EE2703: Applied Programming Lab
Assignment 5
The Resistor Problem: Solving Laplace equation using Finite Difference Methods

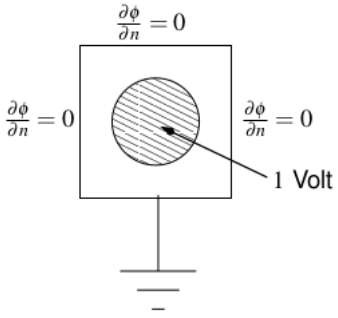
Arjun Menon Vadakkeveedu
EE18B104
Electrical Engineering, IIT Madras

March 4, 2020

Contents

1 Introduction

The goal is to obtain the potential distribution and the current density profile of a rectangular metallic plate with a central circular portion of the plate maintained at a potential V_o and an edge of the plate connected to GND.



To solve this, we resort to Poisson's equation (which is obtained from Gauss' Law and Conductivity relation):

$$\nabla^2\phi = \frac{1}{\sigma}\frac{\partial\rho}{\partial t}$$

Which in case of a constant potential maintained in the central region becomes the Laplace Equation:

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

$$\implies \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = 0 \tag{2}$$

A simple (but not time-efficient method as will be evident in the subsequent sections) method of solving Laplace's equation is the Finite Difference Method. In this method, the differential equation for ϕ_x and ϕ_y is converted into a difference equation by considering the Taylor series expansion of ϕ up to order 2.

---

2 Approach

Finite Difference Method: Taking the first order Taylor approximation of ϕ for solving Equation (2), we obtain:

$$\frac{\phi(i+1,j) - 2\phi(i,j) + \phi(i-1,j)}{(\delta x)^2} + \frac{\phi(i,j+1) - 2\phi(i,j) + \phi(i,j+1)}{(\delta y)^2} = 0$$

Taking equal stepsizes in x and y, i.e.: $\delta x = \delta y$, the expression for $\phi(i,j)$ becomes a simple average of neighbours:

$$\phi(i,j) = \frac{\phi(i+1,j) + \phi(i-1,j) + \phi(i,j+1) + \phi(i,j-1)}{4} \tag{3}$$

The approach used in this program is to assert the above condition repeatedly for a fixed number of iterations. As the number of iterations increase, the error between consecutive iterations of ϕ reduce. One may stop the iteration once the error between consecutive iterations go below a particular threshold. In addition to (3), one must also apply the following constraints:

1. At bottom, left and bottom edges of the plate, the normal component of the Electric Field is 0 (as a result of Gauss' law) which implies that the gradient of ϕ is 0.
   This reduces to $\phi(i,0) = \phi(i,1)$ for the left edge (and similarly for the other edges)

2. Since the top edge of the plate is maintained at 0V, the row corresponding to the top edge in the ϕ matrix must be identically 0. As the matrix is initialised as a 0-matrix, the row corresponding to x = 0.5cm in phi need not be manipulated.

3. The central circular region of the plate is constantly maintained at V_o volts. This must be asserted after each iteration.

---

3 Code Structure

3.1 Important blocks used in the program

1: Updating ϕ iteratively:

```
errors = zeros(Niter)
for k in range(Niter):
 oldphi = phi.copy()
 phi[1:-1, 1:-1] = 0.25*(phi[1:-1, 0:-2] + phi[1:-1, 2:] +
                   phi[0:-2, 1:-1] + phi[2:, 1:-1])
 #Poisson Update
 phi[1:-1, 0] = phi[1:-1, 1]
 # Boundary Condition for Left Surface
 phi[1:-1, -1] = phi[1:-1, -2]
 # Boundary Condition for Right Surface
```

```
 phi[0, 1:-1] = phi[1, 1:-1]
 # Boundary Condition for Top Surface
 phi[ii] = 1.0
 errors[k] = (abs(phi - oldphi)).max()
```

The matrix phi is initialised to a zero matrix of shape [Nx, Ny] where Nx and Ny are the user-input sizes along x and y respectively.

2: Fitting error in phi to an exponential model:

```
def fit_err(k, slope, intercept):
 return k*slope + intercept
popt1, __ = curve_fit(fit_err, range(Niter), log(errors))
popt2, __ = curve_fit(fit_err, range(500, Niter), log(errors[500:]))
true_err = (popt2[1]/popt2[0])*exp(popt2[0]*(Niter+0.5))
# error in log scale
print("Maximum bound for error = ", true_err)
print("Last iteration value of error = ", errors[-1])
```

3.2 Input and Output formats of the program

Format for running file on Terminal:

```
python3 EE18B104_Assign4_code.py python3 EE18B104_Assign5_code.py -h
optional arguments:
  -h, --help       show this help message and exit
  --Nx NX          size along x (default = 25)
  --Ny NY          size along y (default = 25)
  --radius RADIUS  radius of central lead (< 1) (default = 0.35cm)
  --Niter NITER    number of iterations (default = 1500)
```

Output format for a sample input [35, 35, 0.4, 1500]:

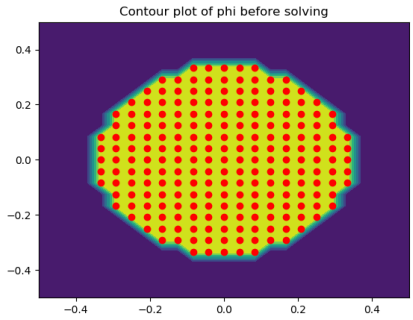
```
python3 EE18B104_Assign5_code.py --Nx 35 --Ny 35 --radius 0.4
Maximum bound for error =  2.2578098837756867e-05
Last iteration value of error =  1.1730878490823216e-09
```

The subsequent sections cover the plots and discuss the inferences drawn from them.

