# Field Normalization

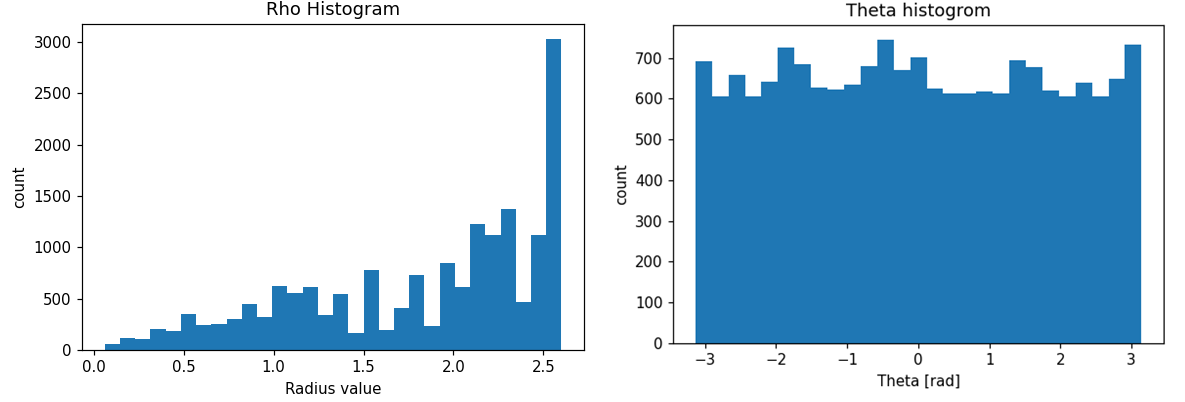
I loaded the data we received from Andrea. The data is of the format:

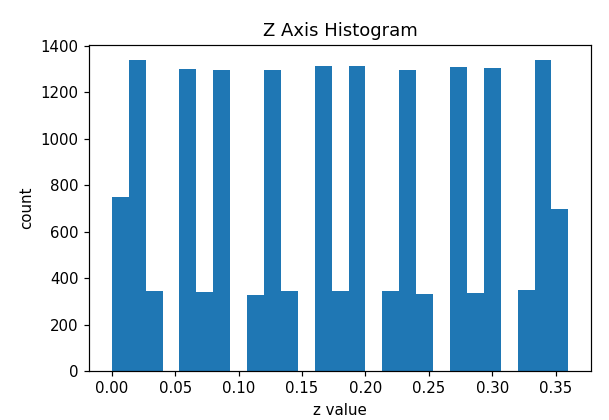
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X\_1 | Y\_1 | Z\_1 | Ex\_1 | Ey\_1 | Ez\_1 |
|  |  |  |  |  |  |
| X\_N | Y\_N | Z\_N | Ex\_N | Ey\_N | Ez\_N |

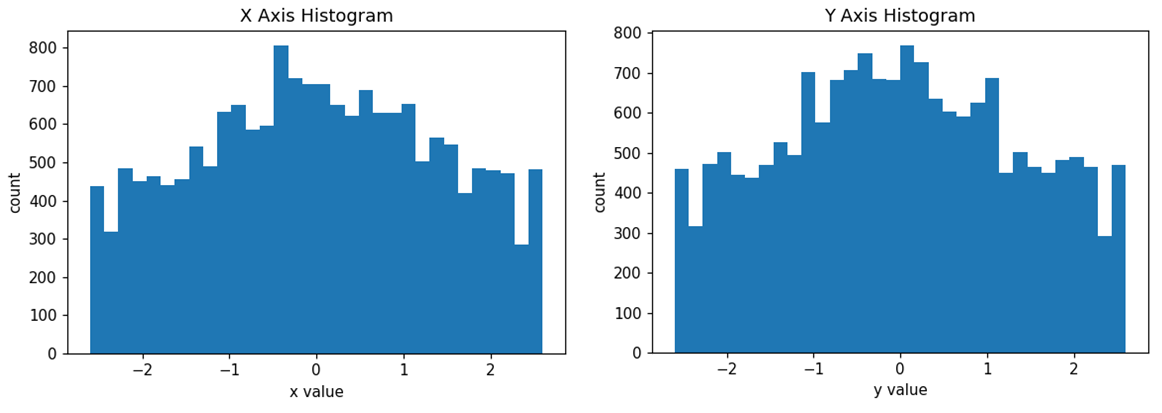
Each data file we have represents a mode. We distinguish between the data by some parameters: the frequency of the field, the radius of the device and of the graphene.

