Assignment No 5

Prasanna Bartakke EE19B106

March 24, 2021

1 Introduction

We wish to solve for the currents in a resistor. The currents depend on the shape of the resistor and we also want to know which part of the resistor is likely to get hottest.

2 Equations

A cylindrical wire is soldered to the middle of a copper plate and its voltage is held at 1 Volt. One side of the plate is grounded, while the remaining are floating. The plate is 1 cm by 1 cm in size.
Conductivity

$$\vec{j} = \sigma \vec{E} \tag{1}$$

Electric field is the gradient of the potential

$$\vec{E} = -\nabla \phi \tag{2}$$

Continuity of charge

$$\nabla \vec{j} = -\frac{\partial \rho}{\partial t} \tag{3}$$

Combining the equations ans substituting $\rho = 0$, we get

$$\nabla^2 \phi = 0 \tag{4}$$

3 Defining parameters and Initialising potential

We choose a 25x25 grid with a circle of radius 8 centrally located maintained at V = 1V by default. We also choose to run the difference equation for 1500 iterations by default. We start by creating an zero 2-D array of size Nx x Ny. then a list of coordinates lying within the radius is generated and these points are initialized to 1.

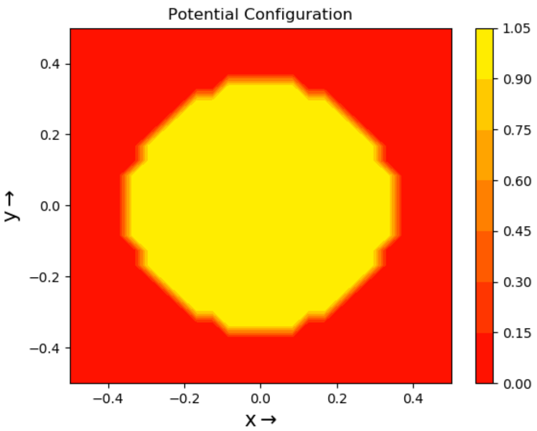

Figure 1: Initial Potential

4 Performing iterations

4.1 Updating the potential

We convert the differential equation(4) to the following difference equation.

$$\phi_{i,j} = \frac{\phi_{i-1,j} + \phi_{i,j-1} + \phi_{i+1,j} + \phi_{i,j+1}}{4} \tag{5}$$

```
def update_phi(phi,phiold):
    phi[1:-1,1:-1]=0.25*(phiold[1:-1,0:-2]+ phiold[1:-1,2:]+ phiold[0:-2,1:-1] + phi
    return phi
```

4.2 Enforcing Boundary Conditions

The bottom boundary is grounded. The other 3 boundaries have a normal potential difference zero.

```
def boundary(phi,mask = np.where(X**2+Y**2<(0.35)**2)):
    #Left
    phi[:,0]=phi[:,1]
    #Right
    phi[:,Nx-1]=phi[:,Nx-2]
    #Top
    phi[0,:]=phi[1,:]
    #Bottom
    phi[Ny-1,:]=0
    #wire
    phi[mask]=1.0
    return phi
```

4.3 Calculating error after each iteration

```
err = np.zeros(Niter)
for k in range(Niter):
    phiold = phi.copy()
    phi = update_phi(phi,phiold)
    phi = boundary(phi)
    err[k] = np.max(np.abs(phi-phiold))
```

4.4 Plotting Error

Plot the errors on semi-log and log-log plots. As we can see that the error falls slowly and this is one of the reasons why this method of solving the Laplace equation is not preferred.

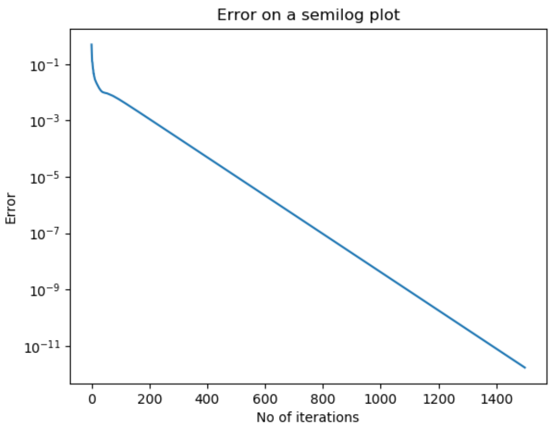

Figure 2: Semilog plot of errors

5 Fitting the error

The error is a decaying potential for higher iterations. We attempt to fit a function

$$y = Ae^{Bx} \tag{6}$$

$$log(y) = log(A) + Bx \tag{7}$$

We estimate log(A) and B with the least squares method. fit1 is considering all the iterations and fit2 is considering iterations from 500 onwards. There is very little difference between the two fits.

