EE2703 : Applied Programming Lab End Semester Exam : The Antenna Problem

Aditya Suresh
EE20B004

May 13, 2022

AIM

- To verify the standard assumption that current through antenna in given by
$$I = \begin{cases} I_m sin(k(l-z)), & \text{if } 0 \leq z \leq l \\ I_m sin(k(l+z)), & \text{if} -l \leq z < 0 \end{cases}$$
- To find Unknown current vector through combined use of Amperes law and Vector potential.
- To plot and compare the obtained current vector with the assumed sinusoidal response of the system.

Theory

- A long wire carries a current I(z) in a dipole antenna with half length of 50cm - so the antenna is a metre long, and has a wavelength of 2 metres. We want to determine the currents in the two wires of the antenna. The standard analysis assumes that the antenna current is given by
$$I = \begin{cases} I_m sin(k(l-z)), & \text{if } 0 \leq z \leq l \\ I_m sin(k(l+z)), & \text{if} -l \leq z < 0 \end{cases}$$
- We will use Ampere's law to obtain a Matrix equation between unknown current vector (J_i) and H_ϕ given by:
$$2\pi a H_\phi = J_i$$
Which when translated to matrix form looks like:
$$\begin{pmatrix} H_\Phi[z_1] \\ \cdots \\ H_\Phi[z_{N-1}] \\ H_\Phi[z_{N+1}] \\ \cdots \\ H_\Phi[z_{2n-1}] \end{pmatrix} = \frac{1}{2\pi a} \begin{pmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} J_1 \\ \cdots \\ J_{N-1} \\ J_{N+1} \\ \cdots \\ J_{2N-1} \end{pmatrix} = M{*}J$$
- The vector potential A(r,z) can be written as :
$$A(\vec{r}, z)\frac{\mu_0}{4\pi}\int \frac{I(z')\hat{z}e^{-jkR}\,dz'}{R}$$

This can be reduce to the following sum:

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

- Also

$$H_\phi(r,z) = -\frac{1}{\mu}\frac{\partial A_z}{\partial r}$$

Solving which we get

$$H_{phi}(r,z_i) = \sum_j Q_{ij}J_j + Q_{Bi}I_m$$

where

$$Q_{ij} = P_i j\frac{r}{\mu_0}(\frac{jk}{R_{ij}} + \frac{1}{R_{ij}^2})$$

and

$$Q_{Bi} = P_B\frac{r}{\mu_0}(\frac{jk}{R_{iN}} + \frac{1}{R_{iN}^2})$$

- Our final equation is :

$$MJ = QJ + Q_B I_m$$

i.e.,

$$(M-Q)J = Q_B I_m$$

We obtain $\vec{J}$ and hence $\vec{I}$ by substituting appropriate boundary conditions(zero at i=0, i=2N, and I_m at i=N). The current vector can be compared to the standard expression given at the top of the assignment.

Analysis and Plot

Divide the wire

1. We will divide the wire into pieces of length dz.Ideally we should number the pieces with indices going from N to +N. Unfortunately, Python does not allow negative array indices, so we will have an array with

indices going from 0 to 2N (2N + 1 elements, with element N being the feed of the antenna)

So define z by using :

$$z = i \times dz, \ -N \leq i \leq N$$

These are the points at which we compute the currents. The currents at end of the wire are zero, while the currents at z = 0 are prescribed by the circuit driving the antenna. So there are 2N +1 currents, with 2N 2 currents unknown (The end currents are known to be zero and current at the centre is given as I_m.) The 2N 2 locations of unknown currents are computed and kept in array u. The current vectors corresponding to z and u are computed and stored in current vector I and current vector J. The following code is used for the above implementation

```
z = np.linspace(-N,N,2*N+1)*dz
I = Im*np.sin(k*(l - abs(z)))
u = np.delete(z,[0,N if N%2 ==0 else N+1,2*N])
J = Im*np.sin( k*(l - abs(u)) )
```