The first step is to normalize the modes. It is done by calculating a numeric integral over the norm of the field at each point, and then diving the samples by the result in order to obtain the normalized functions :

The samples are given in Cartesian coordinates, and the integral is calculated over the volume of the cylindric device. It sounds like it would be easier to work with cylindric coordinates.

Calculating the integral might sound not so difficult, but let’s have a look at the histograms of the coordinates of the samples for the mode :



And in Cartesian coordinates the situation isn’t better:

In this case, we can’t calculate the integral by summing as because the intervals at each axis are not uniformly distributed.

The method I used was:

* Split the space into tetrahedrons with our samples coordinates as vertices. This is done by applying **Delaunay triangulation**.
* For each one of the resulted tetrahedrons:
  + Calculate its volume
  + Calculate the mean of our target function at each vertex (similar to trapezoidal integration for 1D)
* Sum over the above to obtain the integral:

This method yields pretty good results compared to the results of the continuous function integration over the cylinder:

|  |  |  |  |
| --- | --- | --- | --- |
| Function | My integration | Continuous integration | Error |
|  | 7.6241 | 7.6454 | 0.28% |
|  | 1.3733 | 1.3762 | 0.21% |
|  | 12.8908 | 12.9207 | 0.23% |
|  | 4.6411 | 4.6514 | 0.22% |

# Transition rates

## Calculation

The next step after normalizing the modes is to calculate the amplification of the spontaneous emission (Purcell factor) for mode :

Where - is the spontaneous emission rate in free space. It is given in Novotny’s book as:

And we know that when assuming that the dipoles are , so according to Fermi’s golden rule we get the transition rate:

Assuming **a single mode (k=1 only)** the transition rate for a mode ­ would be:

So, for a mode with a corresponding frequency we get:

The density of states is the line-shape of spontaneous emission which is given by a normalized Lorentzian function:

Where . So, we get:

And:

The layers of the devices are constructed in the following order:

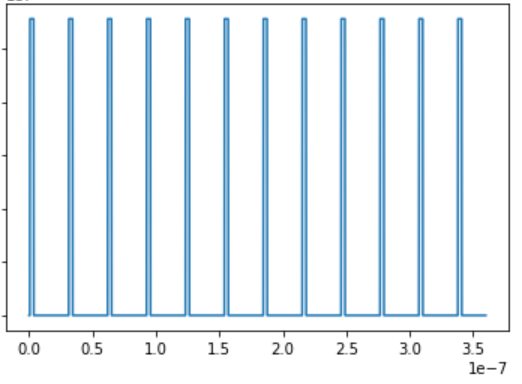
d

d

So, I assume the dipole distribution should be:

Where - is the offset in the z axis of the first dipole layer, and is the width of one period. And is the step function:

For example (when the y-axis has arbitrary values):



Notice that , and but the length of the device is . Because of the constraint which says that the layer next to the graphene should have a width of 8.28nm, I assume that the layers around in the 1st period are omitted.

We want to calculate the effective emission of the device for a specific mode m. Using the normalized dipole density we need to calculate:

This calculation is done numerically in the following steps:

1. Calculating for each point in space and frequency
2. Splitting the space into planes with different z values
3. Integrating over XY each z value plane using Delaunay triangulation and multiplying the area of each triangle by the average of values of it vertices.
4. Interpolating the results over the z axis.
5. Multiplying by and calculating the one dimensional integral

