WIRELESS COMM NET (ECE 353s)
COURSE PROJECT REPORT

Simple Planning Tool for Service Provider

Spring ‘24

NAMES	ID	SECTION
Mostafa Alaa Abdelfattah	2001308	3
John Mamdouh Ramses	2001889	3
Ahmed Amr Ahmed	2000137	3
Abdelrahman Ali Mohamed	2000460	3

Submitted to:
Dr. Michael Naiem Ibrahim

The Report index and Contents:

This Report is structured to explain the code written required on matlab using algorithms seeking to design and validate a simple planning tool for a service provider. This is all done whilst satisfying the required points and showing project development through salient points arranged through this index.

1. PART 1 – Simple Planning Tool Design

- **1.1.** Analytical Design Process

The Steps required to design the planning tool was through some segmented functions which will be illustrated through mathematical equations arranged through wanted design parameters in following steps:

1. Calculating the Cluster Size and No. of Channels per Cell:

$$N = \frac{1}{3}[(I \times 10^{\frac{SIRMIN}{10}} + 1)^{\frac{1}{n}}]^2 \quad (1)$$

$$K = \frac{S}{N} \quad (2)$$

$$C = \frac{K}{J} \quad (3)$$

$$M = \frac{U}{S} \quad (4)$$

$$N = i^2 + ik + k^2 \quad (5)$$

Where:

- N is No. of cells per cluster.
- I is No. of interferers according to sectorization I = [1,2,6].
- S is No. of channels per cluster.
- K is No. of channels per cell.
- C is No. of channels per cell sector.
- J is No. of sectors per cells.
- U is No. of users in area.
- M is No. of clusters in area at no sectorization.

This method is illustrated in the function “clusterSize” in the code. In the previous laws, we extracted the no. of cells per cluster required from SIR_{min} given input from the GUI. After calculating the original cluster size, we ceil it the first applicable value according to $N = i^2 + ik + k^2$ as we know that N has specific values as 1,3,4… etc.

2. Traffic Intensity Per Cell and Sector:

$$GOS = \frac{\frac{A_{sector}{}^{C}}{C!}}{\sum_{m=0}^{C} \frac{A_{sector}^{m}}{m!}} \quad (6)$$

$$A_{cell} = A_{sector} \times J \quad (7)$$

Where:

- A_{sector} is traffic intensity in a sector.
- A_{cell} is Traffic Intensity per cell.

Using Erlang-B "Blocked Calls Cleared" model we found traffic intensity in sector and cell using the stated functions in code below [3:5].

3. Cell Radius Calculation:

$$U_{cell} = \frac{A}{A_u} \quad (8)$$

$$Area_{cell} = \frac{U}{u_{density}} = 1.5\sqrt{3}R^2 \quad (9)$$

$$No._{cells} = \frac{Area_{city}}{Area_{cell}} \quad (10)$$

Where:

- A is total traffic intensity in a cell.
- A_u is Traffic Intensity per user.
- R is radius of one cell.
- U_{cell} is users of one cell.

Using those equations the number of users in a cell is obtained. It is then used to obtain the total number of cells needed in a city and the cell's radius. All of this of course is done assuming hexagonal cells in "calculateParameters" function.

4. Base Station Transmitted Power and Received power vs Distance:

$$Path\ loss = 69.55 + 26.16\log(fc) - 13.82log10(hB) - CH + \big(44.9 - 6.55log10(hM)\big)log10(d)$$

(11)

$$CH = 0.8 + (1.1\log(fc) - 0.7)hM - 1.56\log10(fc)$$

(12)

$$MS_{power} = BS_{power} - Lu$$

(13)

Where:

- Lu is Path loss in dB.
- MS is received power in dB.
- BS is transmitted power in dB.
- Fc is band frequency in MHz.
- hB is height of transmitting antenna in meters.
- hM is height of receiving antenna in meters.
- CH is antenna height correction factor.
- d is distance between transmitting and receiving antenna in kilometers.

The Hata model is a widely used empirical model for estimating path loss in outdoor environments. It provides a basic framework for predicting the signal strength attenuation between a transmitter and receiver based on the distance between them. [1]

- **1.2.** Code Flow Explanation

The core flow is present inside a class named “cellPlanning” containing functions used to calculate traffic intensity, cluster size, and other cell parameters.

1. Explanation of cellPlanning.m:

1. “clusterSize” Function:
   Initially, the function calculates an original estimate of the cluster size using equation (1). This estimate is then adjusted upwards to ensure that the SIR remains above the specified minimum SIR, effectively avoiding performance degradation.

   Next, the function generates a grid of potential cluster sizes using the meshgrid function, where each point in the grid corresponds to a combination of two parameters (I and K) derived from a range of integers. These parameters are determined based on the square root of the original cluster size estimate. Subsequently, the function evaluates these grid points using equation (5), which computes theoretical values of cluster sizes.

   Finally, the function selects the smallest cluster size from the computed values that exceed the original estimate. This ensures that the chosen cluster size is the minimum size that still satisfies the imposed constraints.

2. “calculateParameters” Function:
   Givens are userDensity , city area and Au from inputs and A_cell from explained latter function of “A_cell”. Using the total traffic intensity in a cell divided by per user, the total No. of users per cell is obtained, floored of course to be in safe scenario. We then divide that by user density to see the area of one cell covering those no. of users and obtain cell radius whilst seeing number of cells needed in total to cover whole area of city inputted.

3. “calculateACell” Function:
   This Function is used to find the required traffic intensity for one cell using the procedure that was explained in part 1. Users can choose to solve between three different methods: custom implementation, fslove, and fzero. They will be explained below.

4. “calculateASectorCustom” Function:

This is a custom algorithm which is developed by us to calculate traffic intensity using inverse Erlang B formula and newton’s method of optimization. Here's how it works:

- Inverse Erlang B Calculation:

The algorithm starts by invoking the inverse Erlang B function, which takes a traffic intensity guess and the number of channels as inputs. It returns the calculated Grade of Service (GOS) and the gradient of the function. The GOS represents the probability of blocking for a given traffic load.

