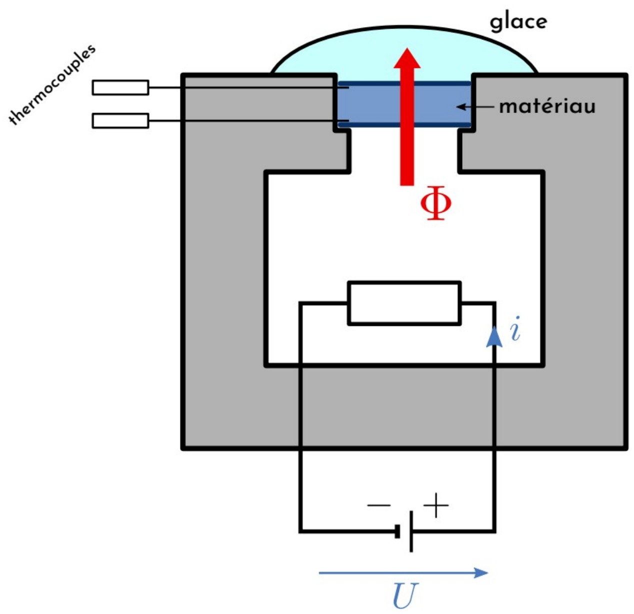
Mesure de conductivité thermique

Capacités exigibles

- Mettre en œuvre un dispositif de mesure de conductivité thermique.
- À l'aide d'un langage de programmation, résoudre l'équation de la diffusion thermique à une dimension par une méthode des différences finies dérivée de la méthode d'Euler explicite de résolution des équations différentielles ordinaires.

I Documents

Document 1 : Mesure d'une conductivité



Un four est alimenté en puissance via un circuit électrique en régime continu. La chambre est censée supporter des température **inférieures à** $60\,°\mathrm{C}$. L'enceinte est calorifugée, sauf au niveau d'une ouverture où on laissera s'échapper le flux thermique à travers le matériau qui nous intéresse. Pour augmenter l'efficacité du transfert, on place une poche étanche de glace pilée au dessus du matériau.

La mesure de la puissance se fait via un multimètre dans le circuit d'alimentation du four, et la différence de température est obtenue à l'aide de deux thermocouples placées de chaque côté du matériau.

En régime permanent, on peut montrer que le flux Φ est lié à la conductivité thermique du matériau λ, à sa section S, son épaisseur e et la différence de température ΔT d'un côté et de l'autre :

$$\Phi = \frac{\lambda S}{e}\Delta T$$

Document 2 : Équation de la chaleur

On rappelle la loi d'évolution d'un profil de température T dans le temps :

$$\frac{\partial T}{\partial t} = D\Delta T$$

Avec D le **coefficient de diffusion**, proportionnel à la conductivité λ et Δ l'opérateur **laplacien**. En particulier en une dimension x :

$$\frac{\partial T}{\partial t} = D\frac{\partial^2 T}{\partial x^2}$$

II Énoncé

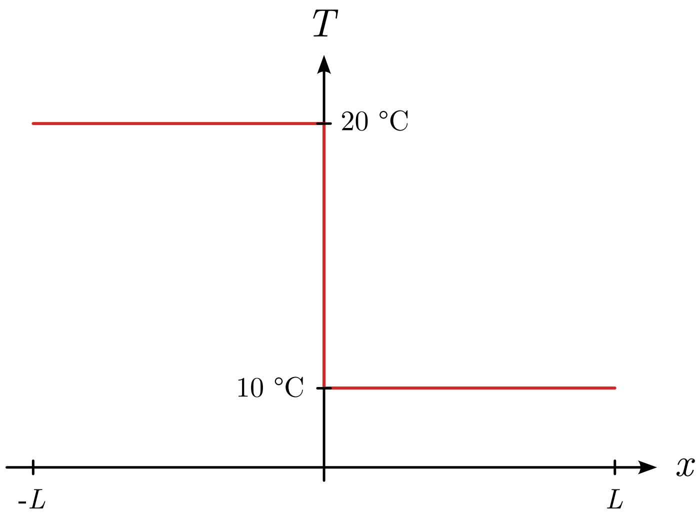
A Mesure de conductivité

1. Donner le lien entre U, i et Φ.

2. Imaginer un protocole permettant de mesurer la conductivité d'un matériau donné. Essayer d'inclure une méthode de régression linéaire.

3. Réaliser ce protocole et donner la conductivité d'un matériau.

B Simulation numérique



Le but ici est de simuler numériquement l'évolution d'un système en une dimension ayant pour condition initiale :

$$T(t=0,x) = \begin{cases} T_1 = 10\,°\mathrm{C} & \text{pour } x \in [-L, 0[\\ T_2 = 20\,°\mathrm{C} & \text{pour } x \in]0, L] \end{cases}$$

Pour cela, vous partirez du code donné, que vous rechercherez à compléter (remplacer les ? par de vraies lignes de code).

1. On va commencer par implémenter le calcul du laplacien (dérivée seconde).
   a) Faire les développements limités à l'ordre 2 de $T(t, x-\mathrm{d}x)$ et $T(t, x+\mathrm{d}x)$ et en déduire la relation suivante :

$$\frac{\partial^2 T}{\partial x^2}(t,x) = \frac{1}{\mathrm{d}x^2}\left(T(t,x+\mathrm{d}x) - 2T(t,x) + T(t,x-\mathrm{d}x)\right)$$

   b) En déduire comment compléter la fonction `laplacien(Y)`.

2. À présent, il faut calculer l'évolution temporelle. Pour cela, on utilise la **méthode d'Euler**.
   - Écrire le développement limité de $T(t+\mathrm{d}t, x)$ et en déduire comment compéter la ligne 40 (en utilisant notamment la fonction `laplacien`).
   - On souhaite simuler un système isolé de l'extérieur, indiquer théoriquement les valeurs de

$$\frac{\partial T}{\partial x}(t, x=\pm L)$$

   - En déduire un complétion des lignes 41 42.
     NB : l'indice `-1` dans une liste ou un tableau signifie "dernière valeur".
   - En complétant les lignes `plt.plot(?, ?)`, affichez le profil de température à l'instant initial, après 30% de t_{max} et à l'instant final t_{max}.

III Annexe

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

# Fonctions utiles

def laplacien(Y):

    dx = ?
    Z = np.zeros(Nx)        # Créer une ligne contenant Nx zéros
    for i in range(?, ?):
        Z[i] = ?
    return Z

# Grandeurs physiques, en système international

T1 = 10      # en °C
T2 = 20      # en °C
D = 12e-6    # en m² / s
L = 0.01     # en m

# Grandeurs de la modélisation, en système international

dt = .001        # Pas de temps de la simulation
t_fin = 10       # Temps final de la simulation
Nt = int(t_fin / dt)    # Nombre d'instants que l'on va calculer

Nx = 100                        # Nombre de points selon l'axe x
x = np.linspace(-L, L, Nx)  # Tableau contenant toutes les abscisses de -L à L

# Initialisation

T = np.zeros((Nt, Nx))  # Créer un tableau de Nt lignes et Nx colonnes uniquement rempli de 0

T[0, : Nx // 2] = T1    # Le début de la première ligne est rempli de T1
T[0, Nx // 2 :] = T2    # La fin de la première ligne est rempli de T2

# Calcul de l'évolution

for k in range(?, ?):
    T[k+1] = ?
    T[k+1, 0] = ?
    T[k+1, -1] = ?

plt.plot(?, ?)
plt.plot(?, ?)
plt.plot(?, ?)
plt.xlabel('Position (m)')
plt.ylabel('Température (°C)')
plt.show()
```