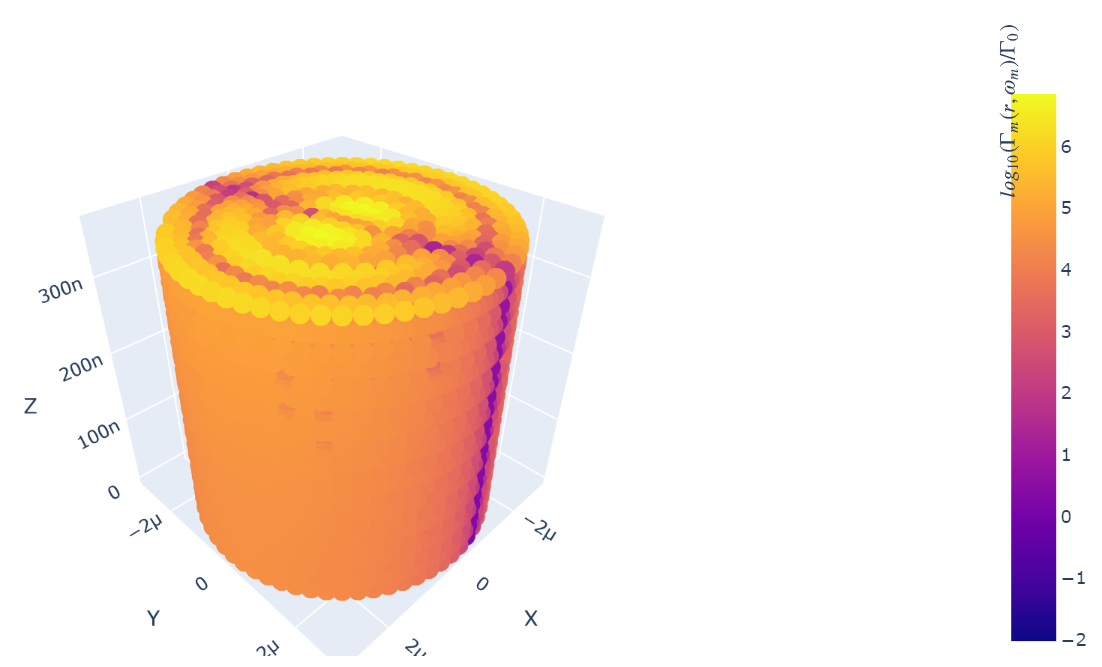
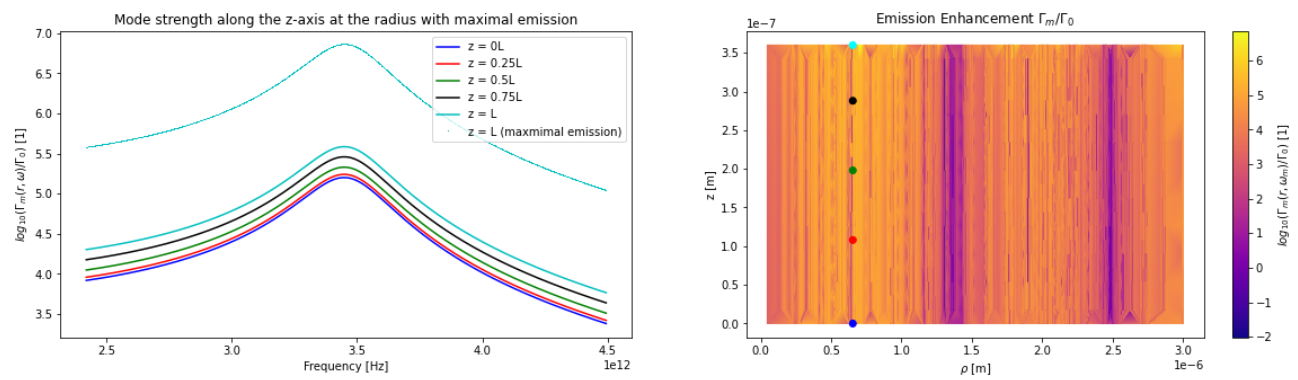
## Plotting - Regardless of the Dipole Density

The spatial distribution of for a certain frequency is proportional to .   
Plotting the for the modes of the device with:

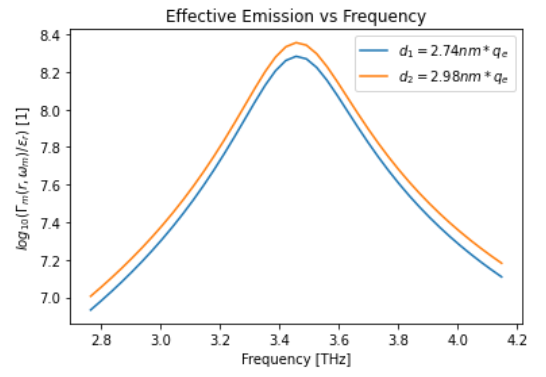


The plot is on a logarithmic scale (.

