Atividade da unidade 2

Equação do Calor 2D

$\frac{\partial \phi}{\partial t} = k(\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}) + f(x, y, t)$

Das diferenças finitas + método de Euler explícito, com $\Delta x = \Delta y = h$:

$\phi_{ij}^{n+1} = \phi_{ij}^{n} + k\frac{\Delta t}{h^2}(\phi_{i-1,j}^{n} + \phi_{i,j-1}^{n} - 4\phi_{i,j}^{n} + \phi_{i+1,j}^{n} + \phi_{i,j+1}^{n}) + \Delta t f(x, y, t)$

Das diferenças finitas + método de Euler implícito, com $\Delta x = \Delta y = h$:

$\phi_{ij}^{n+1} - k\frac{\Delta t}{h^2}(\phi_{i-1,j}^{n+1} + \phi_{i,j-1}^{n+1} - 4\phi_{i,j}^{n+1} + \phi_{i+1,j}^{n+1} + \phi_{i,j+1}^{n+1}) = \phi_{ij}^{n+1} + \Delta t f(x, y, t)$

Tratamento para cond. fronteira.

Cond. de contorno vem do problema.

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.sparse import diags, csr_matrix
from scipy.sparse.linalg import spsolve

def phi(x,y,t):
    return (np.exp(-t))*np.sin(2*np.pi*x)*np.sin(2*np.pi*y)

def fu(x,y,t,k):
    return ((8*k*np.pi**2 - 1)*phi(x,y,t))

def fronteira(Np1,phiold,x,y,t,a,b):
    #células fantasmas
    N = Np1 - 2
    for i in range(1,N+1):
        phiold[i][0] = 2*phi(x[i],a,t) - phiold[i,1]
        phiold[i,N+1] = 2*phi(x[i],b,t) - phiold[i,N]
    for j in range(1,N+1):
        phiold[0][j] = 2*phi(a,y[j],t) - phiold[1,j]
        phiold[N+1,j] = 2*phi(b,y[j],t) - phiold[N,j]
    return phiold

def norma(m,N):
    maximo = 0
    im = 0
    jm = 0
    for i in range(N):
        for j in range(N):
            if(maximo < abs(m[i][j])):
                maximo = abs(m[i][j])
                im = i
                jm = j
    return (maximo,im,jm)

def main():
    N = 8
    a = 0
    b = 1
    k = 1
```

```
    dx = (b-a)/N
    dy = (b-a)/N
    h = dx

    t = 0
    tf = 0.25
    dt = 0.25*dx**2
    dt0 = dt

    c = k*(dt0/(h**2))


    Np1 = N + 2
    phiex = np.zeros((Np1,Np1))
    phiaprox = np.zeros((Np1,Np1))
    phiold = np.zeros((Np1,Np1))
    f = np.zeros((Np1,Np1))

    x = np.zeros(Np1)
    y= np.zeros(Np1)
    for i in range(Np1):
        x[i] = a + (i+1)*dx/2
        y[i] = a + (i+1)*dx/2

    for i in range(Np1):
        for j in range(Np1):
            phiold[i][j] = phi(x[i],y[j],0)


    fronteira(Np1,phiold,x,y,t,a,b)

    while(t<tf+dt/2):
        for i in range(1,N+1):
            for j in range(1,N+1):
                phiex[i][j] = phi(x[i],y[j],t+dt)
                phiaprox[i][j] = phiold[i][j] + c*(phiold[i-1][j] + phiold[i][j-1] -4*phic
        dt = min(dt, tf-t)
        t = t+dt
        c = k*(dt/(h**2))
        phiaprox = fronteira(Np1,phiaprox,x,y,t,a,b)
        phiold = phiaprox
        #print(phiaprox.max())

    #print(abs(phiex-phiaprox))
    err = np.zeros((Np1,Np1))
    for i in range(1,N+1):
            for j in range(1,N+1):
                err[i][j] = abs(phiex[i][j] - phiaprox[i][j])
    #print("N =", N, "| err ->", np.linalg.norm(err, ord = np.inf,axis=1))
    #print(err.max())
    print(norma(err,Np1))

main()
```

```
(1.7822137936934959, 4, 4)
```

In [ ]: