ENDSEMESTER EE2703 - APPLIED PROGRAMMING LAB

SHAILESH PUPALWAR, EE20B100

May 12, 2022

1 Abstract

- To find the antenna currents in a half-wave dipole antenna using:

  (i) Standard Expression. (ii) Magnetic Vector Potential and approximating

- To study the difference between the graphs obtained via estimation and actual values

2 Introduction

We have a long wire carrying a current I(z) in dipole antenna with half length 0f 50cm(=l) so, wavelength = 2m. Next, we need to determine the currents in the two wires of the antenna. Next, we have the expressions to calculate the value of currents.

$$I = I_m sin(k(l - z))$$

$$0 \leq z \leq l$$

In the next process, we calculate the magnetic vector potential by approximating the integrals (in terms of summation); we next find out P_{ij} and P_B.

$$A_{z,i} = \sum_j P_{ij} I_j + P_B I_N = \sum_j I_j \left(\frac{\mu_0}{4\pi} \frac{exp(-jkR_{ij})}{R_{ij}} dz'_j\right)$$

$$P_B = \frac{\mu_0}{4\pi} \frac{exp(-jkR_{iN})}{R_{iN} dz'_j}$$

Then, we use the Ampere's circuital law to calculate H_ϕ. Again, we get it in terms of some summation involving the matrices Q_{ij} and Q_B.

$$H_\phi(r, z_i) = \sum_j Q_{ij} J_j + Q_{Bi} I_m = -\sum_j P_{ij} \frac{r}{\mu_0} \left(\frac{-jk}{R_{ij}} - \frac{1}{R_{ij}^2}\right) + P_B \frac{r}{\mu_0} \left(\frac{-jk}{R_{iN}} - \frac{1}{R_{iN}^2}\right)$$

At last we solve the matrix equation to find out the current vector J and then find out I.

$$MJ = QJ + Q_B I_m$$

3 Assignment questions

3.1 Question-1

According to the question, we now need to find vector z and u. And then find the current vectors I (at loactions of z) and J (at locations of u) respectively.

The following code snippet does the job!

```
# Defining all the given variables

import pylab
import numpy as np

pi = np.pi

N = 4  # Number of sections in each half section of the antenna
Im = 1.0  # Current injected into the antenna
len = 0.5  # Quarter wavelength
w_no = pi  # Wave number = (2*pi)/(lambda)
dz = len/N  # Spacing of current samples

z = np.linspace(-len,len,2*N+1) # Points at which we determine the currents.
u = np.delete(z,[0,N,2*N]) #2*(N-1) locations of unknown currents
a = 0.01 # Radius of wire
mu_0 = 4e-7*pi # Permeability of free space
```

```
The values obtained after running the code are:

z = [-0.5  -0.38 -0.25 -0.12  0.    0.12  0.25  0.38  0.5 ]
u = [-0.38 -0.25 -0.12  0.12  0.25  0.38]
```

3.2 Question-2

According to the question, we now need determine the M vector.I defined a function to determine M vector

The following code snippet does the job!

```
# Defining matrix M
```

```
def M(): # Function to determine and return the matrix M.
  M = (np.identity(2*(N-1)))*(1/(2*pi*a))
  return M
M = M()
```

```
The values obtained after running the code are:
M:  [[15.92  0.    0.    0.    0.    0.  ]
 [ 0.   15.92  0.    0.    0.    0.  ]
 [ 0.    0.   15.92  0.    0.    0.  ]
 [ 0.    0.    0.   15.92  0.    0.  ]
 [ 0.    0.    0.    0.   15.92  0.  ]
 [ 0.    0.    0.    0.    0.   15.92]]
```

3.3 Question-3

According to the question, we now need determine the matrices $R_z, R_u, R_in, PandPB$.

