EE2703 : Applied Programming Lab
Week5 : The Resistor Problem

Aditya Suresh
EE20B004

March 11, 2022

AIM

- To solve for potential and current through a resistor
- To make corresponding plots

Theory

A 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. So:

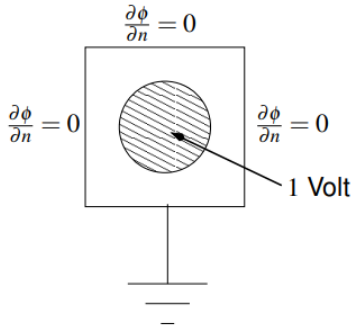

Figure 1: Figure of the problem

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

And Electric field is given by gradient of potential:

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

Continuity of charge yields :

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

Combining them :

$$\nabla.(-\sigma\nabla\phi) = -\frac{\partial\rho}{\partial t}$$

So:

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

For DC currents :

$$\nabla^2\phi = 0$$

Plots and Analysis

Forming Potential ϕ and seeing its contour

1. The potential is initialised and contour is plotted via :-

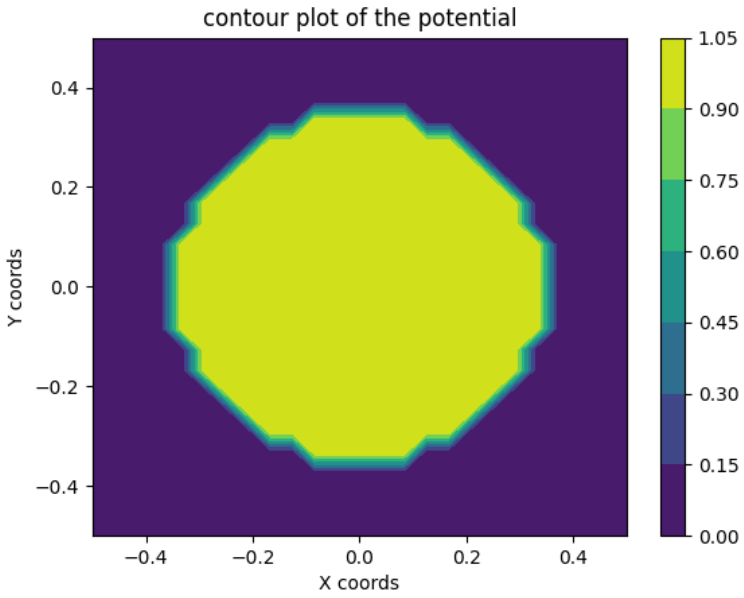

Figure 2: Potential ϕ contour

```
phi = np.zeros((Ny,Nx))
for k in range(Niter):


    oldphi = phi.copy()
    left_matrix = oldphi[1:-1,0:-2]
    right_matrix = oldphi[1:-1 ,2: ]
    top_matrix = oldphi[0:-2 ,1:-1 ]
    down_matrix= oldphi[2: ,1:-1 ]

    phi[1:-1, 1:-1] =0.25*(left_matrix + right_matrix + top_matrix + down_ma

    phi[:,0]=phi[:1,1]
    phi[:, -1] = phi[:, -2]
```

```
phi[ 0 ,:] = phi[1, :]
phi[-1, :] = 0

phi[ii] =1.0
```

Error Plot on semilog and loglog:

1. Using library matplotlib's, semilogy and loglog to plot-

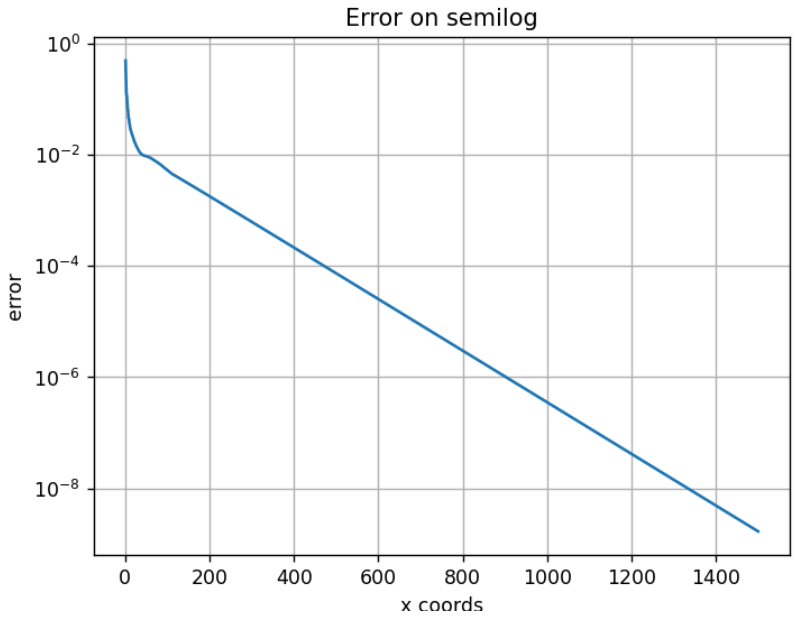

Figure 3: Error on semilog

```
figure(2)
title("Error on semilog")
xlabel("x coords")
ylabel("error")
grid()
semilogy(Nither_as_x, errors)
show()
```

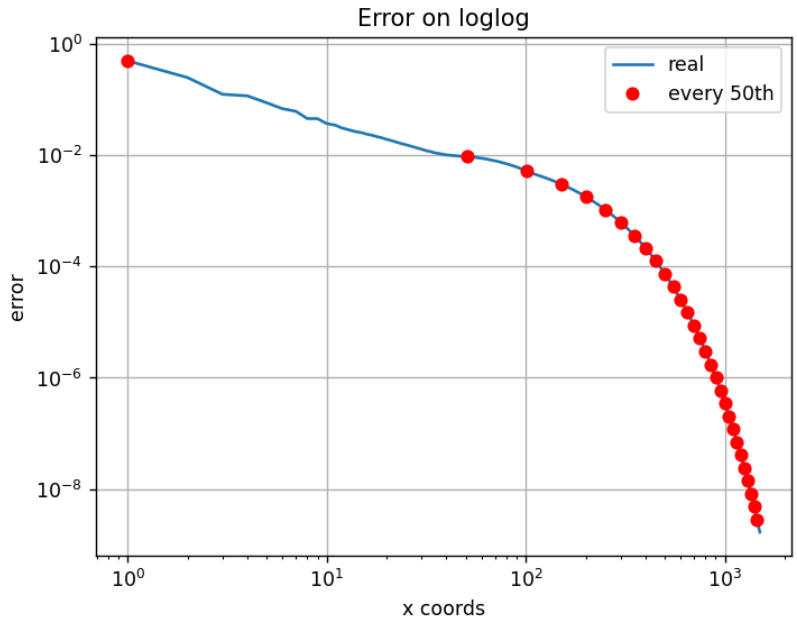

Figure 4: Error on loglog

```
#error on loglog
figure(2)
title("Error on loglog")
xlabel("x coords")
ylabel("error")
grid()
loglog(Nither_as_x, errors, label ="real")
loglog(Nither_as_x[::50], errors[::50], "ro", label = "every 50th")
legend()
show()
```

Error plot :Fit1 Fit2

1. Here we find the A and B via lstsq method and plot the corresponding fit1 and fit2 along with true graph on loglog scale

