# Developing Metrics for Networks Embedded in Physical Space

A Thesis Presented by

James Stevick

To the Keck Science Department
Of Claremont McKenna, Pitzer, and Scripps Colleges

In partial fulfillment of
The degree of the Bachelor of Arts

Senior Thesis in Physics
May 2015

# Table of Contents

# *Acknowledgements*

*I would like to thank Dr. Adam Landsberg for his guidance throughout this project. I would also like to thank Dr. Julia Owen and Dr. Pratik Mukherjee, Department of Radiology and Biomedical Imaging, U.C. San Francisco, for providing the data set. Thank you to Boyle Ke for his technical assistance and support and endurance through frustrating bugs and issues in program instrumentation. Finally, I would like to thank my parents for their tremendous emotional support.*

# Abstract

This project attempts to quantify a common property amongst networks in which spatial information is relevant to the structure and function of the network. Analyzing data from a high-resolution brain network from dMRI imaging, we find a correlation between edges in terms of start and end points. The edges in networks embedded in physical space, specifically our brain network, tend to organize into clusters. This project aims to develop new metrics for quantifying the spatial clustering in brain networks.

# Chapter 1: Introduction

## 1.1 Networks

A network is a group of points, also known as nodes or vertices, joined into pairs by lines or edges. Networks consist of a set of vertices and a set of edges connecting the vertex points. As shown below in Figure 1 a network is typically depicted as a group of circles, representing the nodes, connected by lines, representing the edges.[1]

Figure 1: Example of a network with 13 nodes and 12 edges

Networks can be used to model, study, and analyze a number of entities in the physical, biological, and social sciences. The applications of networks span a broad spectrum of fields in the real world including social networks like communities, information networks like actors appearing in films together, technological networks like power grids, and biological networks like prey and predators in an ecosystem. Observations of the properties of networks and success in modeling these properties have caused an explosion of research on the mathematics and metrics of networks.[2]

---

[1] Newman, M. E. J., *Networks: An Introduction*, Oxford: Oxford UP, 2010. Print.
[2] Newman, M. E. J., *The Structure and Function of Complex Networks,* Department of Physics, University of Michigan.

### 1.1.1 Varying Types of Networks

The various applications of networks have resulted in a multitude of network species. Networks can be, and often are, much more complex than the one presented above. A network can be weighted or unweighted. In a weighted network, the edges connecting nodes carry a certain strength or weight. An example of this is a network of roads where the nodes represent cities and the weights of the edges represent the distances between connecting cities. In an unweighted network, there is either a connection between each pair of nodes or there is not; the weights are essentially binary. An example of this is a network of Facebook friends, where the nodes represent people and a connecting edge represents whether or not two people are "Facebook friends". Directed graphs, or digraphs, are comprised of edges with a specific direction. For example, a network of literature citations would be represented as a directed graph. Networks may have two different types of nodes, called a bipartite graph, where edges run between nodes of unlike types. An example is an affiliation network, wherein vertices represent groups and individuals and edges affiliate individuals with different groups. There are many additional complexities, including any combination of those listed above.

### 1.1.2 Properties of a Network

In order to characterize a network, a number of metrics or properties have been developed that are used to compare and contrast different networks. The study of networks, in recent years, has revolved around creating random networks and analyzing how real-world networks differ. The ways in which real networks are nonrandom offer deep insight into the driving forces and mechanisms behind network structure and formation. The properties, on which I will go into detail, are the same general properties with which researchers create random networks, providing parameters for specific mean values and distributions of various parameters.  The following properties will refer to the network in Figure 2 for explanations and calculations.

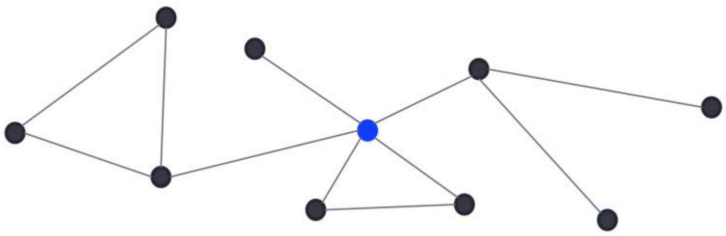


Figure 2:  Example of a network with 10 nodes and 11 edges

#### 1.1.2.1 Average Path Length

The path length is defined as the shortest geodesic distance between two vertices.  In other words, it is the shortest number of steps between each pair of vertices.  This metric can be averaged over all pairs of nodes in order to determine the average path length.  Mathematically, this metric represents the mean of all geodesic path lengths in a network and is calculated using the Equation 1:

$$APL = \frac{2}{n * (n-1)} * \sum_{i \neq j} d_{ij}$$

Equation 1: Average Path Length

In Equation 1, n is the number of nodes and $d_{ij}$ represents the geodesic path length from node i to node j.  Treating an edge as a step, the average path length represents how many steps it takes to get from one vertex to another on average.  In other words, the average path length determines a sort of easiness with which one can reach one vertex from another vertex. The average path length of the network in Figure 2 is $\frac{2}{9*10} * 101 = 2.24$.

#### 1.1.2.2 Average Degree

The degree of a vertex is defined as the number of edges connected to that vertex. The degree of the blue vertex in Figure 2 is five. The average degree simply represents the mean of all vertex degrees in the entire network. This metric is useful in determining the density of a matrix. Referring to the network in Figure 2, the average degree is $\frac{22}{10} = 2.2$.

#### 1.1.2.3 Transitivity

Transitivity or the clustering coefficient is a measurement of how well a network or specific neighborhood of a network is connected. Any given node in a network may have multiple neighbors or connected nodes. The clustering coefficient measures the probability that two neighbors of node i are also neighbors to each other. In other words, when node i is connected to node j and node i is also connected to node k, a connected triple is formed. If nodes j and k are connected then a closed loop or triangle is formed. The clustering coefficient can be defined as the number of triangles multiplied by 3, divided by the number of connected triples. For the network in Figure 2 the clustering coefficient is $3 * \frac{2}{20} = 0.3$.

## 1.2 Spatial Networks

While the properties described above are among the most commonly studied metrics in network theory, they fail to account for a very important piece of information: the physical location of nodes. Before delving into the importance of spatial information, I must highlight how the networks above were arbitrarily depicted. I use the term topological to describe a network which is void of spatial information, the nodes of which are not necessarily embedded in physical space. Therefore, a topological network can be described in its entirety by a set of nodes and a set of edges joining nodes together. The network in Figure 2 is topological, and represents the exact same information no matter where I depict the nodes in Euclidean space. In other words, if I were to scramble around the nodes into different places leaving the weights of the edges intact, the average path length, average degree, and clustering coefficient would not change.

For many reasons, the spatial location of nodes provides much insight into the dynamics and structure of a network, allowing for a more in-depth study of complex systems. In most real-world networks, distance between nodes implies an associated cost and probability of a connection. For example, modeling transportation networks requires embedding the nodes in physical space, where the edges represent pathways of transportation and the nodes represent locations. Intuitively, longer distances between nodes imply a larger cost of transportation (e.g. fuel, compensation, food, etc.). Figure 3 displays the number of flights between airports as a function of the distance between the airports. It is a histogram of all flights according to the distance of their flight paths.

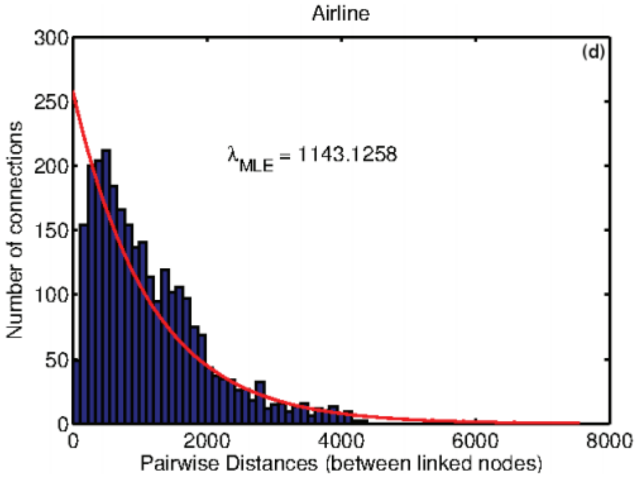



Figure 3: Distribution of pairwise distances between linked nodes[3]

It is clear how the likely hood of a connection between two nodes decreases as the distance between the nodes increases. The mechanism behind this effect is the increasing transportation costs associated with increased distances between nodes. Thusly, we see an exponential decay, barring the sharp drop as pairwise distances approach zero: operation costs discourage flights under a couple hundred miles. As another example, consider a social network. Users are more likely to share a connection with other users within a physical proximity simply because the likelihood of interaction is higher.[4] While the connections do not have associated costs, the proximity effect of physical distances has an effect on the likely hood of connections of a specific distance. These examples show how the physical location of nodes and edges in a network affect the structure and topological metrics in a network.

Networks embedded in physical space can be modeled with a link-cost function, which describes how the costs of connecting nodes (and therefore the probability of a

[3] Larusso, N. D., Ruttenberg, B. E. & Singh, A. (2013) A Latent Parameter Node-Centric Model for Spatial Networks. PLoS ONE 8(9), e71293: doi:10.1371/journal.pone.0071293.g001
[4] Larusso, Ruttenberg, & Singh

connection) vary with the distance between nodes. The simplest property with which researchers create a link-cost function is the degree of a node; however, multiple studies have reduced error in predicting links by including information about each node's specific surroundings based on other network properties.[5] These studies have included transitivity and average path length and I would like to see them include the spatial metrics developed in this project in order to further reduce prediction error.

### 1.2.1 The Brain Network

This project analyzes a weighted, undirected human brain network. The data was taken from Diffusion Magnetic Resonance Imaging (dMRI), which creates a map from the diffusion of molecules, usually water molecules, through the brain. Each node in the data set represents a small cubic millimeter of grey matter called a voxel. The edges of the network represent white-matter fiber bundles.[6] The edge weights represent the strength of the connection, directly measured as the rate of water diffusion at that location. The network is undirected and therefore symmetric, in that if node i is connected to node j with strength X, then node j is also connected to node i with strength X. The diffusive patterns of the brain uncover important information regarding the state of the structure, function, and health of the brain.

### 1.2.2 Previous Studies on Brain Networks

The spatial arrangement of connected voxels in the human brain has been through years of evolution guided by selection pressures. Intuitively, longer neuron pathways

---

[5] Larusso, Ruttenberg, & Singh

[6] Zalesky, Andrew, Alex Fornito, Ian H. Harding, Luca Cocchi, Murat Yucel, Christos Pantelis, Ed T. Bullmore, *Whole-Brain Anatomical Networks: Does the Choice of Nodes Matter?*, NeuroImage, vol. 50, Issue 3, 15 April 2010, pp. 970 - 983

or axonal projections have higher material, temporal, and energy costs.[7] Additionally, there are evolutionary advantages of intelligence and coordination that can be achieved through higher network complexity; longer axonal projections connecting distant modules of voxels provide integration of functions for synergistic advantages. The pressures of minimizing energy costs, maximizing efficiency, and increasing complexity have formed the topological and physical properties of the average human brain neural network, known as the human connectome.[8]

Previous studies on brain networks have investigated the resting state of the brain (fMRI imaging while a subject is not performing a specific task) and the progressive nature of various diseases. Eric Freidman of Berkeley describes the connectome as an impressively adaptive arrangement of neural pathways. Damaged brains follow the same efficiency maximization as healthy brains during development. Through network analysis, contrasting the structures of healthy and damaged brains can predict and ameliorate harmful abnormalities through preventative measures. Additionally, by mapping the progression of disease onset, researchers discover the signals and tendencies of a spreading disease, allowing for earlier and more effective preventative action. [9]

Prior studies have shown brain networks to consistently exhibit the small-world effect and modularity, where a network is comprised of densely clustered hubs with nodes of high vertex degree. Networks divided into a multitude of highly connected clusters

[7] Cherniak, C. Component Placement Optimization in the Brain, J. Neurosci. 14, 2418-2427 (1994)
[8] Bullmore, Ed & Olaf Sporns, Complex Brain Networks: Graph Theoretical Analysis of Structural and Functional Systems, Nature Reviews, Neuroscience, Vol. 10 (2009)
[9] Friedman, Eric J. et al., "The Structural Connectome of the Human Brain in Agenesis of the Corpus Callosum", NeuroImage, 2012

with sparse edges between hubs are characterized as having high modularity.[10] The goal of this project is to effectively quantify the spatial correlations between edges and their tendencies to cluster in order to contribute a significant property towards the aforementioned fields of research. Dr. Julia Owen and Dr. Pratik Mukherjee, Department of Radiology and Biomedical Imaging, U.C. San Francisco, provided the data from a diffusion magnetic resonance imaging (dMRI) scan of a human brain.

