EE2703 Assignment 5

Valipireddy Venkata Satyanarayana Reddy, EE20B145

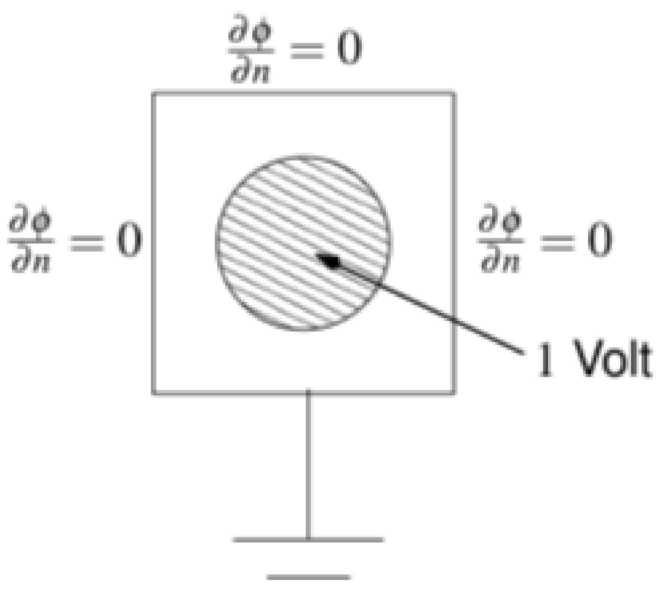
March 11, 2022

1 Introduction

This assignment focusses on finding out the flow of currents in a resistor in a conductor. We also wish to find out the part of the conductor which is likely to get hottest.

1.1 Abstract

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.



We have to solve the equation $\nabla^2 \phi = 0$
i.e. $\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0$
Solving this numerically we get:
$\phi_{i,j} = \frac{\phi_{i+1,j}+\phi_{i-1,j}+\phi_{i,j+1}+\phi_{i,j-1}}{4}$
So a matrix of potential 'ϕ' is initialized. And we have to update the potential using the above equation.

The boundary condition used is that $\frac{\partial \phi}{\partial n} = 0$.
Thus the potential doesn't change in the normal direction at the boundaries.

2 Import Libraries

```
from pylab import *
import numpy as np
import mpl_toolkits.mplot3d.axes3d as p3
```

3 Set the parameters.

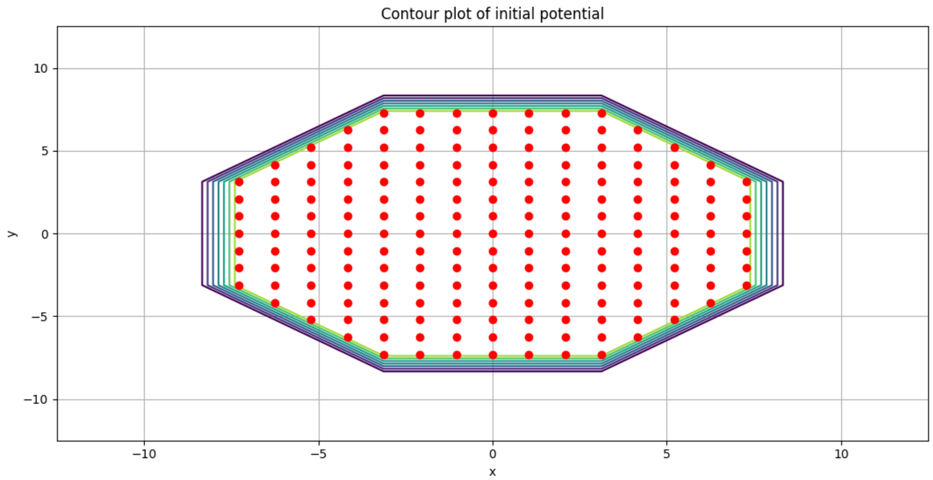
```
Nx = 25         # No. of steps along the x direction
Ny = 25         # No. of steps along the y direction
radius = 8   # Radius of the wire loop
Niter = 1500    # No. of iterations to find potential
errors = np.zeros(Niter)   # Error array is declared
```

4 To obtain the coordinates of the wire on the conductor

```
x = np.linspace(-12.5,12.5,Nx)    # x coordinate array
y = np.linspace(12.5,-12.5,Ny)    # y coordinate array
X,Y = meshgrid(x,y)             # The 2D grid of x and y coordinates
phi = np.zeros((Nx,Ny))         # Potential is initialised with zeros
Rc = radius
ii = where(X*X + Y*Y <= (Rc*Rc))  # Area covered by ring is found
phi[ii] = 1.0                   # Area covered by ring is initialised with 1 V
```

5 Plot the Contour Plot of the potential

```
contour(X,Y,phi)
plot(x[ii[0]],y[ii[1]],'ro')
grid()
title('Contour plot of initial potential')
xlabel('x')
ylabel('y')
show()
```



6 Update the potential matrix along with the error in each iteration

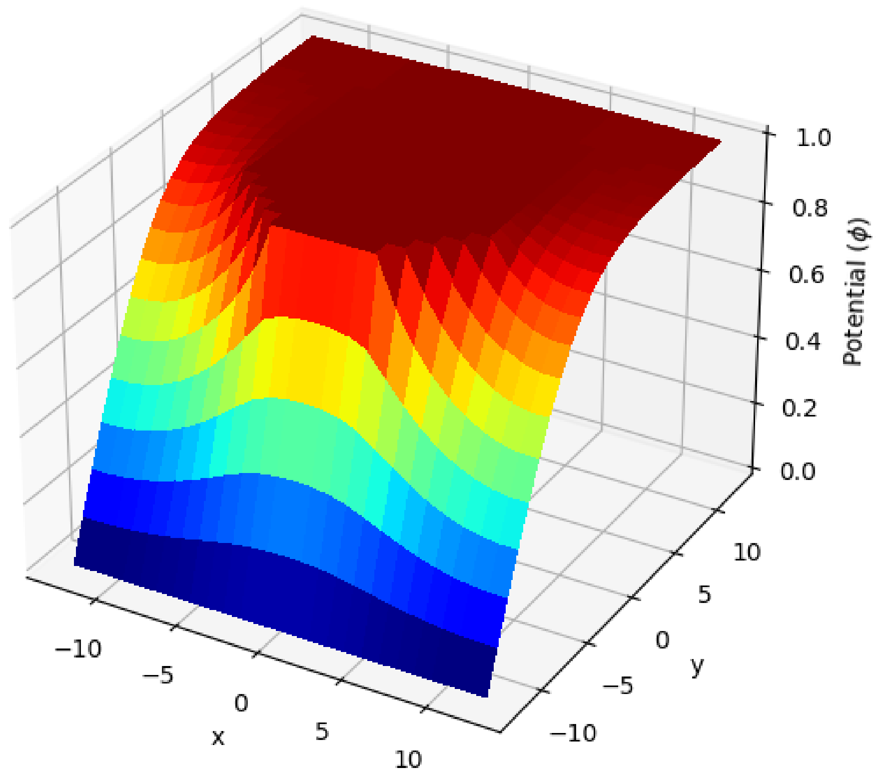
```
newphi = np.zeros((Nx,Ny)) # This is to temporarily store the newly calculated values so as to c
for k in range(Niter):
    oldphi = phi.copy()    # Phi before iteration is stored to calculate error
    newphi[1:-1,1:-1] = 0.25*(phi[1:-1,0:-2] + phi[1:-1,2:] + phi[0:-2,1:-1] + phi[2:,1:-1])  #

    newphi[1:-1,0] = newphi[1:-1,1]          # Boundary conditions applied
    newphi[1:-1,Nx-1] = newphi[1:-1,Nx-2]
    newphi[0,1:-1] = newphi[1,1:-1]
    newphi[ii] = 1.0

    errors[k] = max(np.absolute(np.subtract(oldphi.flatten(),newphi.flatten())))   # Error calcu
    phi = newphi.copy()
```

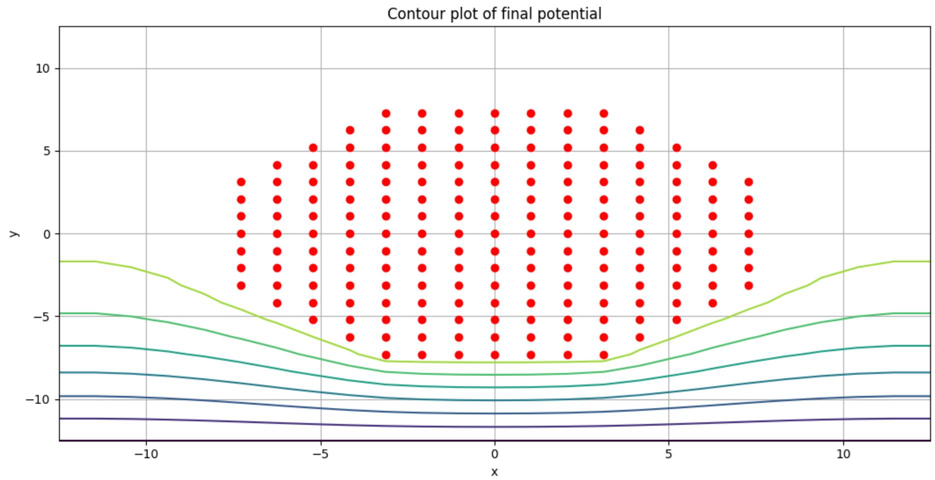
7 Plot the 3D figure of the potential after the updates

```
fig1 = figure(4)
ax = p3.Axes3D(fig1)
title('The 3-D surface plot of the potential')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('Potential $(\phi)$')
surf = ax.plot_surface(X, Y, phi, rstride=1, cstride=1, cmap=cm.jet,linewidth=0, antialiased=Fal
show()
```



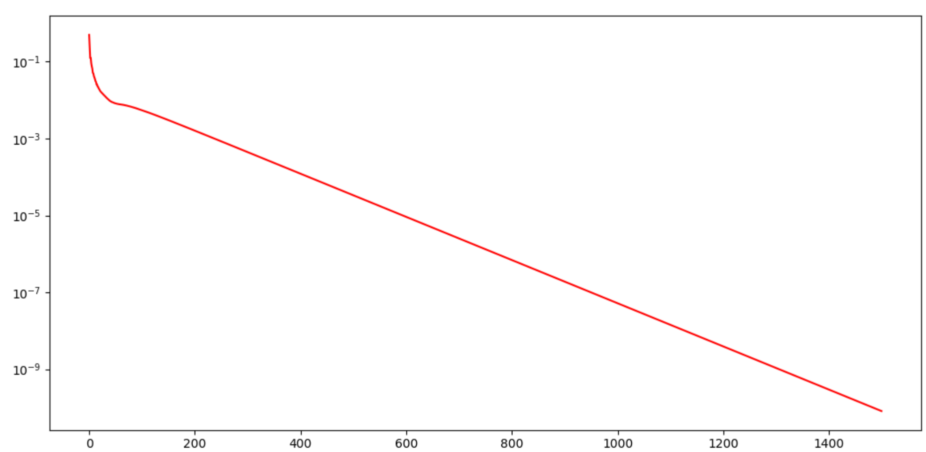
8 Plot of the Contour Diagram of the potential

```
contour(x,y,phi)
plot(x[ii[0]],y[ii[1]],'ro')
xlabel('x')
ylabel('y')
title('Contour plot of final potential')
grid()
show()
```



9 Plot of error vs iterations

```
plt.semilogy(range(Niter),errors,"r")
plt.show()
```



10 Error Estimation

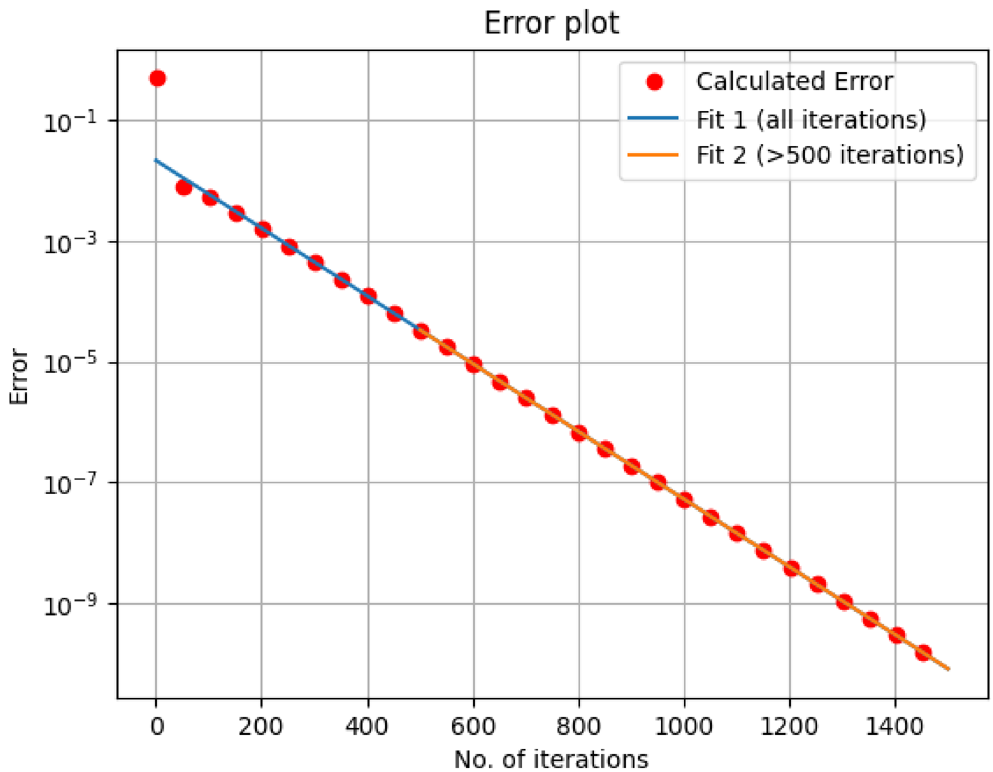
The error in this algorithm of updates is of the form Ae^{bx}
$\therefore y = Ae^{bx}$

$\therefore \log y = \log A + bx$
Therefore if we fit this using least squares method we can estimate $logA$ and b

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \cdot \begin{bmatrix} logA \\ b \end{bmatrix} = \begin{bmatrix} logy_1 \\ logy_2 \\ \vdots \\ logy_n \end{bmatrix} \tag{1}$$

Both fits one for total 1500 and one for after 500 iters.