---

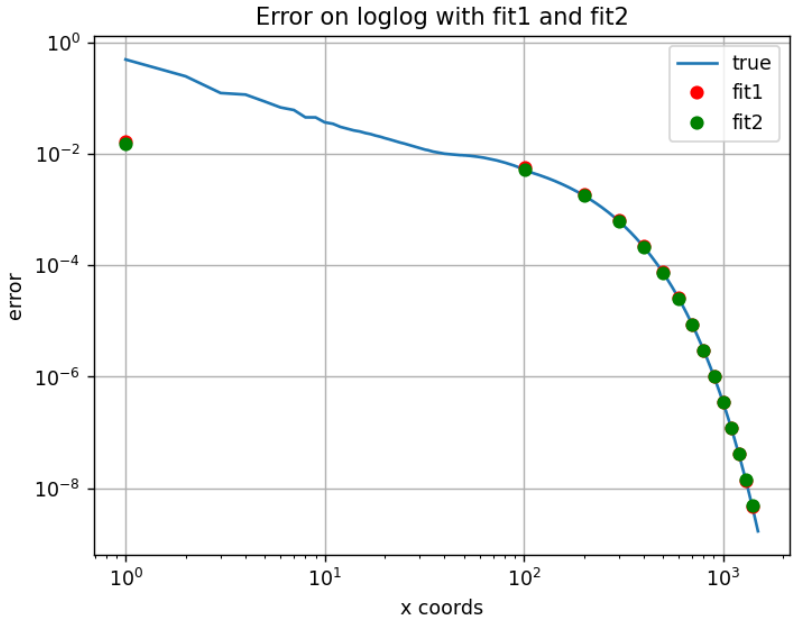

Figure 5: Errorbar plot :fit1 fit2(for 500 iterations)

```
def finding_B_A (error, Niter, no_of_points=0):
    log_err = np.log(error)[-no_of_points:]
    M =  c_( (np.array(range(Niter)) +1)[-no_of_points:], np.ones(log_err.sh
    x = s.lstsq(M, log_err)

    return x[0][0], x[0][1]

B,A = finding_B_A(errors,Niter)
B2,A2 = finding_B_A(errors, Niter, 500)

figure(3)
title("Error on loglog with fit1 and fit2")
xlabel("x coords")
ylabel("error")
grid()
loglog(Nither_as_x, errors, label = "true")
loglog(Nither_as_x[::100], np.exp((B*Nither_as_x + A)[::100]), "ro" , label
loglog(Nither_as_x[::100], np.exp((B2*Nither_as_x + A2)[::100]), "go", label
legend()
show()
```

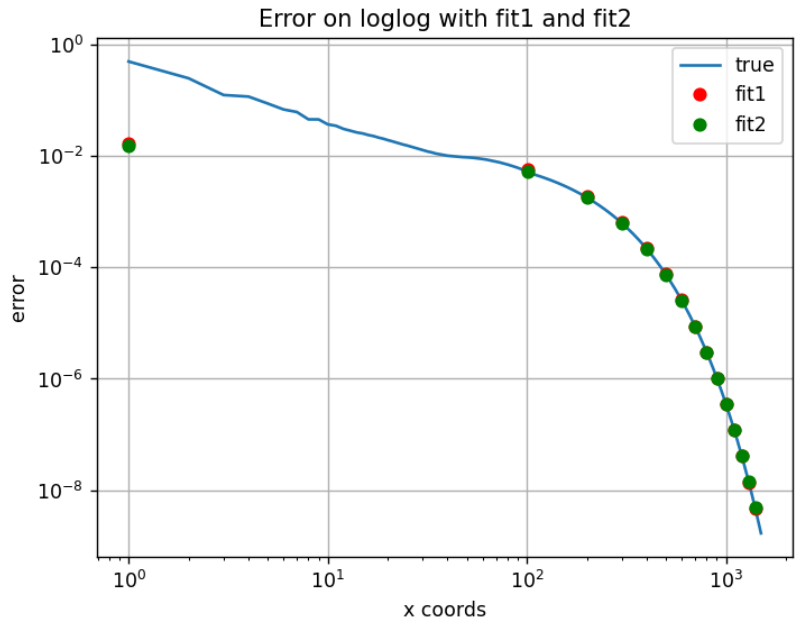

Figure 6: Errorbar plot on semilog :fit1 fit2(for 500 iterations)

```
figure(3)
title("Error on semilog with fit1 and fit2 ")
xlabel("x coords")
ylabel("error")
grid()
semilogy(Nither_as_x, errors)
semilogy(Nither_as_x[::100], np.exp((B*Nither_as_x + A)[::100]), "ro"   , lab
semilogy(Nither_as_x[::100], np.exp((B2*Nither_as_x + A2)[::100]), "ro", lab
show()
```

Cummulative error on loglog scale

1. This method of solving Laplace's Equation is known to be one of the worst available. This is because of the very slow coefficient with which

the error reduces.