[10] Van Den Heuvel, Martijn P., et al. *Small-world and scale-free organization of voxel-based resting-state functional connectivity in the human brain*, Neuroimage 43.3, 528-539 (2008)

# Chapter 2: Preprocessing

## 2.1 The Original Data

The original data file of node connections, T_symm-2.mat, comprised 164,000 nodes and 68,006,762 edges. The data forms three vectors of information. The first two columns represent a pair of connecting nodes, while the third column denotes the weight of the edge connecting the pair of nodes. As shown in Figure 4, a small sample of the data file denotes, highlighted in light blue, that node 1526 is connected to node 1933 with a weighting or strength of 557.

| | 1 | 2 | 3 |
|---|---|---|---|
| 443585 | 1526 | 1931 | 9 |
| 443586 | 1526 | 1932 | 61 |
| 443587 | 1526 | 1933 | 557 |
| 443588 | 1526 | 1934 | 2953 |
| 443589 | 1526 | 1935 | 342 |
| 443590 | 1526 | 1942 | 19 |

Figure 4: Small sample of T_symm-2.mat data file

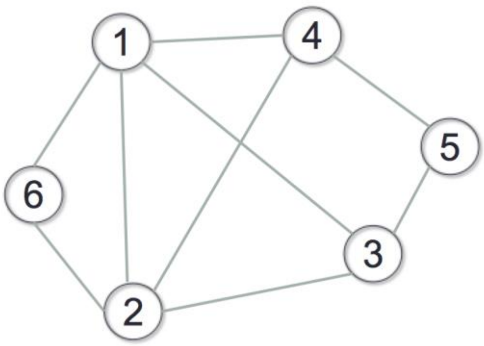
The spatial location data file, Coords.mat, specifies the x-y-z coordinates of each node in the matrix. Each row of this 164,000 by 3 matrix represents the x, y, and z coordinates of each node in millimeters. A sample of this data set is shown below, highlighted in light blue, indicating the x, y, and z coordinates of node 3.

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 71 | 135 | 22 |
| 2 | 70 | 135 | 22 |
| 3 | 71 | 136 | 22 |
| 4 | 85 | 139 | 22 |
| 5 | 84 | 139 | 22 |

Figure 4: Small sample of Coords.mat data file

### 2.2 Adjacency Matrix vs. Sparse Matrix

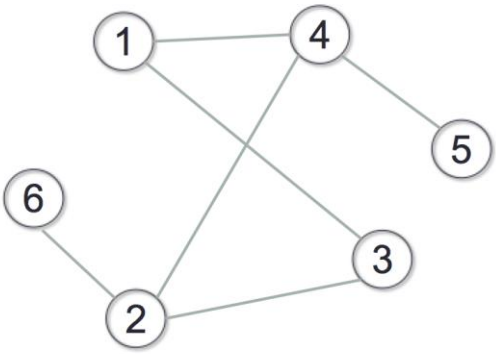
There are multiple ways to store the information of a network in order to preform analysis on a computer. One commonly used, and efficient method, is to create an adjacency matrix. An adjacency matrix is a square matrix in which each cell represents a connection between a pair of nodes indicated by the row and column of that specific cell. Each cell is a Boolean, where one represents a connection and zero represents no connection. As an example, an unweighted network and its corresponding adjacency matrix is shown below in Figure 5. A connection can be seen between nodes 1 and 3. Consequently the adjacency matrix contains a one in row 1, column 3 and in row 3, column 1. Nodes 3 and 4 are not connected resulting in a zero in the respective cells of the adjacency matrix. The main diagonal is empty, representing a weight of zero, since it is nonsensical for a node to have a connection with itself.




| | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|
| **1** | - | 1 | 1 | 1 | 0 | 1 |
| **2** | 1 | - | 1 | 1 | 0 | 1 |
| **3** | 1 | 1 | - | 0 | 1 | 0 |
| **4** | 1 | 1 | 0 | - | 1 | 0 |
| **5** | 0 | 0 | 1 | 1 | - | 0 |
| **6** | 1 | 1 | 0 | 0 | 0 | - |

Figure 5: Sample network and its corresponding adjacency matrix

However, some matrices have very few connections and it becomes inefficient to store each data point considering that majority of the values are zero. These matrices are called sparse matrices since nonzero connection weights are, as the name implies, sparse. After removing a few connections from the network above, the data as stored in an adjacency matrix is comprised mostly of zeroes. Therefore, it should be stored as a sparse matrix rather than an adjacency matrix, as shown in Figure 6 below.




| | |
|---|---|
| A ( 1 , 3 ) = 1 | A ( 3 , 1 ) = 1 |
| A ( 1 , 4 ) = 1 | A ( 4 , 1 ) = 1 |
| A ( 2 , 3 ) = 1 | A ( 3 , 2 ) = 1 |
| A ( 2 , 4 ) = 1 | A ( 4 , 2 ) = 1 |
| A ( 2 , 6 ) = 1 | A ( 6 , 3 ) = 1 |
| A ( 4 , 5 ) = 1 | A ( 5 , 4 ) = 1 |

Figure 6: Sample Network and its corresponding sparse matrix

While the benefits of a sparse matrix are less noticeable in the example of a small network above, the difference is more noticeable in larger networks. The dMRI data set represented by an adjacency matrix would contain 27 billion cells representing 27 billion edges as a 164,000 x 164000 matrix. However, most of these edges are non-existent or, in other words, have a weight of zero. Therefore the data is more efficiently represented as a sparse matrix containing only 68 million edges, nearly 500 times less.

## 2.3 Preprocessing Steps

While loading the data onto the computer for analysis, three steps of preprocessing took place. The spatial information is simply placed into a 164000 x 3 matrix named xyz. Loading the nodes and their corresponding weights is a more complex process.

### 2.3.1 Symmetrization

The code, LoadDataNodes.m, first creates a sparse matrix for the data. The process of data collection resulted in a number of edges between identical nodes. An edge existing between a node and itself is nonsensical, so each was reassigned a value of zero since they cannot carry any weight by definition. The second step was to ensure symmetry of the matrix. If node i and node j are connected, then there must exist a connection between node j and node i. If any reciprocal did not exist, the pair was added to the matrix with a weight of zero. Finally, it is required that each pair of nodes, and their reciprocal, carries the same weight. Averaging the two weightings (the weight of the edge between nodes i and j and the weight between the edge of nodes j and i), symmetrized the pairs of nodes and their reciprocals. To accomplish these two steps, the sparse matrix was momentarily converted into an adjacency matrix. The matrix was then added to its transpose and each element was divided by two. This accomplished all requirements of symmetry with one quick calculation.

### 2.3.2 Strength Thresholding

The weight of an edge connecting a pair of nodes together represents the strength of a neural connection between the two different areas of the brain (voxels). Therefore, it is important to consider what value constitutes a connection. The weights of the edges connecting nodes together in the brain data range from 0 to 9407. If the strength is zero then there is no semblance of a connection. One might expect that a strength value of 2 should represent a weak connection. However, error can often produce this value even in the absence of any neural connection; therefore, certain low strengths should represent an absence of a connection. Therefore we define a threshold value, known as thresh in the code, over which constitutes an edge, under which represents the absence of a connection.

# Chapter 3: Self Edge Correlation (SEC)

The self-edge correlation (SEC) is a measure of how neighboring nodes are spatially located in relation to each other, specifically addressing the spatial correlation between edges. The neighbors of a given node are defined as all other nodes connected by an edge. As mentioned before, the mechanisms driving the structure of networks can cause groupings of vertices and clusters of edges and the SEC quantifies these clusters. The SEC, when carried out on a single node in a network, refers to its neighboring nodes for the calculations. A cluster of nodes is determined by a specific radial proximity. In Figure 7 below, there are two groupings to the left and right of the blue node. The metric itself spews out the probability that two randomly chosen nodes are from the same bunch. The SEC of the network in Figure 7 is $\frac{1}{2}$. The inverse of the SEC has a more intuitive interpretation, indicating the number of groupings in the network. In the network below, the inverse of the SEC is 2, which clearly reflects two clusters of nodes neighboring node B.

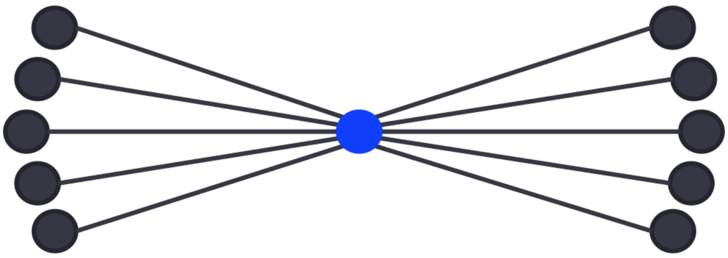


Figure 7: Sample network to demonstrate groupings

The process of calculating the SEC takes several steps and involves many concepts and preliminary calculations, which I will outline below. Each step will refer to the sample network below and focus on the central blue node for calculations.

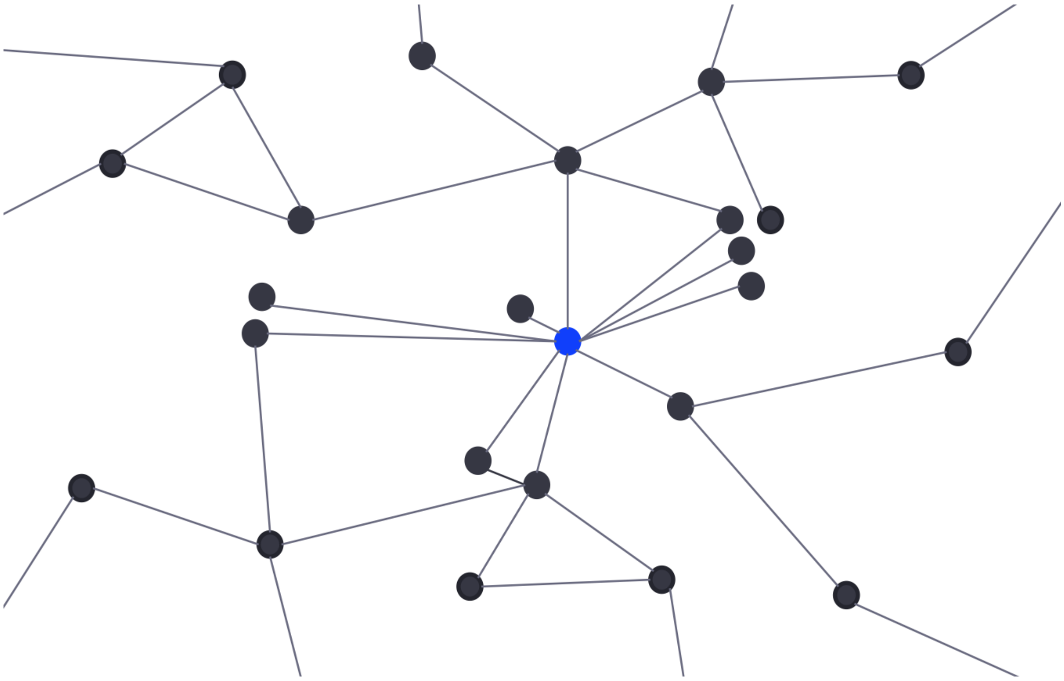


Figure 8: Sample network to outline the calculations of SEC

### 3.1 Global vs Local Neighbors

First, it is important to make the distinction of a local versus a global node.  If node j is connected to node i, then node j is a global or local neighbor of node i depending on the relative distance between node i and node j.  A cutoff distance is set such that if node j is within a radial distance ($D_C$) of node i, than it is a local neighbor; if it is outside of the radial distance then it is a global neighbor.  In Figure 9, node 2 is a global neighbor to node 1 and node 3 is a local neighbor to node 1.

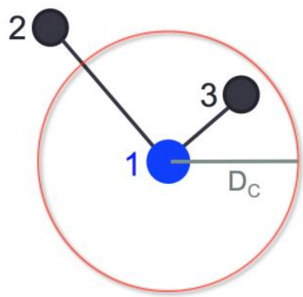



Figure 9: Small 3-node network demonstrating the cutoff distance

### 3.2 SEC Calculations

Each step of calculating the self-edge correlation requires the understanding of lower level functions. The series of images below represent the different steps involved in calculating the SEC. The descriptions of each lower level code will refer to specific steps depicted below in Figure 10.

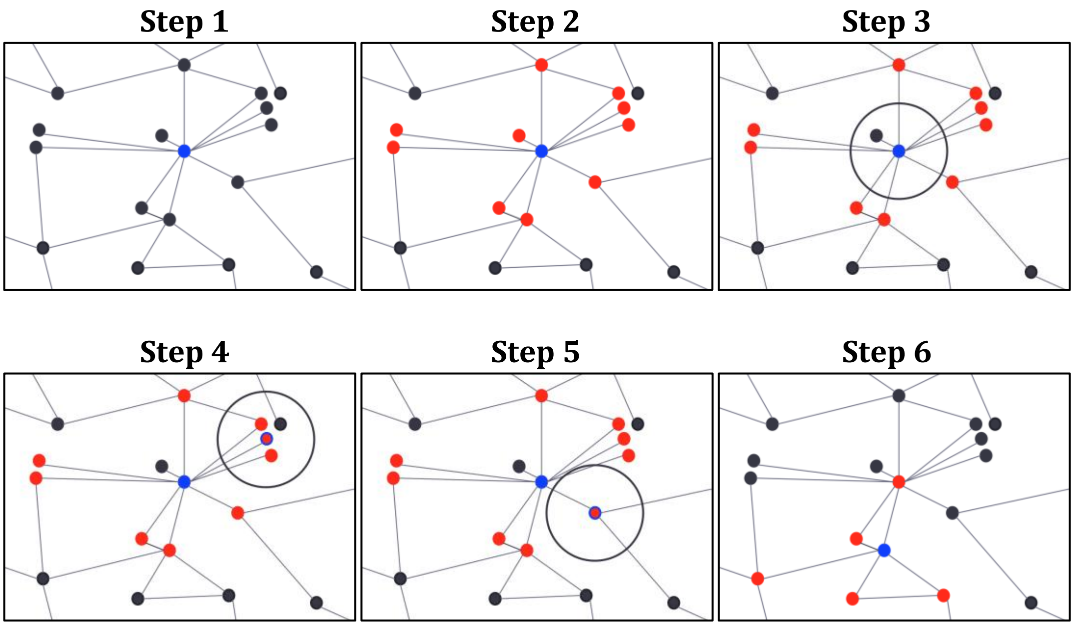



Figure 10: Depicting the calculation of SEC

#### 3.2.1 Gamma

The code gamma_n.m performs the function $gamma(\ n\ )$ that takes in a node n and returns a list of all nodes that are global neighbors to node n. This list, called *glob* in the code, holds all nodes that are connected to node n and are outside of a radial cutoff distance from node n. Consider the blue node in Step 1 of Figure 10, named node B for convenience. $gamma(\ B\ )$ finds all the neighbors of the blue node and selects those that are global to the blue node. In Step 2, all neighbors of the blue node are highlighted in red. However, considering the radial cutoff distance depicted in

Step 3, one of the neighbors is not included in the set *glob* because it is a local neighbor. The function gamma, carried out on node B, returns a list *glob* containing 9 elements representing the 9 red nodes shown in Step 3.

### 3.2.2 Phi

Whereas n refers to the central node of a calculation, the neighboring nodes stored in *glob* are referred to by the variable m ( *glob* = [ $m_1$, $m_2$ , $m_3$, …] ). These m's can be found as the red nodes in Step 3. After the set *glob* is attained from running $gamma(\, n\,)$ the function $phi(\, n\,,\ m\,)$, returns the number of other m's (global neighbors of n) that are local to node m. Since *glob* usually contains multiple nodes, the calculation of $phi(glob)$ returns the phi of each element in glob and returns the mean:

$$phi\,\left(\,gamma(\,n\,)\right) = \ phi\,\left(\,[\,m1, m2\,, m3, \ldots\,]\,\right)$$

$$= \frac{phi(\,m1\,)\ +\ phi(\,m2\,)\ +\ phi(\,m3\,)\ +\ \ldots}{\#\ elements\ in\ glob}$$

Equation 1: *phi ( gamma( n ) )*

Back to our example from Figure 10, consider a single global neighbor of node B. In Step 4, one of the red global neighbors is selected and outlined in blue. When the phi function is carried out on this node, the function counts the number of other global neighbors (red nodes) that are within the cutoff distance of that node. As depicted in Step 4, the function phi finds two other global neighbors within the radial cutoff distance. While there is a third node within the radial cutoff distance, it is not a global neighbor of node B so it is not included in $glob$ nor is it included in the function phi. Since *glob* contains multiple nodes, the function moves on to the next global neighbor of node B as shown in Step 5. In Step 5, there are no nodes within the *radial* cutoff distance, so the phi function returns zero. $Phi\ (\ gamma(\ n\ )\ )$ returns the average phi value when carried out over all global neighbors of a central node. In Figure 10,

$$phi\ (\ B) \text{ returns } \frac{2+2+2+1+1+0+0+1+1}{\#\ elements\ in\ glob} = \frac{10}{9} = 1.111$$

### 3.2.3 Self-Edge Correlation (SEC)

The self-edge correlation is a simple calculation after $gamma(\ n\ )$ and $phi(\ gamma(\ n\ )\ )$ have been calculated.

$$SEC = \frac{1 + phi(\ gamma(\ n\ )\ )}{|gamma(\ n\ )|}$$

Equation 2: Self-Edge Correlation

where $|gamma(\ n\ )|$ = the number of elements in the set $gamma(\ n\ )$.

The SEC of the blue node from Figure 8 is,

$$SEC = \frac{1 + phi(\ gamma(\ B\ )\ )}{|\ gamma(\ B\ )\ |} = \frac{1 + 1.111}{9} = 0.235$$

This means that if we were to randomly select two edges attached to node B, there is a 23.5% chance that they would be from the same cluster (defined by the radial cutoff distance). The inverse of the $SEC$ equals $\frac{1}{SEC} = \frac{1}{0.235} = 4.26$, meaning that there are approximately 4.26 clusters of nodes. Looking at the network, one might intuitively guess that there are 5 groups of nodes: a group of 3, two groups of 2, and 2 single nodes. However, the global neighbors that stand alone (i.e. physically distant from other global neighbors) contribute less to the SEC value. This makes sense because it is nonsensical to count a single node as a cluster. Therefore the SEC value of 4.26 makes more sense in that there are three clear groups of two or more nodes, while the two single node "clusters" count as a little over one cluster (the remaining 1.26) rather than two full clusters.

### 3.2.4 Average SEC

In figure 8, each node surrounding the central blue node has edges connecting to other nodes. Although we previously calculated the SEC on a single node, one could imagine calculating the SEC for each node in a network. In Step 6 of Figure 10, the central blue node has moved on to another node in the network. The calculations of SEC begin with the global neighbors of this new node B. The SEC value of a network requires the calculation of the SEC for each node in the network and an average is subsequently taken. Therefore the SEC values represent the average SEC value of each node in the network. In order to attain a SEC metric for a very large network, this SEC value is averaged over a sample of many nodes. Considering the massive size of the network in this project (164,000), 10,000 nodes were randomly sampled to attain an average SEC that is representative of the whole network. Table 1 shows a few SEC values of the brain network dMRI data sampled over 10,000 nodes and at various cutoff distances and edge strength thresholds.

### 3.2.5 SEC Results

As expected, the SEC values increased with distance cutoff holding the strength threshold constant. For instance, the SEC at a strength threshold of 5 is 0.1483 for a distance cutoff of 3 versus 0.2658 for a distance cut off of 5. It continues to increase as a distance cutoff of 8, giving a SEC value of 0.4011. This makes sense because a larger radial distance cutoff allows for more local neighbors and therefore a higher $phi(\ gamma(\ n\ )\ )$ in the numerator of SEC from Equation 2. One might expect that increasing the cutoff distances also decreases the number of global neighbors and therefore reduce the SEC. While it does reduce the number of global neighbors, it has little effect on the function phi and will only decrease the numerator of Equation 2 as the number of global neighbors decreases, further increasing the SEC.

| Distance Cutoff | Strength Threshold | Sample Size | Self-Edge Correlation |
|---|---|---|---|
| **3** | 5 | 10,000 | 0.1483 |
| **3** | 64 | 10,000 | 0.3712 |
| **5** | 1 | 10,000 | 0.1848 |
| **5** | 5 | 10,000 | 0.2658 |
| **5** | 128 | 10,000 | 0.5492 |
| **8** | 5 | 10,000 | 0.4011 |

Table 1: Table of SEC values using a variety of parameters

The effect of the varying strength thresholds was less expected. As the strength threshold increased, eliminating weak neural connections between nodes, SEC values increased. The reasons contributing to this effect are less apparent. Due to the fact that many of the connections with low strengths arose out of randomness, their structure is similar to that of a random network. Random networks tend to exhibit less clustering of nodes or edges and therefore would produce lower SEC values. Eliminating these low strength edges leaves the connections that more accurately reflect the neural paths of the brain, and therefore have higher SEC values.

The SEC values of the brain network were significantly larger than those of a random network holding the distance cutoff and strength threshold constant. For random networks, we would expect very low SEC values for any reasonable parameters. While the SEC of a random network increases with distance cutoff and strength threshold, the values are much closer to zero.[11] While the step-by-step description of calculating the SEC refers to specific nodes for each function, this project is concerned with the clustering of edges and their spatial correlations—as the name "Self-Edge Correlation" implies. However, the edges and their boundaries are defined by the nodes they are connected to, so the two are interchangeable in terms of clustering. A more complete list of SEC values over various distance cutoffs and strength thresholds can be found in appendix I.

[11] E. Friedman & A. Landsberg, Unpublished

# Chapter 4: Differential SEC

Another way to analyze the clustering of edges in a network embedded in physical space is through the metric called the differential self-edge correlation (dSEC). The function essentially computes the number of global neighbors that are local to each other, also known as a cluster. Over a large sample of nodes, the distribution of the many dSEC values delineates the most common dSEC values and provides insight to the spatial correlation between edges and their tendency to cluster.

## 4.1 Differential Self-Edge Correlation of a Single Node

Mathematically, the calculation of the differential SEC is carried out on a single node within the set of global neighbors to a central node of interest. After $gamma(\, n\,)$ has computed a list of global neighbors ($m_1$, $m_2$, $m_3$, …), one of these global neighbors is selected, say $m_1$. The differential SEC of that node $m_1$ is the fraction of all other global neighbors ( $m_2$, $m_3$, … $m_n$) that are within the radial cutoff distance of $m_1$.

$$dSEC\ (\, n\,, m1\,) = \frac{\#\ \ of\ nodes\ in\ gamma(\, n\,)\ local\ to\ m1}{\#\ of\ nodes\ in\ gamma(\, n\,)}$$

$$= \frac{\#\ of\ nodes\ in\ [\, m2\,, m3,\ \ m4 \ldots]\ local\ to\ m1}{|gamma(\, n\,)|}$$

Equation 3: Differential Self-Edge Correlation - $dSEC\ (\, n\,, m\,)$

Referring to Figure 12 below, dSEC ( blue node , red node ) is $\frac{4}{13}$ because any red node will have four other red nodes within a radial cutoff distance and there are a total of 13 global neighbors of the blue node. This value of dSEC holds for all red nodes. Similarly, the dSEC of the orange nodes is $\frac{3}{13}$, the dSEC of the green nodes is $\frac{2}{13}$, and the dSEC of the grey node is zero. In order to visualize the dSEC values around the blue node, we look at the distribution of the dSEC fractions of the blue node's global neighbors in a histogram:

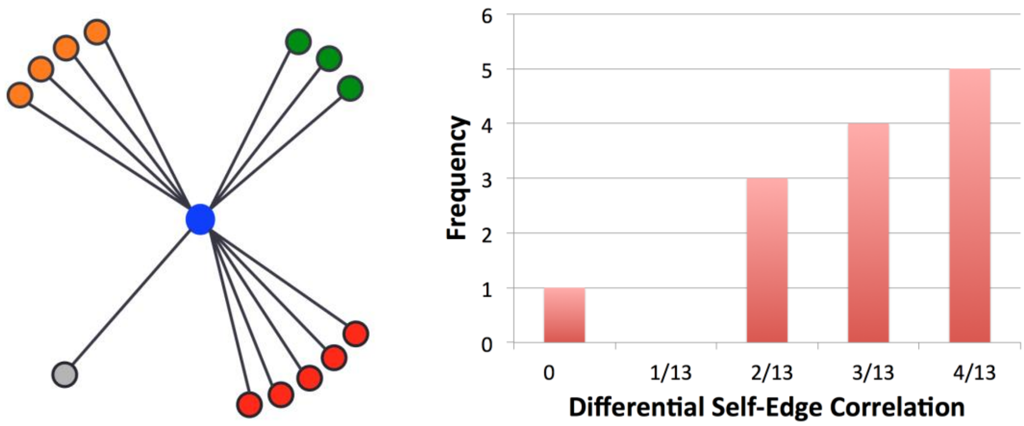



Figure 12: Small Example Network and Corresponding dSEC Distribution

In Figure 12, the largest cluster corresponds to the largest differential self-edge correlation values. In some sense, this small-scale distribution depicts four frequency peaks each representing a cluster of edges. Each peak takes place at a specific dSEC value, offering insight into the size of the cluster. The medium sized frequency peak of three at a dSEC value of $\frac{2}{13}$ corresponds to the cluster of three green nodes, each with that dSEC value. Each node within the cluster is located within the radial cutoff distance of two other nodes that are also members of the 13 global neighbors of the central blue node. The slightly larger frequency peak exists at a dSEC value of $\frac{3}{13}$, indicating that each of the four orange nodes is local to three of the thirteen global neighbors of the blue node. The largest peak corresponds to the largest cluster of five red nodes. Lastly, the single grey node produces a dSEC value of zero, due to its lack of local neighbors, with a frequency of one.