The following code snippet does the job!

```
# Computing vectors R_z, R_u and matrices PB, P

Z = np.meshgrid(z,z)
z_i, z_j = Z[0], Z[1]

#R_z determines distances from source and observer where source is point of wire.
R_z = np.sqrt((z_i-z_j)**2 + np.ones([2*N+1,2*N+1],dtype=complex)*(a**2))

U = np.meshgrid(u, u)
u_i, u_j = U[0], U[1]

#R_u determines distances from source and observer where observer is point where we want the
R_u = np.sqrt((u_i-u_j)**2 + np.ones([2*N-2,2*N-2],dtype=complex)*(a**2))

# Distances with respect to z = 0
R_in = np.delete(R_z[:][N],[N*2,0,N])  # Removing the three elements (first, middle, last)


# P is the contribution to the vector potential due to unknown currents
P = (mu_0/(4*pi))*(np.cos(w_no*R_u)-(np.sin(w_no*R_u))*1j)*(1/R_u)*(dz)

# PB is the contribution to the vector potential due to current "In"
PB = (mu_0/(4*pi))*(np.cos(w_no*R_in)-(np.sin(w_no*R_in))*1j)*(R_in)*(dz)
```

```
The values obtained after running the code are:
R_z =
[[0.01+0.j 0.13+0.j 0.25+0.j 0.38+0.j 0.5 +0.j 0.63+0.j 0.75+0.j 0.88+0.j
```

```
  1.  +0.j]
 [0.13+0.j 0.01+0.j 0.13+0.j 0.25+0.j 0.38+0.j 0.5 +0.j 0.63+0.j 0.75+0.j
  0.88+0.j]
 [0.25+0.j 0.13+0.j 0.01+0.j 0.13+0.j 0.25+0.j 0.38+0.j 0.5 +0.j 0.63+0.j
  0.75+0.j]
 [0.38+0.j 0.25+0.j 0.13+0.j 0.01+0.j 0.13+0.j 0.25+0.j 0.38+0.j 0.5 +0.j
  0.63+0.j]
 [0.5 +0.j 0.38+0.j 0.25+0.j 0.13+0.j 0.01+0.j 0.13+0.j 0.25+0.j 0.38+0.j
  0.5 +0.j]
 [0.63+0.j 0.5 +0.j 0.38+0.j 0.25+0.j 0.13+0.j 0.01+0.j 0.13+0.j 0.25+0.j
  0.38+0.j]
 [0.75+0.j 0.63+0.j 0.5 +0.j 0.38+0.j 0.25+0.j 0.13+0.j 0.01+0.j 0.13+0.j
  0.25+0.j]
 [0.88+0.j 0.75+0.j 0.63+0.j 0.5 +0.j 0.38+0.j 0.25+0.j 0.13+0.j 0.01+0.j
  0.13+0.j]
 [1.  +0.j 0.88+0.j 0.75+0.j 0.63+0.j 0.5 +0.j 0.38+0.j 0.25+0.j 0.13+0.j
  0.01+0.j]]

R_u =
[[0.01+0.j 0.13+0.j 0.25+0.j 0.5 +0.j 0.63+0.j 0.75+0.j]
 [0.13+0.j 0.01+0.j 0.13+0.j 0.38+0.j 0.5 +0.j 0.63+0.j]
 [0.25+0.j 0.13+0.j 0.01+0.j 0.25+0.j 0.38+0.j 0.5 +0.j]
 [0.5 +0.j 0.38+0.j 0.25+0.j 0.01+0.j 0.13+0.j 0.25+0.j]
 [0.63+0.j 0.5 +0.j 0.38+0.j 0.13+0.j 0.01+0.j 0.13+0.j]
 [0.75+0.j 0.63+0.j 0.5 +0.j 0.25+0.j 0.13+0.j 0.01+0.j]]

R_in =
[0.38+0.j 0.25+0.j 0.13+0.j 0.13+0.j 0.25+0.j 0.38+0.j]

P*1e8
[[124.94-3.93j   9.2 -3.83j   3.53-3.53j  -0.  -2.5j   -0.77-1.85j
   -1.18-1.18j]
 [  9.2 -3.83j 124.94-3.93j   9.2 -3.83j   1.27-3.08j  -0.  -2.5j
   -0.77-1.85j]
 [  3.53-3.53j   9.2 -3.83j 124.94-3.93j   3.53-3.53j   1.27-3.08j
   -0.  -2.5j ]
 [ -0.  -2.5j    1.27-3.08j   3.53-3.53j 124.94-3.93j   9.2 -3.83j
    3.53-3.53j]
 [ -0.77-1.85j  -0.  -2.5j    1.27-3.08j   9.2 -3.83j 124.94-3.93j
    9.2 -3.83j]
 [ -1.18-1.18j  -0.77-1.85j  -0.  -2.5j    3.53-3.53j   9.2 -3.83j
  124.94-3.93j]]

PB*1e8 =
[0.18-0.43j 0.22-0.22j 0.14-0.06j 0.14-0.06j 0.22-0.22j 0.18-0.43j]
```

3.4 Question-4

According to the question, we now need determine the matrices Q and QB.

The following code snippet does the job!

```
# Computing Q and QB.

# Matrix corresponding to unknown currents
Q = (a/mu_0)*(P)*((1j)*(w_no)+(1/R_u))*(1/R_u)

# Matrix corresponding to the boundary current
QB = (a/mu_0)*(PB)*((1j)*(w_no)+(1/R_in))*(1/R_in)
```

```
The values obtained after running the code are:

Q =
[[9.952e+01-0.j 5.000e-02-0.j 1.000e-02-0.j 0.000e+00-0.j 0.000e+00-0.j
  0.000e+00-0.j]
 [5.000e-02-0.j 9.952e+01-0.j 5.000e-02-0.j 0.000e+00-0.j 0.000e+00-0.j
  0.000e+00-0.j]
 [1.000e-02-0.j 5.000e-02-0.j 9.952e+01-0.j 1.000e-02-0.j 0.000e+00-0.j
  0.000e+00-0.j]
 [0.000e+00-0.j 0.000e+00-0.j 1.000e-02-0.j 9.952e+01-0.j 5.000e-02-0.j
  1.000e-02-0.j]
 [0.000e+00-0.j 0.000e+00-0.j 0.000e+00-0.j 5.000e-02-0.j 9.952e+01-0.j
  5.000e-02-0.j]
 [0.000e+00-0.j 0.000e+00-0.j 0.000e+00-0.j 1.000e-02-0.j 5.000e-02-0.j
  9.952e+01-0.j]]

QB = [0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j 0.-0.j]
```

3.5 Question-5

According to the question, we now need determine the current density vector J and currents I_esm, I_asm.I defined a function to determine M vector
The following code snippet does the job!

```
# Here we are computing Estimated currents and Assumed currents.

J = np.linalg.inv(M-Q)@QB*Im
#estimated currents
I_esm = np.concatenate(([0],J[:N-1],[Im],J[N-1:],[0]))
#assumed currents
I_asm = Im*np.sin(w_no*(len-abs(z)))
```

The values obtained after running the code are:

I_esm = [0. 0. 0. 0. 1. 0. 0. 0. 0.]

I_asm = [0. 0.38 0.71 0.92 1. 0.92 0.71 0.38 0.]

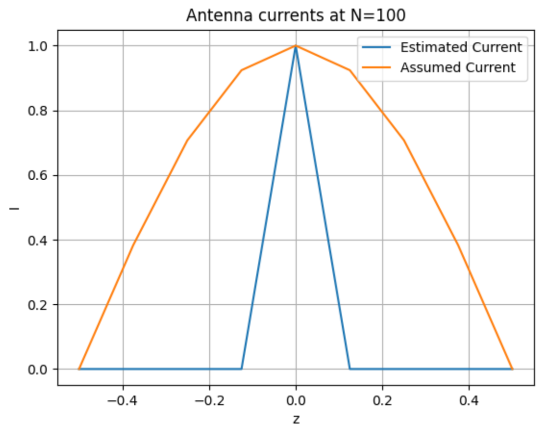

Figure 1: Plot of assumed current Vs estimated current

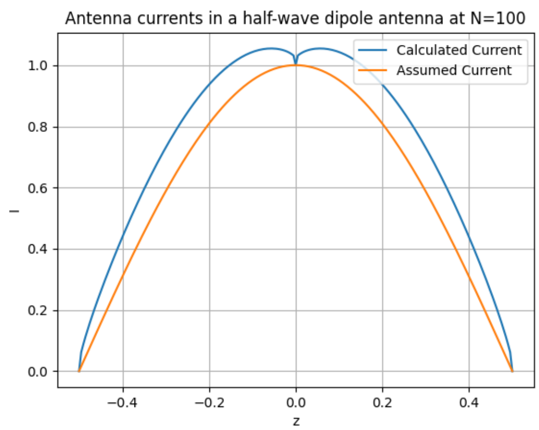

Figure 2: Plot of assumed current Vs estimated current

4 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.