

Vers l'autonomie

fonctions · while · fichiers · dérivée · polyfit · Euler · dichotomie

ÉCRIRE UNE FONCTION

définir `def f(x, y):` + bloc décalérenvoyer `return x * y`appeler `E = f(2.0, 3.0)`sans `return` la fonction renvoie `None`afficher ≠ renvoyer `print` montre, `return` rendfonction en paramètre `dichotomie(f, 0, 1, 1e-6)`

RÉPÉTER

tant que `while A > 0.01 * A0:`n fois `for i in range(n):`ajouter à une liste `L.append(x)`exponentielle `np.exp(x)`

LIRE UN FICHIER CSV

lire `t, y = np.loadtxt('chute.csv',
delimter=';', skiprows=1, unpack=True)`séparateur `delimter=';'`sauter l'en-tête `skiprows=1`séparer les colonnes `unpack=True`

DÉRIVER DES MESURES

vitesse centrée `(x[2:] - x[:-2]) / (t[2:] - t[:-2])`ses dates `t[1:-1]` (deux valeurs de moins)dernière valeur `x[-1]`flèches `plt.quiver(x, y, ax, ay, ...)`

MODÉLISER PAR UNE DROITE

 $y = ax + b$ `a, b = np.polyfit(x, y, 1)`tracer le modèle `plt.plot(x, a * x + b, 'r-')`proportionnalité points alignés et $b \approx 0$ exponentielle tracer `np.log(y)` : pente $-\lambda$

EULER ET DICHOTOMIE

 $y' = ay + b$ `y = y + (a * y + b) * dt`pas `dt` au moins dix fois plus petit que τ milieu `m = (a + b) / 2`garder la bonne moitié `if f(a) * f(m) <= 0: b = m
else: a = m`précision `while b - a > 1e-6:`

EXEMPLE COMMENTÉ 1 · DU FICHIER DE MESURES AU MODÈLE

```
import numpy as np
import matplotlib.pyplot as plt
```

```
# 1. lire le fichier : séparateur ; et une ligne d'en-tête « t (s);y (m) » à sauter
t, y = np.loadtxt('chute.csv', delimiter=';', skiprows=1, unpack=True)
```

```
# 2. dériver : vitesse centrée, 2 valeurs de moins que les positions
vy = (y[2:] - y[:-2]) / (t[2:] - t[:-2])
tv = t[1:-1] # les dates des vitesses
print(len(t), 'positions,', len(vy), 'vitesses')
```

```
# 3. modéliser : droite  $vy = a \times t + b$  ; les abscisses d'abord ; ici  $a \approx -g$  et  $b \approx 0$ 
a, b = np.polyfit(tv, vy, 1)
print('a =', round(a, 2), 'm/s²', 'b =', round(b, 3), 'm/s')
```

```
# 4. tracer mesures et modèle, avec des tableaux de même longueur
plt.plot(tv, vy, 'b+', label='mesures')
plt.plot(tv, a * tv + b, 'r-', label='modèle')
plt.xlabel('t (s)')
plt.ylabel('vy (m/s)')
plt.legend()
plt.show()
```

```
14 positions, 12 vitesses
a = -9.79 m/s²    b = -0.005 m/s
```

EXEMPLE COMMENTÉ 2 · LA MÉTHODE D'EULER, RANGÉE DANS UNE FONCTION

```
import numpy as np

def euler(a, b, y0, dt, n):      # résout  $y' = a \times y + b$ , en  $n$  pas de durée  $dt$ 
    y = y0                      # valeur initiale  $y(0)$ 
    liste_y = [y]
    for i in range(n):
        y = y + (a * y + b) * dt  # un pas :  $y(t + dt) \approx y(t) + y'(t) \times dt$ 
        liste_y.append(y)
    return liste_y              # renvoyer, pas afficher : on calcule ensuite avec

# charge d'un condensateur :  $du/dt = (E - u) / \tau$ , soit  $a = -1/\tau$  et  $b = E/\tau$ 
E, tau = 12.0, 22.0             # en V et en s
u = euler(-1 / tau, E / tau, 0, 1.0, 140)  # pas de 1 s, de 0 à 140 s
print(len(u), 'valeurs ; u(22 s) =', round(u[22], 2), 'V')  # pas de 1 s : u[22] est à t = 22 s
print('solution exacte :', round(E * (1 - np.exp(-1)), 2), 'V')
```

```
141 valeurs ; u(22 s) = 7.69 V
solution exacte : 7.59 V
```

À la date τ , 63 % de E . L'écart avec la solution exacte diminue si l'on réduit le pas. La même fonction résout la décroissance radioactive : `euler(-lam, 0, N0, dt, n)`.

