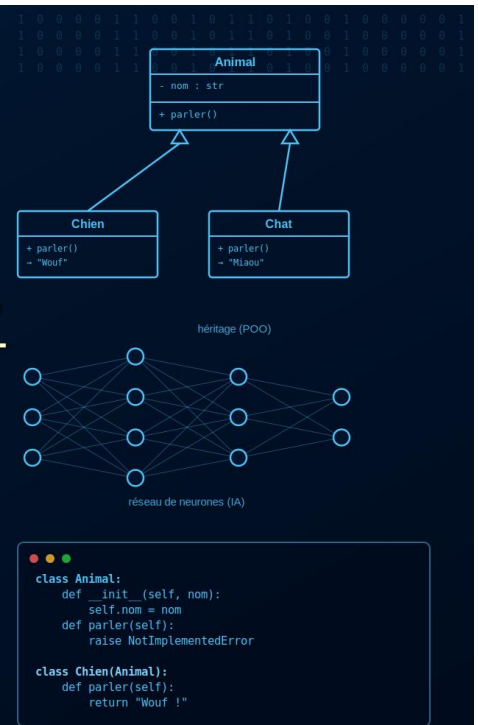


V. Visualisation scientifique



M. Martin

1

1



ENSGTI
ÉCOLE D'INGÉNIEURS

V. Visualisation scientifique

Pourquoi visualiser les données?

- La visualisation est essentielle pour analyser et comprendre les phénomènes étudiés
- Un graphique permet de détecter rapidement :
 - des tendances (croissance, décroissance, oscillations)
 - des anomalies (valeurs aberrantes)
 - des corrélations entre grandeurs
 - des erreurs de mesure ou de simulation

En physique comme en ingénierie, un graphique bien conçu vaut mieux qu'un tableau de nombres trop long.

Dans ce cours, nous utilisons principalement :

- (NumPy : génération et manipulation des données numériques)
- Matplotlib : création de figures scientifiques de qualité

2

2



Problème

Comment passer des données brutes à une figure exploitable?

Solution

1. Générer / charger les données → NumPy
2. Nettoyer / filtrer → masques booléens, statistiques
3. Visualiser → Matplotlib (courbes, histogrammes, nuages de points, images)
4. Mettre en valeur → titres, unités, légendes, annotations
5. Exporter →
 - formats vectoriels (PDF/SVG) pour publication
 - PNG haute résolution pour diaporamas ou rapports texte



3



Pièges fréquents et contre-mesures :

- Axes sans unités → toujours préciser grandeur et unité
- Légende ambiguë → noms explicites, symboles cohérents
- Grille absente → difficile d'estimer des valeurs
- Couleurs inadaptées pour déficients visuels → utiliser des gammes de couleurs perceptibles « viridis », « cividis »



4

Générer des données avec NumPy

NumPy est utilisé pour **créer et manipuler des tableaux numériques**.

Problème

Comment créer rapidement un signal simulé (ex. signal amorti et bruité) de la forme :

$$f = Ae^{-\alpha t} \cos(\omega t) + b$$

Solution

```
import numpy as np

t = np.linspace(0, 10, 501) # Définition de l'axe temporel
A, alpha, omega = 5.0, 0.5, 2*np.pi
f = A * np.exp(-alpha * t) * np.cos(omega * t) # Signal amorti
rng = np.random.default_rng(0)
bruit = rng.normal(0, 0.2, size=t.shape) # Ajout d'un bruit gaussien
f_bruite = f + bruit
```

5

5

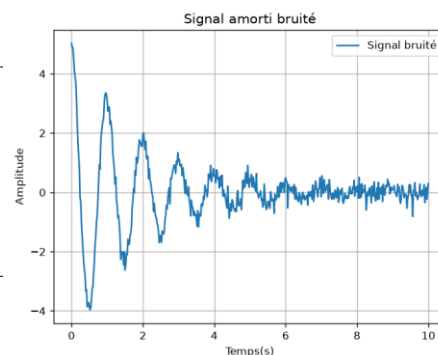
Matplotlib et le tracé de courbes

Problème

Comment représenter une grandeur en fonction du temps (ex. : le signal précédent) ?

Solution

```
import matplotlib.pyplot as plt
plt.plot(t, f_bruite, label="Signal bruité")
plt.xlabel("Temps(s)")
plt.ylabel("Amplitude")
plt.title("Signal amorti bruité")
plt.legend()
plt.grid(True)
plt.show()
```



6

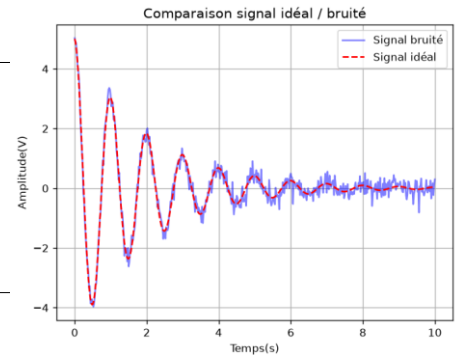
6

Problème

Comment comparer données expérimentales et modèle théorique ?