2. we know error scales as :

$$error = Ae^{Bk}$$

where k is no. of iteration

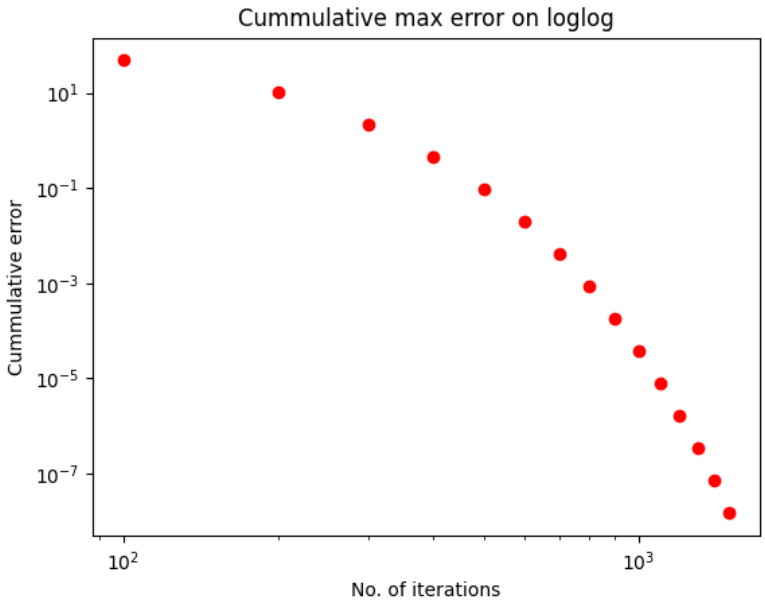

Figure 7: Cummulative error on loglog scale

Surface Plot of Potential

```
fig4=figure(4) # open a new figure
title("3-D surface plot of the potential")
ax=p3.Axes3D(fig4) # Axes3D is the means to do a surface plot
title("The 3-D surface plot of the potential")
surf = ax.plot_surface(Y, X, phi.T, rstride=1, cstride=1, cmap=cm.jet)
show()
```

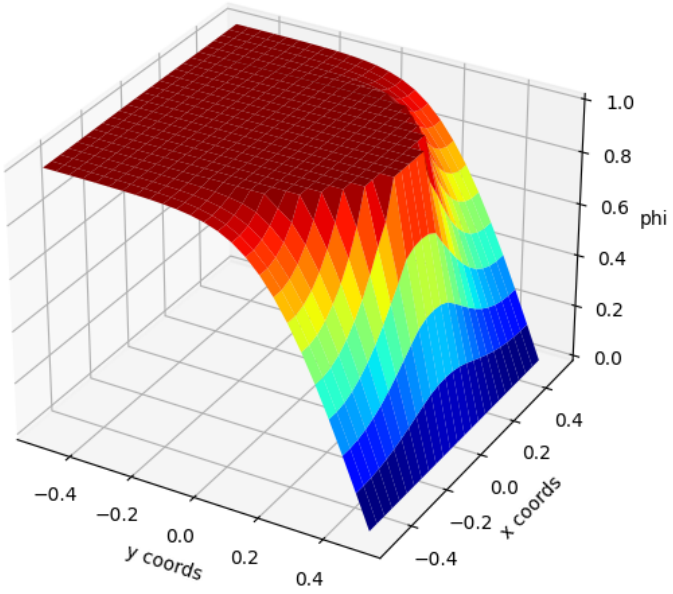

Figure 8: 3-D Surface Plot of Potential

Contour Plot of the Potential

1. Contour plots are plotted using the library matplotlib counterf:

```
title("2D Contour plot of potential")
xlabel("X")
ylabel("Y")
grid()
plot((ii[0]-Nx/2)/Nx,(ii[1]-Ny/2)/Ny,'ro')
contourf(Y,X[::-1],phi)
colorbar()
show()
```