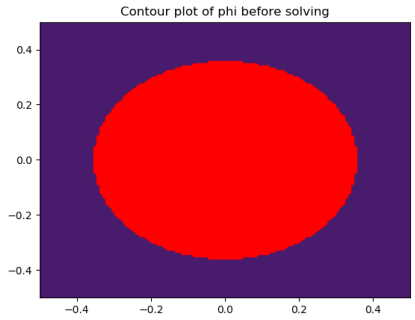
---

4 Plots

4.1 Contour of Plate Potential before solving

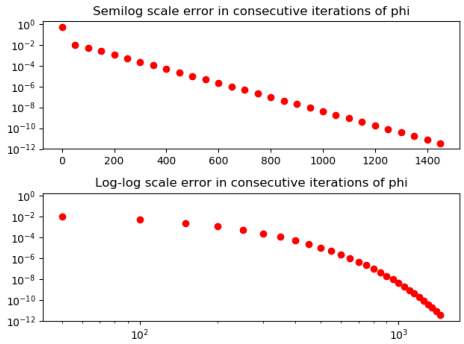


(a) Contour plot for Nx = Ny = 25, Niter = 1500

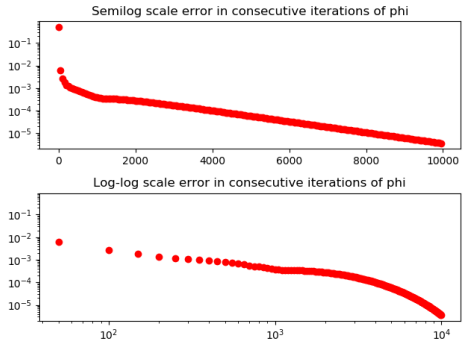


(b) Contour plot for Nx = Ny = 125, Niter = 10000

4.2 Error between Consecutive iterations of phi

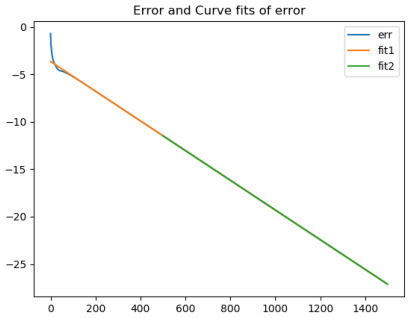


(c) Error in semilog and log-log scales for Nx = Ny = 25, Niter = 1500

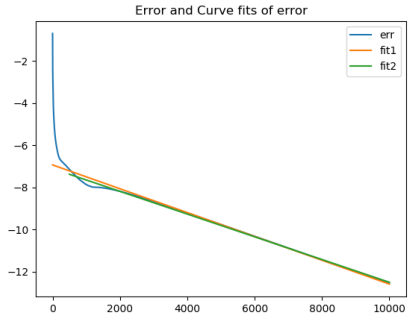


(d) Error in semilog and log-log scales for Nx = Ny = 125, Niter = 10000

4.3 Exponential Model Fits of the error



(e) Fit for Nx = Ny = 25, Niter = 1500



(f) Fit for Nx = Ny = 125, Niter = 10000

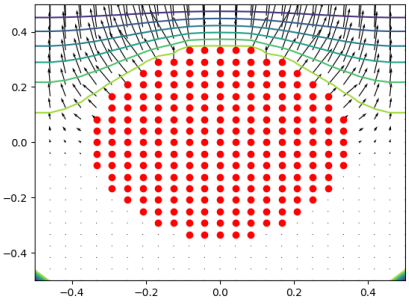
For Nx = Ny = 25:

```
Maximum bound for error =  1.4804273293889298e-08
Last iteration value of error =  1.6930901125533637e-12
```

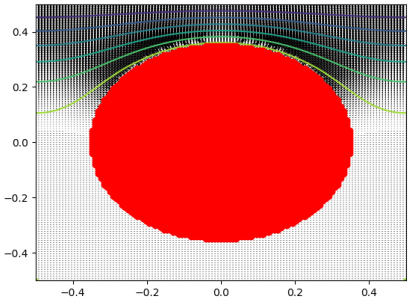
For Nx = Ny = 125:

```
Maximum bound for error =  2164.823666820692
Last iteration value of error =  0.00032880648570704496
```

4.4 Contour and Quiver Plot of Potential and Current Distributions

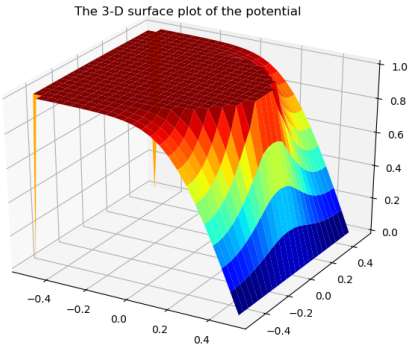


(g) Contour Plot for Nx= Ny = 25, Niter = 1500

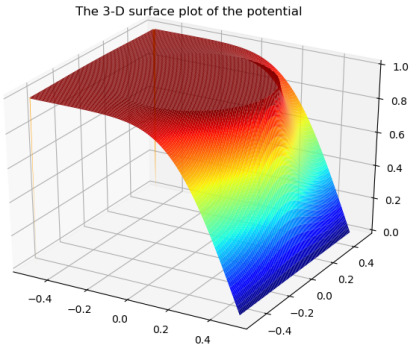


(h) Contour Plot for Nx= Ny = 125, Niter = 10000

4.5 Surface Plot of Potential



(i) Surface Plot for Nx= Ny = 25, Niter = 1500



(j) Surface Plot for Nx= Ny = 125, Niter = 10000

5 Inferences

5.1 Error Behaviour

1. The error in consecutive iterations of phi falls linearly in the semi-log scale. This is because the error is of the form Aexp(Bx).

2. In the log-log scale, an interesting behaviour is observed- the fall in error is steeper beyond about 500 iterations.

3. Note that the saturating value of error for granularity of 125 at the 1500^{th} iteration is of the order of 10^{-3} while in case of granularity being 25, the error is of the order of 10^{-11}.

4. The error fits, obtained by using the curve_fit() method of scipy.optimize has a small deviation as the number of iterations become larger.

5.2 Behaviour of Potential

1. Although the potential is defined initially as 1 within the central region and 0 everywhere else, the contour plot for the potential has a few steps between 0 and 1. This is because the contour function is calculated at intermediate values (where x is not defined in the input array) and the values at these intermediate points are interpolated by python.

2. The currents in the plate are normal to the contour lines. In addition, most of the current flow is between the two electrodes with some fringing of the current near the edges.

3. Very little current flows through the other edges of the plate. This is primarily because most of the drop in the potential on the plate is between the central region and the top edge of the plate (as is clear from the surface plot).

4. The plate gets hot due to Joule-heating effects. The heat generated is proportional to the square of the current density. Thus, the top edge of the plate would be much hotter in comparison to the other regions of the plate.

$\star\star\star$