

But : tracer la représentation graphique d'une fonction réelle d'une variable réelle.

I L'essentiel : ce qu'il faut savoir faire

Pour représenter sur un intervalle $[a, b]$ une fonction f préalablement définie, on code :

```
import matplotlib.pyplot as plt
import numpy as np
X = np.linspace(a,b,200)
Y = f(X)
plt.plot(X,Y)
plt.show()
```

II Documentation

1 Modules utilisés

- `pyplot` (contenu dans la bibliothèque `matplotlib`) : outils graphiques.
- `numpy` : contient de nombreuses fonctions utiles (Numerical Python extensions).

2 Méthode de représentation

Pour représenter graphiquement une fonction f , on définit une liste de points du graphe de la fonction :

- une liste `X` qui contient les abscisses des points : `X = [x1, ..., xn]`,
- une liste `Y` qui contient les ordonnées des points : `Y = [f(x1), ..., f(xn)]`.

La fonction `plot` définit une liste de points M_1, \dots, M_n de coordonnées : $\forall i \in \llbracket 1, n \rrbracket, M_i = (x_i, f(x_i))$.

La fonction `show` produit une fenêtre d'affichage dans laquelle ces points sont représentés, et reliés par des segments de droite.

3 La liste d'abscisses

`X = np.linspace(a, b, N)` définit une liste de N flottants, régulièrement espacés entre a et b .

En pratique, $N = 200$ suffit pour avoir une bonne précision.

4 La liste d'ordonnées

- `Y = f(X)` # quand `X` a été créée avec `linspace`
- `Y = [f(x) for x in X]` # sinon.

5 Fonction `plot` et ses arguments (optionnels)

`plt.plot(X,Y)` crée une liste de points, dont les abscisses sont les éléments de la liste `X`, et dont les ordonnées sont ceux de la liste `Y`.

A noter : la commande `plot(X)` crée une liste de points en utilisant `X` comme liste d'ordonnées et $\llbracket 0, n-1 \rrbracket$ comme liste d'abscisses, où n est la taille (*length*) de la liste `X`.

La commande `plot` peut être enrichie de plusieurs arguments, permettant de donner une légende à une courbe, de définir la couleur, l'épaisseur ou le style de trait. La syntaxe est :

`plt.plot(X,Y, 'style de trait', color = 'couleur', label = 'étiquette', linewidth = 1)`

Styles de traits :

<i>rien</i> ou <code>'-'</code>	traits pleins	<code>'-.'</code>	tirets-points	<code>'x'</code>	croix	<code>'p'</code>	pentagones
<code>'--'</code>	tirets	<code>'.'</code>	points non reliés	<code>'*'</code>	étoiles	<code>'h'</code>	hexagones
<code>':'</code>	points	<code>','</code>	petit points	<code>'v'</code> ou <code>'^'</code>	triangles	<code>'d'</code> ou <code>'D'</code>	losanges
<code>'o'</code> ou <code>'8'</code>	octogones	<code>'+'</code>	croix non reliées	<code>'s'</code>	carrés	<code>'_'</code> ou <code>' '</code>	barres H/V
<code>'1'</code>	flèches haut	<code>'2'</code>	flèches bas	<code>'3'</code>	flèches gauche	<code>'4'</code>	flèches droite

Exemple : `plt.plot(X,Y,'8--')` représente les points par de petits octogones, reliés par des tirets.

Usage : * pouvoir distinguer plusieurs courbes (imprimantes, photocopieuses en noir en blanc),

* Représenter une courbe possédant des discontinuités.