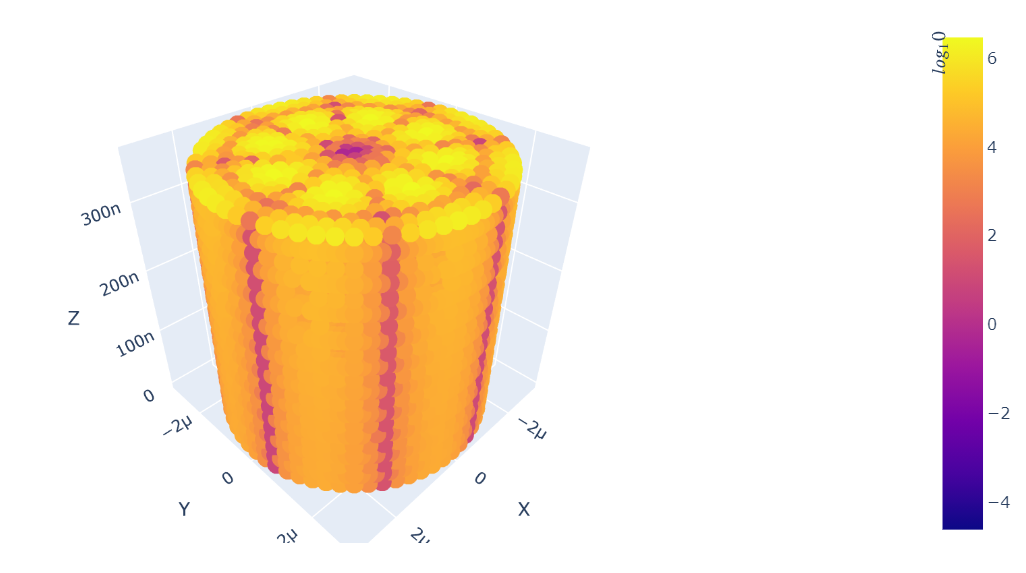
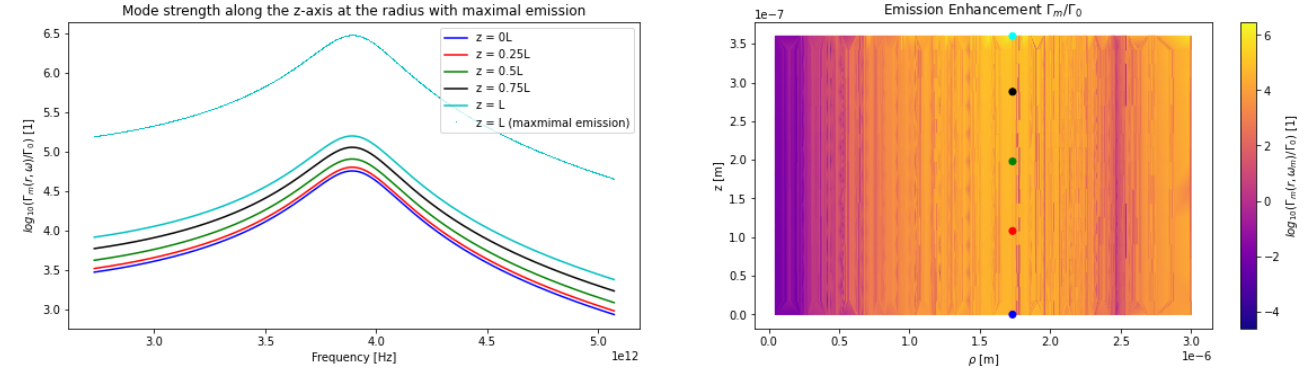
For



