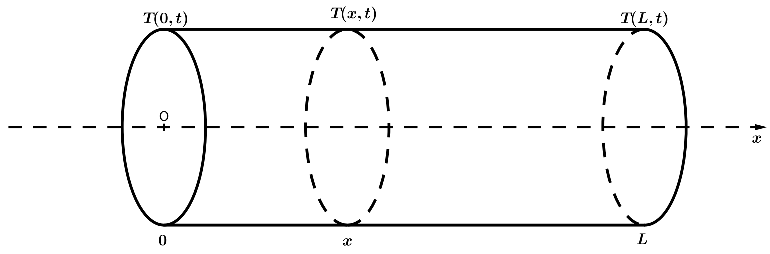
Informatique et simulation numérique

On considère un cylindre d'axe (O,x), de section S, de conductivité thermique λ, de masse volumique ρ, de capacité thermique massique c et de longueur L.



On s'intéresse à la température du cylindre. On suppose que cette température ne dépend que du temps t et de la valeur de x. Donc à un temps t fixé, la température est la même pour toute la section d'abscisse x.
On notera $T(x,t)$ la température au temps t pour tous les points du cylindre d'abscisse x.
Au début tous les points du cylindre sont à la température 283 K ($\simeq 10^o$C).
Au temps $t=0$ on va supposer que, de façon abrupte, la température du cylindre pour les points d'abscisse 0 descend à 258 K ($\simeq -15^o$C) et la température du cylindre pour les points d'abscisse L monte à 298 K ($\simeq 25^o$C).
On supposera ensuite qu'à chaque instant t la température du cylindre pour les points d'abscisse 0 est maintenu à 258 K et la température du cylindre pour les points d'abscisse L est maintenu à 298 K.
L'évolution de la température $T(x,t)$ dans le cylindre est donnée par l'équation de la chaleur :

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

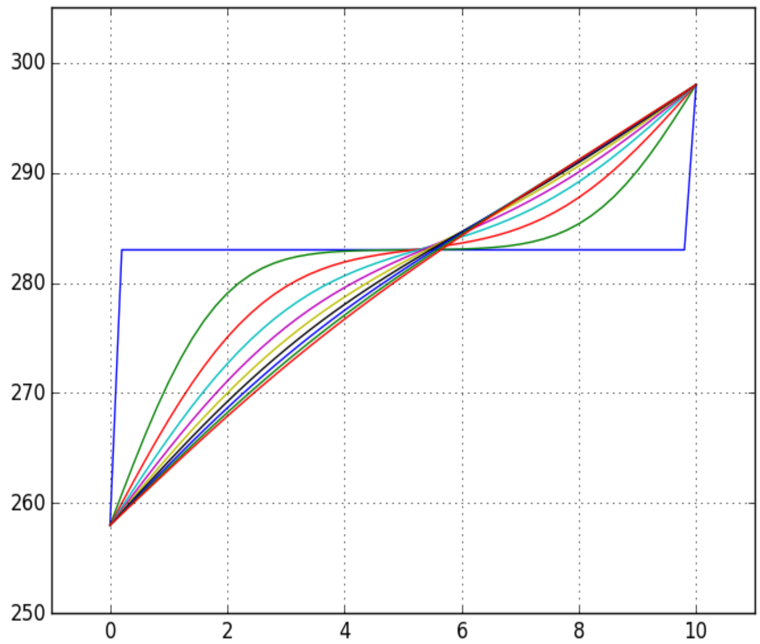
avec

$$D = \frac{\lambda}{\rho\, c}$$

et avec les conditions initiales,

$$\forall t,\ T(0,t) = 258\,K, \quad \forall t,\ T(L,t) = 298\,K \quad \text{et si} \quad 0 < x < L,\ T(x,0) = 283\,K$$

Par exemple le graphique ci-dessous représente la température d'un cylindre de 10cm pour diverses valeurs du temps.



Partie I : résolution numérique de l'équation de la chaleur

On cherche à résoudre l'équation de la chaleur en régime variable de façon numérique. Pour cela on commence par discrétiser cette équation : on cherche une valeur approchée des températures en plusieurs points du cylindre, pour plusieurs instants successifs.

- Pour l'approximation spatiale, on utilisera un nombre N_x de subdivisions du segment $[0, L]$, et un pas $\mathrm{d}x = \dfrac{L}{N_x}$. On posera, pour $k \in [\![0, N_x]\!]$, $x_k = k\,\mathrm{d}x$.
- Pour l'approximation temporelle, on supposera que le temps varie entre 0 (temps initial) et un temps final noté *duree*. On utilisera un nombre N_t de subdivisions du segment $[0,duree]$, et un pas de $\mathrm{d}t = \dfrac{duree}{N_t}$. On posera, pour $i \in [\![0, N_t]\!]$, $t_i = i\,\mathrm{d}t$.

Nous allons utiliser les méthodes d'Euler explicite et implicite pour faire une approximation de la valeur exacte de $T[x_k, t_i]$. On notera T_k^i l'approximation de $T[x_k, t_i]$.

1. En partant de l'équation de la chaleur, en utilisant avec discernement les deux méthodes d'Euler précitées, justifier que, si $0 < k < N_x$ et $0 \leqslant i < N_t$, on a :

$$T_k^{i+1} = T_k^i + \frac{\mathrm{d}t\, D}{(\mathrm{d}x)^2}\,\left(T_{k+1}^i - 2\,T_k^i + T_{k-1}^i\right)$$

2. Analyse d'un script Python.

a. La capture d'écran ci-dessous donne une proposition de script Python qui permet d'obtenir la liste des températures pour chaque valeur des deux subdivisions de $[0, L]$ et $[0,duree]$.

On a pris $D = 0.001$.