Couleurs :

<i>rien</i> ou 'blue' ou 'b'	bleu	'red' ou 'r'	rouge	'green' ou 'g'	vert
'magenta' ou 'm'	magenta	'cyan' ou 'c'	cyan	'yellow' ou 'y'	jaune
'black' ou 'k'	noir	'white' ou 'w'	blanc	'grey'	gris
'orange'	orange	'purple'	violet	'brown'	brun
'pink'	rose

On peut aussi définir un triplet de flottants entre 0 et 1, qui sera interprété comme une couleur *RGB* :

```
maCouleur=(0.3,0.7,0.6)          # intensité du rouge 30%, vert 70% et bleu 60%.
plt.plot(X,Y,color = maCouleur)
```

Étiquettes : Une chaîne de caractères associée à la courbe (nom de la courbe ou son équation).

Exemples : `plt.plot(X,Y,label = 'Cf')`.
`plt.plot(X,Y,label = r'$ y = \sqrt{x}$')`.

Les formules entre \$ font appelle à une syntaxe *LaTeX*.

6 Fenêtre d'affichage

Par défaut, la fenêtre d'affichage s'adapte aux valeurs extrêmes d'abscisse et d'ordonnée.

On peut préciser ce fonctionnement par défaut à l'aide des commandes :

```
plt.autoscale(enable = True, axis = 'both')
plt.autoscale(enable = True, axis = 'x')
plt.autoscale(enable = True, axis = 'y')
plt.autoscale(enable = False)
```

On peut aussi définir explicitement les dimensions de la fenêtre d'affichage :

```
plt.xlim(-10,5)          # définit une plage d'abscisses  $x$  tels que :  $-10 \leq x \leq 5$ 
plt.ylim(0,100)          # définit une plage d'ordonnées  $y$  tels que :  $0 \leq y \leq 100$ 
```

La commande suivante permet d'imposer un repère orthonormé :

```
plt.axis('equal')
```

7 Affichage : la fonction show

La commande `plt.show()` permet d'afficher les graphiques.

Elle terminera systématiquement un script d'affichage graphique.

Cette commande ne doit être écrite qu'une seule fois (même si plusieurs graphiques sont tracés).

8 Problème d'ensemble de définition

Un message d'erreur est renvoyé si on tente de calculer $f(x)$ pour une valeur de x extérieure à \mathcal{D}_f .

On peut contourner le problème en utilisant `nan`, importé du module `math` ou `numpy`.

Exemple avec $f(x) = \sqrt{x}$:
`nan` signifie 'not a number'

```
def f(x) :
    try : return sqrt(x)
    except : return nan
```

9 Présentation

Les commandes suivantes permettent de finaliser la présentation d'un graphique.

```
plt.title('Titre du graphique')      # Donne un titre au graphique.
plt.xlabel('abscisses')               # Donne une légende à l'axe des abscisses.
plt.ylabel('ordonnées')              # Donne une légende à l'axe des ordonnées.
plt.grid(True) ou grid(False)        # Affiche (ou pas) une grille.
plt.legend() ou legend(loc='best')    # Affiche les étiquettes des courbes.
plt.text(2.,-0.2,'maximum')          # Affiche un texte à la position donnée
```

III Travail à réaliser

(1) Représenter sur un même graphique les fonctions suivantes sur $[-5, 5]$, en prenant soin d'indiquer une légende claire :

$$f_1 : x \mapsto \sin(2x) , \quad f_2 : x \mapsto \frac{1}{3}x^2 - x - 1 , \quad f_3 : x \mapsto \frac{\ln(x+1)}{x-3}$$

$$f_4 : x \mapsto \frac{\sin(x)}{x} , \quad f_5 : x \mapsto \frac{1}{2}[x] + 1 , \quad f_6 : x \mapsto e^{-x} \cos(3x)$$

(2) Expliquer pourquoi la commande : `X = [a+(b-a)*k/N for k in range(N+1)]` crée une liste de $N + 1$ flottants régulièrement espacés, dont le premier vaut **a** et le dernier **b**. Quel est alors le *pas* de cette liste (l'écart commun entre deux valeurs) ?