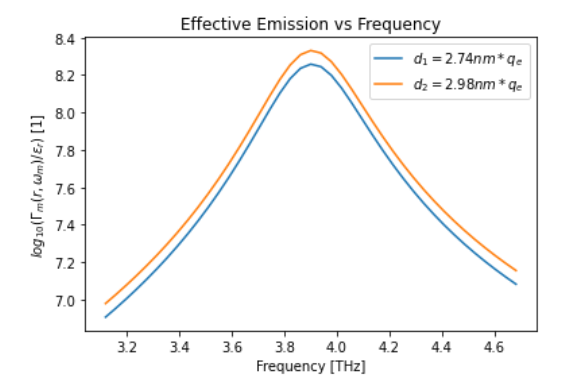
For



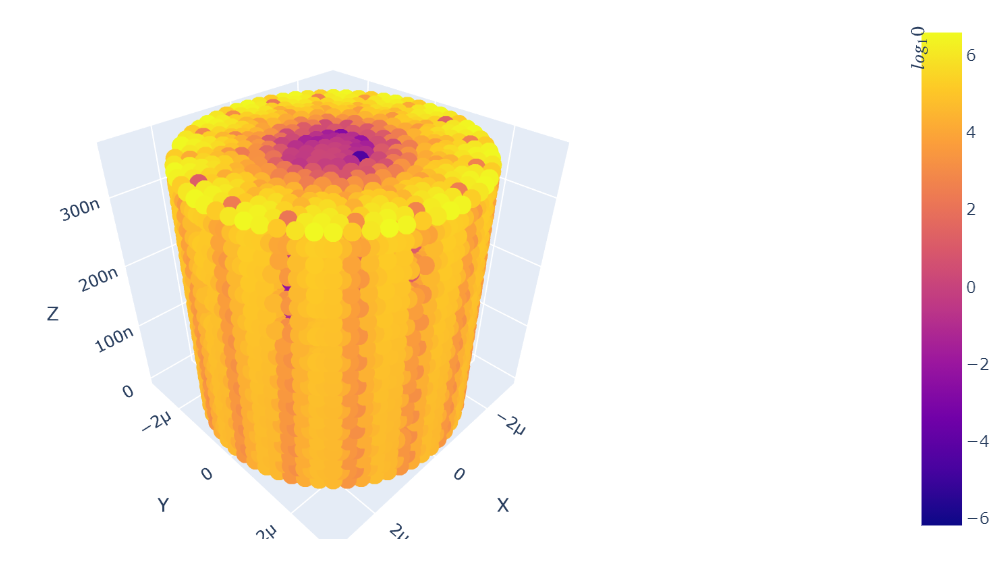
For

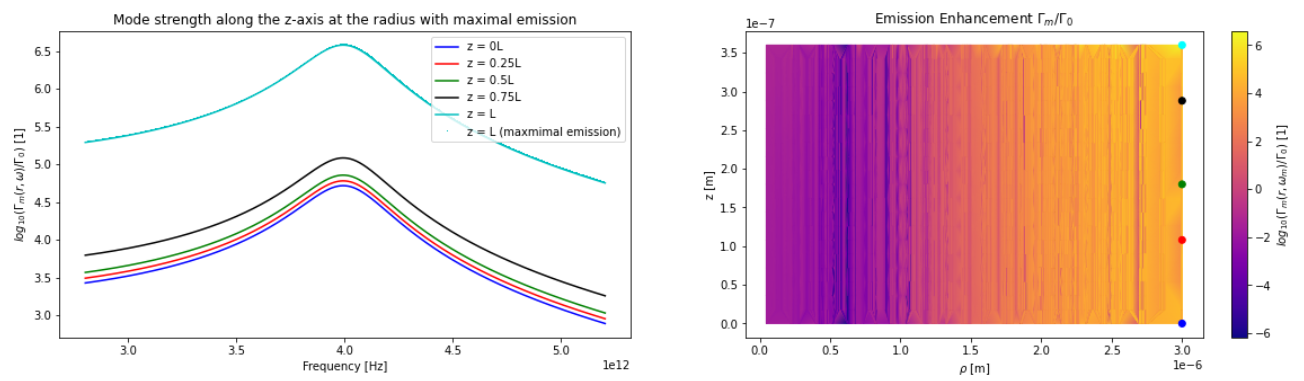


