# Assignment 5: The Resistor Problem

Manognya Param Koolath

March 5, 2019

## The task

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 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.

## Procedure

We use a matrix $\phi$ to store the potential value. We initialize it to 1 where the wire is, and zero elsewhere. Then we iteratively calculate the potential by using the Laplace equations in a vectorised fashion. We calculate the errors for each iteration.
We assume that the errors follow an exponential fit. But practically, it actually follows after around 500 iterations. We fit the two sets of data to the following curve:

$$y = Ae^{Bx}$$

$$\log y = \log A + Bx$$

We fit the curves using the python function `lstsq` from the scipy library. We estimate the error values using these fits and have plotted them in Figure 4. We have plotted the 3D surface plot of the potential in Figure 5. The current density is calculated from the potential values in the $\phi$ matrix. The vector plot of this is plotted in Figure 6.

## Python code

The code is properly commented and completely vectorised.

**Assignment 4 Code**

```
#Assignment 5
import numpy as np
import matplotlib.pyplot as plt
import scipy.special as sp
from scipy.linalg import lstsq
import scipy.integrate as spint
import mpl_toolkits.mplot3d.axes3d as p3
from scipy import ndimage

Nx = 25 #Size along x
Ny = 25 #Size along y
radius = 0.35 #Radius of central lead
Niter = 1500 #Number of iterations

phi = np.full((Nx,Ny),0.0)
# print(Ny//2)
x = np.linspace(-1, 1, Nx)
y = np.linspace(-1, 1, Ny)
Y, X = np.meshgrid(y,x)
# print(Y.shape, X.shape)
# print(Y)
ii = np.where(X*X + Y*Y <= radius*radius)
# print(origin)
# print(ii)
phi[ii] = 1.0
# print(phi)

#Plotting a contour plot:
plt.figure()
cp = plt.contour(X,Y,phi.T)#,10)
plt.plot(x[ii[0]], y[ii[1]], 'ro')
plt.clabel(cp,inline=True, fontsize=7)
plt.title(r'Contour plot of phi')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()

# print(phi)
# k=np.array([[0,1,0],[1,0,1],[0,1,0]])
# print(k.shape)
errors=np.full(Niter,0.0)
for m in range(Niter):
```

```
        oldphi=phi.copy()
        # phi=(ndimage.convolve(oldphi,k,mode='constant',cval=0.0))/4.0
        phi[1:-1,1:-1]=0.25*(phi[1:-1,0:-2] + phi[1:-1,2:] + phi[0:-2,1:-
        #Boundary conditions
        phi[1:-1,0]=phi[1:-1,1]
        phi[1:-1,-1]=phi[1:-1,-2]
        phi[0,1:-1]=0.0
        phi[-1,1:-1]=phi[-2,1:-1]
        phi[ii]=1.0
        errors[m]=(np.abs(np.subtract(phi,oldphi))).max()
        # if(m%50==0):
        #       print(errors[m])

#Plotting a contour plot:
plt.figure()
cp = plt.contour(X,Y,phi.T,10)
plt.plot(x[ii[0]], y[ii[1]], 'ro')
plt.clabel(cp,inline=True, fontsize=7)
plt.title(r'Contour plot of phi after iterations')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()

# #Error plot:
# plt.plot(np.arange(30),errors[::50])
# plt.xlabel('Iteration x 50',size=20)
# plt.ylabel('errors',size=20)
# plt.title(r'Error plot for every 50th iteration')
# plt.show()

#Error plot in semilogy:
plt.semilogy(np.arange(30),errors[::50])
plt.xlabel('Iteration x 50',size=20)
plt.ylabel('errors',size=20)
plt.title(r'Error plot for every 50th iteration')
plt.show()

#Curve fitting:
error_all=np.log(errors)[:,None]
error_500=np.log(errors[500:])[:,None]
# print(error_500.shape,error_all.shape)

x_all=np.concatenate((np.ones((1500,1)),np.arange(1500)[:,None]),axis=1)
x_500=np.concatenate((np.ones((1000,1)),(np.arange(1000)[:,None] + 500)),
```

```
# print(x_all.shape)

fit_all=lstsq(x_all, error_all)[0]
fit_500=lstsq(x_500, error_500)[0]
# print(fit_all.shape)
# print(fit_all)
# print(fit_500)

predicted_all=np.exp(fit_all[0])*np.exp(fit_all[1]*np.arange(1500))
predicted_500=np.exp(fit_500[0])*np.exp(fit_500[1]*np.arange(1500))

# #Error plot:
# plt.plot(np.arange(30),errors[::50],label='original error')
# plt.plot(np.arange(30),predicted_all[::50],label='Fit 1')
# plt.plot(np.arange(30),predicted_500[::50],label='Fit 2')
# plt.legend(loc='upper right')
# plt.xlabel('Iteration x 50',size=20)
# plt.ylabel('errors',size=20)
# plt.title(r'Error plot for every 50th iteration')
# plt.show()

#Error plot in semilogy:
plt.semilogy(np.arange(30),errors[::50],label='original error')
plt.semilogy(np.arange(30),predicted_all[::50],label='Predicted error fro
plt.semilogy(np.arange(30),predicted_500[::50],label='Predicted error fro
plt.legend(loc='upper right')
plt.xlabel('Iteration x 50',size=20)
plt.ylabel('errors',size=20)
plt.title(r'Error plot for every 50th iteration')
plt.show()

# #3D plot of potential:
# # print(phi)

fig1 = plt.figure()#figure(4)
ax=p3.Axes3D(fig1)
plt.title('3D surface plot of the potential')
surf = ax.plot_surface(Y, X, phi.T, rstride=1, cstride=1, cmap='viridis',
fig1.colorbar(surf, shrink=0.5, aspect=5)
plt.show()

# k=np.array([[0,1,0],[0,0,0],[0,-1,0]])
# Jx=(ndimage.convolve(phi,k,mode='constant',cval=0.0))/2.0
# k=np.array([[0,0,0],[1,0,-1],[0,0,0]])
```

```
# Jy=(ndimage.convolve(phi,k,mode='constant',cval=0.0))/2.0

Jx = np.full((Nx, Ny), 0.0)
Jy = np.full((Nx, Ny), 0.0)
Jx[1:-1, 1:-1] = 0.5*(phi[:-2,1:-1] - phi[2:,1:-1])
Jy[1:-1, 1:-1] = 0.5*(phi[1:-1,:-2] - phi[1:-1,2:])


# print(Jx[::-1,:].shape, Jx.shape)

plt.quiver(Y, X, Jy[0:-1,:], Jx[0:-1,:])
plt.plot(x[ii[0]], y[ii[1]], 'ro',label = "Points with V = 1.0 volt")
plt.title("Vector plot of the Current flow", size = 16)
plt.xlabel("Grounded side of the plate")
plt.legend(loc = "upper right")
plt.show()
```