On a supposé que $L = 10$, puis on a choisi $N_x = 10$ pour l'approximation spatiale.

Pour le temps on a choisi *duree* = 5000 avec $N_t = 400$.

```
11 D=0.001
12
13 L=10
14 N_x=5
15 dx=L/(N_x)
16
17 duree=5000
18 N_t=400
19 dt=duree/(N_t)
20
21 T_debut=[283 for i in range(N_x+1)]
22 T_debut[0]=258
23 T_debut[N_x]=298
24 T=[T_debut]
25
26 t=1
27 while t<=N_t:
28     T_suivant=[]
29     for x in range(N_x+1):
30         if x==0:
31             T_suivant.append( ? )
32         elif x==N_x:
33             T_suivant.append( ? )
34         else:
35             a=  ?
36             T_suivant.append(a)
37     T.append(T_suivant)
38     t=t+1
```

i. Compléter les lignes 31, 33 et 35 en remplaçant les points d'interrogations par des valeurs ou des commandes appropriées.
Remarque. Soit une liste de listes : $L = [\,[1,2,3],[4,5,6],[7,8,9],[10,11,12]\,]$.
On a $L[0][0] = 1, L[0][1] = 2, L[1][0] = 4, L[2][1] = 8, L[3][2] = 12$.

ii. Justifier la terminaison de l'algorithme situé entre les lignes 26 et 38.

iii. Calculer la complexité du script situé entre les lignes 27 et 38.
On comptera les opérations arithmétiques et la commande `append`.

b. Étude matricielle.

On note $X_i = \begin{pmatrix} T_0^i \\ \vdots \\ T_k^i \\ \vdots \\ T_{N_x}^i \end{pmatrix} \in \mathcal{M}_{N_x+1,1}$ la matrice unicolonne des valeurs prises par les températures au points de la sudivison $[0, L]$ au temps $t = i\,\mathrm{d}t$.

i. Déterminer une matrice carrée A telle que $X_{i+1} = A\,X_i$.

ii. Rappeller succintement la méthode du pivot de Gauss pour une matrice carrée quelconque.
Dans le cas général, quelle est la complexité de cette méthode ?

iii. Si on considère la matrice A ci-dessus, proposer un algorithme adapté de la méthode du pivot de Gauss de complexité inférieur à la complexité précédente. On supposera que tous les “ pivots ” sont non nuls.

Partie II : analyse d'un fichier

On utilise les notations de la partie **I**.

On suppose que les températures obtenues par le script Python sont reportées dans un fichier nommé `Temperatures.txt`.

Le fichier aura $N_t + 1$ lignes, chaque ligne correspondant à une valeur du temps t.

Chaque ligne du fichier contient les $N_x + 1$ températures de la subdivision de $[0, L]$ pour le temps t.

Si par exemple $N_t = 4$ et $N_x = 5$, le fichier pourra être du type :

258, 283 , 283 , 283 , 283 , 298

258, 277.10 , 282.22 , 283.41 , 286.54 , 298

258, 273.68 , 280.91 , 283.98 , 288.55 , 298

258, 271.56 , 279.70 , 284.33 , 289.74 , 298

258, 270.17 , 278.72 , 284.48 , 290.43 , 298

On importe ce fichier dans Python avec le script suivant : `Fichier=open('Temperatures.txt','r')`

La commande `Lignes=Fichier.readlines()` crée une liste qui, dans l'exemple précédent, est :

[' 258 , 283 , 283 , 283 , 283 , 298 \n' , ' 258, 277.10 , 282.22 , 283.41 , 286.54 , 298 \n' , ' 258, 273.68 , 280.91 , 283.98 , 288.55 , 298 \n' , ' 258, 271.56 , 279.70 , 284.33 , 289.74 , 298 \n' , ' 258, 270.17 , 278.72 , 284.48 , 290.43 , 298 \n']

La commande `print(Lignes[1].strip().split(','))` affiche alors :

[' 258 ' , ' 277.10 ' , ' 282.22 ' , ' 283.41 ' , ' 286.54 ' , ' 298 ']

a. Proposer un script Python qui affiche la liste des moyennes des températures pour chaque temps t, soit la moyenne des températures pour chaque ligne du fichier `Temperatures.txt`.

b. On constate sur la figure des températures qui se trouve au début de l'énoncé que les courbes " tendent " vers la droite passant par les points de coordonnées (0,258) et (10,298).

  i. Écrire une équation de cette droite.

  ii. Si on note $y = f(x)$ l'équation précédente, proposer un script qui, pour tout x, retourne la valeur de $f(x)$.

  iii. On se propose, pour chaque valeur du temps t, d'évaluer l'écart maximum en valeur absolue entre les ordonnées des points de la courbe des températures et les ordonnées des points de la droite précédente, pour des points de même abscisse.

  Proposer un script Python qui affiche la liste de ces écarts pour chaque temps t.

  Remarque : la commande `abs(a)` retourne la valeur absolue du réel a.

c. On suppose, qu'à partir du fichier précédent, on a créé une seule liste de toutes les températures, liste nommée `Liste_des_temperatures`, les températures étant dans cette liste des réels (et pas des chaînes de caractères). Par exemple, dans le cas précdent, on aura :

`Liste_des_temperatures=` [258 , 283 , 283 , 283 , 283 , 298 , 258, 277.10 , 282.22 , 283.41 , 286.54 , 298 , 258, 273.68 , 280.91 , 283.98 , 288.55 , 298 , 258, 271.56 , 279.70 , 284.33 , 289.74 , 298 , 258, 270.17 , 278.72 , 284.48 , 290.43 , 298]

Proposer un algorithme de tri qui ordonne cette liste dans le sens croissant. Donner sans démonstration ce que vous connaissez concernant la complexité de cet algorithme de tri.