EXEMPLE COMMENTÉ 3 · UNE DICHOTOMIE RÉUTILISABLE

```
import numpy as np

def dichotomie(f, a, b, precision):  # f(a) et f(b) doivent être de signes contraires
    while b - a > precision:         # tant que l'encadrement est trop large
        m = (a + b) / 2
        if f(a) * f(m) <= 0:         # le changement de signe est dans [a ; m]
            b = m
        else:                         # sinon, il est dans [m ; b]
            a = m
    return (a + b) / 2

# acide éthanoinique à  $c = 1,0 \times 10^{-2}$  mol/L :  $x = [H3O+]$  vérifie  $x^2 + K_a x - K_a c = 0$ 
c = 1.0e-2
Ka = 10 ** (-4.76)

def f(x):
    return x ** 2 + Ka * x - Ka * c

x = dichotomie(f, 0, c, 1e-9)      # le nom f, sans parenthèses : on donne la fonction
print('[H3O+] =', round(x, 6), 'mol/L')
print('tau =', round(x / c, 3), '; pH =', round(-np.log10(x), 2))
```

```
[H3O+] = 0.000408 mol/L
tau = 0.041 ; pH = 3.39
```

Taux d'avancement final $\tau \approx 4\%$: la réaction de l'acide sur l'eau est très limitée. Chaque tour divise l'encadrement par deux : 24 tours ici, pour passer de 10^{-2} à moins de 10^{-9} .

LES PIÈGES

renvoyer	<code>return</code>	Une fonction qui fait <code>print</code> au lieu de <code>return</code> renvoie <code>None</code> : <code>TypeError ... 'NoneType'</code> au calcul suivant.
variable locale	dans <code>def</code>	Une variable créée dans une fonction n'existe pas en dehors : <code>NameError</code> .
boucle infinie	<code>while</code>	Une variable de la condition doit changer dans la boucle, sinon elle ne s'arrête jamais.
en-tête	<code>skiprows=1</code>	Sans lui : <code>could not convert string 't (s)' to float</code> .
séparateur	<code>delimiter=';'</code>	Celui du fichier : regarder une ligne de données avant d'écrire le programme.
0,04	<code>0.04</code>	numpy ne lit pas la virgule décimale : l'exporter avec un point.
longueurs	<code>t[1:-1]</code>	Les vitesses ont deux valeurs de moins que les positions : <code>ValueError</code> au tracé avec <code>t</code> .
<code>polyfit(x, y)</code>	abscisses d'abord	<code>np.polyfit(y, x, 1)</code> calcule une autre droite, sans message d'erreur.
<code>ln, log</code>	<code>np.log</code>	Pour linéariser une exponentielle, c'est <code>ln</code> : <code>np.log</code> , pas <code>np.log10</code> .
pas d'Euler	<code>dt</code> petit	Un pas plus long que τ donne un résultat faux, parfois absurde.

NOUVEAUX MESSAGES D'ERREUR

<code>FileNotFoundError</code>	fichier introuvable : nom ou extension mal écrits, fichier dans un autre dossier
<code>ValueError</code> de <code>loadtxt</code>	<code>could not convert string ... to float</code> : en-tête non sauté, mauvais séparateur, virgule décimale
<code>ValueError</code> <code>unpack</code>	<code>not enough values to unpack</code> : plus de variables que de colonnes
<code>TypeError</code> <code>NoneType</code>	calcul avec <code>None</code> : une fonction sans <code>return</code>
boucle trop longue	arrêt au bout de 5 s : la condition du <code>while</code> reste vraie

CE QUE JE SAIS FAIRE

- ☐ Écrire et appeler une fonction ; distinguer `return` et `print`.
- ☐ Répéter tant qu'une condition est vraie, avec `while`.
- ☐ Lire un fichier de mesures avec `np.loadtxt`.
- ☐ Calculer vitesse, accélération, vitesse volumique.
- ☐ Modéliser par une droite ; linéariser une exponentielle.
- ☐ Résoudre $y' = ay + b$ par la méthode d'Euler.
- ☐ Résoudre une équation par dichotomie.
- ☐ Écrire un programme complet, du fichier au modèle.

Point à revoir :