ENDSEM 2022: ANTENA CURRENT IN HALF WAVE DIPOLE ANTENA

SUHAS C - EE20B132

May 13, 2022

1 Aim :

- Obtaining the current using known values like boundary conditions.
- Finding J (M-Q)*J=Im*QB equation

2 Pseudo-code

- First run the program and call current function.
- Allocate some variable to the given values and find Rz,Ru,P,PB,Q,QB J and current using (M-Q)*J=Im*QB equation and by initial conditions respectively
- Plot currents and compare them.

```
from pylab import *
import system-function as name
Note: lstsq is found as scipy.linalg.lstsq
ones(List)
zeros(List)
range(N0,N1,Nstep)
arange(N0,N1,Nstep)
linspace(a,b,N)
logspace(log10(a),log10(b),N)
X,Y=meshgrid(x,y)
where(condition)
where(condition & condition)
where(condition | condition)
a=b.copy()
lstsq(A,b) to fit A*x=b
A.max() to find max value of numpy array (similalry min)
A.astype(type) to convert a numpy array to another type (eg int)
def func(args):
...
return List
```

```
matrix=c_[vector,vector,...]  to create a matrix from vectors
figure(n) to switch to, or start a new figure labelled n
plot(x,y,style,...,lw=...)
semilogx(x,y,style,...,lw=...)
semilogy(x,y,style,...,lw=...)
loglog(x,y,style,...,lw=...)
```

3 Code

3.1 Question 1

```
"""Question 1"""
#Calculating Vector Z
z = np.linspace(-l,l,2*N+1)

#Calculating u vector
u = np.arange(1, 2*N)
#Removing the middlemost element
u = np.delete(u, N-1, axis=0)

#Calculating the I vector(standard expression ) - the actual I
Current_Vector = np.zeros(2*N+1)
Current_Vector[0:N] = Maximum_Current*sin(k*(l+z[0:N])) # for -l < z < 0
Current_Vector[N:2*N+1] = Maximum_Current*sin(k*(l-z[N:2*N+1])) # for 0 < z < l
I = Current_Vector

#Applying the given Boundary Condition

Unknown_Current=I[u]
```

3.2 Question 2

```
""" QUESTION - 2 """
# Function to compute and return matrix M, H_phi
def H_p(J,n=N,r=a):
        Matrix = (1/(2*pi*r))*(identity(2*N-2))
        V = np.dot(Matrix,J)
        return Matrix,V

Matrix,H_phi = H_p(Unknown_Current,N,a) # Getting the matrix M
```

3.3 Question 3

```
""" QUESTION - 3 """
# Computing vectors Rz, Ru and matrices PB, P
r = a
def Rij(z,r=a):
```

```
        ziz, zjz = np.meshgrid(z,z) #Return coordinate Matrices from coordinate vectors
        Radius = sqrt((ziz-zjz)**2 + r**2*np.ones((2*N+1,2*N+1)))
        return Radius

Rz = Rij(z,a)

def Riju(z,u,r=a):
        ziu, zju = np.meshgrid(z[u],z[u]) #Return coordinate Matrices from coordinate v
        R = sqrt((ziu-zju)**2 + r**2*np.ones((2*N-2,2*N-2)))
        return R

Ru = Riju(z,u,a)

def pij(r,k=k,z=dz):
        P = ((mu0/(4*pi))*(exp(-1j*k*r))*z/r)
        return P

P_ij = pij(Ru,k,dz)

RiN = Rz[N]
RiN = np.delete(RiN,[0,N,2*N],0)

Eq_0 = (exp(-1j*k*RiN))

P_B = ((mu0/(4*pi))*(Eq_0)*dz/RiN)

Eq_1 = (-1j*k/Ru)-(1/Ru**2)
Eq_2 = (-1j*k/RiN)-(1/RiN**2)
```

3.4 Question 4

```
""" QUESTION - 4 """
# Computing the matrices Qij and QB

Q_ij = -P_ij*(r/mu0)*(Eq_1)
Q_B = -P_B*(r/mu0)*(Eq_2)
```