## 4.2 Differential Self-Edge Correlation over a Network

In the case of Figure 12, the simplicity of a single central node directly causes a perfect correlation between the frequency of dSEC values and the dSEC values themselves. In a more extensive and complex network, the dSEC distribution can be created by aggregating dSEC values from multiple nodes. In this case, the histogram creates a smoother distribution, generally peaking at a low dSEC value since perfectly assembled clusters are unrealistic, as shown in Figure 13. The location of the frequency peak depends on the cutoff distance and the strength threshold; however, the low dSEC frequency peak mathematically results from either a multitude of small clusters or large clusters that are dispersed enough to comprise many small clusters according to the radial cutoff distance. While these theories are simple speculation based on the mathematical equation of the differential self-edge correlation, the distribution shows a peak at a higher dSEC value than that of a random network.[12] The random network would peak much closer to a dSEC of zero and exhibit an exponential decay, since majority of the nodes would produce a small dSEC value due to the lack of clustering.

## 4.3 dSEC Results

Figure 13 shows the dSEC distribution over a randomly selected sample of 200 nodes with a distance cutoff of 3 and a strength threshold of 5; the dSEC values peak at a very low value and taper off. As shown later in the section, a random network would exhibit a much higher peak at a dSEC value similarly close to zero; however it would taper off much faster, with essentially zero dSEC values occurring past 0.4.

[12] E. Friedman & A. Landsberg, Unpublished

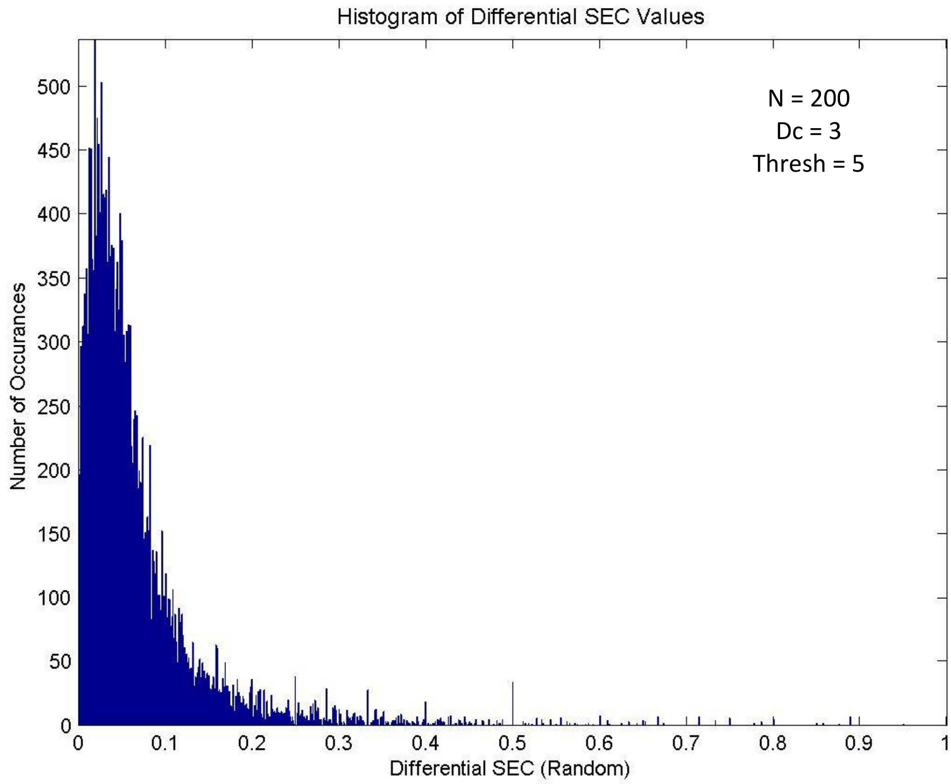



Figure 13: dSEC Distribution with Random Sampling

Randomly selecting multiple nodes from our network to create an aggregate dSEC distribution muddles information. With large samples, the massive number of dSEC values are too randomized and widely distributed, creating an expected distribution covering a wide spectrum of dSEC values void of significant frequency peaks. Therefore, we compare the dSEC distribution of nodes with a specific self-edge correlation value. Higher SEC values are associated with large clusters and thusly we expect to see a distribution with peaks at higher dSEC values. Accordingly, the dSEC distribution over nodes with lower SEC values should result in frequency peaks at lower values of dSEC. Figure 14 shows the dSEC distribution over 200 nodes with SEC values between 0 and 0.2. By selecting nodes with low SEC values, the distribution resembles what we would expect of a random network as a random network would comprise of lower SEC nodes more consistently in the range of 0 to 0.2.

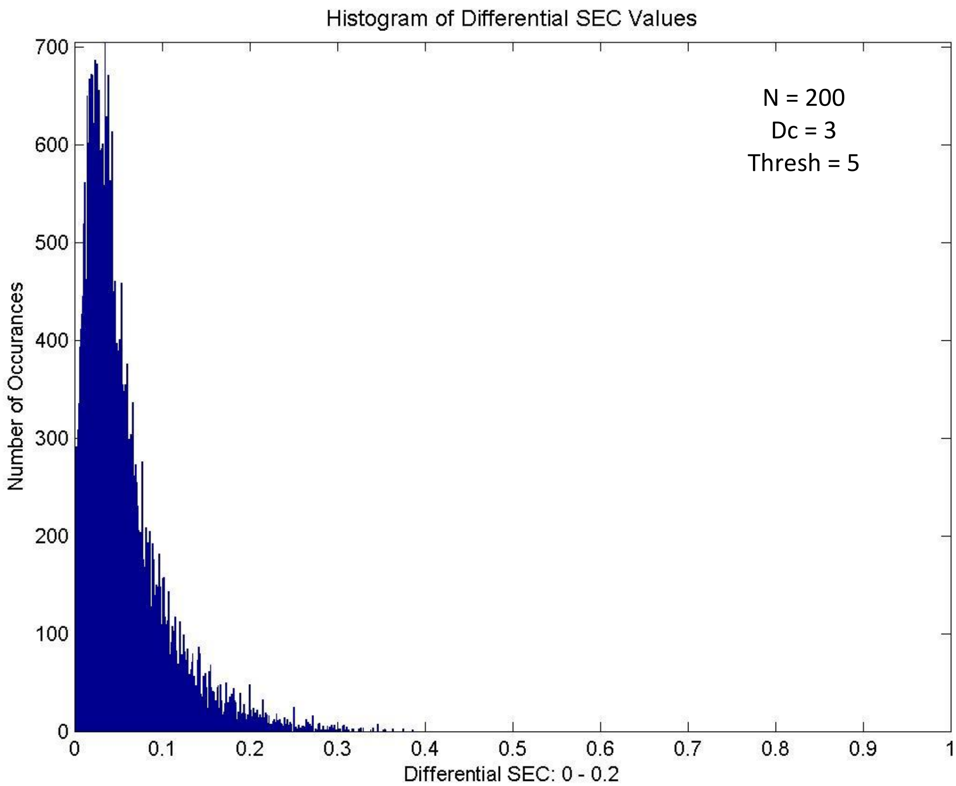



Figure 14: dSEC Distribution with SEC values between 0 and 0.2

As expected, the frequency at low dSEC values (approximately 0.05) is higher compared to the random sampling from our brain network—up to 700 from 550—and the histogram shows no sign of dSEC values above 0.4. This more closely resembles the dSEC distribution of a random network. Additionally, there is a clear correlation between the SEC values of the selection pool for the dSEC distribution and the dSEC value at which the distribution peaks. By moving the selection pool to nodes with SEC values between 0.2 and 0.4, the peak moves significantly to the right (approximately 0.25) as shown in Figure 15:

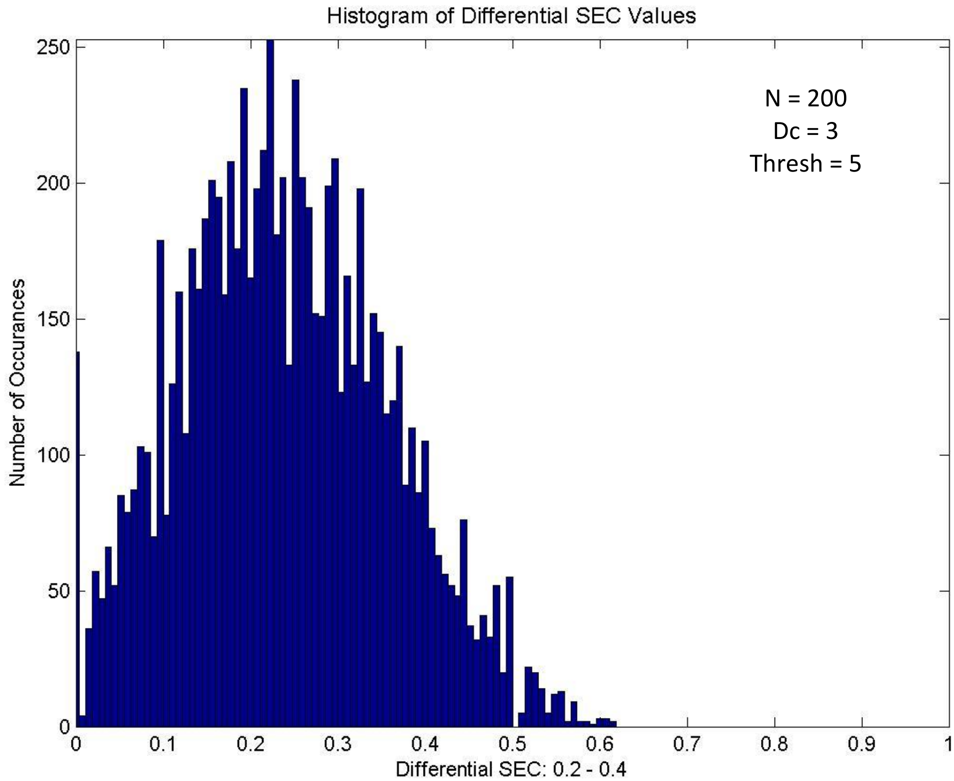



Figure 15: dSEC Distribution with SEC values between 0.2 and 0.4

The dSEC value at which the distribution peaks increases with the SEC values of the selection pool from which the nodes are sampled. The distribution also becomes less concentrated as the pressures from the natural high frequency of low dSEC values and sampling from a selection pool of high SEC values oppose each other. Figure 16 further demonstrates this effect as the selection pool moves to nodes with a SEC value between 0.5 and 1, the peak of the distribution moves to approximately 0.7, and the distribution becomes even less concentrated.

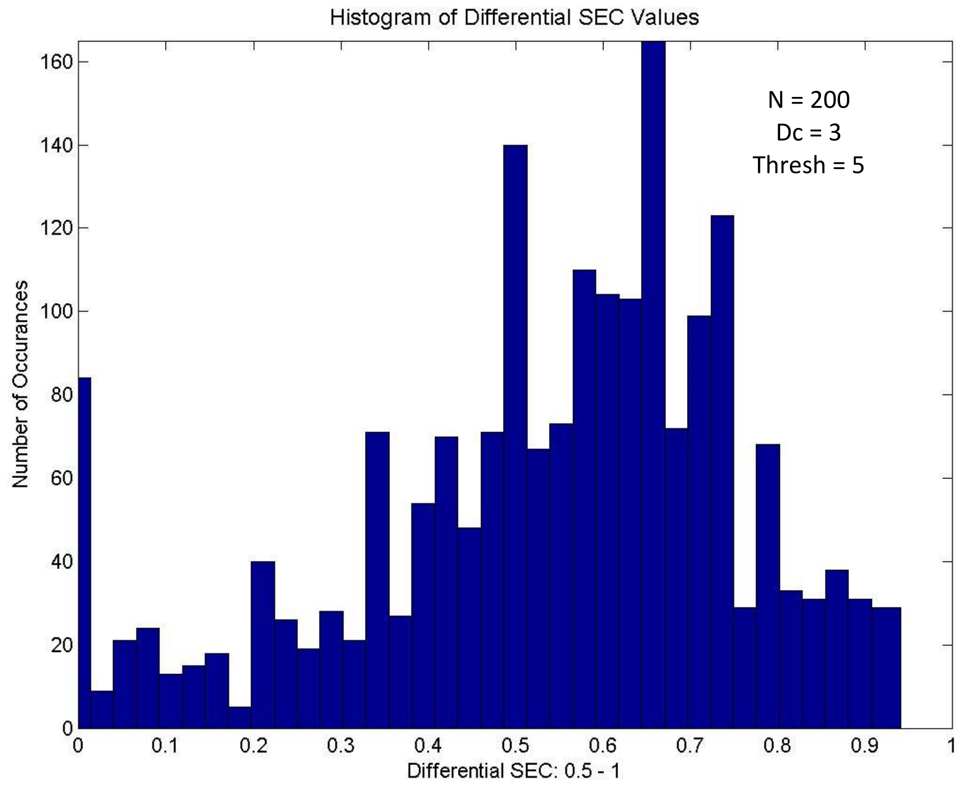



Figure 15: dSEC Distribution with SEC values between 0.5 and 1

A more extensive set of dSEC distributions at various distance cutoffs, strength thresholds, and selection pools based on SEC values can be found in appendix II.

# Chapter 5: Conclusion

This project analyzed the human brain network as represented from a high resolution diffusion MRI scan of a human brain. The edges of a brain network tend to cluster into groups and the goal of this project is to quantify this effect. These new metrics have the potential to characterize networks under a new criterion, adding to the extensive list of metrics that help us understand the structure and function of the human brain.

Extended work on spatial properties should analyze the integrity of these metrics in reducing error on predictions of the formation and maintenance of connections. Additionally, future work could analyze how these metrics vary with other metrics established in the literature of network theory, such as the average degree, transitivity, and the average path length. These metrics contribute towards the studies of adaption, disease, and intelligence as functions of the complex structure of brain networks. The greater our understanding of the techniques of characterizing and quantifying the properties of a brain network, the further we progress in revolutionary research on the human brain.

# References

Data set obtained from Dr. Julia Own and Dr. Pratik Mukherjee, Department of Radiology and Biomedical Imaging, U.C. San Francisco

Newman, M. E. J. *Networks: An Introduction*, Oxford: Oxford UP, 2010. Print

Newman, M. E. J. *The Structure and Function of Complex Networks*, Department of Physics, University of Michigan

Larusso, N. D., Ruttenberg, B. E. & Singh, A. (2013) A Latent Parameter Node-Centric Model for Spatial Networks. PLoS ONE 8(9), e71293:

Zalesky, Andrew, Alex Fornito, Ian H. Harding, Luca Cocchi, Murat Yucel, Christos Pantelis, Ed T. Bullmore, *Whole-Brain Anatomical Networks: Does the Choice of Nodes Matter?*, NeuroImage, vol. 50, Issue 3, 15 April 2010, pp. 970 - 983

Cherniak, C. Component Placement Optimization in the Brain, J. Neurosci. 14, 2418-2427 (1994)

Bullmore, Ed & Olaf Sporns, *Complex Brain Networks: Graph Theoretical Analysis of Structural and Functional Systems*, Nature Reviews, Neuroscience, Vol. 10 (2009)

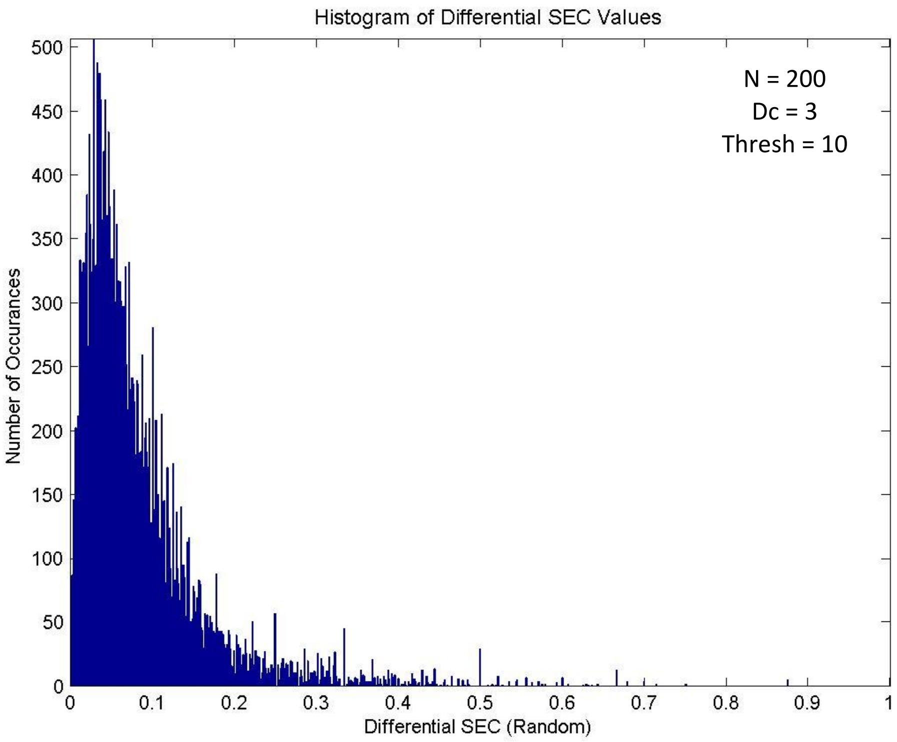
Friedman, Eric J. et al., “The Structural Connectome of the Human Brain in Agenesis of the Corpus Callosum”, NeuroImage, 2012

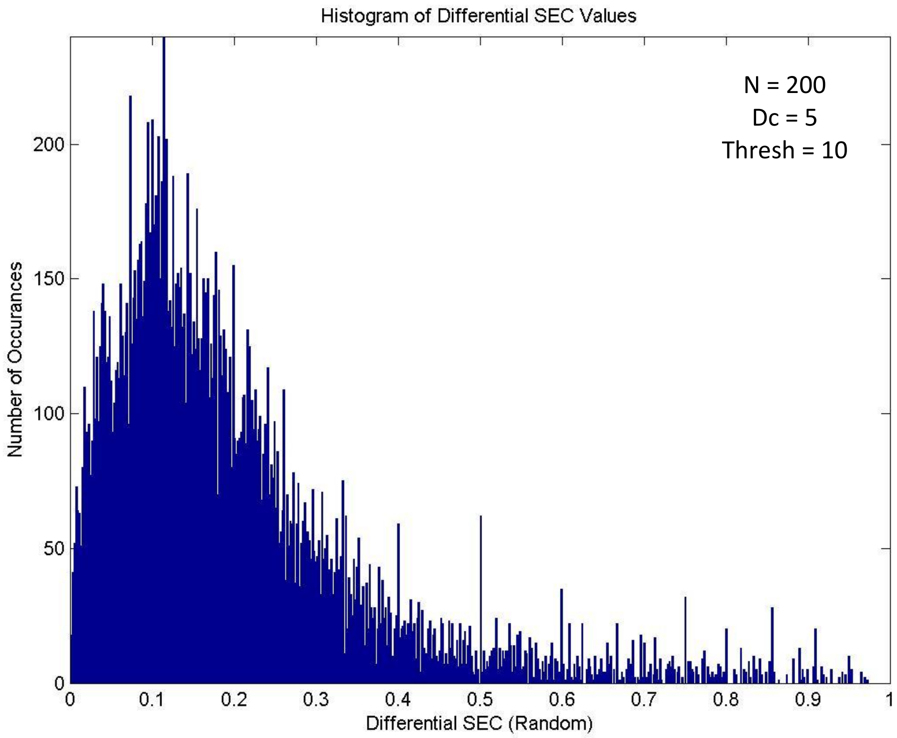
Van Den Heuvel, Martijn P., et al. *Small-world and scale-free organization of voxel-based resting-state functional connectivity in the human brain*, Neuroimage 43.3, 528-539 (2008)

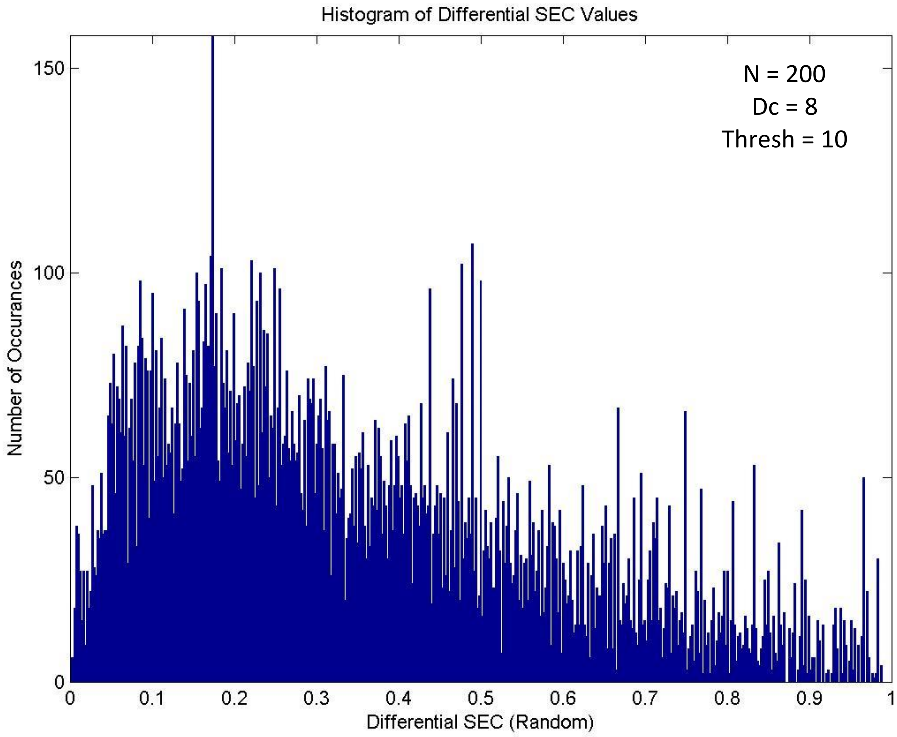
# Appendix I: SEC Values

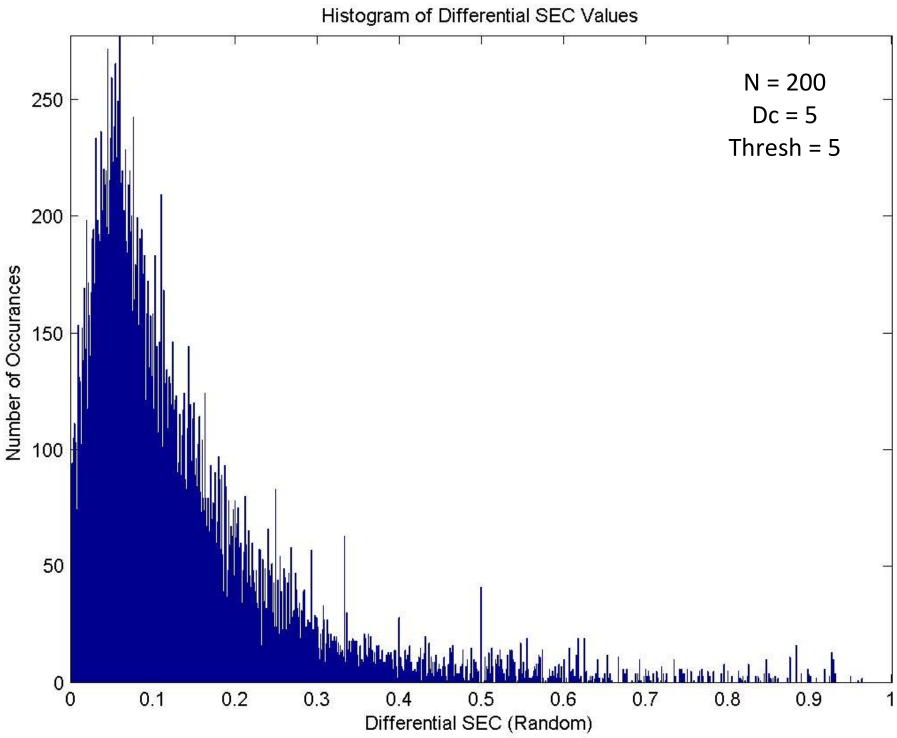
| Self-Edge Correlation | Distance Cutoff | Strength Threshold | Sample Size |
|---|---|---|---|
| 0.0954 | 3 | 1 | 10000 |
| 0.0978 | 3 | 2 | 10000 |
| 0.1160 | 3 | 3 | 10000 |
| 0.1324 | 3 | 4 | 10000 |
| 0.1483 | 3 | 5 | 10000 |
| 0.1557 | 3 | 6 | 10000 |
| 0.1655 | 3 | 7 | 10000 |
| 0.1729 | 3 | 8 | 10000 |
| 0.1870 | 3 | 10 | 10000 |
| 0.2924 | 3 | 32 | 10000 |
| 0.3712 | 3 | 64 | 10000 |
| 0.4494 | 3 | 128 | 10000 |
| 0.4957 | 3 | 256 | 10000 |
| 0.1848 | 5 | 1 | 10000 |
| 0.1880 | 5 | 2 | 10000 |
| 0.2210 | 5 | 3 | 10000 |
| 0.2454 | 5 | 4 | 10000 |
| 0.2658 | 5 | 5 | 10000 |
| 0.2805 | 5 | 6 | 10000 |
| 0.2972 | 5 | 7 | 10000 |
| 0.3098 | 5 | 8 | 10000 |
| 0.3279 | 5 | 10 | 10000 |
| 0.4446 | 5 | 32 | 10000 |
| 0.5066 | 5 | 64 | 10000 |
| 0.5492 | 5 | 128 | 10000 |
| 0.5144 | 5 | 256 | 10000 |
| 0.3033 | 8 | 1 | 10000 |
| 0.3010 | 8 | 2 | 10000 |
| 0.3492 | 8 | 3 | 10000 |
| 0.3784 | 8 | 4 | 10000 |
| 0.4011 | 8 | 5 | 10000 |
| 0.4194 | 8 | 6 | 10000 |
| 0.4399 | 8 | 7 | 10000 |
| 0.4530 | 8 | 8 | 10000 |
| 0.4653 | 8 | 10 | 10000 |
| 0.5464 | 8 | 32 | 10000 |
| 0.5570 | 8 | 64 | 10000 |
| 0.4895 | 8 | 128 | 10000 |
| 0.3887 | 8 | 256 | 10000 |

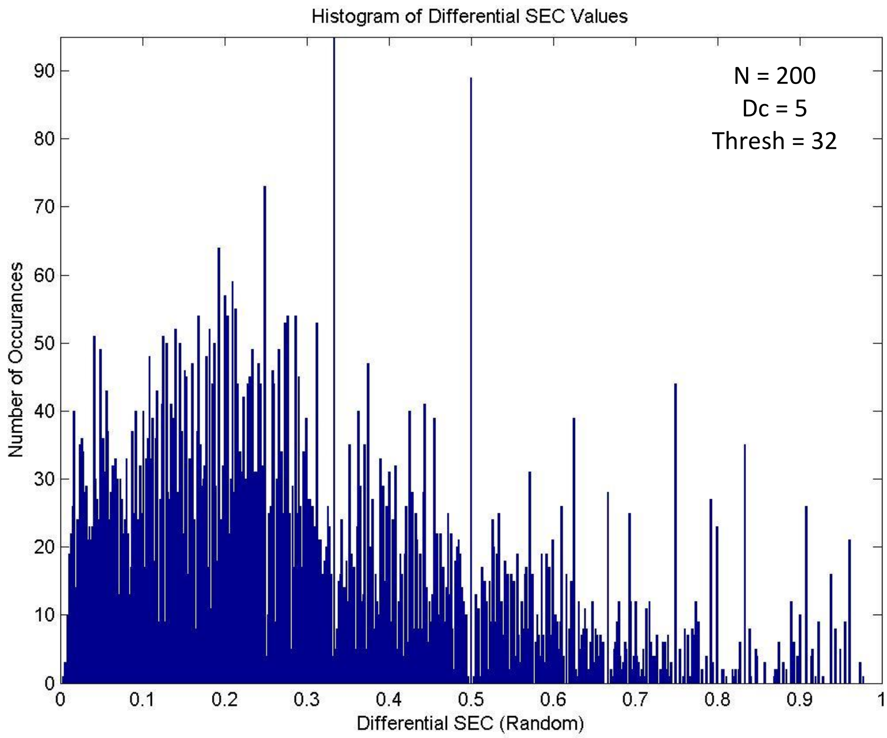
# Appendix II: dSEC Distributions

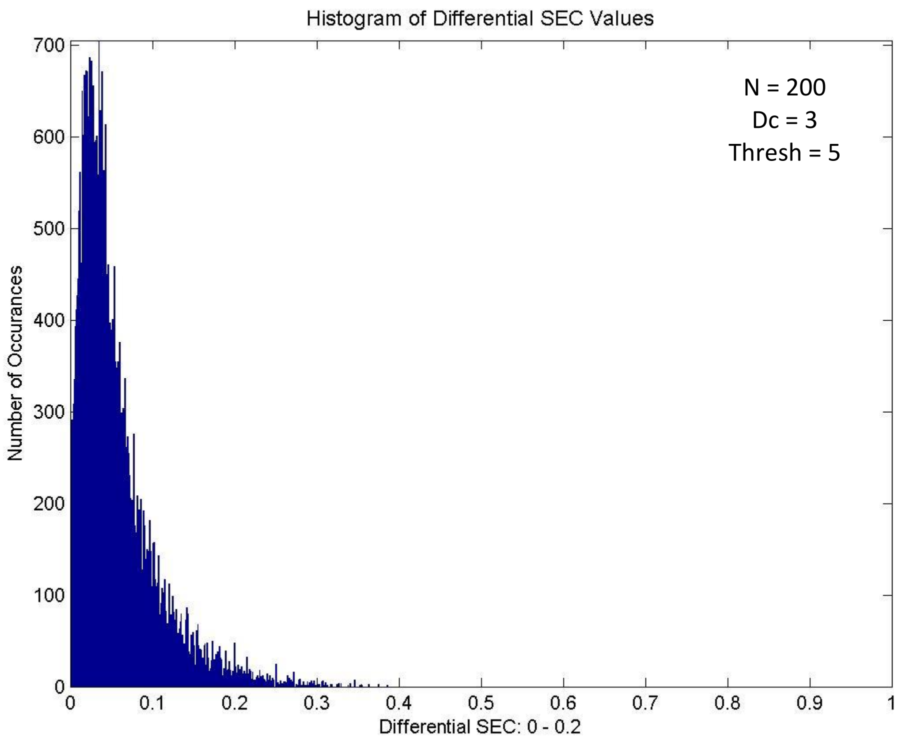


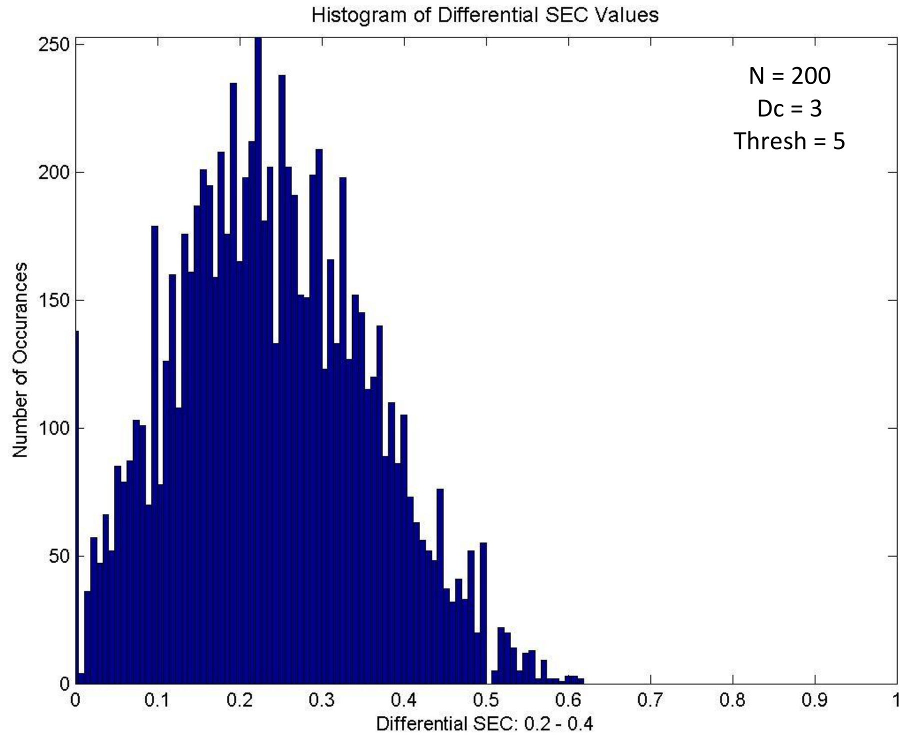



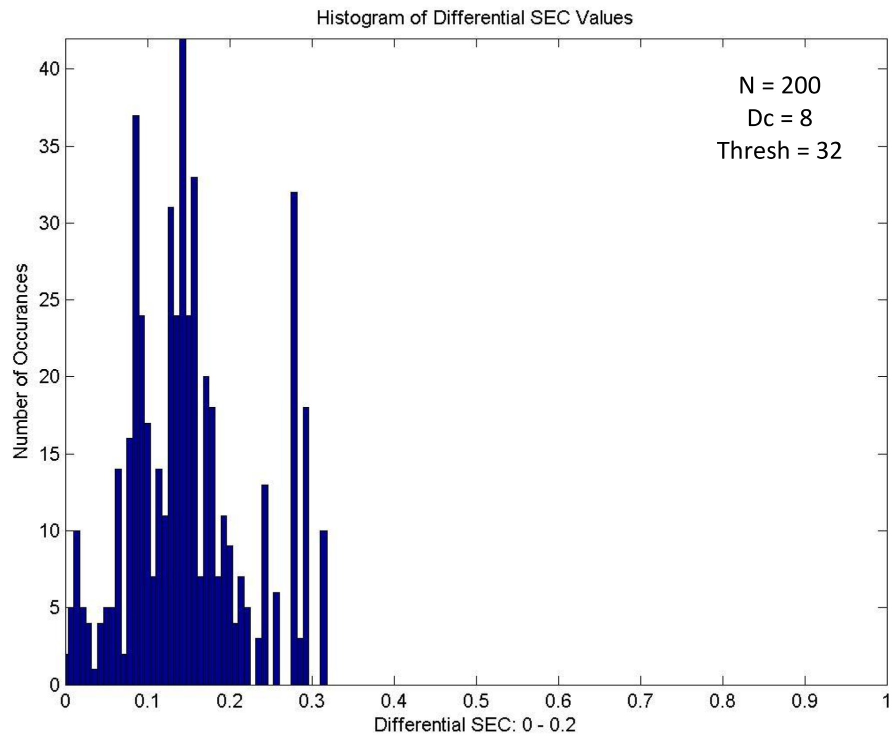

Histogram of Differential SEC Values
N = 200
Dc = 8
Thresh = 10
Number of Occurances
Differential SEC (Random)
0
50
100
150
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

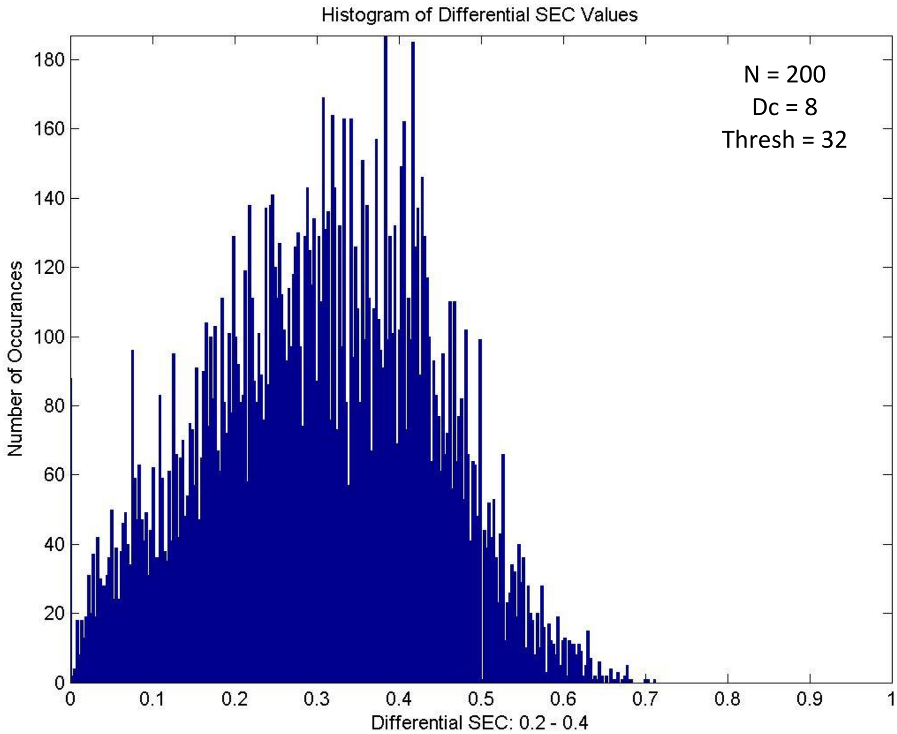

Histogram of Differential SEC Values
N = 200
Dc = 5
Thresh = 5
Number of Occurances
Differential SEC (Random)
0
50
100
150
200
250
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

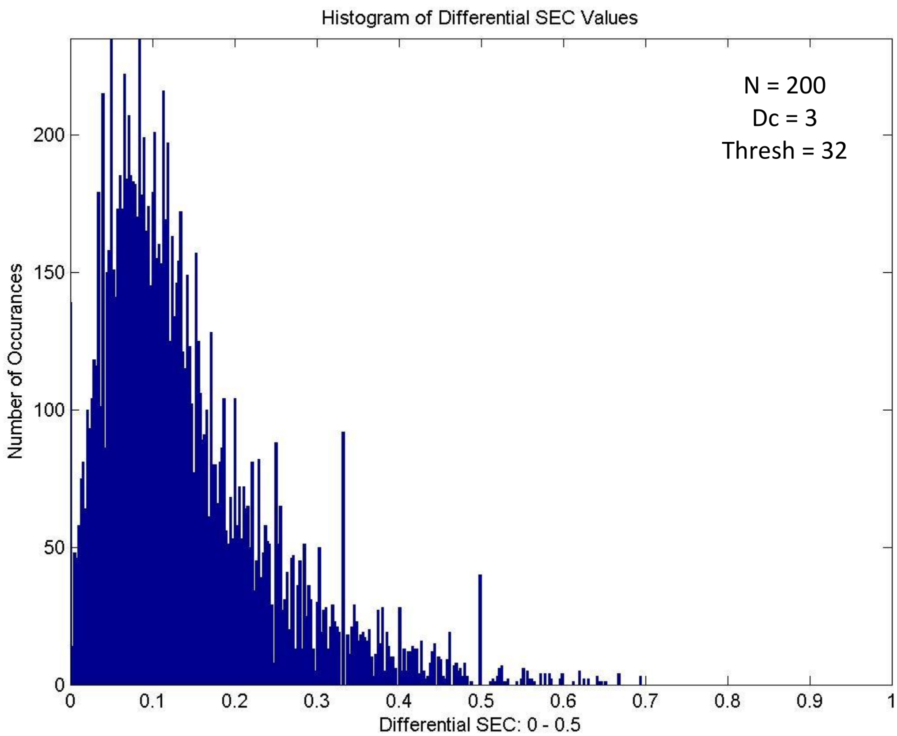

Histogram of Differential SEC Values
N = 200
Dc = 5
Thresh = 32
Number of Occurances
Differential SEC (Random)

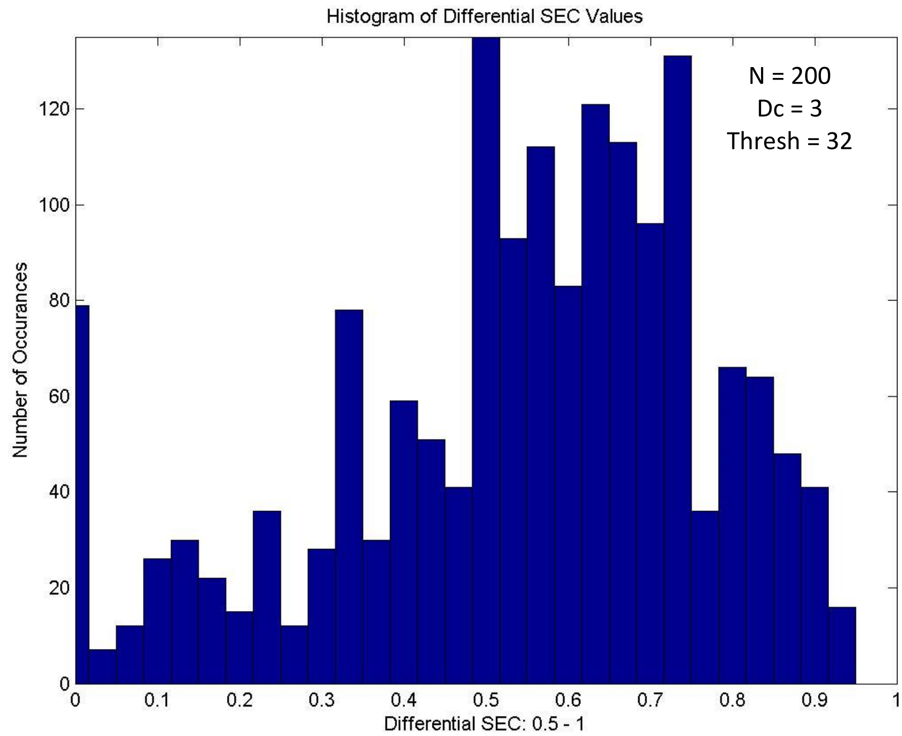

Histogram of Differential SEC Values
N = 200
Dc = 3
Thresh = 5
Number of Occurances
Differential SEC: 0 - 0.2

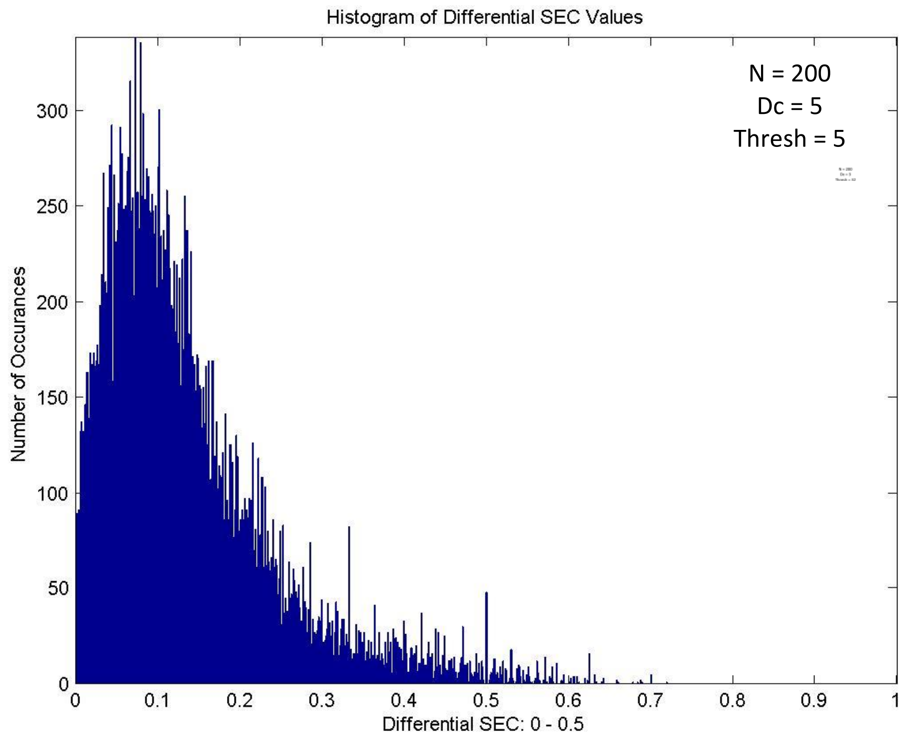

Histogram of Differential SEC Values
N = 200
Dc = 3
Thresh = 5
Number of Occurances
Differential SEC: 0.2 - 0.4
0
50
100
150
200
250
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

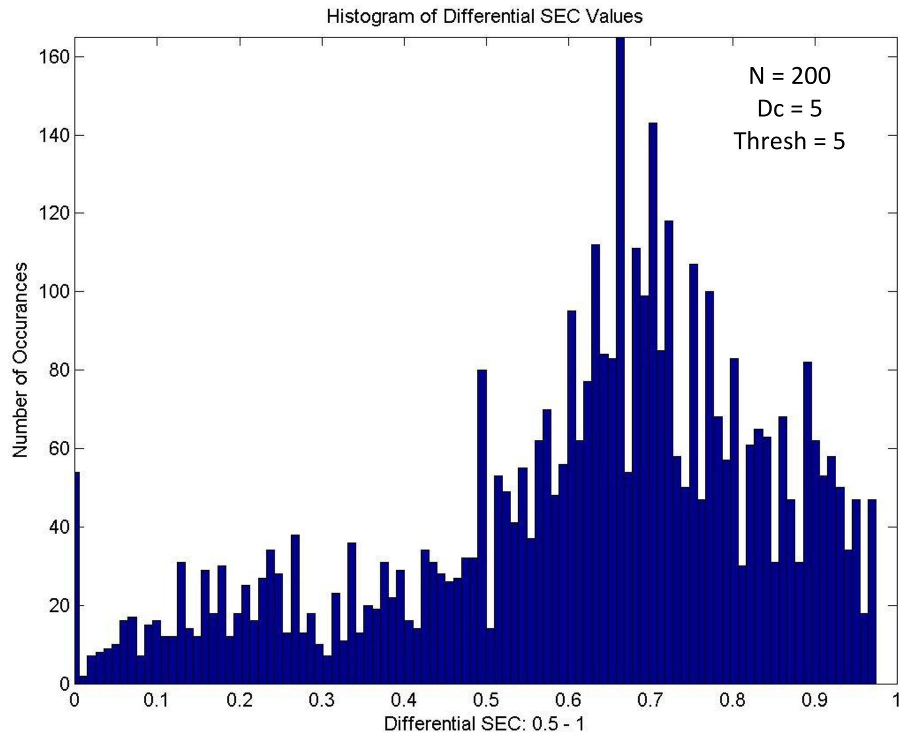

Histogram of Differential SEC Values
N = 200
Dc = 8
Thresh = 32
Number of Occurances
Differential SEC: 0 - 0.2
0
5
10
15
20
25
30
35
40
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