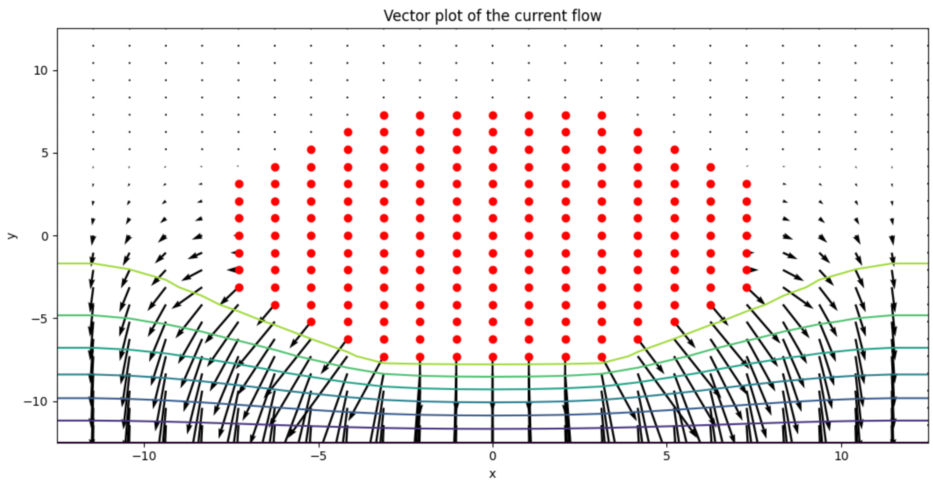
```
xError = np.linspace(1,Niter,1500)   # x Values for the equation
yError = np.log(errors)              # y values for equation
A=np.zeros((Niter,2))                # 2D matrix initialised
A[:,0] = 1
A[:,1] = xError
const = lstsq(A,yError)[0]           # parameters log(A) and B are found
yError = const[0] + const[1]*xError  # Above mentioned equation applied to find best fit line
yError = np.exp(yError)
xError2 = np.linspace(501,Niter,1000)
yError2 = np.log(errors[500:])
B=np.zeros((Niter-500,2))
B[:,0] = 1
B[:,1] = xError2
const = lstsq(B,yError2)[0]
yError2 = const[0] + const[1]*xError2
yError2 = np.exp(yError2)
semilogy(np.arange(1,1501,50),errors[0::50],'ro')
plot(xError,yError)
plot(xError2, yError2)
grid()
title('Error plot')
xlabel('No. of iterations')
ylabel('Error')
legend(('Calculated Error','Fit 1 (all iterations)','Fit 2 (>500 iterations)'))
show()
semilogy(np.arange(1,1501,50),errors[0::50],'ro')
plot(xError,yError)
plot(xError2, yError2)
grid()
title('Error plot')
xlabel('No. of iterations')
ylabel('Error')
legend(('Calculated Error','Fit 1 (all iterations)','Fit 2 (>500 iterations)'))
show()
```



11 Extracting the currents from the potential equation

We have $J = \sigma.E$
$\therefore J_x = -\sigma.\frac{\partial\phi}{\partial x}$
$\therefore J_y = -\sigma.\frac{\partial\phi}{\partial y}$
Taking $\sigma = 1$ for the sake of just getting the profile of the currents.
Thus, $J_{x,i,j} = \frac{\phi_{i,j-1}-\phi_{i,j+1}}{2}$,
$J_{y,i,j} = \frac{\phi_{i-1,j}-\phi_{i+1,j}}{2}$

```
Jx = np.zeros((Nx,Ny))
Jy = np.zeros((Nx,Ny))
Jy[1:-1,1:-1] = 0.5*(phi[1:-1,2:] - phi[1:-1,0:-2])
Jx[1:-1,1:-1] = 0.5*(phi[2:,1:-1] - phi[0:-2,1:-1])
plot(x[ii[0]],y[ii[1]],'ro')
xlabel('x')
ylabel('y')
title('Vector plot of the current flow')
quiver(y,x,Jy[::-1,:],Jx[::-1,:])
contour(x,y,phi)
show()
```



Note:
The currents are perpendicular to the equipotential lines in the graph.
The magnitudes of the current are scaled to $\frac{1}{8}$th of their original values for neatness in the graph

12 Heat Map of the conductor

As the current flows in the conductor, it heats up. Thus increasing it's temperature.This phenomenon is called Joule Heating.
The heat equation is given by :
$\kappa\nabla^2 T = -\frac{1}{\sigma}|J|^2$
We take $\kappa = 1$,$\sigma = 1$ and $\Delta x = 1$ for simplicity.
Thus expanding this equation gives us:
$T_{i,j} = \frac{T_{i+1,j}+T_{i-1,j}+T_{i,j+1}+T_{i,j-1}+|J|^2}{4(\Delta x)^2}$
Thus by updating the temperature `Niter` times we get a temperature which converges.
The boundary condition is that at the boundary $\frac{\partial T}{\partial n} = 0$

13 Discussions and Conclusions

1:The potential matrix of the conductor converges to a solution using the update algorithm with an error of Ae^{bx} where x is the number of iterations.
2:The currents flow mostly on the lower part of the condutor where the potential drop is maximum as seen in the graphs.
3:The currents are perpendicular to the equipotential lines in the graph.
4:The conductor gets the hottest at the lower part of the conductor where most of the current is flowing.