

Importer Matplotlib

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

Fonctions communes à tous les graphiques

- **plt.title('Titre')** : Ajoute un titre au graphique.
- **plt.xlabel('Étiquette de l'axe X')** : Ajoute une étiquette à l'axe des abscisses.
- **plt.ylabel('Étiquette de l'axe Y')** : Ajoute une étiquette à l'axe des ordonnées.
- **plt.grid(True)** : Ajoute un quadrillage au graphique.
- **plt.legend()** : Ajoute une légende au graphique.
- **plt.show()** : Affiche le graphique.

Graphique en ligne

```
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
```

```
plt.plot(x, y) # Crée le graphique
plt.show() # Affiche le graphique
```

- **plt.plot(x, y)** : Crée un graphique en ligne.
 - **x** : Les valeurs sur l'axe des abscisses.
 - **y** : Les valeurs sur l'axe des ordonnées.
 - D'autres paramètres : **color** (couleur de la ligne), **linestyle** (style de la ligne, par exemple, solide, pointillé), **linewidth** (épaisseur de la ligne), etc.

Graphique en barres

```
etiquettes = ['A', 'B', 'C']
valeurs = [10, 20, 15]
```

```
plt.bar(etiquettes, valeurs) # Crée le graphique
plt.show() # Affiche le graphique
```

- **plt.bar(etiquettes, valeurs)** : Crée un diagramme en barres.
 - **etiquettes** : Les étiquettes pour chaque barre.
 - **valeurs** : La hauteur de chaque barre.

- D'autres paramètres : `color` (couleur des barres), `edgecolor` (couleur des contours des barres), `width` (largeur des barres), etc.

Histogramme

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

```
plt.hist(donnees, bins=5) # Crée le graphique avec 5 colonnes  
plt.show() # Affiche le graphique
```

- **`plt.hist(donnees, bins=5)`** : Crée un histogramme de 5 colonnes.
 - **`donnees`** : Les données à distribuer dans les 5 colonnes.
 - **`bins`** : Nombre de colonnes (intervalles) à créer.
 - D'autres paramètres : `color` (couleur des barres), `alpha` (transparence des barres), `edgecolor` (couleur des bords des barres), etc.

Graphique en secteurs

```
etiquettes = ['A', 'B', 'C']  
tailles = [10, 20, 30]
```

```
plt.pie(tailles, labels=etiquettes) # Crée le graphique  
plt.show() # Affiche le graphique
```

- **`plt.pie(tailles, labels=etiquettes)`** : Crée un pie chart.
 - **`tailles`** : Les proportions pour chaque secteur.
 - **`etiquettes`** : Les étiquettes pour chaque secteur.
 - D'autres paramètres : `colors` (couleurs des segments), `autopct` (pourcentage affiché sur chaque segment), `shadow` (ombre derrière le graphique), etc.

Graphique en nuage de points

```
x = [1, 2, 3, 4]  
y = [10, 20, 25, 30]
```

```
plt.scatter(x, y) # Crée le graphique  
plt.show() # Affiche le graphique
```

- **`plt.scatter(x, y)`** : Crée un nuage de points.
 - **`x`** : Les valeurs sur l'axe des abscisses.

- **y** : Les valeurs sur l'axe des ordonnées.
- D'autres paramètres : `color` (couleur des points), `marker` (forme des points), `s` (taille des points), etc.

Sauvegarder un graphique

- **Enregistrer le graphique au format png, au même endroit que le fichier .ipynb :**

```
plt.savefig('graphique.png')
```

Un exemple complet pour un graphique linéaire

```
import matplotlib.pyplot as plt

# Données
x = [1, 2, 3, 4, 5]
y = [10, 20, 25, 30, 35]

# Création du graphique en ligne
plt.plot(x, y, color='blue', linestyle='-', linewidth=2, marker='o', markersize=5)

# Ajout de titres du graphique et des axes
plt.title('Exemple de graphique en ligne')
plt.xlabel('Axe X')
plt.ylabel('Axe Y')

# Affichage du quadrillage
plt.grid(True)

# Sauvegarde du graphique
plt.savefig('graphique.png')

# Affichage du graphique
plt.show()
```

Ajout d'une droite de régression linéaire

- Voir le cheatsheet sur NumPy