Finding M matrix

1. We will define a function to find M matrix defined in theory section as $H_\phi = M * J$
2. From the theory section, it is obvious that M is scaled identity matrix of the order 2N-2 which can be obtained from the following code

```
def Matrix_M(N):
    return (1/(2*pi*a))*np.identity(2*N-2)
```

Computation of $A(\vec{r}, z)$

1. As discused in Theory section, we need to define 2 matrices P(2N-2,2N-2) and P_B(2N-1,1)

2. for computing P we need R_{ij} which is basically a (2N-2,2N-2) matrix containing distances between all possible points in u. This can be obtained using meshgrid as follows:

```
c1, c2 =  np.meshgrid(u,u)
Ru = np.sqrt((c1 - c2)**2 + a**2)
P = (mu0/(4*pi)) * ((np.exp(-1j*Ru*k))/ Ru) *dz
```

---

3. for computing P_B we need R_{iN} which is basically a (2N-2,1) matrix containing distances between all possible points in u from z=0 point. This can be obtained as follows:

```
RiN = np.sqrt(a**2 + u**2)
Pb = (mu0/(4*pi)) * (np.exp(-1j*RiN*k)) *dz/ RiN
```

Calculating $H_\phi(r, z)$

1. As discuused in theory section , to find $H_\phi(r, z)$ we need to find Q and Q_B

2. The formula to find Q and Q_B is straight forward :

```
Qbi = Pb*a/mu0*(1j*k/RiN + 1/RiN**2)
Q = P*a/mu0*(1j*k/Ru +1/Ru**2)
```

Calculating $\vec{J}$

1. All need to be done is to solve the Matrix equation $(M - Q)J = Q_B I_m$
   This can be easily done using the any of the following code:

```
1.J_new = np.linalg.solve(M-Q, Qbi*Im)
2.J_new = np.dot(inverse(M-Q), Qbi*Im)
3.J_new = np.linalg.lstsq(M-Q, Qbi*Im)[0]
```

Appending the boundary condition

1. We now need to add the 3 known currents which are 0(at -l and l) and I_m (at 0) to the above computed current vector to obtain the total current vecor I.

2. This is done by the following code which converts an array to list and uses insert function of list to append the 3 values at the needed loaction and converts it to an array befor returning

---

```
def Insert( J,val2, n1 =  N if N%2 ==0 else N+1 , val1 = 0,val3= 0):
    lst = J.tolist()
    lst.insert(0, val1)
    lst.insert(n1, val2)
    lst.append(val3)
    return np.array(lst)
```

Plot of Currents)

1. We will use matplotlib's plot function to plot the theoretical and computed value of Currents at various locations of the antenna.

```
plot(z,I, label = "True Current Vector")
plot(z,real(J_new), label = "Calculated Current vector")
grid(True)
legend()
show()
```

