Equação do transporte

$$\frac{\partial \phi}{\partial t} + \frac{\partial(\rho u \phi)}{\partial x} + \frac{\partial(\rho v \phi)}{\partial y} = \frac{\partial}{\partial x}(k\frac{\partial \phi}{\partial x}) + \frac{\partial}{\partial y}(k\frac{\partial \phi}{\partial y}) + f(x,y)$$

Alterando da eq. do calor (trabalho atividade 2)

$$\rho u = \rho v = k = 1$$

$$f = -\frac{\partial^2 \phi}{\partial x^2} - \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial \phi}{\partial x} + \frac{\partial \phi}{\partial y} + \frac{\partial \phi}{\partial t}$$

Alterar a f

$$f(x,y,t) = e^{-t}[(8\pi^2 - 1)sin(2\pi x)sin(2\pi y) + 2\pi cos(2\pi x)sin(2\pi y) + 2\pi sin(2\pi x)sin(2\pi y)]$$

Das diferenças finitas + Euler explícito, e fazendo $\Delta x = \Delta y \equiv h$

$$\frac{\phi^{n+1}_{i,j} - \phi^n_{i,j}}{\Delta t} + \frac{\phi^n_{i+1,j} - \phi^n_{i-1,j}}{2\Delta x} + \frac{\phi^n_{i,j+1} - \phi^n_{i,j-1}}{2\Delta y} = \frac{\phi^n_{i+1,j} - 2\phi^n_{i,j} + \phi^n_{i-1,j}}{\Delta x^2} + \frac{\phi^n_{i,j+1} - 2\phi^n_{i,j} + \phi^n_{i,j-1}}{\Delta y^2} + f(x,y,t)$$

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

adicionar as derivadas extras no cálculo da phi aprox

In [5]:

```python
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):
    return ((8*np.pi**2 - 1)*phi(x,y,t)) + 2*np.pi*(np.exp(-t))*np.cos(2*np.pi*x)*np.sin(2

def fronteira(Np2,phiold,x,y,t,a,b):
    #células fantasmas
    N = Np2 - 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 = 0.125*dx**2
    dt0 = dt

    c1 = dt0/(2*h)
    c2 = dt0/(h**2)

    Np2 = N + 2
    phiex = np.zeros((Np2,Np2))
    phiaprox = np.zeros((Np2,Np2))
    phiold = np.zeros((Np2,Np2))
    f = np.zeros((Np2,Np2))

    x = np.zeros(Np2)
    y= np.zeros(Np2)
    for i in range(Np2):
        x[i] = a + (i+0.5)*dx
        y[i] = a + (i+0.5)*dx

    for i in range(Np2):
        for j in range(Np2):
            phiold[i][j] = phi(x[i],y[j],0)


    fronteira(Np2,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] -c1*(phiold[i+1][j]-phiold[i-1][j]+phiold[i]
        dt = min(dt, tf-t)
        t = t+dt
        c = k*(dt/(h**2))
        phiaprox = fronteira(Np2,phiaprox,x,y,t,a,b)
        phiold = phiaprox
        #print(phiaprox.max())

    #print(abs(phiex-phiaprox))
    err = np.zeros((Np2,Np2))
    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(err.max())
    print(norma(err,Np2))

main()
```

```
(0.4254473986710457, 1, 1)
```

erro para eq transporte 2d

N = 8,e = 0.4254473986710457

N = 16, e = 0.2601869543902819

N = 32, e = 0.1463291654053212

In [4]:

```
#Convergência da sequência
print(0.4254473986710457/0.2601869543902819)
print(0.2601869543902819/0.1463291654053212)
```

```
1.635160377921455
1.7780936129143008
```

In [ ]: