# Epi Stroma New Approach

Based on the conversation had with Benjamin, it is clear that there will never be a need to display the entire 20,000 x 20,000 matrix. Instead what will be happening is the following:

1. A User will input some list of up to 10 genes of interest
2. Those genes along with some amount of first and second neighbours will be returned to the user and a graph will be displayed. From then on, the user is only going to be interacting with this particular subgraph. So we are only ever returning a submatrix of the big matrix from our R scripts to the server and back to the client.
3. Note that there is a maximum number of nodes that can be displayed on the screen before everything turns into a hairball. Therefore the amount of first children nodes and second children nodes we display will be a function of that limit.

So I guess the first part of all this would be to develop a script that returns the appropriate submatrix from our original matrix based on the genes of interest specified.

A good question is are we going to be choosing only epi nodes, stroma nodes, or a combo of both at the same time.

It might not be feasible to simply take a submatrix. Say that we specify

Based on this paradigm, everything will be computed on the fly so there won’t be any caching done on the Node JS side. We will however compute the correlation matrices, apply the PValue test and store those results in Rdata files.

Basically what we are doing with the selected nodes is running the find correlations algo a bunch of times. Since we are limiting the amount of neighbours and only taking the top strongest interactions, we run into the following issue:

Say we select two genes of interest: A and B

1. We get the 3 strongest interactions for gene A as well as the 3 strongest interactions for each of those 3 interactions.
2. We get the 3 strongest interactions for gene B as well as the 3 strongest interactions for each of those 3 interactions.
3. WE need to account for the possibility that gene A is one of the top three genes for gene B, but not vice versa. In this case, we need to ensure that we are not adding a duplicate gene A and that we are only adding the edge between B and A.
4. What might be doable is to get the names of all of the genes and their interactios, and then take a submatrix.

Unfortunately R is kind of silly when it comes to indexing. If you specify an index multiple times, it rends up repeating that row or column multiple times which is not what we want. What we want to be able to do is have a set data structure that limits us to only one occurrence of each gene name.

However, R does have a function called unique() which will return the unique elements of a vector

Major issue that R is serializing the matrix column by column. We need to change our createElements method in order to reflect this.

\*Got rid of the old way that we would read the correlation matrix. We are no longer read the upper triangular part, lower triangular part, then diagonal. We are now reading left to right, top to bottom.

## Things to do over the weekend

1. Test first neighbours

2. Test second neighbours

3. Test the genes of interest

4. Test the filtering by edge weight

5. Test varying the layout

6. Test varying the p value

\*Our current logic in getting the relevant nodes and their neighbours has an issue. The R script makes use of a function

\*For some reason R is very stupid when subsetting and getting only a single value back from a matrix. I won’t return the rowName of that value which is very annoying

To prevent R from returning a vector when indexing into a matrix, using the argument drop=FALSE as the 3rd index.