Solution

```
import matplotlib.pyplot as plt
plt.plot(t, f_bruite, "b-", alpha=0.5, label="Signal bruité")
plt.plot(t, f, "r--", label="Signal idéal")
plt.xlabel("Temps(s)")
plt.ylabel("Amplitude(V)")
plt.title("Comparaison signal idéal / bruité")
plt.legend()
plt.grid(True)
plt.show()
```



- alpha = transparence (utile si on superpose plusieurs distributions)

7

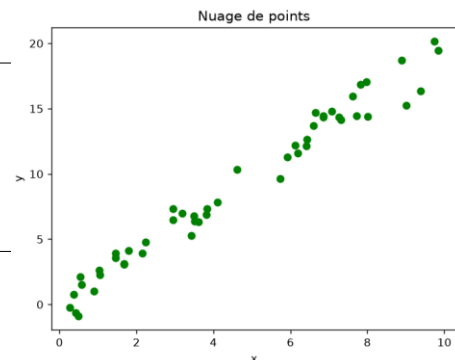
7

Problème

Comment visualiser la corrélation entre deux variables expérimentales ?

Solution

```
x = np.random.rand(50) * 10
y = 2 * x + np.random.randn(50)
plt.scatter(x, y, c="green", marker="o")
plt.xlabel("x")
plt.ylabel("y")
plt.title("Nuage de points")
plt.show()
```



8

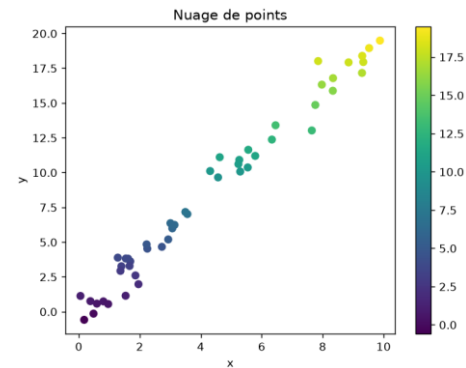
8

Problème

Comment visualiser la corrélation entre deux variables expérimentales en associant des couleurs à l'une ?

Solution

```
x = np.random.rand(50) * 10
y = 2 * x + np.random.randn(50)
plt.scatter(x, y, c=y, marker="o", cmap="viridis")
plt.xlabel("x")
plt.ylabel("y")
plt.title("Nuage de points")
plt.colorbar()
plt.show()
```



9

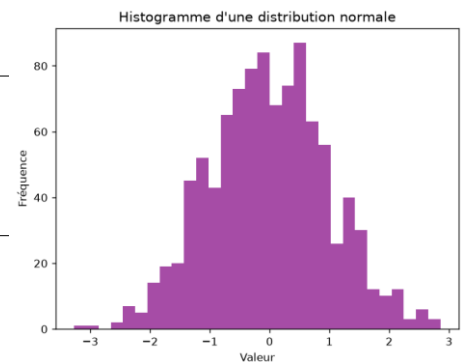
9

Problème

Comment étudier la distribution de mesures (ex. loi normale) ?

Solution

```
donnees = np.random.randn(1000)
plt.hist(donnees, bins=30, color="purple", alpha=0.7)
plt.xlabel("Valeur")
plt.ylabel("Fréquence")
plt.title("Histogramme d'une distribution normale")
plt.show()
```



- bins = nombre de classes

10

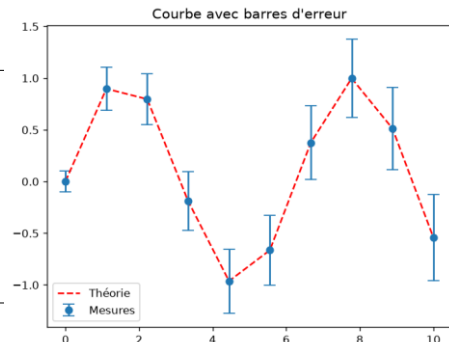
10

Problème

Comment montrer une incertitude expérimentale ?