## Results and plots

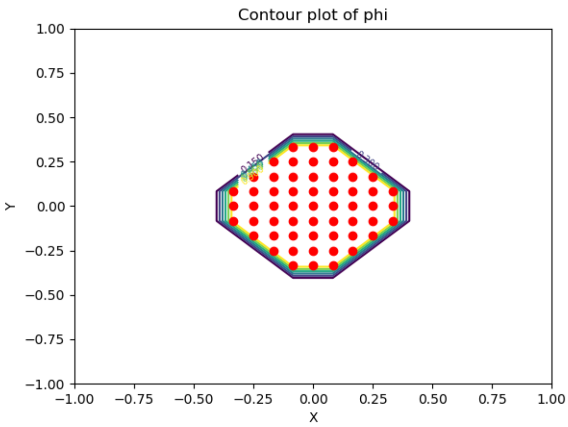


Figure 1: Initial contour plot of the potential $\phi$

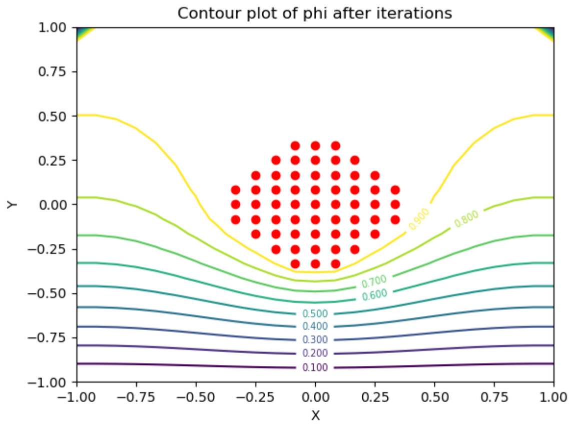


Figure 2: The contour plot of the potential after the iterative updation of the potential $\phi$