3.5 Question 5

```
""" QUESTION - 5 """
# Solving for the current vector, I and comparing with the actual expression
#M1 = identity(2*N-2)/(2*pi*a)

Matrix_1 = np.identity(2*N-2)/(2*pi*r)
Unknown_Current_1 = dot(linalg.inv(Matrix_1-Q_ij),Q_B)

#Printing all the Matrices
print((Rz).round(2))
print((Ru).round(2))
print((P_ij).round(2))
```

```
print((P_B).round(2))
print((Q_ij).round(2))
print((Q_B).round(2))

# Finding I(expected value of current)
# Adding the three values given in question

Current_1 = np.insert(Unknown_Current_1,0,0)
Current_1 = np.insert(Current_1,N,Maximum_Current)
Current_1 = np.insert(Current_1,2*N,0)

plt.figure(1)
plt.plot(z,Current_1)
plt.plot(z,I)
plt.grid(True)
plt.savefig("/figure1.png")
plt.show()

print((I).round(2))
print((Current_1).round(2))
```

4 Printing the matrices

At N = 4

4.1

Matrix Rz:

```
[[0.01 0.13 0.25 0.38 0.5  0.63 0.75 0.88 1.  ]
 [0.13 0.01 0.13 0.25 0.38 0.5  0.63 0.75 0.88]
 [0.25 0.13 0.01 0.13 0.25 0.38 0.5  0.63 0.75]
 [0.38 0.25 0.13 0.01 0.13 0.25 0.38 0.5  0.63]
 [0.5  0.38 0.25 0.13 0.01 0.13 0.25 0.38 0.5 ]
 [0.63 0.5  0.38 0.25 0.13 0.01 0.13 0.25 0.38]
 [0.75 0.63 0.5  0.38 0.25 0.13 0.01 0.13 0.25]
 [0.88 0.75 0.63 0.5  0.38 0.25 0.13 0.01 0.13]
 [1.   0.88 0.75 0.63 0.5  0.38 0.25 0.13 0.01]]
```

4.2

Matrix Ru:

```
[[0.01 0.13 0.25 0.5  0.63 0.75]
 [0.13 0.01 0.13 0.38 0.5  0.63]
 [0.25 0.13 0.01 0.25 0.38 0.5 ]
 [0.5  0.38 0.25 0.01 0.13 0.25]
 [0.63 0.5  0.38 0.13 0.01 0.13]
 [0.75 0.63 0.5  0.25 0.13 0.01]]
```

4.3
Matrix P:

```
[0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j]
```

4.4
Matrix PB:

```
[0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j]
```

4.5
Matrix Q:

```
[[7.96170e+02-1.000e-02j 1.02400e+01+3.670e+00j 3.91000e+00+2.880e+00j
  1.67000e+00+2.450e+00j 1.30000e+00+2.360e+00j 1.06000e+00+2.300e+00j]
 [5.12000e+00+1.830e+00j 1.59233e+03-2.000e-02j 1.53600e+01+5.500e+00j
  2.93000e+00+3.240e+00j 2.00000e+00+2.940e+00j 1.51000e+00+2.750e+00j]
 [1.30000e+00+9.600e-01j 1.02400e+01+3.670e+00j 2.38850e+03-2.000e-02j
  6.51000e+00+4.790e+00j 3.52000e+00+3.890e+00j 2.34000e+00+3.430e+00j]
 [3.30000e-01+4.900e-01j 1.17000e+00+1.300e+00j 3.91000e+00+2.880e+00j
  3.98084e+03-4.000e-02j 3.07200e+01+1.100e+01j 9.11000e+00+6.710e+00j]
 [2.20000e-01+3.900e-01j 6.70000e-01+9.800e-01j 1.76000e+00+1.950e+00j
  2.56000e+01+9.170e+00j 4.77700e+03-5.000e-02j 3.58400e+01+1.284e+01j]
 [1.50000e-01+3.300e-01j 4.30000e-01+7.900e-01j 1.00000e+00+1.470e+00j
  6.51000e+00+4.790e+00j 3.07200e+01+1.100e+01j 5.57317e+03-6.000e-02j]]
```

4.6
Matrix QB:

```
[0.  -0.j 0.01-0.j 0.05-0.j 0.05-0.j 0.01-0.j 0.  -0.j]
```

4.7

I assumed:

```
[0.   0.38 0.71 0.92 1.   0.92 0.71 0.38 0.  ]
```

4.8

I derived:

```
[ 0.+0.j -0.+0.j -0.+0.j -0.+0.j  1.+0.j -0.+0.j -0.+0.j -0.+0.j  0.+0.j]
```

```
[0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j]
```