Solution

```
x = np.linspace(0, 10, 10)
y = np.sin(x)
erreurs = 0.1 + 0.1 * np.sqrt(x)
plt.errorbar(x, y, yerr=erreurs, fmt="o",
             capsize=5, label="Mesures")
plt.plot(x, np.sin(x), "r--", label="Théorie")
plt.legend()
plt.title("Courbe avec barres d'erreur")
plt.show()
```



11

11

Problème

Comment comparer les résultats de plusieurs expériences ?

Solution

```
categories = ["Exp1", "Exp2", "Exp3", "Exp4"]
valeurs = [5.1, 7.3, 6.8, 8.0]
plt.bar(categories, valeurs, color="orange")
plt.title("Résultats expérimentaux")
plt.ylabel("Valeur")
plt.show()
```



12

12

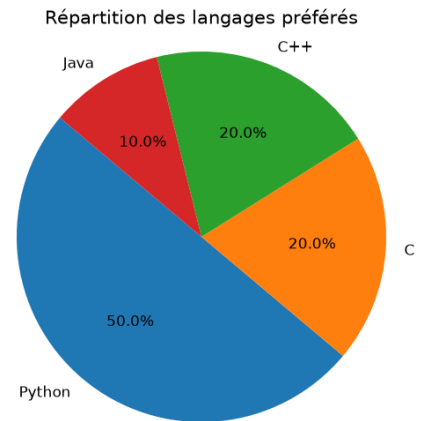
Diagrammes circulaires (camemberts)

Problème

Comment visualiser la répartition d'un ensemble en pourcentage ?

Solution

```
labels = ["Python", "C", "C++", "Java"]
parts = [50, 20, 20, 10]
plt.pie(parts, labels=labels, autopct="%1.1f%", startangle=140)
plt.title("Répartition des langages préférés")
plt.axis("equal")
plt.show()
```



- Remarque : peu utilisés en sciences dures, mais parfois utiles pour des répartition qualitatives

13

13

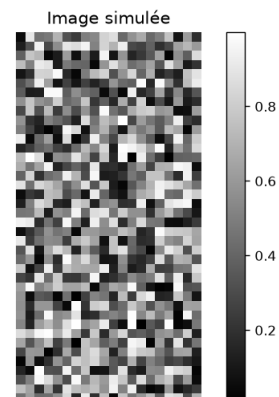
Affichage d'images

Problème

Comment afficher une image en niveaux de gris ?

Solution

```
image = np.random.rand(40, 20)
plt.imshow(image, cmap="gray")
# Spécifie la gamme de couleurs en "niveaux de gris"
plt.title("Image simulée")
plt.colorbar()
plt.axis("off")
plt.show()
```



14

14

Problème

Comment afficher une image couleur ?

Solution

```
import matplotlib.image as mpimg
image = mpimg.imread("mon_image.jpg")
plt.imshow(image)
plt.axis("off")
plt.title("Image couleur")
plt.show()
```

Image couleur



15

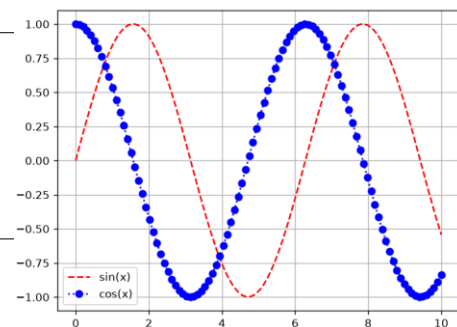
15

Problème

Comment change-t-on le style des lignes et les marqueurs ?

Solution

```
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)
plt.plot(x, y1, "r--", label="sin(x)")
plt.plot(x, y2, "b:", marker="o", label="cos(x)")
plt.legend()
plt.grid(True)
plt.show()
```



16

16

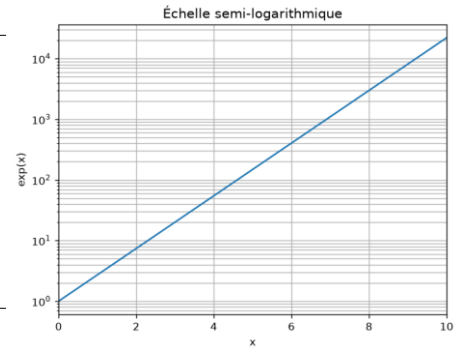
Problème

Comment analyser des phénomènes qui couvrent plusieurs ordres de grandeur ?

Solution

On fixe des limites aux axes et on modifie les échelles.

```
x = np.linspace(0, 10, 100)
plt.plot(x, np.exp(x))
plt.xlim(0, 10) # Limite l'axe des abscisses à [0, 10]
plt.yscale("log")
# L'axe des ordonnées est en échelle logarithmique
plt.title("Échelle semi-logarithmique")
plt.xlabel("x")
plt.ylabel("exp(x)")
plt.grid(True, which="both")
# Grille sur graduations majeures et mineures
plt.show()
```



17

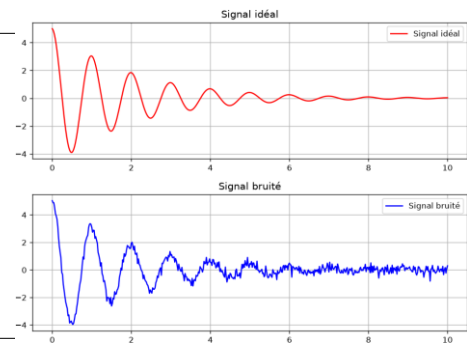
17

Problème

Comment présenter plusieurs résultats dans une même figure ?

Solution

```
fig, zones_trace = plt.subplots(2, 1, figsize=(8, 6))
zones_trace[0].plot(t, f, "r-", label="Signal idéal")
zones_trace[0].legend()
zones_trace[0].set_title("Signal idéal")
zones_trace[0].grid(True)
zones_trace[1].plot(t, f_bruite, "b-",
                    label="Signal bruité")
zones_trace[1].legend()
zones_trace[1].set_title("Signal bruité")
zones_trace[1].grid(True)
plt.tight_layout()
plt.show()
```



18

18

Problème

Comment afficher deux images côte à côte dans une même figure ?

Solution

```
image1 = mpimg.imread('votre_image1.jpg')
image2 = mpimg.imread('votre_image2.jpg')
# 1 ligne, 2 colonnes
fig, zones_trace = plt.subplots(1, 2, figsize=(8, 8))
zones_trace[0].imshow(image1)
zones_trace[0].set_title('Image 1')
zones_trace[0].axis('off')
zones_trace[1].imshow(image2)
zones_trace[1].set_title('Image 2')
zones_trace[1].axis('off')
plt.tight_layout()
plt.show()
```



19

19

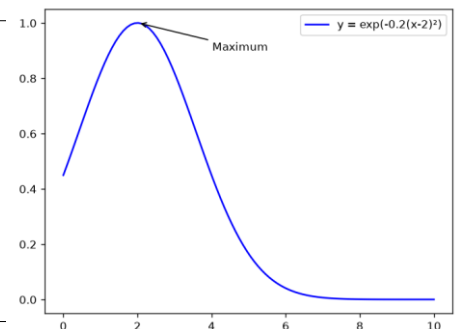
Problème

Comment ajouter une annotation sur une figure ?

Solution

```
x = np.linspace(0, 10, 100)
y = np.exp(-0.2 * ((x - 2) ** 2))
plt.plot(x, y, label="y = exp(-0.2(x-2)^2)", color="blue")
x_max = x[np.argmax(y)]
y_max = max(y)
plt.annotate("Maximum",
             xy=(x_max, y_max),
             xytext=(x_max + 2, y_max - 0.1),
             arrowprops=dict(facecolor="black",
                             arrowstyle="->"))

plt.legend()
plt.show()
```



20

20

Problème

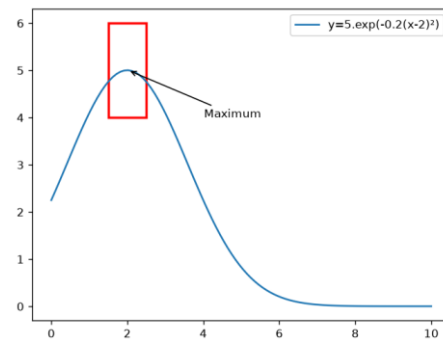
Comment mettre en évidence un maximum, une zone particulière, une valeur critique ?

Solution

```
x = np.linspace(0, 10, 100)
y = 5 * np.exp(-0.2 * ((x - 2) ** 2))
fig, ax = plt.subplots()
ax.plot(x, y, label="y=5.exp(-0.2(x-2)²)")
x_max = x[np.argmax(y)]
y_max = max(y)
ax.annotate("Maximum", xy=(x_max, y_max),
            xytext=(x_max + 2, y_max - 1),
            arrowprops=dict(facecolor="black",
                            arrowstyle="->"))

ax.legend()
rect = plt.Rectangle((1.5, 4), 1, 2,
                    linewidth=2,
                    edgecolor="red",
                    facecolor="none")

ax.add_patch(rect)
plt.show()
```



21

21

Problème

Comment enregistrer une figure pour un rapport ou un diaporama ?

Solution

```
plt.plot(t, f_bruite)
plt.title("Signal bruité")
plt.savefig("results/figure.pdf") # vectoriel : pour publication
plt.savefig("results/figure.png", dpi=300) # pour diaporama
plt.show()
```

22

22