Computer Methods in Enigineering
Exercise on elliptic equations

In this exercise we will work on solutions to Elliptic equations, more specifically, the Laplace equation $\nabla^2 f = 0$. In our case, we will consider flow in porous media. Such flow is governed by the Darcy equation:

$$\vec{q} = -\frac{k}{\mu}\nabla p$$

Here q is the volumetric flow rate, k is the permeability (a measure for how well the porous medium allows for transport of fluids), μ is the viscosity of the fluid, and p is the fluid pressure. As shown in the lecture, at steady state this gives the Laplace equation $\nabla^2 p = 0$. We will consider a two-dimensional model, thus

$$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$$

The solution of the Laplace equations can be obtained by different methods. In this exercise we will use two solution techniques. Both will be based on the finite difference approximation. The centered difference for the second order derivative is given as

$$\frac{\partial^2 p}{\partial x^2} \simeq \frac{p(x+\Delta x, y) - 2p(x,y) + p(x-\Delta x, y)}{(\Delta x)^2}$$

Equivalently, in the y-direction, we have

$$\frac{\partial^2 p}{\partial y^2} \simeq \frac{p(x, y+\Delta y) - 2p(x,y) + p(x, y-\Delta y)}{(\Delta y)^2}$$

For the two-dimensional system, our Laplace equation $\nabla^2 p = 0$ is given as,

$$\nabla^2 p = 0 \tag{1}$$

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) p = 0 \tag{2}$$

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = 0 \tag{3}$$

These two second order derivatives can then be approximated using finite differences as

$$P_{i+1,j} - 2P_{i,j} + P_{i-1,j} + P_{i,j+1} - 2P_{i,j} + P_{i,j-1} = 0 \tag{4}$$

$$P_{i+1,j} + P_{i,j+1} - 4P_{i,j} + P_{i-1,j} + P_{i,j-1} = 0 \tag{5}$$

where we have used the index notation $P_{i,j} = P(i\Delta x, j\Delta y)$.

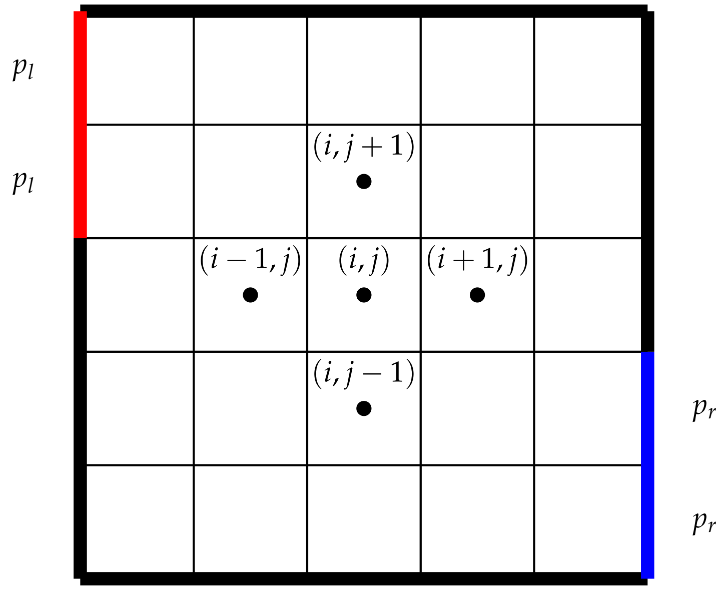


Figure 1: Grid representation of the sand-body connecting the two reservoirs. Thick black lines indicate no-flow boundaries, while red lines indicate boundaries connected to the left reservoir, and blue lines indicate boundaries connected to the right reservoir.

Assume a sand-body connecting two fluid reservoirs at different pressure. The left reservoir has a pressure $p_l = 1 \times 10^5\,\mathrm{Pa}$, while the right reservoir has a pressure $p_r = 2 \times 10^5\,\mathrm{Pa}$. The sand-body has a shape between the two reservoirs as outlined in Fig. 1, where the grid cell size is $100\,\mathrm{m} \times 100\,\mathrm{m}$. Further, assume a viscosity of $1 \times 10^{-3}\,\mathrm{Pa\,s}$, a permeability of $1 \times 10^{-10}\,\mathrm{m}^2$, and assume a sand body thickness of $10\,\mathrm{m}$.

Problem 1

Assume the porous medium is filled with water, and fully connected to the left and right boundary at the indicated boundaries. To simplify, we can assume that the porous medium stretches a half grid block into the two reservoirs where the connections are indicated.

a) Set up the linear equations describing this system.

Solution:

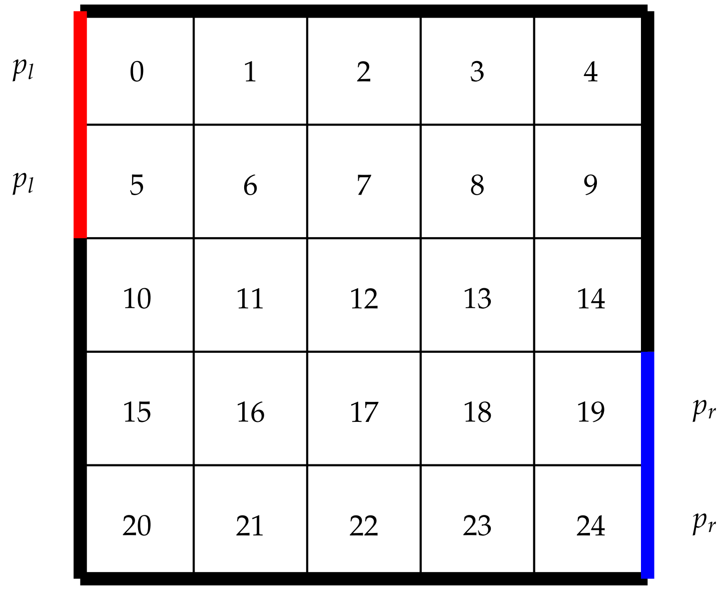


Figure 2: Ordering of grid cells.

If we order the grid cells as shown in Fig. 2, then we have the following set of equations.

$$P_1 + P_5 - 3P_0 = -p_l \tag{6}$$
$$P_{i-1} + P_{i+1} + P_{i+5} - 3P_i = 0, \quad i \in \{1,2,3\} \tag{7}$$
$$P_3 + P_9 - 2P_4 = 0 \tag{8}$$
$$P_6 + P_0 + P_{10} - 4P_5 = -p_l \tag{9}$$
$$P_{i-1} + P_{i+1} + P_{i-5} + P_{i+5} - 4P_i = 0, \quad i \in \{6,7,8,11,12,13,16,17,18\} \tag{10}$$
$$P_8 + P_4 + P_{14} - 3P_9 = 0 \tag{11}$$
$$P_{11} + P_5 + P_{15} - 3P_{10} = 0 \tag{12}$$
$$P_{13} + P_9 + P_{19} - 3P_{14} = 0 \tag{13}$$
$$P_{16} + P_{10} + P_{20} - 3P_{15} = 0 \tag{14}$$
$$P_{18} + P_{14} + P_{24} - 4P_{19} = -p_r \tag{15}$$
$$P_{21} + P_{15} - 2P_{20} = 0 \tag{16}$$
$$P_{i-1} + P_{i+1} + P_{i-5} - 3P_i = 0, \quad i \in \{21,22,23\} \tag{17}$$
$$P_{23} + P_{19} - 3P_{24} = -p_r \tag{18}$$

b) We want to solve the system of equations using matrix form, $\mathbf{A}\vec{p} = \vec{b}$. Set up the matrix $\mathbf{A}$ and vector $\vec{b}$ in Python.

Solution:

```python
import numpy as np
import matplotlib.pyplot as plt

#Grid size
innSide=5
#innCells=(innSide,innSide)
```

```
#Boundary conditions
pl=1E5
pr=2E5

A=np.zeros((innSide**2,innSide**2))
b=np.zeros(innSide**2)

#Set up matrix A for all internal cells
for jj in range(0,innSide):
  for ii in range(0,innSide):
    cellNum=ii+innSide*jj
    if ii>0:
      A[cellNum,cellNum]-=1
      A[cellNum,cellNum-1]+=1
    if ii<innSide-1:
      A[cellNum,cellNum]-=1
      A[cellNum,cellNum+1]+=1
    if jj>0:
      A[cellNum,cellNum]-=1
      A[cellNum,ii+innSide*(jj-1)]+=1
    if jj<innSide-1:
      A[cellNum,cellNum]-=1
      A[cellNum,ii+innSide*(jj+1)]+=1

#Add boundaries to matrix A
#and to vector b
A[0,0]-=1
b[0]=-pl
A[5,5]-=1
b[5]=-pl
A[19,19]-=1
b[19]=-pr
A[24,24]-=1
b[24]=-pr

print(A)
print(b)
```

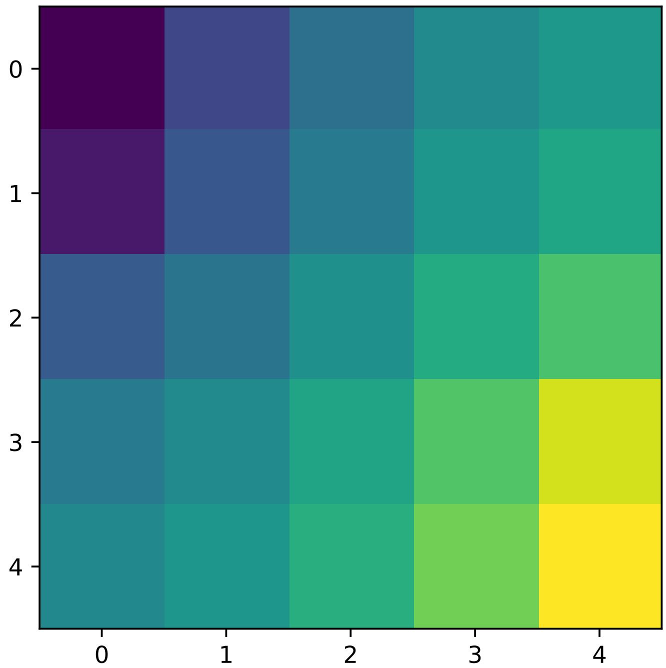
c) Solve for the pressure field $\vec{p}$, and plot the resulting pressure field.

Solution:

```
#Calculate the pressure field
p=np.dot(np.linalg.inv(A),b)

#Plot the pressure field
plt.imshow(p.reshape(innSide,innSide), interpolation='none')
plt.savefig("pressureImplicit.pdf")
```

Running this code gives the output shown in the figure below.

d) Solve for the pressure using the Gauss-Seidel relaxation method, with relaxation parameter $\alpha = 1$ and using 10 iterations. Plot the resulting pressure field.

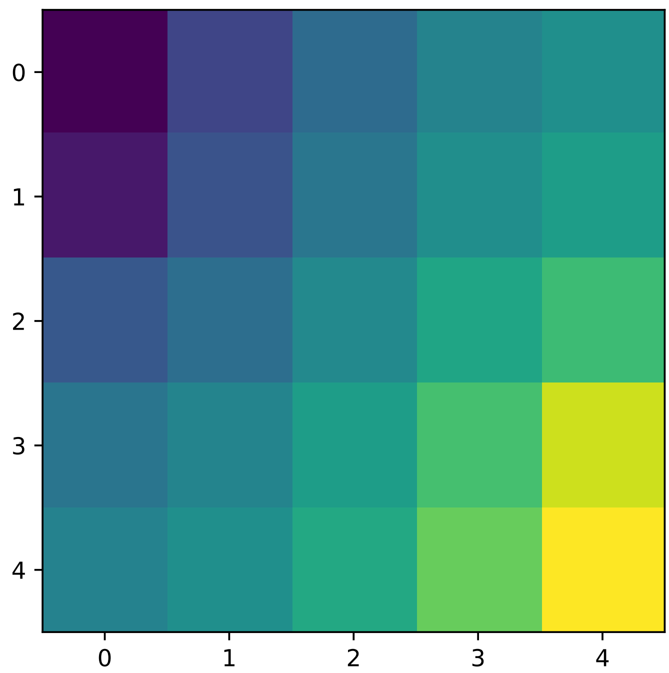


Solution:

```
def solveIterative(innSide,alpha,pl,pr,innIte):
  #Create an extended matrix, including boundaries
  E=np.zeros((innSide+2,innSide+2))
  E[:]=(pl+pr)/2

  #Add boundary conditions
  E[1,0]=pl
  E[2,0]=pl
  E[-2,-1]=pr
  E[-3,-1]=pr

  for nnit in range(0,innIte):
    for jj in range(1,innSide+1):
      for ii in range(1,innSide+1):
        E[ii,jj]=(1.0-alpha)*E[ii,jj]+alpha*(1/4)*(E[ii-1,jj]+E[ii+1,jj]+E[ii,jj-1]+E[ii,jj+1])
    #Update boundaries
    E[0,1:-1]=E[1,1:-1]
    E[-1,1:-1]=E[-2,1:-1]
    E[3:-1,0]=E[3:-1,1]
    E[1:-3,-1]=E[1:-3,-2]
  return E[1:-1,1:-1]


#Set relaxation parameter
alpha=1.0
#Set number of iterations
```

```
innIte=10

E=solveIterative(innSide,alpha,pl,pr,innIte)
#Plot the pressure field
plt.imshow(E, interpolation='none')
plt.savefig("pressureRelaxAlpha1.pdf")
```

Running this code gives the output shown in the figure below.



e) Compare the solution with the implicit method above. Assuming the implicit method gives the correct answer, what is the error when using 10 and 100 iterations.

Solution:

Total error for 10 iterations: 50501.676877330945
Max cell error for 10 iterations: 2882.636651455774
Total error for 100 iterations: 1626.005155947947
Max cell error for 100 iterations: 82.16249600527226

f) Use the Gauss-Seidel relaxation method with relaxation parameter $\alpha = 1.8$. How does the change in α value affect the error?

Solution:

Total error for 100 iterations: 0.007593311398522928
Max cell error for 100 iterations: 0.00041365617653355

g) Calculate the fluid flow from the right to the left reservoir.

Solution:

The fluid flow is given by the Darcy equation. We can calculate the fluid flow over any cross-section of the reservoir. E.g., we can choose the blue lines to calculate the fluid flow.

This is done by finding the two pressure drops over the blue line, Δp_1 and Δp_2, and then calculating

$$Q = Q_1 + Q_2 = lh(q_1 + q_2) = lh\frac{k}{\mu}(\Delta p_1 + \Delta p_2)$$

This can be done in Python as

```
l=100
h=10
k=1E-10
mu=1E-3

Q=l*h*k/mu*(pr-p[19]+pr-p[24])

print('Total flow rate: ',Q)
```

Resulting in a rate of $4\,\mathrm{m^3/s}$.