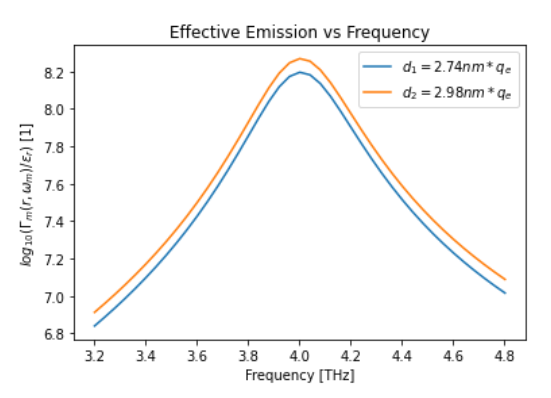
For



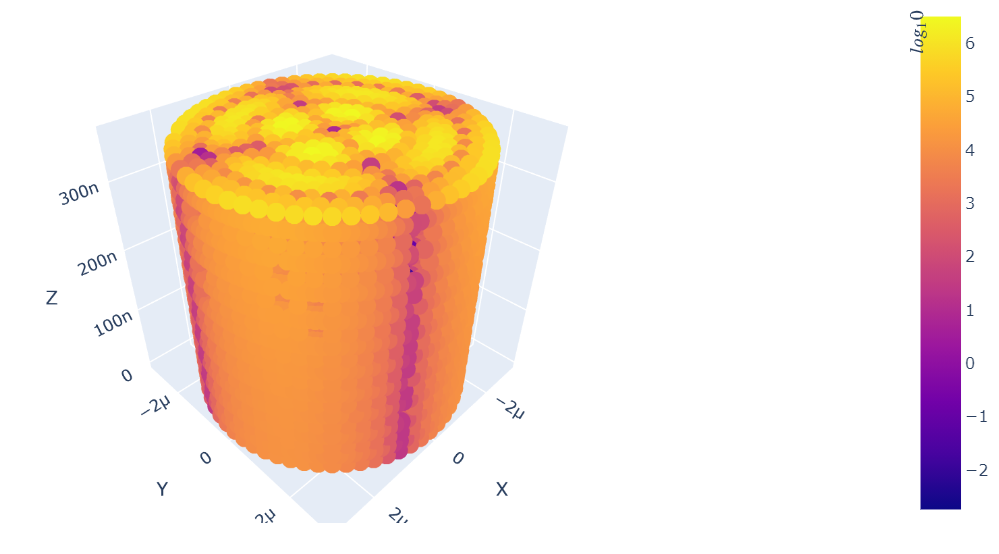
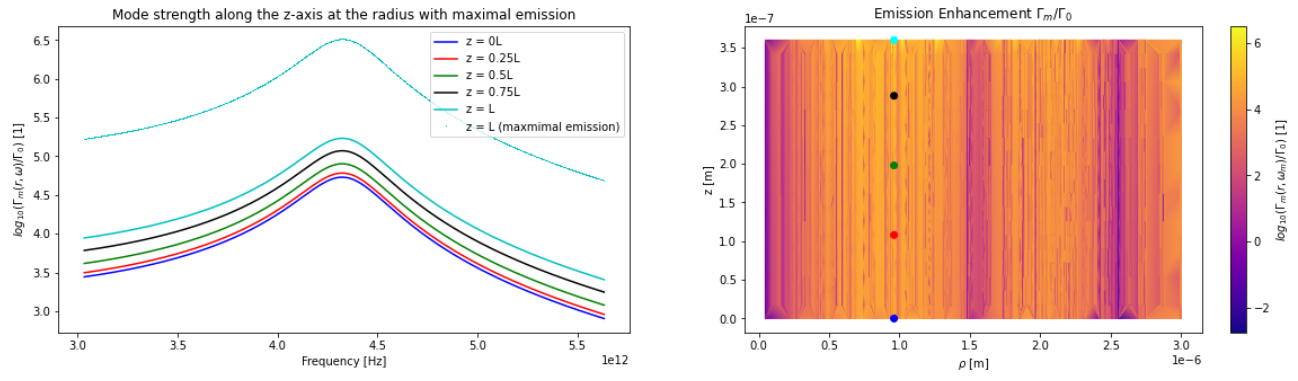
For



