (`plt.xlabel(), plt.ylabel()`), une grille (`plt.grid()`).
Vous pouvez si le cœur vous en dit consulter les 1200 pages de documentation sur `matplotlib`...

Activité 1 : tracé de la courbe $y = \sin(x)$

On commence par créer une subdivision sous forme de liste. C'est une liste de flottants, on préfère travailler avec des listes de type `array` (du module `numpy` avec lequel travaille `matplotlib`). Pour cela on peut utiliser `arange` mais `linspace` est beaucoup plus adaptée.
`X = linspace(0, 5*np.pi, 201)` crée une subdivision de 0 à 5π (**compris**) avec 200 petits intervalles (ou 201 valeurs).
Le module `numpy` reprend les fonctions (et les opérations) mathématiques classiques (par exemple du module `math`) mais généralisé sous forme ‹‹ vectorisée ›› pour les `array`. Par exemple `Y = sin(X)**2` construit directement la liste de type `array [sin`^{2}`(`y_i`)]`.
Pour tracer la courbe $y = \sin(x)$, on écrira donc

```
X = np.linspace(0.0, 10.0, 200)
Y = np.sin(X)
plt.plot(X ,Y, 'b')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

▶ Tester ce programme.

Activité 2 : tracé de la courbe $y = xe^{-x}$

▶ Effectuer maintenant le tracé de la courbe représentative de la fonction $x \mapsto xe^{-x}$ sur $[0, 10]$.

Activité 3 : tracé du courbe paramétrée

▶ Tracer la jolie courbe paramétrée suivante

$$\begin{cases} x = \sin^3 t \\ y = \cos(t) - \cos^4 t \end{cases} \quad \text{pour } t \in [-\pi, \pi].$$

Vous pouvez rajouter la commande `plt.fill(X, Y, 'r')` pour remplir de rouge (**r**ed) cette courbe fermée.

Lecture d'un fichier de TP et tracé d'une courbe expérimentale

▶ Récupérer le fichier `mesureportailTD.csv` (et éventuellement le fichier `mesureportailTD.xls`) qui est un enregistrement de mesures effectuées lors d'un T.P. de S.I. de M. Levavasseur. C'est un fichier texte, les colonnes sont séparées par des ; et j'ai américanisé les flottants (. au lieu de ,). Lisez-le avec un éditeur de fichier texte (`Notepad++` par exemple).
▶ Lire ce fichier et mettre dans deux listes `X` $= [x_i]$ et `Y` $= [y_i]$ les colonnes correspondant aux **angles du bras** et aux **angles du vantail**. Tester le programme pour quelques lignes du tableau.
Pour rappel, voici le petit programme concernant les fichiers `.csv` présenté lors du cours ‹‹ manipulation de fichiers ››

```
liste = []
f = open('marchemodif.csv','r')
for L in f:
    fruit, region, prixstr = L.strip().split(';')
    prix = float(prixstr)
    liste.append({"fruit":fruit.strip(),
    "region":region.strip(),"prix":prix})
f.close()
print(liste)
```

On lit une ligne avec l'instruction `f.readline()`.

▶ Visualiser le nuage de points (x_i, y_i). En théorie, ces points sont tracés sur une courbe d'une fonction théorique étudiée en S.I.I.

Il existe bien sûr des modules spécialisés dans la lecture de fichiers `.csv` ou même `.xls` (`excel`)...

Suites adjacentes et visualisation

On pose $S_n = \sum_{k=1}^{n} \frac{(-1)^k}{\sqrt{k}}$; on montre que (S_{2n}, S_{2n+1}) est un couple de suites adjacentes.

▶ Visualiser les 100 premiers termes des suites (S_{2n}) et (S_{2n+1}).

▶ Donner une valeur approchée de $\lim_{n\to+\infty} S_n$ à 10^{-3} près.

Construction graphique de $u_{n+1} = f(u_n)$

On considère la fonction $f_\mu : \left\{ \begin{array}{ccc} [0,1] & \longrightarrow & \mathbb{R} \\ x & \longmapsto & \mu x(1-x) \end{array} \right.$.

▶ Tracer le graphe de la fonction f_μ pour $\mu = 3.7$ et la première bissectrice avec des couleurs différentes. Choisir un valeur x au hasard entre 0 et 1.

```
from random import random
x = random()
```

On construit alors la ligne polygonale

```
plt.plot([x, x], [0, x], 'k')# k pour black
```

▶ Poursuivre la construction, disons une vingtaine de fois pour montrer la construction en escalier ou spirale de la suite définie par

$$\left\{ \begin{array}{l} u_0 = x \\ u_{n+1} = f(u_n). \end{array} \right.$$