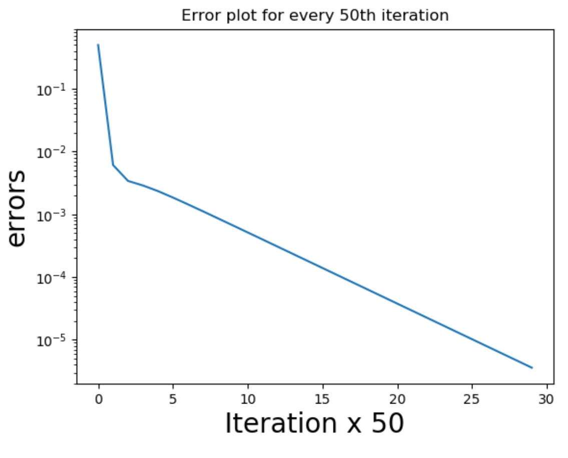


Figure 3: The error plot against number of iterations in semilog scale

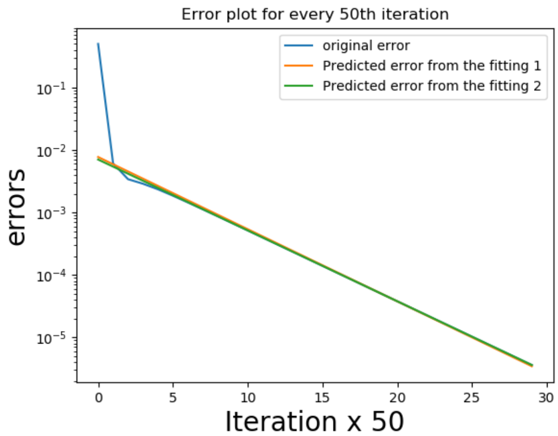


Figure 4: The error plot against number of iterations with thire estimates from fitting 1 and 2 in semilog scale

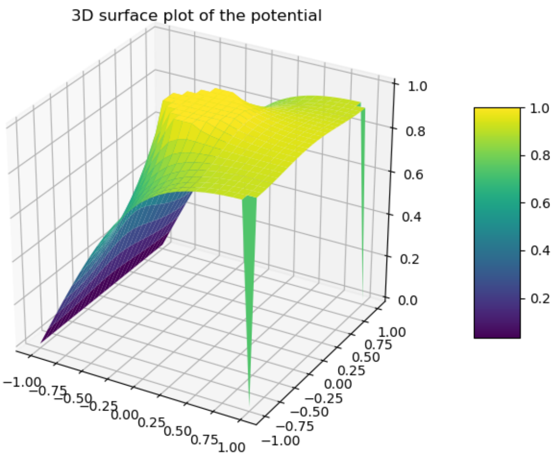


Figure 5: The 3D surface plot of the potential $\phi$

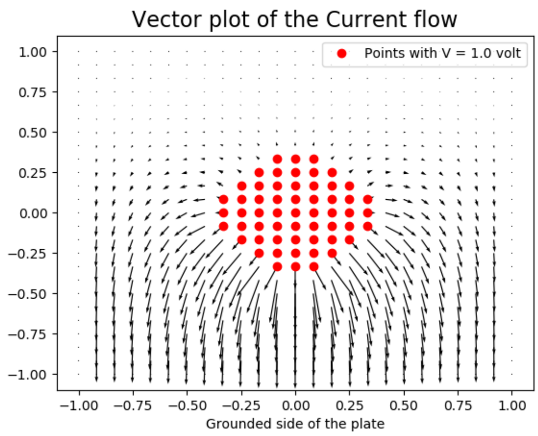


Figure 6: The vector plot of the current flow