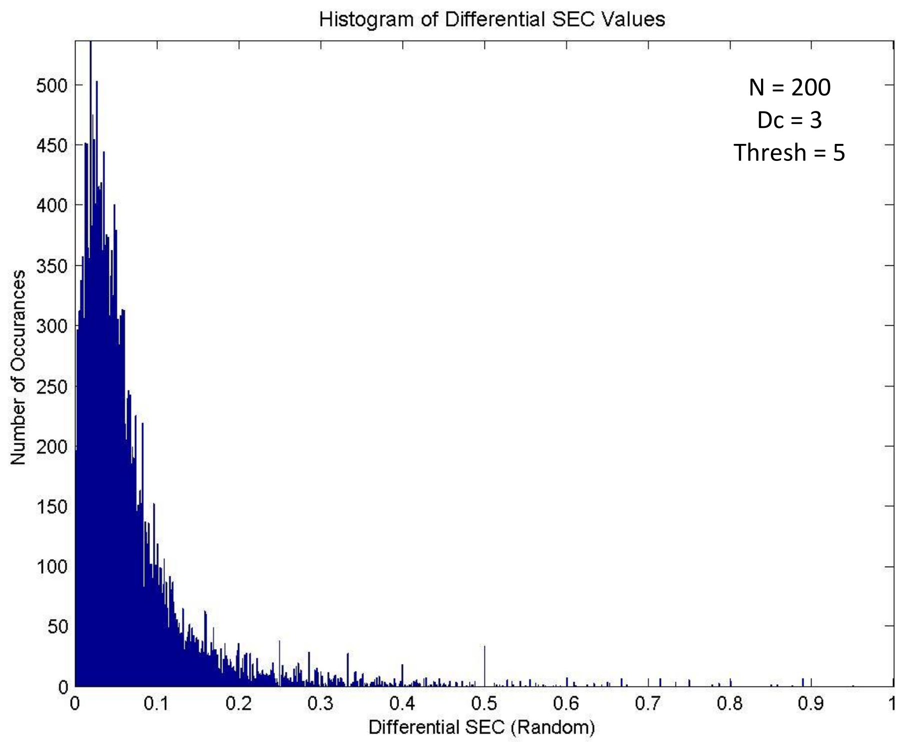

Histogram of Differential SEC Values
N = 200
Dc = 8
Thresh = 32
Number of Occurances
Differential SEC: 0.2 - 0.4
0
20
40
60
80
100
120
140
160
180
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Histogram of Differential SEC Values
N = 200
Dc = 3
Thresh = 32
Number of Occurances
Differential SEC: 0 - 0.5
0
50
100
150
200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Histogram of Differential SEC Values
N = 200
Dc = 3
Thresh = 32
Number of Occurances
Differential SEC: 0.5 - 1


Histogram of Differential SEC Values
N = 200
Dc = 5
Thresh = 5
Number of Occurances
Differential SEC: 0 - 0.5


Histogram of Differential SEC Values
N = 200
Dc = 5
Thresh = 5
Number of Occurances
Differential SEC: 0.5 - 1


Histogram of Differential SEC Values
N = 200
Dc = 3
Thresh = 5
Number of Occurances
Differential SEC (Random)

# Appendix III: Matlab Codes

Pre-processing and Data Loading Code:

```
%   Loads data stored in T_symm-2 and stores in a sparse matrix
%   Preprocessing also takes place
%       Self Edges are given a value of zero
%       Inconsistent strengths averaged
%%

tic;
load('T_symm-2.mat');
[a,b] = size(T_symm);
[c,d] = size(coords);
X = [T_symm(:,1)];
Y = [T_symm(:,2)];
S = [T_symm(:,3)];
X = double(X);
Y = double(Y);
S = double(S);
B = sparse(X,Y,S);
[e,f] = size(B);
display('Sparse Matrix A Created');
toc; display(' ');

tic;
for x = 1:e;
    B(x,x) = 0;
end
display('Self Edges Have Zero Strength');
toc; display(' ');

tic;
C = B.';
D = B + C;
A = D./2;
display('Incosistent Strengths Have Been Averaged');
toc; display(' ');
```

```
%   Loads data stored in Coords.mat into a matrix
%   Grabs the size of the matrix and stores into a and b
%%
tic;
load('Coords.mat');
[a,b] = size(coords);
for x = 1:a;
    xyz(x,1)=coords(x,1);
    xyz(x,2)=coords(x,2);
    xyz(x,3)=coords(x,3);
end
display('Coordinates created');
toc; display(' ')
```

Calculating Self-Edge Correlation Code:

```
function glob = gamma_n(A,xy,n,dc,thresh);
%%
%   Inputs:    A:          Adjacency matrix in sparse form
%              xy:         x and y (and z) coordinates of matrix A
%              n:          Central node
%              dc:         Maximum distance at which a neighbor is local,
%                          after which it is global
%              thresh:     Minimum strength to be considered connected
%
%   Outputs:   glob:       A set of nodes global to node n
%
%%

[row col] = find(A);                        % grab row and column of A

glob = [];                                  % Initialize global nodes
colind = find( col == n );                  % Grab indices where column is n
colind = colind'; row = row';               % Transpose vectors into lists

for x = colind(1:end);                      % Run through nodes connected to n
    y = row(x);                             % Finds node from index
        if A(n,y) >= thresh                 % Check that strength is above threshold
            if dc <= pyth_dist_three((xy(n,1))-(xy(y,1)),(xy(n,2))-(xy(y,2)),(xy(n,3))-(xy(y,3)));
                glob = [glob y];            % Add to global if node is global
            end
        end
end
```

```
function [loc_avg num_loc] = phi_m(A,xy,glob,dc);
%%
%   Inputs:     A:          Adjacency matrix in sparse form
%               xy:         x and y (and z) coordinates of matrix A
%               glob:       Single node global to node n or set of
%                           connected nodes global to node n
%               dc:         Maximum distance at which a neighbor is local,
%                           after which it is global
%
%   Outputs:    num_loc:    Returns number of local nodes (also connected to
%                           to m) respectively to each node m in the set glob
%               loc_avg:    Returns average of num_loc, which will be num_loc
%                           if glob is a single node
%
%%

loc = [];                                   % Initiate set local to gather local nodes
num_loc = [];                               % Initiate num_loc to gather # of local nodes
for x = glob;                               % Loop x through all set glob
    for y = glob;                           % Loop y through all set glob
        if x ~= y;                          % Eliminate self pairing
            if dc > pyth_dist_three((xy(x,1))-(xy(y,1)),(xy(x,2))-(xy(y,2)),(xy(x,3))-(xy(y,3)));
                loc = [loc y];              % Add local nodes to set of local nodes
            end                             % Loc is a set of nodes local to a node m
        end
    end
                                            % Determine the # of local nodes
    num_loc = [num_loc length(loc)];        % Add the length of loc to set num_loc
    loc = [];                               % Reset the set loc and repeat
end

loc_avg = sum(num_loc)/length(num_loc);     % Average the set of num_loc
```

```
function sec = delta_sec(A,xy,n,dc,thresh)
%%
%   Inputs:     A:          Adjacency matrix in sparse form
%               xy:         x and y (and z) coordinates of matrix A
%               n:          Central node
%               dc:         Maximum distance at which a neighbor is local,
%                           after which it is global
%               thresh:     Minimum strength to be considered connected
%
%   Outputs:    sec:        Returns the value of the metric sec around a node
%                           n, which can be interpreted as the probability of
%                           randomly selecting two nodes from the same
%                           spatial group.  The inverse can be interpreted
%                           as the number of spatial groupings
%
%%
    glob = gamma_n(A,xy,n,dc,thresh);       % Find the set nodes global to n
    if length(glob)>=1;                     % Run if there exists nodes global to n
        loc_avg = phi_m(A,xy,glob,dc);      % Compute phi(glob)
        sec = (1 + loc_avg)/(length(glob));% Calculate the metric sec
    else;
        sec = 0;                            % Return zero if n has zero global nodes
    end
```

```matlab
function [avg_sec T] = avg_sec(A,xy,dc,thresh,N);
%%
%   Inputs:     A:          Adjacency matrix in sparse form
%               xy:         x and y (and z) coordinates of matrix A
%               dc:         Maximum distance at which a neighbor is local,
%                           after which it is global
%               thresh:     Minimum strength to be considered connected
%               N:          Number of nodes to sample from matrix A
%
%   Outputs:    avg_sec:    The average sec value over the number of selected
%                           sample nodes N
%               T           One row of a table holding the average SEC value,
%                           distance cutoff, strength threshold, and sample
%                           size
%
%%
tic;
sec_set = [];                               % Initiate set of sec values
if N < num_vertices(xy);                    % Check if sample is whole matrix
    selection = [];                         % Initiate the sample
    for x = 1:N;                            % Loop through the number of samples
        int = randi(num_vertices(xy));      % Randomly select N nodes
        selection = [selection int];
    end
    for x = selection;                      % For each randomly selected node
        sec = delta_sec(A,xy,x,dc,thresh); % Calculate the sec value
        sec_set = [sec_set sec];            % Add sec value to the list sec_set
    end
elseif N == num_vertices(xy);               % If sample is the whole matrix
    for x = [1:num_vertices(xy)];           % Loop through all nodes in A
        sec = delta_sec(A,xy,x,dc,thresh); % Calculate the sec value
        sec_set = [sec_set sec];            % Add sec value to the list sec_set
    end
delta_secs_total = [];
globs = [];

for p = set;
glob = gamma_n(A,xy,p,dc,thresh);
globs = [globs length(glob)];
[loc_avg num_loc] = phi_m(A,xy,glob,dc);
delta_secs = [num_loc]/(length(glob));
delta_secs_total = [delta_secs_total delta_secs];
end

xbins=[0:(1/max(globs)):max(delta_secs_total)];
figure();
hist(delta_secs_total,xbins);
title('Histogram of Differential SEC Values','FontSize', 11);
xlabel(['Differential SEC: ' num2str(low_sec) ' - ' num2str(high_sec)],'FontSize', 10);
ylabel('Number of Occurances','FontSize', 10);
axis([0 1 0 inf]);
saveas( gcf, name , 'jpg' );
[counts centers] = hist(delta_secs_total,xbins);
```

Calculating dSEC and dSEC Distribution Code:

```
function [counts centers] = diffsec2(A,xy,dc,thresh,N);
%%
%   Inputs:    A:          Adjacency matrix in sparse form
%              xy:         x and y (and z) coordinates of matrix A
%              n:          Central node
%              dc:         Maximum distance at which a neighbor is local,
%                          after which it is global
%              thresh:     Minimum strength to be considered connected
%
%   Outputs:   Histogram:  Histogram of the diff_secs for all m global to n
%              Counts:     Number of each diff_sec value
%              Centers:    The center values of each bin of diff_secs values
%
%%

delta_secs_total = [];
globs = [];

for x = 1:N;                                 % For N sampled nodes
int = randi(num_vertices(xy));               % Select a random number
glob = gamma_n(A,xy,int,dc,thresh);          % Find its global neighbors
globs = [globs length(glob)];
[loc_avg num_loc] = phi_m(A,xy,glob,dc);% Find phi values
delta_secs = [num_loc]/(length(glob));       % compute fractions of dSEC
delta_secs_total = [delta_secs_total delta_secs];
end

xbins=[0:(1/max(globs)):max(delta_secs_total)];
figure();
hist(delta_secs_total,xbins);
title('Histogram of Differential SEC Values','FontSize', 11);
xlabel('Differential SEC (Random)','FontSize', 10);
ylabel('Number of Occurances','FontSize', 10);
[counts centers] = hist(delta_secs_total,xbins);
```

```
function [counts centers] = dsec_distr(A,xy,dc,thresh,N,low_sec,high_sec,name);
%%
%   Inputs:    A:         Adjacency matrix in sparse form
%              xy:        x and y (and z) coordinates of matrix A
%              n:         Central node
%              dc:        Maximum distance at which a neighbor is local,
%                         after which it is global
%              thresh:    Minimum strength to be considered connected
%              low_sec:   Lower limit on SEC values for selection pool
%              high_sec:  Upper limit on SEC values for selection pool
%              name:      String of file name to save distribution as .png
%
%   Outputs:   Histogram: Histogram of the diff_secs for all m global to n
%              Counts:    Number of each diff_sec value
%              Centers:   The center values of each bin of diff_secs values
%
%%
set = [];
while length(set) < N;
    int = randi(num_vertices(xy));
    sec = delta_sec(A,xy,int,dc,thresh);
    if sec >= low_sec && sec <= high_sec;
        dbl = ismember(int,set);
        if dbl == 0;
            set = [set int];
        end
    end
end

delta_secs_total = [];
globs = [];

for p = set;
glob = gamma_n(A,xy,p,dc,thresh);
globs = [globs length(glob)];
[loc_avg num_loc] = phi_m(A,xy,glob,dc);
delta_secs = [num_loc]/(length(glob));
delta_secs_total = [delta_secs_total delta_secs];
end

xbins=[0:(1/max(globs)):max(delta_secs_total)];
figure();
hist(delta_secs_total,xbins);
title('Histogram of Differential SEC Values','FontSize', 11);
xlabel(['Differential SEC: ' num2str(low_sec) ' - ' num2str(high_sec)],'FontSize', 10);
ylabel('Number of Occurances','FontSize', 10);
axis([0 1 0 inf]);
saveas( gcf, name , 'jpg' );
[counts centers] = hist(delta_secs_total,xbins);
```