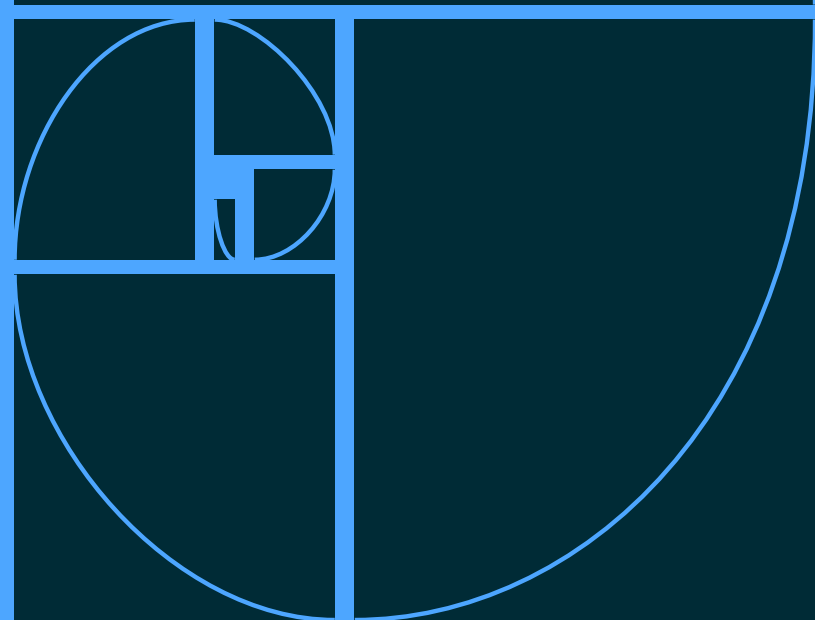
Projet: simulation numerique de l'equation de chaleur en 1D avec maillage non uniforme

Étudiant en Licence 2
Modélisation
Mathématique, Analyse et
Simulation Numérique



Yaya Touré / 76-593-03-47
Prof: M. Thiame | Ingenieur en simulation numerique

Centre National de Calculs Scientifiques | Diamniadio

Description de l'équation de chaleur:

Une équation différentielle partielle est une équation qui relie une fonction de plus d'une variable à ses dérivées partielles.
Pour introduire les EDP, nous allons faire une simulation numerique de l'equation de chaleur en 1D avec maillage non-uniforme (c'est-a-dire:$\Delta_{xi} \neq \Delta_x \ , \forall i$). En cours de route, nous allons travailler avec un coefficient de difusion $(\alpha = 1)$ l'équation de chaleur unidimensionnelle à partir de principes physiques et la résoudrons par ***la methode des volumes finis.***

Expression de l'equation de chaleur

$$\frac{\partial U}{\partial t} = \alpha . \frac{\partial^2 U}{\partial x^2} \text{ avec: } (\alpha = 1)$$

Comprendre la notion de maillage non-uniforme

De nombreuses structures optiques ont des propriétés extrêmement sensibles aux emplacements précis des interfaces matérielles. Cela est particulièrement vrai des structures qui ont des caractéristiques de résonance ou des interfaces à contraste d'indice élevé. *Le maillage non-uniforme* permet de prendre en compte la localisation géométrique précise de ces interfaces en utilisant des mailles fines dans les régions où les interfaces sont importantes et des mailles plus larges dans les régions massives où il n'y a pas d'interfaces importantes comme avec les matériaux homogènes.
La fonction de maillage automatique(c'est-á-dire : qui configure automatiquement le maillage non uniforme pour minimiser les effets de la dispersion numérique) qui fonctionne autrement comme elle le fait normalement, prendra alors également en compte les régions de remplacement, tout en continuant à réduire la dispersion numérique dans tout le volume de simulation. La fonction de remplacement de maillage est particulièrement utile pour les structures avec des interfaces métalliques.

