Assignment V: Laplace Equation

Adithya, EE17B115

March 7, 2019

1

1.1 Defining Parameters and Plotting Potential

We first define the arrays and plot the potential before iterating.

```
Nx=25; # size along x
Ny=25; # size along y
radius=0.35;# radius of central lead
Niter=1500; # number of iterations to perform

potential=np.zeros((Ny,Nx))
x=np.linspace(-12,12,25)
y=np.linspace(-12,12,25)
Y,X=np.meshgrid(y,x)
ii=np.where(X*X+Y*Y<(0.35*25)**2)
potential[ii]=1.0
```

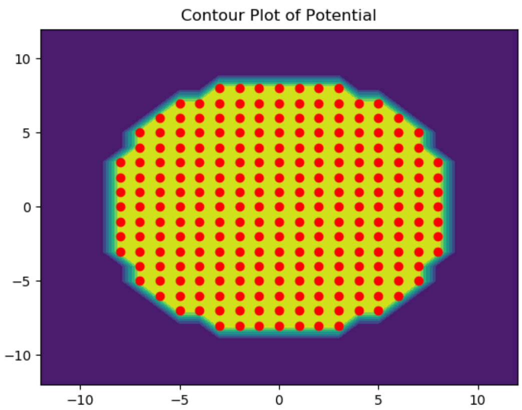


Figure 1: Contour Plot of Potential

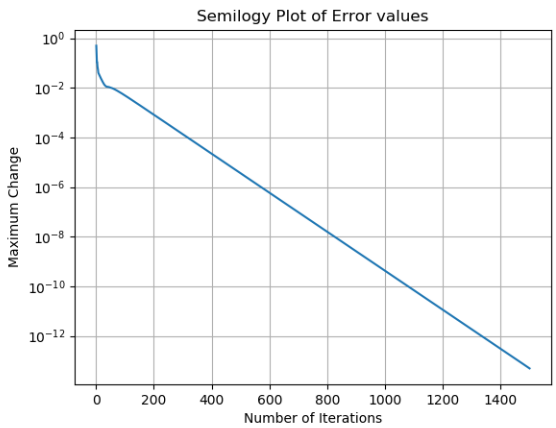
1.2 Performing Iterations

We perform iterations, in which each iteration consists of updating the potential, taking boundary conditions into account.
We should change the potential of metal also.

```
for k in range(Niter):
    oldphi=potential.copy()
    potential[1:-1,1:-1]=0.25*(oldphi[1:-1,0:-2]+ oldphi[1:-1,2:]+ oldphi[0:-2,1:-1]
    potential[1:-1,0]=potential[1:-1,1] # Left Boundary
    potential[1:-1,Nx-1]=potential[1:-1,Nx-2] # Right Boundary
    potential[0,1:-1]=potential[1,1:-1] # Top Boundary
    potential[Ny-1,1:-1]=0
    potential[xn,yn]=1.0
    errors[k]=(np.abs(potential-oldphi)).max();
```

1.3 Plotting Errors

Let us plot the maximum changes in potential matrix after every iteration in log-log and semi-log plots. To see individual data points, let us also plot every 50th point.



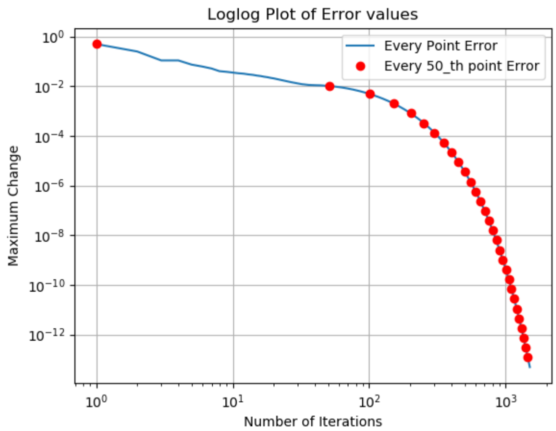


Figure 2: Errors in semi-log and log-log Vs number of iterations

1.4 Fitting the errors curve exponentially

Noticing that the errors follow an exponential curve for Niter¿500, we fit the points ¿ 500 iterations with an exponential curve.

```
def soln(x,y):
    logy=np.log(y)
    vect=np.zeros((len(x),2))
    vect[:,0]=1
    vect[:,1]=x
    logA,B=np.linalg.lstsq(vect, logy.transpose())[0]
    return (np.exp(logA),B)

A1,B1=soln(np.arange(Niter)+1,errors)
A2,B2=soln((np.arange(Niter)+1)[500:],errors[500:])
```

