Complément TDEM4 – Capacités numériques : équation de Laplace

Capacités exigibles	ChEM4
Équation de Poisson et équation de Laplace de l'électrostatique. Capacité numérique : à l'aide d'un langage de programmation résoudre numériquement l'équation de Laplace à une ou deux dimensions, les conditions aux limites (CL) étant fixées. Choisir un pas spatial adapté à la résolution numérique d'une équation de Laplace dans un contexte physique donné. Implémenter un schéma itératif fourni pour résoudre l'équation de Laplace avec des CL de type Dirichlet.	Annexe 2

CCS2 MPI 2023 – Partie IV – Résolution équation de Laplace 1D-2D

Ce sujet utilise la syntaxe des annotations pour préciser le type des arguments et du résultat des fonctions Python. Ainsi,

def maFonction(n:int, x:float)-> (int, np.ndarray):

signifie que la fonction maFonction admet deux arguments, le premier est un entier, le second un nombre à virgule flottante et qu'elle renvoie un couple dont le premier élément est un entier et le deuxième un tableau numpy. Il n'est pas demandé aux candidats de recopier les entêtes avec annotations telles qu'elles sont fournies dans ce sujet, ils peuvent utiliser des entêtes classiques.

IV Étude d'un congélateur

Afin de reproduire les conditions d'un environnement à très basse température, des congélateurs sont utilisés par les constructeurs de téléphones portables. Ces congélateurs sont l'objet de cette partie. La température au sein du congélateur utilisé pour mettre à l'épreuve le téléphone portable est de −18 °C et celle de la salle de laboratoire est de 19 °C.

IV.B – Résolution approchée de l'équation de la diffusion thermique dans le congélateur

On s'intéresse à un modèle en deux dimensions du congélateur : voir la géométrie représentée en haut de la figure 13a. On modélise la paroi du congélateur par l'espace compris entre deux carrés : la température sur le carré le plus grand est égale à la température du laboratoire et la température sur le carré le plus petit est égale à la température intérieure du congélateur.

Nous étudions une méthode numérique permettant de déterminer de façon approchée la température qui règne, en régime permanent, entre deux carrés. Celle-ci vérifie l'équation de Laplace,

$$\frac{\partial^2 T(x,y)}{\partial x^2} + \frac{\partial^2 T(x,y)}{\partial y^2} = 0,$$

où $T(x,y)$ est la température au sein de la paroi du congélateur au point de coordonnées (x,y).

On utilise pour cela une méthode des différences finies associée à une méthode de relaxation. Cette dernière permet de calculer le champ de température par une méthode itérative qui s'appuie sur les valeurs de température aux frontières : à savoir à l'intérieur et à l'extérieur de la paroi d'isolant.

IV.B.1) Modèle à une dimension

On envisage tout d'abord la résolution approchée d'un problème simplifié à une dimension $\frac{\mathrm{d}^2 T(x)}{\mathrm{d}x^2} = 0$ sur l'intervalle $[0, D]$ où D est l'épaisseur du mur. On divise cet intervalle en N points, numérotés de 0 à $N-1$, régulièrement espacés de Δx (figure 12). Cette division est appelée « discrétisation ». La distance Δx est appelée le « pas de discrétisation ». À l'intérieur du mur (frontières intérieure et extérieure exclues) se trouvent donc $N-2$ points. On cherche à obtenir la température en ces points particuliers de manière itérative.

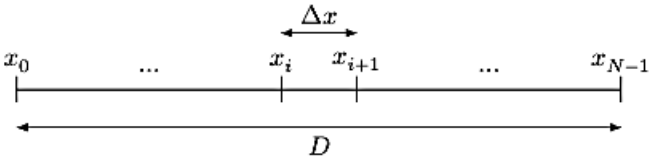

Figure 12 Discrétisation dans la direction x

Q 40. Exprimer Δx en fonction de N et de l'épaisseur du mur D. En déduire l'abscisse x_i du i-ème point en fonction de i et Δx, sachant que $x_0 = 0$ et $x_{N-1} = D$.

Q 41. À l'aide d'un développement limité de la fonction $x \mapsto T(x)$, donner une expression de $T(x + \Delta x)$ à l'ordre 3 ($o(\Delta x^3)$) en fonction de T et de ses dérivées partielles par rapport à x évaluées en (x). De même, donner une expression de $T(x - \Delta x)$ à l'ordre 3.

Q 42. En déduire une expression approchée à l'ordre 1 ($o(\Delta x)$) de $\dfrac{\mathrm{d}^2T(x)}{\mathrm{d}x^2}$ en fonction de $T(x+\Delta x)$, $T(x-\Delta x)$, $T(x)$ et Δx.

On note T_i la température $T(x_i)$, évaluée au point d'abscisse x_i. De même, on note $T_{i+1} = T(x_i + \Delta x)$ et $T_{i-1} = T(x_i - \Delta x,)$.

