### Introduction à Matplotlib

Matplotlib est une bibliothèque de visualisation de données en Python, largement utilisée pour générer des graphiques de haute qualité. Elle permet aux scientifiques et ingénieurs de créer des visualisations statiques, animées et interactives de données.

#### Installation

Avant de commencer, assure-toi d'avoir Matplotlib installé. Tu peux l'installer via pip:

```bash

pip install matplotlib

```

#### Composants de base

1. \*\*Pyplot\*\*: C'est un module de Matplotlib qui offre une interface de tracé simple. Il est généralement importé comme suit:

```python

import matplotlib.pyplot as plt

```

2. \*\*Figure et Axes\*\*: La Figure est la fenêtre ou la page globale où les graphiques sont dessinés. Les Axes sont une partie de la figure où les données sont tracées.

```python

fig, ax = plt.subplots()

```

3. \*\*Types de graphiques courants\*\*:

- \*\*Graphique linéaire\*\*: Utilisé pour tracer des points connectés par des lignes.

- \*\*Histogramme\*\*: Utilisé pour afficher la distribution des données.

- \*\*Graphique en barres\*\*: Utilisé pour comparer des catégories.

- \*\*Graphique en dispersion (scatter plot)\*\*: Utilisé pour montrer la relation entre deux variables.

#### Exemples de base

1. \*\*Graphique linéaire\*\*:

```python

import matplotlib.pyplot as plt

# Données

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

y = [2, 3, 5, 7, 11]

# Création du graphique

plt.plot(x, y)

# Ajout de labels et titre

plt.xlabel('x')

plt.ylabel('y')

plt.title('Graphique linéaire')

# Affichage

plt.show()

```

2. \*\*Histogramme\*\*:

```python

import matplotlib.pyplot as plt

# Données

data = [1, 2, 2, 3, 3, 3, 4, 4, 4, 4]

# Création de l'histogramme

plt.hist(data, bins=4)

# Ajout de labels et titre

plt.xlabel('Valeur')

plt.ylabel('Fréquence')

plt.title('Histogramme')

# Affichage

plt.show()

```

3. \*\*Graphique en barres\*\*:

```python

import matplotlib.pyplot as plt

# Données

categories = ['A', 'B', 'C']

values = [5, 7, 3]

# Création du graphique en barres

plt.bar(categories, values)

# Ajout de labels et titre

plt.xlabel('Catégories')

plt.ylabel('Valeurs')

plt.title('Graphique en barres')

# Affichage

plt.show()

```

4. \*\*Graphique en dispersion\*\*:

```python

import matplotlib.pyplot as plt

# Données

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

y = [2, 3, 5, 7, 11]

# Création du graphique en dispersion

plt.scatter(x, y)

# Ajout de labels et titre

plt.xlabel('x')

plt.ylabel('y')

plt.title('Graphique en dispersion')

# Affichage

plt.show()

```

### Tableau des fonctions et méthodes incontournables

| Fonction/Méthode | Description |

|-------------------------|--------------------------------------------------------------------------|

| `plt.plot()` | Crée un graphique linéaire. |

| `plt.hist()` | Crée un histogramme. |

| `plt.bar()` | Crée un graphique en barres. |

| `plt.scatter()` | Crée un graphique en dispersion. |

| `plt.xlabel()` | Ajoute une étiquette à l'axe des abscisses. |

| `plt.ylabel()` | Ajoute une étiquette à l'axe des ordonnées. |

| `plt.title()` | Ajoute un titre au graphique. |

| `plt.show()` | Affiche le graphique. |

| `plt.savefig()` | Enregistre le graphique sous forme d'image. |

| `fig, ax = plt.subplots()`| Crée une figure et des axes, permettant des graphes plus complexes. |

Voilà un aperçu simple et détaillé de Matplotlib pour te lancer! 🚀

### Introduction à Pandas avec Matplotlib

Pandas est une bibliothèque puissante pour la manipulation et l'analyse de données en Python. Elle permet de gérer des données sous forme de DataFrames, qui sont des structures de données tabulaires similaires aux tableaux Excel. Lorsque l’on combine Pandas avec Matplotlib, on peut créer des graphiques directement à partir des DataFrames, ce qui rend le processus de visualisation des données beaucoup plus simple et efficace.