Approximation par Volumes Finis | Introduction :

La méthode des Volumes Finis consiste à intégrer, sur des volumes élémentaires, les équations écrites sous forme intégrale. C'est une méthode particulièrement bien adaptée à la discrétisation spatiale des lois de conservation, contrairement aux Eléments Finis, et est ainsi très utilisée en mécanique des fluides.

Principes :

La discretisation numerique par methodes de volumes finies s'appuie :

1. sur une ecriture sous forme divergente des equations ;
2. sur une intégration des equations dans un volume de contrôle s'appuyant sur un maillage :
3. sur la construction de flux numériques avec python pour clore la construction.

Conditions:

Pour une barre de 1m de longueur nous avons les conditions suivantes :

✓conditions aux extrémités de la barre : $U(0,t) = U_g$ et $U(1,t) = U_d$ / g:gauche de la barre et d :droite de la barre

✓condition initiale : $U(x,0) = U_0$

Paramétres:

✓$[0,1]$: intervalle de discretisation en N mailles.

✓x_i : centre *(for i in range(1,N)*

✓$\Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$: taille constante

✓Δt : temps de discretisation soit $t = n\Delta t$ et U_i^n : une valeur approchee de la moyenne sur la maille de centre x_i / $i.e$: dans la $i-$éme maille

Discretisation Spaciale:

On sait que pour $(\alpha = 1)$ on a: $\dfrac{\partial U}{\partial t} = \dfrac{\partial^2 U}{\partial x^2}$ $(*)$ En intergant la relation $(*)$ de $x_{i-\frac{1}{2}}$ á $x_{i+\frac{1}{2}}$ on obtient :

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{\partial U}{\partial t} dx = \int_{x_{i-1/2}}^{x_{i+1/2}} \frac{\partial^2 U}{\partial x^2} dx$$

Utilisations du schéma d'Euler explicite:

En faisant une géneralisation, c'est-á-dire pour n élélements de cette évaluation de la dérivée temporaire(d'apres $(*)$) donne :

$$\Delta x \frac{U_i^{n+1} - U_i^n}{\Delta t} = \left[\left(\frac{\partial U}{\partial x} \right)_{x_{i+1/2}}^n - \left(\frac{\partial U}{\partial x} \right)_{x_{i-1/2}}^n \right]$$

Une determination de la valeur moyenne de $\frac{\partial U}{\partial x}$ sur le segment $[x_i, x_{i+1}]$,permet d'obtenir :

$$\left(\frac{\partial U}{\partial x}\right)^n_{x_{i+1/2}} = \frac{1}{\Delta x}\int_{x_i}^{x_{i+1}} \frac{\partial U}{\partial x} dx = \frac{U^n_{i+1} - U^n_i}{\Delta x}$$

Une determination de la valeur moyenne de $\frac{\partial U}{\partial x}$ sur le segment $[x_N, 1]$,permet d'obtenir :

$$\left(\frac{\partial U}{\partial x}\right)^n_{x_{N+1/2}} = \frac{2}{\Delta x}\int_{x_N}^{1} \frac{\partial U}{\partial x} dx = 2\frac{U^n_d - U^N_i}{\Delta x}$$

d : maille de droite

Une determination de la valeur moyenne de $\frac{\partial U}{\partial x}$ sur le segment $[x_{i-1}, x_i]$,permet d'obtenir :

$$\left(\frac{\partial U}{\partial x}\right)^n_{x_{i-1/2}} = \frac{2}{\Delta x}\int_{x_{i-1}}^{x_i} \frac{\partial U}{\partial x} dx = \frac{U_i^n - U_{i-1}^n}{\Delta x}$$