- Newton's Optimization Method:

With the initial GOS guess and gradient obtained from the inverse Erlang B function, the algorithm proceeds to calculate the error between the calculated GOS and the required GOS specified by the user. It then adjusts the traffic intensity guess using Newton's method of optimization, which iteratively refines the guess based on the gradient of the function until convergence is achieved. The convergence criteria are defined by a specified tolerance level.

- Benefits of Inverse Erlang B:

This algorithm employs the inverse Erlang B function due to its computational efficiency and support for large numbers of channels. Unlike the traditional Erlang B function, which may encounter computational challenges with large factorials and powers, the inverse Erlang B function remains robust and efficient.

- Equations Used

$$P(A,c) = \frac{A^c}{c!\sum_{k=0}^{c}\frac{A^k}{k!}} = \frac{1}{\sum_{k=0}^{c}\frac{c!}{k!}\left(\frac{1}{A}\right)^{c-k}} = \frac{1}{\sum_{k=0}^{c}\frac{c}{A}\frac{c-1}{A}\dots\frac{k+1}{A}}$$

(14)

By using the equation above, we won't encounter super large factorials anymore.

5. “calculateASectorFsolve” Function:

Uses the matlab function Fsolve and uses the inverse Erlang B function and its gradient explained above to iteratively search for the root of the objective function, representing the offered traffic intensity that satisfies the specified GOS requirement. It is better than Fzero because it uses the gradient to reach solution faster and can compute for large number of channels.

6. “calculateASectorFzero” Function:

Employs fzero to iteratively search for the root of the objective function, representing the offered traffic intensity that satisfies the specified GOS requirement.

2. Explanation of PartA .m:

The main function determines the position and dimensions of the GUI objects and create a dropdown menu to choose the design degree of freedoms. When the user chooses them and type it directly goes to call back function that delete the old objects and create a new one. When the user changes the value of one of the controllers it directly passes the values from all the controllers to call back function that perform the previous algorithms to compute the design parameters.

3. Explanation of PartB .m:

This part uses the core function to plot many curves to test the function and our understanding of the cell planning system.

4. Comparison between three different methods of calculating Traffic intensity:

- "calculateASectorCustom" Function:

The Fastest one, the lowest number of iterations and can compute traffic intensity even if the number of channels is very large.

- "calculateASectorFsolve" Function:

The 2nd fastest in terms of number of iterations and it can also compute traffic intensity even if the number of channels is very large.

- "calculateASectorFzero" Function:

The slowest and least efficient of the bunch, and it has limitations on the maximum number of channels it can solve for.

- Benchmark

```
Number of iterations (Custom Function): 6
Number of iterations (Fsolve Function): 8
Number of iterations (Fzero Function): 15
Traffic Intensity (Custom Function): 36.1086 Erlangs
Traffic Intensity (Fsolve Function): 36.1086 Erlangs
Traffic Intensity (Fzero  Function): 36.1086 Erlangs
>>
```

- **1.3.** Required Plots

1. **GUI Inputs:**

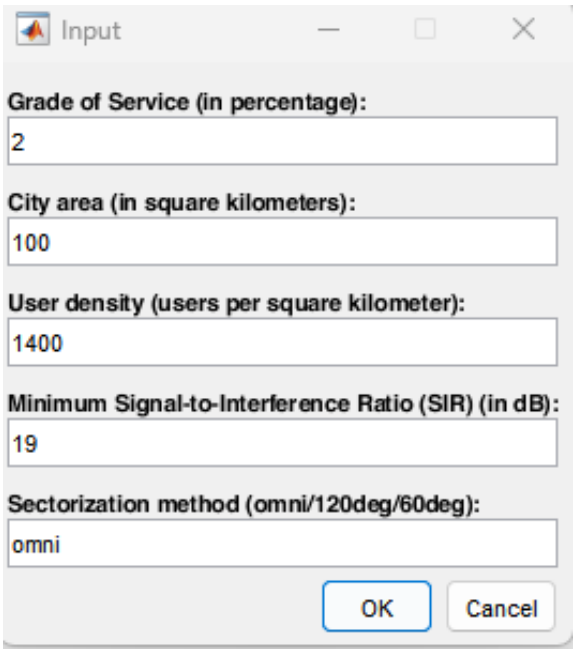


Figure 1. GUI Inputs

2. Part A Outputs:

```
No. of channels per cluster is 340 path loss exponent is  4 ,  user Traffic intensity is 25e-3
User Inputs:
Grade of Service: 2
City area: 100
User density: 1400
Minimum SIR: 19
Sectorization method: omni
Traffic Intensity (Custom Function): 20.1504 Erlangs
Traffic Intensity (Fzero  Function): 20.1504 Erlangs
-------------------------------------
Outputs:
1) Cluster Size: 12
2) Number of cells: 174 cells
3) Cell Radius: 0.47074 kilometer
4) Traffic intensity per cell: 20.1504 Erlang
5) Traffic intensity per sector: 20.1504 Erlang
6) Base station transmitted power: 21.9331 dBm
-------------------------------------
```