Let us plot the fittings we got vs original function. Overlap of green and red dots show that both are almost same.

```
plt.figure(3)
plt.grid(True)
plt.title("Original vs Fitted")
plt.loglog(np.arange(len(errors))+1,errors,label='Every Point Error')
plt.loglog((np.arange(len(errors))+1)[::50],A1*np.exp(B1*(np.arange(len(errors))+1)
plt.loglog((np.arange(len(errors))+1)[::50],A2*np.exp(B2*(np.arange(len(errors))+1)
plt.xlabel('Number of Iterations')
plt.ylabel('Maximum Change')
plt.legend()
plt.show()
```

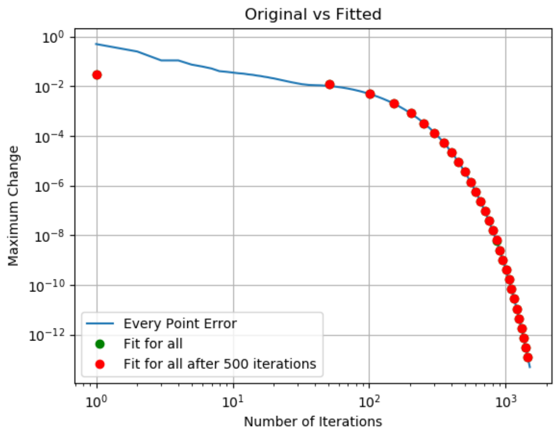


Figure 3: Fit vs Original in case of errors

1.5 Surface plot of Potential

The code that performs this task is given below:

```
fig1=plt.figure(4)
ax=p3.Axes3D(fig1) # Axes3D is the means to do a surface plot
plt.title('The 3-D surface plot of the potential')
surf = ax.plot_surface(Y, X, potential .T, rstride=1, cstride=1, cmap=plt.cm.jet)
plt.show()
```

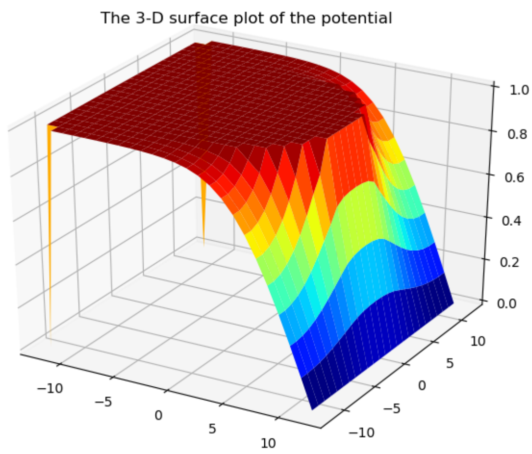


Figure 4: The 3-D surface plot of the potential

1.6 Contour Plot of the Potential

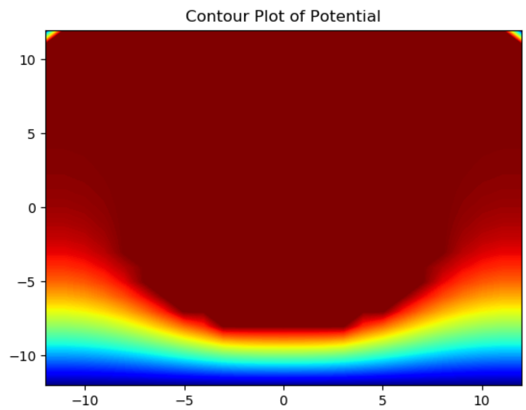


Figure 5: Contour plot of potential

1.7 Vector Plot of Currents

Let us first calculate current vectors.

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

Let us plot the current and also mark the metal area.

```
plt.figure(6)
plt.quiver(Y[1:-1,1:-1],-X[1:-1,1:-1],-Jx[:,::-1],-Jy)
xn,yn=np.where(X*X+Y*Y<(0.35*25)**2)
plt.plot(xn-12,yn-12,'ro')
plt.title('Vector Plot of current Flow')
plt.show()
```

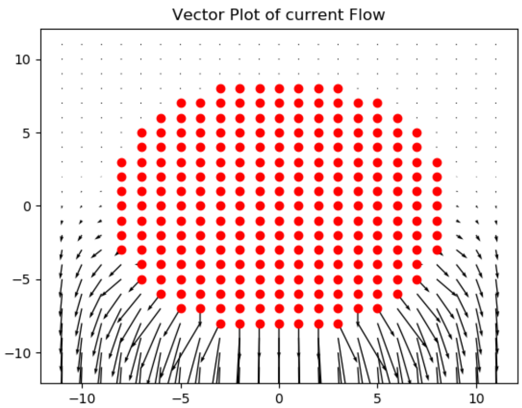


Figure 6: Vector plot of Current using quiver