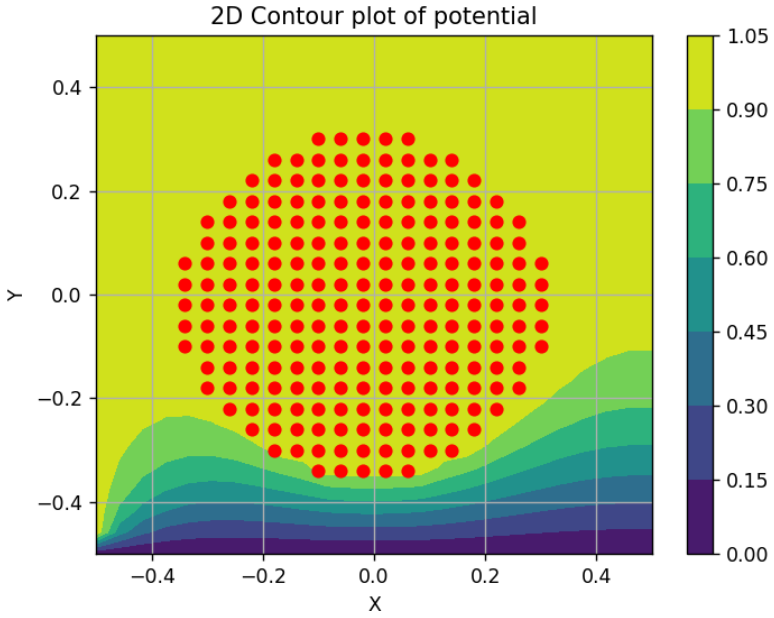

Figure 9: Contour Plot of ϕ

Vector Plot of Currents

We know :

$$j_x = -\frac{\partial \phi}{x}$$

$$j_y = -\frac{\partial \phi}{y}$$

Which numerically translates to :

$$J_{x,ij} = \frac{1}{2}(\phi_{i,j-1} - \phi_{i,j+1})$$

And

$$J_{y,ij} = \frac{1}{2}(\phi_{i_1,j} - \phi_{i+1,j})$$

```
Jx,Jy = (1/2*(phi[1:-1,0:-2]-phi[1:-1,2:]),1/2*(phi[:-2,1:-1]-phi[2:,1:-1]))

#plotting current density

title("Vector plot of current flow")
```

```
quiver(Y[1:-1,1:-1],-X[1:-1,1:-1],-Jx[:,::-1],-Jy)
plot((ii[0]-Nx/2)/Nx,(ii[1]-Ny/2)/Ny,'ro')
grid()
show()
```

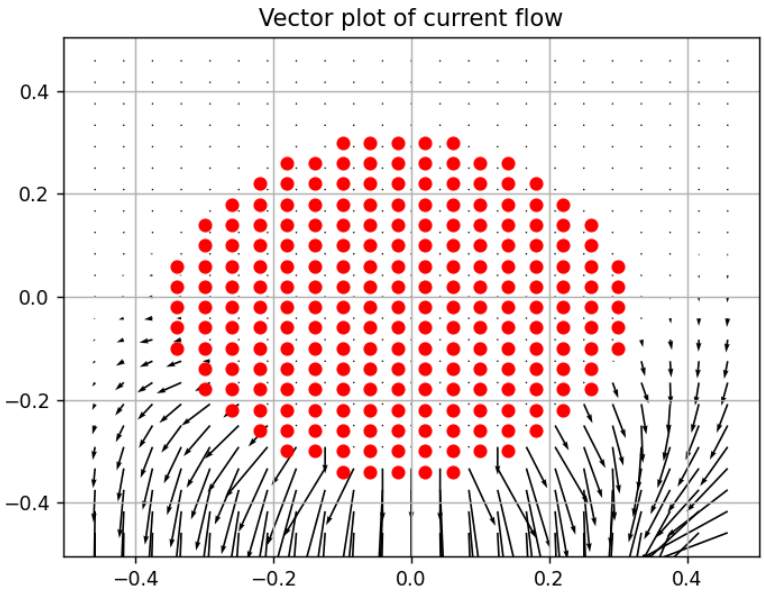

Figure 10: The vector plot of the current flow

Conclusions

We have solved the potential and current vectors via Laplace's equation and differentiation and some approximation. We have also plotted the corresponding plots to visualise error , potential and current