Figure 2. Part A Outputs

3. The MS received power in dBm versus the receiver distance from the BS:

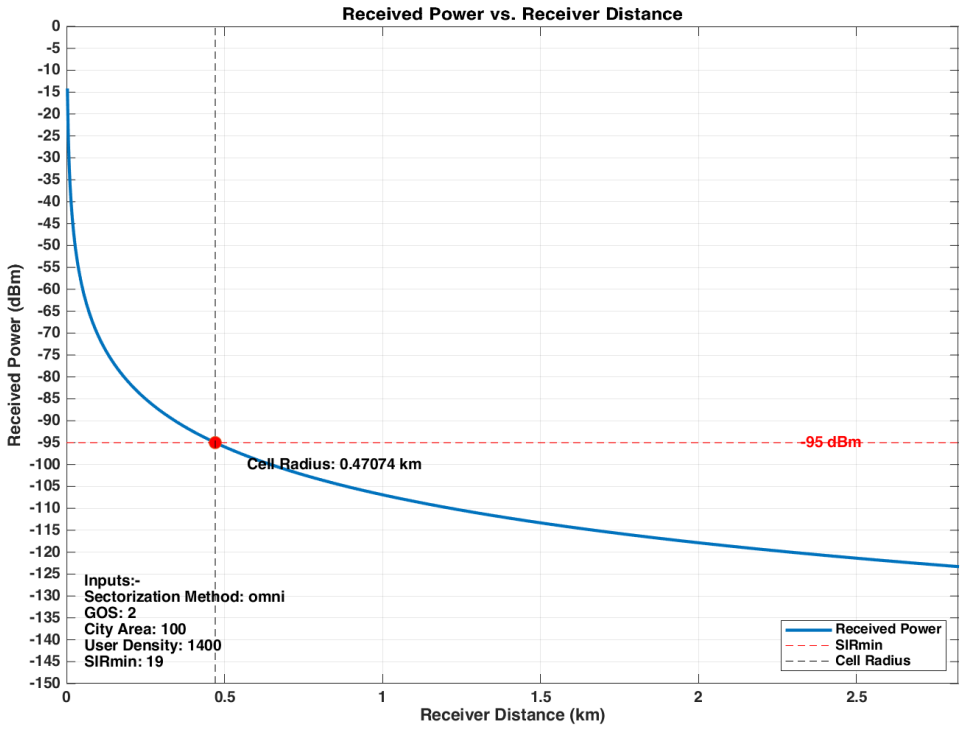


Figure 3. MS Power Vs Distance

Comment

As observed, following Hata model, The Power received decreases as going further from the Mobile station "cell" which is logical.

At distance equaling the cell radius, the power received = MS sensitivity = -95dBm as designed in the first place.

Note: The curve power at the start is highly sensitive to the step-in distance, increasing the step will get us different initial value.

2. PART 2 – Planning Tool Validation

- **2.1.** Required Plots

1. The cluster size versus $SIRmin$ with range from $1dB\ to\ 30\ dB$:

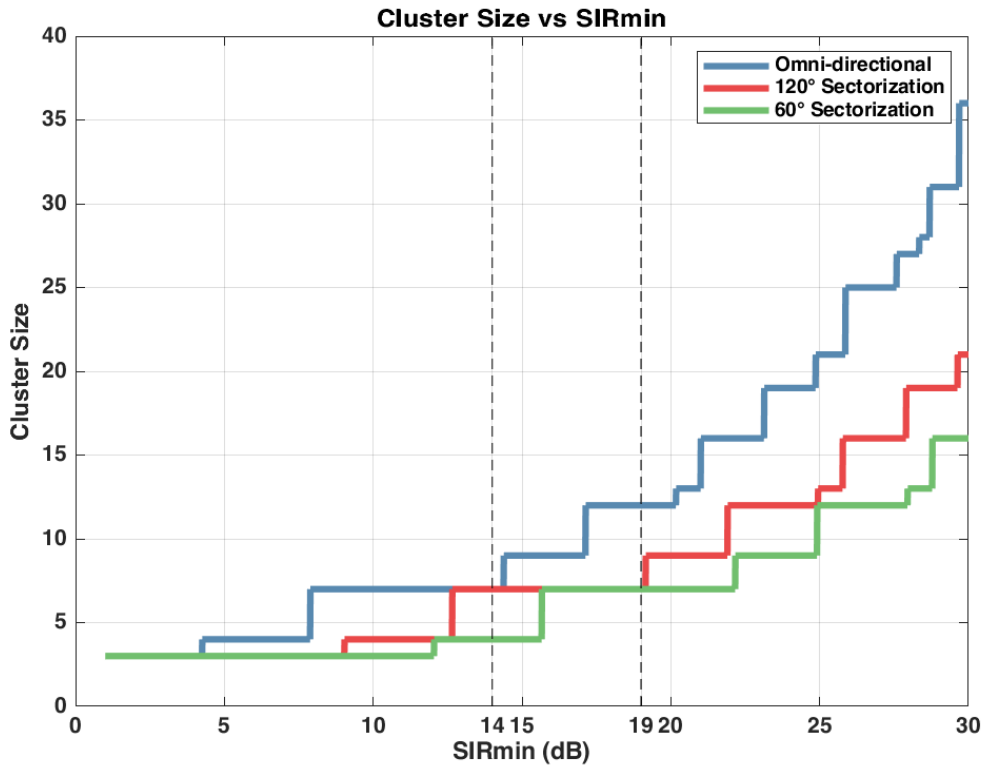


Figure 4. Cluster Size Vs SIRmin

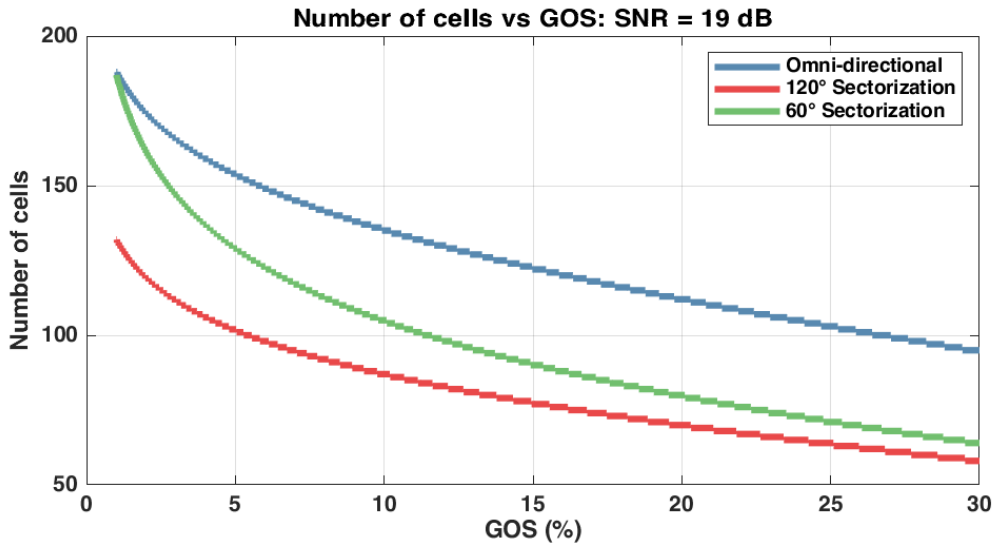
Comment

The cluster size (N) increases as the required SIR increases which highlights the effect of technology in lowering the SIR needed to decrease cluster size (N) leading to increase in the capacity of the network as we need more cells to cover the same area.

Also, increasing No. of sectors decreases cluster size needed for specific SIR with acceptable values by the constraints governed in equation 5.

We can notice that at SIR = 14 dB the 120-degree and Omni-directional sectorization gives us the same cluster size while at SIR = 19 dB, the 120° and 60° sectorization gives us the same cluster size.

2. For $SIRmin$ = 19dB & user density= 1400 $users/km2$:
    1. Plot the number of cells versus GOS (1% to 30%).
    2. Plot the traffic intensity per cell versus GOS (1%to 30%)



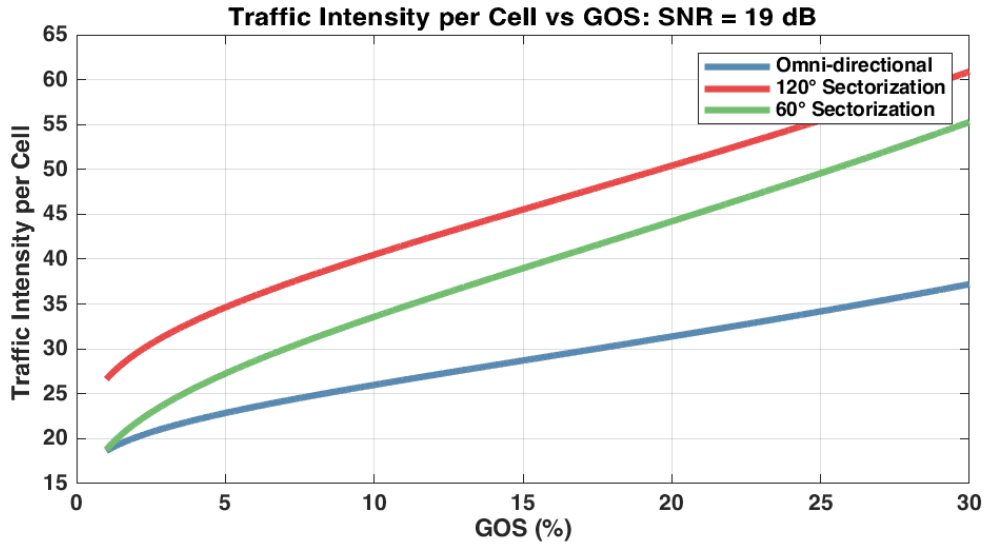


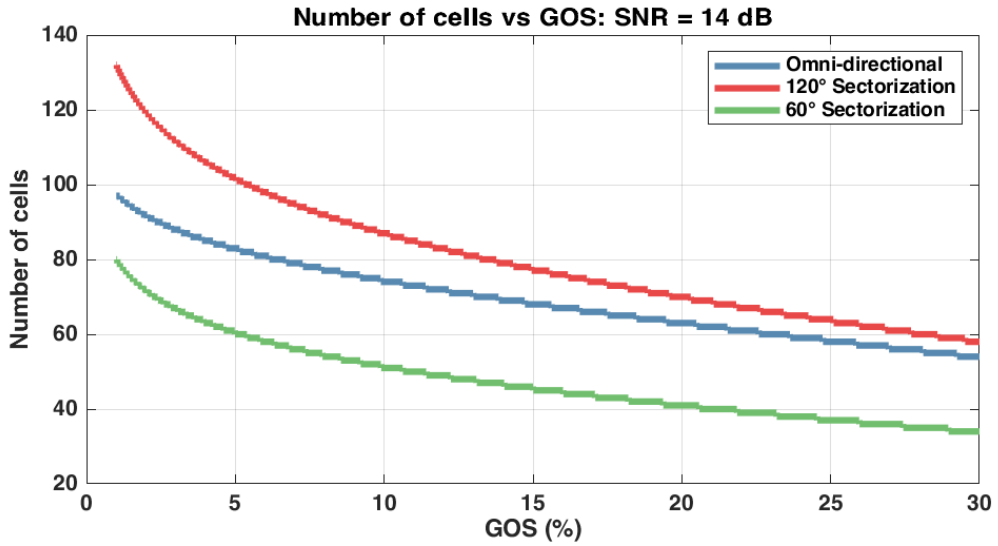
Figure 5. N,A_Cell Vs GOS 1

Comment

As GOS requirements increase, traffic intensity rises while the number of cells decreases. This is because a lower required blocking probability results in higher traffic intensity. Comparing cluster sizes against the minimum SIR curve, both 120° and 60° sectorization exhibit the same cluster size at SIR = 19dB. However, 60° sectorization has fewer channels per sector compared to 120° sectorization. Consequently, 60° sectorization exhibits lower traffic intensity. Thus, at a specific GOS, 60° sectorization requires more cells to accommodate a specific user density compared to 120° sectorization. Additionally, omnidirectional sectorization has the largest cluster size, resulting in the lowest number of channels, lowest traffic intensity, and the greatest number of required cells. In summary, two factors influence channels per sector:

- Cluster size: Decreasing cluster size increases channels per sector.
- Number of sectors: Increasing the number of sectors decreases channels per sector.
- The dominant effect determines the outcome.

3. At $SIRmin$ = 14dB & user density= 1400 $users/km2$
    1. Plot the number of cells versus GOS (1% to 30%).
    2. Plot the traffic intensity per cell versus GOS (1% to 30%).