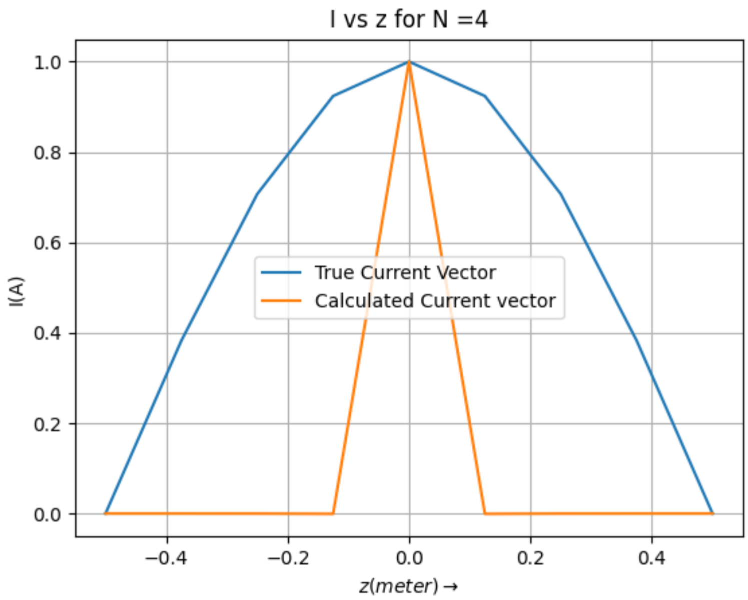

Figure 1: I vs z for N=4

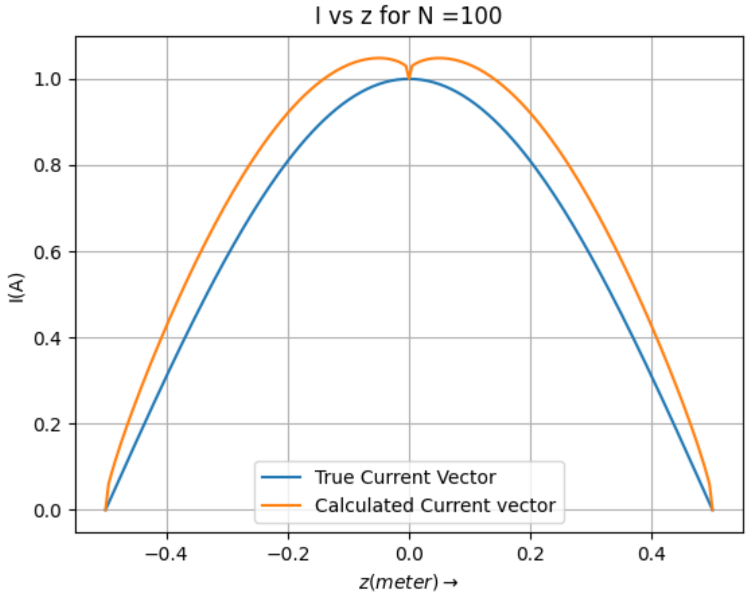

Figure 2: I vs z for N=100

Conclusions

Comparing the plots of $I_{theoretical}$ with $I_{calculated}$ we can clearly see a larger match when N =100. Comparatively N=4 gives an extremely poor match, This is quite an obvious observation as integration is summation as N tends to infinity. Since we have approximated integration's as a finite sum, the accuracy of summation would depend on how large the value of N is.

Printing Values for N=4

z = [-0.5 -0.375 -0.25 -0.125 0. 0.125 0.25 0.375 0.5]

u = [-0.375 -0.25 -0.125 0.125 0.25 0.375]

$I_{theoretical}$:
[[0.
[0.38]
[0.71]
[0.92]
[1.]
[0.92]
[0.71]
[0.38]

[0.]]

$J_{theoretical}$:
[[0.38]
[0.71]
[0.92]
[0.92]
[0.71]
[0.38]]

$J_{calculated} * 1e5$:
[[0.000e+00+0.j]
[-3.300e+00+1.06j]
[-9.550e+00+1.15j]
[-6.483e+01+1.21j]
[1.000e+05+0.j]
[-6.483e+01+1.21j]
[-9.550e+00+1.15j]
[-3.300e+00+1.06j]
[0.000e+00+0.j]]

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]]

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]]

$P*1e8$:
[[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]]

P_b*1e8:
[[1.27-3.08j]
[3.53-3.53j]
[9.2 -3.83j]
[9.2 -3.83j]
[3.53-3.53j]
[1.27-3.08j]]

$A*1e8$:
[[57.86-14.62j]
[104.78-16.63j]
[136.65-17.91j]
[136.65-17.91j]
[104.78-16.63j]
[57.86-14.62j]]

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

Hphi:
[[38.14-0.j]
[70.45-0.j]
[92.05-0.j]
[92.05-0.j]
[70.45-0.j]
[38.14-0.j]]