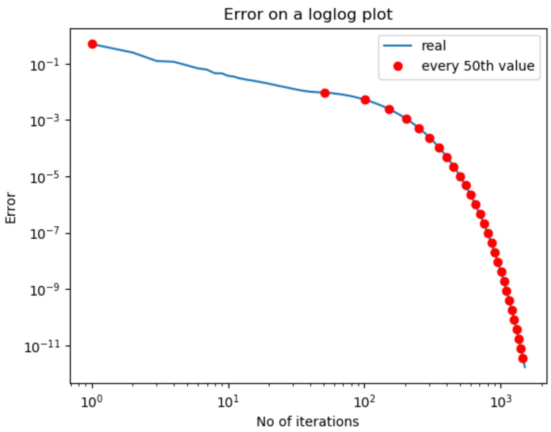

Figure 3: Loglog plot of errors

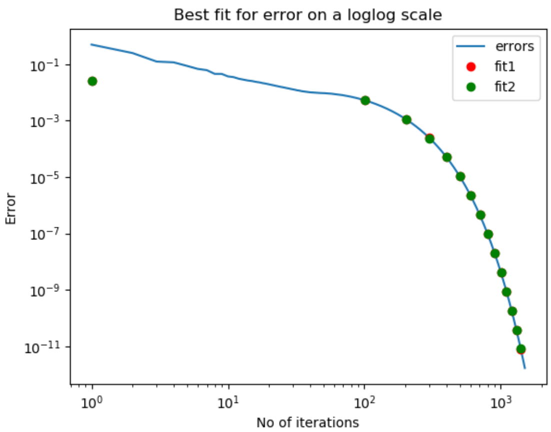

Figure 4: Best fit of errors

6 Plotting maximum error

7 Plotting Potential

8 Calculating current density

$$J_{x,ij} = \frac{\phi_{i,j-1} - \phi i, j+1}{2} \tag{8}$$

$$J_{y,ij} = \frac{\phi_{i-1,j} - \phi i+1, j}{2} \tag{9}$$

From the current density plot, we can see that hardly any current flows through the top part of the wire. We can conclude that the lower surface

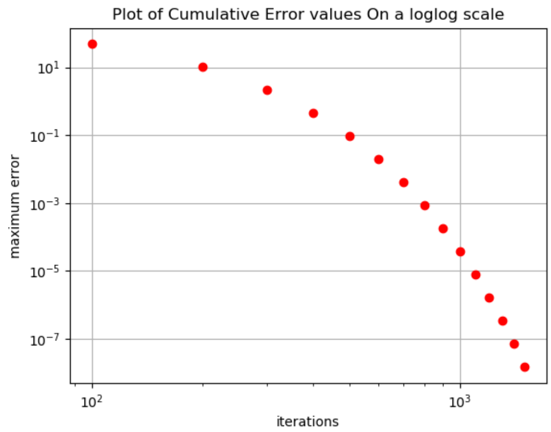

Figure 5: Cumulative error values on a log log scale

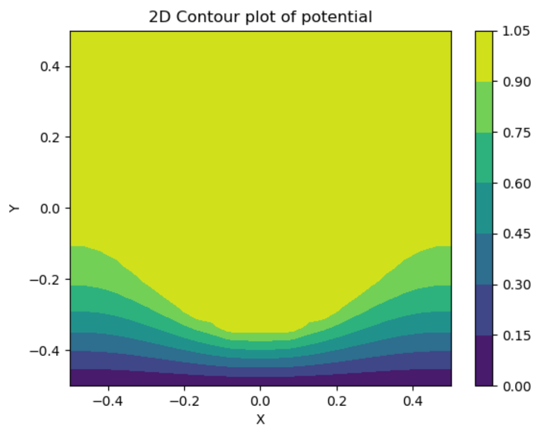

Figure 6: 2d plot of potential

being grounded, the easiest way for charge carriers to flow from the electrode would be directly through the lower half of the wire, thus avoiding a longer, more resistive path through the top half of the wire.

9 Conclusion

We can find solution to Laplace's equation for a given system using a finite differentiation approximation. The error is seen to decay at a gradual pace. Thus the chosen method of solving Laplace's equation is inefficient. On analysing the vector plot of the currents, we can conclude that the current was mostly restricted to the bottom of the wire.

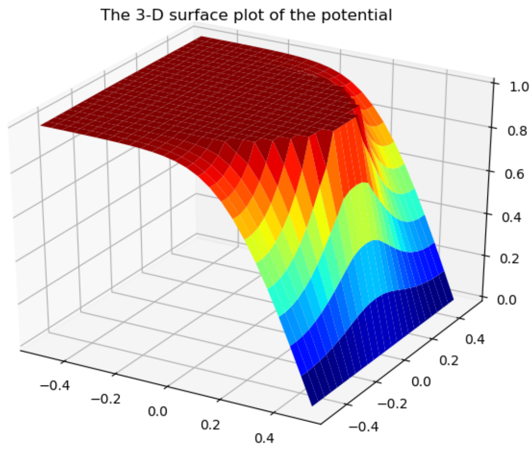

Figure 7: 3d plot of potential

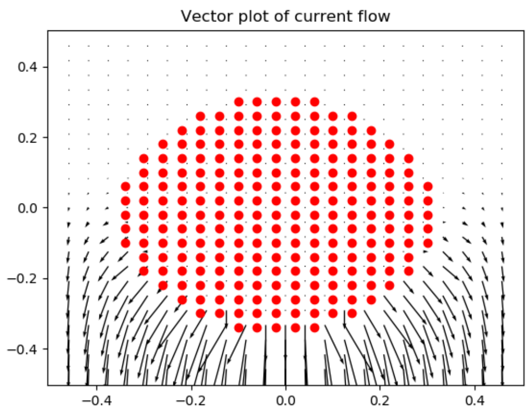

Figure 8: Vector plot of current