**Real Model Steps:**

To generate real model embedding’s and visualize it using t-SNE we first create a table using postgis query:

*SELECT traj\_id, cell\_id, cell\_names*

*FROM (*

*SELECT tr.traj\_id, ce.cell\_id, ce.cell\_names, ce.grid\_id,*

*ST\_LineLocatePoint(tr.traj\_path, ST\_Centroid(ce.coordinates)) AS frac*

*FROM cells AS ce*

*JOIN traj AS tr*

*ON ST\_Intersects(ce.coordinates, tr.traj\_path)*

*where ce.grid\_id = 2.84302880299e+19*

*) q*

*ORDER BY*

*frac*

*;*

This table stores the *trajectory* and *cell* information through which traj is passing through.

This table is then read by the script ***real\_model\_graph.py***that outputs walks i.e. a sequence of *lists* containing *cell ids* for each trajectory through which it is passed through.

These walks are given to another script ***real\_model\_main.py*** which is the modified version of *node2vec* that avoids generating random walks and accepts the given real walks of trajectories as random walks. This script generates the embedding.

Further these embedding are given to t-SNE for visualization.

**Null Model Steps:**

To generate null model embedding’s and visualize it using t-SNE we will generate a graph of grid cells. Each cell is a node and its connected to its adjacent cells by an edge.

The above graph is generated by script ***embeddings\_tsne.py*** which outputs an *edgelist* of cell ids in a format accepted by *node2vec*.

***Node2vec*** generates embedding’s which are given to t-SNE for visualization.

**Cosine Similarity:**

To calculate cosine similarity, embedding file is given to script ***getting\_vectors\_cosine\_sim.py*** which returns cosine similarity for both real and null model. We store these cosine similarities as separate *real* and *null* model cosine similarities.

**Difference Cosine Similarity (Discuss):**

We fetch both cosine similarities for real and null model and calculate their difference using script ***cosine\_sim\_diff.py.***