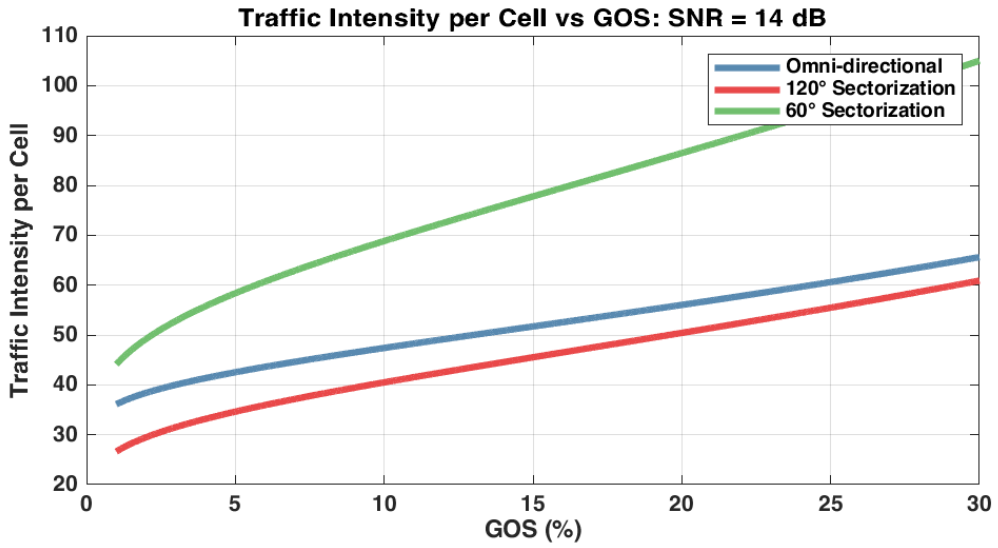


Figure 6. N,A_Cell Vs GOS 2

Comment

This plot has the same trend as the point before but has significant change in which sectorization method has the highest traffic intensity or lowest number of cells. Going to the cluster size vs SIR curves, we can see that at SIR = 14 dB the 120-degree and Omni-directional sectorization gives us the same cluster size

60° sectorization has higher channels per sector compared to 120° sectorization as it has lower cluster size which dominates the effect of decreasing the channels due to sectorization. Consequently, 60° sectorization exhibits higher traffic intensity. Additionally, omnidirectional sectorization has the same cluster size as 120° sectorization, resulting in the lowest number of channels for the 120° sectorization, lowest traffic intensity, and the greatest number of required cells.

Compare:

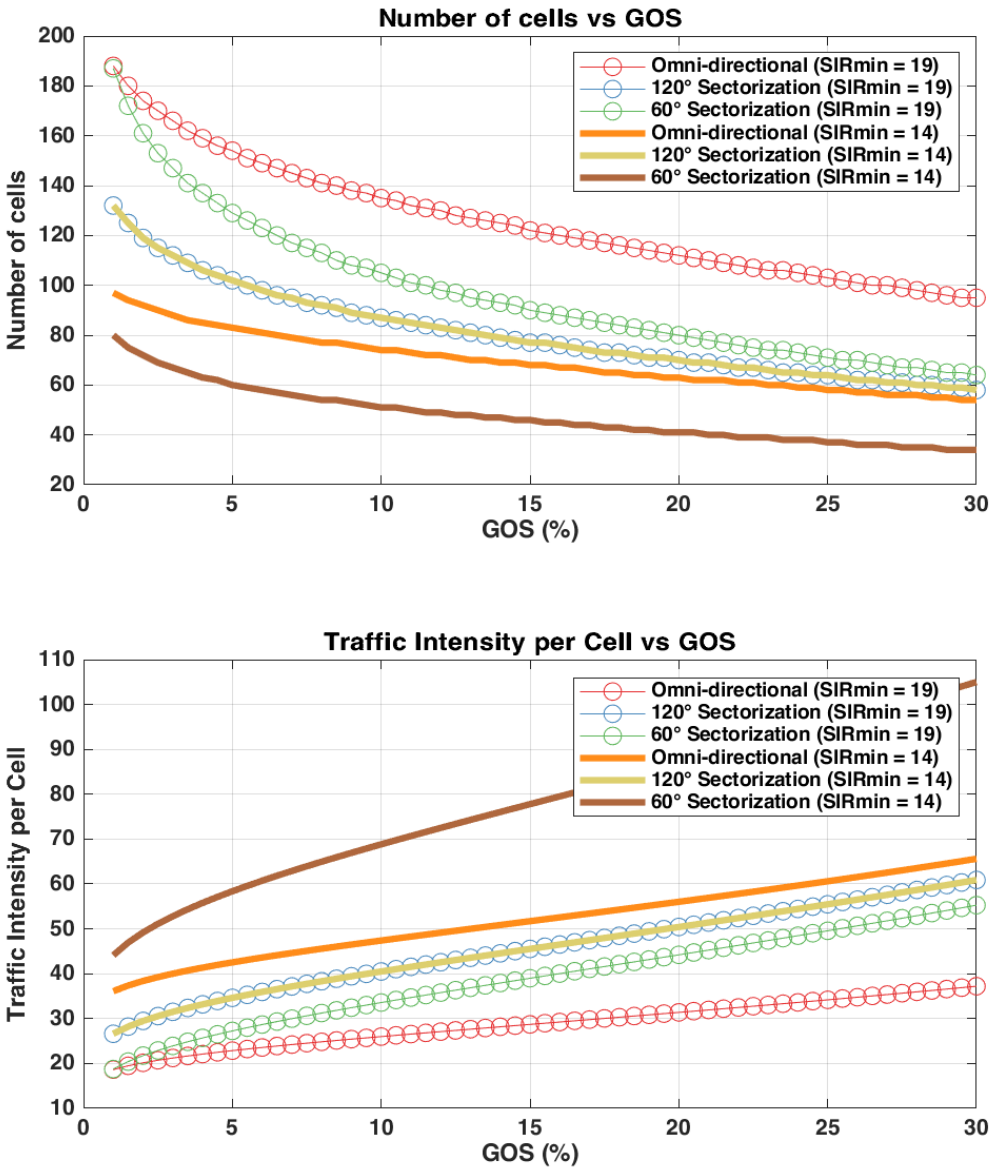


Figure 7. N,A_Cell Vs GOS CMP

Comparing the curves at different SIR, we notice that generally higher SIR result in lower traffic intensity as generally the cluster size increases which decreases the number of channels per sector. The exception here is 120° sectorization which has the same curve for both SIR = 14 and SIR = 19 which is logical as the cluster size doesn't change in this region as highlighted by figure 4 and of course it has the same number of sectors.