#### Installation

Si ce n’est pas déjà fait, installe Pandas avec pip:

```bash

pip install pandas

```

#### Exemple d'utilisation de Pandas avec Matplotlib

1. \*\*Charger des données et créer un graphique à barres\*\*:

```python

import pandas as pd

import matplotlib.pyplot as plt

# Données sous forme de dictionnaire

data = {'Nom': ['Alice', 'Bob', 'Charlie', 'David'],

'Score': [88, 92, 85, 90]}

# Création d'un DataFrame

df = pd.DataFrame(data)

# Création du graphique à barres à partir du DataFrame

df.plot(kind='bar', x='Nom', y='Score')

# Ajout de labels et titre

plt.xlabel('Nom')

plt.ylabel('Score')

plt.title('Scores des étudiants')

# Affichage

plt.show()

```

2. \*\*Graphique linéaire à partir de données temporelles\*\*:

```python

import pandas as pd

import matplotlib.pyplot as plt

# Création d'une série de données temporelles

date\_range = pd.date\_range(start='1/1/2022', periods=5, freq='D')

data = {'Température': [20, 21, 19, 22, 20]}

df = pd.DataFrame(data, index=date\_range)

# Création du graphique linéaire

df.plot()

# Ajout de labels et titre

plt.xlabel('Date')

plt.ylabel('Température (°C)')

plt.title('Variation de la température au cours des jours')

# Affichage

plt.show()

```

### Tableau des fonctions et méthodes incontournables

| Fonction/Méthode | Description |

|-------------------------|--------------------------------------------------------------------------|

| `plt.plot()` | Crée un graphique linéaire. |

| `plt.hist()` | Crée un histogramme. |

| `plt.bar()` | Crée un graphique en barres. |

| `plt.scatter()` | Crée un graphique en dispersion. |

| `plt.xlabel()` | Ajoute une étiquette à l'axe des abscisses. |

| `plt.ylabel()` | Ajoute une étiquette à l'axe des ordonnées. |

| `plt.title()` | Ajoute un titre au graphique. |

| `plt.show()` | Affiche le graphique. |

| `plt.savefig()` | Enregistre le graphique sous forme d'image. |

| `fig, ax = plt.subplots()`| Crée une figure et des axes, permettant des graphes plus complexes. |

| `pd.read\_csv()` | Charge un fichier CSV dans un DataFrame Pandas. |

| `pd.DataFrame()` | Crée un DataFrame Pandas à partir de données diverses. |

| `df.plot()` | Crée un graphique à partir d'un DataFrame Pandas. |

Voilà de quoi te donner une bonne base pour utiliser Matplotlib avec Pandas pour tes visualisations de données en sciences de la nature! 🚀

Parfait! Voici quelques exercices pratiques pour utiliser Pandas et Matplotlib avec des fichiers CSV. Ces exercices couvrent des thèmes des sciences de la nature et de la santé.

### Exercice 1: Analyse des données de température

#### Objectif:

Analyser les données de température sur une année et visualiser les variations mensuelles.

#### Données:

Un fichier CSV nommé `temperatures.csv` contenant deux colonnes : `Date` (format AAAA-MM-JJ) et `Température` (en °C).

#### Instructions:

1. \*\*Charger les données\*\*:

```python

import pandas as pd

df = pd.read\_csv('temperatures.csv')

```

2. \*\*Convertir la colonne `Date` en type datetime\*\*:

```python

df['Date'] = pd.to\_datetime(df['Date'])

```

3. \*\*Ajouter une colonne `Mois`\*\*:

```python

df['Mois'] = df['Date'].dt.month

```

4. \*\*Calculer la température moyenne pour chaque mois\*\*:

```python

moyennes\_mensuelles = df.groupby('Mois')['Température'].mean()

```

5. \*\*Visualiser les données avec un graphique linéaire\*\*:

```python

import matplotlib.pyplot as plt

plt.plot(moyennes\_mensuelles.index, moyennes\_mensuelles.values)

plt.xlabel('Mois')

plt.ylabel('Température moyenne (°C)')

plt.title('Variation mensuelle de la température')

plt.show()

```

### Exercice 2: Analyse des données de patients

#### Objectif:

Étudier la distribution d'âges des patients et comparer les taux de cholestérol par groupe d'âge.

#### Données:

Un fichier CSV nommé `patients.csv` contenant les colonnes : `Age`, `Sexe`, `Cholestérol` (en mg/dL).

#### Instructions:

1. \*\*Charger les données\*\*:

```python

import pandas as pd

df = pd.read\_csv('patients.csv')

```

2. \*\*Créer des groupes d'âge\*\*:

```python

bins = [0, 20, 40, 60, 80, 100]

labels = ['0-20', '21-40', '41-60', '61-80', '81-100']

df['Groupe d\'âge'] = pd.cut(df['Age'], bins=bins, labels=labels)

```

3. \*\*Calculer le taux moyen de cholestérol par groupe d'âge\*\*:

```python

cholesterol\_par\_age = df.groupby('Groupe d\'âge')['Cholestérol'].mean()

```

4. \*\*Visualiser les données avec un graphique en barres\*\*:

```python

import matplotlib.pyplot as plt

cholesterol\_par\_age.plot(kind='bar')

plt.xlabel('Groupe d\'âge')

plt.ylabel('Cholestérol moyen (mg/dL)')

plt.title('Cholestérol moyen par groupe d\'âge')

plt.show()

```

### Exercice 3: Analyse des précipitations annuelles

#### Objectif:

Analyser les précipitations annuelles sur plusieurs années et visualiser les tendances.

