# Brain Network Analysis and Matrix Theory

## Abstract

Utilizing matrix theory to model networks in the brain, we can detect the most influential nodes in the neural network. We are able to accomplish this by analyzing the structure of connectivity matrices and their associated eigenpairs. In this paper, we will use matrix theory to describe basic network types, demonstrate how to model neuroanatomy and brain networks, and define node centrality.

## Contributing Authors:

Ronald Tang
Robert Aroutiounian
Eliyahu B. Addess

## Project Outline:

1. Explain basic neuroanatomy and how it gives rise to networks.
2. Describe the basic network types (undirected/ directed, binary/weighted).
3. Explain how we can succinctly capture these networks with a connectivity matrix (and define the adjacency matrix).
4. Define node degree and demonstrate how to calculate it using the adjacency matrix.
5. Present the issue of node centrality and drawbacks of defining it purely based on node degree.
6. Derive eigenvector centrality and explain how it offers a more accurate definition of node centrality.

## References:

Byrne, J.H. (ed.) *Neuroscience Online: An Electronic Textbook for the Neurosciences*
http://nba.uth.tmc.edu/neuroscience/
Department of Neurobiology and Anatomy
McGovern Medical School at The University of Texas Health Science Center at Houston (UTHealth)

Fornito, A., Zalesky, A., & Bullmore, E. T. (2016). *Fundamentals of Brain Network Analysis*. Elsevier.
Newman, M. E. J. (2018). *Networks*. Oxford University Press.

# Brain Network Analysis and Matrix Theory

## Section 1: Neuroanatomy and Networks[1] - Robert Aroutiounian

The brain can be represented as spatially distributed neural elements. These elements "could be individual neurons, specific neuronal populations, or large-scale brain regions" (Fornito, 2016, 89). The *neuron* is the fundamental brain cell. Neurons are composed of several regions. The *soma* is the main cell body. The *dendrites* extend out of the soma and receive information from other neurons. The *axon* is a relatively long protrusion from the soma which branches off into *synapses*. The neuron relays information to other neurons via the axon and synapses. "It has been estimated that one neuron can receive contacts from up to 10,000 other cells. . . Similarly, any one neuron can contact up to 10,000 postsynaptic cells"(Byrne, 1997). Since there are approximately 100 billion neurons, this leads to an incredibly complex network.[2]

One challenge faced is an elegant way to mathematically represent these neural elements and their connections. Fornito (2016) states that this is known as **connectomics**, a comprehensive map of connections between anatomically distributed neural elements. These connections can get very large, as any network with N nodes can have a possible number of connections in the order of $N^2$. One way to capture this mapping is through the use of a two-dimension matrix, where each row and column correspond to a different node; this is referred to as the connectivity matrix. Before delving into the connectivity matrix, we will provide a brief introduction to network analysis.

## Section 2: Basic Network Types - Ronald Tang

The branch of mathematics that deals with network analysis is known as graph theory. Graph theory defines networks in terms of *nodes* and *edges*. Nodes are the elements of the network, while edges are the connections between them. There are two characteristics an edge can have: undirected versus directed or unweighted versus weighted edges. These two characteristics can determine whether one path through a graph is more efficient to move through than the others; in terms of mapping a brain, it could determine how important a neuron is in a network.

First, we will introduce the idea of undirected versus directed edges.[3] Graphs with undirected edges have edges that can go in both directions; in other words, there is a connection from node i to node j and vice versa. Given an connectivity matrix undirected edges in matrix $\mathbf{A}$ would be represented by a non-zero value at $A_{ij}$ as well as at $A_{ji}$, which would represent a connection from node i to node j and a connection from node j to node i. An undirected graph will have a symmetric matrix. On the other hand, graphs with directed edges

[1] Sections 1-5 are primarily based on Fornito et al., 2016, except where otherwise stated.
[2] Neuroanatomy description based on Byrne, 1997.
[3] The reader may find it helpful to reference the visual at the end of the paper while reading this section.

have edges that can go in only one direction. This means that a connection from node i to node j does not necessarily mean that there is a connection from node j to node i. On an connectivity matrix $\mathbf{A}$, a non-zero value at $A_{ij}$ could represent an edge from node i to node j or an edge from node j to node i. We adopt the convention that an entry at the i,j$^{th}$ element represents an edge *from* node j *to* node i.

Next, we will describe unweighted versus weighted edges. In an unweighted graph, each of the edges have the same value or cost of traversing the edge. When represented by an connectivity matrix, each of the non-zero values representing an edge in a graph would be a 1. Since the graph is represented by a matrix of 1's and 0's, another name for this type of connectivity matrix is a binary connectivity matrix. A weighted graph, on the other hand, will have different values or costs for traversing an edge. When represented by an connectivity matrix, the cost of traversing an edge will be the value representing that edge. For example, if the cost of traversing an edge from node i to node j was 5, the connectivity matrix $\mathbf{A}$ would have the value 5 at $A_{ij}$. Now that we have defined the distinct types of networks, we can describe in more detail how we can represent a network with an connectivity matrix.

## Section 3: The Connectivity Matrix - Robert Aroutiounian

The connectivity matrix is able to computationally capture the relationship of nodes and the effect they have on one another. These are square NxN matrices where each row and column represent a unique network node. Furthermore, according to Fornito (2016) the diagonal of the matrix captures intrinsic properties of the node itself. In neural networks, these diagonal values can be used to capture distinct kinds of neurons; in mesoscopic or macroscopic brain networks, it can be used for the function of different brain regions; in functional connectivity networks, it could reflect dynamics in each brain region. The off-diagonal is how nodes impact each other and this can then be used to represent the type and strength between the different regions.

In our connectivity matrix, is it not the case that $C_{ij}$ values will always equal $C_{ji}$ values. One node can affect another way differently than how it gets affected by the other. Moreover, it is important to understand how the connectivity matrix was formed, as in some cases "efferent projections are listed down the columns and afferent projections along the rows of the matrix; in other cases, this convention is reversed" (Fornito, 2016, 91). Cases where only one node projects onto another, say we have nodes A and B, and A projects onto B, but not the other way around, then we have a nonzero value for $C_{B,A}$ and a zero value for $C_{A,B}$. In cases where our $C_{ij}$ values will equal our $C_{ji}$ values, this tells us there is a link between the two, but no other information can be drawn such as direction of information flow or how one neural element projects onto another one. This is commonly referred to as a directed graph.

As previously mentioned, one neural element may only project onto another but not vice versa. This can also be captured by difference in connectivity weight where said weight between nodes *i* and *j* are denoted as $w_{ij}$. Depending on the method used to capture the weight, this can represent wide variations in the brain mapping. In one case, "if we use electron microscopy to reconstruct each synaptic connection between neurons, the values will range between zero (reflecting no connection) and an upper limit that is determined by spatial and physiological

constraints on neuronal wiring" (Fornito, 2016, 93). Different methods will have different ranges in the weight, some can range from 0 to 1, others from -1 to 1.

With any data collection, an important step is data cleaning. Since various factors such as resolution, size, image quality, etc. can affect how the data on the brain is collected, some form of thresholding is applied to get rid of any noise. Before any cleanup is done, the matrix is called the connectivity matrix, and after it goes through the filtering, the output is called the **adjacency matrix**. In some cases, these two matrices can be the same. In practice, the adjacency matrix is used for the rest of the research, as it doesn't have any noise that can result in inaccurate results.

## Section 4: Node Degree - Ronald Tang

The degree of a node in a binary network is defined as the number of edges that are connected to that node. A loop, or when a node is connected to itself, adds a node degree of 2 when it is in an undirected network. We can calculate the node degree by counting the non-zero values in the row representing the node (adding an additional degree if there is a loop at that node). For example, if we are trying to find the node degree of node i, we would count the number of non-zero values of $A_i$, where i is the i$^{th}$ row of connectivity matrix A. If node i had a loop and was connected to itself, we would add one more degree to find the correct node degree.

For directed networks, the in-degree of a node is defined as the number of edges that are pointing towards that node. To calculate the in-degree of a node in a directed network, we count the number of non-zero values in the respective row of the adjacency matrix. For example, if I wanted to calculate the in-degree of node i, I would count all of the non-zero values on the i$^{th}$ row of adjacency matrix A, since a value at the i$^{th}$ row represents an edge pointed towards the i$^{th}$ node.[4]

Inversely, the number of edges that are pointing out of that node is called an out-degree. To calculate the out-degree of a node in a directed network, we count the number of non-zero values in the respective column of the adjacency matrix. For example, if I wanted to calculate the out-degree of node j, I would count all of the non-zero values on the j$^{th}$ column of adjacency matrix A, since a value at the j$^{th}$ column represents an edge leading out of the j$^{th}$ node.

## Section 5: Centrality - Eliyahu B. Addess

A fundamental goal of brain network analysis is to measure the influence a given node has on the network in general. Equivalently, we seek to determine which nodes exert the greatest influence on the network. One area where this knowledge can be useful is to predict the effect of node failure. For example, if we have determined that a particular region of the brain is highly influential, we can expect that damages to this region will have a broad effect on neurological functionality. This can have clinical implications as well, as clinicians must take extra precautions not to damage such a region during surgery and other procedures.

[4] See the visual at the end of the paper.

In network science, the extent to which a node can influence (or be influenced by) other nodes is known as **centrality**. In order to calculate node centrality, we must first provide a mathematical definition of centrality. In fact, several definitions of centrality have been suggested, each providing unique advantages and drawbacks. Here we will present two definitions of centrality which apply to binary networks.

The most simple, and perhaps most intuitive, approach is to define the centrality of a node as being equal to its degree. So-called **degree centrality** implies that the more edges a particular node has, the more influential it is. Conversely, degree centrality interprets a node with few edges as having little impact on the network.

As discussed earlier, we can use the adjacency matrix to compute node degree. Therefore for undirected networks we define the degree centrality of the i[th] node as

$$C_D(i) = \sum_{j \neq i} A_{ij}$$

.

In directed networks, each node has an in-degree and out-degree. Depending on our research objective, both may be informative measures of centrality.[5] For example, a neural node with a high in-degree might indicate a region responsible for processing a large volume of information, comparable to a computer's CPU. Using our matrix terminology, we define the in-degree centrality of the ith node and the out-degree centrality of the jth node as

$$C_D^{in}(i) = \sum_{j \neq i} A_{ij} \qquad C_D^{out}(j) = \sum_{i \neq j} A_{ij}$$

.

There is a significant drawback to using node degree as a measure of influence. Consider the situation where we have two nodes, i and j, each with degree 2. Suppose that the two neighbors of node i have degree 100, and the two neighbors of node j have degree 1. According to degree centrality, node i and node j exert equal influence on the network. But this is clearly not true. With just two steps, node i can reach far more network elements than node j can! This example shows that degree centrality is flawed in that it only takes the connectivity of a given node into account and not the connectivity of its neighbors, which certainly do contribute to the influence of a given node. Can we find a definition of node centrality which considers both the degree of the node and the degree of its neighbors? Using matrix theory, we can.

## Section 6: Eigenvector Centrality[6] - Eliyahu B. Addess

Consider a binary, undirected network. Let's define the centrality score $x_i$ of node i as being proportional to the sum of the centrality scores of its neighbors. That is

$$x_i = K^{-1} \sum x_j,$$

where the $x_j$ are the centrality scores of nodes j which are neighbors of node i, and $K^{-1}$ is a constant of proportionality. The reason we chose $K^{-1}$ will become apparent as we proceed. According to this definition, a node will receive a high centrality score if it has many connections, even if its neighbors have relatively low scores. Moreover, if a node's neighbors have high

[5] As pointed out by Newman, 2018.
[6] This section is based on Newman, 2018.

centrality scores, then the node will also be awarded high centrality, *even if it has few neighbors*. This fits with our intuition about the importance of a given node, and solves the problems attendant to degree centrality.

However, at first glance this definition seems absurd. In order to calculate the centrality of any given node, we must first know the centrality of each of its neighbors. Thus it seems we would never be able to begin to figure out any node's centrality! Here though we can employ the beauty of matrix theory.

First, we rewrite the above definition in terms of the adjacency matrix as

$$x_i = K^{-1} \sum_{j=1}^{n} A_{ij} x_j$$

To explain, the term $A_{ij}$ equals 1 if the j[th] node is connected to node $i$ and equals 0 if it is not connected. Thus the term $A_{ij}$ ensures that we only sum the over the centrality scores of the neighboring nodes. If we define the vector $\mathbf{x}$ as $[x_i]$ for $1 \leq i \leq n$, then we can write this equation as a matrix-vector multiplication. That is:

$$\mathbf{x} = K^{-1}\mathbf{A}\mathbf{x}$$

Multiplying on the left by $K$ gives us:

$$K\mathbf{x} = \mathbf{A}\mathbf{x}$$

In other words, $\mathbf{x}$ is an eigenvector of $\mathbf{A}$ corresponding to the eigenvalue $K$! So computing centrality comes down to a straightforward eigenvector calculation. Hence, this definition of centrality is known as **eigenvector centrality**.

There is still one loose end. The adjacency matrix has multiple eigenpairs. In order to unambiguously define the centrality vector, we must select a specific eigenpair. It is more convenient to have all non-negative centrality scores. According to the *Perron-Frobenius* theorem[7], for a matrix $\mathbf{A}$ whose entries are non-negative, the only eigenvector with all elements non-negative is the one corresponding to the largest eigenvalue, denoted $\mathbf{x}'$. Thus since the adjacency matrix has all non-negative entries, and we would like all the centrality scores to be non-negative, $\mathbf{x}'$ is a natural choice for the centrality vector.

Now, we have still not fixed the centrality vector completely, because any scalar multiple of $\mathbf{x}'$ is also an eigenvector of the largest, or leading, eigenvalue $K'$.[8] However, we may leave this choice arbitrary because we are primarily interested in the most important nodes *within* the network. Any multiple of $\mathbf{x}'$ will preserve the relative difference between centrality scores and thus sufficiently defines the centrality vector.

In short, we have discovered a practical definition of node centrality, namely **eigenvector centrality**. According to this definition, the centrality score of the i[th] node is the i[th]

---

[7] Newman, 2018 points to Strang, G., *Introduction to Linear Algebra*, Wellesley Cambridge Press, Wellesley, MA (2009) for a discussion and proof of *Perron-Frobenius*. It seems that most proofs of this theorem rely on Brouwer's Fixed Point Theorem which is beyond the scope of this course.

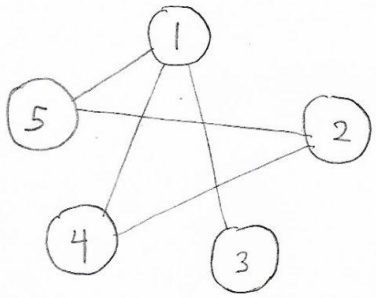
[8] To see this, consider A(cx') for any scalar c.
A(cx') = cAx' = cK'x = K'(cx)

element of the the eigenvector $\mathbf{x}'$ corresponding to the leading eigenvalue of the adjacency matrix. This definition can be generalized to directed networks, with some refinement.

In conclusion, we have demonstrated the necessity to develop a clear mathematical model for neural networks and shown how this can be accomplished using the adjacency matrix. Moreover, we have derived an effective method to compute node centrality using the leading eigenvalue of the adjacency matrix. This is a small foray into the fascinating world of connectomics and we hope that the reader will be inspired to further research in this area.

# Visual Depiction of Network Types and their Matrices

## Undirected / Binary Network

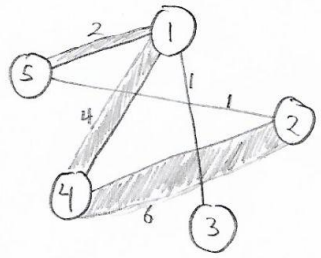



Graph Form

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

Matrix Form
(Note the symmetry)

## Undirected / Weighted Network

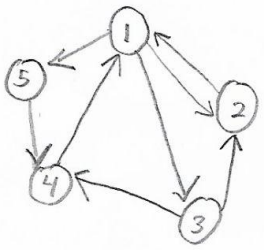



Graph Form

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

Matrix Form
(Note the weighted values)

## Directed / Binary Network




Graph Form

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

Matrix Form
(Note the asymmetry.)