4. At $SIRmin$ = 14dB & GOS= 2%,

    1. Plot the number of cells versus user density (100 to 2000 $users/km2$):

    2. Plot the cell radius versus user density (100 to 2000 $users/km2$):

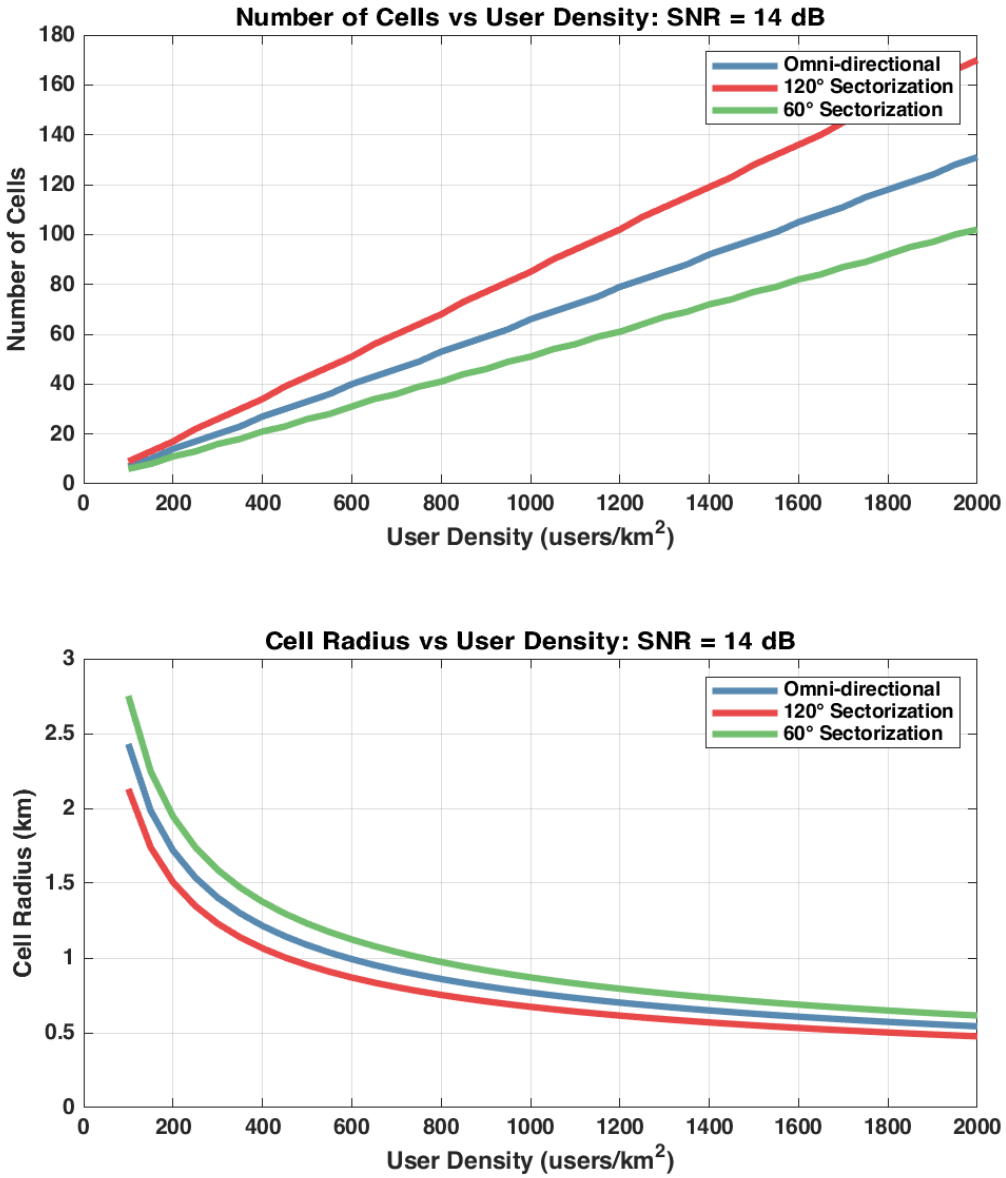


Figure 8. N,Cel_Radl Vs u_density 1

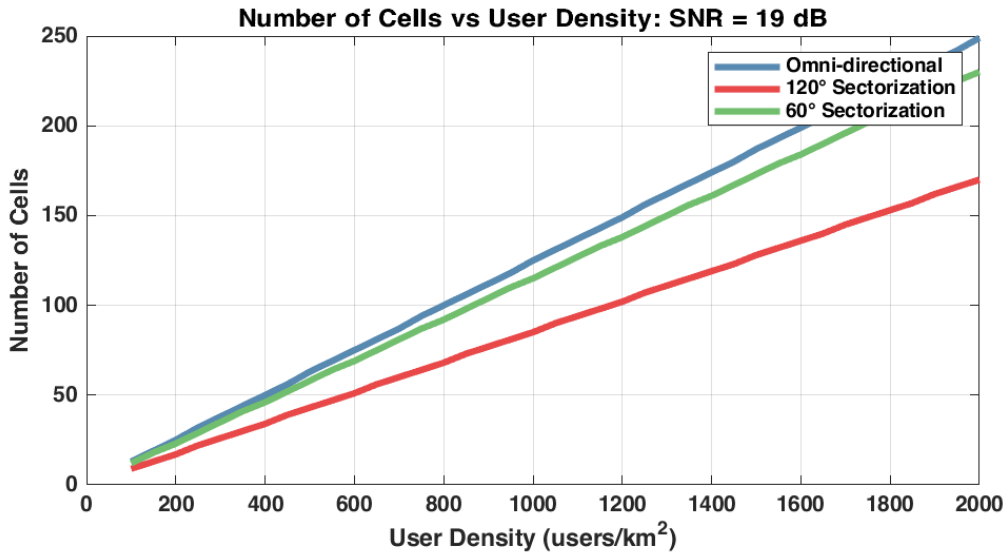
Comment

As user density increases, we need more cells to serve all users as our channels are limited by interference. Increasing the number of cells in a constant geographical area means that the area of each cell should decrease so cell radius decreases as user density increases.

We can notice that 60° sectorization has the highest user density for specific number of cells then omni directional then 120° sectorization. This is due to the change in cluster size, where 60° sectorization has the lowest cluster size so it has highest traffic intensity so it can serve more users. Again, all this is affected by the relation in figure 4.

5. At $SIRmin$ = 19dB & GOS= 2%,

    1. Plot the number of cells versus user density (100 to 2000 $users/km2$):
    2. Plot the cell radius versus user density (100 to 2000 $users/km2$):



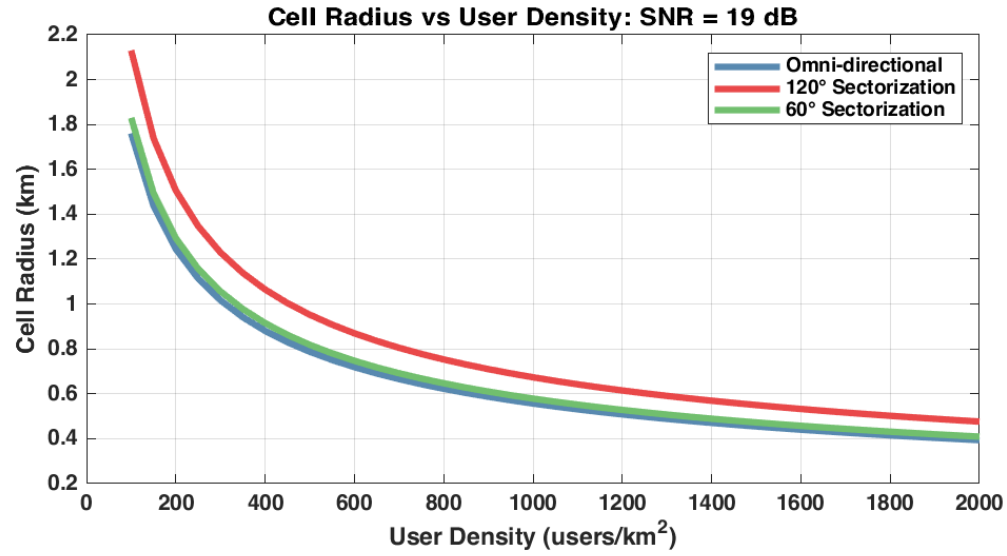


Figure 9. N,Cel_Radl Vs u_density 2

Comment

We can notice that 60° sectorization doesn't have the highest user density unlike the last part and 120° sectorization provides higher user density at constant number of cells. This is because 120° sectorization provides us with the same cluster size with more channels/sector so higher capacity.

This leads to 120° sectorization having the highest cell area and cell radius as it can accommodate the highest number of users and have the highest traffic intensity.

Again, all this is affected by the relation in figure 4.

Compare:

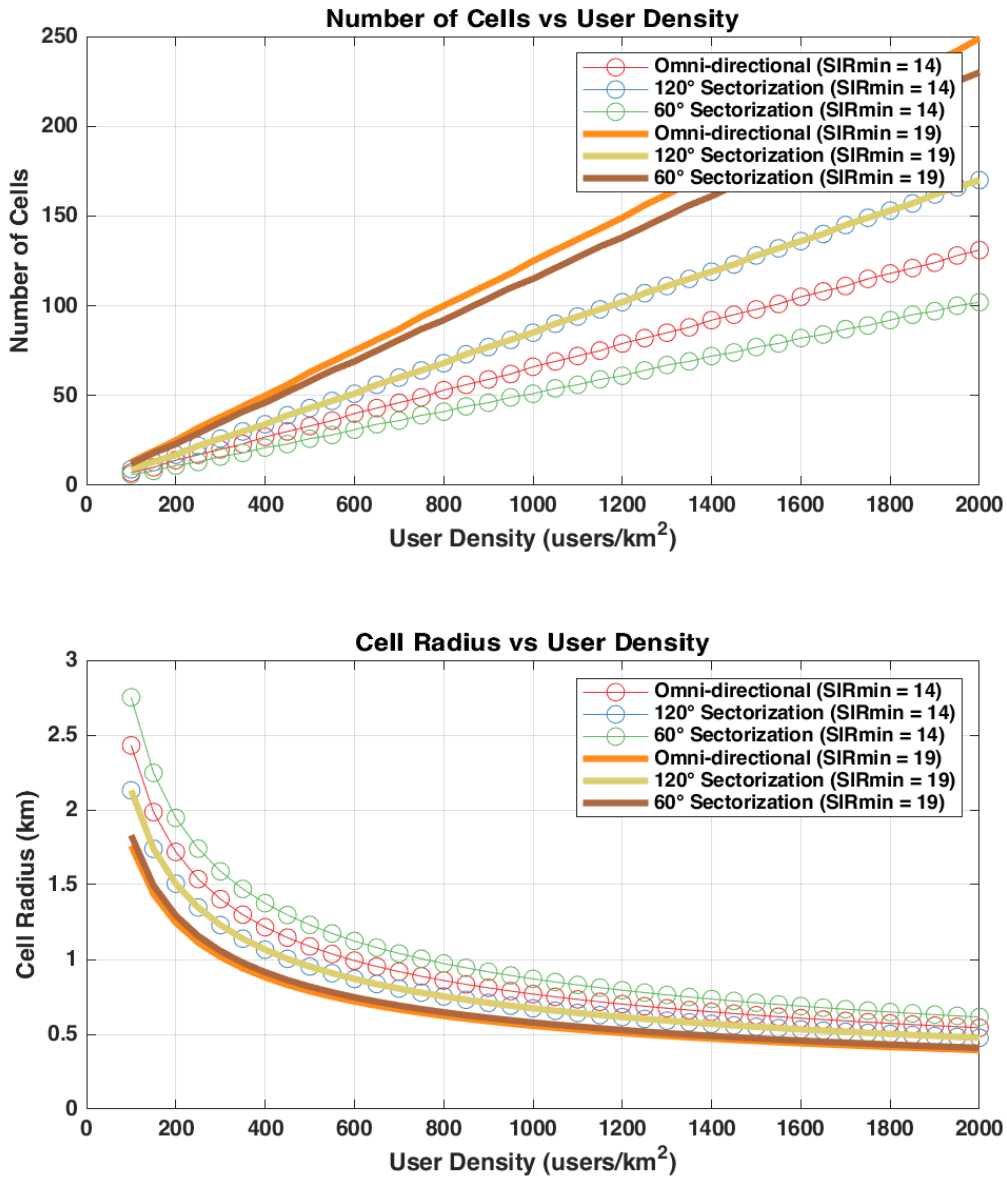
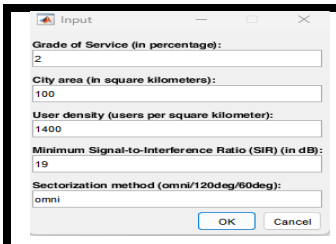
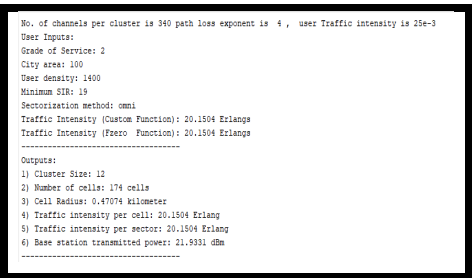
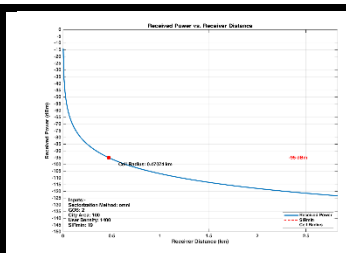
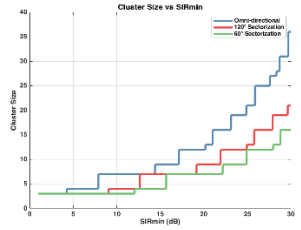
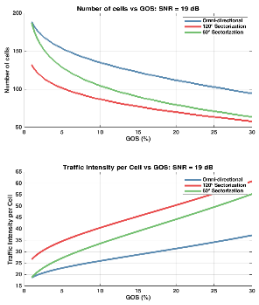
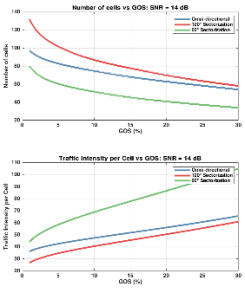
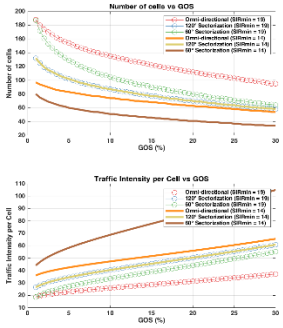
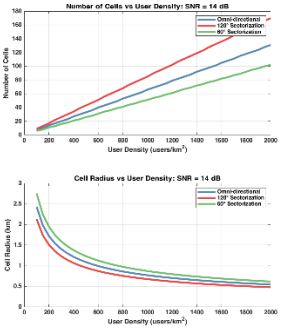
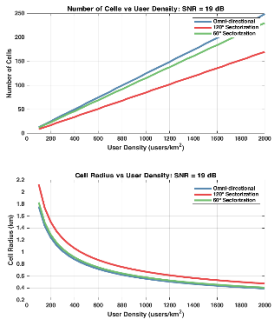
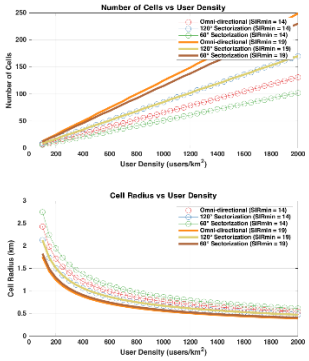


Figure 10. N,Cel_Radl Vs u_density CMP

As SIR increases from “14dB to 19dB” Number of cells increases as explained before. Also, as SIR increases from “14dB to 19dB” Number of cells increases so to serve the same area we should reduce the area of the cell by reducing the radius.

3. PART 3 – Appendix

- **3.1.** Table of Figures

<table>
<tr>
<td> 
</td>
<td> 
</td>
<td> 
</td>
<td> 
</td>
</tr>
<tr><td>Fig.1</td><td>Fig.2</td><td>Fig.3</td><td>Fig.4</td></tr>
<tr><td>GUI inputs</td><td>Part A Outputs</td><td>MS Power Vs Distance</td><td>Cluster Size Vs SIRmin</td></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td> 
</td>
<td> 
</td>
</tr>
<tr><td>Fig.5</td><td>Fig.6</td><td>Fig.7</td><td>Fig.8</td></tr>
<tr><td>N,A_Cell Vs GOS @SIR 19 dB</td><td>N,A_Cell Vs GOS @SIR 14 dB</td><td>N,A_Cell Vs GOS @SIR 19,14 dB</td><td>N,Cel_Radl Vs u_density @SIR 14 dB</td></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td></td>
<td></td>
</tr>
<tr><td>Fig.9</td><td>Fig.10</td><td></td><td></td></tr>
<tr><td>N,Cel_Radl Vs u_density @SIR 19 dB</td><td>N,Cel_Radl Vs u_density @SIR 14,19 dB</td><td></td><td></td></tr>
</table>

- **3.2.** References Used

[1]: Signal Propagation and Path Loss Models - Stanford University

- **3.3.** Table of equations

[1:5]: Calculating the Cluster Size and No. of Channels per Cell.

[6:8]: Cell Radius Calculation.

[9:10]: Traffic Intensity Per Cell and Sector:

[11:13]: Base Station Transmitted Power and Received power vs Distance:

[14]: Inverse Erlang B function:

4. PART 4 – Essential Additions

- **4.1.** Additional Deliverables

Link to the files: Link to the Files