#### Données:

Un fichier CSV nommé `precipitations.csv` contenant les colonnes : `Année` et `Précipitation` (en mm).

#### Instructions:

1. \*\*Charger les données\*\*:

```python

import pandas as pd

df = pd.read\_csv('precipitations.csv')

```

2. \*\*Calculer la précipitation totale par année\*\*:

```python

total\_par\_annee = df.groupby('Année')['Précipitation'].sum()

```

3. \*\*Visualiser les données avec un graphique linéaire\*\*:

```python

import matplotlib.pyplot as plt

plt.plot(total\_par\_annee.index, total\_par\_annee.values)

plt.xlabel('Année')

plt.ylabel('Précipitation totale (mm)')

plt.title('Précipitations annuelles')

plt.show()

```

Ces exercices te permettront de te familiariser avec les fonctionnalités de Pandas et Matplotlib en utilisant des données réelles. Bon courage! 🚀

EXERCICES BIOLOGIE

Bien sûr ! Voici quelques exercices pratiques pour analyser des données scientifiques en utilisant les bibliothèques **pandas** et **matplotlib** en Python. Nous allons également calculer et tracer une droite de régression sans utiliser de bibliothèque scientifique pour les calculs.

**Exercice 1 : Analyse des données de tests biologiques**

**Objectif**

Analyser les résultats de tests biologiques pour déterminer la relation entre deux variables.

**Étapes**

**Importer les bibliothèques nécessaires :**

import pandas as pd

import matplotlib.pyplot as plt

**Charger les données :**

data = {

'Concentration': [0.1, 0.2, 0.3, 0.4, 0.5],

'Absorbance': [0.05, 0.10, 0.15, 0.20, 0.25]

}

df = pd.DataFrame(data)

**Tracer les données :**

plt.scatter(df['Concentration'], df['Absorbance'])

plt.xlabel('Concentration (M)')

plt.ylabel('Absorbance')

plt.title('Relation entre la concentration et l\'absorbance')

plt.show()

**Calculer la droite de régression :**

x = df['Concentration']

y = df['Absorbance']

n = len(x)

sum\_x = sum(x)

sum\_y = sum(y)

sum\_xy = sum(x \* y)

sum\_x2 = sum(x \*\* 2)

# Calcul des coefficients de la droite de régression y = mx + b

m = (n \* sum\_xy - sum\_x \* sum\_y) / (n \* sum\_x2 - sum\_x \*\* 2)

b = (sum\_y - m \* sum\_x) / n

print(f"Équation de la droite de régression : y = {m:.2f}x + {b:.2f}")

**Tracer la droite de régression :**

regression\_line = [m \* xi + b for xi in x]

plt.scatter(x, y)

plt.plot(x, regression\_line, color='red')

plt.xlabel('Concentration (M)')

plt.ylabel('Absorbance')

plt.title('Droite de régression')

plt.show()

**Exercice 2 : Analyse des données de tests chimiques**

**Objectif**

Analyser les résultats de tests chimiques pour déterminer la relation entre deux variables.

**Étapes**

1. **Importer les bibliothèques nécessaires :**
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. **Charger les données :**
5. data = {
6. 'Temps (s)': [1, 2, 3, 4, 5],
7. 'Réaction (mol)': [0.2, 0.4, 0.6, 0.8, 1.0]
8. }
9. df = pd.DataFrame(data)
10. **Tracer les données :**
11. plt.scatter(df['Temps (s)'], df['Réaction (mol)'])
12. plt.xlabel('Temps (s)')
13. plt.ylabel('Réaction (mol)')
14. plt.title('Relation entre le temps et la réaction')
15. plt.show()
16. **Calculer la droite de régression :**
17. x = df['Temps (s)']
18. y = df['Réaction (mol)']
19. n = len(x)
20. sum\_x = sum(x)
21. sum\_y = sum(y)
22. sum\_xy = sum(x \* y)
23. sum\_x2 = sum(x \*\* 2)
24. # Calcul des coefficients de la droite de régression y = mx + b
25. m = (n \* sum\_xy - sum\_x \* sum\_y) / (n \* sum\_x2 - sum\_x \*\* 2)
26. b = (sum\_y - m \* sum\_x) / n
27. print(f"Équation de la droite de régression : y = {m:.2f}x + {b:.2f}")
28. **Tracer la droite de régression :**
29. regression\_line = [m \* xi + b for xi in x]
30. plt.scatter(x, y)
31. plt.plot(x, regression\_line, color='red')
32. plt.xlabel('Temps (s)')
33. plt.ylabel('Réaction (mol)')
34. plt.title('Droite de régression')
35. plt.show()

Ces exercices devraient vous aider à pratiquer l’analyse de données scientifiques en utilisant **pandas** et **matplotlib**, tout en calculant et traçant une droite de régression sans utiliser de bibliothèque scientifique pour les calculs. Si vous avez des questions ou besoin de plus d’exercices, n’hésitez pas à demander !