5 plot

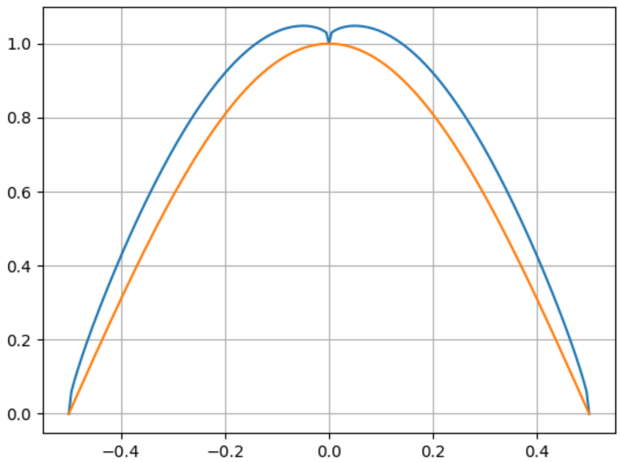
Our final equation is

```
MJ = QJ +QB
```

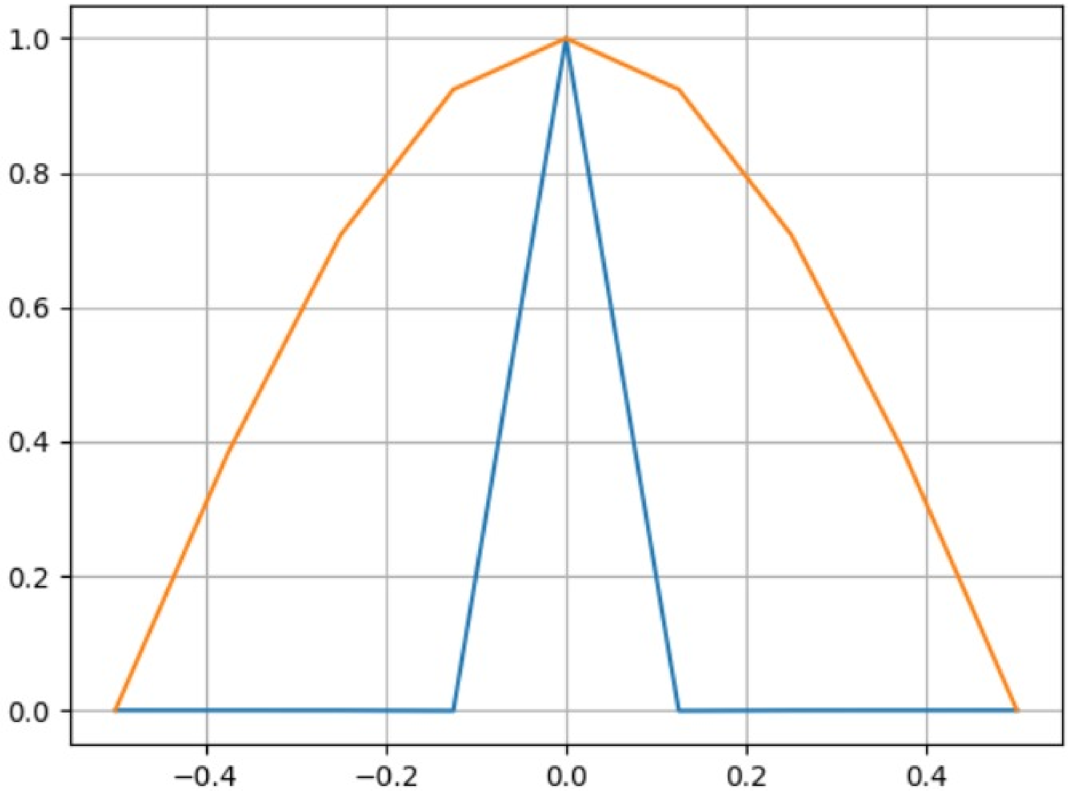
Invert (M-Q) and obtain J. Use inv(M-Q) in python. Add the Boundary currents (zero at i=0, i=2N, and Im at i=N). Then plot this current vs. z and also plot the equation assumed for current at the top of this question paper. The python code used is as follows

```
J = matmul(inv(Matrix(N, a) - Q) , Q_B) * Im
```

5.1 For N = 100



5.2 For N = 4



6 Conclusion :

- On increasing the value of N,the both graph will merge each other.
- On increasing N,the magnitude of point which are away from the centre are increasing.