# *Homework 4 submission*

ECET 512 — Wireless Systems


**Chris Uzokwe**
**ID: cnu25**

*February 10, 2021*

# 1 Submitted files

For this assignment, besides this report, the following archives were created:

## 1.1 SRC Folder

+ *"rpattern.py"*: This **script** simulates the radiation pattern of an antenna. It plots the intensity vs orientation in 3D, and also displays the azimuth plane and elevation planes.

+ *"sectoredantennas.py"*: This **script** extends the one created in homework 3 that displayed random movement of users through a cell cluster. This script used sectorized cell clusters at 120 degrees, and displays the number of users in the sectorized and non sectorized systems.

## 1.2 DOC Folder

+ *"rhornpattern.png"* A 3D illustration of the horn antenna's radiation pattern.

+ *"rhornpattern-azumith.png"* A 2D representation of the horn antenna's radiation pattern, from the azimuth plane.

+ *"rhornpattern-elevation.png"* A 2D representation of the horn antenna's radiation pattern, from the elevation plane.

+ *"userpercell.png"* A graph which shows the number of users in each cell at each time step. Cells which do not have users are omitted.

+ *"userspercell-sectored120.png"* A graph which shows the number of users in each cell (with 120 degree sectoring) at each time step. Cells which do not have users are omitted.

## 2 Radiation Pattern of a Horn Antenna

*Below is the simulation of a radiation pattern of the horn antenna. This cell might be sectored by degrees from the center, since it exhibits a circular shape.*

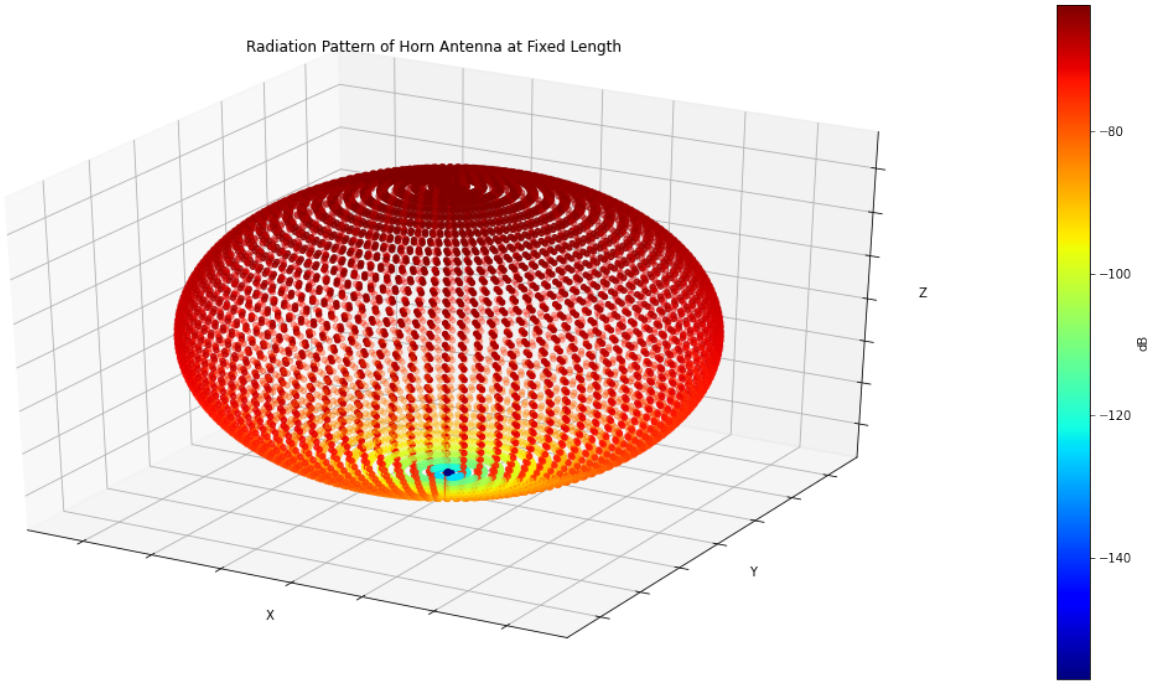


Figure 1: 3D view of the horn antenna radiation pattern.

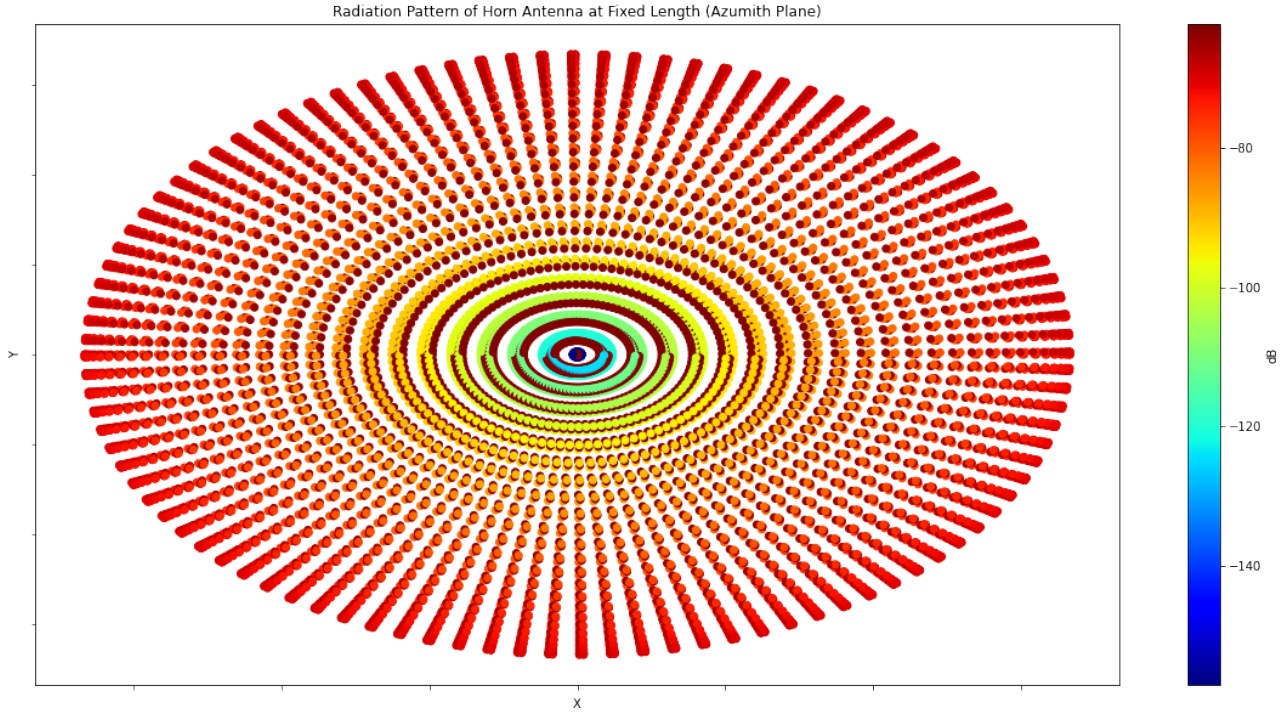


Figure 2: Horn antenna radiation pattern from the azimuth plane.

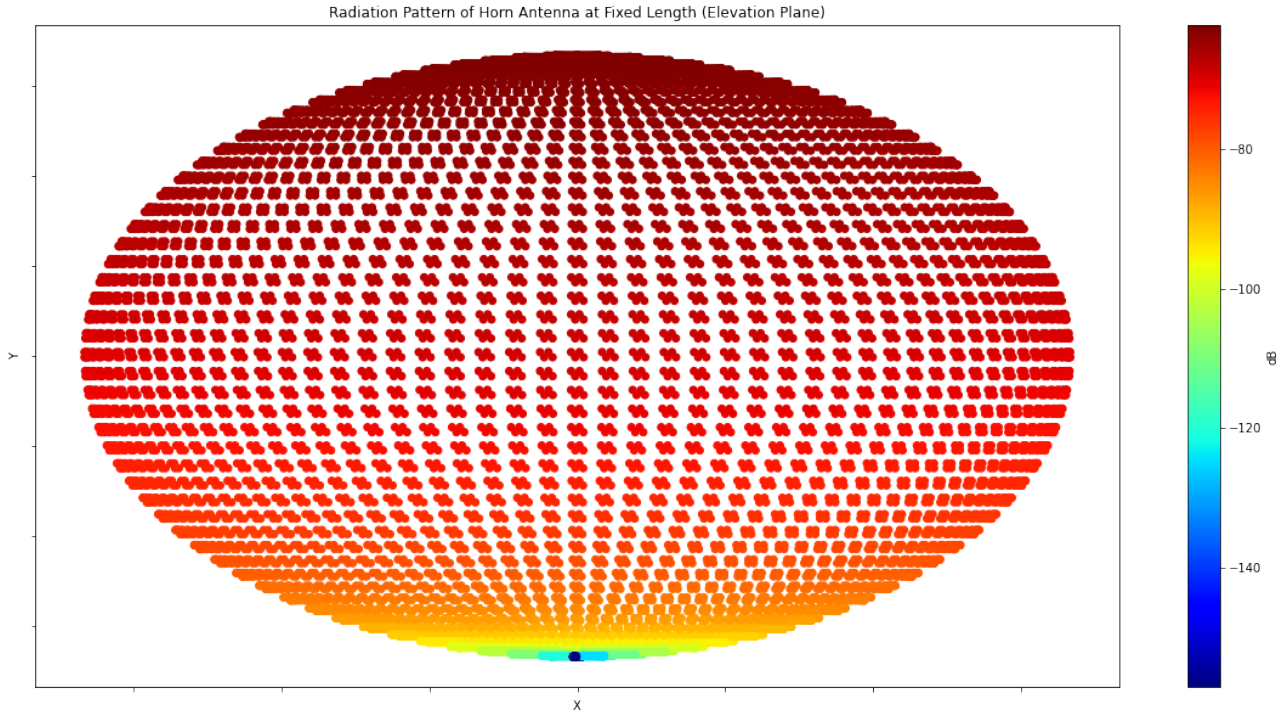


Figure 3: Horn antenna radiation pattern from the elevation plane.

# 3 Sectored and non-Sectored Cell Systems Count

*Demonstration of the multi user pathway propagation. We can see from the graph, that there exists points in the movement where certain cells are free for use, and trunking would allow the users to be handed off to different resources. This is where call blocking comes in, because as users are handed off to different cells, when resources are not available, the call may be dropped. We can see that in the system with sectoring, the trunking efficiency is decreased because there are more handoffs between cells. Still we have reduced the largest number at one point in a single frequency band.*

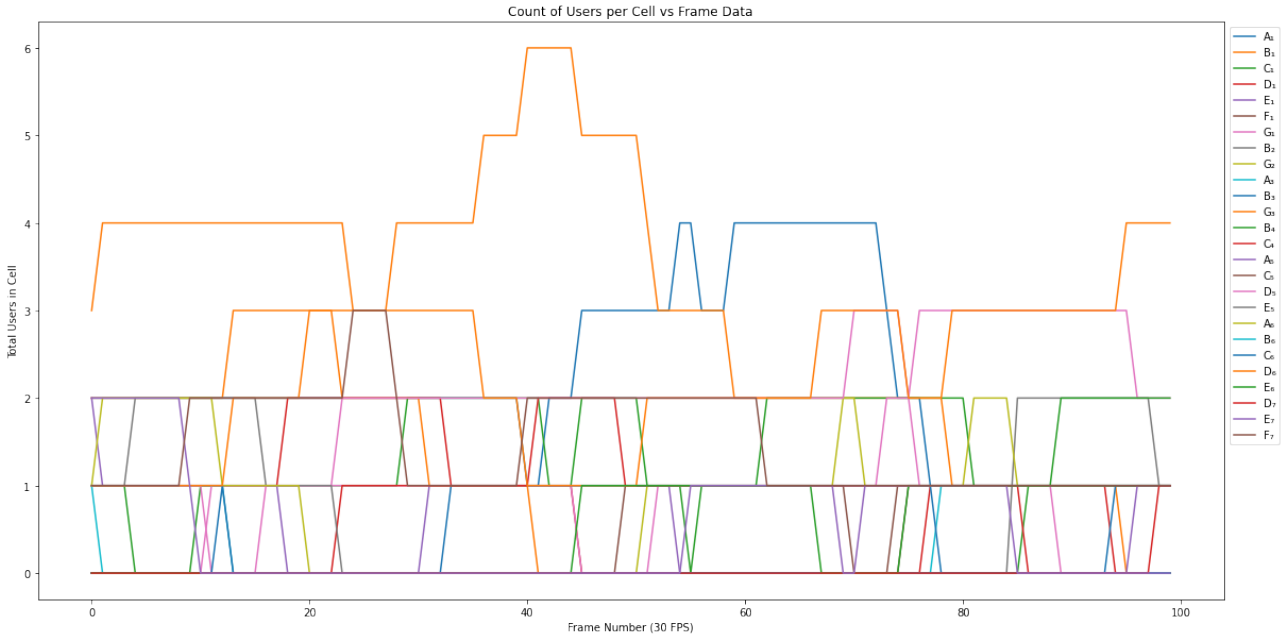


Figure 4: Count of users per cell as time increases.

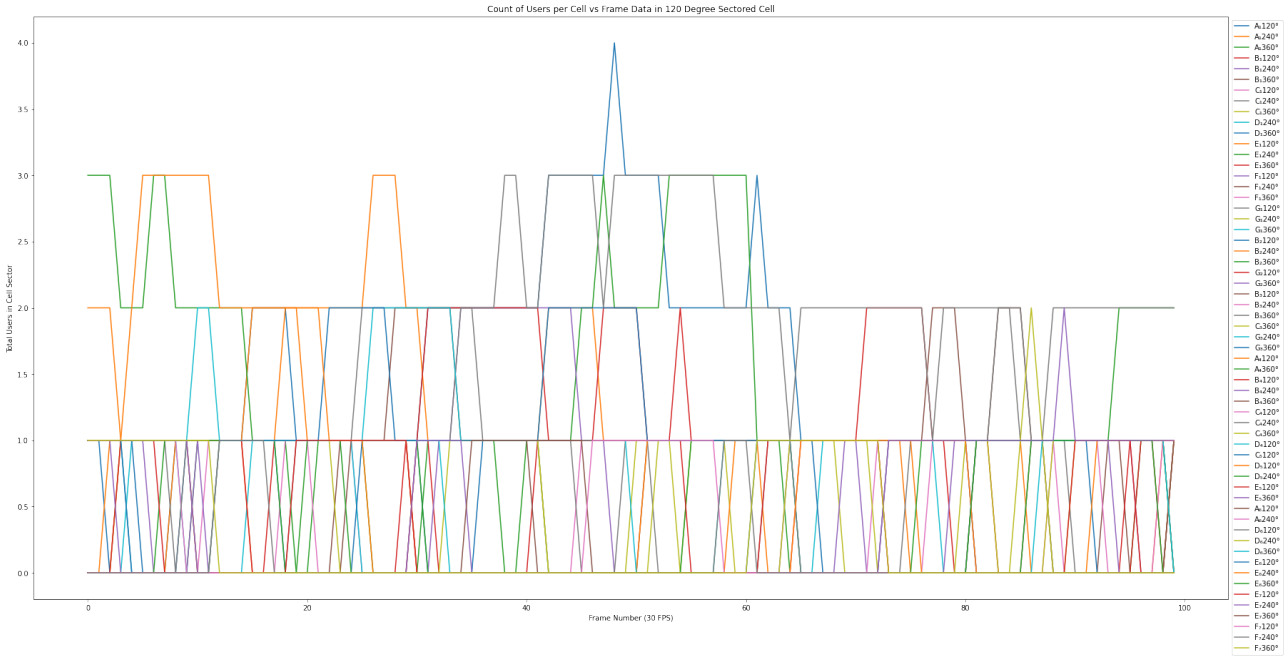


Figure 5: Count of users per cell as time increases. (120° sectoring)

# 4 Python code

```
[21] def rpattern(R, theta, phi):
       Eo = 0.8
       l = (3E8)/(900E6)
       r = 2*l*R
       n = 377

       A = 0.5*l
       B = 0.25*l
       a = 0.25*l
       b = 0.25*l

       k = (2*np.pi)/l

       Vx = A/l * np.sin(theta) * np.cos(phi)
       Vy = B/l * np.sin(theta) * np.sin(phi)

       F1 = 4/np.pi * (np.cos(np.pi*Vx)/(1-4*Vx**2))
       F0 = 2/np.pi * (np.sin(np.pi*Vy)/Vy)

       Etheta = 1j * (np.exp(-1*k*r*1j))/(l*r) * Eo * (A*B/4) * ((1+np.cos(theta))/2) * np.sin(phi) * F1 * F0
       Ephi = 1j * (np.exp(-1*k*r*1j))/(l*r) * Eo * (A*B/4) * ((1+np.cos(theta))/2) * np.cos(phi) * F1 * F0

       U = r**2 * (abs(Etheta)**2 + abs(Ephi)**2) / (2*n)
       return U
```

```
[22] theta = np.arange(0.0001, 2*np.pi, 0.063)
     phi = np.arange(0.0001, 2*np.pi, 0.063)
     r = 2*(3E8)/(900E6)

     theta, phi = np.meshgrid(theta, phi)

     X = r*np.sin(theta)*np.cos(phi)
     Y = r*np.sin(theta)*np.sin(phi)
     Z = r*np.cos(theta)

     r_intensity = [10*np.log10(rpattern(10, theta, phi)) for theta, phi in zip(theta, phi)]

     r_intensity = np.array(r_intensity)
     r_intensity.shape
```

Figure 6: Snippet of rpattern.py.

Plotting Users in Each Cell per Time Step

```
[5]  fig = plt.figure(figsize=(30, 15))

     for i in range(len(zippedcells)):

       frameTotal120 = []
       frameTotal240 = []
       frameTotal360 = []

       for j in range(numFrames):
         fc120 = 0
         fc240 = 0
         fc360 = 0

         for user, k in enumerate(indexs[j]):
           if k == i:

             label = sector([usersX[0:, user][j], usersY[0:, user][j]], zippedcells[i][0])

             if label == '120' + u"\N{DEGREE SIGN}":
               fc120 = fc120 + 1

             if label == '240' + u"\N{DEGREE SIGN}":
               fc240 = fc240 + 1

             if label == '360' + u"\N{DEGREE SIGN}":
               fc360 = fc360 + 1

         frameTotal120.append(fc120)
         frameTotal240.append(fc240)
         frameTotal360.append(fc360)

       if sum(frameTotal120) == 0:
         pass
       else:
         plt.plot(frameTotal120, label=zippedcells[i][1] + '120' + u"\N{DEGREE SIGN}")

       if sum(frameTotal240) == 0:
         pass
       else:
```

Figure 7: Snippet of sectoredantennas.py.