Une determination de la valeur moyenne de $\frac{\partial U}{\partial x}$ sur le segment $[0, x_1]$,permet d'obtenir :

$$\left(\frac{\partial U}{\partial x}\right)^n_{x_{1/2}} = \frac{2}{\Delta x}\int_{1}^{x_1} \frac{\partial U}{\partial x} dx = 2\frac{U_0^n - U_g^N}{\Delta x}$$

g: maille de gauche

En posant $\lambda = \frac{\Delta t}{\Delta x^2}$, la température à l'itération $n+1$ est donnée par :

$$U_i^{n+1} = \lambda U_{i-1}^n + (1-2\lambda) U_i^n + \lambda U_{i+1}^n \quad i \text{ variant de 2 à N-1}$$

$$U_1^{n+1} = 2\lambda U_g + (1-3\lambda) U_1^n + \lambda U_2^n$$

$$U_N^{n+1} = \lambda U_{N-1}^n + (1-3\lambda) U_N^n + 2\lambda U_d^n$$

Une ecriture sous forme matricielle donne :

$$\begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{N-2} \\ U_{N-1} \end{bmatrix}^{n+1} = \begin{bmatrix} 1-3\lambda & \lambda & 0 & \cdots & 0 \\ \lambda & 1-2\lambda & \lambda & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \lambda & 1-2\lambda & \lambda \\ 0 & 0 & 0 & \lambda & 1-3\lambda \end{bmatrix}$$

En tenant compte de l'expression :

$$\begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{N-2} \\ U_{N-1} \end{bmatrix}^n + 2\lambda \begin{bmatrix} U_g \\ 0 \\ \vdots \\ 0 \\ U_d \end{bmatrix}$$

Porgramme Python

Dans ce qui suit nous allons coder les expressions explicites :
U_i^{n+1}, U_1^{n+1} *et* U_N^{n+1}

Programme Python

```
10
11                                  #~~~~~~~Début_Projet~~~~~~~~~#
12  #Ceci est un encodage python / sur sublime text :->
13  #utf-8:coding
14  # Nous avons besoin que 2 Libraries
15  import numpy as np
16
17  import matplotlib.pyplot as plt
18
19  dx = 0.025
20  dt = 0.025
21
22  #Une vision de notre barre de longueur L (voir figure) -
23  """
24  t
25  ^
26  |
27  |
28  |------------------------------
29  |                              |
30  |                              |
31  |--------------------------------->x
32  """
```

```
"""
    #sur l'axe des x,
    tient compte du nombres de points,
     qui se trouvent dans le domaine [0;1]
"""
x = np.arange(0, 1+dx, dx)

"""
    #sur l'axe des t,
    tient compte du nombres de points,
     qui se trouvent dans le domaine [0;1]
"""
t = np.arange(0, 0.1+dt, dt)

ConditionsAuxbors = [0, 0] #
ConditionInitial = np.sin(np.pi*x)

n = len(x)
m = len(t)
T = np.zeros((n,m))

#En tenant compte des conditions aux bords
T[0, :] = ConditionsAuxbors[0]
T[-1,:] = ConditionsAuxbors[1]
```

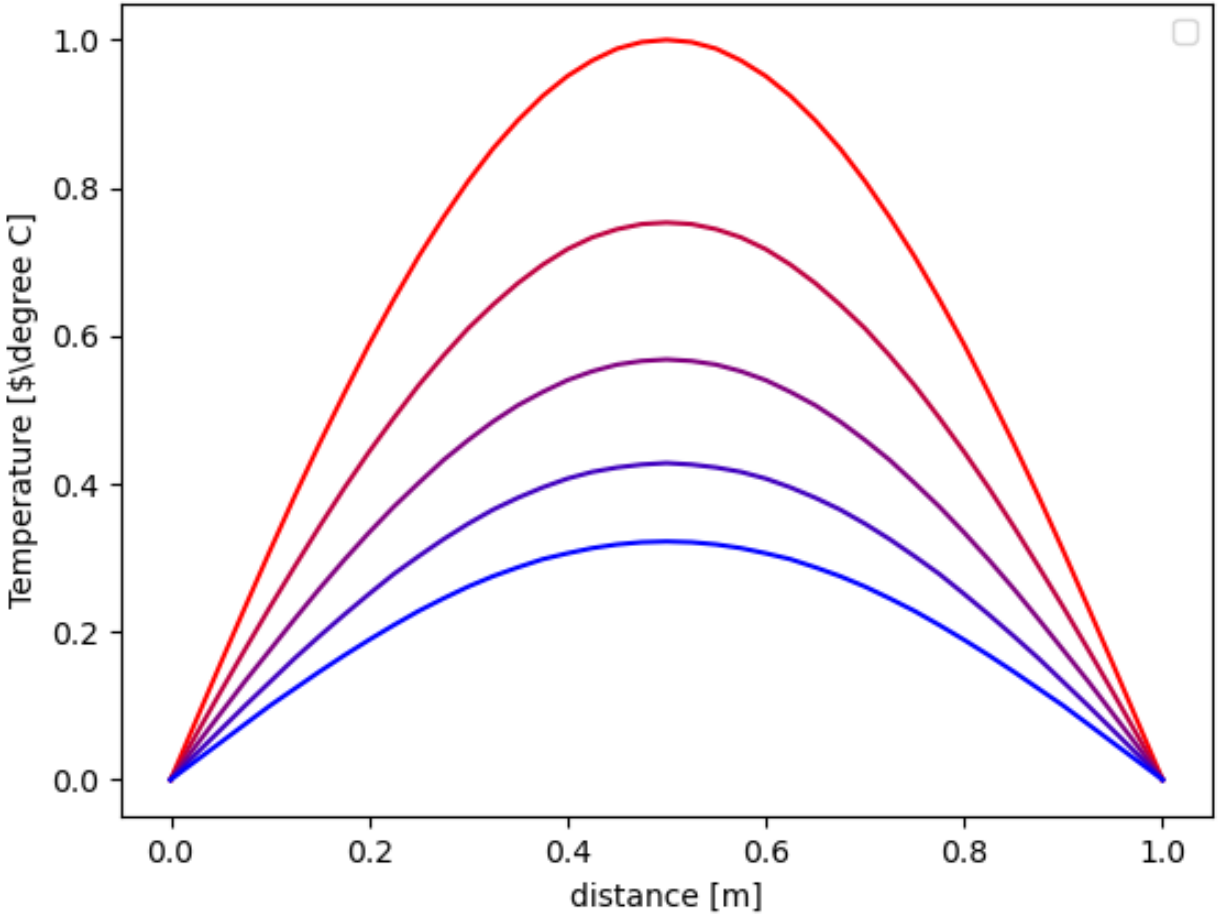
```
#Notre condition initial
T[:, 0] = ConditionInitial
#T.round(3)
Lamda = dt/dx**2


for j in range(1, m):
    for i in range(1, n-1):
        #codage de l'equation
        #T[i,0] = Lamda*T[i-1, j-1] + (1-2*Lamda)*T[i, j-1] + Lamda*T[i+1, j-1]
         T[i,j] = Lamda*T[i-1, j-1] + (1-2*Lamda)*T[i, j-1] + Lamda*T[i+1, j-1]
        #T[i,0] = Lamda*T[i-1,0] + (1-2*Lamda)*T[i, 0] + Lamda*T[i+1, 0]


#xi=centre(expl: xi=3), on lui dit fait le tour,
#Afin de tenir en compte chaque xi, en utilisant la fonction <<round()>>
T = T.round(3)
T


R = np.linspace(1,0,m)
B = np.linspace(0,1,m)
G = 0


```

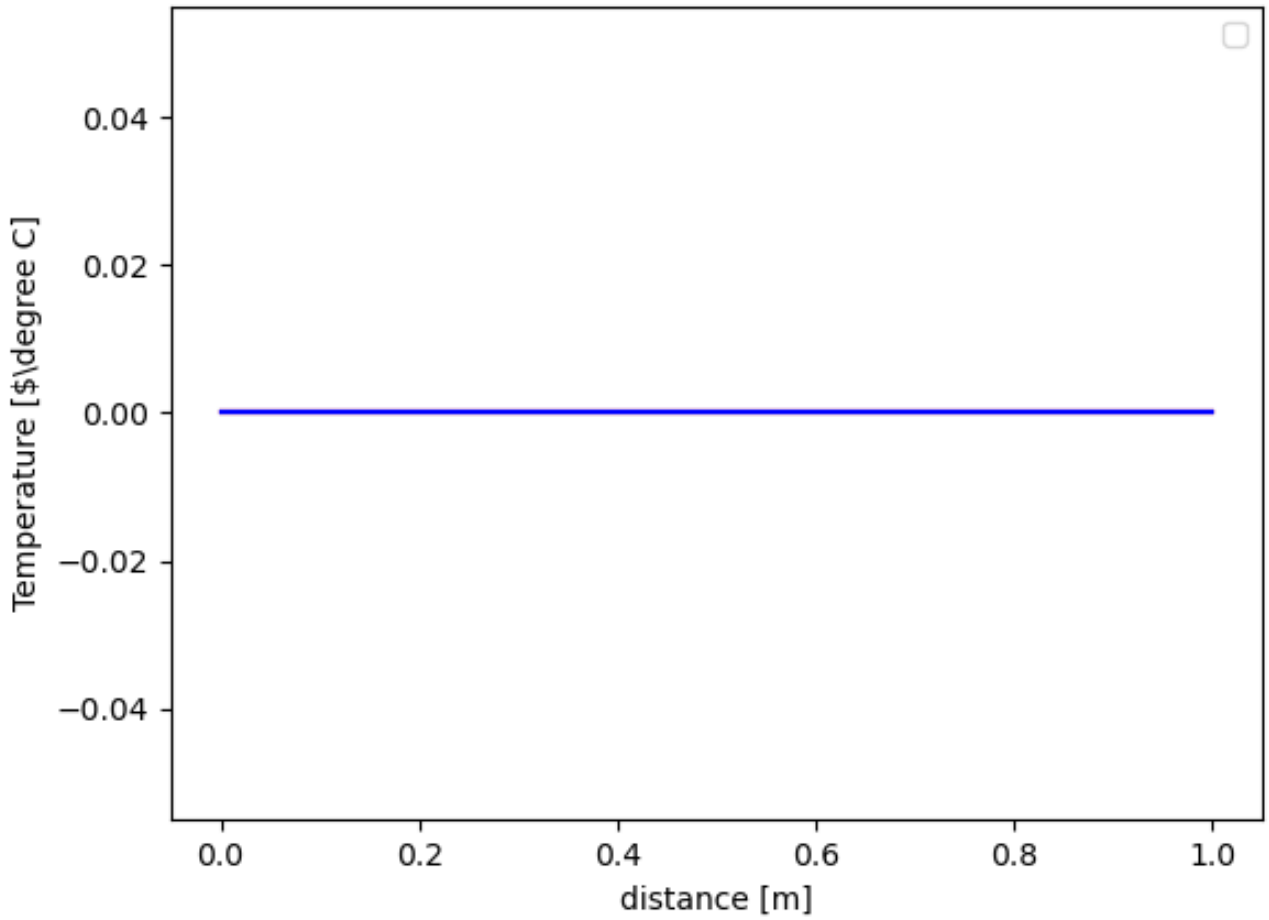
```
82
83  for j in range(m):
84      plt.plot(x, T[:, j], color = [R[j],G,B[j]]) # gestion d'Affich
85
86  plt.xlabel('distance [m]')
87  plt.ylabel('Temperature [$\degree C]')
88  plt.legend(f't = {value} s' for value in t)
89  plt.show() #rendre visible la figure
90
91                                  #~~~~~~~Fin_Projet~~~~~~~~~~#
```