(3) Représenter sur le même graphique et sur l'intervalle $[-2\pi, 2\pi]$ les fonctions :

$$s_1 : x \mapsto x, \quad s_2 : x \mapsto s_1(x) - \frac{x^3}{3!}, \quad s_3 : x \mapsto s_2(x) + \frac{x^5}{5!}, \quad s_4 : x \mapsto s_3(x) - \frac{x^7}{7!},$$

$$s_5 : x \mapsto s_4(x) + \frac{x^9}{9!}, \quad s_6 : x \mapsto s_5(x) - \frac{x^{11}}{11!}, \quad s_7 : x \mapsto s_6(x) + \frac{x^{13}}{13!}, \quad s : x \mapsto \sin(x).$$

On pourra importer la fonction `factorial`, ou utiliser un travail précédent.

Utiliser des pointillés pour représenter la courbe du sinus.

Que peut-on constater ?

(4) Pour $n \in \mathbf{N}$, on définit la fonction f_n par : $f_n(x) = \exp\left(\frac{n}{100} - x\right) \cos(x\sqrt{n})$.

Représenter sur le même graphique et sur l'intervalle $[-2, 3]$ les fonctions f_n pour $n \in \llbracket 0, 100 \rrbracket$. On utilisera une boucle portant sur un entier n variant de 0 à 100.

(5) Représenter les fonctions $f_n : x \mapsto \sum_{k=1}^n \frac{\sin(k!x)}{k^2}$ sur l'intervalle $\left[-\frac{\pi}{2}, \pi\right]$, pour $n \in \llbracket 1, 6 \rrbracket$.

Quelle particularité peut-on imaginer pour la courbe de f_n lorsque $n \rightarrow +\infty$?

(6) Définir et représenter une courbe par sa représentation paramétrique ($a, b, r, R, d \in \mathbf{R}$) :

On pourra ici définir une liste **T** de valeurs pour le paramètre t , puis deux listes **X**, **Y** correspondant aux formules proposées. Les courbes seront alors définies par la commande : `plot(X,Y)`.

Choisir librement les constantes $a, b, r, R, d \in \mathbf{R}$, sans hésiter à changer leurs valeurs.

<i>ellipses :</i>	<i>courbes de Lissajous :</i>	<i>astroïde :</i>
$C_1 \begin{cases} x(t) = a \cos(t) \\ y(t) = b \sin(t) \end{cases}$	$C_2 \begin{cases} x(t) = \cos(at) \\ y(t) = \sin(bt) \end{cases}$	$C_3 \begin{cases} x(t) = \cos^3(t) \\ y(t) = \sin^3(t) \end{cases}$
<i>cycloïde :</i>	<i>cardioïde :</i>	<i>spirale d'Archimède :</i>
$C_4 \begin{cases} x(t) = R(t - \sin t) \\ y(t) = R(1 - \cos t) \end{cases}$	$C_5 \begin{cases} x(t) = a \cos(t)(1 + \cos t) \\ y(t) = a \sin(t)(1 + \cos t) \end{cases}$	$C_6 \begin{cases} x(t) = at \cos(t) \\ y(t) = bt \sin(t) \end{cases}$
<i>limaçon :</i>	<i>trochoïde :</i>	<i>spirale exponentielle :</i>
$C_7 \begin{cases} x(t) = (\cos t)(a + b \cos t) \\ y(t) = (\sin t)(a + b \cos t) \end{cases}$	$C_8 \begin{cases} x(t) = at - b \sin t \\ y(t) = a - b \cos t \end{cases}$	$C_9 \begin{cases} x(t) = a \ln t \cos(t) \\ y(t) = b \ln t \sin(t) \end{cases}$

et *hypotrochoïdes* : $C_{10} \begin{cases} x(t) = (R-r) \cos t + d \cos\left(\frac{R-r}{r}t\right) \\ y(t) = (R-r) \sin t - d \sin\left(\frac{R-r}{r}t\right) \end{cases}, \quad 0 < r < R$