Assignment - 1

Roll No: 241110035
Name: Khushwant Kaswan

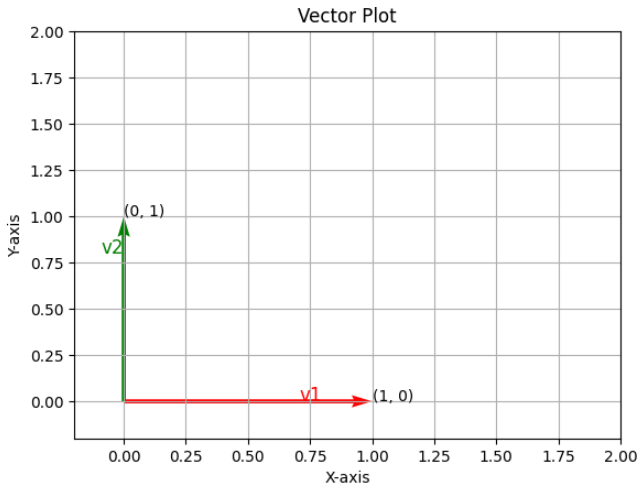
CS724: SENSING COMMUNICATIONS AND NETWORKING FOR SMART WIRELESS DEVICES

IIT Kanpur
MTech CSE

Question 1

1. $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} What\ is\ C(A)?$

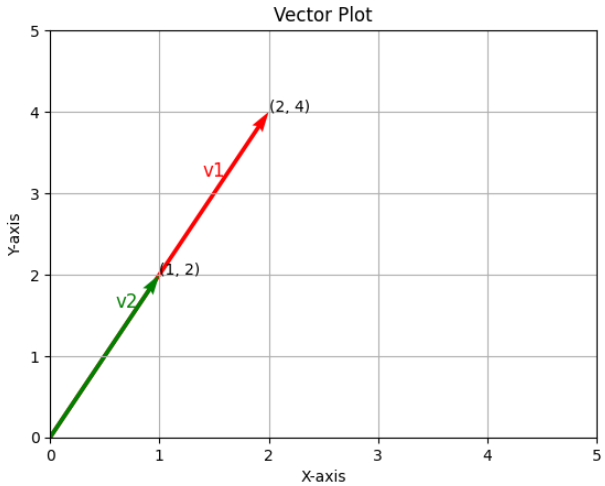
The plot of the column vectors (1,0) and (0,1) shows that v1 and v2 do not lie on the same line through the origin, ie. they are linearly independent. So, the column space of the matrix (formed by v1,v2) is the entire 2D plane ie R^2. $C(A) = 2$



**Code used to plot points included in .ipynb file attached*

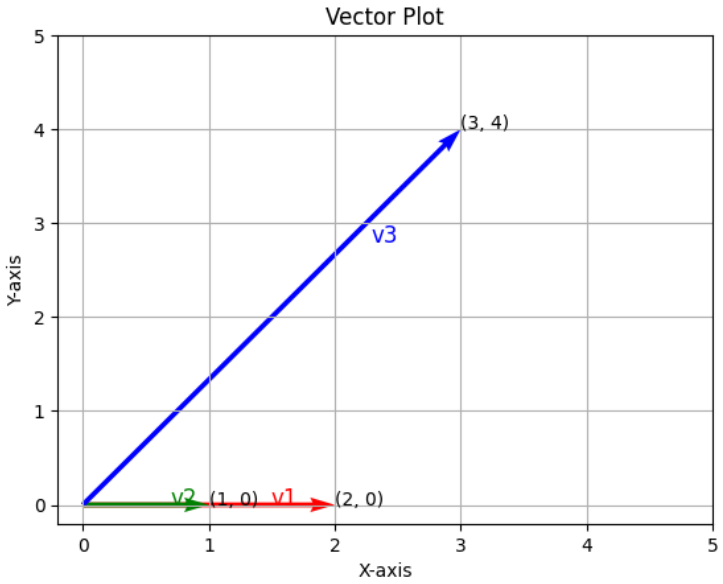
2. $A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} What\ is\ C(A)?$

The plot of the column vectors (1,2) and (2,4) shows that v1 and v2 lie on the same line through the origin, ie. they are linearly dependent. So, the column space of the matrix (formed by v1,v2) is the line on which these vectors lie (a one-dimensional subspace of R2). ie R^1. $C(A) = 1$



3. $A = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 4 \end{bmatrix}$ $What\ is\ C(A)?$

The plot of the column vectors (1,0), (2,0) and (3,4) shows that v1 and v2 lie on the same line through the origin, ie. they are linearly dependent.v3 lies on a completely different line (not collinear with v1 and v2), ie. v3 is not a linear combination of v1 and v2, so v3 is not in the same 1-dimensional subspace spanned by v1 and v2. So, the column space of the matrix (formed by v1,v2 and v3) is the entire 2D plane ie R^2. $C(A) = 2$



Question 2

1. **Fix the user at (100,100,100). Put 5 satellites at any random locations (you can manually put their locations), and fix their positions. Now calculate the time it takes for a signal to arrive from each one of these satellites to the user.**

Used random function to generate random x,y,z cordinates for 5 satellites.

```
x = [random.randrange(0, 10000) for _ in range(5)]
```

Used c=300000000 in m/s units

Sample Output :

```
User's Location : [100 100 100]
Location of satellite  1  : [ 650 , 5828 , 475 ]
Location of satellite  2  : [ 340 , 3419 , 1646 ]
Location of satellite  3  : [ 4935 , 8201 , 671 ]
Location of satellite  4  : [ 1999 , 5982 , 4484 ]
Location of satellite  5  : [ 7791 , 2665 , 1957 ]
```

```
Time taken by a signal to arrive from satellite 1 to the user is 1.9221836251744756e-05 seconds. ie. about 0.00001922 seconds
Time taken by a signal to arrive from satellite 2 to the user is 1.223087032426102e-05 seconds. ie. about 0.00001223 seconds
Time taken by a signal to arrive from satellite 3 to the user is 3.150475572565725e-05 seconds. ie. about 0.00003150 seconds
Time taken by a signal to arrive from satellite 4 to the user is 2.525944949694844e-05 seconds. ie. about 0.00002526 seconds
Time taken by a signal to arrive from satellite 5 to the user is 2.77246691193561e-05 seconds. ie. about 0.00002772 seconds
```

2. **Now lets do the opposite operation, i.e. use the satellite locations and the times to find out the location of the user. Check whether it is coming exactly as (100,100,100).**

Using the formula $X = (A^T A)^{-1} A^T B$:

```
X = np.matmul(np.linalg.inv(np.matmul(A.transpose() , A)) , np.matmul(A.transpose() , B))
```

Created a function that takes array of time and positions of all 5 satellites and returns the positions by solving the linear equation.

After solving the equations, we get values very close 100,100,100 ie. user's location.

Output :

```
Matrix A :
 [[ -620 -4818  2342]
 [ 9190  9564 -1950]
 [-5872 -4438  7626]
 [11584 -6634 -5054]]
Matrix B :
 [[-309600.        ]
 [1680400.        ]
 [-268400.00000001]
 [ -10399.99999999]]
Calculated user position= ( 100.0 , 99.99999999999972 , 99.99999999999895 )
Localization error: 1.0893340855055158e-12
```

3. **Now add some random errors with the times (you can use function likes “rand” in matlab). Check how much location inaccuracy it showing up.**

Added random error using the following code :

```
rand_error=np.random.rand()/10**8
t_new = [t + rand_error for t in time]
t_new=np.array(t_new)
X_new=get_user_pos(t_new,x,y,z)
```

```
Original Time :  [1.92218363e-05 1.22308703e-05 3.15047557e-05 2.52594495e-05
 2.77246691e-05]
New Time with added error :  [1.92296883e-05 1.22387224e-05 3.15126078e-05 2.52673015e-05
 2.77325211e-05]
Calculated user position= ( 98.51874849968773 , 98.38918166542142 , 99.25484929461041 )
Localization error: 2.3117290688864185
```

4. **Now increase the amount of the random errors with time, and check what is the effect of this change on the localization error. You can plot a graph on the amount of timing errors vs localization error to see the effect. You need to run the program multiple time and then can take the average localization errors).**

Generated an error amount list contaning 25 error amounts in increasing order.

```
error_list=[]
for i in range(25):
  error_list.append(0.00000001*i)
```

For each error amount we will be calculating localization error `tests` (15) times and then take average of all. With each iteration, a random change of timing is made within the current error amount limits and a localization error is calculated.

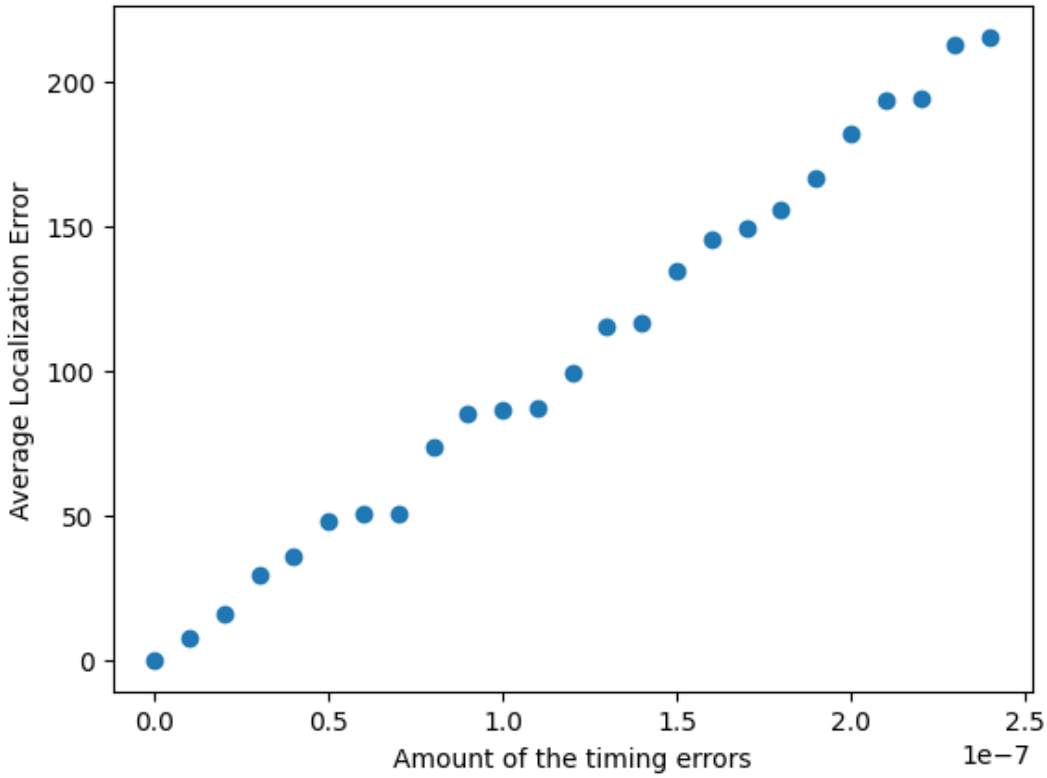
```
tests=15
for e in error_list:
 curr_loc_error=0
  for i in range(tests):
    time_plus_error=[t+np.random.normal(scale=e) for t in time]
    time_plus_error=np.array(time_plus_error)
```

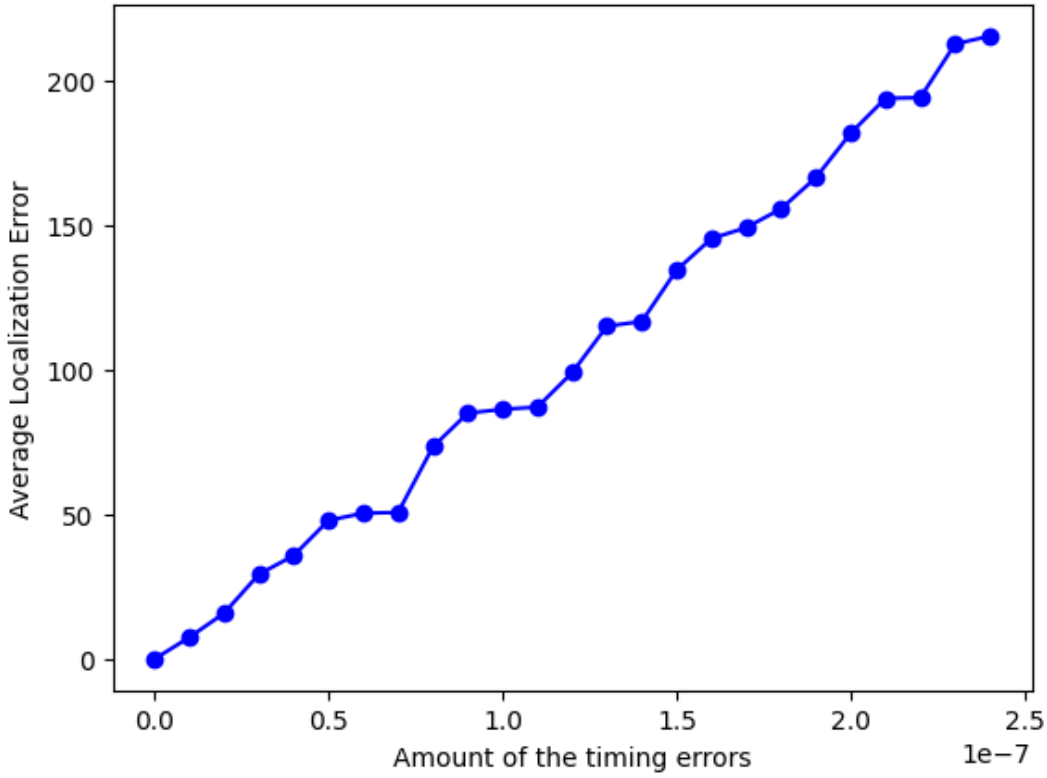
For each error amount, Localization error calculation is done `tests` (15) and then average of all those values is stored as localization error corresponding to the error amount and further used to plot the graph.

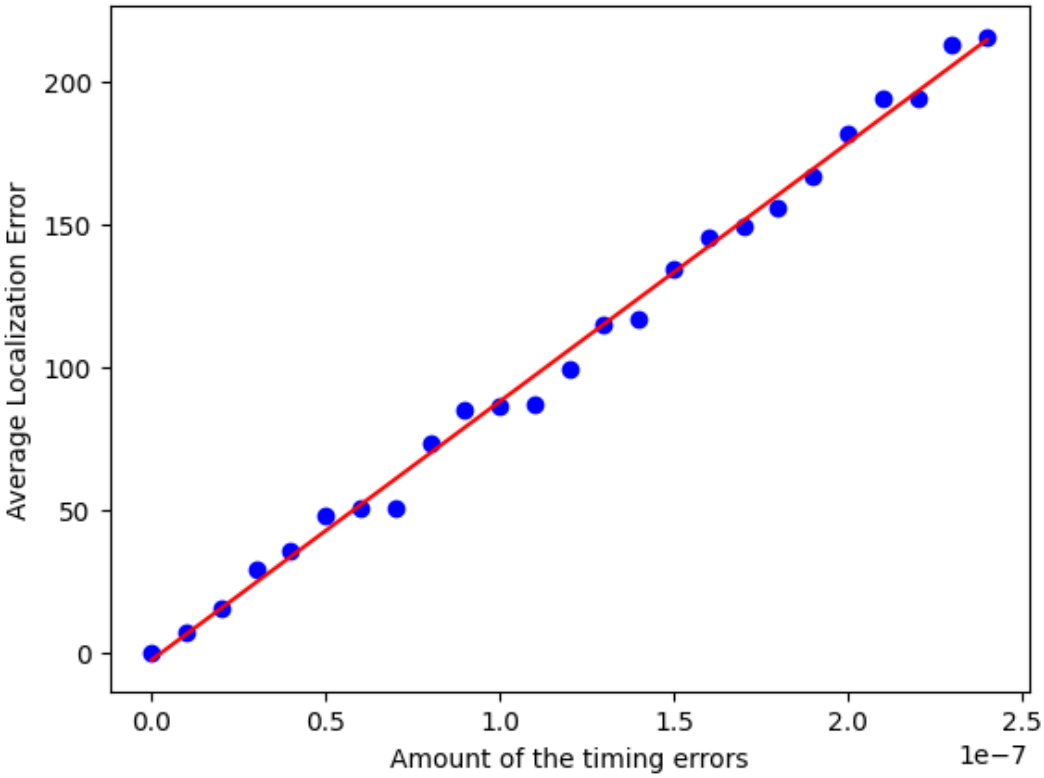
```
curr_avg_err=curr_loc_error/tests
avg_localization_errors.append(curr_avg_err)
```

```
Average Localization error for 0.00000000 amount of timing error : 0.00000000
Average Localization error for 0.00000001 amount of timing error : 7.48254557
Average Localization error for 0.00000002 amount of timing error : 15.83729041
Average Localization error for 0.00000003 amount of timing error : 29.29257178
Average Localization error for 0.00000004 amount of timing error : 35.76443730
Average Localization error for 0.00000005 amount of timing error : 50.66477982
Average Localization error for 0.00000006 amount of timing error : 47.93475917
Average Localization error for 0.00000007 amount of timing error : 50.41705516
Average Localization error for 0.00000008 amount of timing error : 73.34798094
Average Localization error for 0.00000009 amount of timing error : 86.33026626
Average Localization error for 0.00000010 amount of timing error : 87.08200717
Average Localization error for 0.00000011 amount of timing error : 84.99772888
Average Localization error for 0.00000012 amount of timing error : 98.99600843
Average Localization error for 0.00000013 amount of timing error : 115.06651627
Average Localization error for 0.00000014 amount of timing error : 134.46570932
Average Localization error for 0.00000015 amount of timing error : 116.68837469
Average Localization error for 0.00000016 amount of timing error : 155.62331254
Average Localization error for 0.00000017 amount of timing error : 149.11537606
Average Localization error for 0.00000018 amount of timing error : 145.31065613
Average Localization error for 0.00000019 amount of timing error : 166.37419388
Average Localization error for 0.00000020 amount of timing error : 181.74594114
Average Localization error for 0.00000021 amount of timing error : 193.80151517
Average Localization error for 0.00000022 amount of timing error : 212.58207010
Average Localization error for 0.00000023 amount of timing error : 194.04221509
Average Localization error for 0.00000024 amount of timing error : 215.32184292
```

Plots corresponding to above output :


Average Localization Error
200
150
100
50
0
0.0
0.5
1.0
1.5
2.0
2.5
Amount of the timing errors
1e−7


Average Localization Error
200
150
100
50
0
0.0
0.5
1.0
1.5
2.0
2.5
Amount of the timing errors
1e−7


Average Localization Error
200
150
100
50
0
0.0
0.5
1.0
1.5
2.0
2.5
Amount of the timing errors
1e−7