TP # 18

Représentations graphiques

Partie I - Représentation de suites

Pour effectuer n'importe quelle représentation graphique, on commencera toujours par importer la bibliothèque `matplotlib.pyplot`. Importation suggérée par le programme :

Bibliothèque matplotlib.pyplot

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

Afficher une "ligne brisée"

Python est incapable de représenter des courbes "lisses et continues", constituées d'un nombre infini de points... On peut en revanche afficher des "lignes brisées", c'est à dire des points du plan $M_1, M_2, \dots, M_n$ reliés par des segments.

Relier des points avec `plt.plot(X,Y)`

Après avoir importé `matplotlib.pyplot as plt` :

1. On construit la liste/le vecteur des abscisses : `X = [` $x_1, x_2, \dots, x_n$ `]`
2. On construit la liste/le vecteur des ordonnées : `Y = [` $y_1, y_2, \dots, y_n$ `]`
3. L'instruction `plt.plot(X,Y)` relie les points du plan :

$$M_1 = (x_1, y_1),\ M_2 = (x_2, y_2),\ \dots,\ M_n = (x_n, y_n).$$

4. On affiche la représentation produite avec l'instruction `plt.show()`

Exercice 1

Taper le script suivant dans l'éditeur de texte et exécutez-le.

```
import matplotlib.pyplot as plt

X = [0,1,2,4,6]
Y = [1,1,5,-2,0]

plt.plot(X,Y)
plt.show()
```

Reproduire la courbe affichée par le programme de l'Exercice 1, en indiquant l'abscisse et l'ordonnée des points :

Il est possible d'inclure des "options de tracé" pour changer la couleur et le style.

Options de tracé (hors programme)

On peut ajouter un troisième paramètre : `plt.plot(X,Y,option)`

`option` est une chaîne de caractère qui peut contenir une lettre et deux symboles :

- La lettre détermine la couleur de la courbe (bleue si on ne met rien) :
`b` (bleu), `g` (vert), `r` (rouge), `c` (cyan), `m` (magenta), `y` (jaune), `k` (noir), `w` (blanc).
- Le premier symbole détermine le style des lignes (ligne continue par défaut) :
`-` (ligne continue), `--` (tirets), `:` (pointillés), `-.` (alternance tiret-point)
- Le second symbole détermine l'affichage des points (invisibles si on ne met rien) :
`*` (étoile), `o` (cercle), `+` (plus), `x` (croix), `.` (point), `d` (diamant), etc...

Exemple : `plt.plot(X,Y,'g-*')` : ligne continue verte avec des étoiles.

Exercice 2

Dans le script précédent, tester quelques options de tracé : `'r--'`, `'cx'`, `'k:o'`

Représenter une suite

Représenter les premières valeurs d'une suite $(u_n)_{n\in\mathbb{N}}$ consiste à relier les points du plan :

$$M_0 = (0, u_0),\ M_1 = (1, u_1),\ M_2 = (2, u_2)\ \dots,\ M_n = (n, u_n).$$

Il faut ainsi :

1. Créer la liste/vecteur des abscisses : `X = [` $0, 1, \dots, n$ `]` (avec `range` ou `np.arange`)
2. Construire la liste/vecteur des ordonnées : `Y = [` $u_0, u_1, \dots, u_n$ `]`
3. Relier les points avec `plt.plot(X,Y)` puis afficher la représentation avec `plt.show()`.

• **Pour une suite explicite**, le vecteur des ordonnées `Y` peut être construit directement et très facilement en appliquant par exemple une opération sur le vecteur `X`.

Exemple

Pour représenter les 11 premiers termes de la suite définie par $\forall n \in \mathbb{N}, u_n = \frac{n(n+1)}{2}$,
On voudrait : `X = [` $0, 1, \dots, 10$ `]` et `Y = [` $\frac{0(0+1)}{2}, \frac{1(1+1)}{2}, \dots, \frac{10(10+1)}{2}$ `]`.
On peut donc taper :

```
X = range(11)  et  Y = [k*(k+1)/2 for k in range(11)]
```

ou bien : `X = np.arange(11)` et `Y = X*(X+1)/2` (calcul avec les vecteurs)

Exercice 3

Des suites explicites

1. On pose : $\forall n \in \mathbb{N}, \ u_n = \dfrac{(-1)^n}{n+1}$.

Représenter les 31 premiers termes de la suite u (donc de u_0 à u_{30}).

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

X = ........................

Y = ........................

plt.plot(X,Y,'-*')

plt.show()
```

2. On pose : $\forall n \in \mathbb{N}^*, \ v_n = \left(1 + \dfrac{1}{n}\right)^n$. (Attention : la suite démarre à l'indice 1 !)

(a) Représenter les 200 premiers termes de la suite v (donc de v_1 à v_{200}).

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

X = ........................

Y = ........................

plt.plot(X,Y)

plt.show()
```

(b) Juste avant `plt.show()`, ajouter la ligne suivante :

```
plt.plot( [1,200] , [np.e,np.e] , 'r')
```

Quelle droite cela trace-t-il ? .....

On notera qu'il est ainsi possible d'afficher **plusieurs graphes sur une même figure !**
Il suffit pour cela d'enchaîner plusieurs "`plt.plot`" d'affilée, avant le "`plt.show()`".

• **Pour une suite récurrente**, on construit `Y` petit à petit dans une boucle `for`.

Exemple

Pour les 11 premiers termes de la suite définie par $u_0 = 3$ et $\forall n \in \mathbb{N}, \ u_{n+1} = \sqrt{u_n}$:

```
# Vecteur des abscisses :
X = range(11); # X = [0,1,...,10]
# Vecteur des ordonnees :
Y = np.zeros(11); Y[0] = 3
for k in range(10) :
    Y[k+1] = np.sqrt(Y[k])
```

Exercice 4

Une suite récurrente

On pose : $u_0 = 2$ et $\forall n \in \mathbb{N}, \ u_{n+1} = 1 + \dfrac{4}{u_n}$.

Compléter le programme suivant pour qu'il affiche les 21 premières valeurs de la suite u (donc de u_0 à u_{20}).

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


# Construction du "vecteur des abscisses" :

X = ..............................

# Construction du "vecteur des ordonnees" :

Y = np.zeros( ..... ) ; Y[0] = .....

for k in ........................ :

        Y[k+1] = ................

# Affichage :

................................................
```

2. Constater graphiquement que cette suite converge vers un $\ell \in \mathbb{R}_+$.
Déterminer (rapidement) la valeur exacte de cette limite ℓ :