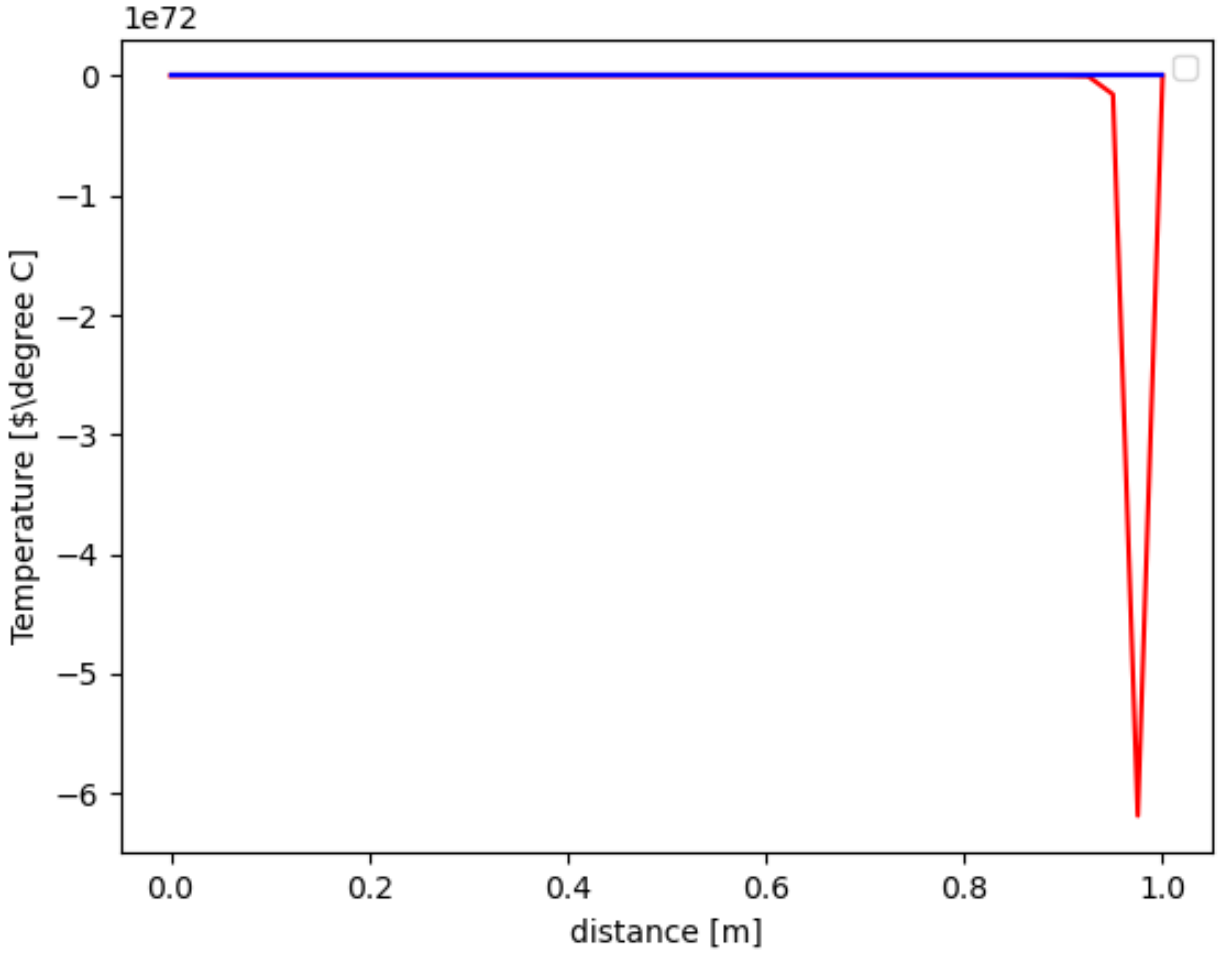
Indication pour execution

Dans cette etape pour obtenir chacune des graphiques, il faut commenter les 2 expressions et laisser l'une, ainsi de suite on obtient chaque graphe

Apres execution du programme




0.04
0.02
0.00
−0.02
−0.04
Temperature [$\degree C]
0.0
0.2
0.4
0.6
0.8
1.0
distance [m]


1e72
0
−1
−2
−3
−4
−5
−6
Temperature [$\degree C]
0.0
0.2
0.4
0.6
0.8
1.0
distance [m]