# Graphiques avec matplotlib.pyplot

Rappelons le programme officiel :

| Domaines numériques | Capacités exigibles |
|---|---|
| **1. Outils graphiques** | |
| Représentation graphique d'un nuage de points. | Utiliser les fonctions de base de la bibliothèque **matplotlib** pour représenter un nuage de points. |
| Représentation graphique d'une fonction. | Utiliser les fonctions de base de la bibliothèque **matplotlib** pour tracer la courbe représentative d'une fonction. |
| Courbes planes paramétrées. | Utiliser les fonctions de base de la bibliothèque **matplotlib** pour tracer une courbe plane paramétrée. |

On importe la bibliothèque matplotlib.pyplot avec l'alias plt. On appellera alors les différentes fonctions de cette bibliothèque avec le préfixe plt. (avec un point séparant plt du nom de la fonction).

Pour entrer les données, on peut écrire une liste, comme :

```
x = [1, 2, 3, 4, 5]
```

On peut créer une colonne de valeurs avec numpy, ce qui est commode pour travailler ensuite sur ces valeurs . On doit au préalable importer numpy (par exemple avec l'alias np) :

```
import numpy as np
x = np.array([1, 2, 3, 4, 5])
```

Souvent on fait varier x dans un intervalle, entre une valeur minimale xmin et une valeur maximale xmax, en prenant un certain nombre de valeurs n sur cet intervalle :

```
import numpy as np
x = np.linspace(xmin, xmax, n)
```

Commande pour visualiser y en fonction de x :

```
plt.plot(x, y)
plt.show()
```

**Remarque : On doit terminer le programme par plt.show() pour que le graphique s'affiche.**

Et on peut s'occuper des finitions :

```
plt.title("Titre")        Pour avoir un titre
plt.xlabel( 'x')                indique ce que l'on a en abscisses
plt.ylabel('y')                 indique ce que l'on a en ordonnées
plt.grid()                      pour avoir une grille de fond
```

Dans plt.plot, on peut choisir la couleur ( r pour red, b pour blue, g pour green, k pour black..), la forme des points (+, o, x, s (pour square cad carré), v pour triangle), la taille des points (marquersize = 'valeur'), le style (- pour un trait continu, : pour des pointillés), l'épaisseur des traits (linewidth = 'valeur').

On peut compacter ; par exemple :

```
plt.plot(x, y, 'ro')        # Trace y en fonction de x, les points sont marqués par des ronds rouges.
```

Pour écrire une légende :

```
plt.plot( x, y, label = ‘ légende’)
plt.legend()
```

Réglage des axes :

```
plt.axis( [ xmin, xmax, ymin, ymax]) permet de préciser les axes.
```