

# Corrigé du TD Python : Résolution de l'équation de diffusion thermique par la méthode d'Euler explicite

## I. Problème 1D

```
import matplotlib.pyplot as plt
import numpy as np
NX=100
dx=1./ (NX)
x=np.arange(0., 1.0+dx, dx)
K=0.0012
tau=1/K

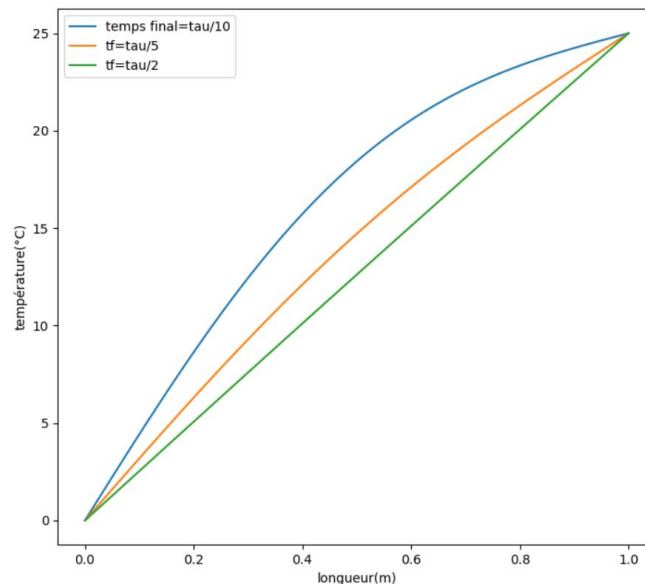
#Résolution
def résoudre (Nt, tmax):
    #Conditions initiales
    TT=np.zeros (NX+1)
    TT0=np.zeros (NX+1)
    #Conditions aux limites
    TT[0]=1
    TT0[0]=1
    b = 0
    t=0
    dt=1/Nt
    while t<tmax:
        t +=dt
        for j in range(1, NX):
            TT[j] = TT0[j] + dt * (TT0[j - 1] - 2 * TT0[j] + TT0[j + 1]) / (dx**2)
            if abs(TT[j])>1:
                b=1
                break
        if b != 0:
            print ('diverge au temps',t)
            break
        TT0=np.copy(TT)
    print ('Temps final',t)
    return TT0

#Tracé:
def tracé (Nt, tmax):
    plt.plot(x, 25* (-résoudre (Nt, tmax)+1))

    plt.xlabel('longueur (m)')
    plt.ylabel('température (°C)')
def affiche():
    plt.legend(['temps final=tau/10', 'tf=tau/5', 'tf=tau/2'])
    plt.show()

tracé(100000, 1/10)
tracé(100000, 1/5)
tracé(100000, 1/2)
#tracé(19900, 1/10)
#tracé(10000, 1/10)

affiche()
```



## II. Problème 2D

```

import matplotlib.pyplot as plt
import numpy as np

Nx=30
Ny=30

#Initialisation des tableaux
TT1=np.zeros((Nx,Ny))
TT=np.zeros((Nx,Ny))

#Conditions aux limites: températures autour
# i repère les lignes et j les colonnes...
for i in range(0,Nx):
    TT[i,0]=1
    TT[i,Ny-1]=0
    TT[0,i]=1
    TT[Nx-1,i]=0
    TT1[i,0]=1
    TT1[0,i]=1
tt=0
dt=1/100000
K=0.00012
dx=1/Nx

#Itérations
while tt<1/10:
    tt +=dt
    for i in range(1,Nx-1):
        for j in range(1, Ny - 1):
            TT1[i,j] = TT[i,j] + dt* (TT[i - 1,j] - 2 * TT[i,j] + TT[i + 1,j] + TT[i,j-1]
            |- 2 * TT[i,j] + TT[i,j+1]) / (dx**2)

    TT=np.copy(TT1)
T=25*(1-TT)

#Tracé du tableau
plt.imshow(T)
plt.colorbar()
plt.title('Profil de température')
plt.savefig("EquChaleur2D.jpeg")

plt.show()

```

