Chapter 9

Partial Differential Equations

Soon-Hyung Yook

May 16, 2017

Table of Contents I

Laplacian Operator

Extending the relaxation method to 2-dimensional case is straightforward. Consider the partial differential equation:

$$\nabla^2 f = g(x,y) \tag{1}$$

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = g(x,y) \tag{2}$$

Second derivative of f in the x-direction

$$\frac{\partial^2 f}{\partial x^2} = \frac{f(x+h,y) + f(x-h,y) - 2f(x,y)}{h^2} \tag{3}$$

Similarly,

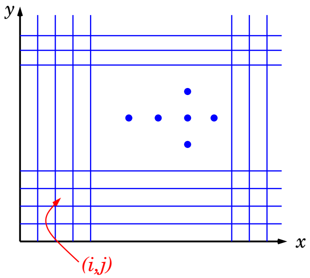
$$\frac{\partial^2 f}{\partial y^2} = \frac{f(x,y+h) + f(x,y-h) - 2f(x,y)}{h^2} \tag{4}$$

Laplacian Operator

Combine Eqs. (3) and (4) together, then the Laplacian operator is written as:

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \frac{f(x+h,y) + f(x-h,y) + f(x,y+h) + f(x,y-h) - 4f(x,y)}{h^2} \tag{5}$$

- Divide the given region into very small area.
- Label each gird with a pair of consecutive integer (i, j).



Relaxation Method in 2-Dimensional Space

Then we can re-express Eq. (1) as:

$$\frac{f(i+1,j)-2f(i,j)+f(i-1,j)}{h^2}+\frac{f(i,j+1)-2f(i,j)+f(i,j-1)}{h^2}=g(i,j)$$

Relaxation Method in 2D (Jacobi Method)

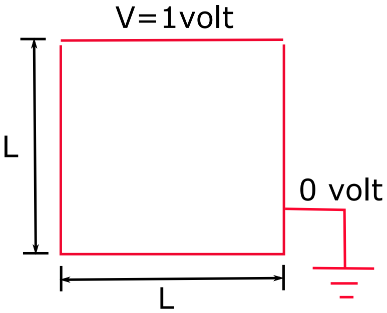
$$f^{(n+1)}(i,j)=\frac{f^{(n)}(i+1,j)+f^{(n)}(i-1,j)+f^{(n)}(i,j+1)+f^{(n)}(i,j-1)-h^2g(i,j)}{4}$$

Example:9.1 I

Solve the Laplace equation which corresponds to the case $g(x, y) = 0$ in Eq. (1).

$$\nabla^2 \phi(x, y) = 0, \tag{6}$$

with b.c. $V = 1$volt, and grid spacing $h = 1$cm and $L = 1$m.



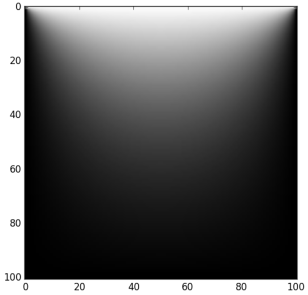
Example:9.1 II

```
from numpy import empty,zeros,max,copy
from pylab import imshow,gray,show

# Constants
M=100   # Grid squares on a side
V=1.0
tolerance=1e-6

# Creat arrays to hold potential

phi=zeros([M+1,M+1],float)
phi[0,:]=V
phitmp=empty([M+1,M+1],float)

# Main loop
delta=1.0
while delta>tolerance:
  # Calculate new values of the potential
  for i in range(M+1):
    for j in range(M+1):
      if i==0 or i==M or j==0 or j==M:
        phitmp[i,j]=phi[i,j]
      else:
        phitmp[i,j]=(phi[i-1,j]+phi[i+1,j] \
                                  +phi[i,j-1]+phi[i,j+1])/4
```

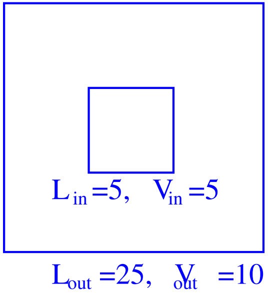
Example:9.1 III

```
        # Calculate maximum diffference from old values
    delta=max(abs(phi-phitmp))

    phi,phitmp=phitmp,phi
#   phi=copy(phitmp)


imshow(phi)
gray()
show()
```

Example:9.1 IV



Homework



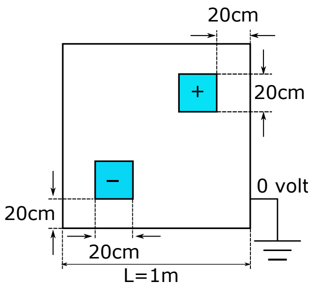
Use a relaxation method to compute the potential distribution between the two concentric square cylinders shown in the figure. The potential and the length of the square are given in the figure. Make a density plot of the electric potential.

Poisson Equation I

Solve the Poisson equation

$$\nabla^2 \phi = -\frac{\rho}{\epsilon_0} \tag{7}$$

with the boundary conditions depicted in the figure below. The two charges are each 20cm on a side and 20cm from the walls of the box and have charge density $\pm 1\mathrm{Cm}^{-2}$.



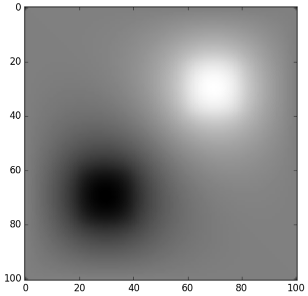
Poisson Equation II

```
from numpy import empty,zeros,max,copy
from pylab import imshow,gray,show

# Constants
M=100   # Grid squares on a side
V=1.0
h=0.01
tolerance=1e-6

# Creat arrays to hold potential

phi=zeros([M+1,M+1],float)
phitmp=empty([M+1,M+1],float)
rho=zeros([M+1,M+1],float)

# Initialize rho
for i in range(60,80):
  for j in range(20,40):
    rho[i,j]=1.0
    rho[j,i]=-1.0


# Main loop
delta=1.0
while delta>tolerance:
```

Poisson Equation III

```
  # Calculate new values of the potential
  for i in range(M+1):
    for j in range(M+1):
      if i==0 or i==M or j==0 or j==M:
        phitmp[i,j]=phi[i,j]
      else:
        phitmp[i,j]=(phi[i-1,j]+phi[i+1,j] \
                                    +phi[i,j-1]+phi[i,j+1]-rho[i,j]*h**2)/4

        # Calculate maximum diffference from old values
  delta=max(abs(phi-phitmp))

  phi,phitmp=phitmp,phi
#  phi=copy(phitmp)


imshow(phi)
gray()
show()
```

Poisson Equation IV



Successive Over Relaxation Method in 2D

$$f^{(n+1)}(i,j) = w\,\frac{f^{(n)}(i+1,j) + f^{(n)}(i-1,j) + f^{(n)}(i,j+1) + f^{(n)}(i,j-1) - h^2 g(i,j)}{4} + (1-w)f^{(n)}(i,j)$$