Q 43. Montrer que les valeurs discrètes T_i de la température vérifient

$$T_i = \frac{1}{2}\left(T_{i-1} + T_{i+1}\right).$$

Cette équation nous permet de calculer la température en un point à partir du point précédent et du point suivant. Il faudra faire attention lors de la programmation au fait que cette équation n'est pas valable en 0 et $N-1$ où son calcul fait appel aux indices $i = -1$ et $i = N$ qui ne sont pas définis.

IV.B.2) Modèle à deux dimensions x et y

Pour le problème bidimensionnel, on effectue une discrétisation selon les axes (Ox) et (Oy) avec le même pas de discrétisation dans les deux directions. Dans ce cas, on admettra que $T_{i,j}$, la température en un point d'abscisse x_i et d'ordonnée y_j vérifie

$$T_{i,j} = \frac{1}{4}\left(T_{i-1,j} + T_{i+1,j} + T_{i,j+1} + T_{i,j-1}\right).$$

L'algorithme de relaxation permet d'obtenir des valeurs de la température vérifiant cette relation et respectant les conditions aux limites imposées. Les températures $T_{i,j}^{(k+1)}$ déterminées à l'étape $k+1$ sont calculées à partir du champ de température déterminé à l'étape k par

$$T_{i,j}^{(k+1)} = \frac{1}{4}\left(T_{i-1,j}^{(k)} + T_{i+1,j}^{(k)} + T_{i,j+1}^{(k)} + T_{i,j-1}^{(k)}\right).$$

À partir d'un tableau initial $T_{i,j}^{(0)}$, choisi arbitrairement, mais vérifiant les conditions aux limites, cette procédure converge vers un tableau vérifiant la relation recherchée. La convergence est évaluée en calculant à chaque étape le résidu $r = \max\left(T_{i,j}^{(k)} - T_{i,j}^{(k-1)}\right)$, pour $k \geqslant 1$. Tant que la valeur de r excède une valeur choisie $r_{\max}$, le calcul est répété.

La figure D du document réponse propose un extrait du programme utilisé. Avec quelques instructions de tracé, ce programme produit le graphique de la figure 13b.

Q 44. Quelle est la valeur choisie pour $r_{\max}$? La figure 13b est obtenue avec le programme tel qu'il est écrit. Ce programme permet-il vraiment de s'assurer de la qualité de la convergence de l'algorithme ?

Q 45. Proposer une modification du programme permettant de vérifier que le champ de température obtenu vérifie bien le critère de convergence (les figures 13a et 13c ont été obtenues après correction). On ne demande pas d'ajouter les lignes de code relatives au tracé des graphiques.

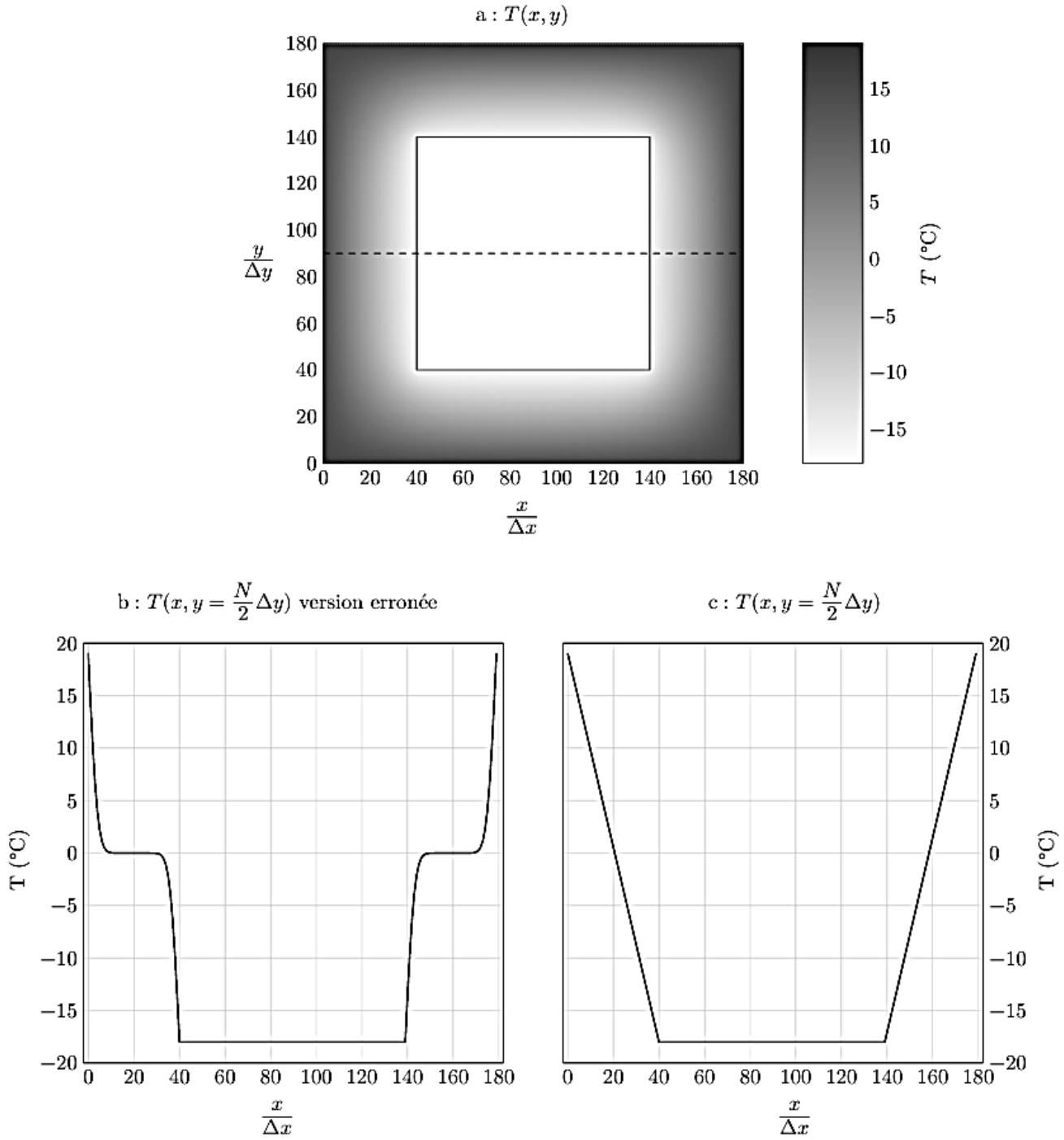

Figure 13

a : contours de $T(x, y)$ corrects dans les parois du congélateur
b : tracé erroné de $T(x, y)$ le long de la ligne horizontale en pointillés
c : tracé correct de $T(x, y)$ le long de la ligne horizontale en pointillés

Questions 44 et 45

```
1   import numpy as np

2   N_int = 100 # nombre de points par axe à l'intérieur du congélateur
3   N_paroi = 40 # nombre de points dans l'épaisseur de chaque paroi
4   T_out = 19.0 # température extérieure en °C
5   T_in = -18.0 # température intérieure en °C

6   res = 1 # erreur résiduelle initialisée à une valeur élevée
7   compteur = 0 # compteur d'itérations
8   N = N_int + 2 * N_paroi # nombre total de points sur chaque axe (x et y)

9   def cond_lim(T:np.ndarray) -> None:
10      # Fixer les conditions aux limites (modifie T)
11      T[N_paroi:N_int+N_paroi, N_paroi:N_int+N_paroi] = T_in # intérieur
12      T[:, 0] = T_out # extérieur de la paroi gauche
13      T[:, N-1] = T_out # extérieur de la paroi droite
14      T[0, :] = T_out # extérieur de la paroi supérieure
15      T[N-1, :] = T_out # extérieur de la paroi inférieure

16  def iterate(T:np.ndarray, old:np.ndarray) -> None:
17      # Calculer la température moyenne en chacun des points de la grille
18      T_s = old[0:N-2, 1:N-1] # les valeurs supérieures de l'ancienne grille [i-1, j]
19      T_i = old[2:N, 1:N-1] # les valeurs inférieures de l'ancienne grille [i+1; j]
20      T_g = old[1:N-1, 0:N-2] # les valeurs de gauche de l'ancienne grille [i, j-1]
21      T_d = old[1:N-1, 2:N] # les valeurs de droite de l'ancienne grille [i, j+1]
22      T[1:N-1, 1:N-1] = (T_s + T_i + T_g + T_d) / 4

23  T = np.zeros((N, N)) # grille de calcul
24  cond_lim(T) # la valeur de départ (arbitraire) doit vérifier les conditions aux limites
25  while (res > 1e-3 and compteur < 20):
26     old = np.copy(T) # mémoriser le résultat du tour précédent
27     iterate(T, old) # calculer la nouvelle grille de température
28     cond_lim(T) # s'assurer de respecter les conditions aux limites
29     res = np.max(np.abs(T - old)) # variation apportée par cette itération
30     compteur += 1 # comptabiliser une nouvelle itération
```

Figure D Calcul du profil de température dans la paroi du congélateur

Instructions Python : opérations sur les tableaux numpy

- `np.zeros(n)`, `np.zeros((n, m))` crée respectivement un vecteur à `n` éléments ou une matrice à `n` lignes et `m` colonnes dont les coefficients sont tous nuls.
- `np.ones(n)`, `np.ones((n, m))` fonctionne comme `np.zeros` en initialisant chaque coefficient à la valeur un.
- `np.min(a)`, `np.max(a)` renvoie la valeur du plus petit (respectivement plus grand) élément du tableau `a`.
- `np.abs(a)` crée un tableau similaire au tableau `a` dont les coefficients sont les valeurs absolues des coefficients de `a`.