Exercice7_SDZ

November 26, 2021

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

Problème 7. Discretisez l'équation

$$-\Delta u = f(x,y) \qquad \text{sur } \Omega = (0,1)\times(0,1)$$

et $u = 0$ au bord $\partial\Omega$ par des différences finies (stencil à 5 points) et un maillage isotrope avec une constante taille de maille $h = \frac{1}{3}$.
Solution: la solution analytique est $u(x,y) = x\,(1-x)\,y\,(1-y)\,e^{x+y}$, *et la solution approchée est* $\vec{u} = (0.0900, 0.1266, 0.1266, 0.1788)^T$.

Comme f n'est pas donné, on doit le retrouver à partir de la solution

$$\frac{\partial u}{\partial x} = \left(y - xy - y^2 + xy^2 - x^2y + y^2y^2\right)e^{x+y}$$

$$\frac{\partial u}{\partial y} = \left(x - x^2 - xy + x^2y - xy^2 + x^2y^2\right)e^{x+y}$$

$$\frac{\partial^2 u}{\partial x^2} = (-3xy + 3xy^2 - x^2y + x^2y^2)e^{x+y}$$

$$\frac{\partial^2 u}{\partial y^2} = (-3xy + 3x^2y - xy^2 + x^2y^2)e^{x+y}$$

$$f(x,y) = -\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = -e^{x+y}\left(-6xy + xy^2 + 2x^2y + 2x^2y^2\right)$$

On discrétise u'' en fonction de x et de y

$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = -\left(\frac{u(x-h,y) - 2u(x,y) + u(x+h,y)}{h^2} + \frac{u(x,y-h) - 2u(x,y) + u(x,y+h)}{h^2}\right)$$

Qui est égal à $f(x,y)$. On peut donc écrire la matrice A

$$\frac{1}{h^2}\begin{pmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{pmatrix}\begin{pmatrix} u_{11} \\ u_{12} \\ u_{21} \\ u_{22} \end{pmatrix} = \begin{pmatrix} f_{11} \\ f_{12} \\ f_{21} \\ f_{22} \end{pmatrix} + \begin{pmatrix} 0+\cdots \\ 0+\cdots \\ 0+\cdots \\ 0+\cdots \end{pmatrix}$$

On a des termes 0 car tous les bords sont égaux à 0 Si on effectue la résolution sur Python, on trouve bien les valeurs données

[27]:
```
def f(x,y):
    return -np.exp(x+y)*(-6*x*y+2*x*y**2+2*x**2*y+2*x**2*y**2)
h = 1/3
N = 4
A = 1/h**2*np.matrix('[4 -1 -1 0;-1 4 0 -1;-1 0 4 -1; 0 -1 -1 4')

positions = h*np.array([(1,1), (1,2), (2,1), (2,2)])
u = np.zeros([N,1])
F = np.zeros_like(u)
for i, p in enumerate(positions):
    F[i] = f(p[0],p[1])

u = A.I @ F
print(u)
```

```
[[0.09002563]
 [0.12661549]
 [0.12661549]
 [0.17779429]]
```