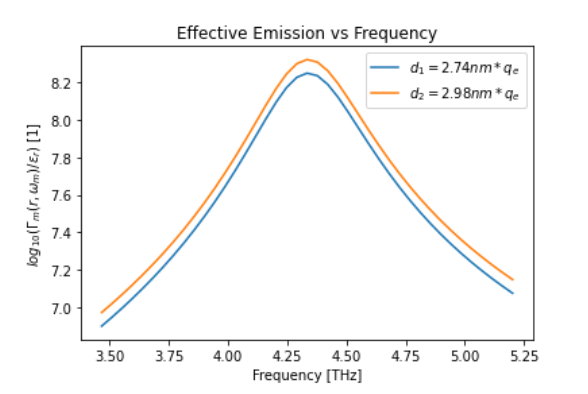
 For



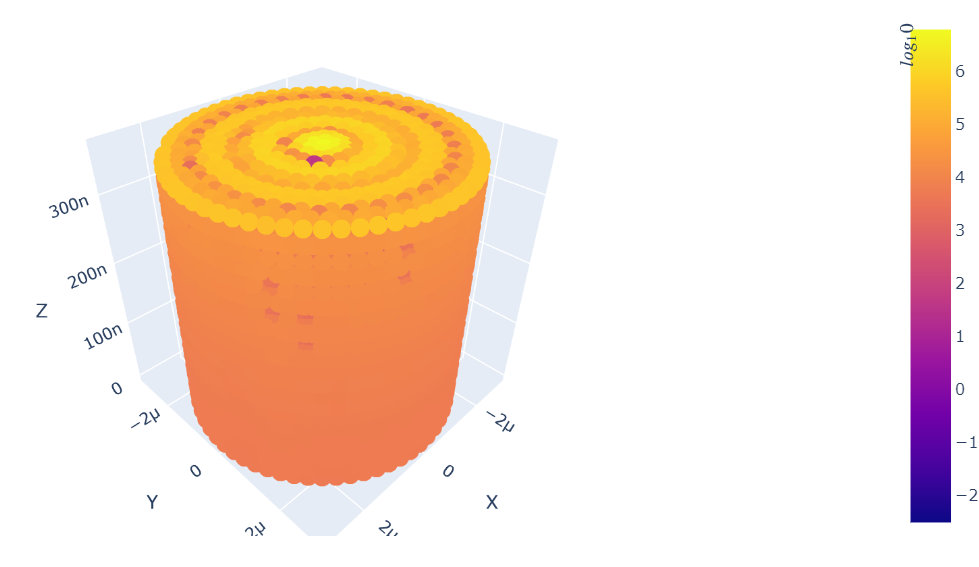
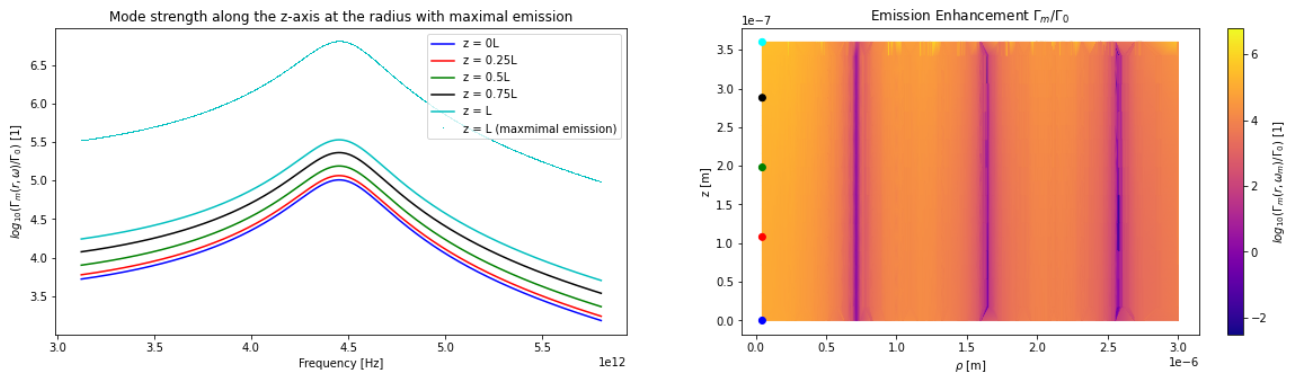
For



For



For



For

