TD N°6 : Résolution des équations aux dérivées partielles

Ahmed Ammar (ahmed.ammar@fst.utm.tn)

Institut Préparatoire aux Études Scientifiques et Techniques, Université de Carthage.

Apr 22, 2020

Contents

Exercice 1: Équation de la chaleur 2D

On considère une plaque de fer carrée de côté L = 10 mm, de coefficient de diffusion D = 4 $mm^2.s^{-1}$.

L'équation de diffusion bidimensionnelle est

$$\frac{\partial U}{\partial t} = D\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}\right) \tag{1}$$

Considérons l'équation de diffusion appliquée à une plaque métallique carrée de côté L = 10 mm, de coefficient de diffusion D = 4 $mm^2.s^{-1}$. La plaque est initialement à la température $T_{froid} = 300\ K$ en dehors d'un disque (centré sur $x_c = 5$, y_c =5 et de rayon r = 2 mm) qui est à la température $T_{chaud} = 1000\ K$. Nous supposons que les bords de la plaque sont maintenus fixes à T_{froid}.

a) Rappeler la définition d'un schéma FTCS et donner l'approximation numérique de l'équation de chaleur 2D.

Solution. Le schéma FTCS (Forward Time Centred Space) est un schéma explicite d'ordre 1 en temps et d'ordre 2 en espace. Il s'agit de discrétiser les deux dérivées partielles en utilisant un développement de Taylor. Le membre de gauche peut donc s'écrire en approximant :

$$u_{i,j}^{(n+1)} = u_{i,j}^{(n)} + D\Delta t\left[\frac{u_{i+1,j}^{(n)} - 2u_{i,j}^{(n)} + u_{i-1,j}^{(n)}}{(\Delta x)^2} + \frac{u_{i,j+1}^{(n)} - 2u_{i,j}^{(n)} + u_{i,j-1}^{(n)}}{(\Delta y)^2}\right]. \tag{2}$$

b) Quelle est la valeur du temps maximum Δt que nous pouvons autoriser sans que le processus ne devienne instable?

Solution. On peut montrer que le pas de temps maximum, Δt que nous pouvons autoriser sans que le processus ne devienne instable est :

$$\Delta t = \frac{1}{2D} \frac{(\Delta x \Delta y)^2}{(\Delta x)^2 + (\Delta y)^2}. \tag{3}$$

c) Terminer lorsqu'il y a des points d'interrogation (???) dans le script Python `EquDiff2D.py` ci-dessous afin d'obtenir la sortie comme indiqué dans la figure 1.

```
## NOM DU PROGRAMME: EquDiff2D.py
#% IMPORTATION
import numpy as np
import matplotlib.pyplot as plt
# taille de la plaque, en [mm]
L = 10.
# intervalles dans les directions x, y, en [mm]
dx = dy = 0.1
# Coefficient de diffusion thermique de l'acier, en [mm2.s-1]
D = 4.
Tfroid, Tchaud = 300, 1000
# Nombre de pas de temps
nsteps = 250
nx, ny = int(L/dx), int(L/dy)
dx2, dy2 = dx*dx, dy*dy
# pas de temps maximum, dans question b)
dt = ???
u = Tfroid * np.ones((nx, ny))
# Conditions initiales - disque de rayon r, centrée sur (cx, cy), en [mm]
r, cx, cy = 2, 5, 5
r2 = r**2
for i in range(nx):
    for j in range(ny):
        p2 = ???
        if p2 < r2:
            u[i,j] = Tchaud
# Sortie de 4 graphiques à quatre instants dans l'intervalle de temps
mfig = [0, 50, 100, 200]
fignum = 0 # initialisation
fig = plt.figure()
# implémentation du schéma FTCS
for m in range(nsteps):
    for i in range(1, nx-1):
        for j in range(1, ny-1):
            uxx = ???
            uyy = ???
            u[i,j] += ???
    if m in mfig:
        fignum += 1
        print(m, fignum)
        ax = fig.add_subplot(220 + fignum)
        im = ax.imshow(u, cmap=plt.get_cmap('hot'), vmin=Tfroid,vmax=Tchaud)
        ax.set_axis_off()
        ax.set_title('{:.1f} ms'.format(m*dt*1000))
```

```
fig.subplots_adjust(right=0.85)
cbar_ax = fig.add_axes([0.9, 0.15, 0.03, 0.7])
cbar_ax.set_xlabel('T [K]', labelpad=20)
fig.colorbar(im, cax=cbar_ax)
plt.savefig("EquDiff2D.png"); plt.savefig("EquDiff2D.pdf")
plt.show()
```

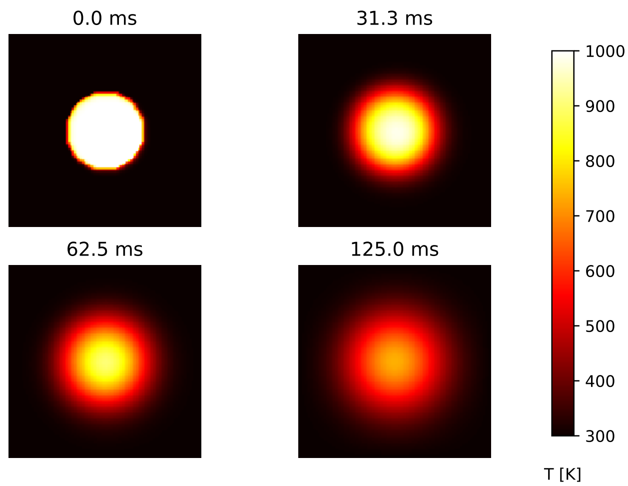


Figure 1:

Exercice 2: Condensateur plan

Résoudre l'équation de Laplace en deux dimensions pour le potentiel dans un condensateur plan

$$\frac{\partial^2 u(x,y)}{\partial x^2} + \frac{\partial^2 u(x,y)}{\partial y^2} = 0 \tag{4}$$

Soit un condensateur à plaques parallèles avec des potentiels V = ± 1 V contenus dans une région carrée mise à la terre de longueur latérale L comme indiqué sur la figure 2.

a) Rappeler la définition de la méthode Gauss-Seidel et donner l'approximation numérique de l'équation de Laplace 2D.

b) Adapter le script `Laplace_surrelax.py` étudié dans le cours afin d'avoir les résultats sur les figures 3 et 4.

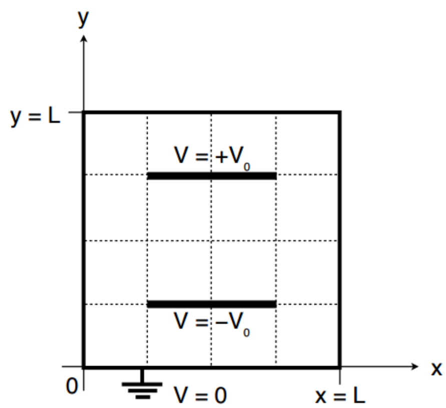


Figure 2:

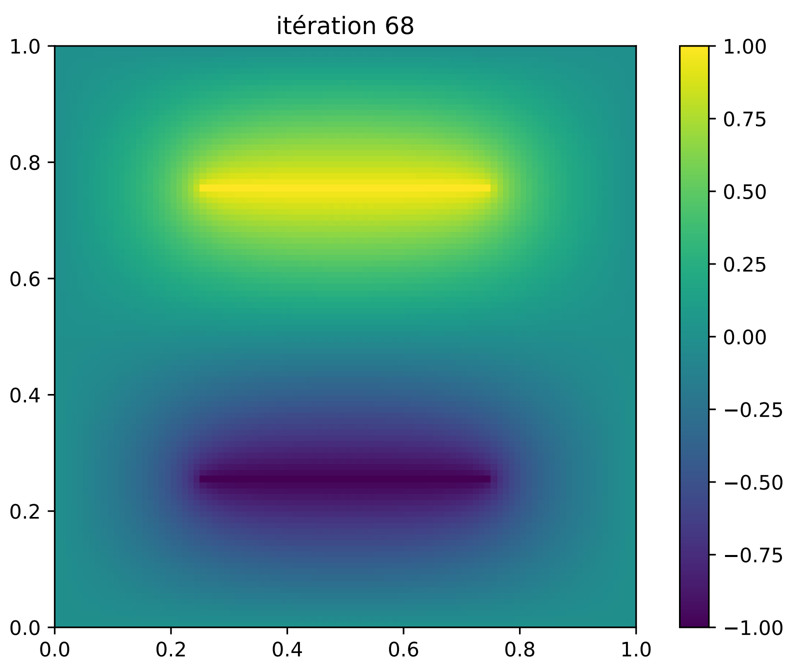
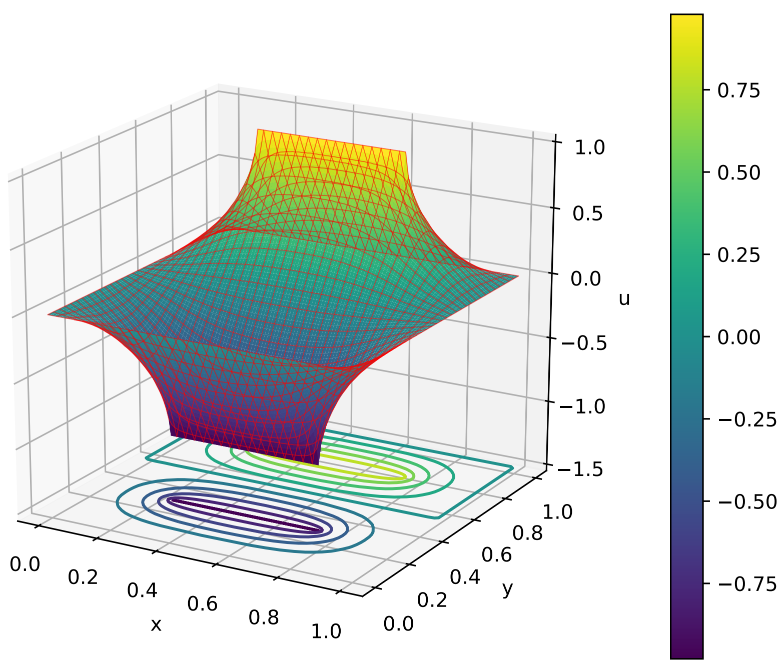


Figure 3:



0.0
0.2
0.4
0.6
0.8
1.0
x
0.0
0.2
0.4
0.6
0.8
1.0
y
1.0
0.5
0.0
−0.5
−1.0
−1.5
u
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

Figure 4: