Algorithms for exploring structure in complex networks

Azhar Aleidan

Department of Mathematics and Statistics,

University of Strathclyde,

Glasgow, U.K.

A thesis submitted for the degree of

Doctor of Philosophy

September 18, 2023

Copyright Declaration

Acknowledgements

First and foremost, I want to thank God, for his continual great blessings on me and for giving me the ability to complete my thesis.

Throughout the writing of this thesis, I have received a great deal of support and assistance from many people. This dissertation would not have been possible without their guidance and help. I would first like to express my deepest gratitude to my supervisor, Dr. Philip Knight, for his continuous support, knowledge, guidance, valuable comments, and advice which have made this work possible. Dr. Knight, thank you for helping me to become a better researcher.

I would also like to give special thanks to my family who endured this long process with me and always offered their support and love. Their belief in me has kept me motivated and my spirits high during this process. I want to thank my parents for everything I have achieved. I want to thank my brothers and sisters for their support. I want to thank my husband, Khaled, who supported me through this journey. I want to thank my children, Abdullah, Sara, Dana, Yara, and Ahmed for their patience with my preoccupation.

I would also like to thank Princess Nourah bint Abdulrahman University for providing the funding for me to complete my PhD and for the University of Strathclyde for accepting me into this program and providing me with the resources to successfully complete it.

Finally, thank God, for letting me through all the difficulties to end this thesis.

Abstract

As real-world networks grow increasingly complex, new and adaptable methods are required to analyse and understand the defining features of these networks. To find and exploit these unique network features, current methods, primarily for random model generation and detection of anti-communities, are explored, tested, and adapted with a wide range of networks from disparate fields including neuroscience, ecology, geology, social ecology, linguistics, network theory, psychology, microbiology, gene sequencing, business corporations, and sociology. In particular, to better approximate typical structural features of real-world networks, Erdős–Rényi and scale-free network random models are modified by adding select subgraphs to match real-world networks with the aim of improving their usefulness in network analysis.

Before looking for anti-communities we first look at ways to partition graphs into communities. As well as generic techniques such as local improvement methods and spectral partitioning, a number of specialised methods are studied. These include link centrality, similarity, communicability, optimisation, and the Louvain method.

We then look at topics associated with near bipartivity in real world graphs. Particular attention is given to the measurement of bipartivity and anti-communities, and their definitions are broadened, refined and tested so that many more networks that demonstrate varying degrees of bipartivity can be analysed using those methods. We prove new theoretical results showing how widely measures can differ and then look to determine the best measures to use, as well as the most effective structure-revealing algorithms. Thorough testing involves nearly one hundred real-world networks including some which have a known tendency for bipartivity (such as airlines networks, and fullerene graphs) as well as near-bipartite graphs based on random trees (including those generated from Prüfer sequences, and other artificially constructed examples.

We are able to give conclusions about the measures and methods that should be employed in practice.

Contents

Chapter 1

Introduction

1.1 Introduction

According to the Oxford English Dictionary the use of the word ‘network’ dates back to the 16th century being “an arrangement of intersecting horizontal and vertical lines” and “a group or system of interconnected people or things”. When we consider the image of these lines and points of connection between these lines, then any group of items that are related in some way can then form such a system. The world around us is filled with systems that are connected in ways both tangible and intangible. With the general concept of the word having such wide-ranging applications, it should come as no surprise that a quick internet search of the word results in almost 12 billion search results. The applications are seemingly endless from physical connections such as railways, telephone lines and roads to energy connections such as chemical and molecular bonds to social connections such as relationships between friends, families, or coworkers; it is evident and expected that the world around us is composed of related items connected in various ways in order to fulfill vast and varied functions [22].

Networks (known at the time by the synonymous term ”graphs”) were first introduced in the 18th century to try to describe a type of maths where the topology was based on links rather than distances. As related to mathematics, the origin of network theory is said to have begun with a problem encountered by Leonhard Euler, the Seven Bridges of Konigsberg in Prussia (1736). Some would imagine a new type of mathematics arising from the need to solve a serious problem with significant implications, but the beginning of network theory or graph theory can be traced back to high society in Konigsberg, Prussia where the residents entertained themselves with a witty puzzle that involved arranging a route to tour the town but crossing each of the town's seven bridges only once. The puzzle was considered unsolvable by some when Euler addressed the problem mathematically by graphing the problem which required use of a math that considered the "geometry of position" a term which Euler uses but acknowledges was coined by Gottfried Leibniz. Euler wrote an article describing the problem and a general method for solving it. Most notably, he replaced the map of Konigsberg with a diagram of the main features and formulated the problem. In so doing, he not only proved its impossibility but his approach marks the advent of a new branch of mathematics- modern graph (network) theory. Important to note is that Euler's approach to the real-world Konigsberg Bridge Problem could not have been solved with the mathematical tools known at the time. The problem did not require knowledge or measurements of lengths or geometry, only on how they were connected to each other or their topology [22].

In 1878, James J. Sylvester published an article in Nature entitled 'Chemistry and Algebra' wherein he used the term graph to depictions of molecular structure. In, 1912 the relationship between predator and prey was depicted in an ecological network demonstrating that there is a long history of interdisciplinary use of network theory which has been expounded upon by Frank Harary, a mathematician considered to be the father of modern graph theory [22].

Some seminal work on the theory of graphs was done by Erdos and others in the 1950s and their work and others has been used in countless areas of mathematics and physics since then. Subsequent advances in graph theory include discovery of trees in the mid-19th century by Gustav Kirchhoff in electrical networks which were later used by Cayley, Sylvester and Polya in solving molecular problems, discovery of graph algorithms using the maze problem discussed above, the travelling salesman problem in the 1930s by Dantzig, Fulkerson and Johnson at Princeton, the minimum connector problem, and the shortest and longest path problems in the 1940s and 50s by the US Navy [34].

Erdos and Renyi created a random graph model to gain insight into certain properties of graphs. Contemporaneously, Edgar Gilbert also created a random graph model. Over the last few years people have looked in detail at real world networks. In real world networks we find structures which do not necessarily appear in more mathematical models of graphs. These include communities, which are a hugely important topic in the study of social networks. Or more generally, small groups of nodes which make in particular shapes.

In this thesis, we're going to look at various efficient ways of understanding what some of these real-world structures are and how we can derive numerical techniques to find important structures. To start with, we look at various tools for clustering. In the third chapter, we look at how you can generate graphs which have more realistic frequencies of some of the small graphlets which you see in real world networks. Then, we show that by a simple application of adding triangles to standard random network models, we can add a lot more other features that are seen in real world graphs. We then move onto the important topic of bipartivity and approximate bipartivity. We show that the multiple different definitions of bipartivity can lead to very different understandings of when a graph is bipartite or not bipartite. In particular, we are able to come up with a notion of local and approximate bipartivity. We then exploit these ideas to look at spectral techniques for finding anticommunities which follow in the spirit of finding spectral communities. There are multiple different ways of doing this and we give exhaustive tests and show that using either the normalized or the sign of the Laplacian seems to give the best results especially if it's combined with a very simple method of local improvement. Here in the exact opposite of what we want to do in finding communities we look for ways of reducing modularity as much as possible.

Original material in Chapters 4 and 5 has been published in [3].

1.2 Definitions

We introduce some basic definitions and theorems for networks.

Definition 1.2.1. A graph (network) G, is a pair (V, E) where V is a set of nodes and E is a set of edges.

- A network G is undirected if E is symmetric (if $(u, v) \in E \Leftrightarrow (v, u) \in E$).
- A network has a loop if it is contain an edge of the form (v, v).
- A network G is a simple graph if E is symmetric and anti-reflexive.

  In our work, we will use a simple graph.

  We introduce the graph G_r in the Figure 1.1. The graph G_r has 6 nodes with 7 edges and the edges set of the graph G_r is

  $E_r = \{(1,2),(1,4),(2,1),(2,3),(2,4),(2,6),(3,2),(3,4),(4,1),(4,2),(4,3),(4,5),(5,4),(6,2)\}$

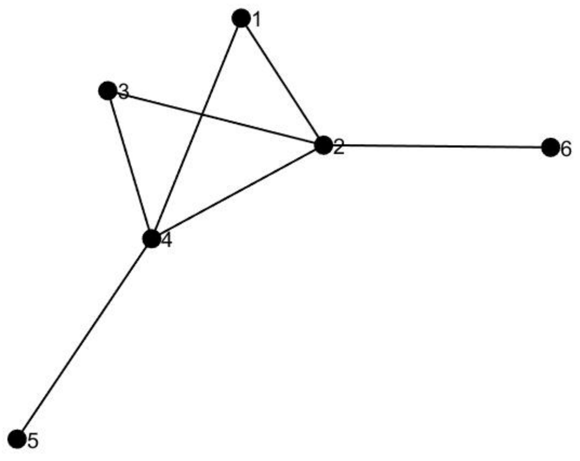


Figure 1.1: The graph G_r.

- $G_k = (V_k, E_k)$ is a subgraph of $G(V, E)$ if $V_k \subseteq V$ and $E_k \subseteq V_k \otimes V_k \cap E$.
- If $e = (v_1, v_2)$ is an edge in the network G then we say v_1 is incident to and v_2 is incident from e and v_1 is adjacent to v_2.

- $\bar{G} = (V, \bar{E})$ is the complement of G if $\bar{G}$ is simple graph with the same vertices as G and if $(u, v) \in E \Leftrightarrow (u, v) \notin \bar{E}$.
- The degree of the node is the number of edges incident on it.
- We will use $k_{\max}$ to denote the maximum degree of any node of a simple network.
- If a node has degree 0 then we say the node is an isolated node.

Example 1.

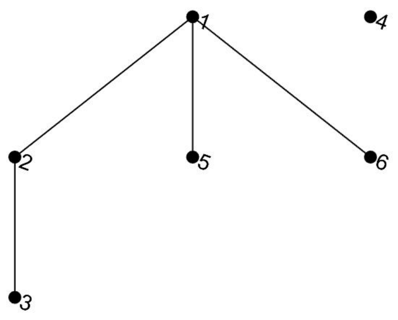


Figure 1.2: Example of degree.

The degree of the nodes in the network in Figure 1.2 are 3, 2,1,0,1 and 1. The maximum degree is 3 at node 1 and the node 4 is an isolated node.

- The complete graph K_n with n nodes is a graph with an edge between every distinct node. Hence each node has the degree $n - 1$ (see Figure 1.3).

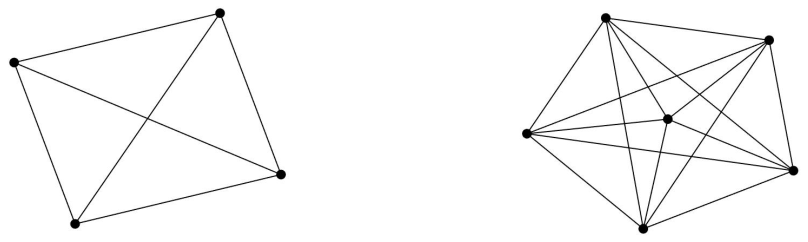


Figure 1.3: The complete graphs K_4 and K_6.

- A network is regular if all nodes have the same degree.
- The null graph N_n is the graph with n nodes and no edges.
- If a graph contains at least one edge then we say the graph is nontrivial.
- The cycle graph C_n is a graph with n nodes which can be ordered so each is connected to only its immediate neighbours. We can get the path graph P_{n-1} with n nodes and $n-1$ edges if we remove an edge from C_n, and we can get a wheel graph W_{n+1} with $n+1$ nodes if we add a node to C_n and connected it with every other node (see Figure 1.4).

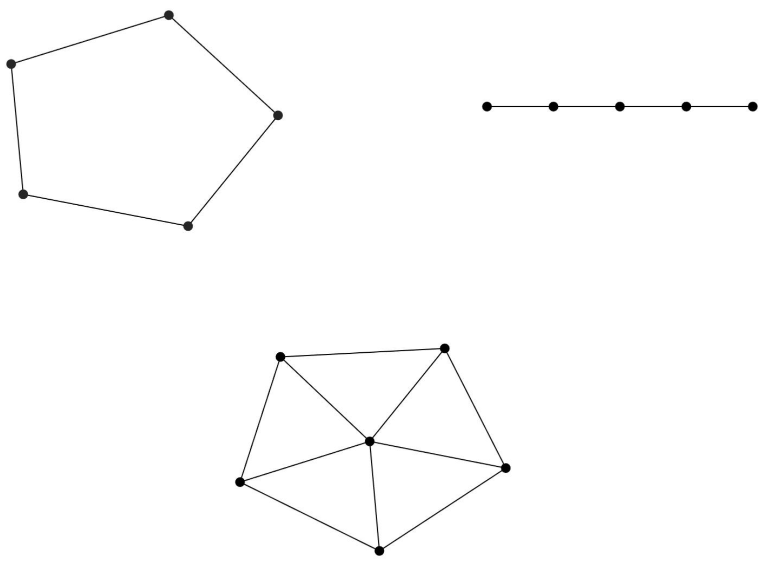


Figure 1.4: C_5, P_4 and W_6.

- The star graph $S_{1,n-1}$ is a graph with a single central node connected to all other $n-1$ nodes and no other edges.

Definition 1.2.2. The adjacency matrix A of the graph $G(V, E)$ is a square matrix $n \times n$ with

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \end{cases}.$$

Definition 1.2.3. A walk in the graph $G(V, E)$ is a series of edges

$$(u_1, v_1), (u_2, v_2), \ldots, (u_p, v_p)$$

where $v_i = u_{i+1} \quad (i = 1, 2, \ldots, p-1)$. The walk is closed if $v_p = u_1$. A walk in which all the edges are distinct is a trail. A trail in which all the u_i are distinct is a path. A cycle is a closed path. A triangle is a cycle of length 3 (C_3).

Definition 1.2.4. If $G = (V, E)$ is a graph with n nodes and m edges ($E = \{e_1, e_2, \ldots, e_m\}$) with $e_i = (u_i, v_i)$, for $1 \leq i \leq m$, $1 \leq j \leq n$

$$b_{ij} = \begin{cases} 1 & \text{if } \; u_i = j \in E \\ -1 & \text{if } \; v_i = j \\ 0 \;\; \text{otherwise} \end{cases},$$

then the matrix $B = (b_{ij})$ is called incidence matrix of the graph G.

Definition 1.2.5. A connected network with no cycles is a tree. A union of trees is a forest.

Definition 1.2.6. A network is bipartite if the nodes of the network $G(V, E)$ can be divided into disjoint sets $V = V_1 \cup V_2$ such that for all $(u, v) \in E$, either $u \in V_1$ and $v \in V_2$ or $u \in V_2$ and $v \in V_1$.

There is a relationship between networks and matrices because we can represent networks with matrices. In this section, we define the determinant of the matrix A, the eigenvalues of A, the eigenvectors of A, and give some definitions of matrices.

Definition 1.2.7. The determinant of the matrix $A \in \mathbb{R}^{n \times n}$, is given by

$$\det(A) = \begin{cases} A & \text{if } \; n = 1 \\ \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(A_{ij}) & \text{if } \; n > 1 \end{cases}$$

for any fixed i, where A_{ij} is the submatrix formed from A by deleting its ith row and the jth column.

Definition 1.2.8. For the square matrix A with dimension n, there exist a scalar value λ and vector $\mathbf{x}$ such that

$$A\mathbf{x} = \lambda\mathbf{x}.$$

We call λ an eigenvalue of A and x an eigenvector.

Definition 1.2.9. The polynomial

$$\det(\lambda I - A) = b_0 + b_1\lambda + b_2\lambda^2 + \cdots + b_n\lambda^n$$

is the (c.p) characteristic polynomial of A and $\det(\lambda I - A) = 0$ is the characteristic equation.

Definition 1.2.10. The set of all eigenvalues is the spectrum of A

$$\sigma(A) = \{\lambda : \det(\lambda I - A) = 0\}$$

and the spectral radius of A is the modulus of the largest eigenvalue

$$\rho(A) = \max_{\lambda\in\sigma(A)} \mid \lambda \mid .$$

Definition 1.2.11. If the adjacency matrix of a simple graph G has distinct eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_k$ with multiplicities $p_1, \ldots, p_k$ respectively then we say that this is the spectrum of G, and we write

$$\sigma(G) = \{[\lambda_1]^{p_1}, [\lambda_2]^{p_2}, \ldots, [\lambda_k]^{p_k}\}.$$

For certain graphs we know $\sigma(G)$ explicitly.

Example 2.

- In the complete graph K_n, the adjacency matrix is $A = E - I$, where E is a matrix of ones. Since an eigenvalue of E is zero with algebraic multiplicity $n-1$ then A has an eigenvalue of -1 of algebraic multiplicity $n-1$ and the remaining eigenvalue is $n-1$ since $Ae = (n-1)e$.

Hence

$$\sigma(K_n) = \{[n-1]^1, [-1]^{n-1}\}.$$

- In the cycle graph C_n, the adjacency matrix is

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 1 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 0 & 1 & 0 & \ddots & & 0 \\ \vdots & 0 & 1 & \ddots & \ddots & \vdots \\ 0 & \vdots & & \ddots & 0 & 1 \\ 1 & 0 & \dots & 0 & 1 & 0 \end{bmatrix}.$$

Suppose that $\omega^n = 1$ and $v = \begin{bmatrix} 1 & \omega & \omega^2 & \dots & \omega^{n-1} \end{bmatrix}^T$. Then

$$Av = \begin{bmatrix} \omega + \omega^{-1} \\ 1 + \omega^2 \\ \omega + \omega^3 \\ \vdots \\ \omega^{n-3} + \omega^{n-1} \\ 1 + \omega^{n-2} \end{bmatrix}$$

then $\omega + \omega^{-1} = 2\mathbf{Re}\omega$ is an eigenvalue of A for any root of unity ω. So, the spectrum of C_n is

$$\sigma(C_n) = \{2\cos(\frac{2\pi j}{n}), j = 1, \dots, n\}.$$

The adjacency matrices are always nonnegative and so we can use the Perron Theorem for any nonnegative matrices.

Theorem 1 (Perron [25]). Suppose $A \in \mathbb{R}^{n \times n}$ and $A > 0$. Then A has an eigenvalue λ that satisfies the following properties.

1. $\lambda = \rho(A)$.

2. If $\mu \in \sigma(A)$ and $\mu \neq \lambda$ then $|\mu| < \lambda$.

3. λ has algebraic multiplicity 1.

4. If $A\mathbf{y} = \lambda\mathbf{y}$ then $\mathbf{y} = \alpha\mathbf{x}$ where $\mathbf{x} > 0$ and $\alpha \in \mathbb{C}$.

Definition 1.2.12. If $A \in \mathbb{R}^{n\times n}$ we say A is reducible if there exists P (a permutation) such that

$$A = P \begin{bmatrix} X & Y \\ O & Z \end{bmatrix} P^T$$

where X, Y and Z are square matrices. We say $A \in \mathbb{R}^{n\times n}$ is fully decomposable if there exists P and Q (permutations) matrices such that

$$A = P \begin{bmatrix} X & Y \\ O & Z \end{bmatrix} Q.$$

The square matrix is irreducible if is not reducible and fully indecomposable if it is not decomposable.

Theorem 2 (Perron-Frobenius [25])**.** If the matrix A is fully indecomposable and non-negative then the properties the Theorem 1 still hold and if the matrix is irreducible then the properties 1,3 and 4 in Theorem 1 are guaranteed to hold.

The Perron-Frobenius theorem is relevant to networks, in particular we can refer directly to connectively

- If A is symmetric and reducible then it can be permuted into block $\begin{bmatrix} X & O \\ O & Z \end{bmatrix}$, where X and Z are completely independent of each other.

- If either X or Z is reducible it can also be permuted into diagonal form, and we know that any any symmetric matrix can be written in the form $P\text{diag}(A_1, A_2, A_3, \ldots, A_k)P^T$, where each of $A_1, A_2, \ldots, A_k$ is irreducible. So, the Perron-Frobenius theorem can then applied to each block.

- A network G is connected if and only if its adjacency matrix A is irreducible.

1.2.1 Clustering Coefficients of Networks

Many networks contain a large number of triangles. This feature of the network is a general consequence of high transitivity which means that the network has communities of nodes that are densely connected internally. Clustering coefficients measure this and are a widely-used statistic in network theory. They measure how many 2-paths (P_2) in a network are closed to form triangles (C_3).

We will define two clustering coefficients. The first is the Watts-Strogatz clustering coefficient. We define the clustering of a node by

$$C_i = \frac{t_i}{k_i(k_i-1)/2} = \frac{2t_i}{k_i(k_i-1)}$$

where the t_i is the number of triangles attached to node i of degree k_i. The Watts-Strogatz clustering coefficient is the average of the clustering coefficient of each node in a network:

$$\bar{C} = \frac{1}{n}\sum_i C_i.$$

The Newman clustering coefficient, also known as the transitivity index of the network, is defined as

$$C = \frac{3 \mid C_3 \mid}{\mid P_2 \mid} = \frac{3t}{\mid P_2 \mid}$$

where $\mid C_3 \mid$ is the total number of triangles and $\mid P_2 \mid$ is the number of paths of length 2 [25].

Example 3.

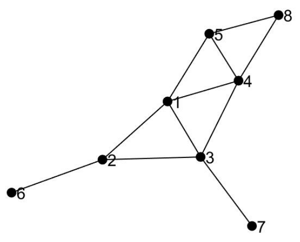


Figure 1.5: Example network.

First, we will compute the Watts-Strogatz clustering coefficient for the network in Figure 1.5.

$$C_1 = \frac{2.(3)}{4(3)} = \frac{1}{2}, C_2 = \frac{2.(1)}{4(3)} = \frac{1}{3}, C_3 = \frac{2.(2)}{4(3)} = \frac{1}{3}, C_4 = \frac{2.(3)}{4(3)} = \frac{1}{2}, C_5 = \frac{2.(2)}{3(2)} = \frac{2}{3}, C_6 = C_7 = 0, C_8 = 1$$

$$\Rightarrow \bar{C} = \frac{1}{8}(\frac{10}{3}) = \frac{5}{12}.$$

Second, we will use the Newman clustering coefficient to compute the clustering coefficient for the graph in Figure 1.5:

$$t = \mid C_3 \mid = \frac{1}{6}\text{tr}(A^3) = \frac{24}{6} = 4$$

and

$$P_2 = \sum_{i=1}^{n} \frac{k_i(k_i-1)}{2} = \frac{4(3)+3(2)+4(3)+4(3)+3(2)+1(0)+1(0)+2(1)}{2} = 25$$

$$\Rightarrow C = \frac{3(4)}{25} = \frac{12}{25} = 0.48.$$

We see that $0 \leq C, \bar{C} \leq 1$. For all networks $C, \bar{C} \in [0, 1]$.

1.2.2 Graph Laplacian

In graph theory, we often use the graph Laplacian which is a matrix with the same information as the adjacency matrix but different useful and important properties.

1.2.3 The Laplacian

Definition 1.2.13. [14] If G is a simple graph and k_v denotes the degree of vertex v, the Laplacian graph L is defined as

$$L(u,v) = \begin{cases} k_v, & \text{if } \ u = v, \\ -1, & \text{if } \ u, v \ \text{ are adjacent}, \\ 0, & \text{otherwise}. \end{cases}$$

This can also be written as $L = D - A$ where D is the diagonal matrix of degree or $L = BB^T$, where B is the incidence matrix defined in definition 1.2.4.

1.2.4 The eigenvalues and eigenvectors of graph Laplacian

The set of all the Laplacian eigenvalues is called the Laplacian spectrum of the graph G.

Definition 1.2.14. The vector $e = (1, 1, \ldots, 1)^T$ is always an eigenvector of Laplacian for eigenvalue 0 because $A\mathbf{e} = D\mathbf{e}$ and we know that all the Laplacian eigenvalues are nonnegative because the Laplacian is positive semidefinite (because $x^T L(G) x = x^T(\sum_{e \in E} L_e)x = \sum_{e \in E} x^T L_e x = \sum_{(i,j) \in E}(x_i - x_j)^2 \Rightarrow x^T L x \geq 0$).

Theorem 3. Suppose the eigenvalues of the Laplacian of a graph are ordered such that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{n-1} \geq \lambda_n = 0$$

then the following results holds [25]:

- $\lambda_{n-1} > 0 \Leftrightarrow G$ is connected.
- $k_{\max} \leq \lambda_1 \leq 2k_{\max}$, where $k_{\max}$ is the maximal degree of a node in the associated network.
- λ_1 is bounded above by the maximum of the sum of degree of adjacent nodes. That is,

$$\lambda_1 \leq \max\nolimits_{(i,j)|a_{ij}=1}(k_i + k_j)$$

1.2.5 The normalized Laplacian

Definition 1.2.15. Let D denote the diagonal with the v-entry having value k_v.
The Normalized Laplacian of G is defined by

$$\mathcal{L}(u,v) = \begin{cases} 1 & \text{if } u = v \text{ and } k_u \neq 0 \\ -\frac{1}{\sqrt{k_u k_v}} & \text{if } u, v \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

so $\mathcal{L} = D^{\frac{-1}{2}} L D^{\frac{-1}{2}}$, with $D^{-1}(u,v) = 0$ for $k_v = 0$.

We say v is an isolated node if $k_v = 0$.
If G is k-regular, then $\mathcal{L} = I - \frac{1}{k}A$ where A is the adjacency matrix and I is the identity matrix.
For a graph G without isolates nodes.

$$\mathcal{L} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}.$$

$\mathcal{L}$ can also be written as $\mathcal{L} = SS^T$ where S is the matrix whose rows are indexed by the vertices and whose columns are indexed by the edges of G such that each column corresponding to an edge $e = (u,v)$ has an entry $\frac{1}{\sqrt{k_u}}$ in the row corresponding to u, an entry $\frac{1}{\sqrt{k_v}}$ in the row corresponding to v, and has zero entries elsewhere. (As it turns out, the choice of signs can be arbitrary as long as one is positive and the other is negative) [14].

Since $\mathcal{L}$ is symmetric, its eigenvalues are real and we can assume that they are ordered, i.e. $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

The vector $D^{\frac{1}{2}}\mathbf{e} = (\sqrt{d_1}, \sqrt{d_2}, \ldots, \sqrt{d_n})^T$ is an eigenvector of the eigenvalue 0 because:

$$\mathcal{L} D^{\frac{1}{2}} e = D^{-1/2} L D^{-1/2} D^{\frac{1}{2}} e = D^{-1/2} L e = 0.$$

Since $\mathcal{L} = SS^T$ the normalized Laplacian $\mathcal{L}$ is positive semidefinite.

Lemma 1. Let G be a graph on n nodes with normalized Laplacian eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. Then $0 \leq \lambda_i \leq 2$ and $\lambda_n \geq \frac{n}{n-1}$ [89].

Example 4.

- For the complete graph K_n on n vertices, the eigenvalues of $\mathcal{L}$ are 0 and $\frac{n}{n-1}$ (with multiplicity $(n-1)$.
- For the complete bipartite graph $K_{m,n}$ on $m+n$ vertices, the eigenvalues of $\mathcal{L}$ are $0, 1$ (with multiplicity $(m+n-2)$), and 2.
- For the star $S_{1,n-1}$ on n vertices, the eigenvalues of $\mathcal{L}$ are $0, 1$ (with multiplicity $(n-2)$, and 2.
- For the path P_n on n vertices, the eigenvalues of $\mathcal{L}$ are $1-\cos\frac{2\pi k}{n-1}$ for $k = 0, \ldots, n-1$.
- For the cycle C_n on n vertices, the eigenvalues of $\mathcal{L}$ are $1-\cos\frac{2\pi k}{n}$ for $k = 0, \ldots, n-1$[14].

1.2.6 The Signless Laplacian

The matrix $L_s = D + A$ is called the signless Laplacian. We can also consider:

Definition 1.2.16. Let G be a graph with n vertices and m edges. Suppose each edge of G is assigned an orientation, which is arbitrary but fixed. The vertex-edge incidence matrix of G, denoted by $R(G)$, is the $n \times m$ matrix defined as follows. The rows and the columns of $R(G)$ are indexed by n vertices and m edges, respectively. The (i, j)-entry of $R(G)$ is 0 if vertex i and edge e_j are not incident, and otherwise it is 1 or -1 as e_j originates or terminates at i, respectively. We often denote $R(G)$ simply by R.

Let n, m, R be the number of vertices, the number of edges and the vertex-edge incidence matrix of a graph G. Then

$$RR^T = D + A, R^T R = A + 2I.$$

The eigenvalues of RR^T and R^TR are the same.

Example 5.

- Consider the graph shown in Figure 1.6. Its incidence matrix is given by

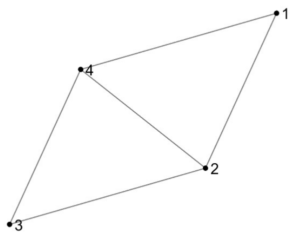


Figure 1.6: Example of normalized and signless laplacian.

$$R = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

where rows represent vertices and columns represent edges.

$$L_s = RR^T = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 3 & 1 & 1 \\ 0 & 1 & 2 & 1 \\ 1 & 1 & 1 & 3 \end{bmatrix}$$

$$S = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 & 0 & \frac{1}{\sqrt{3}} \\ 0 & \frac{-1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{3}} \end{bmatrix}$$

$$\mathcal{L} = SS^T = \begin{bmatrix} 1 & -0.4082 & 0 & -0.4082 \\ -0.4082 & 1 & -0.4082 & -0.3333 \\ 0 - 0.4082 & 1 & -0.4082 & \\ -0.4082 & -0.3333 & -0.4082 & 1 \end{bmatrix}$$

- Consider T_{31} shown in Figure 1.7.

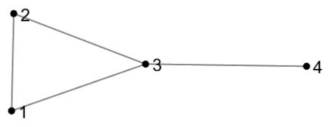


Figure 1.7: Example of normalized and signless Laplacian for T_{31}.

$$R = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$L_s = RR^T = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 1 & 3 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$S = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & \frac{-1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{3}} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

$$\mathcal{L} = SS^T = \begin{bmatrix} 1 & -0.5 & -0.4082 & 0 \\ -0.5 & 1 & -0.4082 & 0 \\ -0.4082 & -0.4082 & 1 & -0.5774 \\ 0 & 0 & -0.5774 & 1 \end{bmatrix}$$

1.3 Test Networks

Throughout this thesis, we will work with a selection of real-world networks taken from the literature. In many cases, some of the properties of these networks (for example their community structure) are well-known and so we can compare algorithmic results to the so-called "ground truth". We now describe the source of the data behind these networks.

1.3.1 Network Descriptions

Brain networks

Neurons C elegans

The network of neurons of the Caenorhabditis elegans which is a nematode that was determined to be small enough to be reconstructed and analyzed because it has 302 neurons that are identifiable and consistent across individuals within the species. Two networks have been used to analyse this species: a non-directional gap junction network and a directional chemical synapse network. We use just the second [83].

Ecological networks

Ecological networks are complex networks and we can divide this network in to three types: food webs, mutualistic networks and host–parasitoid networks. In community studies, the nodes are comprised of individuals that make up species populations, and the links connecting the nodes indicate population effects [43] .

Stony

Food network in a stream in New Zealand which studied food webs comprising fish, macroinvertebrates, and algae, called Stony stream, at an altitude of 800 metres in an area approximate 16 km^2. [79].

Canton

Another stream site in New Zealand. This network was one of ten study sites chosen for a study of food web architecture. It is 30 metres long and is a tributary of the Taieri River in the south of New Zealand. It occurred in grassland catchments but was developed for pasture [79].

El Verde

Species interactions in a rainforest in Puerto Rico. This network is an simplified yet aggregated food web of the El Verde community in a rainforest in Puerto Rico. The dominant predators are frogs and lizards with population densities that some of the highest ever recorded. First level consumers are species such as bats, snails and birds, second level consumers consist of some other species of birds, arboreal invertebrates, rats and other lizards. Third level consumers consist of frogs, bats, some species of birds and arboreal arachnids and anoline lizards and fourth level consumers consist of snakes, some arboreal arachnids and mongoose. Finally, fifth level consumers are certain species of birds. Species such as snakes, birds, rats and arboreal arachnids are important at different places in the web and so show up at different consumer levels or overlap more than one level because they are omnivorous [76].

Ythan

The Ythan network is a food web from the Ythan estuary in North East Scotland. The total number of species in the web is 134 and four versions were made including one with more than two basal elements, one which included parasite species, one with hypothesized links between hosts and parasite species, and one where parasites had two or more life stages. We just used the version 3 [42].

Scotch

The network comprises a single field site of a community of the Scotch broom species, Cystisus scoparius, from a field site in England. Its data consists of a food web that describes the trophic relationships of 154 species, one plant, 19 herbivores, five omnivores, 66 parisitoids, 60 predators, and three pathogens [59].

Little Rock

Species interactions in a lake in Wisconsin. Various taxonomies at all the tropic levels such as fish, invertebrate predators, predators, zooplankton and the like, have been sorted into life stages characterized by changes in diet. It contains a total of 220 taxa and the webs vary in size from 10 to 74 species [37].

Wildbird

The social ecology of wild birds in Wytham Woods in Oxford in the UK was examined by studying great tits, blue tits, Eurasian nuthatches that were caught in nestboxes during the breeding season or in the winter using mist-netting. The data was collected between November 2013 and April 2014 using selective feeding sites to study association patterns between species [29].

Dolphin

The Dolphin network is a group of bottlenose dolphins in New Zealand. Between 1994 and 2001, Lusseau et al. [58] conducted surveys of bottlenose dolphins in Doubtful Bay, New Zealand. During the study period, 40 dolphins were analysed for association out of 83 that were identified as they were observed for frequency of occurrence in the schools the researchers intended to observe. The data collected was analysed which identified communities or clusters of individuals by preferred partnerships and least preferred partners [58].

TM1,2 and 3

A network of three-dimensional galleries in termite nests which are not the result of repetition of patterns but a system of galleries and chambers inside the nest. They are, in effect, three-dimensional transportation networks with topological properties relevant to ecology and survival. There are few 3-D networks in the literature because of their complexity and the difficulty in capturing their spatial patterns. These nests are from different locations in Central African Republic and Cameroon [71].

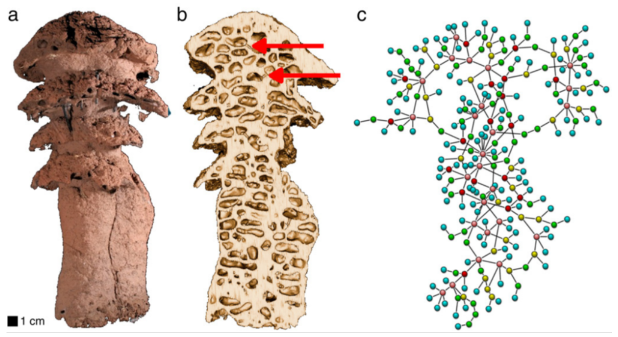


Figure 1.8: Left: Picture of one nest; centre: a tomographical cut of the same nest (the arrows indicate two of the corridors between chambers); right: a representation of chambers and galleries as a network, where each vertex corresponds to a chamber in the original nest and each edge to a corridor. The colour of the vertices reflects their degree; vertex positions have been adjusted to improve visibility. [71].

Informational networks

Centrality

This network is a collection of citations published about network centrality from 1948 to 1979 in order to analyze the structure of centrality productivity for this period [41].

GD

Citation network of papers published in ”Proceedings of Graph Drawing”. These graphs were obtained using a program called Pajek which is available for free online. Graph A is a citations networks analysis where each unit's importance or weight is determined using three proposed methods: NPCC method, paths count method and SPLC method. The original graph has 311 vertices and 6 weak components. Graph B was done using an energy drawing and then the position of the vertices was done manually using a grid. Graph C is an acyclic directed graph that can be sorted by topology. It is a dense graph so a matrix representation was used to visualize it [6].

Roget's Thesaurus

Network of words related by their definitions in Roget's Thesaurus. Roget's Thesaurus was created by Dr Peter Mark Roget who first listed over 15,000 synonyms for words in 1805 and has increased to over a 250,000 synonyms as of 1992. The vertices relate to the 1879 edition and each one represents one of 1022 categories listed there. If there was a reference between two categories this was represented by an edge `http://vlado.fmf.uni-lj.si/pub/networks/data/dic/roget/roget.htm`.

Small world

Network of papers citing S. Milgram's 1967 Psychology Today paper (An experimental study of the small world problem) [81]. In Stanley Milgram's small world study, people volunteered to be part of a study on social contact in America in response to an ad. Each participant was asked to send a packet of information to an individual by giving it to someone they knew on a first-name basis—someone who knew the individual or who would most likely know someone who knew them. The play Six Degrees of Separation referred to Milgram's study. The degrees are the number of people who were intermediaries in the chain. People with more social connections have a lower degree of separation then people with fewer or no social connections. Since his original study done in 1967, his experiment has been cited in many papers which therefore forms a citation network [45].

Biological networks

PIN Malaria

A PIN is a protein interaction network. PIN Malaria is a biological network that contains data on the position of proteins in the human malaria parasite. The parasite which causes the most dangerous form of malaria, *Plasmodium falciparum*, has not had most of its proteins characterized. 2,846 protein-protein interactions were characterized in this network using information analyses and gene co-expression [54].

PIN H Pyl

The *H.Pylori* Database of Protein Interactomes (hp-DPI) contains *H.Pylori* protein interactions that are both experimental and predicted with added annotations. The database was created with protein domain scanning tools to define particular protein-domain relationships [56].

PIN E Coli

The DNA cassettes in the *E. Coli* chromosome have been targeted to create certain allelles. 857 proteins were tagged and their interacting protein partners were identified revealing a protein-protein interaction network of proteins that are involved in varied biological processes [12].

Trans E Coli

One of the best-characterized networks are the direct transcriptional interactions in Escherichia coli. The network is made up of three distinctive motifs each having a specific function in determining gene expression. The structure of the motifs allows the entire network to be interpreted [78].

Trans-Yeast

Direct transcriptional regulation between genes in yeast. This network is part of a superfamily that contains sensory transcription networks that control gene expression in bacteria and yeast in response to external stimuli. In a network such as this, nodes represent genes and edges are direct transcriptional regulation. The yeast strain is *Saccharomyces cerevisiae* [60].

Social and economic networks

Corporate people

The network properties of directors and companies from three years: 1982, 1990 and 1999, are used to examine the corporate elite network structure in the US in order to help analyze corporate change on a macro level. Corporate people is an economic network that includes board members from American Fortune 1000 companies in 1999 [20].

Drugs

Social network of injecting drug users. The data in this network comes from the Colorado Springs Project 90 study which was a project funded by the Centre for Disease Control and Prevention which focused on HIV transmission in heterosexual injecting drug user populations. 595 respondents from face-to-face interviews within a four-year period from 1988 to 1992 using an open cohort design. The type of data collected included focus on bringing forth characteristics of risk partnership networks to allow researchers to identify and collect data from as many people in the target population as possible including injecting drug users, prostitutes, and their sex and needle-sharing partners. Researchers also wanted to assess the size, structure and potential for epidemic of the entire high-risk partnership network. Two networks were constructed which represented sexual contact ties, ties involving shared drugs and ties involving both sexual and drug-contact sharing. There were 1296 reported connections and data from the respondent-only network was extracted from the larger network which included all ties including those who were not participants in the study [1].

ColoSpg

The risk network of persons with HIV infection in Colorado Springs. AIDS had been a condition that was required to be reported since 1983 and HIV infection since November 1985. The data used in this risk network therefore started from late 1985 through to 1999. From the spring of 1987, the military tracked its own HIV data and released information only about non-military or civilian partners making their data unavailable for analysis. Contact tracing was available from 1985 onwards and information from other routine STD programmes such as syphilis, gonorrhoea, and chlamydia were used to add sex-linked pairs of men to the HIV contact tracing database. The data was divided into three periods: (1) early: from 1982 to 1989, (2) middle from 1990 to 1994, and (3) late from 1995 to 1999. A total of 1,323 adult HIV/AIDS cases were reported between 1982 and 1999 [73].

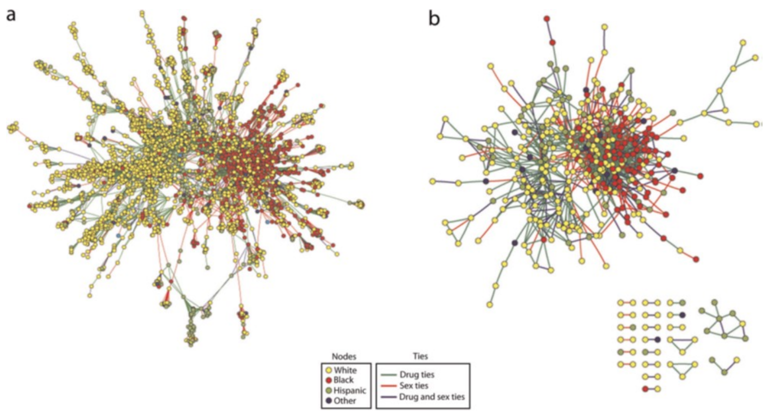


Figure 1.9: Ties among (a) the giant connected component for the full network, and (b) the observed respondent-only network: Colorado Springs Project 90, Colorado Springs, CO, 1988–1992.[1].

Revere

A network of people connected to American revolution. The network includes membership lists of key political groups during the period of the American Revolution compiled in 1994 from of a group of seven lists. These lists are select examples from organizations in Boston that played significant roles in the movement. The network itself consists of five groups and 137 people [36].

FB Reed

The Facebook network at Reed College is from a single day snapshot in September 2005 [80].

Karate

This network contains a university-based karate club, which was divided into two organizations after having broken into two factions based on group members' beliefs about the leadership. Observations of the club were made over a three-year period from 1970 to 1972. The history of the club prior to these dates was also reconstructed using archives and records from that time. During the three-year period, the club's membership ranged from 50 to 100 members and individuals met together for social events such as parties, dances, and banquets in addition to the scheduled karate lessons. It had an informal political organization although it did have a set of guidelines laid out in 'constitution' and there were four officers that loosely ran some aspects of the organization. Despite these officers, the majority of decisions were made by consensus at club meetings. The club employed a part-time karate instructor referred to as Mr. Hi. Initially, there was a conflict between the club president, John A., and Mr. Hi over the price of karate lessons. Mr. Hi wanted to raise prices and have the authority to do so with this own lesson fees whereas John A. wanted to have one stable price that he set in his role as the club's chief administrator. Over time, club members became divided over this issue and then the divide grew beyond the issue of prices to one of ideology where supporters of the club president viewed him as a father figure and spiritual and physical mentor. They viewed

Mr. Hi as a paid employee who simply wanted to earn more money. After some marked disagreements, John A. fired Mr. Hi for trying to raise prices which resulted in Mr. Hi's supporters retaliating by resigning from the club and forming a new organization. During this time, the confrontations between the factions took place at the meetings where if one faction held the majority, they would attempt to pass decisions that were in their favour and vice versa. Neither of the factions had a name and in situations where they were not in direct conflict, members from respective factions spoke and interacted with one another. Over time, bonds within the two groups were strengthened while bonds between them weakened thus eventually pulling the club apart [86].

Mafia

Data is on attendance of suspected members of the Ndrangheta criminal organization at summits (meetings whose purpose is to make important decisions and/or affiliations, but also to solve internal problems and to establish roles and powers) taking place between 2007 and 2009 `https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/ndrangheta-mafia-2`.

Caviar

A network of Canadian drug importers. Project Caviar was an investigation targeting a network of hashish and cocaine importers operating out of Montreal. It was conducted by the RCMP and the Montreal Police between 1994 and 1996 in conjunction with other regional law enforcement agencies from countries in England and various parts of Europe and South America. It is considered a unique case because of its investigative approach which was termed "seize and wait". The goal was to seize drug consignments but not arrest any of the identified associated individuals, thus, 11 consignments were seized while the arrests only took place at the end of the investigation. Because of the change in strategy, the change in network structure and how network participants react and adapt could be assessed. The data was from evidence submitted in the trial with 22 participants in the

Caviar network and includes thousands of paragraphs of electronically recorded telephone conversations. Of over three hundred individuals, 208 were not implicated in the drug trafficking ring so the final network was composed of 110 participants [61].

Siren

A network of car thieves. A stolen vehicle exportation ring were obtained from a task force project called Project CERVO between 1993 and 2005. The exportation of stolen luxury vehicles from the Port of Montreal was monitored with information supplied from maritime shipping companies. Project Siren started in February 1998 with an informant providing information on stolen vehicles shipped to Ghana which was seized in Belgium. It was a four-month investigation (to June 1998) wherein 35 cars were retrieved [61].

Mali

A terrorist network in Mali is analyzed for relationships between individuals belong to Islamist and rebel groups. The ties between agents and the ways they interact produce a network structure that can be analyzed [84].

Fifa

Two Networks of Standing Committee membership. These are overt networks with covert elements `https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/fifa`.

Dublin

This dataset was taken from an exhibition entitled Infectious that was held in Dublin, Ireland at the Science Gallery from April to June 2009. The interactions between conference participants with respect to their proximity and static and dynamic interactions [44].

Technological networks

Electronic1,2 and 3

Electronic1, Electronic2 and Electronic3 are electronic logic circuits with 31 digital sequential circuits that are described at the gate level. By analysing this circuit, we can calculate various parameters that are connected in an electrical network [10].

US-Air

This network considers the topology or structure of the US air transport network which considers 500 of the highest traffic airports according to http://www.iata.org. [16].

Software networks

Software Digital, Software-Abi, Software VTK, Software My SQL

A class collaboration is the process by which more complex, multifunctional classes are built from simpler ones. Classes interact through a process called inheritance where one class is defined as a subclass of another and through aggregation where one class is defined to hold an instance of another class. Software Digital is a collaboration network associated with six different open-source software systems. It is a class collaboration graph from version 4.0 of the VTK visualization library, the CVS snapshot dated 4/3/2002 of Digital Material (DM), a library for atomistic simulation of materials, and version 1.0.2 of the AbiWord word processing program [62].

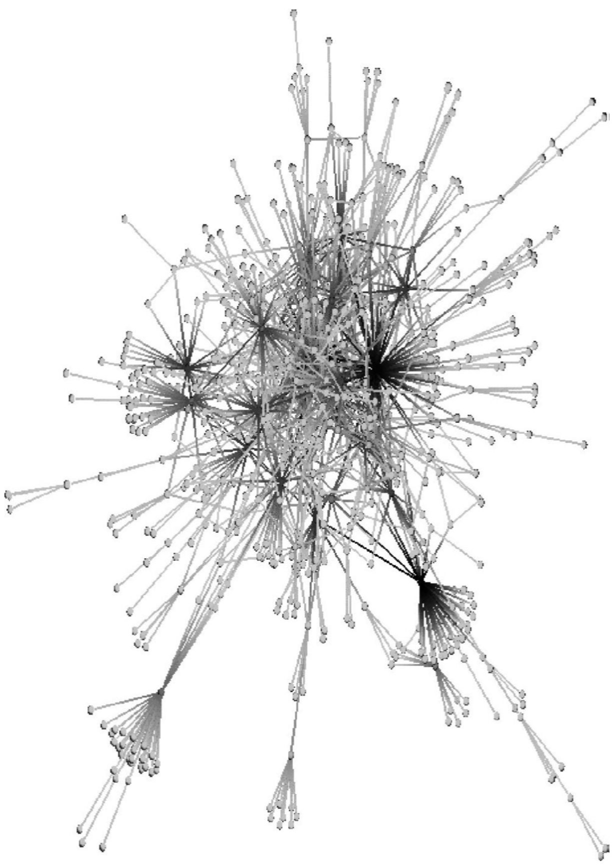


Figure 1.10: VTK network [62].

In the following Tables the number of nodes is n, the number of edges is m and the $k_{\max}$ is the maximum number of the node degree in the network, C_3 is the total number of triangles and C is the Newman clustering coefficient.

Table 1.1: Test Networks 1.

	n	m	$k_{\max}$	C	C_3
Brain networks					
Neurons C. elegans	280	1973	77	0.1987	2808
Ecological networks					
Canton Creek	108	707	47	0.0256	116
El Verde	156	1439	83	0.2322	3511
Little Rock	181	2318	105	0.3368	10812
Scotch Broom	154	366	36	0.2078	358
Stony Stream	112	830	45	0.0201	124
Wildbird	169	2758	81	0.6645	27886
Dolphins	62	159	12	0.3088	95
Ythan	134	593	65	0.1443	507
Informational networks					
Centrality	118	613	66	0.274	1107
GD	249	635	20	0.2202	327
Roget Thesaurus	994	3640	28	0.1338	1550
Small World	233	994	147	0.16	1637

Table 1.2: Test Networks 2.

	n	m	$k_{\max}$	C	C_3
Social and economic networks					
Corporate	1586	11540	65	0.3881	28002
Drugs	616	2012	58	0.3681	3598
ColoSpg	324	347	20	0.0385	17
Revere	254	9807	217	0.7354	208074
FB_Reed	962	18812	313	0.2207	97137
Karate	34	78	17	0.2557	45
Mafia	151	1619	75	0.5825	9575
Caviar	110	205	60	0.1229	130
Siren	44	103	33	0.4061	142
Mali	36	67	11	0.3945	43
FIFA	307	3070	87	0.6528	18817
Dublin	410	2765	50	0.4357	7114
Software networks					
Software Digital	150	198	25	0.0668	24
Software MySQL	1480	4190	220	0.058	1796
Software VTK	771	1357	83	0.0438	236
Software_Abi	1035	1719	89	0.0387	219

Table 1.3: Test Networks 3.

	n	m	$k_{\max}$	C	C_3
Biological networks					
PPI Malaria	229	604	35	0.1159	201
PPI H. pylori	710	1396	55	0.0152	76
PPI E. coli	272	3430	104	0.6306	42549
Trans_Ecoli	328	456	72	0.0243	42
Trans_Yeast	662	1062	71	0.0163	72
Technological and infrastructural networks					
Electronic 1	122	189	10	0.0574	10
Electronic 2	252	399	14	0.0517	20
Electronic 3	512	819	22	0.0484	40
USAir97	332	2126	139	0.3964	12181
Other Networks					
TM3	268	437	12	0.1381	78
TM2	260	280	12	0.0087	2
TM1	507	676	10	0.0478	33

Chapter 2

Community Detection

2.1 Introduction

This chapter is a brief survey of existing results in community detection. We develop some ideas on community structure in Chapter 3 and we look at anti-communities in Chapter 5. One of the most important ways complex networks can be structured is into communities: for undirected networks, these are typically groups of nodes that are more densely connected than in other parts of the network. These clusters of nodes can form for many reasons, such as in a social network where nodes represent individuals that may have ties with certain individuals and yet not others. In a biological network where nodes represent proteins, communities could represent proteins that have a similar function.
Finding these communities based on the structure of the network alone is useful to scientists in their respective fields since these clusters or communities represent shared properties between these nodes. The properties within these communities will frequently share characteristics that are not shared by nodes that are outside of this dense cluster making them of special interest to researchers. Additionally, community detection may aid researchers in determining the organisation of a network. Within the detected communities, nodes that lie in the middle of their respective community versus at the edge may represent some difference in their roles within that community.
Traditionally, communities were seen as dense subgraphs that are clearly separated from each other as shown in Figure 2.1.

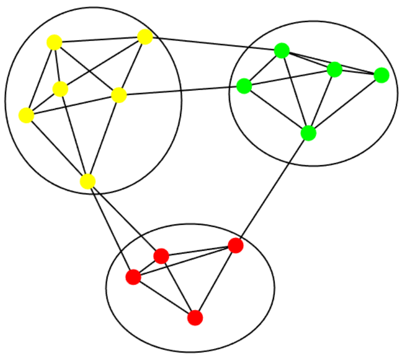


Figure 2.1: A graph with three Communities [32].

Communities can also be overlapping at their boundaries as shown in Figure 2.2.

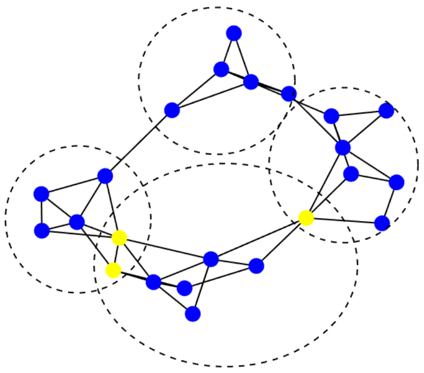


Figure 2.2: Overlapping Communities [32].

Some researchers have coined the terms soft clustering to define communities that overlap each other and hard clustering where there is no overlap where the boundaries of the community are clearly defined. In hard clustering (e.g.,in Figure 2.1), each data point belongs to a cluster completely or not at all, whereas Figure 2.2 is an example of what would be termed soft clustering where there is overlap at its boundaries with each data point assigned a probability or likelihood of being in a certain cluster. A subgraph with vertices or nodes that connect to all the others is called a clique. Cliques are considered complete graphs and the simplest example is a triangle. Communities, however, are not usually complete graphs. All vertices in a clique have identical properties which is rarely if ever the case in a real-world network where some vertices are more important than

others which are reflected in heterogeneous links throughout [32].
A better community definition should consider the level or degree of internal cohesiveness in the network and how separated it is from the rest of the network. A simple way to enforce this is to insist that in a community the number of internal edges is larger than the number of external edges. This leads to the definition of a strong community which is one where the internal degree of each vertex is greater than its external degree [75]. This in turn leads to the idea that in a network with more than one community the definition of a strong community is one where the internal degree of any vertex within that community is greater than the internal degree of the vertex in any other community [40]. Conversely, the community is weak if its internal degree is not greater than the total internal degree of its vertices in every other community. According to these definitions, the network in Figure 2.3 shows both strong and weak communities. The four subgraphs which are encircled in Figure 2.3 would all be considered weak communities according to the definitions by both Hu et al. and Radicchi et al., but according to Hu et al., they are also strong because the number of edges joining the vertex with the other vertices of all the other subgraphs is less than the internal degree of each vertex. According to Radicchi et al., three of these communities are not strong because some vertices have external degrees larger than their internal degrees as some vertices (indicated in blue)[40].

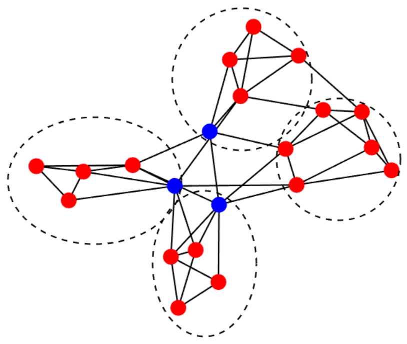


Figure 2.3: Strong and weak Communities [32].

Alternative definitions of community examine the probability that nodes in clusters share edges. Because the concept of communities implies a greater density of vertices or nodes, it makes sense that vertices within the same community have a higher probability of forming edges with those vertices more densely clustered around them than other vertices. Therefore the definition mentioned above can be modified as strong communities' vertices having higher probability to be linked to every vertex of the subgraph than to any other vertex in the graph. The edge probability is P_{in} between vertices in the same group and $P_{\text{out}} < P_{\text{in}}$ for vertices in different groups (A/B) as shown below

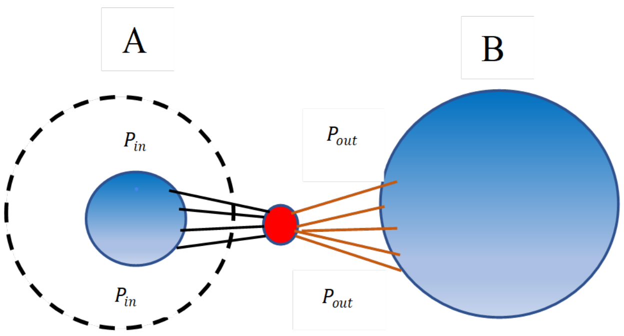


Figure 2.4: Problems of the classic notions of strong and weak communities [32].

Depending on the type of network, edges are formed differently which then presents the problem of how to calculate edge probabilities.

2.1.1 Definitions

If all probabilities are equal, then we derive the classic random graph model Erdős–Rényi (see Chapter 3) where there is no community structure.

We now give some formal definition of quantities that are useful when discussing communities. Given a subgraph $G_1(C, E_1) \subseteq G(V, E)$ and $n_C =\mid C \mid$, the number of edges the node $k_i \in C$ has to other nodes internally and externally are defined as

$$k_i^{\text{int}} = \sum_{j \in C} a_{ij}, \quad k_i^{\text{ext}} = \sum_{j \in \bar{C}} a_{ij} \tag{2.1}$$

where $\bar{C}$ is the complement of C and A is the adjacency matrix of G. We determine the number of links that connect nodes internally using $m_C = \frac{1}{2}\sum_{i \in C} k_i^{\text{int}}$.

Intracluster density, that is the density of nodes within a cluster, is

$$\delta_{\text{int}}(C) = \frac{m_C}{n_C(n_C-1)/2} = \frac{\sum_{i \in C} k_i^{\text{int}}}{n_C(n_C-1)}, \tag{2.2}$$

while inter-cluster density, that is the density of nodes between or outside of a cluster or community, is defined as

$$\delta_{\text{ext}}(C) = \frac{m_{C-\bar{C}}}{n_C(n-n_C)} = \frac{\sum_{i \in C} k_i^{\text{ext}}}{n_C(n-n_C)}. \tag{2.3}$$

Using networks which have communities, when we calculate the quantities we expect the value for the internal density of the cluster to be markedly larger than its external density and the internal density of a cluster to be larger than the total density of the network. Nodes within the same community are all connected by at least one edge, thus a community of a network can be defined as a set of nodes which are internally connected and has an internal density that is significantly larger than the external density of the whole network. Networks constructed with explicit clusters are referred to as benchmark graphs.

Defining community based on the topology and dynamics of a network can be addressed with diffusion. Random walks are simple diffusion where the vertex reached at t is a random neighbour of the vertex reached at $t-1$. Because there are few routes to the outside, the random walker could be spending a long time within the community [32]. The result of using random walks this way also depends on the routes the walkers can follow, and their distribution, which means these dynamics necessitate factor in many variables into the calculation. The advantage of this complexity is it will reflect more features of a networks which are usually quite complex with the drawback being the additional time it takes to make complex calculations [32].

2.1.2 Stochastic Block Model

The stochastic block model (SBM) is a model for data from a network where the nodes are separated into groups called blocks where the links between them are dependent on which block the nodes belong [38].

In the $k \times k$ SBM we partition the adjacency matrix into the structure

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1k} \\ A_{21} & A_{22} & \dots & A_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ A_{k1} & A_{k2} & \dots & A_{kk} \end{bmatrix}.$$

where the probability of an entry equalling 1 being fixed in each block. A simple example is the 2×2 model

$$A = \begin{bmatrix} P_1 & Q \\ Q' & P_2 \end{bmatrix}$$

where the edges between nodes in the diagonal blocks P_1 and P_2 have a fixed probability $0 \leq p \leq 1$ of existing and the edges between blocks in Q exist with a fixed probability $0 \leq q \leq 1$. By varying the relative size of p and q we can emphasise different network structures. If $p >> q$ then A will represent a network with 2 strong communities. If $p << q$ we have an almost bipartite graph.

The SBM ability to produce communities and clusters makes it the standard for studying community detection. It is possible with this model to derive assortative structure as bonds between vertices in the same group are prioritized. Disassortative structure can also be derived because edges are more likely to occur between the blocks than inside them. Multipartite structure arise if there are edges only between blocks and core-periphery structure if the vertices are well connected in the core structure and to peripheral vertices that do not interact between themselves very much [32]. For example: by choosing $p = 0.75$ and $q = 0.15$ with all blocks being 5×5 we generated the following adjacency matrix. The community structure is evident in Figure 2.5

$$\text{SBM} = \begin{bmatrix} P_1 & Q \\ Q' & P_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

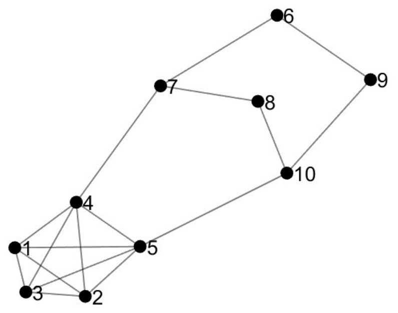


Figure 2.5: Example of Stochastic Block Model.

2.1.3 Examples

In this section, we briefly introduce some classic examples of communities emerging in real-world networks.

American politics

In 2003, Valdis Krebs studied a network of associated book purchases on amazon.com with the theory that the book network could be an accurate proxy network for the human network that reads those books [50]. In order to do this, he used Amazon's databases of customers' book purchases and linked two books if they were bought together and followed the connections between the books until two distinct clusters emerged removing any books that had two links or less. When he did this, a pattern emerged with the cluster on the right showing a densely connected set of a small number of books and the cluster on the left showing a broader array of books that was less dense. Further study of the proxy network revealed two divisions of political thought with dense intraconnectivity and sparse interconnectivity showing each group in an echo chamber of sorts reading similar ideas in their chosen books. The connectivity between the two clusters is conjectured to reflect academics or political commentators who needed to be well-read in both areas of political thought. The connection between the two books were several common purchasers who identified as conservative and liberal which are seen in Figure 2.6 in green and purple.

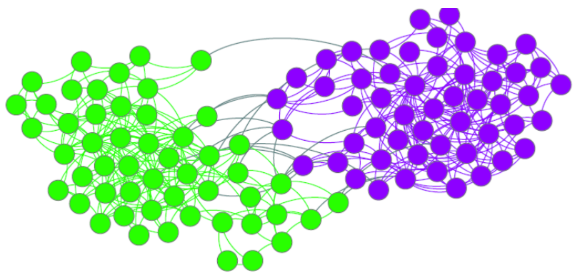


Figure 2.6: Conservative and liberal purchasers of books on amazon.com

A karate club

In 1977, Wayne W. Zachary conducted a social anthropological study on how fission takes place in bonded groups using a social network approach. While it was a karate club, it also held social activities such as parties and dances as well as regularly scheduled karate lessons. A conflict over pricing between the club president who wanted to stabilize prices and the karate instructor who wanted to increase prices resulted in over time the majority of the members of the club becoming ideologically divided but not organisationally until they formally separated into two organisations. Prior to the events that preceded the fission, decisions made about the club took place at meetings where if the group that supported the president held majority would vote in their favour and vice versa when the group supporting the instructor held majority. These groups were not recognised to exist by the club members, even unofficially. Rather, they were groups that formed during crisis moments, such as when voting on issues at meetings, from pre-existing friendships within the club and because of this key difference of opinion. These meetings where the members had to vote on club issues then served to strengthen the existing friendship bonds within these ideological groups while also weakening the bonds between them. Each subsequent political crises in the club reinforced and further strengthened the bonds within group and weakened the bonds between groups until the group completely and formally separated. The connections between club members prior to fission is shown below in Figure 2.7 where one group is represented in green and the other in blue[86].

Zachary then created a mathematical model based on precise descriptions and the relationships between categories clearly defined. The most significant component of this system was the network of friendship relationships amongst the members. The network was represented as a graph and then more formally as an adjacency matrix[86].

In his social network model, the social relationships in the karate club are represented on the graph with each member of the club represented by a point (or node) on the graph with a line (or edge) between any of the two points representing the friendship between the two members. A friendship would be defined as two members that consistently interacted outside of normal club activities. As a result, although the club membership over the three-year period was between 50 and 100, there are only 34 individuals in this graph and matrix because they would have been reflected in unconnected nodes and rows and columns with zeros which Zachary represented on the graph in Figure 2.7 below. The edges represent interactions in both directions so are termed nondirectional and thus, the graph is symmetrical[86].

The network and matrix only contain the members that interacted with other club members outside of meetings. The model allowed for the location of the fission within the group to be predicted to more than 97% accuracy[86].

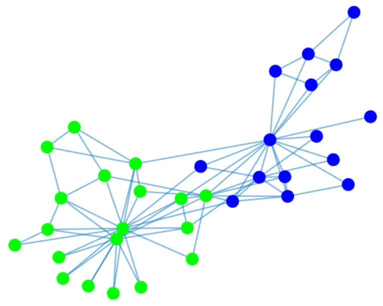


Figure 2.7: The karate club network.

A dolphin community

A third example that is frequently cited is a group of bottlenose dolphins in New Zealand [58]. Between 1994 and 2001, Lusseau et al. [58] conducted surveys of bottlenose dolphins in Doubtful Bay, New Zealand. During the study period, 40 dolphins were analysed for association out of 83 that were identified as they were observed for frequency of occurrence in the schools the researchers intended to observe. The data collected was analysed which identified communities or clusters of individuals by preferred partnerships and least preferred partners. Two individuals would be identified if they were observed associating more often or if they did not associate with individuals outside their cluster[58].
The cluster analysis did not show any clear divisions in the community but did identify three groups which associated more with each other more than all the individuals did on average. Group 1 was seen less and so observed less by the researchers. Groups 2 and 3 had one individual which had a central position in both groups. These two groups also each had a male network and a female network, and Group 2 was more complex with

some groups of two and three individuals. Lusseau depicted the three communities they observed in this social network of dolphins as shown below in the sociogram in Figure 2.8 [58].

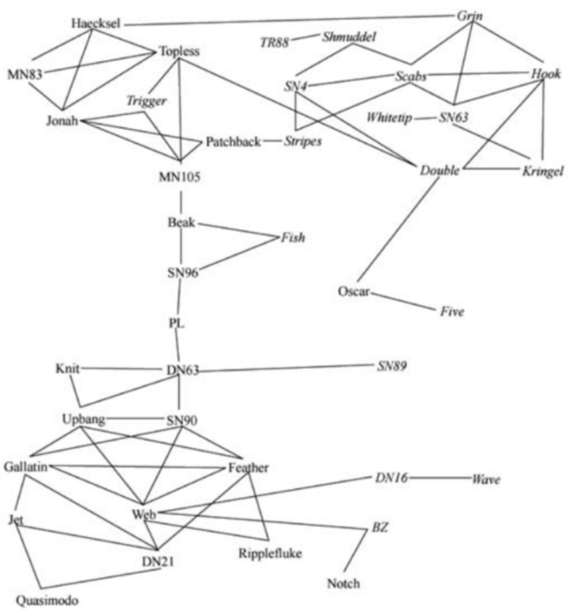


Figure 2.8: Sociogram of Dolphin.

They observed that the dolphin population with an unusually large number of long-lasting bonds within the Doubtful Sound population that could not be attributed to seasonal factors nor reproductive, foraging or defence advantages. They concluded that there was strong evidence that the uncharacteristically stable social organisation could be explained by the population being isolated in a fjord which was the source of ecological constraints that required more group stability for survival in this habitat [58].

2.2 Algorithms to Partition

Whereas community detection searches for community structure in order to look for particular special properties of a respective network such as social groupings or protein linkages, graph partitioning refers to separating a larger graph into roughly equal sized sections while keeping the number of edges to a minimum between the different partitions. In fields such as parallel computing, there is significant value in being able to partition graphs this way so that each can be assigned to a processor so as to minimize the time it takes the computer to analyse the entire graph. In short, graph partitioning allows for more efficient graph analysis and problem solving of larger graphs. Other areas of study where being able to section equal sized parts of a graph are VLSI layout, sparse linear system solving, and circuit testing and simulation [4].

The aim is to partition networks into p disjoint sets of nodes with the following properties:

- $\bigcup_{i=1} V_i = V$ and $V_i \cap V_j = \emptyset$.
- The number of links crossing between subsets (cut-set size or boundary) is minimized.
- $\mid V_i \mid \approx \frac{n}{p}$ for $1 \leq i \leq p$.

This process can be generalised to weighted networks by defining the cut-set as the sum of the weights of the links that cross the subsets and rewriting the final condition. We then refer to this partition as balanced. We will discuss some algorithms that can be used to partition a network.

2.2.1 Local Improvement Methods

In 1970, B.W. Kernighan and S. Lin [47] proposed taking a bisection of a network and decreasing the cut-set using a local search approach. The initial partition can be generated randomly or done by using any bisection method. If the random approach is used, then several should be generated and the one with the minimum cut size selected. Considering only unweighted networks to simplify the process, the internal and external weight for each node of a bisection of the graph is defined by splitting its degree with the cut size calculated using

$$C(V_1, V_2) = \frac{1}{2} \sum_{i \in V} W_i^{\text{ext}} \tag{2.4}$$

where V_1, V_2 a bisection of the network and $W_i^{\text{ext}} = \sum_{i,j \in E} w_{ij}$ and $w_{i,j}$ is the weight of the link between nodes i and j.

A reduction in the cut size can be gained by moving one node from one subset to the other and we can measure the gain in the cut size ($g_i = W_i^{\text{ext}} - W_i^{\text{int}}$). In order to prevent an imbalance in the number of nodes in the partition as seen in Figure 2.10, the gain produced should be quantified by transposing two nodes from different partitions. Figure 2.9 is an illustration of a random partition of a network.

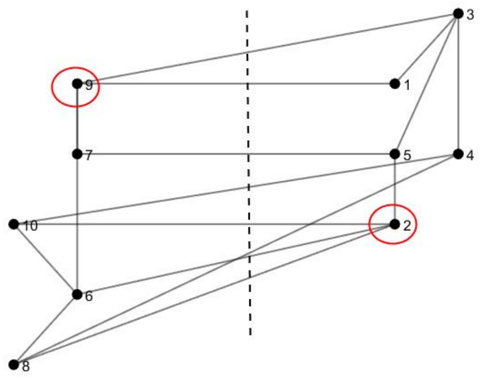


Figure 2.9: A random partition of a network

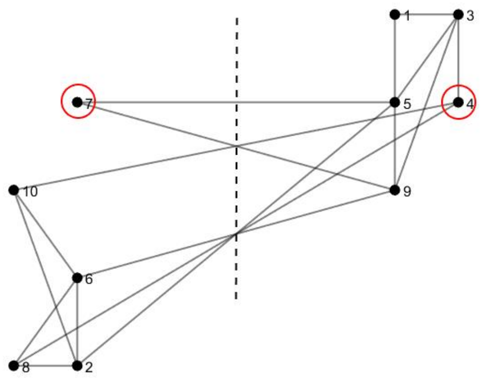


Figure 2.10: An improvement to the partition in Figure 2.9

Once the gain is quantified, the cut size in our example is reduced to 2 from 5 as shown in Figure 2.11.

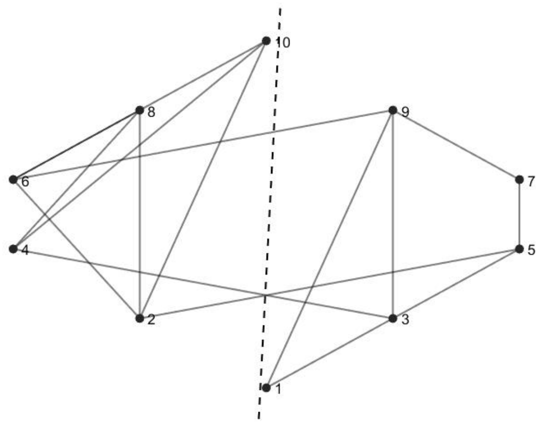


Figure 2.11: An even better partition of Figure 2.9

This process of swapping nodes is the basis of what is known as the Kernighan-Lin algorithm which starts with a balanced bisection, a cutsize is computed, a pair of nodes are found which give the biggest value, they are labelled, the value of $g(u)$ is updated,

$$C_k(V_1, V_2) = C_{k-1}(V_1, V_2) - g(v_1^k, v_2^k), \quad k = 1, ..., r, \tag{2.5}$$

the partitions are updated by moving the sets and the process is repeated until no further improvement can be gained. A drawback to this algorithm is the amount of time required, however improving the process for switching nodes, using a fixed number of iterations, and choosing nodes close to the partition boundary only to evaluate gain can reduce costs. While this algorithm is not used today to detect communities, exploring how it functions and its methodology demonstrate the rationale behind community detection methods [25] and it can be used to improve partition provided by other algorithms.

2.2.2 Spectral Partitioning

Spectral partitioning is basically a way of dividing a graph into two subgraphs so that each subgraph has nearly an equal number of vertices and edges between the two subgraphs are minimised. It uses the eigenvalues of the matrix associated with the graph to partition optimally.

Laplacian spectral partitioning

The Fiedler Vector

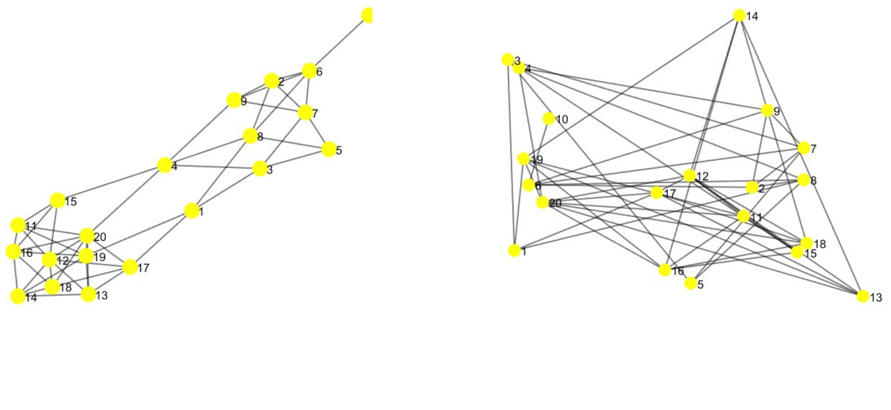
One example of spectral partitioning which was put forth in the early 1970s is the use of the eigenvector associated with the second smallest eigenvalue of a graph Laplacian matrix called the Fiedler vector to bipartition a network. Let ϕ_2 denote the Fiedler vector. In this method, two nodes (v_1 and v_2) are considered in the same partition if $\text{sgn}(\phi_2)(v_1) = \text{sgn}(\phi_2)(v_2)$ so then any two nodes other than these are part of the other partition or cluster. This method can be used on Zachary's katate club and partition reproduces the ground truth [22].

The Fiedler vector is known to provide a powerful approach to identifying communities, searching for significant links between communities (known as bottleneck links), and to increase the general connectivity of a network [7].

The Fiedler Vector and Clusters

Given any connected network, one way the Fiedler vector can be used with these clusters is in identifying subgraphs that are densely connected and sparsely linked to other clusters (that is, graph partitioning). If we compare two identical networks, the network 2.12 (a) can be clearly seen to show three clusters of nodes with each cluster sharing two edges with the other two clusters. Whereas in the second network, Figure 2.12 (b) it is much more difficult to partition the network into subgraphs for a human observer and even devising an algorithm to identify these clusters is difficult, particularly since the nodes have a limited area in the entire network graph. Finding the ratio cut (which minimizes the edge density defined as $\rho(V_1, V_2) = \frac{|E(V_1,V_2)|}{|V_1||V_2|}$ where $V_1 \cap V_2 = \phi, V_1 \cup V_2 = V$ and $E(V_1, V_2)$

denotes the set of edges in G that are shared between V_1 and V_2) presents a NP-complete problem, so entries in the Fiedler vector would commonly be used to divide the network into two clusters in $\mathbf{x}_F$ with positive entries being one subgraph and negative entries being the other subgraph. For dividing into more than two subgraphs, after the Fiedler vector partitions the network into two subgraphs, the Fiedler vector of each subgraph is calculated and this is continued until preferred number of subgraphs is found [7].



(a) **(b)**

Figure 2.12: Two different visualizations of a network consisting of $k = 20$ nodes. (a) Ordened node placement.(b) Random node placement.

Adjacency Spectral Partitioning or Sign Partitioning

A second spectral approach which was put forth in the late 1980s was to partition a network into more than two parts by using eigenvectors of the adjacency matrix which produces both positive and negative values which can be used to partition according to the sign pattern. Using extra eigenvectors to extend spectral techniques can be used to detect more than two communities in a network. Using sign patterns for further eigenvectors produces partitions of the matrix as quadrants, octants, and so forth [22].

Normalised Cut Criterion

This approach was proposed by Shi and Malik [22] and uses the measure

$$C_N(V_1, V_2) = C(V_1, V_2)\left(\frac{1}{Vol(V_1)} + \frac{1}{Vol(V_2)}\right) \tag{2.6}$$

where $Vol(V_1) = \sum_{i \in V_1} k_i$ and V_1, V_2 is a partition of the nodes. This method involves calculating a row-stochastic matrix, computing the eigenvector of the second largest eigenvalue, sorting the elements in increasing order, repeating $n-1$ in computing $C_N(V_i, \bar{V_i})$ where $V_i = \{v_1, \ldots, v_i\}$ and $\bar{V_i} = \{v_{i+1}, \ldots, v_n\}$, and partitioning the network into two clusters where $i_0 = \min_i C_N(V_i, \bar{V_i})$. This algorithm is repeated on the cluster with the largest value of λ_2^P until K clusters are acquired.

Variations on this algorithm have been proposed by Kannan et al., Ng et al., and Meila and Shi [22]. Meila and Shi [22] propose using eigenvectors that correspond to the largest eigenvalues of the community and then clustering the points in a condensed eigenvector matrix [22].

The main advantage of Meila and Shi's approach is that all eigenvectors are used to obtain the solution which addresses the problem of continuing to bisect a graph even when it in some cases does not leave us with the most natural partitions and not knowing when to stop bisecting [85]. Two possible explanations for why spectral methods make these kinds of partitions in networks are the polarization theorem and the consideration of the stochastic matrix P in the calculation of the normalised cutset. We can define P implicitly from the expression

$$P_{V_1,V_2} \frac{\sum_{i \in V_1, j \in V_2} \pi_i P_{ij}}{\pi_A} = \frac{\sum_{i \in V_1, j \in V_2} M_{ij}}{Vol(V_1)} = \frac{C(V_1, V2)}{Vol(V_1)} \tag{2.7}$$

where P_{V_1,V_2} the transition probability for a random walk going from the subset V_1 to V_2 in one step. By substitution in the expression for the normalised cut gives

$$C_N(V_1, V_2) = \frac{C(V_1, V_2)}{Vol(V_1)} + = \frac{C(V_1, V_2)}{Vol(V_2)} = P_{V_1 V_2} + P_{V_2 V_1}. \tag{2.8}$$

A small value for $P_{V_1 V_2}$ would show that random walks are localized in each partition resulting in two communities of nodes [22].

2.3 Quality of partitions

2.3.1 Modularity

Quality functions represent how good a partition or cluster is by finding the maximum of a function over the space of all possible clusters. Newman and Girvan's [64] general expression of modularity is

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij})\delta(C_i, Cj), \tag{2.9}$$

where m is the number of edges in the network, A_{ij} is the element of the adjacency matrix, P_{ij} is the null model term showing the average adjacency matrix of a group of networks, C_i and C_j are the communities of i and j and $\delta(C_i, Cj) = 1$ if i and j in the same community and $\delta(C_i, Cj) = 0$ if i and j in the different communities. Modularity measures how different the graph is from randomisations reflecting the rationale that randomising a network would destroy community structure so comparing the actual structure and its randomised one shows how non-random the group structure is. One usual approach is the term for the average adjacency matrix of a group of networks is taken to be $P_{ij} = \frac{k_i k_j}{2m}$, where k_i and k_j are taken to be the degrees of i and j which link to the number of edges we would expect for vertices i and j if the edges were rearranged in order to preserve the average degrees of all the vertices. The result is the classic form of modularity

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(C_i, Cj), \tag{2.10}$$

Other approaches involve using specific network features such as bipartiteness, correlations, signed edges, and space embeddedness. To extend the classic form of modularity to weighted networks is fairly simple (Newman, 2004), but generally the focus is on unweighted graphs. Vertex pairs belonging to the same cluster are grouped together and the sum is rewritten over the vertex pairs as a sum over the clusters is termed an additive as

$$Q = \sum_{C}[\frac{l_C}{m} - (\frac{k_C}{m})^2]. \tag{2.11}$$

Here l_C is the total number of edges joining vertices of community C and k_C is the sum of the degrees of the vertices of C. The difference in the square brackets above shows how non-random the subgraph C is. Larger values indicate that the arrangement of the edges in C is not random which reflects a high quality partition. $Q = 0$ when the number of intracluster links is not greater than the expected value in a random network. The highest value Q can have is 1 which would show a strong community structure. Values of Q are used to decide which partitions are of particular quality. Among the algorithms that use modularity optimisation to detect communities are agglomerative techniques and some modifications such as simulated annealing, extremal optimisation techniques and spectral methods. Similarly to Meila and Shi, White and Smyth [22] apply k-clustering techniques to split the network into various groups, calculate modularity for each possible group in every dimension for an eigenvector space and then find the maximum possible numbers of eigenvectors [22].

Performance Index

Van Dongen [22] defined the performance of a cluster by measuring coverage. The larger the value of coverage, the higher the quality of the cluster. Smaller values of the performance of the cluster, the better the quality of the cluster.

$$\text{Per}(C) = 1 - \frac{2m[1 - 2\text{Cov}(C)] + \sum_{i=j}^{c} \mid C_i \mid (\mid C_i \mid -1)}{n(n-1)} \tag{2.12}$$

where $\text{Cov}(C)$ is the ‘coverage’ of the cluster as characterized below

$$\text{Cov}(C) = \frac{1}{m}\sum_{i=1}^{c} \omega(C_i) \tag{2.13}$$

where $\omega(C_i)$ is the number of links in the cluster C_i, $\mid C_i \mid$ is the number of nodes in the cluster C_i, and m is the total number of links.

The Davies-Bouldin Validation Index

The Davies-Bouldin Validation Index, named for Davies and Bouldin [22] who proposed it, measures quality based on intracluster and intercluster distances and is defined as

$$DB(C) = \frac{1}{m}\sum_{i=1}^{c} \max_{i\neq j}\{\frac{\triangle(C_i)+\triangle(C_j)}{\delta(C_i,C_j)}\}. \tag{2.14}$$

There are several different definitions of the distance functions. In this index, dense clusters with centres far away from each other have small values of the index. As a result, when comparing two communities within a network, the smallest value of this index is considered to be of higher quality. There are many other measures for validation in the literature but we will focus only on modularity clearly the results we produce later could be checked using these measures in future work.

2.3.2 Similarity

The idea that communities share certain properties or that certain properties within a community have similarities can also be exploited to find communities within a network. However, Similarity measures need to be obtained in order to apply this idea to networks. These measures will use data from the adjacency matrix of the network.

The cosine similarity, which is the angle formed between two vectors, can be calculated as

$$\sigma_{ij} = \cos\phi_{ij} = \frac{x^T y}{\parallel x \parallel \parallel y \parallel} \tag{2.15}$$

To consider this similarity, the ith and jth columns of the adjacency matrix are two vectors with corresponding nodes. We use the cosine of the angle between them to measure node to node similarity . If the angle between them is perpendicular, the nodes are considered dissimilar with the lowest possibility of similarity between them. When this measure is used with two equivalent nodes, the angle between them is zero.

The Pearson correlation coefficient is another popular similarity measure. It is defined as

$$r_{ij} = \frac{na_i^T a_j - (a_1^T e)(a_j^T \mathbf{e})}{\sqrt{na_i^T a_i - (a_1^T \mathbf{e})^2}\sqrt{na_j^T a_j - (a_j^T \mathbf{e})^2}} \tag{2.16}$$

where a_i, a_j are columns of the adjacency matrix A and $\mathbf{e}$ is a vector of ones. Other measures such as the Manhattan or taxicab norm

$$\parallel a_i - a_j \parallel_1 = \sum_{k=1}^{n} \mid a_i(k) - a_j(k) \mid, \tag{2.17}$$

Euclidean norm

$$\parallel a_i - a_j \parallel_2 = \sqrt{\sum_{k=1}^{n} [a_i(k) - a_j(k)]^2}, \tag{2.18}$$

and Infinity norm

$$\parallel a_i - a_j \parallel_\infty = \max_{k \in [1,n]} \mid a_i(k) - a_j(k) \mid, \tag{2.19}$$

are norms of difference between corresponding columns of the adjacency matrix.

To use a similarity measure, start by choosing one for the nodes of a network where we want to detect a community and then find successive clusters using various linkage approaches.

One such approach, single linkage, whereby determining the nearest neighbours in different clusters, the distance between the clusters can be calculated.

The complete linkage approach takes the opposite approach were the farthest neighbours are calculated using the greatest distance between any nodes in two different clusters. If the clusters are chain-linked, this approach will not work.

When the average distance between nodes in two different clusters and in all pairs are calculated, we can use the unweighted pair-group average to find clusters whereas in the similar weighted pair-group average, the size of each cluster is used as a weight.

The unweighted pair-group centroid approach is when the centre of a uniform density cluster is calculated using the average point in the multi-dimensional space and similarly, the weighted pair-group centroid (median) is the same except the weighting of each cluster

is taken into account using computations to allow for differences in cluster size.
Finally, Ward's method analyses the variance approach to calculate the distances between clusters [22].

2.4 Community detection methods

Centrality is a very important concept in network theory which used to determine the important nodes in a network. Centrality comes in different ways and each way determines a node's importance from a different perspective and further gives us analytical information about the nodes and the graph. There are various measures for centrality like degree centrality, closeness centrality and betweenness centrality. We will look at some of them in this thesis.

2.4.1 Link centrality

Links that are central to communication within clusters can be used to detect communities. Removing them one by one in succession is one approach by Girvan and Newman (2002, 2004) which is based on the idea of calculating edge betweenness centrality. Calculating this betweenness centrality involves counting the number of shortest paths between two nodes that pass through a particular link and dividing by the total number of shortest paths. Links between nodes in different communities have the highest edge betweenness. As such, using this method provides a way to find links between different communities.
Once this edge betweenness is calculated for all links in a network, applying Girvan and Newman's approach as summarised above, then recalculating the edge betweenness for the remaining links repeating until all links have been removed, and finally plotting a dendrogram (which is a hierarchical tree with clusters as a nodes in tree, and single nodes as leaves) in order to analyse the community structure of the network enables us to find the links that are essential to communication between communities. A range of partitions are possible with this method and in the next section we will discuss more about how to decide how many divisions is most useful.
Another way to use link centrality is a link clustering coefficient studied by both Radicchi

et al. and Watts and Strogatz [22] which can be defined as

$$C^l_{i,j} = \frac{z^l_{i,j} + 1}{s^l_{i,j}} \tag{2.20}$$

where $z^l_{i,j}$ is the number of cycles of length l that pass through the link between i and j, and $s_{i,j}$ is the maximum possible value that $z^l_{i,j}$ can take. By removing the links with the smallest clustering coefficient and recalculating for $C^l_{i,j}$ for the remaining links until none are remaining in the network. We produce a clustering coefficient that takes advantage of the fact that a community is characterised by a high density of links which are not expected to be shared between two partitions within the same network.

2.4.2 Communicability

We can measure how much information is transferred from one node to another in a network which is termed ‘communicability’. In fact, communities as groups of nodes that communicate more between their members than with people outside their group is another useful definition of a community or cluster within a network. Nodes can be in one of three states: each node can have a positive or negative point of view with respect to some measurable criterion or no opinion (the off state). Those who have similar points of view tend to cluster into communities. This idea can be applied when looking at networks of energy configuration where nodes exist in either a positive or negative state. Networks that want to reflect communities that maximize energy would put nodes in the positive and negative state close together [22].

When we consider a pair of nodes p and q the communicability can be measured by

$$G_{pq}(\beta) = \sum_{j=1}^{n} \phi_j(p)\phi_j(q)e^{\beta\lambda_j} \tag{2.21}$$

where λ_j and ϕ_j are eigenvectors and eigenvalues of the adjacency matrix. Focusing on the case of $\beta = 1$ and recalling that nodes in an off state do not contribute to the communicability function, allows us to express the function as

$$G_{pq} = \phi_1(p)\phi_1(q)e^{\lambda_1} + \sum_{\phi_j(p)\phi_j(q)\geq 0} \phi_j(p)\phi_j(q)e^{\lambda_j} + \sum_{\phi_j(p)\phi_j(q)<0} \phi_j(p)\phi_j(q)e^{\lambda_j}. \quad (2.22)$$

The first term on the righthand side is the consensus configuration where all nodes share the same state. The second term on the righthand side (or intracluster communicability) have the nodes p and q as the same sign of the corresponding eigenvector (positive or negative). The last term on the same side (intercluster communicability) represents a lack of consensus or different signs of the eigenvector component so they do not communicate well meaning they are in different communities in the network [24]. To calculate the difference between intra- and intercluster communicability, we rewrite the consensus configuration as below:

$$\begin{aligned}\Delta G_{pq} &= \sum_{\phi_j(p)\phi_j(q)\geq 0} \phi_j(p)\phi_j(q)e^{\lambda_j} + \sum_{\phi_j(p)\phi_j(q)<0} \phi_j(p)\phi_j(q)e^{\lambda_j} \\ &= \sum_{j=2}^{\text{intracluster}} \phi_j(p)\phi_j(q)e^{\lambda_j} - \left| \sum_{j=2}^{\text{intercluster}} \phi_j(p)\phi_j(q)e^{\lambda_j} \right|.\end{aligned}$$

$\Delta G_{pq} > 0$ demonstrates that p and q show larger intracluster than intercluster communicability.

2.4.3 Optimisation

Optimisation techniques which maximize a function measuring cluster quality have been the area of focus for community detection.

Drawbacks to modularity maximisation

Modularity maximisation is considered NP-hard [9] so the hope is to find decent approximations and there are several approaches with modularity proposed by Girvan and Newman being the most popular. Despite its popularity, there are several problems associated with its use. Because modularity does not take into account the distribution of different network features, it is possible to find high modularity partitions even in random graphs without groups [35].

Resolution Limit Problem or the Ring of Cliques

Consider cliques of four vertices in a ring-like structure with each clique joined to two others by a shared edge. This is named the ring of cliques by Fortunato and Barthélemy [31] as seen in Figure 2.13. We might expect that modularity would reach its maximum for partitions whose cliques are community, but the modularity is larger when subgraphs are put together than when they are treated as separate communities, and the modularity scale is dependent only on the number of edges m, unrelated to community size.

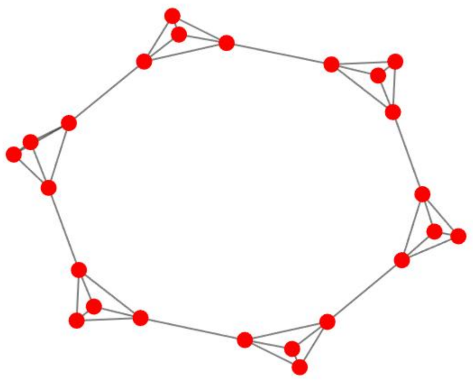


Figure 2.13: Example of the resolution problem of modularity.

This issue is known as resolution limit and occurs where networks with small communities which are adjacent to bigger communities or many small communities that are circularly connected are incorrectly partitioned into pairs of modules rather than single cliques. It is also important to note that these problems are not unique to modularity [22].

A multi-resolution approach, which involves introducing a resolution parameter γ into the modularity formula and tuning it to adjust the resolution scale of the method when going from very large to very small communities [5], is popular but does not always solve the problem reliably because with modularity maximisation large subgraphs are typically split into smaller pieces [55]. This problem has the same origin as the resolution problem, that is, its use of the null model, which assumes that each node can attach to any other node in the network, which does not make sense in a very large network. Furthermore, in a null model, this assumption would result in the expected number of edges in a large enough network being less than one, and one edge shared between two groupings would indicate a strong correlation between the two which with optimisation would merge the two clusters irrespective of their features. The end result is that small communities in large networks are not recognized even when they are well-defined [31].

2.4.4 The Louvain Method

Circumventing the resolution limit issue, which is an actual feature of modularity rather than an issue with methods of maximising it, can be done using the Louvain method first proposed by Blondel et al. [8]. It is a heuristic method based on modularity optimisation that finds high modularity partitions of networks and complete hierarchical community structure in a short time. It is based on a similar earlier method proposed by Clauset, Moore and Newman [15] where communities are merged repeatedly to optimise modularity. The drawbacks to this earlier method were the production of lower modularity values than other methods such as simulated annealing [35], and a propensity to produce 'super-communities' containing many nodes even within networks that had no notable community structure which resulted in slowing processing times so that networks exceeding a million nodes were not practical [15].

In contrast, the Louvain method is a greedy optimisation of Q that has two phases that are repeated iteratively. Starting with a weighted network of N nodes, a different community is assigned to each node of the network so there are as many communities as nodes. Each node is considered and the gain in modularity when i is removed from its community and placed in the community of its neighbour j is evaluated. Node i is then placed in the community only where the gain is maximised and positive. This process is repeated until no further improvements can be made. It should be noted that the order does not affect modularity but it does affect computation time. The formula

$$\Delta Q = [\frac{\sum_{\text{in}} + k_{i,\text{in}}}{2m} - (\frac{\sum_{\text{tot}} + k_i}{2m})^2] - [\frac{\sum_{\text{in}}}{2m} - (\frac{\sum_{\text{tot}}}{2m})^2 - (\frac{k_i}{2m})^2] \tag{2.23}$$

is used to calculate the gain in modularity. $\sum_{\text{in}}$ is the sum of the weights of the links inside the community C, $\sum_{\text{tot}}$ is the sum of the weights of the links incident to nodes in C, k_i is the sum of the weights of the links incident to node i, $k_{i,\text{in}}$ is the sum of the weights of the links from i to nodes in C and m is the sum of the weights of all the links in the network. A similar expression is used to evaluate the change of modularity when i is removed from its community.

In the second phase, a new network whose nodes are now the communities found during the first phase is built using the weights of the links between the new nodes which are found using the sum of the weight of the links between nodes in the two corresponding communities. One cycle of the process of both phase one and phase two is termed a "pass". The number of communities decreases with each pass and the passes continue until there are no more changes and maximum modularity is reached.

There are a number of advantages with this algorithm. It is easy to implement and does not require supervision to achieve the outcome. It is also very fast with most of the calculation time used in the first iteration. It circumvents the resolution limit problem of modularity because there is a low probability that two communities are able to merge by moving nodes one by one. Additionally, this algorithm finds communities at different levels of organization thus uncovering hierarchical structure as it maximises modularity [8]. High modularity partitions are not necessarily similar to each other in spite of the similarity in their modularity scores. The best partition may not have the highest Q-value but amongst a number of high scoring partitions, it may be very difficult to distinguish the best partition from the rest. In order to choose, the user could restrict the clustering to certain parameters based on specific features she could expect the community to have. If she does not know or does not have any particular expectations, consensus clustering could be used and combined with a hierarchical approach could address resolution limit problems as well as helping to avoid finding communities in networks without any [87].

2.4.5 Dynamics

Dynamical processes like diffusion, spin dynamics, and synchronisation can be used to detect communities. Specifically, most of these processes are based on diffusion and spin dynamics.

Vertex Similarity

Methods based on vertex similarity exploit these random walk dynamics. One set of techniques includes using random walk dynamics to estimate similarity between pairs of vertices. One method by Pons and Latapy [72] is called Walktrap, where community structure is calculated in time $\mathcal{O}(mnH)$ where H is the height of the dendrogram. H is usually small and yields the best results when the dendrogram is balanced ($H = \mathcal{O}(\log n)$). Within a random walk, the probability for each step is $P_{ij} = \frac{A_{ij}}{d(i)}$. P is the transition matrix of random walk processes. The vertex similarity can be measured using P and is thought to be considerably higher within groups than between groups. Clusters can be identified by hierarchical cluster or partitional clustering, but this algorithm cannot be used on larger networks [72].

Map Equation

This approach came about when trying to achieve the best way to describe long random walks taking place in the graph. In this process, all vertices are listed sequentially, then assigned a unique codeword although there are places where there is an overlap of names. The equation results in the description length of an infinite random walk in two terms: the Shannon entropy and the minimum description length.

Info map can be applied to both weighted and unweighted, directed and undirected networks. A teleportation probability as in PageRank [11] is introduced for random walk dynamics. The rule was then expanded to detection of hierarchical community structure and clusters which overlap within a network and then to higher order Markov dynamics where transition probabilities are used in other steps thus retaining memory of the recent past [11].

In comparison to structure-based methods such as modularity and optimisation, Infomap and other variants usually return different partitions because they are not based on number of edges, vertex, etc. but on flows running across the system. Most methods in this section discussed so far are global, meaning that they find community structure for the entire network. Random walks can, however, be used to find communities locally using seed vertices [32][46].

When examining some existing methods for community detection, we must recognize that because there is no overriding mathematical definition of community, each method is not necessarily exactly comparable with another given that each set of researchers may define community differently. Methods for partitioning graphs usually require that the number of partitions is known in advance which is not usually the case for complex networks. Modularity methods which were first explored by Newman and Girvan, do not have this issue and are popular because they also evaluate the quality of the partitioning. Other methods which have their basis in community detection are algorithms based on structural equivalence [57], stochastic block model methods [65], [70] , [69], and hierarchical clustering algorithms [30].

2.5 Anti-Communities

Detecting communities allows us to identify components that are functionally related. However, not all communities in complex networks possess this structural feature. The term 'anti-community' was first used by Newman in a 2006 paper where he described bipartite and multipartite structures in networks [63]. An anti-community can be defined as an area of a network which has a significantly low density of connections between its nodes but a high density of connections outside of it. While there has been much discussion within the field of detecting areas of densely connected nodes—communities, there has been much less investigation of algorithms to locate anti-communities.
The idea of clusters or communities in a network has to do with shared characteristics of that network. It is noteworthy that some networks have communities with opposing

properties where the nodes have no connection with other nodes, that is, little to no internal edges but many external edges. Two major studies of anti-communities have been with protein-protein interactions and conflicts in traditional Chinese medicine [27],[88]. These anti-communities can also be viewed as bipartitions since they are those areas where there is less density between nodes. Finding anti-communities is closely connected to finding communities in the graph complement. Mathematically, the same principle can be applied as they are comparable. Practically, the runtime complexity represents a significant barrier in a dense graph that exceeds a certain number of edges.

2.5.1 A Spectral Bipartization Method

Using the spectral structure of a bipartite graph to calculate a permutation in the node of the V_1 and V_2 sets provides us with a method of detecting communities in bipartite graphs. Once the permutation is calculated, we can approximate and then construct a bipartite graph to a connected, undirected graph G which has a perturbed bipartite structure. The authors of this approach were unable to deduce a complete convergence approach and so have classified this method as heuristic [17].

With this approach, there are three problems that need to be addressed. The first is determining a way to estimate the size of sets V_1 and V_2. The authors determine cardinalities of n_1 and n_2 by finding the number of eigenvalues that are around zero. The most reliable method they found was to figure out the largest gap between the small and large eigenvalues by computing the ratios

$$\rho_i = \frac{\mid \lambda_{i+1} \mid}{\mid \lambda_i \mid}$$

where λ_i is the large eigenvalue and considering the index set for the chosen constants R and τ. The index set is as follows

$$\mathcal{J} = \{i \in \{1, 2, \ldots, n-1\} : \rho_i > R \text{ and } \mid \lambda_{i+1} \mid > \tau\}. \tag{2.24}$$

A significant gap between λ_i and λ_{i+1} denotes that $\mathcal{J}$ (see 2.24) has an index. If $\mathcal{J}$ is empty, then the cardinality of V_1 and V_2 are considered to be the same.

The second issue is ordering the nodes in G in a suitable fashion. To address this issue, G is assumed to be bipartite and yet the adjacency matrix A parallels with the nodes in random order so that

$$A = \Pi A_B \Pi^T$$

where $A_B = \begin{bmatrix} O_{n_1} & C \\ C^T & O_{n_2} \end{bmatrix}$, O_k is the $k \times k$ zero matrix, and $C = [c_{i,j}] \in \mathbb{R}^{n_1 \times n_2}$ with $c_{i,j} > 0$. The eigenvector matrix is partitioned so the structure of the eigenvector can be recovered.

The third issue is how to approximate the adjacency matrix by a matrix of the form

$$A_B = \begin{bmatrix} O_{n_1} & C \\ C^T & O_{n_2} \end{bmatrix}$$

First, the eigenvector matrix is approximated W_B by solving

$$\min_{U_1^T U_1 = V^T V = \frac{1}{2} n_2 I, U_2^T U_2 = \frac{1}{2} I n_1 - n_2} = \left\| \begin{bmatrix} U_1 & U_2 & U_3 \\ V & O & -V \end{bmatrix} - \begin{bmatrix} W_{11} & W_{12} & W_{13}Z \\ W_{21} & W_{22} & W_{23}Z \end{bmatrix} \right\|_F$$

(where $\| \ . \ \|_F$ denotes the Frobenius norm) and then approximating the eigenvalues using specified scalars [17].

2.5.2 Anti-modularity

In 2014, Chen et al. published a paper proposing the term 'anti-modularity' to quantitatively measure anti-community partitioning in a network by using a label propagation algorithm [13]. Their rationale for proposing this measure was that all the community detection methods that were adapted to find anti-communities would not necessarily result in the best partitioning for these anti-communities. Consequently, they claimed these adapted methods would not be as precise or as accurate [13]. They define anti-modularity as "the difference between the number of paths connecting the vertices within groups passing a vertex in other group, and the expected number in an equivalent network with edges placed at random" with positive values showing the presence of anti-community structure with the larger the value of anti-modularity, the larger the anti-community. The anti-modularity is defined as:

$$Q = \frac{1}{n} \sum_{g=1}^{c} \sum_{v_j \in v_g} (\sum_{k=1}^{n} a_{ik} a_{kj} - \frac{d_i d_j}{n})$$

To maximise the anti-modularity within all the possible graph partitions, finding the anti-modularity becomes an optimisation problem with it being expressed in terms of the eigenvalues of the anti-modularity matrix

$$M = A^T A - \frac{1}{n} D^T D$$

where $D = (d_1, d_2, \ldots, d_n)$ is the vector with the degrees of the vertices. Chen et al. goes on to show how reliable their measure is by maximising the anti-modularity by clustering the column vectors of the graph in the adjacency matrix which leads to an NP-complete problem [66] since we do not know the size of the communities and guessing will prevent the best solution from being found. High values of Q represent an anti-community partition of high quality so optimising Q over all the possible groupings to find the best one would seem logical. Carrying out a search of all the vertices would require a massive amount of time, so they devise a label propagation algorithm which is faster than all other methods in the literature [13]. They then test their algorithm on artificial and real-world networks.

Further to their work, Zhu et al. have proposed two heuristic methods that attempt to minimise modularity and Fasino et al. puts forth a spectral method to simultaneously detect both communities and anti-communities using modularity minimisation or maximisation [27]. There are few dedicated algorithms for anti-community detection and the ones that do exist are either not feasible because of the storage space it would require and the runtime [53].

Since the optimisation problem is NP-hard, heuristic approximation algorithms where a greedy algorithm makes a locally optimal choice rather than searching for a global optimum which would take much more time.

Chapter 3

Random models with community structure

3.1 Random graphs and graph generators

Working with random graph networks can provide us with insight into some of the issues we encounter when trying to analyse real world networks, so long as the randomly generated networks sufficiently resemble real-world networks. It is well known that some of the basic random models lack crucial features that are important in practice and we aim to devise methods to overcome this lack. For instance, there are many applications where scientists and engineers need to isolate small fragments to understand their role within the whole system and techniques to quantify these fragments or subgraphs can be very useful. The frequency of fragments that inevitably arise through network connectivity may be dissimilar to equivalent random networks, hence poorly chosen random networks may not be useful for inferring behaviour in real world networks [25], [22].

In this section we describe some popular random graph models and list some of their properties.

3.1.1 Erdős–Rényi

Probably the simplest model of random graphs is the Erdős–Rényi (ER) model. Erdős and Rényi characterized a class of random graphs and showed that many of the properties of such networks can be calculated analytically. This minimal model consists of n nodes, joined by edges which are placed between pairs of nodes chosen uniformly at random. Erdős and Rényi gave a number of versions of their model. The most commonly studied is the one denoted $G_{n,p}$, in which each possible edge between two nodes is present with independent probability p, and absent with probability $1-p$. Technically, in fact, the model is the ensemble of graphs of n nodes in which each graph appears with the probability appropriate to its number of edges [25],[22].

We list some properties of the random graph model. To form $G_{n,p}$, each pair out of $N = \frac{n(n-1)}{2}$ pairs of nodes is connected with probability p.

1. The mean number of edges is $\bar{m} = \frac{pn(n-1)}{2}$.

2. The expected node degree $\bar{k} = (n-1)p$.

3. The probability $p(k)$ follows a binomial distribution of the form

$$p(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k}.$$

   For large n with fixed $\bar{k}$, $p(k)$ become

$$p(k) = \frac{e^{-\bar{k}}\bar{k}^k}{k!}$$

   which is the Poisson distribution.

4. The average path length for large n with fixed $\bar{k}$ is $\bar{l}(G) = \frac{\ln n - \gamma}{\ln(\bar{k})} + \frac{1}{2}$ where $\gamma \approx 0.577$ is the Euler–Mascheroni constant.

5. The Watts–Strogatz clustering coefficient is $\bar{C} = p$ which means that the clustering coefficient for sparse Erdős–Rényi random networks is very small, much smaller than for real world networks with the same density [22].

6. As p increases, most nodes tend to be clustered in one giant component, while the rest of nodes are isolated in very small components and the structure of $G_{n,p}$ changes as a function of $\bar{k}$, giving rise to the following three stages.

   - Subcritical ($\bar{k} < 1$), where all components are simple and very small. The size of the largest component is $S = O(\ln n)$.
   - Critical ($\bar{k} = 1$),where the size of the largest component is $O(n^{\frac{2}{3}})$nodes.
   - Supercritical ($\bar{k} > 1$), where the probability that $(f - \epsilon)n < S < (f + \epsilon)n$ is 1 when $n \to \infty, \epsilon > 0$, and where $f = f(\bar{k})$ is the positive solution of the equation $e^{-\bar{k}f} = 1 - f$. The rest of the components are very small, with the second largest having size about $\ln n$.

7. The largest eigenvalue of the adjacency matrix in an ER network grows proportionally to n so that $\lim_{n\to\infty} \frac{\lambda_1(A)}{n} = p$.

8. The second largest eigenvalue grows more slowly than λ_1. In fact, $\lim_{n\to\infty} \frac{\lambda_2(A)}{n^{\epsilon}} = 0$ for every $\epsilon > 0.5$.

9. The most negative eigenvalue grows in a similar way to $\lambda_2(A)$. Namely, $\lim_{n\to\infty} \frac{\lambda_n(A)}{n} = 0$ for every $\epsilon > 0.5$.

10. The spectral density of $G_{n,p}$ follows Wigner's semi circle law [22].

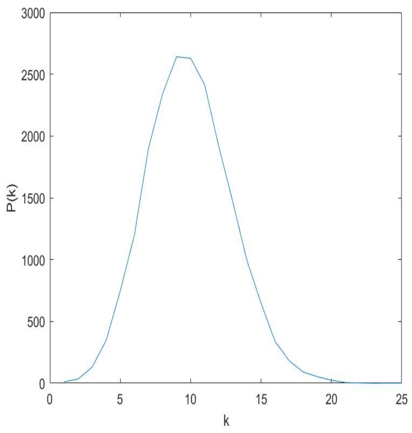


Figure 3.1: Erdős–Rényi degree distribution for an instance.

Figure 3.1 shows degree distribution for average of 100 instances of an Erdős–Rényi network with $n = 100, p = 0.1$.

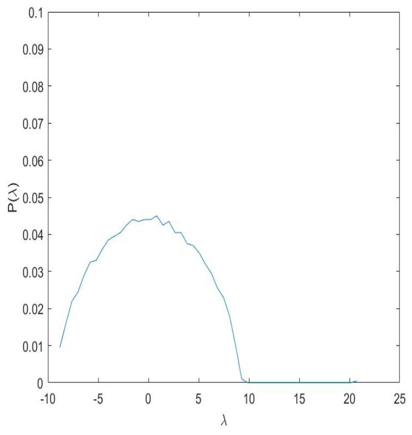


Figure 3.2: Erdős–Rényi eigenvalue distribution.

Figure 3.2 shows the eigenvalue distribution for an instance of an Erdős–Rényi network with $n = 2000, p = 0.1$. Notice the small uptick at the right hand edge of the graph (this is λ_1) [25][22].

3.1.2 Scale-free networks

While ER networks are a very useful theoretical tool they lack an important property found in many real-world networks, namely that while most nodes have only a few links to other nodes, a small number of nodes are highly connected with a huge number of links to other nodes. This leads to the observation that real-world networks do not have nodes with a typical number of neighbors, and in this sense these networks are scale-free. The modern investigation of scale-free networks began with Barabási and Albert.

To generate a Barabási–Albert (BA) network we begin with an initial connected network of m_0 nodes. New nodes are added to the network one at a time. Each new node is connected to $d \leq m_0$ existing nodes with a probability that is proportional to the number of links that the existing nodes already have. This process is known as preferential attachment. For example, we assume that we begin with a connected ER network $G = (V, E)$ with m_0 nodes. In this case the Barabási–Albert (BA) algorithm can be understood as a process in which inhomogeneities in the degree distribution of the Erdős–Rényi (ER) network grow in time.

We summarise some key properties of BA networks.

1. The probability of a node with degree $k \geq d$ is

$$p(k) = \frac{2d(d-1)}{k(k+1)(k+2)} \approx k^{-3}$$

that means the distribution resulting from the BA model close to a power law (power law are usually represented on logarithmic scale then appears as straight line but the degree distribution of some models do not look so smooth) as in Figure 3.3 with $n = 1000, d = 10$.

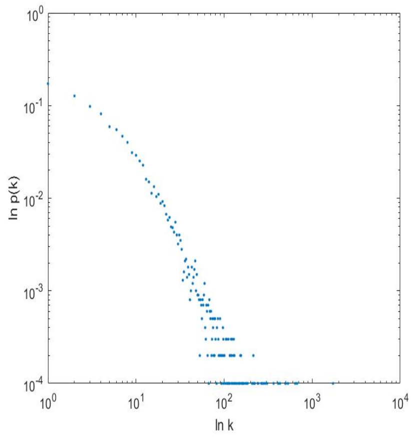


Figure 3.3: The degree distribution of a model BA network

2. The average path length of the BA model is

$$\bar{l} = \frac{\ln n - \ln(\frac{d}{2}) - 1 - \gamma}{\ln \ln n + \ln(\frac{d}{2})} + \frac{3}{2},$$

where γ is the Euler-Mascheroni constant.

3. The cumulative degree distribution is $P(k) \approx k^{-2}$.

4. The expected value for the clustering coefficient, is $\bar{C} = \frac{d-1}{8}\frac{\log^2 n}{n}$ as $n \to \infty$

5. As shown in Figure 3.4 the spectral density of BA model has a different shape from the semicircular spectral density of random graph. It has a triangle-like shape with the top lying well above the semicircle and edges decaying as a power law [25], [22].

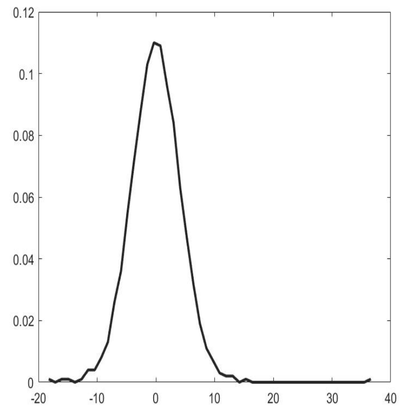


Figure 3.4: Spectral density of a model BA network.

6. The largest eigenvalue of the adjacency matrix in BA network increases approximately as $n^{\frac{1}{4}}$ [2].

3.1.3 Effect of adding triangles on fragments (ER, BA)

Motifs and Fragments

Given the significance of fragments in real world networks, the frequency with which they appear suggests that there is a function or logic to this over expression. Adapting random networks to reflect real world networks by taking the average nodes that have the same degrees as real ones, we can see which fragments can be over and under represented. A fragment that appears in a real network an equal or greater number of times than in random networks is motif. Conversely, fragments which appears less frequently than would be expected are anti-motifs.

Real world networks contain small structural pieces that occur with certain functions of the system see Figure 3.5. Isolating these pieces helps us to understand how they work and how the roles in the system work together.

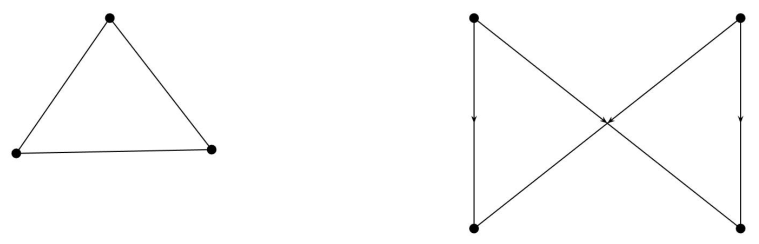

(a) Triangle (b) Bi-fan

Figure 3.5: Triangle and Bi-fan

Various techniques will be discussed to determine the number of these fragments within a network and whether or not their presence indicates a significant function or is simply a reflection of a random process. Another way to describe these fragments are as subgraphs. Some techniques to count different types of subgraphs include counting stars and using closed walks. These two techniques can also be combined. Other techniques can be used to find subgraphs with a diamond shape. While subgraphs with certain shapes can certainly be found in connected networks, their occurrences may be random as we can identify when comparing real world graphs to random graphs. In this circumstance, we cannot use these occurrences whether frequent or not to explain how the structure of that network evolved. Nevertheless, over expression of a fragment beyond that which would be expected suggests there is a structural or functional reason. We can consider this subgraph or fragment to be a network motif when the probability P of its presence in a real world network an equal or greater number of times than in a random network is $P_C = 0.01$. Quantifying statistical significance is calculated using the Z-score (Z_i) where $Z_i = \frac{N_i^{real} - \langle N_i^{random} \rangle}{\sigma_i^{random}}$, where σ_i^{random} is the standard deviation of the number of times that i appears in an ensemble of random networks and the relative abundance of a particular fragment as measured by

$$\alpha_i = \frac{N_i^{real} - \langle N_i^{random} \rangle}{N_i^{real} + \langle N_i^{random} \rangle}, \tag{3.1}$$

where N_i^{real} is the number of times the subgraph i appears in the real network and $\langle N_i^{random} \rangle$ is the average of the number of times that appears in an ensemble of random networks. If α_i is close to zero then the frequency is about right. If it is positive then the real network has more than the random network and the fragment is a motif. A defining characteristic of network motifs are that they are specific to a particular network but it should also be noted that families of networks can be found if they share the same groupings of motifs. Subgraphs are over represented in some networks and underrepresented in others as is usually the case for motifs in undirected networks. This under representation where fragments are less frequently found in real world networks than in a corresponding random network are called anti-motifs.

In our experiment we look at the frequencies of a number of small fragments. F_1 is a path of length two, F_2 is a 3-cycle (or triangle), F_3 is a path of length three, F_4 is a star with 3 points, F_5 is a 4-cycle (or square), F_6 is characterized by having a node which is simultaneously part of a triangle and a path of length one, often known as a tadpole, F_7 is a diamond, F_8 is a 5-cycle (pentagon), F_9 is characterized by having a node which is simultaneously part of a triangle and a path of length two, F_{10} is characterized by having a node which is simultaneously part of a 4-cycle and a path of length one, F_{11} is reindeer, F_{12} is characterized by having a node which is simultaneously part of a triangle and a path of length two, F_{13} is bowtie, F_{14} is a house and F_{15} is 6-cycle. We expect that only some of the fragments would be motifs in real world networks and some of them would not [25].

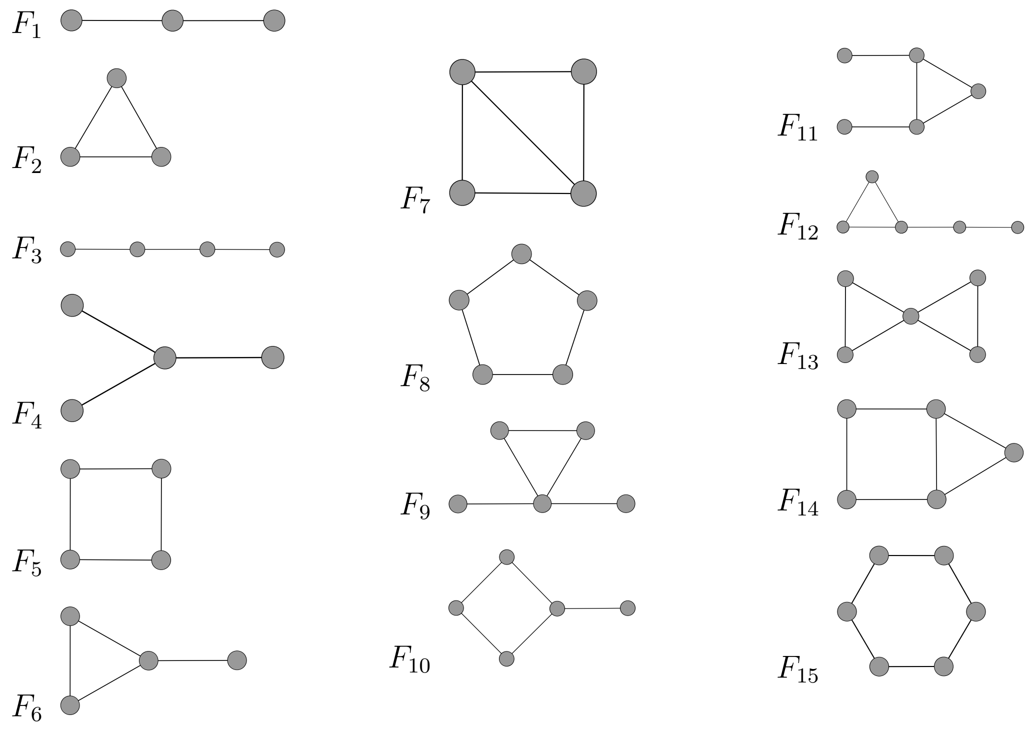


Figure 3.6: $F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8$, F_9, F_{10}, F_{11}, F_{12}, F_{13}, F_{14} and F_{15}.

To explore the question of whether real world networks have motifs, we examined ten real networks (Dolphins, ColoSpg, Centrality, Pin Malaria ,Corporatepeople, Canton, GD, Drugs, Little Rock and Stony) representing different system in the real world by computing 15 fragments F_1-F_{15}. These are illustrated in Figure 3.6. It is well known that triangles frequently appear as a motif in real-world networks.

We wanted to see the effect of artificially adding triangles to random networks and a Block Two-Level Erdős-Renyi (BTER) model, which has a community structure in the form of dense ER subgraphs and matches well with real-world graphs. In the first level, BTER builds a collection of ER blocks in such a way that the specified degree distribution is respected, and the second level interconnects the blocks [77].

The benefit of the BTER model are its dense ER subgraphs and that it pairs well with real-world graphs. In what is termed the preprocessing state, the nodes that have degree 2 or higher are grouped into communities. The desired degree distribution is designated as d_i. Though the process is more complicated, roughly speaking d vertices that are close to d are placed in certain scale-free communities. Then, G_k represents the kth community and k_i is the assigned community for the node i.

In level 1, each of the distributed communities from the preprocessing stage has links modelled using the ER model and their connectivity used as a parameter of the model. Clustering coefficients for real graphs show that degrees with lower vertices have a significantly higher clustering coefficient than the ones with higher degrees. The connectivity is then adjusted as these results illustrate that large communities are less closely connected than smaller ones. Overall, these connections tend to be dense leading to larger clustering coefficients.

In level 2, the same communities that have been locally linked are now globally linked by applying a Chung and Lu (CL) model to the excess degree, e_i of each node. It is computed using

$$e_i = \begin{cases} 1 & \text{if } \ d_i = 1 \\ d_i - \rho_{k_i}(\mid G_k \mid -1), & \text{otherwise} \end{cases}.$$

where G_k is the size of community k. Duplicate links are discarded. It should be noted that in this level, generating extra edges to account for repeats and self-loops can be done. Overall, these connections tend to be sparse leading to heavy-tailed degree distributions.

Comparisons of the BTER to real world networks are well-matched with any degree distribution. However, where the BTER truly stands out is when considering the clustering coefficient where it is seen to provide a much closer match to the real data than other models such as the CL proving the BTER builds communities of differing sizes while also producing a heavy tail. Thus, it is a fitting model to test algorithms and architectures for interaction graphs, that is, graphs of social relationships, collaboration, computer network traffic, and the like. The added benefit of the BTER is that it is scalable and in fact in level 2 could be used to compute the exact excess degree and complete the graph [77].

3.1.4 Methods

1. The 15 fragments for real world networks were computed using Matlab to find the number of nodes, number of edges and the 15 fragments.

2. We used the Erdős–Rényi model, the Barabási–Albert model and BTER model and adapted them to the networks being studied in this experiment by fixing the nodes to match the real-world graphs and ensuring that on average the number of edges was matched, too.

3. We computed the fragments for each one.

4. We calculated the relative abundance of each given fragment by using the statistic α_i defined in equation (3.1). If α_i is close to zero then the frequency is about right. If it is positive then the real network has more motifs than the random network. If it is negative then it has less than the random network. If a subgraph is under represented it is known as anti motifs.

5. Then we added triangles to both models (the Erdős–Rényi model and the Barabási–Albert model) so that they had approximately the same number as their real world counterparts. To add triangles we target 2-paths in A which are not yet triangles. These can be identified by finding entries of A^2 which are nonzero while the corresponding entry of A is zero. If $A(i,j) = 0$ and $A^2(i,j) > 0$ then by setting $A(i,j)$ and $A(j,i)$ to 1 we have added a triangle. Adding k (random) edges adds (at least) k new triangles.

6. Then re-computed the new α_i

$$\alpha_i = \frac{N_i^{\text{real}} - \langle N_i^{\text{random}+T} \rangle}{N_i^{\text{real}} + \langle N_i^{\text{random}+T} \rangle}$$

.

7. Finally, we made a table for each network. The table contains 19 rows and 7 columns. The main row contains real world networks, α_i before adding triangles to ER and α_i after adding triangles to ER, α_i before adding triangles to BA and α_i after adding triangles to BA and α_i for BTER, The main column contains a number of nodes n, number of edges m, the average of edge for random networks, F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9, F_{10}, F_{11}, F_{12}, F_{13}, F_{14} and F_{15} .

We compute the speed of the algorithm in the following table and we see that the time increases as the number of nodes increases:

Networks	n	m	C3	Time for ER+T	Time for BA+T	Time for BTER
Dolphin	62	159	95	0.3	0.8	26
ColoSpg	324	347	17	0.4	1.4	25
Centrality	118	613	1107	0.6	1.5	13
Pin Malaria	229	604	201	0.8	1.8	14
Corporate People	1586	11540	28002	22.4	44.9	371
Canton	108	707	116	0.4	1.5	14
Drugs	616	2012	3598	5.9	4.1	21
GD	249	635	327	0.5	2	14
Little Rock	181	2311	10489	3.1	5.4	21
Stony	112	830	124	0.6	1.6	14

Table 3.1: The speed of the algorithms in milliseconds.

3.1.5 Results

We colour the results in the following tables by green if the relative abundance is in the range $[-0.2, 0.2]$ and by red if it is not in the range $[-0.2, 0.2]$.

In the Dolphin, Centrality and Corporate people, Pin Malaria, Drugs and GD we can see that the relative abundance of BTER is the best. In Dolphin, Centrality ,Canton and GD the relative abundance is decreased after adding triangles to Erdős–Rényi model and Barabási-Albert model.

	real	ER	ER+T	BA	BA+T	BTER
n	62					
m	159					
av(m)		160		161		
F_1	923	0.6525	-0.0238	-0.1791	-0.1964	0.044
$C_3 = F_2$	95	0.9515	0.0146	0.2657	0.0765	0.0281
$P_3 = F_3$	5023	0.8287	-0.0683	-0.2325	-0.2853	0.0358
$S_{13} = F_4$	1861	0.8461	-0.0712	-0.5899	-0.6054	0.0501
$C_4 = F_5$	278	0.9655	0.0692	0.08	-0.0471	0.0423
$T_{31} = F_6$	1644	0.9802	-0.0077	-0.079	-0.2522	0.065
Diamond $= F_7$	300	0.9977	0.1729	0.3551	0.108	0.0334
$C_5 = F_8$	906	0.9771	0.0093	-0.0408	-0.1836	0.0573
$C_r = F_9$	4675	0.9923	-0.0792	-0.6203	-0.713	0.0889
$T_{41} = F_{10}$	5976	0.9848	0.0018	-0.2551	-0.3739	0.0578
Reindeer $= F_{11}$	9012	0.9918	-0.0452	-0.3056	-0.4452	0.0831
$T_{32} = F_{12}$	7879	0.9901	-0.0925	-0.0512	-0.2779	0.0314
Bowtie $= F_{13}$	586	0.998	-0.06242	-0.1016	-0.41	0.0827
House $= F_{14}$	1887	0.9971	0.1232	0.192	-0.0731	0.0386
$C_6 = F_{15}$	3232	0.9896	-0.0708	-0.105	-0.2617	0.0637

Table 3.2: Dolphin fragments.

	real	ER	ER+T	BA	BA+T	BTER
n	324					
m	347					
av(m)		346		320		
F_1	1324	0.8163	0.3542	-0.0141	0.0138	-0.0038
$C_3 = F_2$	17	0.9927	-0.006	1	-0.004	-0.1081
$P_3 = F_3$	2668	0.9565	0.6023	0.291	0.1758	-0.094
$S_{13} = F_4$	3875	0.9809	0.7858	-0.3761	-0.3	0.0695
$C_4 = F_5$	13	0.9986	0.7115	1	0.5051	-0.2249
$T_{31} = F_6$	235	0.9991	0.5303	1	-0.067	-0.1213
Diamond $= F_7$	19	1	0.7394	1	0.5197	-0.0501
$C_5 = F_8$	9	0.9991	0.9217	1	0.7991	-0.3277
$C_r = F_9$	1006	0.9999	0.859	1	-0.482	-0.1144
$T_{41} = F_{10}$	213	0.9998	0.8965	1	0.4299	-0.2755
Reindeer $= F_{11}$	890	0.9999	0.8235	1	0.1961	-0.1764
$T_{32} = F_{12}$	454	0.9999	0.7271	1	0.2416	-0.1996
Bowtie $= F_{13}$	6	1	0.7427	1	-0.093	-0.1208
House $= F_{14}$	16	1	0.9488	1	0.8081	-0.167
$C_6 = F_{15}$	7	0.9997	0.9601	1	0.9267	-0.4621

Table 3.3: ColoSpg fragments.

	real	**ER**	**ER+T**	**BA**	**BA+T**	**BTER**
n	118					
m	613					
av(m)		611		516		
F_1	12119	0.6991	-0.0541	0.0112	-0.057	0.0901
$C_3 = F_2$	1107	0.9715	0.0213	0.5929	0.2495	0.0688
$P_3 = F_3$	205833	0.889	-0.0665	0.1226	-0.053	0.1504
$S_{13} = F_4$	126075	0.8959	-0.1052	-0.3674	-0.417	0.1607
$C_4 = F_5$	14682	0.9874	0.0693	0.5905	0.2504	0.1546
$T_{31} = F_6$	80494	0.9904	-0.0184	0.3963	0.0092	0.1391
Diamond $= F_7$	13395	0.9991	0.1231	0.7815	0.3774	0.1096
$C_5 = F_8$	180042	0.9952	0.0757	0.6384	0.2424	0.2377
$C_r = F_9$	1414176	0.9971	-0.0752	-0.1252	-0.462	0.2206
$T_{41} = F_{10}$	1434083	0.9961	0.0447	0.4506	0.0515	0.2317
Reindeer $= F_{11}$	1766160	0.9971	-0.0212	0.3364	-0.071	0.2148
$T_{32} = F_{12}$	1242809	0.9969	-0.0195	0.5755	0.1054	0.203
Bowtie $= F_{13}$	135737	0.9997	0	0.6196	-0.011	0.1988
House $= F_{14}$	431360	0.9997	0.1662	0.8053	0.3823	0.2054
$C_6 = F_{15}$	2408189	0.9982	0.0692	0.707	0.264	0.3149

Table 3.4: Centrality fragments.

	real	**ER**	**ER+T**	**BA**	**BA+T**	**BTER**
n	229					
m	604					
av(m)		601		650		
F_1	5202	0.7397	0.1773	-0.1862	-0.2019	0.0616
$C_3 = F_2$	201	0.9717	-0.0413	0.23	-0.0292	-0.1104
$P_3 = F_3$	41208	0.9068	0.3078	-0.2205	-0.2628	-0.0079
$S_{13} = F_4$	25244	0.9485	0.5197	-0.5431	-0.5752	0.1497
$C_4 = F_5$	884	0.9878	0.4572	0.0726	-0.0421	-0.1776
$T_{31} = F_6$	6895	0.9933	0.2769	-0.1198	-0.3236	-0.0906
Diamond $= F_7$	623	0.9996	0.3912	0.3113	0.0254	-0.2171
$C_5 = F_8$	4048	0.9941	0.6235	-0.0761	-0.1857	-0.3346
$C_r = F_9$	57150	0.999	0.6356	-0.5774	-0.7067	-0.0229
$T_{41} = F_{10}$	40548	0.9973	0.6649	-0.2392	-0.3563	-0.2253
Reindeer $= F_{11}$	76139	0.9984	0.5315	-0.3158	-0.4604	-0.1613
$T_{32} = F_{12}$	51407	0.9977	0.3811	-0.0508	-0.2965	-0.199
Bowtie $= F_{13}$	2856	0.9999	0.4898	-0.043	-0.4055	-0.1636
House $= F_{14}$	6711	0.9998	0.714	0.1965	-0.0478	-0.3506
$C_6 = F_{15}$	20595	0.9976	0.675	-0.1594	-0.28	-0.4625

Table 3.5: Pin Malaria fragments.

	real	ER	ER+T	BA	BA+T	BTER
n	1586					
m	11540					
av(m)		12333		12328		
F_1	216461	0.6773	-0.3186	-0.2546	-0.3702	0.0013
$C_3 = F_2$	28002	0.9955	-0.2382	0.785	0.3573	0.005
$P_3 = F_3$	4296686	0.8688	-0.5812	-0.3756	-0.6038	-0.0463
$S_{13} = F_4$	1679811	0.887	-0.5937	-0.8437	-0.873	-0.0004
$C_4 = F_5$	186566	0.9963	-0.6406	0.4811	-0.1674	-0.2252
$T_{31} = F_6$	1690907	0.9984	-0.5362	0.2274	-0.4078	0.0034
Diamond $= F_7$	340878	1	-0.4957	0.8431	0.281	-0.1012
$C_5 = F_8$	1575866	0.9975	-0.8535	0.0908	-0.6233	-0.4054
$C_r = F_9$	20510120	0.9995	-0.7498	-0.7663	-0.9349	0.0054
$T_{41} = F_{10}$	15871108	0.9987	-0.8117	-0.2444	-0.7439	-0.2613
Reindeer $= F_{11}$	35442713	0.9994	-0.7467	-0.3258	-0.7405	-0.063
$T_{32} = F_{12}$	34382728	0.9994	-0.7303	0.226	-0.5642	-0.0456
Bowtie $= F_{13}$	3424433	1	-0.7007	0.4764	-0.6232	0.0401
House $= F_{14}$	6425782	1	-0.7809	0.6525	-0.2378	-0.3171
$C_6 = F_{15}$	16505015	0.9985	-0.941	-0.24	-0.8393	-0.5297

Table 3.6: Corporate People fragments.

	real	**ER**	**ER+T**	**BA**	**BA+T**	**BTER**
n	108					
m	707					
av(m)		704		626		
F_1	13602	0.7214	0.0588	0.1668	0.112	0.192
$C_3 = F_2$	116	0.9433	-0.047	0.4608	0.1302	0.0172
$P_3 = F_3$	247305	0.8912	0.0879	0.3025	0.1643	0.2546
$S_{13} = F_4$	103361	0.9299	0.2554	0.034	-0.005	0.4029
$C_4 = F_5$	29263	0.9716	0.047	0.4925	0.1833	0.0953
$T_{31} = F_6$	8034	0.9844	0.108	0.4193	0.1005	0.2152
Diamond $= F_7$	821	0.9972	0.0327	0.6056	0.1454	-0.0019
$C_5 = F_8$	32978	0.9885	0.0956	0.589	0.2439	0.171
$C_r = F_9$	116819	0.9972	0.3928	0.2663	-0.031	0.492
$T_{41} = F_{10}$	2435383	0.9925	0.2004	0.4923	0.1769	0.2676
Reindeer $= F_{11}$	172214	0.9955	0.2246	0.468	0.1513	0.3325
$T_{32} = F_{12}$	126925	0.9939	0.1043	0.5832	0.1962	0.2356
Bowtie $= F_{13}$	2579	0.9994	0.2274	0.6143	0.0942	0.303
House $= F_{14}$	44215	0.9987	0.1373	0.6798	0.2208	0.0868
$C_6 = F_{15}$	5777276	0.9954	0.1081	0.6845	0.3094	0.2258

Table 3.7: Canton fragments.

n	616					
m	2012					
av(m)		2018		1800		
F_1	29327	0.7383	-0.0298	-0.08	-0.377	0.0287
$C_3 = F_2$	3598	0.996	0.0602	0.881	-0.0888	0.0376
$P_3 = F_3$	376645	0.9299	-0.1886	0.0665	-0.6147	0.0487
$S_{13} = F_4$	246315	0.9362	-0.2008	-0.626	-0.7278	0.0581
$C_4 = F_5$	25502	0.9988	-0.371	0.8798	-0.4888	0.0613
$T_{31} = F_6$	181958	0.999	-0.3303	0.6914	-0.5382	0.0566
Diamond $= F_7$	40556	1	-0.4193	0.9707	-0.277	0.0653
$C_5 = F_8$	211825	0.9996	-0.6739	0.8975	-0.7392	0.0839
$C_r = F_9$	2331399	0.9998	-0.6038	-0.02	-0.8505	0.0784
$T_{41} = F_{10}$	1847005	0.9997	-0.6416	0.7401	-0.7525	0.077
Reindeer $= F_{11}$	3083556	0.9998	-0.5957	0.5457	-0.7495	0.0764
$T_{32} = F_{12}$	2431969	0.9998	-0.5766	0.8183	-0.71	0.0737
Bowtie $= F_{13}$	303482	1	-0.7186	0.9261	-0.7253	0.0946
House $= F_{14}$	791935	1	-0.7192	0.9764	-0.6126	0.0877
$C_6 = F_{15}$	1963883	0.9999	-0.8403	0.9206	-0.8713	0.1063

Table 3.8: Drugs fragments.

	real	ER	ER+T	BA	BA+T	BTER
n	249					
m	635					
av(m)		633		710		
F_1	4455	0.6992	0.0256	-0.3134	-0.3355	-0.0022
$C_3 = F_2$	327	0.9853	-0.098	0.4321	-0.0618	-0.0554
$P_3 = F_3$	31503	0.882	0.0491	-0.3961	-0.4965	-0.0683
$S_{13} = F_4$	13130	0.9042	0.0913	-0.7792	-0.7798	-0.0167
$C_4 = F_5$	1191	0.9916	0.1178	0.1997	-0.1946	-0.1048
$T_{31} = F_6$	7704	0.9951	-0.0201	-0.1093	-0.5142	-0.0909
Diamond $= F_7$	1106	0.9998	0.032	0.5461	-0.1319	-0.1799
$C_5 = F_8$	4504	0.9956	0.2301	-0.0426	-0.4424	-0.2339
$C_r = F_9$	34626	0.9985	0.0487	-0.7639	-0.8918	-0.1306
$T_{41} = F_{10}$	37519	0.9974	0.1774	-0.3167	-0.6195	-0.1917
Reindeer $= F_{11}$	60271	0.9984	0.0466	-0.4587	-0.7169	-0.1867
$T_{32} = F_{12}$	53207	0.9983	0.0055	-0.0709	-0.5843	-0.1795
Bowtie $= F_{13}$	3091	0.9999	-0.0547	-0.0299	-0.7259	-0.1973
House $= F_{14}$	9128	0.9999	0.2083	0.3431	-0.3553	-0.3086
$C_6 = F_{15}$	20369	0.998	0.3071	-0.1924	-0.6008	-0.3305

Table 3.9: GD fragments.

	real	ER	ER+T	BA	BA+T	BTER
n	181					
m	2311					
av(m)		2307		1917		
F_1	95534	0.7628	0.236	0.2325	0.2334	0.2421
$C_3 = F_2$	10489	0.9467	0.5568	0.4599	0.4504	0.1125
$P_3 = F_3$	3603217	0.9192	0.4123	0.3768	0.3761	0.3282
$S_{13} = F_4$	1708591	0.9425	0.5453	0.2114	0.2097	0.444
$C_4 = F_5$	421275	0.9881	0.7735	0.6813	0.6797	0.4246
$T_{31} = F_6$	1532094	0.9868	0.7415	0.4896	0.4811	0.3046
Diamond $= F_7$	368113	0.9982	0.9124	0.7053	0.6985	0.2954
$C_5 = F_8$	9541185	0.9951	0.7958	0.6981	0.6967	0.4166
$C_r = F_9$	44856088	0.9974	0.8792	0.3723	0.3557	0.5189
$T_{41} = F_{10}$	75582987	0.9969	0.8654	0.6868	0.6842	0.5348
Reindeer $= F_{11}$	70771628	0.9966	0.85	0.5608	0.5552	0.4413
$T_{32} = F_{12}$	56210239	0.9958	0.8149	0.6317	0.624	0.3729
Bowtie $= F_{13}$	6999260	0.9994	0.94	0.6616	0.6462	0.4118
House $= F_{14}$	28332235	0.9994	0.9436	0.7922	0.7882	0.446
$C_6 = F_{15}$	3.22E+08	0.9986	0.8685	0.7951	0.7942	0.5367

Table 3.10: Little Rock fragments.

	real	**ER**	**ER+T**	**BA**	**BA+T**	**BTER**
n	112					
m	830					
av(m)		825		655		
F_1	18544	0.7469	0.2081	0.2294	0.2337	0.1886
$C_3 = F_2$	124	0.3727	-0.6251	-0.6882	-0.6845	-0.824
$P_3 = F_3$	396443	0.9118	0.3846	0.3928	0.3982	0.2547
$S_{13} = F_4$	158231	0.926	0.459	0.0872	0.0961	0.2955
$C_4 = F_5$	55704	0.99	0.8103	0.7659	0.7696	0.5264
$T_{31} = F_6$	8825	0.7688	-0.4508	-0.7134	-0.7079	-0.8002
Diamond $= F_7$	865	0.9218	-0.2789	-0.7321	-0.7256	-0.8841
$C_5 = F_8$	50051	0.9406	-0.1445	-0.2418	-0.232	-0.6656
$C_r = F_9$	123556	0.9394	-0.1486	-0.818	-0.8129	-0.7475
$T_{41} = F_{10}$	5419604	0.9972	0.8838	0.7608	0.7665	0.585
Reindeer $= F_{11}$	202645	0.9264	-0.2441	-0.6883	-0.6803	-0.7802
$T_{32} = F_{12}$	169920	0.9128	-0.3258	-0.5882	-0.58	-0.7926
Bowtie $= F_{13}$	2335	0.9016	-0.6502	-0.9362	-0.9339	-0.9652
House $= F_{14}$	66644	0.9859	0.2039	-0.3768	-0.3633	-0.7629
$C_6 = F_{15}$	15489577	0.9989	0.9027	0.8887	0.8911	0.6156

Table 3.11: Stony fragments.

3.1.6 Conclusion

In this study, we used random models because random models are good for testing the properties of real-world networks but some existing models lack something that the real-world networks have. So, In this experiment, we added triangles to the random networks and we examined ten real networks by computing 15 fragments. Then we compute the abundance of fragments. In some cases the effect of adding triangles is marginal. But sometimes we find that many other fragments appear in much more realistic frequencies. For example, in Tables 3.2, 3.4, 3.7 and 3.9 ER+T does well. For almost Tables, BTER is the best but much more expensive.

As the existence of these motifs and anti-motifs most likely reflects some structural or functional properties in real world networks, adapting these random graph models to more closely reflect the structures of real world networks should prove useful for further applications of random graph modules to better understand real world networks. So, our conclusion is if we target triangles we affect many more other types of fragments. Sometimes we are very successful in bringing everything in line. The network has much more of the structure of a real world network by doing one simple change making them much more suitable for simulations that mimic real world examples. The reason why some networks are amenable to this process and others aren't is a topic for future research.

Chapter 4

Measuring Bipartivity

4.1 Introduction

Bipartite networks are two sets of nodes connected by links without links in each set. These "two-mode" networks where two sets of nodes are connected by links that represent the relationships between them are described as bipartite [26]. However, some networks, though they do not strictly meet this definition are not completely random and have some bipartitivity. These can arise when the agents within the network tend to collect in diverse groups [39].

There are a lot of real networks represented as bipartite. For example, a set of buyers and a set of sellers: there are links between buyers and sellers in general, but sometimes there are sellers who buy from other sellers. So, the network is no longer bipartite [25]. Also, citation networks in which a set of nodes represented the authors and the other set represented the papers. In general, bipartite graph structure can be found in many real-world networks outside of the biological fields. For example, actors and movies they appear in, scientists and papers they authored, P2P exchange networks, queries and URLs, users and items for recommendations, and analysing internet traffic [68]. Systems that naturally occur as bipartite networks include biochemical networks of chemical substances and chemical reactions and affiliation networks with organisations and individual actors in that organisation. Then there are networks that are nearly bipartite such as ones where there is a tendency for individuals to collect in diverse groups such as in human sexual

contacts and human romance or partnerships. In a webgraph, it has been suggested that in a graph of the world wide web, small bipartite subgraphs have a hub-authority bipartite structure within communities on the web [49].
Applications include the links between enzyme-reactions and metabolic pathways, gene-disease linkages or an ecological network. It should be noted that there has been focus on unipartite graphs but less attention to the unique insights bipartite graphs can bring to biological sciences.

The existence of bipartite graphs has given rise to various algebraic and spectral applications being modified to analyse them since many of the methods that exist to analyse unipartite graphs are not suitable to the bipartite structure [51].

Disassortative mixing or anti-community occurs where vertices have most of their connections outside the group and have few to none connections inside the same group [13]. In situations where networks consist of two such anti-communities, detecting the largest bipartite subgraph within a given graph is necessary. These bipartite graphs are also known as two-mode networks [39].

Before discussing bipartivity measures we give an example to show why it might be useful in practice to measure closeness to bipartivity taken from [23] where it was observed that one can find a degree of bipartivity in a network depicting communication between airports; almost all communication is between the partitions while there is little between members of the same group. The authors focused on European airline networks to see how varying degrees of bipartivity affect their efficiency.

Thirty-three airlines were studied in total with 25 being "traditional" airlines and 8 being "low-cost". The airports represented the nodes and the flights between them are edges. The authors showed that that traditional airlines appear to be much more bipartite than low cost airlines. They have observed large differences between traditional carriers and the low-cost airlines, being the bipartivity of the latter ones much smaller than that of the former. Then, they have shown that alliances and major mergers of traditional airlines lead to a decrease in bipartivity.

To show how bipartivity can affect efficiency, we can use an example of a passenger from City A who is visiting two cities: City B and City C. After her trip she flies back from City C to City B to go back to her home City A. If her airline belongs to an alliance of airlines then she may be able to go directly home to City A from City C which amounts to considerable savings in time and money for both the passenger and airline [23]. Estrada et al. (2016) measure transportation efficiency as the ratio of the number of passengers to the number of hours flown by a carrier. Increasing efficiency means decreasing the degree of bipartivity and is shown when the number of passengers (in millions) is divided by the hours flown and plotted against the bipartivity of air transportation networks. Thus, we can see that a low degree of bipartivity in a transportation network drives airline efficiency. Without a way to measure bipartivity, inefficiencies like the one just discussed would be more difficult to identify [26].

This chapter aims to give a thorough review of characterisations of bipartivity. We establish their equivalence for bipartite networks but when we look at "nearly" bipartite networks we will see that the level by which that the characterisations fail can vary wildly.

We discuss in detail the appearance of bipartivity and near-bipartivity in real world networks and more generally in graph theory to highlight the importance of the subject.

The main contribution of this chapter is to use the characterisations of bipartivity to derive and analyse a number of measures to determine how far a network is from a bipartite network. Some of these measures have appeared previously for example b_e in equation 4.2, and some of them are new for example the measure b_{L_N} but we believe that our comparison of them is new and adds valuable perspective. We show that the different characterisations can lead to profoundly different conclusions about the level of bipartivity within a network. For example we show that is possible to generate a network that is arbitrarily close to being bipartite by one measure while being arbitrarily far by another.

We support our analysis with comprehensive numerical tests both on real world and artificial networks.

The theoretical results in Section 4.2.2 are new, and along with the comprehensive characterisations of bipartivity, they have been published in [3].

4.2 Characterising Bipartivity

Suppose G is a simple network with adjacency matrix A and spectrum

$$\sigma(G) = \{\lambda_1, \lambda_2, \dots, \lambda_n\},$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ (we will make this assumption throughout this chapter). The classic definition of network bipartivity is as follows.

Definition 4.2.1. A network is bipartite if the nodes of the network $G(V, E)$ can be divided into disjoint sets $V = V_1 \cup V_2$ such that for all $(u, v) \in E$, either $u \in V_1$ and $v \in V_2$ or $u \in V_2$ and $v \in V_1$.

We note that the network with n nodes that is the least bipartite is K_n, the complete graph. There are a number of other characterisations that have been proposed in the literature as encapsulated in Theorem 4, below. First, we introduce a useful Lemma

Lemma 2. A simple graph has cycles of odd length if and only if it has closed walks of odd length.

Proof. $\Rightarrow$ Trivial. A cycle is a walk.

$\Leftarrow$ Suppose we have a closed walk of odd length $p \geq 3$, from u to u, say. If the walk is a cycle then we have nothing to prove. Otherwise there are nodes on the walk which we visit more than once. Suppose one of these is v and the walk goes

$$u \to \cdots \to v \to \cdots \to v \to \cdots \to u.$$

Either the section of this walk from v to v, or the walk that skips this section, is a closed walk of odd length $< p$. If this closed walk is a cycle then we are done, otherwise we repeat the process on the new odd length closed walk and one of its repeated notes. This finite process will end when our odd length closed walk no longer has repeated nodes and is therefore a cycle. $\square$

The following is a by no means exhaustive list of equivalent characterisations of bipartivity. All of these conditions are well known but we have collected them together for the first time.

Theorem 4. If $G(V, E)$ is a simple connected network with adjacency matrix A then the following conditions are equivalent.

1. G is bipartite.
2. A of G can be permuted to $\begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$.
3. G has no cycles of odd length.
4. G has no closed walks of odd length.
5. $\mathrm{tr}(\sinh A) = 0$.
6. The modularity of the bipartition is $-1/2$.
7. The spectrum of G is symmetric about 0.
8. $\lambda_1 = -\lambda_n$.
9. The signless Laplacian is singular.
10. The normalised Laplacian has maximum eigenvalue 2.
11. The subgraph centralisation of G equals $\dfrac{1}{n}\displaystyle\sum_{j=1}^{n}\cosh(\lambda_j)$.
12. The spectra of the signless Laplacian ($L_s = A + D$) and the graph Laplacian $L = D - A$ coincide for a bipartite graph.

The results 2,3,4,7,8 and 12 are standard text book results and 5,6,9,10 and 11 can be found in [9, 19, 23, 48].

Proof. $(1 \Rightarrow 2)$ The vertices of G can be split into two disjoint sets $V = V_1 \cup V_2$ such that for any edge in G once incident node is in V_1 and one in V_2. Suppose $|V| = n$, $|V_1| = n_1$, $|V_2| = n_2$ then order the rows of the adjacency matrix so the first n_1 rows correspond to nodes in V_1 and the final n_2 to those in V_2. By definition of bipartivity, $A_{ij} = 0$ if $i, j \in \{1, ..., n_1\}$ or if $i, j \in \{n_1+1, ..., n\}$, so A has the desired structure $A = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$.

$(2 \Rightarrow 1)$ Use the permuted form of the adjacency matrix to partition the nodes into disjoint sets V_1 and V_2. Since $A_{uv} = 0$ for any two nodes in the same partition, there are no inter-partition edges and G is bipartite.

$(1 \Rightarrow 3)$ Suppose G is bipartite and it has an odd cycle $(v_1, v_2, ..., v_{2k+1}, v_1)$. Suppose $v_1 \in V_1$, then by definition, $v_2 \in V_2$ and $v_3 \in V_1$, so $v_{2j} \in V_2$ and $v_{2j+1} \in V_1$. But v_{2k+1} is connected to v_1 so $v_{2k+1} \notin V_1$ a contradiction.

$(3 \Leftrightarrow 4)$ From Lemma 2.

$(4 \Rightarrow 1)$ Choose $x \in V$ and partition nodes into $\{X, Y\}$ where $v \in X \Leftrightarrow d(v, x)$ even (where $d(u, v)$ represents shortest path distance between u and v). Clearly $X \cap Y = \emptyset, X \cup Y = V$. Suppose there exists $v_1, v_2 \in X$, v_1 is adjacent to v_2 (or two such nodes in Y). Then consider the following walk: $x \to \cdots \to v_1 \to v_2 \to \cdots \to x$ where we follow the shortest path from x to v_1 and from v_2 to x. This is a closed walk of odd length, which is a contradiction. We can conclude that if v_1 is adjacent to v_2, $v_1 \in X$ and $v_2 \in Y$ or vice versa and G is bipartite.

$(4 \Leftrightarrow 5)$ Note that all elements of all powers of A are nonnegative. Hence G has no closed walks of odd length $\Leftrightarrow \forall i, k, (A^{2k+1})_{ii} = 0 \Leftrightarrow \forall k, \mathrm{tr}(A^{2k+1}) = 0 \Leftrightarrow \mathrm{tr}(\sinh(A)) = 0$.

$(1 \Leftrightarrow 6)$

For a partition into two sets we can use the formula for modularity

$$Q = \frac{1}{4m}\mathbf{s}^T(A - \frac{kk^T}{2m})\mathbf{s}$$

where $m =\mid E \mid$, A is the adjacency matrix of G, $\mathbf{k}$ is the vector of degrees, k_i is the degree of node i and $\mathbf{s}$ is an indicator vector with $s_i = 1$ if $i \in V_1$, $s_i = -1$ if $i \in V_2$. We assume without lose of generality that $V_1 = \{1, \ldots, n_1\}, V_2 = \{n_1 + 1, \ldots, n\}$. We make use of the following quantities

$$k_1 = \sum_{i \in V_1} k_i, \quad k_2 = \sum_{i \in V_2} k_i, \quad k_{11} = \sum_{i,j \in V_1} a_{ij}, \quad k_{22} = \sum_{i,j \in V_2} a_{ij}, \quad k_{12} = \sum_{i \in V_1, j \in V_2} a_{ij}.$$

Note that

$$k_{12} = m - \frac{(k_{11} + k_{22})}{2}, k_1 = k_{11} + k_{12} = m + \frac{(k_{11} - k_{22})}{2}, k_2 = 2m - k_1 = m + \frac{(k_{22} - k_{11})}{2}$$

Partition $\mathbf{s}$ and A according to V_1 and V_2 and let $\mathbf{e^{(1)}}$ and $\mathbf{e^{(2)}}$ be vectors of $\mathbf{1}_s$ of lengths n_1 and n_2. Then

$$\mathbf{s}^T A \mathbf{s} = \begin{bmatrix} \mathbf{e^{(1)T}} & -\mathbf{e^{(2)T}} \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \mathbf{e^{(1)}} \\ -\mathbf{e^{(2)}} \end{bmatrix} = \mathbf{e^{(1)}}^T A_{11} \mathbf{e^{(1)}} + \mathbf{e^{(1)}}^T A_{22} \mathbf{e^{(2)}} - 2\mathbf{e^{(1)}}^T A_{12} \mathbf{e^{(2)}}$$

So

$$\mathbf{s}^T A \mathbf{s} = k_{11} + k_{22} - 2k_{12} = 2(k_{11} + k_{22} - m).$$

And

$$\mathbf{s}^T \mathbf{k} = \mathbf{e^{(1)}}^T k_1 - \mathbf{e^{(2)}}^T k_2 = k_{11} - k_{22}.$$

Hence

$$Q = \frac{1}{4m}(2(k_{11} + k_{22} - m) - \frac{(k_{11} - k_{22})^2}{2m}) = \frac{-1}{2} + \frac{k_{11} + k_{22}}{2m} - \frac{(k_{11} - k_{22})^2}{8m^2}.$$

Note that if $k_{11} = k_{22} = 0$ then $Q = \frac{-1}{2}$, the desired equality for the case of an exact bipartite. But for a general bound we note that $2m \geq k_{11} + k_{22}$ hence

$$Q \geq \frac{-1}{2} + \frac{1}{8m^2}(2(k_{11} + k_{22})^2 - (k_{11} - k_{22})^2)$$

$$= \frac{-1}{2} + \frac{1}{8m^2}((k_{11} + k_{22})^2 + 4k_{11}k_{22})$$

$$\geq \frac{-1}{2}$$

with equality at the least step if and only if $k_{11} = k_{22} = 0$

$(1 \Rightarrow 7)$

Suppose that G is bipartite and λ is a non-zero eigenvalue of the adjacency matrix A, then there is a vector

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$$

such that

$$\begin{bmatrix} O & B \\ B^T & O \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} B\mathbf{v}_2 \\ B^T\mathbf{v}_1 \end{bmatrix} = \begin{bmatrix} \lambda\mathbf{v}_1 \\ \lambda\mathbf{v}_2 \end{bmatrix}.$$

Thus $B\mathbf{v}_2 = \lambda\mathbf{v}_1$ and $B^T\mathbf{v}_1 = \lambda\mathbf{v}_2$. Note also that as $\mathbf{v} \neq 0$ and $\lambda \neq 0 \Rightarrow \mathbf{v}_1 \neq 0$ and $\mathbf{v}_2 \neq 0$. So we have

$$\begin{bmatrix} O & B \\ B^T & O \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ -\mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} -B\mathbf{v}_2 \\ B^T\mathbf{v}_1 \end{bmatrix} = \begin{bmatrix} -\lambda\mathbf{v}_1 \\ \lambda\mathbf{v}_2 \end{bmatrix} = -\lambda \begin{bmatrix} \mathbf{v}_1 \\ -\mathbf{v}_2 \end{bmatrix}$$

and $-\lambda$ is also an eigenvalue of the adjacency matrix which confirms that the spectrum of G is symmetric about zero.

$(7 \Rightarrow 8)$ If λ is an eigenvalue of the adjacency matrix then $-\lambda$ is also an eigenvalue. So, if λ_1 is the most positive eigenvalue and λ_n is the most negative eigenvalue then $\lambda_1 = -\lambda_n$.

$(8 \Rightarrow 1)$

Suppose that $\lambda_1 = -\lambda_n$. Let $\mathbf{x}$ be the eigenvector of λ_n and $\mathbf{y}$ be the vector $\mathbf{y} = |\mathbf{x}|$. Then

$$|\lambda_n| = |\frac{\mathbf{x}^T A\mathbf{x}}{\mathbf{x}^T\mathbf{x}}| = \left|\Sigma_{u,v}\frac{A(u,v)\mathbf{x}(u)\mathbf{x}(v)}{\mathbf{x^T x}}\right| \leq \Sigma_{u,v}\frac{A(u,v)|\mathbf{x}(u)||\mathbf{x}(v)|}{|\mathbf{x^T x}|} \leq \lambda_1\frac{\mathbf{y^T y}}{\mathbf{y^T y}} = \lambda_1$$

For this to be an equality, it must be the case that all the terms $\mathbf{x}(u)\mathbf{x}(v) \geq 0$ whenever $A(u,v) = 1$. By the Perron-Frobenius theorem, $\mathbf{x}$ must have both positive and negative elements (since it is orthogonal to the Perron vector which is one-signed). Hence if $\mathbf{x}(u)$ and $\mathbf{x}(v)$ are of different signs, $A(u,v) = 0$. Thus splitting nodes according to the signs of $\mathbf{x}$ gives a perfect bipartition of the graph.

$(1 \Leftrightarrow 9)$

First note that $Q = A + D = RR^T$ where R is the vertex-edge incidence matrix of graph G. Then for nonzero $\mathbf{x}$, $Q\mathbf{x} = \mathbf{0} \Leftrightarrow R^T\mathbf{x} = 0$ which can happen if and only if for every edge (u, v) in the graph, $x_u = -x_v$ which can if and only if G is bipartite and u and v are in opposite partitions [18].

$(1 \Leftrightarrow 10)$ For a bipartite graph the eigenvalues of the normalised Laplacian $L = I - D^{-1/2}AD^{-1/2}$ can be deduced from $\hat{L} = D^{-1/2}AD^{-1/2}$ and clearly $2 \in \sigma(L)$ if and only if $-1 \in \sigma(\hat{L})$. Now $\hat{L} \geq 0$ and $\mathbf{x} = D^{\frac{1}{2}}\mathbf{e}$ is clearly the Perron vector of $\hat{L}$ with eigenvalue 1. We can now follow the proof of $(1 \Leftrightarrow 8)$.

$(1 \Rightarrow 11)$ The subgraph centralization is defined as $SC(G) = \frac{1}{n}tr(e^A)$. We can use the identity

$$SC(G) = \frac{1}{n}\sum_{l=0}^{\infty}\frac{\mu_l}{l!} = \frac{1}{n}\sum_{j=1}^{n}(e^{\lambda_j}).$$

Now $SC(G)$ can be expressed as the sum of two contributions, one coming from odd and the other from even closed walks, that is,

$$SC(G) = \frac{1}{n}\sum_{j=1}^{n}[\cosh(\lambda_j) + \sinh(\lambda_j)] = SC(G)_{\text{even}} + SC(G)_{\text{odd}}.$$

If G is bipartite, then by point 5, $SC(G)_{odd} = \frac{1}{n}\sum_{j=1}^{n}\sinh(\lambda_j) = 0$, therefore

$$SC(G) = SC(G)_{even} = \frac{1}{n}\sum_{j=1}^{n}\cosh(\lambda_j).$$

$(11 \Rightarrow 1)$

$$\frac{1}{n}\text{tr}(e^A) = \frac{1}{n}\text{tr}(\cosh(A)) + \frac{1}{n}\text{tr}(\sinh(A)).$$

We know that $\frac{1}{n}\text{tr}(\sinh(A)) \geq 0$ and by 5 $\text{tr}(\sinh(A)) = 0 \Rightarrow G$ bipartite.

$(12 \Leftrightarrow 1)$

Let $L = D - A = \begin{bmatrix} D_1 & -B \\ -B^T & D_2 \end{bmatrix}$, $Q = \begin{bmatrix} I & O \\ O & -I \end{bmatrix} = Q^{-1}$

$$Q^{-1}LQ = \begin{bmatrix} I & O \\ O & -I \end{bmatrix} \begin{bmatrix} D_1 & -B \\ -B^T & D_2 \end{bmatrix} \begin{bmatrix} I & O \\ O & -I \end{bmatrix} = \begin{bmatrix} D_1 & -B \\ -B^T & -D_2 \end{bmatrix} \begin{bmatrix} I & O \\ O & -I \end{bmatrix} = \begin{bmatrix} D_1 & B \\ B^T & D_2 \end{bmatrix} = L_s$$

L and S are similar So they have the same eigenvalues.

□

4.2.1 Bipartivity Measures

Using Theorem 4 we can characterise closeness to bipartivity in myriad ways. In this section we introduce a number of measures. We want to identify computationally tractable measures. These are generally inspired by algebraic properties of the adjacency matrix and Laplacian of a bipartite graph.

Calculating the proportion of links that destroy the bipartivity in a network gives an easy way to define the degree of network bipartivity. That is, we can measure the degree of bipartivity by counting the minimum number of edges we need to remove to create a perfectly bipartite network [25]. We will use the term “frustrated edges” to describe the edges we want to remove. This term was introduced in [39].

Let m_d be the number of such links in a network G and m be the total number of edges. Then the bipartivity of G can be measured by

$$b_c = b_c(G) = 1 - \frac{m_d}{m}$$

where the subindex c is introduced to indicate that this index is combinatorial [25]. Clearly $b_c = 1$ if and only if G is a bipartite.

If $G = K_n$ then we can compute b_c explicitly since the nearest bipartite graph is the complete bipartite graph $K_{\frac{n}{2},\frac{n}{2}}$ (n even, with $n^2/4$ edges) or $K_{\frac{n-1}{2},\frac{n+1}{2}}$ (n odd, with $(n^2-1)/4$ edges).

If n is even this means $m_d = n(n-2)/4$ and if n is odd $m_d = (n-1)^2/4$. In either case, $\lim_{n\to\infty} b_c = 1/2$. For comparison with other measures, we will redefine the measure so that

$$b_c = 1 - \frac{2m_d}{m},$$

and hence for all graphs $0 \leq b_c \leq 1$ with the upper and lower bounds being attainable in the limit.

Unfortunately, this measure is extremely expensive to calculate in practice because computing m_d is NP-complete, meaning that an efficient algorithm providing a solution has yet to be found [25], but since it is such a natural idea we will be interested in how our other measures compare against it. To that end, we provide some theoretical results for certain model networks. And for relatively small networks, where its computation is practical or where we can get decent bounds, we can give empirical comparisons.

A measure similar to b_c was introduced in [39] which tries to quantify bipartivity in terms of the energy in an underlying Ising model. This measure is, again, computationally impractical so the authors present another measure which is computable in polynomial time based on counting odd cycles. The authors note that even though m_d may be small there may be a large number of odd cycles. This is a point we will examine in detail later in this section.

For the measure, which we call b_2, Holme et al. draw a parallel between frustrated edges and edges that appear frequently in cycles of odd length and then attempt to get reasonable estimates of the minimum number of edges one can mark so that every odd cycle has a marked edge. To keep computational costs down, only cycles of relatively short length are considered. The measure can usually be calculated in $O(m^2)$ time and lies in the range $[1/2, 1]$ with $b_2 = 1$ if and only if the graph is bipartite. However the link between values of b_2 close to 1 and closeness to bipartivity are only established heuristically and empirically.

In detail, the measure is calculated for a graph G as follows. A value $\hat{n}$ is chosen to be the minimum value such that G has at least $3m$ odd cycles of length less than or equal to $\hat{n}$ (the choice $3m$ appears to be chosen for expediency). In many cases $\hat{n} = 3$.

Letting C be the set of all odd cycles of maximum length $\hat{n}$ the quantity $\nu(e)$ denotes the number of cycles in C passing through the edge e. It is assumed that if $\nu(e)$ is large then e is likely to be frustrated. The algorithm works as follows.

1. Start with C,
2. Sort the edges in order of ν.
3. Repeat the following while $C \neq \emptyset$:
    - Mark the edge e with highest ν.
    - Remove all cycles in C containing e.
    - Recalculate ν for each edge.

The number of iterations needed for this process is taken as the proxy for m_d

Rather than using odd cycles, we can derive alternative measures of closeness to bipartivity by quantifying the frequency of closed walks in comparison to the number of even walks or as a proportion of the total number of closed walks. This idea is introduced by Estrada in [26].

This can be quantified using subgraph centralisation with the measure

$$b_s = \frac{SC(G)_{even}}{SC(G)} = \frac{SC(G)_{even}}{SC(G)_{even} + SC(G)_{odd}} = \frac{\sum_{j=1}^{n} \cosh(\lambda_j)}{\sum_{j=1}^{n} e^{\lambda_j}} \tag{4.1}$$

where $SC(G)_{\text{even}} = \frac{1}{n}\sum_{j=1}^{n}\cosh(\lambda_j)$ and $SC(G)_{\text{odd}} = \frac{1}{n}\sum_{j=1}^{n}\sinh(\lambda_j)$. Note that $b_s \leq 1$ and $b_s = 1$ if and only if G is bipartite. Furthermore, as $\sinh(\lambda_j) \leq \cosh(\lambda_j)$, $\forall \lambda_j$, then $b_s \geq 1/2$ [26].

The lower bound is reached for K_n as $n \to \infty$. One way of confirming this is through spectral information. Since

$$\sigma(K_n) = \left\{[n-1]^1, [-1]^{n-1}\right\}$$

we have

$$b_s = \frac{\cosh(n-1) + (n-1)\cosh(-1)}{e^{n-1} + (n-1)e^{-1}}$$

and

$$\lim_{n\to\infty} b_s(K_n) = \lim_{n\to\infty} \frac{\cosh(n-1) + (n-1)\cosh(-1)}{e^{n-1} + (n-1)e^{-1}} = \lim_{n\to\infty} \frac{\cosh(n-1)}{e^{n-1}} = \frac{1}{2}.$$

A slight adaptation of (4.1) gives a measure of bipartivity in the range $[0, 1]$ as observed in [23]. Here we compare the number of even and odd walks using the measure

$$b_e = \frac{\mathrm{tr}(\cosh A) - \mathrm{tr}(\sinh A)}{\mathrm{tr}(\cosh A) + \mathrm{tr}(\sinh A)}.$$

Since A is diagonalisable this can be rewritten as

$$b_e = \frac{\sum_{j=1}^{n} e^{-\lambda_j}}{\sum_{j=1}^{n} e^{\lambda_j}} = \frac{\mathrm{tr}(e^{-A})}{\mathrm{tr}(e^{A})}. \tag{4.2}$$

From the fact that the spectrum is symmetric if and only if the network is bipartite, the upper bound of 1 is reached only for bipartite networks. For the complete graph K_n,

$$b_e = \frac{e^{-(n-1)} + (n-1)e}{e^{(n-1)} + (n-1)e^{-1}}$$

and clearly

$$\lim_{n\to\infty} b_e = 0.$$

We can introduce a raft of additional measures using the conditions outlined in Theorem 4. We won't give an exhaustive list, but in order to compare measures we introduce a selection below. We normalise our measures so that they equal 1 if and only if a network is bipartite and so they have an attainable minimum of 0 (possibly for an infinite graph, typically K_n as $n \to \infty$).

Our next measure exploits the observation that if the network is bipartite then the spectrum is symmetric about 0 and so

$$S = |\lambda_1 + \lambda_n| + |\lambda_2 + \lambda_{n-1}| + \cdots + |\lambda_1 + \lambda_n| = 0.$$

On the other hand, for K_n

$$\begin{aligned} S &= \mid \lambda_1 + \lambda_n \mid + \mid \lambda_2 + \lambda_{n-1} \mid + \ldots |\lambda_1 + \lambda_n| \\ &= \mid n-1-1 \mid + \mid -1-1 \mid + \mid -1-1 \mid + \cdots + \mid n-1-1 \mid \\ &= 2(n-2) + 2(n-2) = 4(n-2) \end{aligned}$$

for which $\lim_{n\to\infty} S = \infty$. A measure of bipartivity is given by

$$b_S = \frac{1}{S+1}.$$

If G is bipartite, then $b_S = 1/(0+1)$ while we see that for K_n, $b_S \to 0$ as $n \to \infty$.

We can also derive a measure based on modularity. Recall that $Q = -1/2$ if and only if the graph is bipartite. In this case the least bipartite graph is one where we have two complete graphs K_n connected by a single edge. To compute Q for such a graph note that the total number of edges is $n(n-1)+1$, that the degree of $2n-2$ nodes is $n-1$ and the degree of 2 nodes is n. Then

$$Q = \sum_{k=1}^{n_C} \left[\frac{|E_k|}{m} - \frac{1}{4m^2} (\sum_{j\in V_k} k_j)^2 \right]$$

where we divide the nodes in a network into n_C clusters $V_1, V_2, \ldots, V_{n_C}$.
$\mid E_{C_1} \mid = \frac{1}{2}n(n-1), \mid E_{C_2} \mid = \frac{1}{2}n(n-1), \sum_{j\in V_{c_1}}(k_j) = n(n-1)+1, \sum_{j\in V_{c_2}}(k_j) = n(n-1)+1$

$$Q = \frac{\frac{1}{2}n(n-1)}{n(n-1)+1} - \frac{(n(n-1)+1)^2}{4(n(n-1)+1)^2} + \frac{\frac{1}{2}n(n-1)}{n(n-1)+1} - \frac{(n(n-1)+1)^2}{4(n(n-1)+1)^2}$$

$$\Rightarrow Q = \frac{n(n-1)}{n(n-1)+1} - \frac{1}{2} \Rightarrow \lim_{n\to\infty} \frac{n(n-1)}{n(n-1)+1} - \frac{1}{2} = \frac{1}{1+0} - \frac{1}{2} = \frac{1}{2}.$$

So to get a measure in the range $[0,1]$ we introduce

$$b_Q = \frac{1}{2} - Q.$$

Our next measure is the simple ratio

$$b_\lambda = \frac{|\lambda_n|}{\lambda_1},$$

Where λ_1 and λ_n are the most positive and negative values of the adjacency matrix respectively. This has been used in many places in the literature, for example [52].

From Theorem 4, the network is bipartite if and only if $b_\lambda \leq 1$ with $b_\lambda = 1$ while for the complete graph $b_\lambda = 1/(n-1)$ and $b_\lambda \to 0$ as $n \to \infty$

For example, for the bipartite graph C_{2n} the eigenvalues are

$$\{2\cos[\frac{\pi j}{n}], j = 1, 2, ..., 2n\}.$$

We know that $-1 \leq \cos[\frac{\pi j}{n}] \leq 1 \Rightarrow -2 \leq \cos[\frac{\pi j}{n}] \leq 2$

$$b_\lambda = \frac{|\lambda_n|}{\lambda_1} = \frac{2}{2} = 1.$$

Closeness to bipartivity can be measured by the size of the largest eigenvalue of the normalised Laplacian. We do this with the measure

$$b_{L_N} = |1 - \lambda_{\max}(L_n)|.$$

Again, we know from Theorem 4 that b_{L_N} can only equal 1 for bipartite networks. For the complete graph K_n we can compute the spectrum of N explicitly since

$$L_N = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = I - \frac{1}{n-1}A = \frac{n}{n-1}I - \frac{1}{n-1}E$$

and hence its eigenvalues are 0 and $n/(n-1)$ meaning

$$b_{L_N} = |1 - \lambda_{\max}(L_N)| = |1 - \frac{n}{n-1}| = \frac{1}{n-1}$$

and

$$\lim_{n\to\infty} \frac{1}{n-1} = 0.$$

Since $\operatorname{tr}(\mathcal{L}) = n$ and $0 \in \sigma(\mathcal{L})$ for all graph, $\frac{n}{n-1}$ is the minimum possible spectral radius for the normalised Laplacian of any graph and hence $b_{L_N} \geq 0$ for all graphs.

We can use the signless Laplacian to measure closeness to bipartivity by

$$b_{L_s} = \frac{1}{1 + \lambda_{\min}(L_s)}$$

recalling that a graph is bipartite if and only if L_s is singular.

For the complete graph K_n, $L_s = D + A = (n-2)I + E$ and hence its eigenvalues are $n-2$ and $2n-2$ hence

$$\lim_{n\to\infty} b_{L_s} = \lim_{n\to\infty} \frac{1}{1+n-2} = 0.$$

We collected all Bipartivity Measures in Table 4.1. Note that measures 4,5,7 and 8 are new.

	Bipartivity Measures
1	$b_c = 1 - \frac{m_d}{m}$
2	$b_s = \frac{\sum_{j=1}^{n} \cosh(\lambda_j)}{\sum_{j=1}^{n} e^{\lambda_j}}$
3	$b_e = \frac{\sum_{j=1}^{n} e^{-\lambda_j}}{\sum_{j=1}^{n} e^{\lambda_j}} = \frac{\operatorname{tr}(e^{-A})}{\operatorname{tr}(e^{A})}$
4	$b_S = \frac{1}{S+1}$
5	$b_Q = \frac{1}{2} - Q$
6	$b_\lambda = \frac{\lvert\lambda_n\rvert}{\lambda_1}$
7	$b_{L_N} = \lvert 1 - \lambda_{\max}(L_n)\rvert$
8	$b_{L_s} = \frac{1}{1+\lambda_{\min}(L_s)}$

Table 4.1: Bipartivity Measures.

4.2.2 Comparison of Bipartivity Measures

In this section we give various examples showing the disparity between bipartivity measures on a number of families of networks.

4.2.3 Example 1

Our first example uses a "nearly" bipartite family of matrices suggested by Holmes et al. in [39] illustrated in Figure 4.1. The network (agave graph) is generated by adding a single edge to the complete bipartite graph K_{n2} (the edge is added to the partition with two nodes).

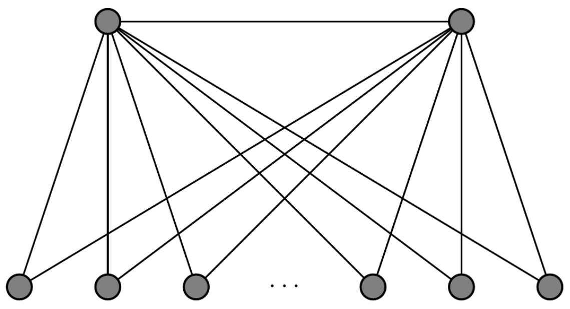


Figure 4.1: A "nearly" bipartite network.

We label this graph G_n. It has adjacency matrix

$$A = \begin{bmatrix} O & E_{n2} \\ E_{n2}^T & P_1 \end{bmatrix}$$

where E_{n2} is an $n \times 2$ matrix of 1s and

$$P_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

While G_n has only one frustrated edge, it also contains many odd cycles and this leads to a significant difference between certain bipartivity measures. For example, consider the measures $b_\lambda = -\frac{\lambda_n}{\lambda_1}$ and $b_e = \frac{\mathrm{tr}(e^{-A})}{\mathrm{tr}(e^{A})}$.

It is straightforward to compute the spectrum of A since it has rank 3 and the nonzero eigenvalues are easy to track down. Note that

$$A\begin{bmatrix}\mathbf{0}\\1\\-1\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\-1\\1\end{bmatrix}=-1\begin{bmatrix}\mathbf{0}\\1\\-1\end{bmatrix}$$

so $-1\in\sigma(A)$. Now let $\mathbf{e}$ be a vector of 1s and if $\mathbf{x}=\begin{bmatrix}\mathbf{e}^T & \alpha & \alpha\end{bmatrix}^T$ then

$$A\mathbf{x}=\begin{bmatrix}2\alpha\mathbf{e}\\n+\alpha\\n+\alpha\end{bmatrix}=\lambda\mathbf{x}$$

and hence $\lambda=2\alpha\Rightarrow(n+\alpha)=\lambda\alpha=2\alpha^2\Rightarrow 2\alpha^2-n-\alpha=0$ which is true if and only if $\alpha=(1\pm\sqrt{1+8n})/4$. Hence

$$\sigma(A)=\left\{[0]^{n-1},[-1]^1,\left[\frac{1}{2}+\frac{\sqrt{8n+1}}{2}\right]^1,\left[\frac{1}{2}-\frac{\sqrt{8n+1}}{2}\right]^1\right\}.$$

Having computed the spectrum we can now compare the bipartivity measures, in particular their limits as n grows. Firstly,

$$b_\lambda=(\sqrt{8n+1}-1)/(\sqrt{8n+1}+1)$$

and $b_\lambda\to 1$ as $n\to\infty$. Secondly,

$$b_e=\frac{n-1+e+e^{-\frac{1}{2}-\frac{\sqrt{8n+1}}{2}}+e^{-\frac{1}{2}+\frac{\sqrt{8n+1}}{2}}}{n-1+e^{-1}+e^{\frac{1}{2}+\frac{\sqrt{8n+1}}{2}}++e^{\frac{1}{2}-\frac{\sqrt{8n+1}}{2}}}=\frac{2e^{-\frac{1}{2}}\cosh\sqrt{8n+1}+e+n-1}{2e^{\frac{1}{2}}\cosh\sqrt{8n+1}+e^{-1}+n-1},$$

hence $b_e \to 2e^{-\frac{1}{2}}/(2e^{\frac{1}{2}}) = e^{-1}$ as $n \to \infty$.

Finally, we compute b_c. The total number of edges $m = 4n + 2$ and the number of edges we want to delete to be bipartite $m_d = 1$. So,

$$b_c = 1 - \frac{m_d}{m} = 1 - \frac{1}{4n + 2}$$

as $n \to \infty \Rightarrow b_c \to 1$.

So, $b_\lambda \to 1, b_c \to 1$ and $b_e \to e^{-1}$.

Next we examine measures based on the signless graph Laplacian matrix. If the adjacency matrix is

$$A = \begin{bmatrix} O & E_{n2} \\ E_{n2}^T & P_1 \end{bmatrix},$$

then the signless Laplacian is

$$L_s = D + A = \begin{bmatrix} 2I & E_{n2} \\ E_{n2}^T & \begin{bmatrix} n+1 & 1 \\ 1 & n+1 \end{bmatrix} \end{bmatrix}.$$

To find the eigenvalues of L_s, we compute $L_s - 2I$ as

$$L_s - 2I = \begin{bmatrix} O & E_{n2} \\ E_{n2}^T & \begin{bmatrix} n-1 & 1 \\ 1 & n-1 \end{bmatrix} \end{bmatrix}.$$

This has a rank of 3 and therefore it has a zero eigenvalue of multiplicity $n-1$. To compute the other eigenvalues we note that $\mathbf{x} = \begin{bmatrix} 0 & \dots & 0 & 1 & -1 \end{bmatrix}^T$ is an eigenvector with eigenvalue $n-2$ since

$$(L_s - 2I)\mathbf{x} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ n-2 \\ 2-n \end{bmatrix} = (n-2) \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ -1 \end{bmatrix} = \lambda \mathbf{x}.$$

Finally, suppose that $\mathbf{x} = \begin{bmatrix} \mathbf{e}^T & \alpha & \alpha \end{bmatrix}^T$ is an eigenvector. Then

$$L_s\mathbf{x} = \begin{bmatrix} (2+2\alpha)\mathbf{e} \\ n+(n+2)\alpha \\ n+(n+2)\alpha \end{bmatrix}$$

and hence $\lambda = 2+2\alpha$ and $\lambda\alpha = n+(n+2)\alpha$ which is equivalent to the identity $2\alpha^2 - n\alpha - n = 0 \Rightarrow \alpha = \frac{n\pm\sqrt{n^2+8n}}{2} \Rightarrow \lambda = 2+2\alpha = 2+\frac{n+\sqrt{n^2+8n}}{2}$.

Finally, we have $\sigma(L_s - 2I) = \{[0]^{n-1}, [n-2]^1, \frac{n+\sqrt{n^2+8n}}{2}, \frac{n-\sqrt{n^2+8n}}{2}\}$ so

$$\sigma(L_s) = \{[2]^{n-1}, [n]^1, 2+\frac{n+\sqrt{n^2+8n}}{2}, 2+\frac{n-\sqrt{n^2+8n}}{2}\}.$$

Then $\lambda_{\min}(L_s) = 2+\frac{n-\sqrt{n^2+8n}}{2}$ and

$$\begin{aligned}
\lim_{n\to\infty} 2+\frac{n-\sqrt{n^2+8n}}{2} &= 2+\lim_{n\to\infty}\frac{-8n}{2(n+\sqrt{n^2+8n})} \\
&= 2-\lim_{n\to\infty}\frac{4}{1+\sqrt{1+\frac{8}{n}}} \\
&= 2-\frac{4}{1+1} = 0
\end{aligned}$$

So, $\lambda_{\min}(L_s) \to 0$ as $n \to \infty$ and hence

$$\lim_{n\to\infty} b_{L_s} = \frac{1}{1+\lambda_{\min}(L_s)} = 1.$$

Turning to a measure based on the normalised Laplacian, for our adjacency matrix A the normalised Laplacian is

$$L_N = I - D^-1/2AD^-1/2 = I - \begin{bmatrix} O & \frac{1}{\sqrt{2(n+1)}}E_{n2} \\ \frac{1}{\sqrt{2(n+1)}}E_{n2}^T & \begin{bmatrix} 0 & \frac{1}{n+1} \\ \frac{1}{n+1} & 0 \end{bmatrix} \end{bmatrix}.$$

Again, since $\text{rank}(A) \leq 3$, we can find the spectrum of L_N by identifying the three eigenvalues of L_D not equal to 1. One of these is zero (corresponding to the eigenvector $D^{1/2}\mathbf{e}$). One can also confirm by direct calculation that $\begin{bmatrix} \mathbf{0}^T & 1 & -1 \end{bmatrix}^T$ is an eigenvector with eigenvalue $1 + 1/(n+1)$ and then since $\text{tr}(L_N) = n+2$ the final eigenvalue is $2 - 1/(n+1)$.

Thus we can conclude that $b_{L_N} = 2 - \lambda_{\max} = 2 - \frac{2n+1}{n+1}$ and $b_{L_N} \to 0$ as $n \to \infty$.

4.2.4 Example 2

We extend Example 1 by adding a node to the smaller partition and an additional edge. Thus we let G be the graph with $n+3$ nodes whose adjacency matrix is

$$A = \begin{bmatrix} P_2 & E_{n3} \\ E_{n3}^T & O \end{bmatrix}$$

where

$$P_2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

represents a path graph.

As in Example 1, A has very low rank (looking at the last n columns we can bound the rank above by 4). In order to find the full spectrum of A we first calculate its null space.

If $\mathbf{x} = \begin{bmatrix} 0 & 0 & 0 & x_4 & x_5 & \dots & x_{n+3} \end{bmatrix}^T$ where $\sum_{j=4}^{n+3} x_j = 0$ then $A\mathbf{x} = 0$. We have $n-1$ degrees of freedom in choosing the x_j and a basis for the null space is given by

$$\left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \dots, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \right\}.$$

We normalise this null space and use it to apply a similarity transformation to A, namely the one induced by

$$V = \begin{bmatrix} I_3 & O \\ O & Z \end{bmatrix},$$

where

$$Z = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & \mathbf{e}^T \\ \mathbf{e} & -I \end{bmatrix}.$$

Since

$$Z^{-1} = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & \mathbf{e}^T \\ \mathbf{e} & E - nI \end{bmatrix},$$

we have

$$V^{-1}AV = \begin{bmatrix} 0 & 1 & 0 & \sqrt{n} & 0 & \dots & 0 \\ 1 & 0 & 1 & \sqrt{n} & 0 & \dots & 0 \\ 0 & 1 & 0 & \sqrt{n} & 0 & \dots & 0 \\ \sqrt{n} & \sqrt{n} & \sqrt{n} & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}.$$

To find the eigenvalues of

$$X = \begin{bmatrix} 0 & 1 & 0 & \sqrt{n} \\ 1 & 0 & 1 & \sqrt{n} \\ 0 & 1 & 0 & \sqrt{n} \\ \sqrt{n} & \sqrt{n} & \sqrt{n} & 0 \end{bmatrix}$$

we use a further similarity transformation. Namely, if

$$Q = \begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{2} & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

then

$$\widehat{A} = QXQ^T = \begin{bmatrix} \sqrt{2} & 0 & 0 & \sqrt{n}(1+\frac{1}{\sqrt{2}}) \\ 0 & -\sqrt{2} & 0 & \sqrt{n}(1-\frac{1}{\sqrt{2}}) \\ 0 & 0 & 0 & 0 \\ \sqrt{n}(1+\frac{1}{\sqrt{2}}) & \sqrt{n}(1-\frac{1}{\sqrt{2}}) & 0 & 0 \end{bmatrix}.$$

Now $\det(\lambda I - \widehat{A}) = \lambda^3 - (3n+2)\lambda - 4n$. So A has spectrum

$$\{[0]^n, [\lambda_1]^1, [\lambda_2]^1, [\lambda_3]^1\},$$

where $\lambda_1, \lambda_2, \lambda_3$ are roots of $\lambda^3 - (3n+2)\lambda - 4n$.

An asymptotic analysis shows that the root of $\lambda^3 - (3n+2)\lambda - 4n$ are $\lambda_1 = \sqrt{3n} + \frac{2}{3} + o(1)$, $\lambda_2 = -\sqrt{3n} + \frac{2}{3} + o(1)$ and $\lambda_3 = \frac{-4}{3} + o(1)$ and so

$$\lim_{n\to\infty} b_\lambda = \lim_{n\to\infty} \frac{\sqrt{3n}}{\sqrt{3n}} = 1,$$

but

$$\lim_{n\to\infty} b_e = \lim_{n\to\infty} \frac{n + e^{-\sqrt{3n}-\frac{2}{3}} + e^{\sqrt{3n}-\frac{2}{3}} + e^{\frac{-4}{3}}}{n + e^{\sqrt{3n}+\frac{2}{3}} + e^{-\sqrt{3n}+\frac{2}{3}} + e^{\frac{4}{3}}} = \frac{e^{\frac{-2}{3}}}{e^{\frac{2}{3}}} = e^{\frac{-4}{3}}.$$

In order to generalize what is going on in the previous examples, let's construct a family of networks parameterized by n and m.

Theorem 5. There is a family of networks parameterised by n such that as $n \to \infty$, $b_\lambda \to 1$, $b_C \to 1$ and $b_e \to 0$.

Proof. We construct a graph G_{nm} where $(m = n^2)$ with $n^2 + n$ nodes where n nodes are connected to every node in the graph (except themselves) but there are no intra-connections between the other n^2 nodes. If $n = 2$ this is an agave graph. The adjacency matrix of the graph G_n is

$$A = \begin{bmatrix} K_n & E_{nn^2} \\ E^T_{nn^2} & O \end{bmatrix},$$

where K_n is the adjacency matrix of a complete graph. We can calculate the spectrum of A explicitly given its relatively low rank. The result can be found in [82, pp. 138–141], where this graph is an instance to the class of networks with an n-fully meshed star topology but here we give a simple derivation of the result. To do this we first apply the similarity transformation induced by $V = \begin{bmatrix} I_n & O \\ O & Z \end{bmatrix}$, where $nZ = \begin{bmatrix} 1 & \mathbf{e}^T \\ \mathbf{e} & -I_{n^2-1} \end{bmatrix}$.

Then

$$V^{-1}AV = \begin{bmatrix} K_n & ne & O \\ ne^T & 0 & O \\ O & O & O \end{bmatrix}.$$

To find the eigenvalues of $X = \begin{bmatrix} K_n & n\mathbf{e} \\ n\mathbf{e}^T & 0 \end{bmatrix}$ we use the spectral decomposition of the complete graph $K_n = QDQ^T$ (where the eigenvalues are ordered so that the bottom right element of D is $n-1$ and hence the final column of Q is $\mathbf{e}/\sqrt{n}$). Thus, using the orthogonality of Q, we find

$$\begin{bmatrix} Q^T & O \\ O & 1 \end{bmatrix} X \begin{bmatrix} Q & O \\ O & 1 \end{bmatrix} = \begin{bmatrix} D & nQ^T\mathbf{e} \\ n\mathbf{e}^TQ & 0 \end{bmatrix} = \begin{bmatrix} -I & \mathbf{0} & \mathbf{0} \\ \mathbf{0}^T & n-1 & n^{3/2} \\ \mathbf{0}^T & n^{3/2} & 0 \end{bmatrix}.$$

Hence

$$\sigma(G_n) = \left\{ [0]^{n^2-1}, [-1]^{n-1}, [f(n)+g(n)]^1, [f(n)-g(n)]^1 \right\},$$

where

$$f(n) = \frac{n-1}{2}, \quad g(n) = \sqrt{\frac{(n-1)^2}{4} + n^3}.$$

Now we compute the bipartivity measures.

Firstly note that we can make the network bipartite by removing the $n(n-1)/2$ edges from the K_n block and hence

$$b_C \geq 1 - \frac{n(n-1)}{n^3 + n(n-1)/2} \geq 1 - \frac{1}{n}.$$

Next,

$$b_\lambda = \frac{g(n) - f(n)}{g(n) + f(n)} = 1 - o(n).$$

Finally,

$$b_e = \frac{(n^2-1) + (n-1)e + e^{-f(n)-g(n)} + e^{-f(n)+g(n)}}{(n^2-1) + (n-1)e^{-1} + e^{f(n)+g(n)} + e^{f(n)-g(n)}} = \frac{2e^{-f(n)}\cosh g(n) + n^2 - 1 + (n-1)e}{2e^{f(n)}\cosh g(n) + n^2 - 1 + (n-1)e^{-1}}$$

which for large n behaves increasingly like e^{1-n} and hence $b_e \to 0$ as $n \to \infty$.

$\square$

Theorem 6. For the family of networks introduced in Theorem 5 (when $m = n^3$) as $n \to \infty$, $b_{L_s} \to 1$, $b_C \to 1$ and $b_e \to 0$.

Proof. We construct the graph G_{nm} with $m^2 + n$ nodes where n nodes are connected to every node in the graph (except themselves) but there are no intra-connections between the other m nodes. As in Theorem 5, the adjacency matrix of the graph G_{nm} is

$$A = \begin{bmatrix} K_n & E_{nm} \\ E_{nm}^T & O \end{bmatrix}.$$

Where K_n is the adjacency matrix of a complete graph. Then the signless Laplacian is

$$L_s = D + A = \begin{bmatrix} E_n + (m+n-2)I & E_{nm} \\ E_{nm}^T & nI \end{bmatrix}.$$

All but two of the eigenvalues can immediately be deduced. The last m rows of $L_s - nI$ are rank 1, so L_s has $m-1$ copies of n as an eigenvalue. $L_s - (m+n-2)I$ first n rows are rank 1 so L_s has $n-1$ copies of $(n+m-2)$ as an eigenvalue.

We assume that for the two remaining eigenvalues $\mathbf{x} = \begin{bmatrix} \alpha \mathbf{e_n} \\ \mathbf{e}_m \end{bmatrix}$ is an eigenvector of L_s. Then

$$L_s\mathbf{x} = \begin{bmatrix} \alpha E_n e_n + \alpha(m+n-2)e_n + E_{nm}e_m \\ \alpha E_{mn} e_n + n e_m \end{bmatrix} = \begin{bmatrix} (\alpha(m+2n-2)+m)e_n \\ n(\alpha+1)e_m \end{bmatrix}$$

from which we deduce $\alpha\lambda = \alpha(m+2n-2)+m$ and $\lambda = (\alpha+1)n$. Solving gives

$$\begin{aligned} \lambda &= (\alpha+1)n = \left(\frac{(n+m-2) \pm \sqrt{(m+n-2)^2 + 4mn}}{2n} + 1 \right) n \\ &= \frac{(3n+m-2) \pm \sqrt{(m+n-2)^2+4mn}}{2} \end{aligned}$$

so

$$\sigma(L_s) = \left\{ [n]^{m-1}, [n+m-2]^{n-1}, \left[\frac{(3n+m-2) \pm \sqrt{(n+m-2)^2+4mn}}{2} \right]^1 \right\}.$$

The smallest eigenvalue is then

$$\lambda_{\min}(L_s) = \frac{(m+3n-2) - \sqrt{(m+n-2)^2+4mn}}{2} = \frac{(m+3n-2)}{2}\left[1 - \sqrt{1+\frac{8(n^2-n)}{(m+3n-2)^2}}\right].$$

And using the power series for $(1+x)^{\frac{1}{2}}$ gives

$$\begin{aligned}
\lambda_{\min}(L_s) &= \frac{(m+3n-2)}{2}\left[1-1+\frac{1}{2}\frac{8(n^2-n)}{(m+3n-2)^2} - \frac{1}{4}\left(\frac{8(n^2-n)}{(m+3n-2)^2}\right)^2 + \dots\right] \\
&= \frac{(m+3n-2)}{2}\left[\frac{4n(n-1)}{(m+3n-2)^2} - \frac{4n^2(n-1)^2}{(m+3n-2)^4} + \dots\right] \\
&= \frac{2n(n-1)}{(m+3n-2)} - \frac{2n^2(n-1)^2}{(m+3n-2)^3} + \dots .
\end{aligned}$$

In particular, if $m = n^3$,

$$\lambda_{\min} = \frac{2n(n-1)}{(n^3+3n-2)} - \frac{4n^2(n-1)^2}{(n^3+3n-2)^3} + \cdots ,$$

so $\lim_{n\to\infty} \lambda_{\min}(L_s) = \lim_{n\to\infty} \frac{2}{n} + O(n^{-2}) = 0$ and $\lim_{n\to\infty} b_{L_s} = \frac{1}{1+0} = 1$, while b_c, b_e and b_λ all behave as in Theorem 5

$\square$

In Theorems 5 and 6 we have described networks where b_λ, b_{L_s} and b_c behave one way while b_e behaves in another. But an almost opposite scenario is possible.

For our second family we consider networks where the path graph P_N is connected by a single edge to K_n. If we let N grow much faster than n then we can show a very different behaviour from the previous one.

Theorem 7. There is a family of networks parameterised by n such that as $n \to \infty$, $b_\lambda \to 0$, $b_C \to 1$ and $b_e \to 1$.

Proof. Consider the graph where the path graph P_N is connected by a single edge to K_n (call it G_n).

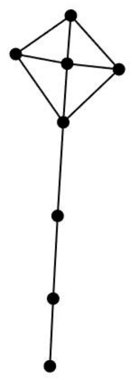


Figure 4.2: Example of G_n when $n = 4$ and $N = 2$.

It has adjacency matrix

$$A = \begin{bmatrix} K_n & C \\ C^T & P_N \end{bmatrix}.$$

where C is a matrix which has a single nonzero entry in its bottom lefthand corner and $\mathbf{P_N}$ is the adjacency matrix of a path graph with N edges.

For sufficiently large n, N we make the assumption that the spectrum of G_n can be approximated to an accuracy of $O(1/n^2)$ by ignoring C and C^T and it is sufficient to consider the graph without the edge linking the path to the complete graph (call this G'_n). We have no analytic proof of this result but the numerical evidence is convincing.

Then

$$\sigma(G'_n) = \sigma(K_n) \cup \sigma(P_N) = \{[n-1]^1, [-1]^{n-1}\} \cup \{2\cos\frac{\pi j}{N+2}, j = 1, \ldots, N+1\}.$$

Turning to the measures, we observe that by removing half the edges from the complete graph we have a bipartite graph and hence $m_d \le n(n-1)/4$ and so so long as $n^2/N = o(n)$ we will have $b_C \to 1$ as $n \to \infty$.

Next note that $\lambda_1(G'(n)) = n - 1$ while $\lambda_{n+N}(G'_n) \ge -2$ and hence $b_\lambda \to 0$ as $n \to \infty$.

Finally,

$$b_e = \frac{(n-1)e + e^{1-n} + S(N)}{(n-1)e + e^{n-1} + S(N)}$$

where

$$S(N) = \sum_{j=1}^{N+1} \exp(2\cos\frac{\pi j}{N+2}) \approx \int_1^{N+1} \exp(2\cos\frac{\pi x}{N+2})dx = \frac{N+2}{\pi}\int_{1/(N+2)}^{\pi-\pi/(N+2)} e^{2\cos x}dx$$

and hence for large N, we can use the approximation

$$S(N) = (N+2)J_0(2i),$$

where $J_0(x)$ is a Bessel function of the first kind. Note that $J_0(2i) \approx 2.28$. We end up with the approximation for large n and N

$$b_e \approx \frac{J_0(2i)N}{e^{n-1} + J_0(2i)N},$$

and if N is sufficiently bigger than n (e.g. $N > e^{2n}$) we see that $b_e \to 1$ as $n \to \infty$. $\square$

We observe that the difference in size of the path graph and the complete graph in this example is rather extreme. To illustrate the result we have computed b_λ and b_e explicitly for $n = 2, \ldots, 8$ with $N = [e^n]$ (with e^{2n} we have a much smaller range of n with which to work computationally). In Figure 4.3 we record the values of b_e and b_λ against n and show the effectiveness of our bounds.

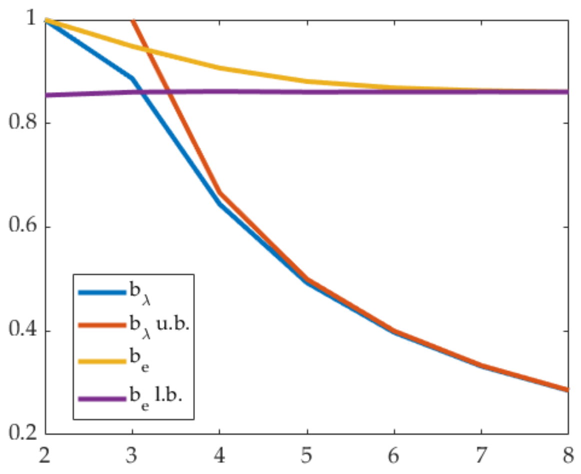


Figure 4.3: b_λ and b_e against n.

4.2.5 Random Networks Bipartivity

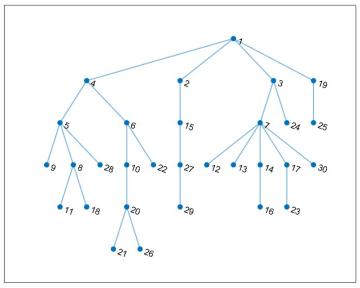
We have introduced a series of measures which are designed to indicate the extent to which a real-world network exhibits bipartite properties. These measures all work on a $[0, 1]$ scale where a network scores 1 if and only if it is exactly bipartite. Typically, a score of 0 is reserved for the least bipartite of all networks, namely large complete graphs.

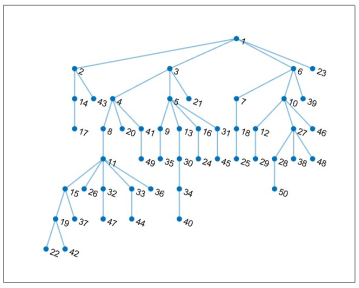
We have seen that the different measures can behave differently in theory. In order to get a feel for which measures are well suited to their purpose we now investigate the performance of some of these measures on various families of random bipartite networks which we then randomly alter so that the original bipartite structure is progressively destroyed. We then analyse results.

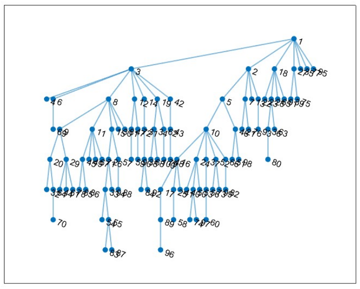
Random trees

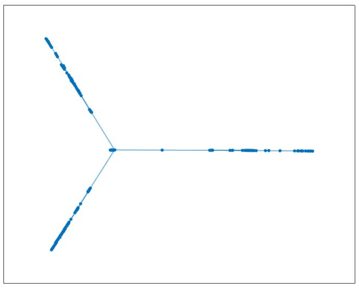
Our first family of bipartite graphs are random trees. They are generated randomly using n nodes where each node is added iteratively so that the kth node is joined randomly to one of the pre-existing $k - 1$ nodes.

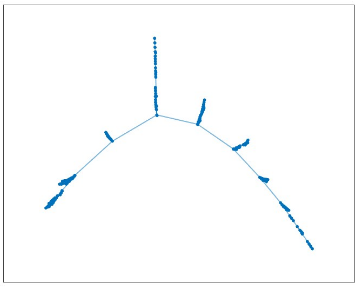
The following figures show some graphs for random trees of different size.

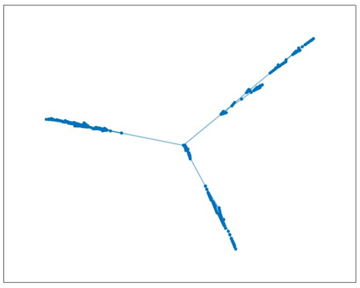

(a) Random tree with $n = 30$.


(b) Random tree with $n = 50$.


(c) Random tree with $n = 100$.


(d) Random tree with $n = 300$.


(e) Random tree with $n = 500$.


(f) Random tree with $n = 800$.

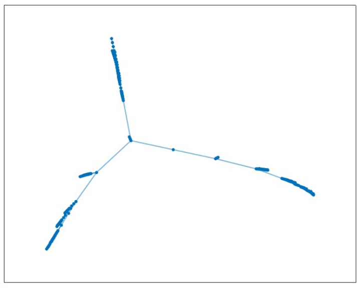
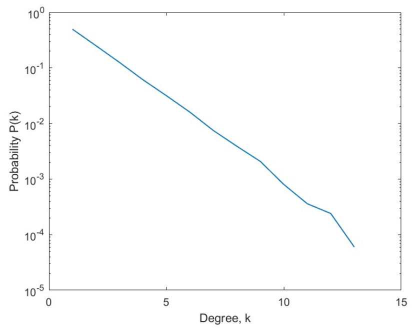
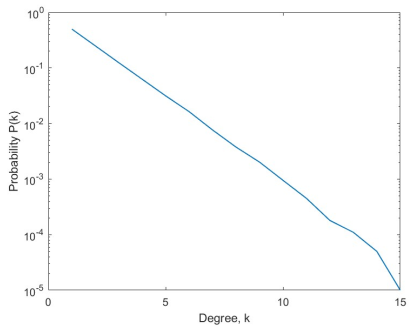

(g) Random tree with $n = 1000$.

Figure 4.4: Random trees with different size.

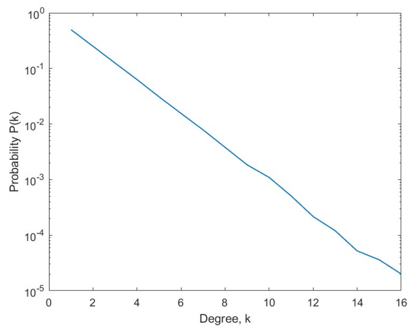
The degree distribution for a random tree follows an exponential distribution as exhibited in the following figures.



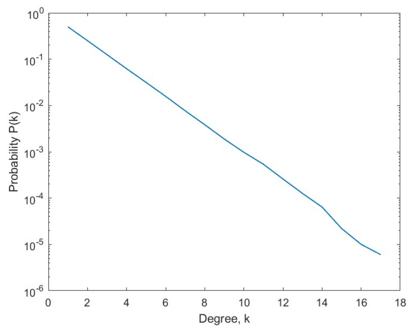
(a) n = 1000.



(b) n = 2000.



(c) n = 5000.



(d) n = 10000.

Figure 4.5: Random tree degree distribution.

Next, we generate another family of trees using Prüfer sequences [74]. To convert a Prüfer sequence into a tree, we complete the following steps: If the Prüfer sequence has n numbers, then the tree has $n + 2$ nodes which are numbered 1 to $n + 2$. The degree is set to the number of times each node appears in the sequence plus 1. We find the first lowest numbered node not included in the sequence and add an edge between the lowest number and the first entry in the sequence. Next, we consider the next number in the sequence and find the corresponding vertex with the least degree not included in the sequence. This process is repeated until we are left with two vertices that we join [33]. For example, convert a Prüfer sequence (1,1,1,1,6,5) into a tree. The given code has 6 entries so, the corresponding tree will have 8 nodes. The first number in Prüfer sequence is 1 and the lowest number not included in the Prüfer sequence is 2 so, 1 connect to 2. We delete the 1 from Prüfer sequence and put 2 at the end (1,1,1,6,5,2). The first number is 1 and the lowest number not included in the Prüfer sequence is 3, so 1 connects to 3. We delete the 1 from Prüfer sequence and put 3 at the end (1,1,6,5,2,3). The first number in Prüfer sequence is 1 and the lowest number not included in the Prüfer sequence is 2 so, 1 connects to 4. We delete the 1 from Prüfer sequence and put 4 at the end (1,6,5,2,3,4). The first number in Prüfer sequence is 1 and the lowest number not included in the Prüfer sequence is 7 so, 1 connects to 7. We delete the 1 from Prüfer sequence and put 7 at the end (6,5,2,3,4,7). The first number in Prüfer sequence is 6 and the lowest number not included in the Prüfer sequence is 1 so, 7 connects to 1. We delete the 6 from Prüfer sequence and put 1 at the end (5,2,3,4,7,1). The first number in Prüfer sequence is 5 and the lowest number not included in the Prüfer sequence is 6 so, 5 connects to 6. We delete the 5 from Prüfer sequence and put 6 at the end (2,3,4,7,1,6). We have iterated all the way through the code and the two numbers missing are 5 and 8. So we connect 5 to 8.

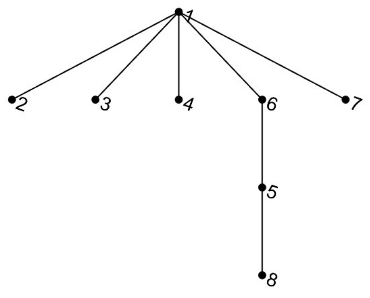
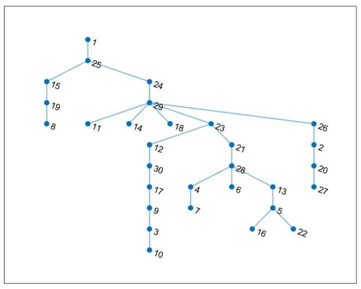
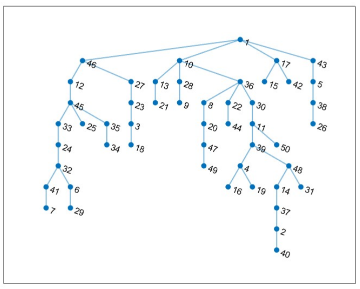


Figure 4.6: Example of convert a Prüfer sequence into a tree.

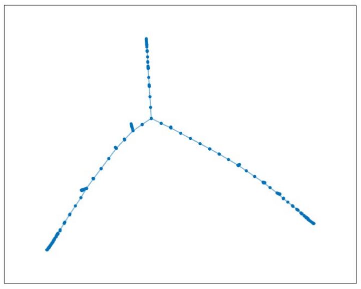
The following figures show some graphs for trees generated by Prüfer sequences of different sizes.



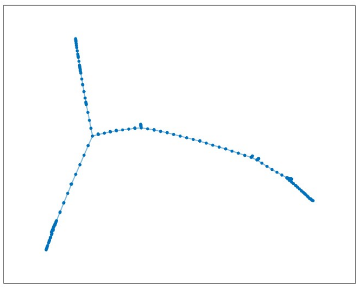
(a) $n = 30$.



(b) $n = 50$.



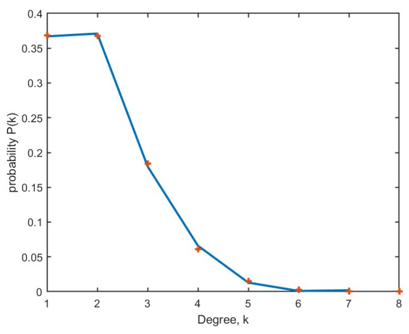
(c) $n = 300$.



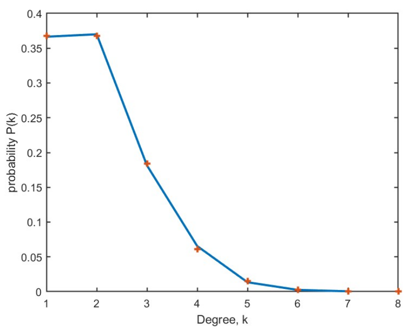
(d) $n = 500$.

Figure 4.7: Tree generated by Prüfer sequences with different size.

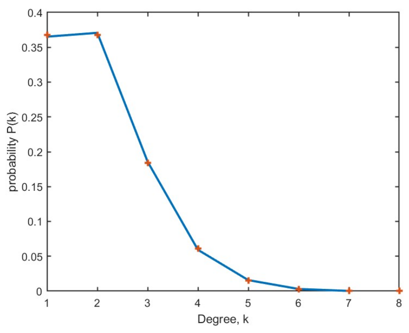
The degree distribution of trees generated by Prüfer sequences follows a Poisson model.



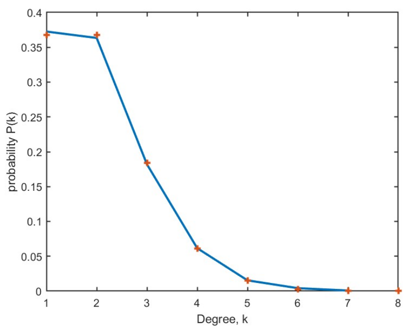
(a) n = 1000.



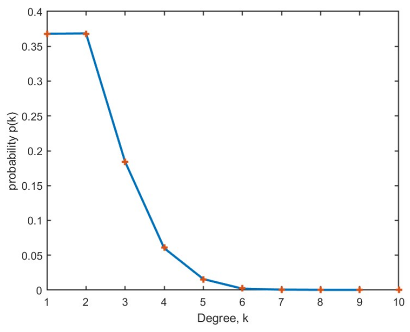
(b) n = 2000.



(c) n = 4000.



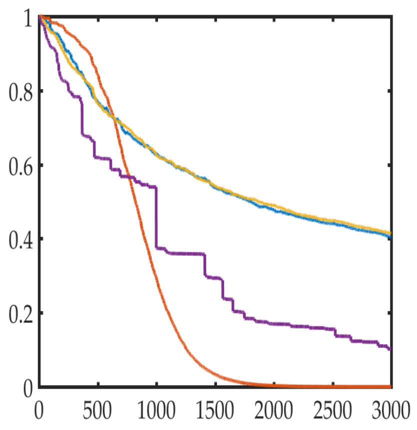
(d) n = 5000.



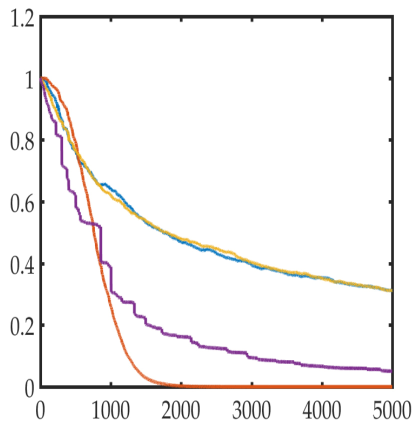
(e) n = 6000.

Figure 4.8: Tree generated by Prufer distribution with different size.

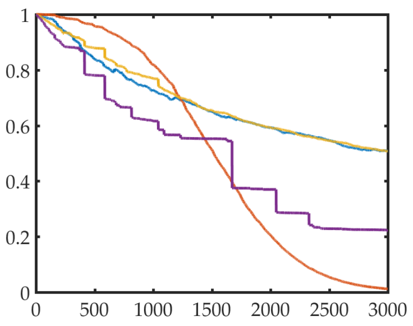
in this experiment, we generated our trees then we added new edges at random. The graphs in Figure 4.9 and 4.10 which have the X-axes are the edges added and Y-axes is the bipartivity measure show the effect of adding edges on the measures b_λ, b_e, b_{L_s} and b_{L_N}. We see from Figure 4.9 and 4.10 that the b_λ (blue line) and b_{L_N} (yellow line) are almost exactly the same they going down slowly, but b_{L_s} (purple line) start with b_λ and b_{L_N} then it much closer to the b_e (red line) which is a bit slower from them then it steeply goes down. So, when we compare the Prufer sequences in Figure 4.10 and random trees in Figure 4.9 we get very similar behavior.



(a) Random tree with n = 300,add = 3000.

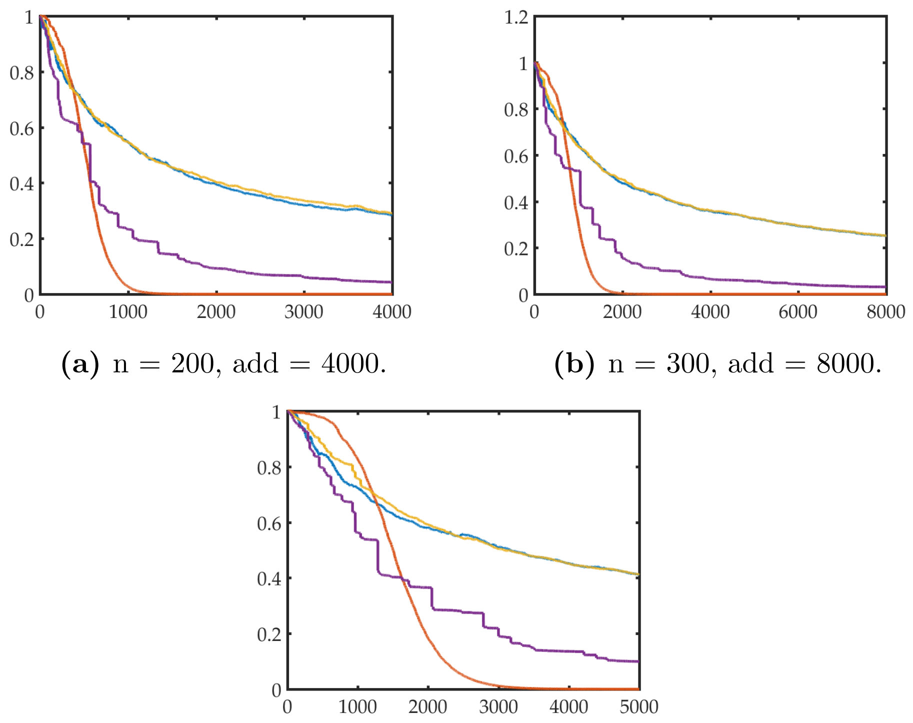


(b) Random tree with n = 300,add = 5000.



(c) Random tree with n=500,add=3000.

Figure 4.9: Bipartivity measures of Random trees with different sizes.



(a) n = 200, add = 4000.

(b) n = 300, add = 8000.

(c) n = 500 ,add = 5000.

Figure 4.10: Trees generated by Prufer sequences.

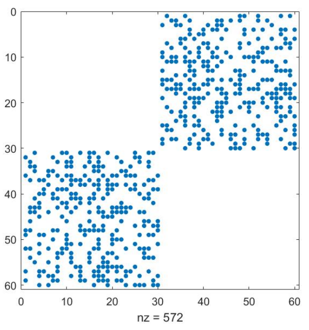
Random graphs

Our next family of random bipartite graphs are formed by generating an adjacency matrix of the form

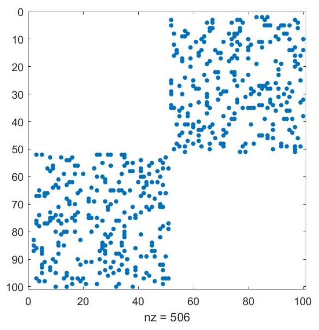
$$\begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$$

where the (binary) entries of B are chosen at random according to some parameter p so that each entry in B has probability p of equaling 1. Thus these matrices are generated in a very similar way to the Erdős–Rényi (ER) model.

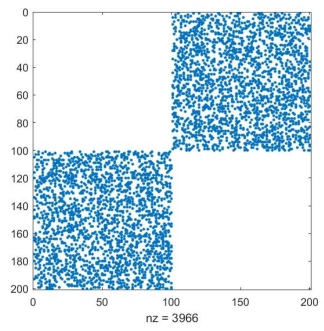
The following figures show some random graphs of different size.



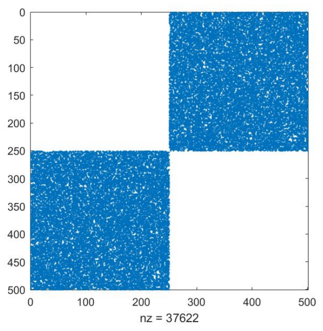
(a) n = 60, p = 0.3.



(b) n = 100, p = 0.1.



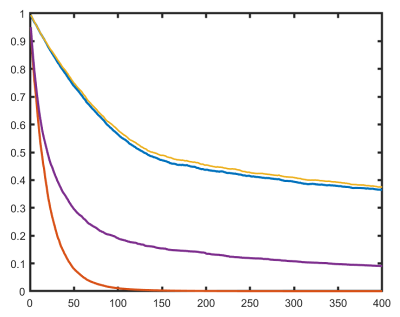
(c) n = 200, p = 0.2.

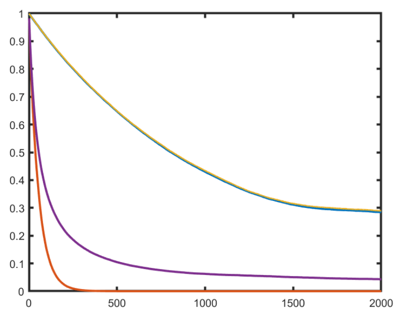


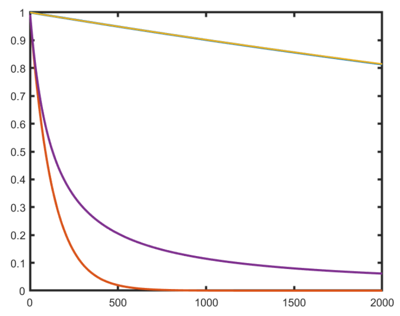
(d) n = 500, p = 0.3.

Figure 4.11: Random graph with different size.

Then, we add some edges to each random graph and compute b_λ, b_e, b_{L_s} and b_{L_N}. Here some figures show the effect of measures when we add edges. We see from Figure 4.12 which have the X-axes are the edges added and Y-axes is the bipartivity measure that b_λ (blue line) and b_{L_N} (yellow line) behave very similar and going down very slowly, but b_e (red line) and b_{L_s} (purple line) goes down steeply and very fast. When we compare this model with the Prufer sequences in Figure 4.10 and random trees in Figure 4.9 we get very different behavior. So, we show that even in random model we get different behavior depending on the type of random model for different bipartivity measures.


(a) random graph with n = 60, p = 0.3, add = 400.


(b) random graph with n = 200, p = 0.2, add = 2000.


(c) random graph with n = 500, p = 0.3, add = 2000.

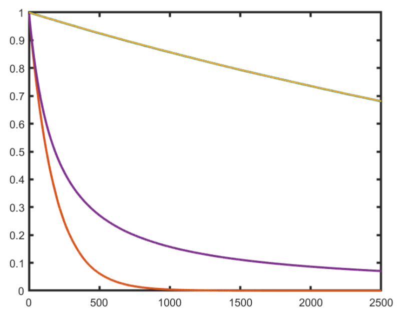

(d) random graph with n = 700, p = 0.1, add = 2500.

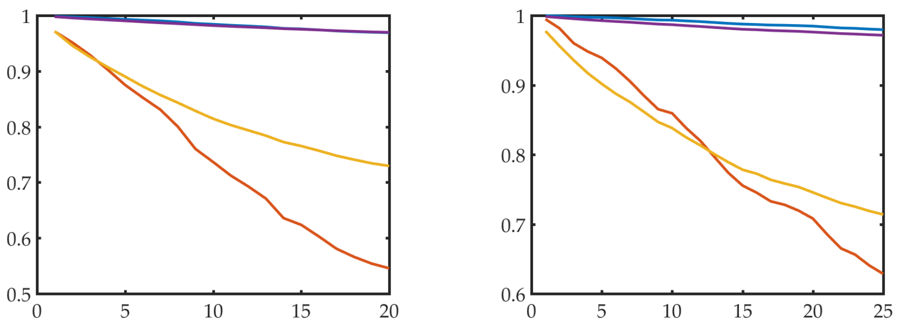
Figure 4.12: Reintroducing edges to random graphs.

4.2.6 Real-world Bipartivity

To get an insight into whether these observed behaviours for purely random graphs are realistic we look at how our bipartivity measures change with some real-world graphs.

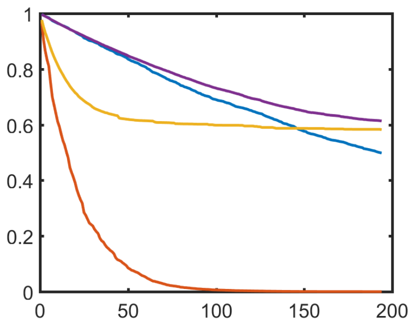
In order to do this we have started with approximate bipartitions we obtain for the real-world graphs by splitting our networks using the signs of the eigenvector of the most negative eigenvalue of the adjacency matrix. We then randomly add back the edges removed in forming these bipartitions and look at how the bipartivity measures evolve. Our graphs show the average changes in 10 different random realisations (the details of this experiment will given in 5).

The measures calculated are b_λ, b_e, b_{L_s} and b_{L_N}.

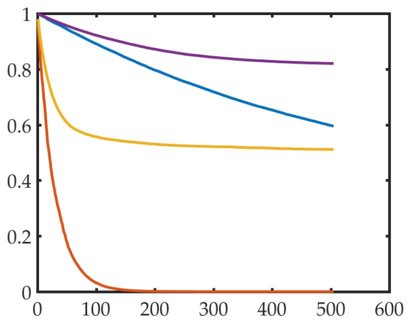


(a) Canton. **(b)** Stony Brook.

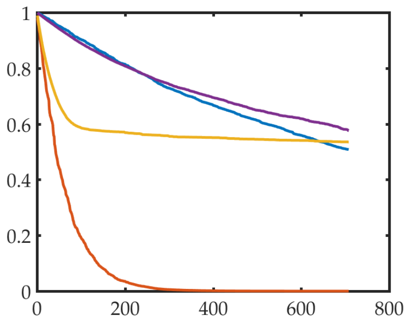
Figure 4.13: Reintroducing edges to real-world graphs.



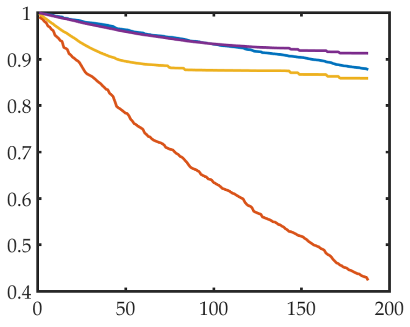
(a) Centrality Literature.



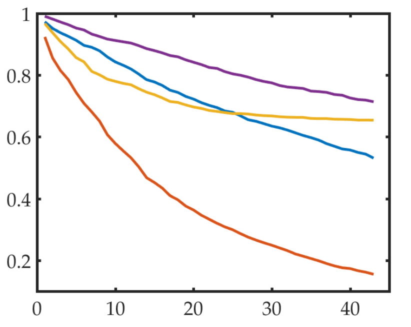
(b) Little Rock.



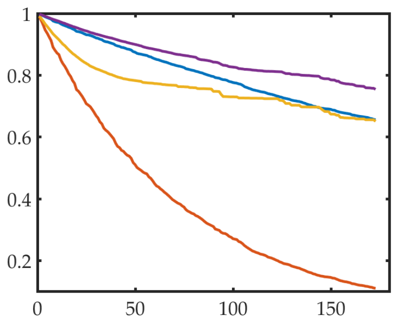
(c) Neurons *C. elegans*.



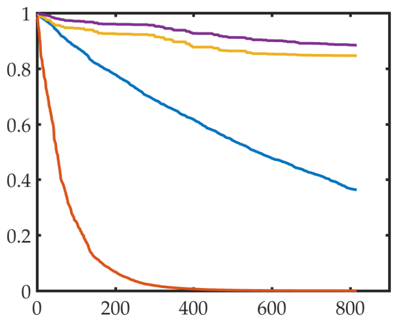
(d) PIN *H. pylori*.



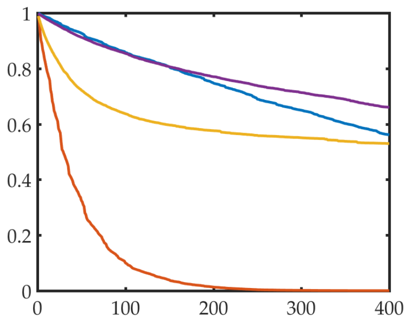
(e) Dolphins.



(f) PIN Malaria.

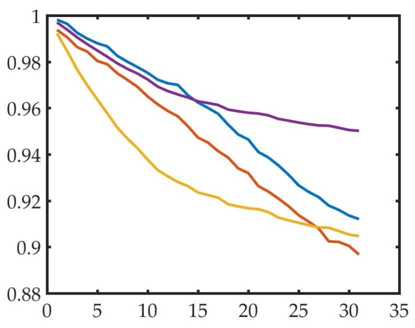


(g) Drugs.

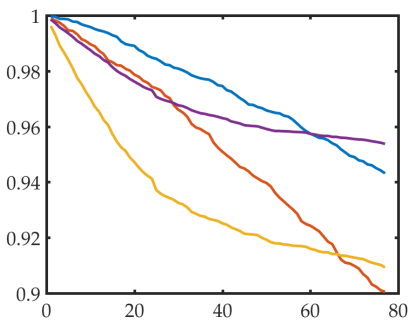


(h) Small World Citations.

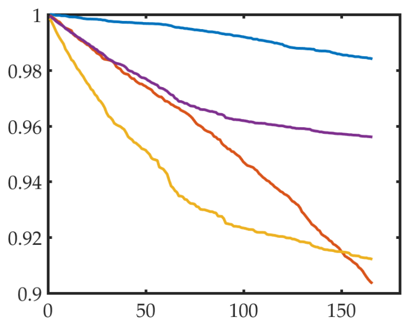
Figure 4.14: Reintroducing edges to real-world graphs.



(a) Electronic 1.

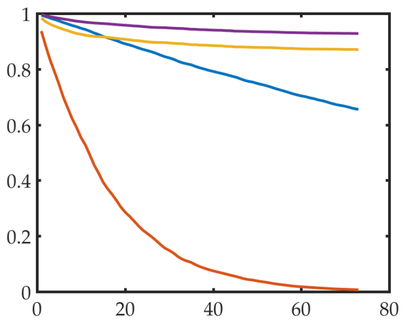


(b) Electronic 2.

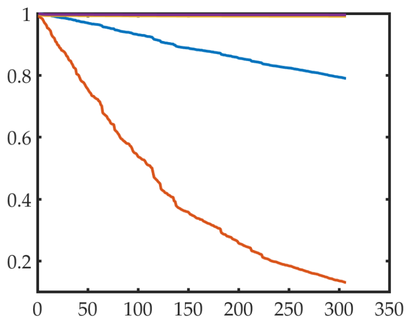


(c) Electronic 3.

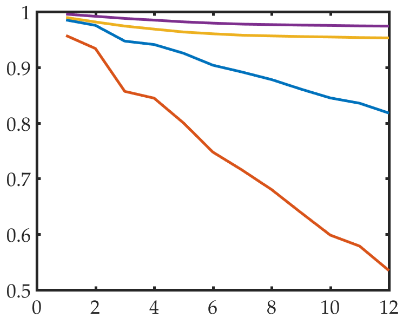
Figure 4.15: Reintroducing edges to real-world graphs.



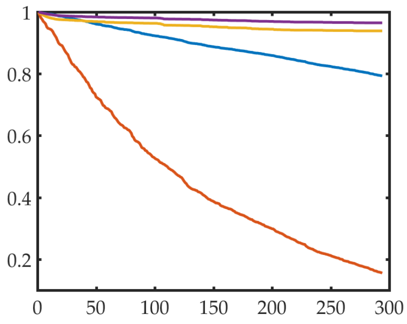
(a) Scotch Broom.



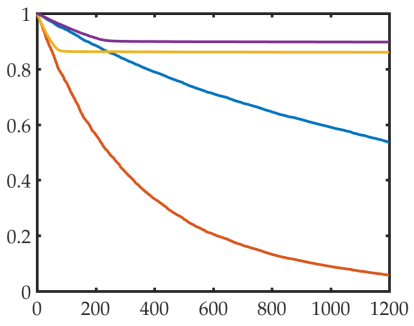
(b) Software Abi.



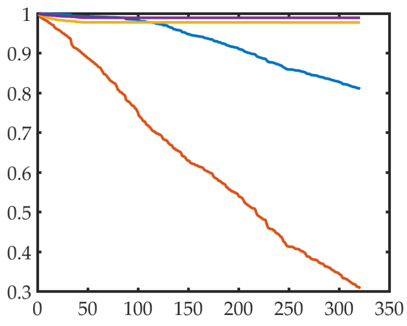
(c) Software Digital.



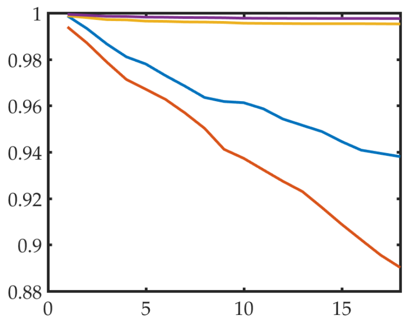
(d) Software VTK.



(e) Roget's Thesaurus.



(f) Software XMS.



(g) Colorado Springs.

Figure 4.16: Reintroducing edges to real-world graphs.

The figures 4.13, 4.14, 4.15 and 4.16 the X-axes are the edges added and Y-axes is the bipartivity measure show the effect of adding edges show groups of real world networks according to the behavior of bipartivity measures. In each group, we see a different type of behavior. We can identify various trends in Figures 4.13, 4.14, 4.15 and 4.16.

We see from Figure 4.13 that b_λ and b_{L_N} are very close together,while b_e decreases faster than b_{L_s}.

We see from Figure 4.14 that b_λ and b_{L_N} are very close together and they decreased very slowly, but b_{L_s} is going down then stop going down and going straight line while the b_e decreased very fast.

We see from Figure 4.15 that b_e decreased very fast and the other measures going down very slowly.

We see from Figure 4.16 that the two lines b_{L_s}, b_{L_N} are very close together and they decreased very slowly, while the lines b_λ and b_e decreased b_e decreased very fast. So, in real world graphs as well as in random graphs, we see that the measures can give us different information and allow us to distinguish between different types of bipartivity.

4.2.7 Fullerene Graphs

Having looked at graphs with varying degrees of randomness we now investigate our measures in real-world graphs which are known to be almost bipartite. Our first experiment is a fullerene molecule.

The importance of studying graphs that model fullerene molecules is that graph properties can be related to the most stable arrangements of these molecules. Certain graph properties have been analyzed to see how well they can predict stability, amongst these is the smallest number in a set of edges that can be removed to make a graph bipartite.

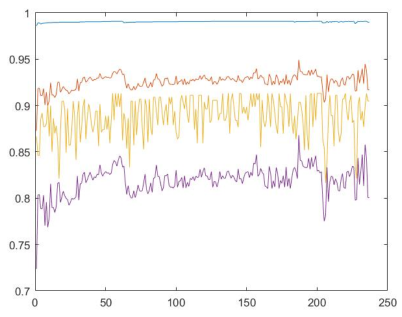
Fullerene molecules often contain isolated pentagons and hence are not bipartite. These are known as IP fullerenes.

There is a relationship between the chemical stability of fullerenes and how non-bipartite they are which is confirmed by computed results that show good performance on eight different arrangements. A question to be considered is whether bipartite edge frustration can predict stability in different fullerene arrangements or isomers.

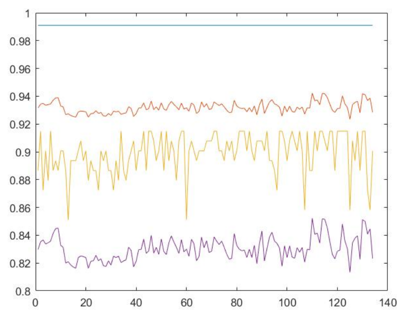
Stability can be measured by calculating the values of certain invariants. While the value ranges for these invariant properties is narrow, there seems to be good distribution as maximal values can be calculated for some isomers. Thus, the potential for application in stability prediction might be in ranges where there are no stable isomers. In this case, the calculation of frustrated edges would have an advantage over other spectral predictors since the computations are all performed exactly [21].

We used the measures $b_e = \frac{\sum_{j=1}^{n} e^{-\lambda_j}}{\sum_{j=1}^{n} e^{\lambda_j}} = \frac{\mathrm{tr}(e^{-A})}{\mathrm{tr}(e^{A})}$, $b_\lambda = \frac{|\lambda_n|}{\lambda_1}$, $b_c = 1 - \frac{m_d}{m}$and $b_{L_s} = \frac{1}{1+\lambda_{\min}(L_s)}$.

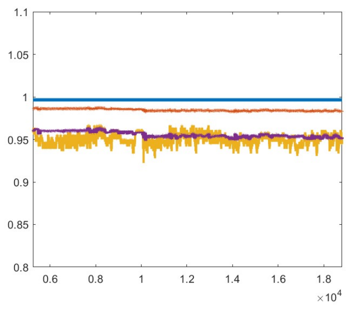
We tested the measures on three sets of fullerene graphs. C_1 is a collection of 237 IP fullerene graphs with between 20 and 92 nodes. C_2 is a collection of 134 IP fullerene graphs all with 94 nodes. C_3 is much bigger and contains C_{240} This is a collection of 18825 IP fullerene graphs with 240 nodes. Then we look to the number of edges we must remove to get bipartite graphs.



(a) C_1



(b) C_2



(c) C_3

Figure 4.17: measure of bipartivity.

In Figure 4.17 we plot the measures b_e (blue line), b_λ (red line), b_c (yellow line) and b_{L_s} (purple line) for the sets C_1, C_2 and C_3 (the x-axis is the index of a graph in a set). In this case all the measures perform similarly although they underestimate b_c. The b_e is closest to 1.

4.2.8 Airline Graphs

In this experiment we are looking at 37 airline networks (Lufthansa, Ryanair, Easyjet, British Airways, Turkish Airlines, Air Berlin, Air France, Scandinavian Airlines, KLM, Alitalia, Swiss International Air Lines, Iberia, Norwegian Air Shuttle, Austrian Airlines, Flybe, Wizz Air, TAP Portugal, Brussels Airlines, Finnair, LOT Polish Airlines, Vueling Airlines, Air Nostrum, Air Lingus, Germanwings, Pegasus Airlines, Netjets, Transavia Holland, Niki, SunExpress, Aegean Airlines, Czech Airlines, European Air Transport, Malev Hungarian Airlines, Air Baltic, Windroe, TNT Airways and Olympic Air). We testing them with different bipartivity measures and with LSA (the details of what is LSA will give in Chapter 5), $b_e = \frac{\sum_{j=1}^{n} e^{-\lambda_j}}{\sum_{j=1}^{n} e^{\lambda_j}} = \frac{\operatorname{tr}(e^{-A})}{\operatorname{tr}(e^{A})}, b_\lambda = \frac{|\lambda_n|}{\lambda_1}, b_{L_s} = \frac{1}{1+\lambda_{\min}(L_s)}, b_{L_N} = |1-\lambda_{\max}(L_N)|, m_1 = 0.5 - Q, m_2 = \frac{0.5-Q}{1.5-Q}$ [67].

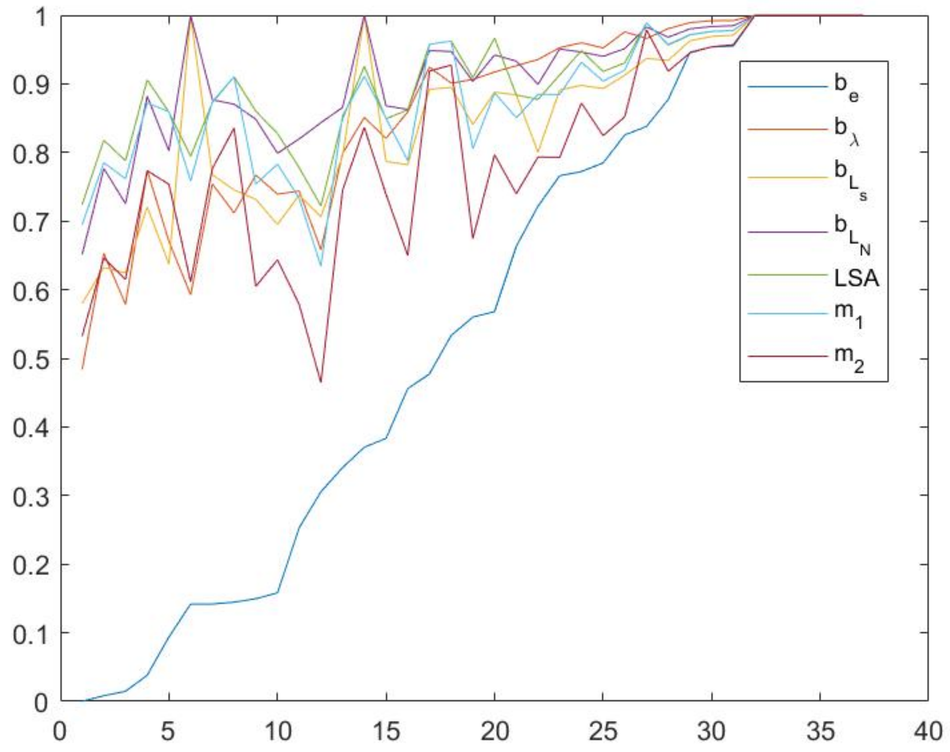


Figure 4.18: Airlines network.

We compute the matrix correlations of the Figure 4.18:

	b_e	b_λ	b_{L_s}	b_{L_N}	LSA	m_1	m_2
b_e	1	0.92669136	0.840815	0.802489	0.812458	0.784726	0.80667
b_λ	0.926691	1	0.809804	0.829564	0.88998	0.830625	0.829653
b_{L_s}	0.840815	0.80980414	1	0.968017	0.802921	0.770344	0.783466
b_{L_N}	0.802489	0.82956402	0.968017	1	0.827835	0.783379	0.78907
LSA	0.812458	0.88998004	0.802921	0.827835	1	0.957836	0.955438
m_1	0.784726	0.83062543	0.770344	0.783379	0.957836	1	0.996158
m_2	0.80667	0.82965299	0.783466	0.78907	0.955438	0.996158	1

Table 4.2: The correlations between the measures.

Each pair of columns in Table 4.2 shows the correlation between two specific measures. For example, the first and second column shows the correlation between b_e and b_λ, which indicates that they're strongly positively correlated. Also, the first and third column shows the correlation between b_e and b_{L_s}, which indicates that they're strongly positively correlated. The first and fourth column shows the correlation between b_e and b_{L_N}, which indicates that they're strongly positively correlated. The most correlation is between m_1 and m_2 then b_{L_s} and b_{L_N}.

4.2.9 Conclusion

Since bipartivity has so many equivalent characterisations there are many ways one can try and measure nearness to the property. We have restricted our analysis to spectral measures, but this still gives us plenty to compare. Measures can be divided into those which use a purely algebraic characterisation of bipartivity (in terms of spectral properties) and those which use the existence of odd cycles as a more direct manifestation of a lack of bipartivity. In experiments it seems that both approaches give insightful results.

In terms of performance versus cost we recommend b_{L_N} since its cost is essentially just that of finding the largest eigenvalue of a symmetric matrix for which an upper bound is already available and thus is generally very cheap to compute for even very large networks.

Overall, it seems that measures based on the adjacency matrix and the normalised Laplacian seem to be closely related to each other, and our results on minimising modularity suggest that these are well correlated to the presence of frustrated edges.

Our theoretical analysis shows that there is the possibility that measures based on very similar spectral ideas can diverge markedly. In particular, measures that look at the discrepancy between the number of even and odd walks between nodes seem to decay very sharply. It may be that we can exploit this to get a more nuanced view of the approximate bipartivity by using more than one measure on the same matrix.

Depending on the measures used to calculate bipartivity within even a single graph can vary significantly thus illustrating the importance of carefully selecting the measures used.

In our experiment, we examine both real networks and random models with different measures and we show that the measures behave differently in theory. Also, the different between these measures vary.

Chapter 5

Finding Anti-communities

5.1 Introduction

We've thoroughly analysed bipartivity and near-bipartivity. We now look at methods that can reliably identify almost bipartite structures in networks.

In this chapter, we aim to find a way of partitioning the graph so that in the partition there are as few connections as possible. Finding the optimal solution is an NP-hard problem and clearly related to finding community structures. We refer to this opposing process as finding anti-communities. According to [63] , if we look at the complement graph we can find communities and anti-communities can be found in the original graph. This approach can be costly but provides us with a starting point.

Newman proposed a method for detecting both communities and anti-communities by minimising modularity. Given that positive eigenvalues of the modularity matrix

$$B = A - \frac{kk^T}{2m}$$

provide information about network structure, it stands to reason that negative eigenvalues of the modularity matrix or would provide us with information about anti-community structure.

In [63], Newman shows that modularity maximisation is analogous to finding the Fiedler vector when one replaces the Laplacian matrix with the modularity matrix B, since $Q = \frac{1}{4m}\mathbf{s}^T B\mathbf{s}$ where $\mathbf{s}$ is the indicator vector for a bipartition. If B has the Schur factorisation $B = \mathbf{u}\hat{B}\mathbf{u}^T$ where $\hat{B} = \mathrm{diag}(\beta_1, \beta_2, \ldots, \beta_n)$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$ then

$$Q = \frac{1}{4m}\sum_{i=1}^{n} a_i^2 \beta_i$$

where we write $\mathbf{s}$ as a linear combination of eigenvectors $\mathbf{s} = \sum_{i=1}^{n} a_i \mathbf{u}_i$ and $a_i = \mathbf{u_i}^T\mathbf{s}$. While the continuous problem

$$\max_{s\in\mathbb{R}^n} Q$$

is straightforward to solve, the discrete problem (with $s_i \in \{-1,1\}$)) is known to be NP-hard but can be approximated by choosing

$$s_i = \begin{cases} 1, & u_i^{(1)} \geq 0 \\ -1, & u_i^{(1)} < 0 \end{cases}.$$

If we want to split into several communities we use the matrix S where

$$s_{ij} = \begin{cases} 1, & \text{if node } i \text{ in community } j, \\ 0 & \text{otherwise,} \end{cases}$$

to measure modularity, then compute

$$Q = \mathrm{tr}(S^T B S). \tag{5.1}$$

For a bipartition, where we wish to minimise modularity, Newman simply suggests using the smallest eigenvalue of B and splitting the nodes according to

$$s_i = \begin{cases} 1, & u_i^{(n)} \geq 0 \\ -1, & u_i^{(n)} < 0 \end{cases}.$$

Newman goes further and suggests a method for finding multiple anti-communities using (5.1) but as our focus solely is on bipartitions we will not consider this further.

Wang et al. (2008) developed an algorithm based on spectral analysis which would show the structure of a complex network using Newman's method of recreating the expression of the modularity using eigenvectors and eigenvalues of the modularity matrix to change the problem into one which questions the partition of the vector and then proposing a new vector partition algorithm [85].

It has been argued that since the modularity matrix or Laplacian matrix used is designed for the community, partitioning with the least modularity will not necessarily be best for partitioning anti-communities. Therefore, the concept of anti-modularity has been proposed by Chen et al. [13] where the characteristics of anti-communities are studied and a label propagation algorithm is presented with experimental results.

In Chapter 4 we classified equivalent definitions of bipartivity. In this chapter, we are going to use these equivalent ways to find the partitions. If we have an approximately bipartite graph we can use these equivalent characterizations to motivate algorithms for finding a way to partition a graph with few frustrated edges as possible. A summary of the algorithms we present and their analysis can be found in [3].

Theorem 8. Suppose G is a connected bipartite graph with partitions P and Q and adjacency matrix A. Label P and Q so that $P \cup Q = \{1, \ldots, n\}$. Let $\mathbf{x}$ be one of the following eigenvectors.

1. The eigenvector associated with the most negative eigenvalue of A.

2. The eigenvector associated with the smallest eigenvalue of the signless Laplacian of G.

3. The eigenvector associated with the largest eigenvalue of the normalised Laplacian of G.

4. The eigenvector associated with the largest eigenvalue of e^{-A}.

5. The eigenvector associated with the Fiedler vector of the complement graph.

6. The eigenvector associated with the largest eigenvalue of the Laplacian matrix.

Then nodes i and j are in the same partition if and only if $x_i x_j > 0$.

The results 1–5 are well known. The results 1–4 can be inferred from [67] and 5 is a simple corollary of Fiedler's results for the Laplacian of graphs [28]. We include the proofs for completeness. 6 is new.

Proof. 1. We know that if a graph is bipartite then the eigenvector we are looking for is guaranteed to give an exact bipartition.

Suppose that the network is bipartite. So, we can write the adjacency matrix

$$A = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$$

and let $\mathbf{x} = \begin{bmatrix} \mathbf{x_1} \\ \mathbf{x_2} \end{bmatrix}$ is eigenvector associated with eigenvalue λ .

$A\mathbf{x} = \lambda\mathbf{x} \Rightarrow \begin{bmatrix} O & B \\ B^T & O \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ \mathbf{x_2} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{x_1} \\ \mathbf{x_2} \end{bmatrix} \Rightarrow B\mathbf{x_2} = \lambda\mathbf{x_1}, B^T\mathbf{x_1} = \lambda\mathbf{x_2}.$

Then if $\mathbf{y} = \begin{bmatrix} \mathbf{x_1} \\ -\mathbf{x_2} \end{bmatrix} \Rightarrow A\mathbf{y} = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ -\mathbf{x_2} \end{bmatrix} = \begin{bmatrix} -B\mathbf{x_2} \\ B^T\mathbf{x_1} \end{bmatrix} = \begin{bmatrix} -\lambda x_1 \\ \lambda x_2 \end{bmatrix} = -\lambda \begin{bmatrix} \mathbf{x_1} \\ -\mathbf{x_2} \end{bmatrix}.$

So, we can use the signs of $\mathbf{y}$ to find the partition.

2. Suppose that $A = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$ where B is an $m \times n$ matrix and let e_p be a vector of p ones.

The signless Laplacian is $L_s = D + A$, where $D = diag(A\mathbf{e})$.

$$Ae = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix} \begin{bmatrix} \mathbf{e_m} \\ \mathbf{e_n} \end{bmatrix} = \begin{bmatrix} B\mathbf{e_n} \\ B^T\mathbf{e_m} \end{bmatrix}, D = \text{diag} \begin{bmatrix} B\mathbf{e_n} \\ B^T\mathbf{e_m} \end{bmatrix}$$

$$L_s = A+D = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix} + \begin{bmatrix} \text{diag}(B\mathbf{e_n}) & O \\ O & \text{diag}(B^T\mathbf{e_m}) \end{bmatrix} = \begin{bmatrix} \text{diag}(Be_n) & B \\ B^T & \text{diag}(B^Te_m) \end{bmatrix}$$

$$L_s \begin{bmatrix} \mathbf{e_m} \\ -\mathbf{e_n} \end{bmatrix} = \begin{bmatrix} \text{diag}(Be_n).e_m - Be_n \\ B^Te_m - \text{diag}(B^Te_m).e_n \end{bmatrix} = 0.$$

So, we can use the sign of $\begin{bmatrix} \mathbf{e_m} \\ -\mathbf{e_n} \end{bmatrix}$ to find the partition.

3. We need to show that the eigenvector associated with the eigenvalue 2 of $L_N = I - D^{-1/2}AD^{-1/2}$ has the form $\begin{bmatrix} x \\ -y \end{bmatrix}$. If z is eigenvector for $I - D^{-1}A$, then $D^{-1/2}z$ is eigenvector of L_N.

   Let $C = I - D^{-1}A = \begin{bmatrix} I & -E \\ -F & I \end{bmatrix}$

$$Cz = C\begin{bmatrix} \mathbf{e_m} \\ -\mathbf{e_n} \end{bmatrix} = \begin{bmatrix} 1+0+0+\cdots+\underbrace{\frac{1}{k_1}+\frac{1}{k_1}+\frac{1}{k_1}+\cdots+\frac{1}{k_1}}_{k_1\text{copies}} \\ 0+1+0+\cdots+\underbrace{\frac{1}{k_2}+\frac{1}{k_2}+\frac{1}{k_2}+\cdots+\frac{1}{k_2}}_{k_2\text{copies}} \\ \vdots \\ \underbrace{\frac{-1}{k_{m-1}}+\frac{-1}{k_{m-1}}+\frac{-1}{k_{m-1}}+\cdots+\frac{-1}{k_{m-1}}}_{k_{m-1}\text{copies}} +0+\ldots 0-1 \\ \underbrace{\frac{-1}{k_m}+\frac{-1}{k_m}+\frac{-1}{k_m}+\frac{-1}{k_m}+\cdots+\frac{-1}{k_m}}_{k_m\text{copies}} +0+\ldots 0-1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+0+0+\cdots+\frac{k_1}{k_1} \\ 0+1+0+\cdots+\frac{k_2}{k_2} \\ \vdots \\ \frac{-k_{m-1}}{k_{m-1}} - 1 + 0 + \ldots 0 \\ \frac{-k_m}{k_m} - 1 + 0 + \ldots 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ \vdots \\ -2 \\ \vdots \\ -2 \end{bmatrix} = 2\begin{bmatrix} \mathbf{e_m} \\ -\mathbf{e_n} \end{bmatrix}$$

So, $\mathbf{z}$ is the eigenvector of C associated with the eigenvalue 2. Let $X = D^{-1/2}z =$

$$\begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix},$$

the $L_N X = \begin{bmatrix} I & -H \\ -H^T & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix} = 2 \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix}$

and we can use the signs of $\begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix}$ to find the partition.

4. $e^{-A} = \sum_{k=1}^{\infty} \frac{(-1)^k}{k!} A^k = \sum_{k=1}^{\infty} \frac{(-1)^k}{2k!} A^{2k} - \sum_{k=1}^{\infty} \frac{(-1)^k}{(2k+1)!} A^{(2k+1)!}$.

Because $A = \begin{bmatrix} O & B \\ B^T & O \end{bmatrix}$, then $A^{2k} = \begin{bmatrix} (BB^T)^k & O \\ O & (B^TB)^k \end{bmatrix}$

and $A^{2k+1} = \begin{bmatrix} O & (BB^T)^k B \\ (B^TB)^k B^T & O \end{bmatrix}$

$$G_{pq} = e^{-A} = \sum_{k=0}^{\infty} \frac{(-1)^k}{2k!} \begin{bmatrix} (BB^T)^k & O \\ O & (B^TB)^k \end{bmatrix} - \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} \begin{bmatrix} O & (BB^T)^k B \\ (B^TB)^k B^T & O \end{bmatrix}.$$

$\Rightarrow G_{pq} = \begin{bmatrix} F & -C \\ -C^T & T \end{bmatrix}$

where F, T and C are greater than zero. The sign of G_{pq} determines whether the corresponding pair of nodes are in the same partition or not. That is, $G_{pq} > 0$ if and only if p and q are in the same partition. From the anti communicability matrix $G = e^{-A}$.

We define

$$\hat{G}_{pq} = \begin{cases} 1 & G_{pq} > 0 \\ 0 & G_{pq} \leq 0 \end{cases}$$

$\Rightarrow \hat{G}_{pq} = \begin{bmatrix} E & O \\ O & E^T \end{bmatrix}$, where E, E^T are matrices of ones.

The anti-communicability graph with adjacency matrix G is the disconnected network.

5. We can use the fact that the eigenvectors of L and L^c are the same and we can show that there a simple relationship between the eigenvalues of L and L^c and show the Fiedler vector of the complement finds the bipartition. Suppose that A is the adjacency matrix of G. So,

$$L^c = D^c - A^c = (n-1)I - D - (E - A - I) = nI - L - E$$

We know that $Le = 0$. So, $L^c e = nIe - Le - Ee = ne - 0 - ne = 0$. Now suppose that $Lx = \lambda x$ then

$$L^c x = nx - Lx - Ex = (n-\lambda)x - Ex = (n-\lambda)x - ee^T x = (n-\lambda)x$$

($e^T x = 0$ because L is symmetric so its eigenvectors are orthogonal $e^T x = 0$) $\Rightarrow L$ and L^c have the same eigenvectors and $\sigma(L) = \{0, \lambda_2, \ldots, \lambda_n\}$ and $0 < \lambda_2 \leq \cdots \leq \lambda_n$, $\sigma(L^c) = \{0, n-\lambda_2, \ldots, n-\lambda_n\}$

If λ_2 is the smallest non-zero eigenvalue of L, then the $n - \lambda_2$ the biggest eigenvalue of L^c and the eigenvector of λ_2 is the Fiedler vector corresponding to the biggest eigenvector of the biggest eigenvalue of L^c. So, the biggest eigenvector of L^c partitions the graph into two sets.

6. Suppose G has bipartition $U \cup V$, where $U = \{1, \ldots, r\}$, $V = \{r+1, \ldots, n\}$ and $\mid E \mid = m$. We can write a signed incidence matrix for G as $[B_1 \;\; B_2]$ where $B_1 \in \mathbb{R}^{m\times r}$, $B_2 \in \mathbb{R}^{m\times(n-r)}$ with $B_1 \geq 0$ and $B_2 \leq 0$. Recall that $L = B^T B$.

   Now $BB^T = B_1 B_1^T + B_2 B_2^T \geq 0$ and so, since it is irreducible by Perron-–Frobenius theorem its dominant eigenvector y is positive (let its eigenvalue be λ). Now clearly $x = B^T y$ (because $B^T(BB^T y) = B^T(\lambda y) = \lambda(B^T y) \Rightarrow B^T B x = \lambda x$) is the eigenvector of the dominant eigenvalue of L (since BB^T and $B^T B$ share the same non-zero eigenvalues) and

$$x = \begin{bmatrix} B_1^T y \\ B_2^T y \end{bmatrix}$$

   so the first r entries of x are positive and the next $n - r$ are negative, as required.

□

5.2 Methods for finding bipartitions

Closed walks, eigenvalues and normalizing the sum of closed walks give measures that quantify bipartivity using the properties of the adjacency matrix and its exponential. The same matrices will now be used in the hope of finding a good partition. We will also use equivalent or similar algorithm to using the graph Laplacian which has long been used to detect communities.

Algorithm 1:

We just find in this algorithm the eigenvectors associated with the most negative eigenvalue of the adjacency matrix and then we partition the graph according to the positive and negative elements of the eigenvector. We know that in bipartite network the most negative eigenvalue will split the networks in two communities.

Algorithm 2:

In this algorithm we use the communicability function $G = \exp(-A)$, then define

$$\hat{G} = \begin{cases} 1 \quad \text{if} \;\; G_{pq} > 0 \\ 0 \quad \text{if} \quad G_{pq} < 0 \;\; \text{or} \;\; p = q \end{cases},$$

after that, we find the eigenvectors associated with the second smallest eigenvalue of the entries of the Laplacian of $\hat{G}$, then we partition the graph according to the positive and negative element of the eigenvector.

Algorithm 3:

In this algorithm we use the signless Laplacian and find the eigenvectors associated with the smallest eigenvalue of the entries of the signless Laplacian, then we partition the graph according to the positive and negative element of the eigenvector.

Algorithm 4:

In this algorithm we use the normalised Laplacian and find the eigenvectors associated with the biggest eigenvalue of the entries of the normalised Laplacian, then we partition the graph according to the positive and negative element of the eigenvector.

Algorithm 5:

In this algorithm we use the Laplacian of complement graphs and find the eigenvectors associated with second smallest eigenvalue of the entries of the Laplacian of the complement, then we partition the graph according to the positive and negative element of the eigenvector.

Algorithm 6:

We use the algorithm of Newman in the paper [63].

5.2.1 Experiment

In this experiment, we are trying to partition the nodes to minimise the number of edges within each partition. So, we have used the algorithm 1, algorithm 2, algorithm 3 and algorithm 4 for computing an approximate bipartition. We haven't used algorithm 5 and algorithm 6 just because they give similar results. We select 20 real world networks (Canton, Centrality, ColoSpg, Corporate people, Dolphins, Drugs, ElVerde, Electronic1, Electronic2, Electronic3, Little Rock, Neurons C elegans, PIN E coli, PIN Malaria, PIN H pylori, Roget Thesaurus, Scotch Broom, Small World Citations, Software Digital and Stony), and used the 4 algorithms to find the bipartition.

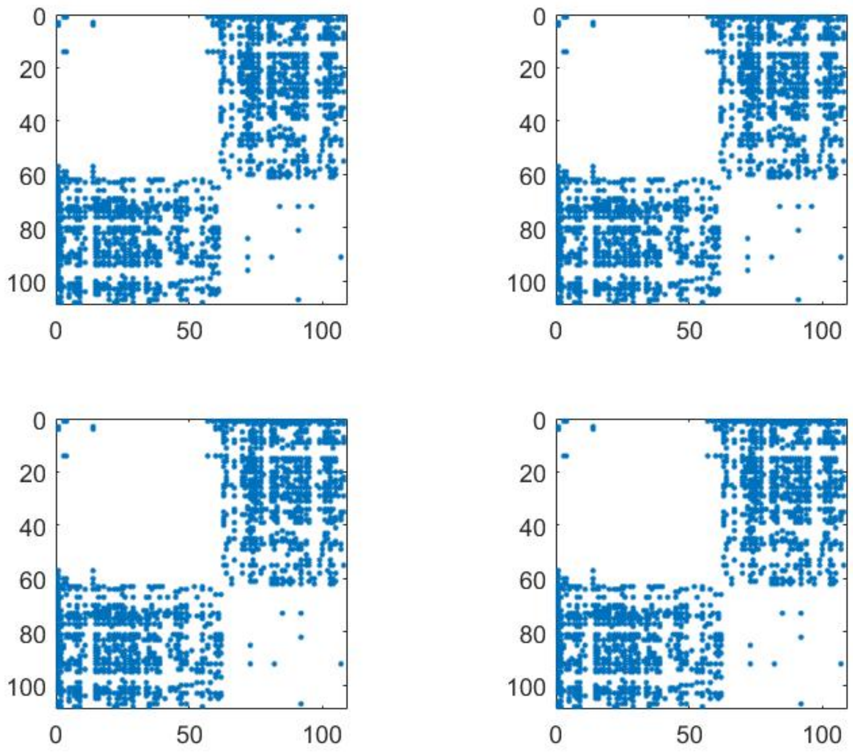


Figure 5.1: Canton.

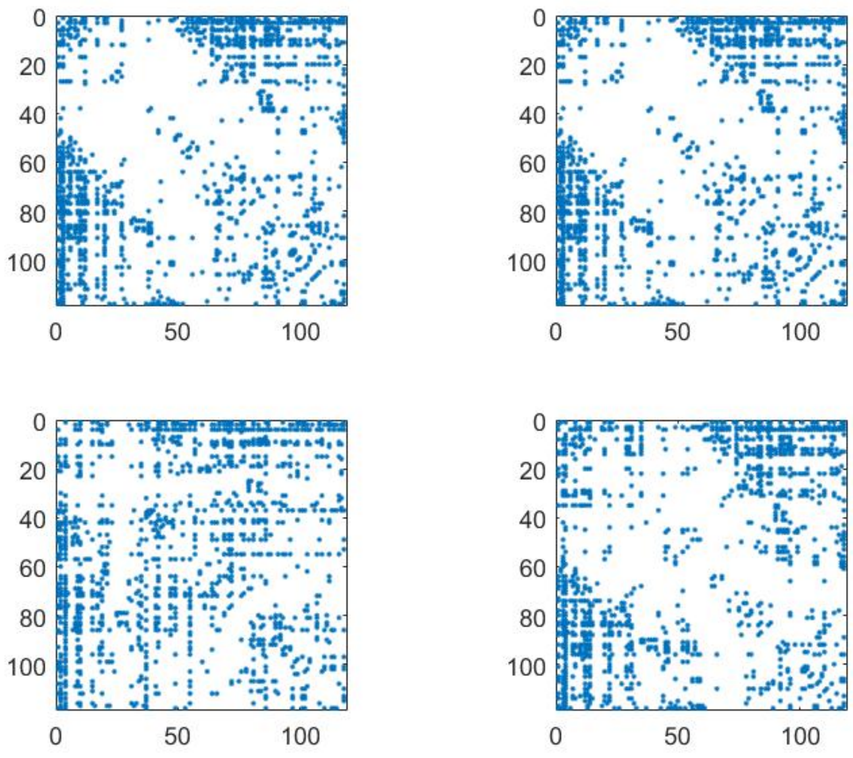


Figure 5.2: Centrality.

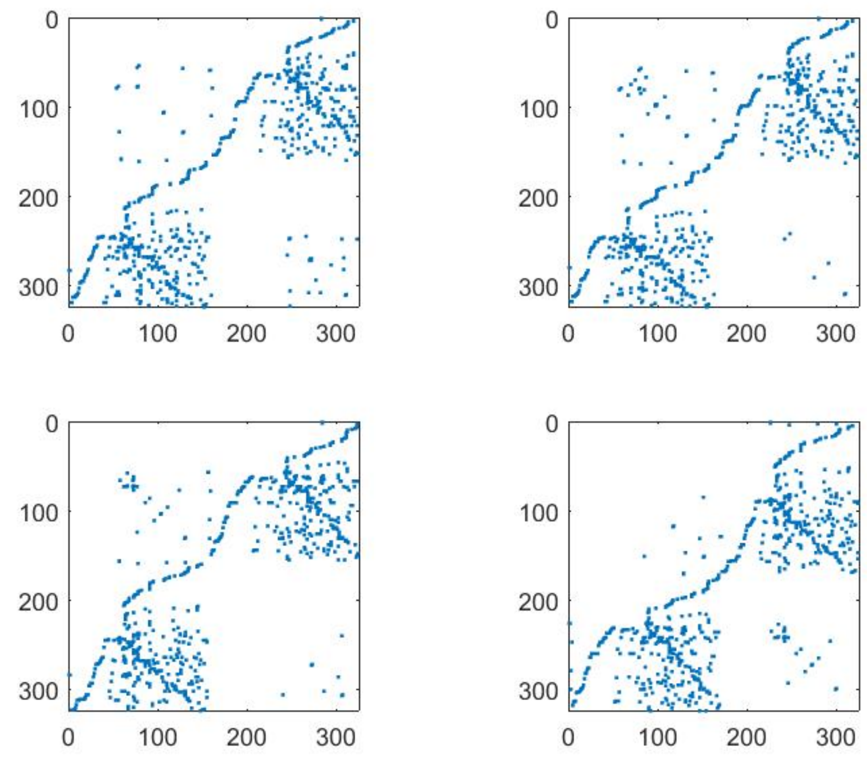


Figure 5.3: ColoSpg.

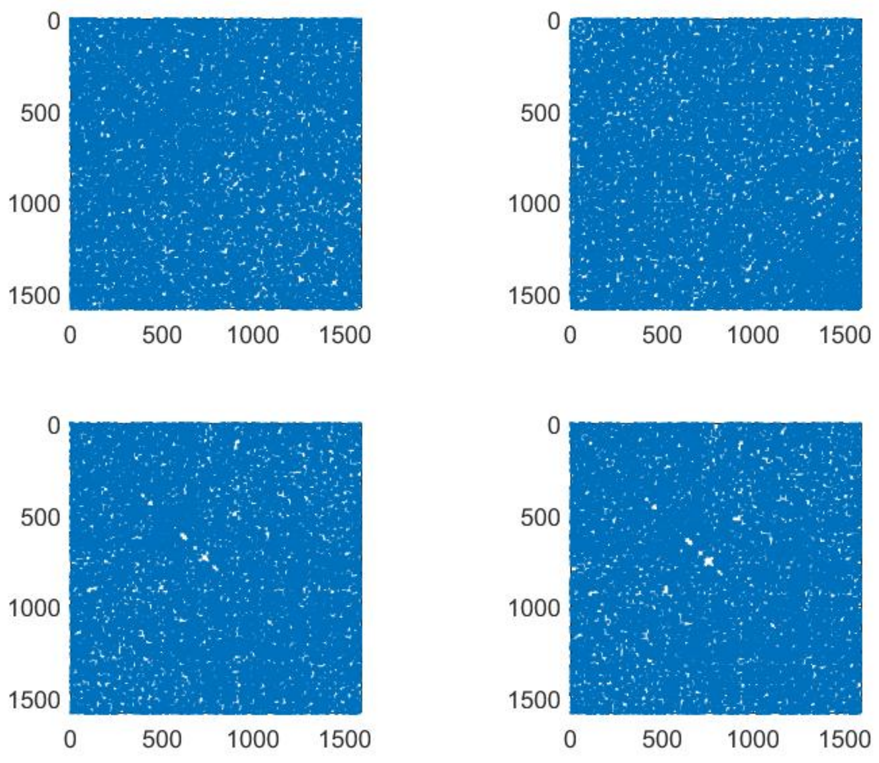


Figure 5.4: Corporate people.

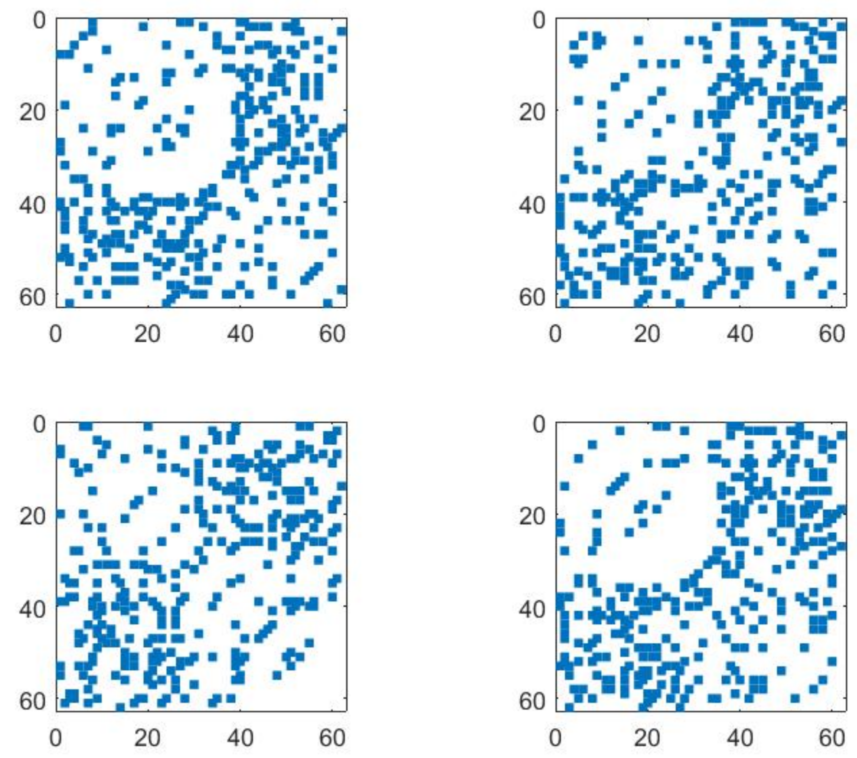


Figure 5.5: Dolphins.

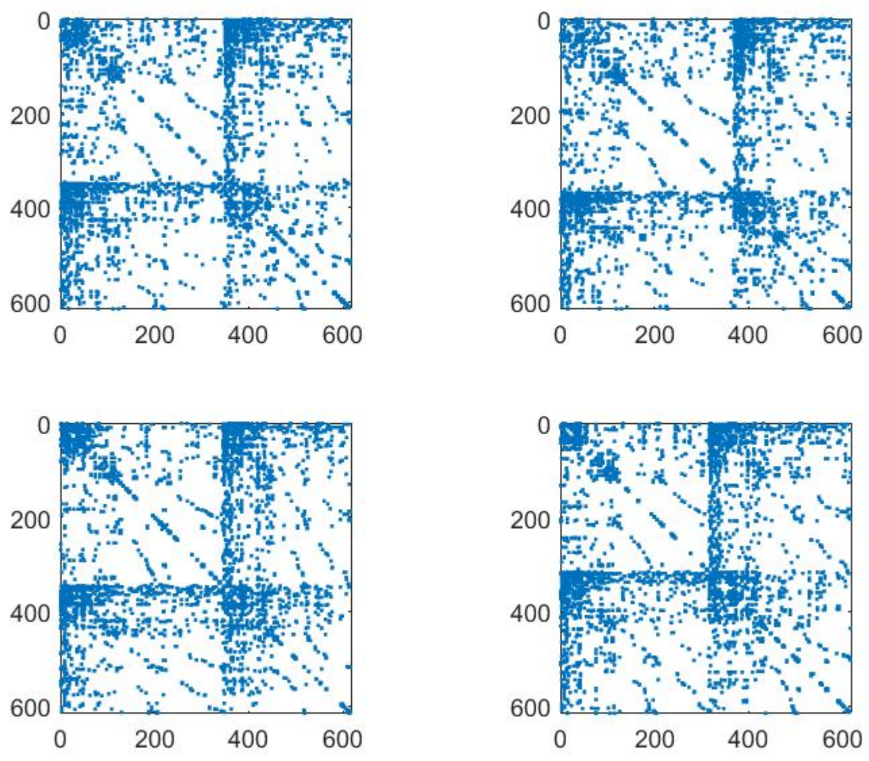


Figure 5.6: Drugs.

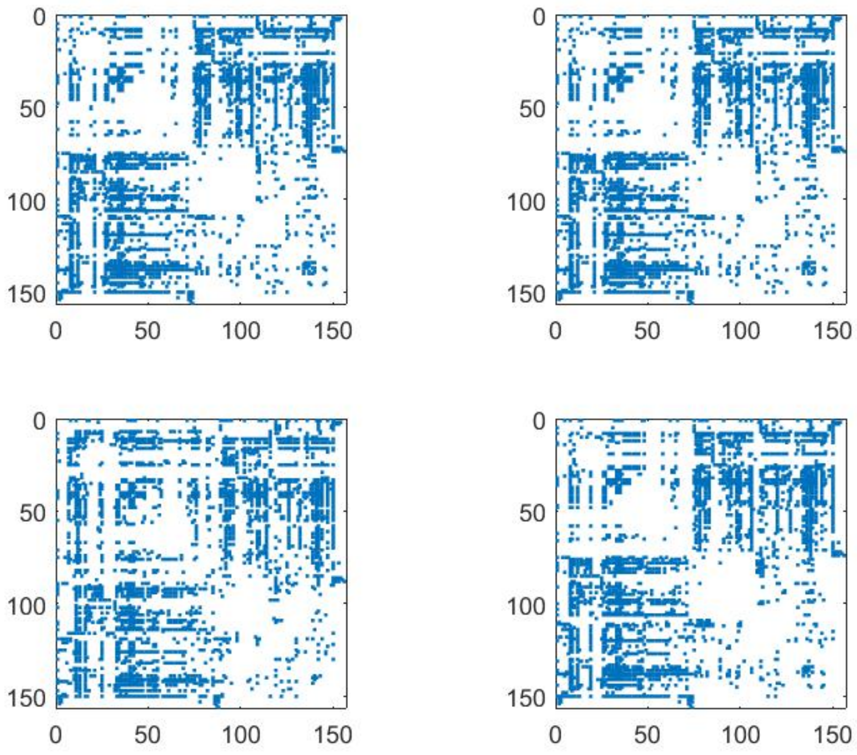


Figure 5.7: ElVerde.

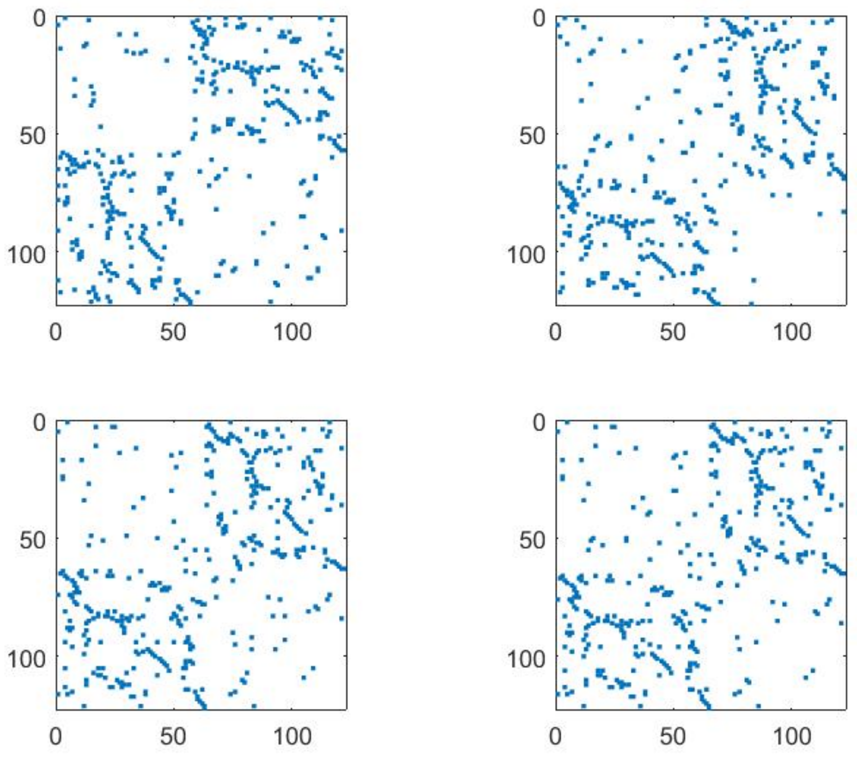


Figure 5.8: Electronic1.

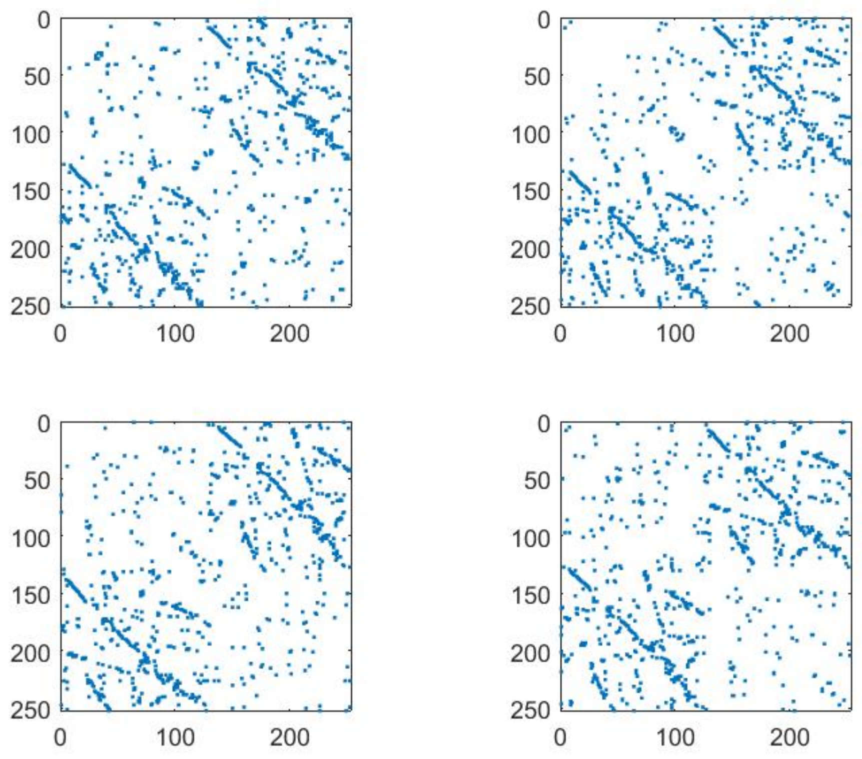


Figure 5.9: Electronic2.

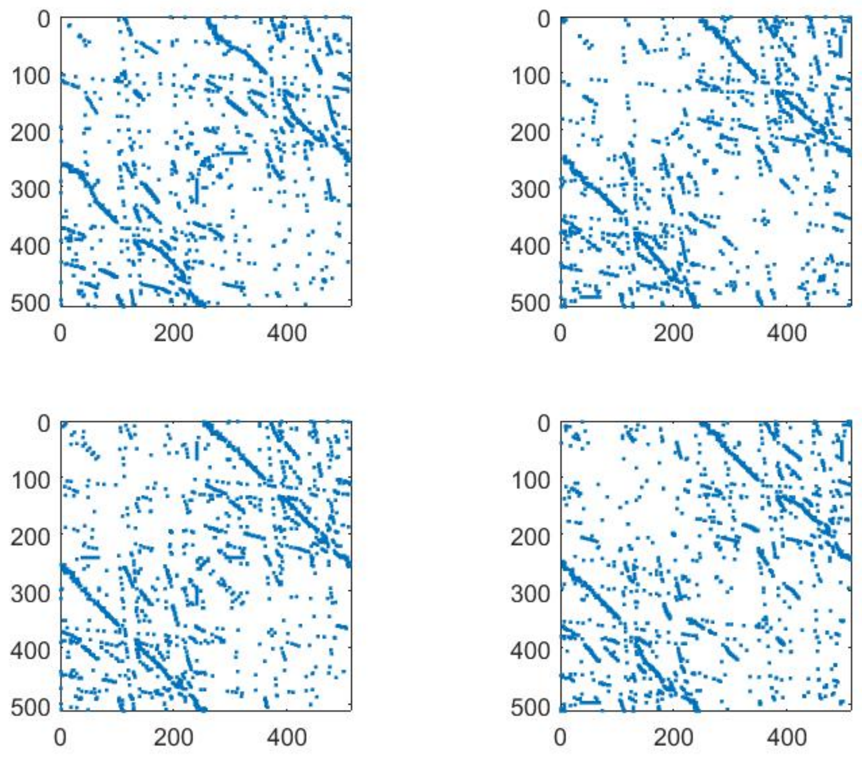


Figure 5.10: Electronic3.

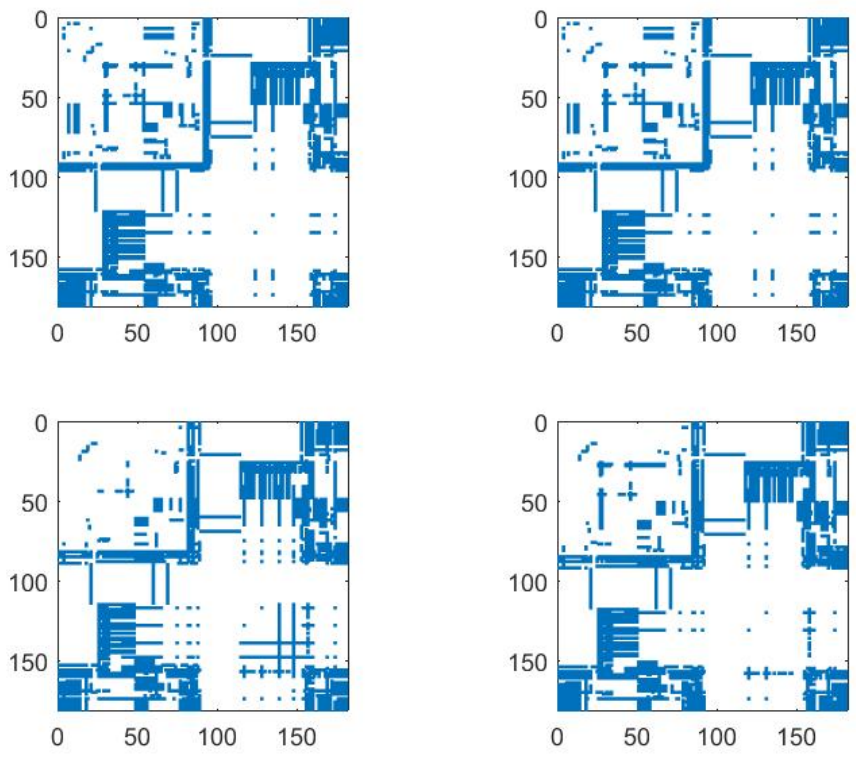


Figure 5.11: Little Rock.

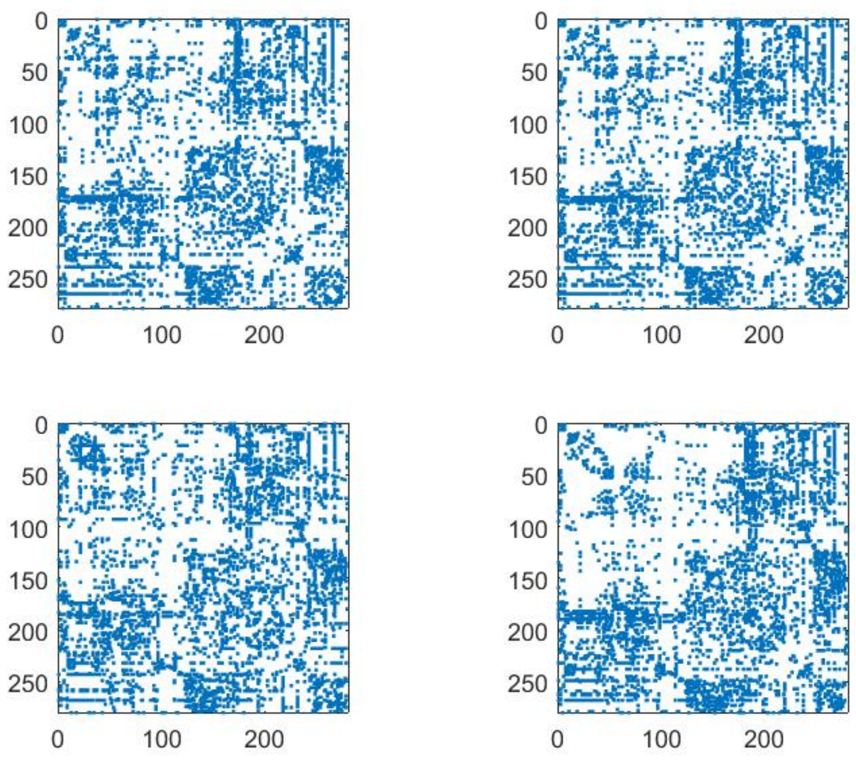


Figure 5.12: Neurons C elegans.

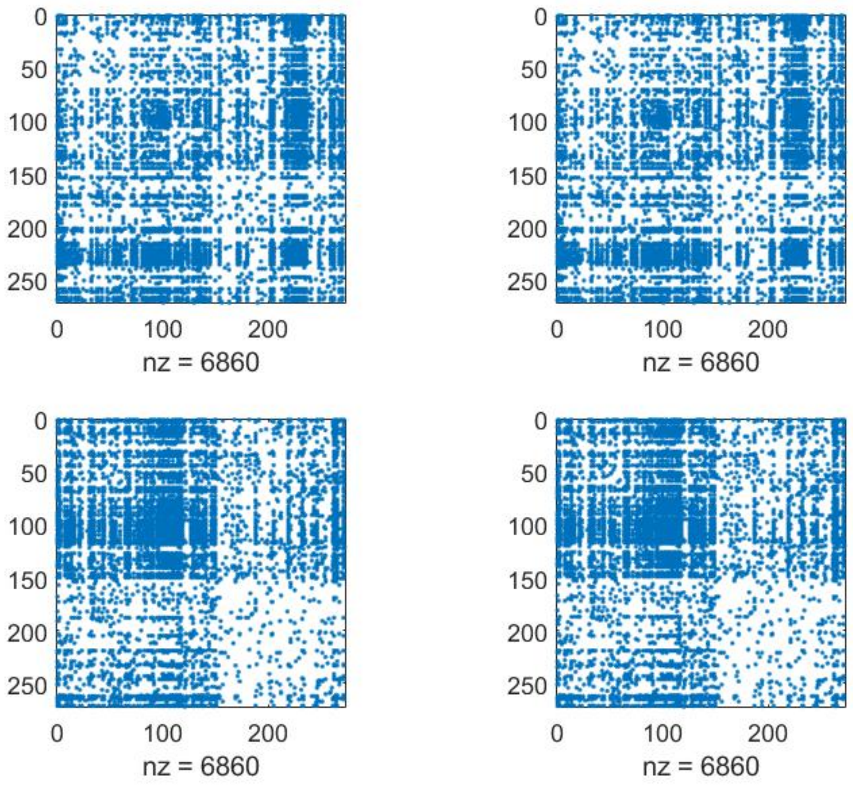


Figure 5.13: PIN E coli.

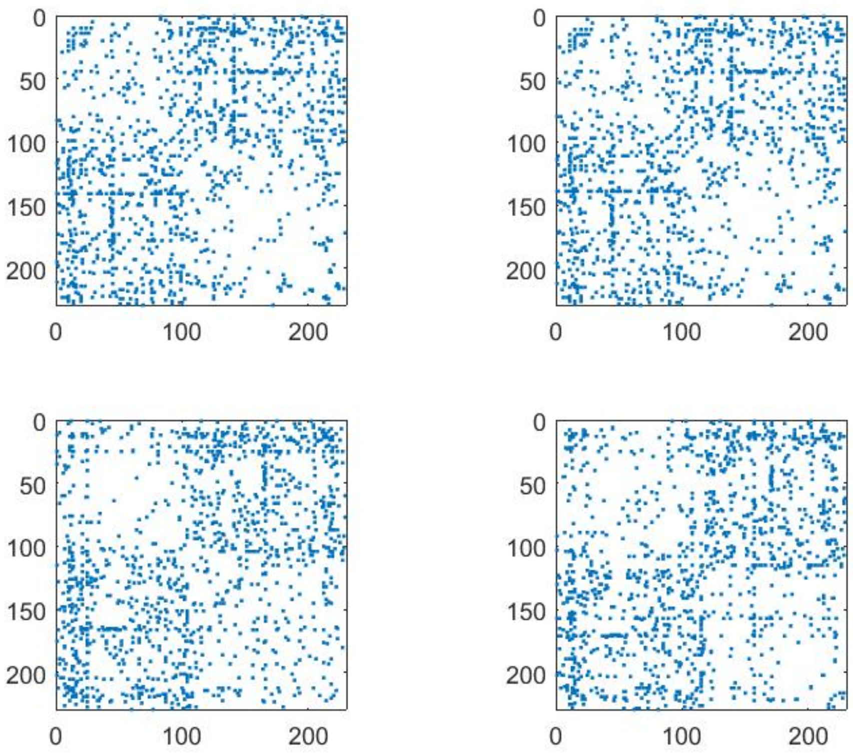


Figure 5.14: PIN Malaria.

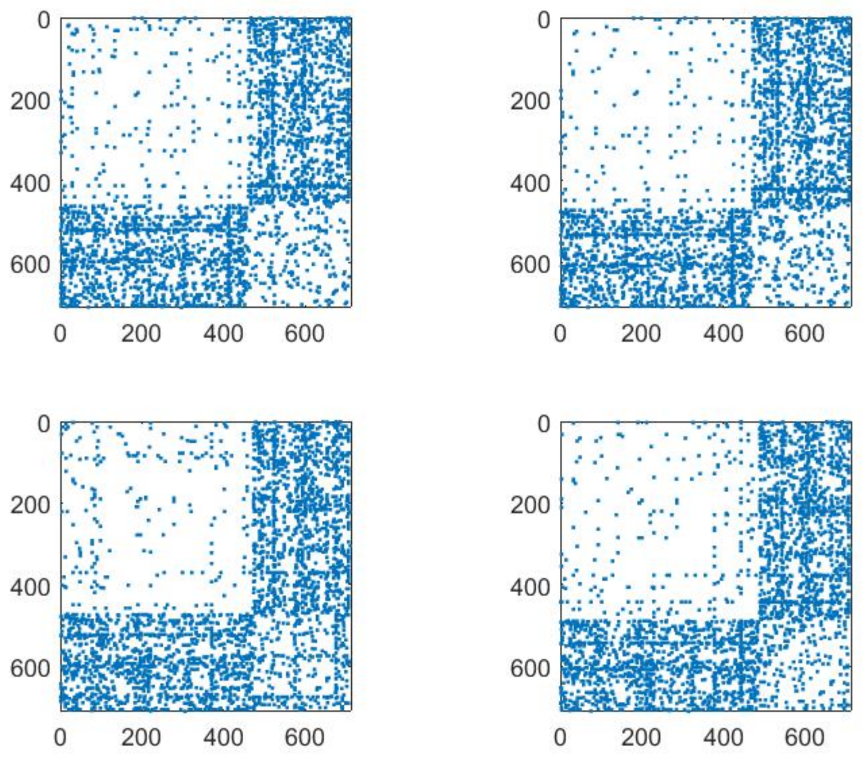


Figure 5.15: PIN H pylori.

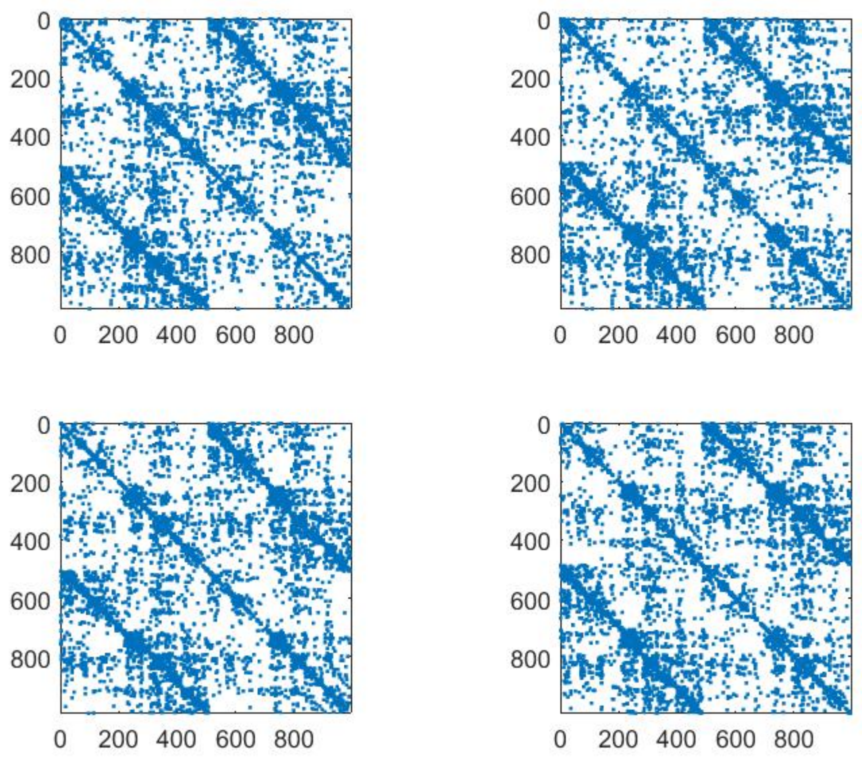


Figure 5.16: Roget Thesaurus.

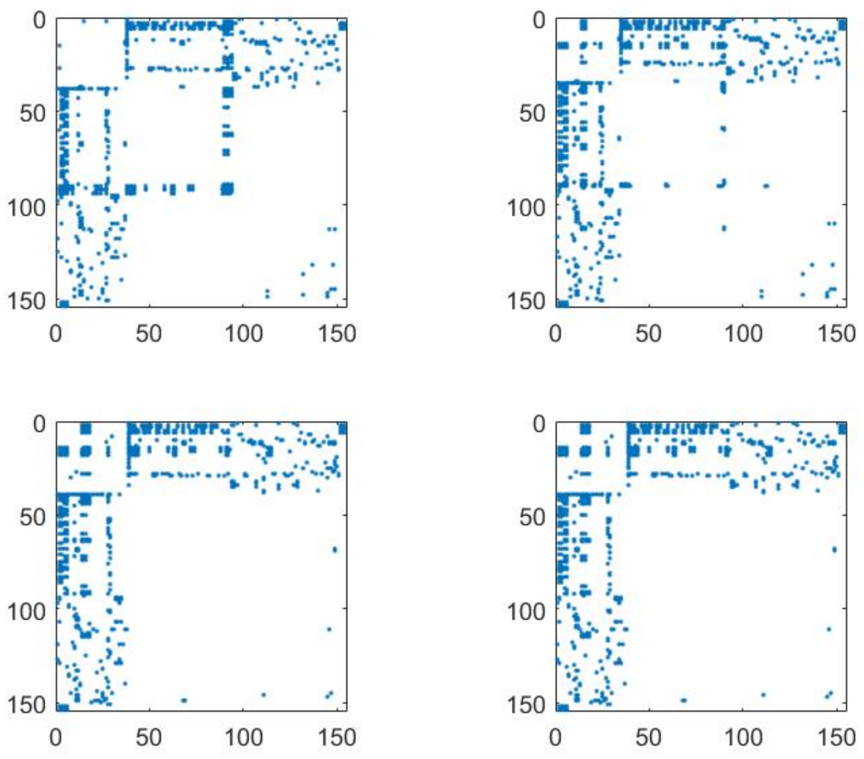


Figure 5.17: Scotch Broom.

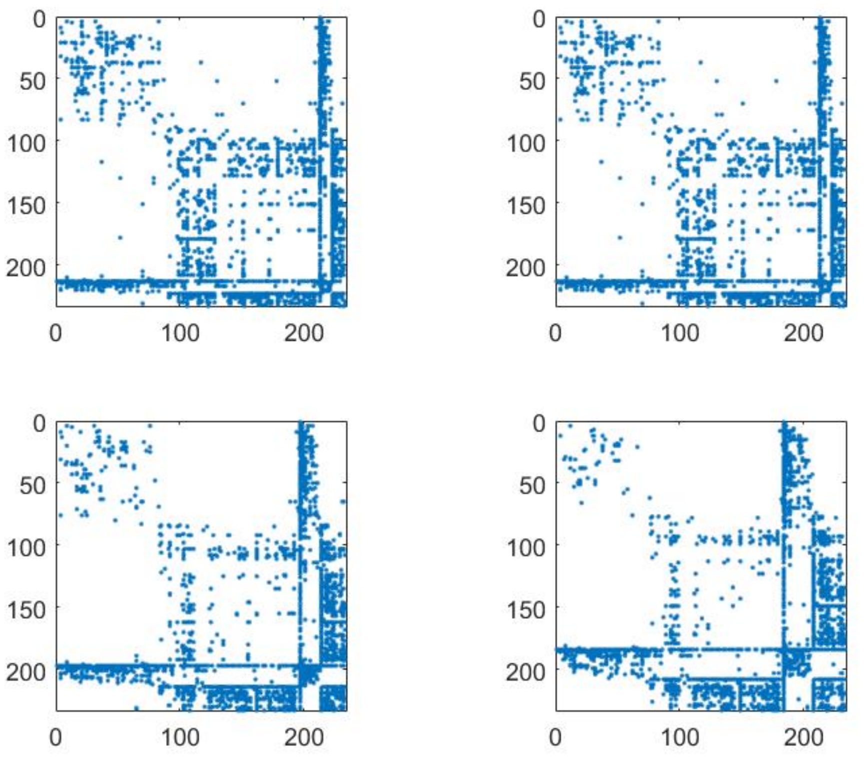


Figure 5.18: Small World Citations.

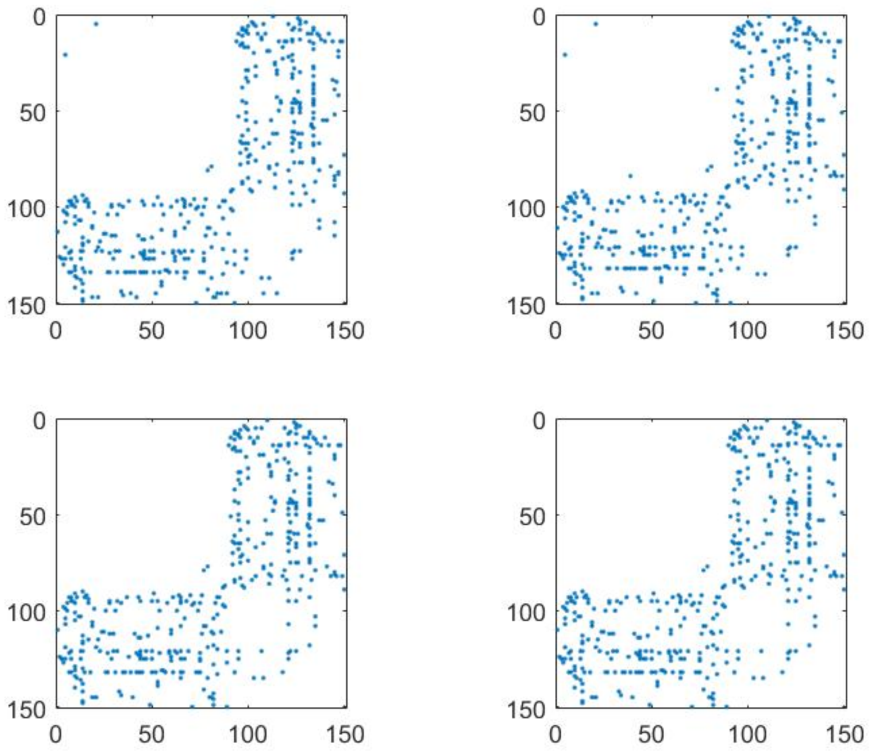


Figure 5.19: Software Digital.

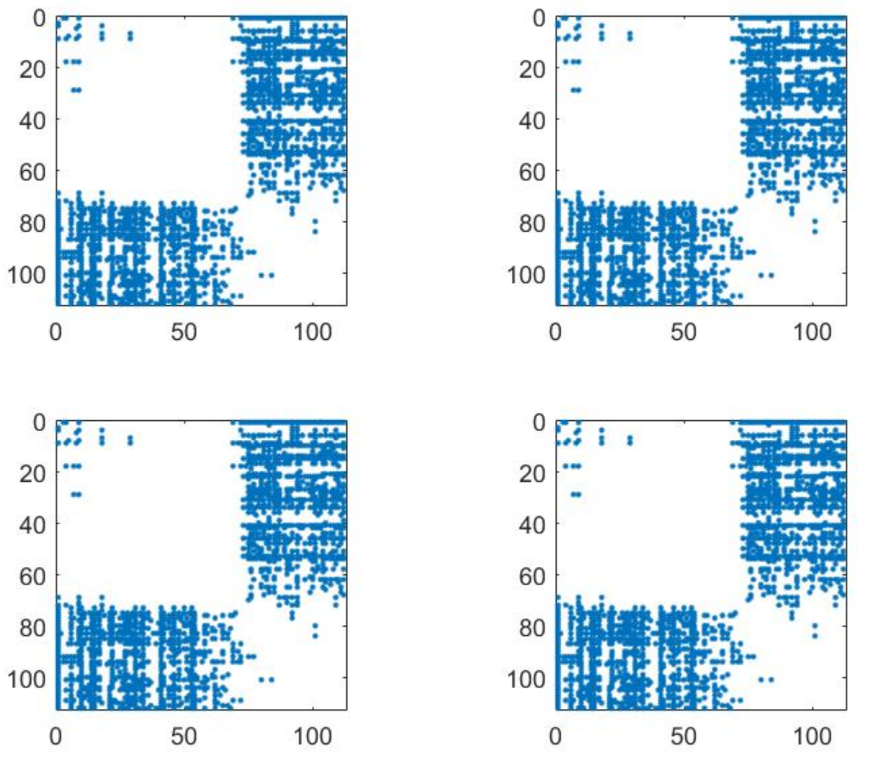


Figure 5.20: Stony.

So, we see from the Figures 5.1 - 5.20 that the results look similar in many cases, but when we focus there are differences between them. We show this in Tables 5.1-5.2. Their effect is almost equal, but with the difference in speed, we find that the algorithm 2 is the slowest and algorithm 3 is the fastest method from Table 5.1.

	1	2	3	4	5	6
canton	0.203	0.453	0.016	0.156	0.063	0.344
Centrality_literature	0.25	0.453	0.016	0.141	0.047	0.391
ColoSpg	0.641	0.953	0.063	0.234	0.25	0.406
Corporate_people	5.422	22.13	2.594	11.48	4.594	5.813
Dolphins	0.422	0.563	0.047	0.047	0.094	0.422
Drugs	1.688	2.766	0.453	1.016	0.516	1.047
El Verde	0.656	0.734	0.063	0.125	0.25	0.469
Electronic1	0.656	0.484	0.031	0.109	0.094	0.281
Electronic2	0.672	0.75	0.109	0.281	0.219	0.563
Elecrtonic3	0.922	1.625	0.234	0.797	0.234	0.688
Little Rock	0.5	0.719	0.063	0.109	0.203	0.594
Neurons C elegans	0.469	0.922	0.094	0.25	0.328	0.375
PIN E coli	0.266	0.688	0.063	0.172	0.234	0.172
PIN Malaria	0.469	0.734	0.156	0.219	0.188	0.313
PIN_H pylori	1.203	2.641	0.594	1.5	0.641	0.844
Roget Thesaurus	2.672	7.094	1	2.859	1.953	1.688
Scotch Broom	0.656	0.453	0.063	0.188	0.188	0.406
Small World Citations	0.422	0.688	0.078	0.234	0.203	0.391
Software Digital	0.406	0.703	0.125	0.063	0.063	0.438
Stony	0.484	0.563	0.016	0.047	0.047	0.328
Total	19.08	46 .1	5.88	20	10.4	15.97

Table 5.1: The speed of the algorithms in seconds.

We compute the modularity of partitions to distinguish between the methods. We see from Tables 5.2 and 5.3 that the method using the normalised Laplacian is the best and the method using the Laplacian of complement graphs never do the best.

We have in Tables 5.2 and 5.3 (7) columns and we put the smallest value in each row in bold.:

Column n is the modularity after using algorithm n. The last column is the best modularity in each row which in this case is the lowest.

	1	2	3	4	5	6	7
canton	**-0.4732**	**-0.4732**	**-0.4732**	**-0.4732**	-0.4357	**-0.4732**	**-0.4732**
Centrality_literature	-0.1938	-0.1938	-0.1295	**-0.2046**	-0.0488	-0.0231	**-0.2046**
ColoSpg	-0.4483	-0.4513	**-0.457**	**-0.457**	-0.4481	-0.1634	**-0.457**
Corporate_people	**0.00211**	0.01259	0.0381	0.03368	0.0885	0.0056	**0.00211**
Dolphins	-0.2305	-0.2023	-0.2426	**-0.2513**	-0.1923	-0.1251	**-0.2513**
Drugs	**-0.0965**	-0.0684	-0.0837	-0.0836	0.0219	-0.016	**-0.0965**
El Verde	**-0.3125**	**-0.3125**	-0.2123	-0.3102	-0.0429	**-0.3125**	**-0.3125**
Electronic1	-0.3371	-0.3375	**-0.3466**	-0.3422	-0.3109	-0.2798	**-0.3466**
Electronic2	-0.3073	-0.3096	**-0.3171**	-0.3121	-0.297	-0.1645	**-0.3171**
Electronic3	-0.2975	-0.2971	-0.3145	**-0.3181**	-0.2719	-0.0231	**-0.3181**
Little Rock	-0.289	-0.289	-0.2829	-0.2901	-0.0575	**-0.2903**	**-0.2903**
Neurons C elegans	-0.1416	-0.1432	-0.116	**-0.1695**	-0.0159	-0.1074	**-0.1695**
PIN E coli	**-0.1175**	**-0.1175**	-0.0132	-0.0105	0.0497	-0.0952	**-0.1175**
PIN Malaria	-0.2137	-0.2006	-0.2204	**-0.2398**	-0.1826	-0.2122	**-0.2398**
PIN_H pylori	-0.3653	**-0.3741**	-0.3237	-0.3625	-0.2445	-0.19	**-0.3741**
Roget Thesaurus	**-0.1714**	-0.1714	-0.1436	-0.1462	-0.1352	-0.1709	**-0.1714**
Scotch Broom	-0.3254	-0.3744	**-0.392**	**-0.392**	-0.3254	-0.2468	**-0.392**
Small World Citations	-0.1514	-0.1514	-0.2176	**-0.25**	-0.081	-0.1634	**-0.25**
Software Digital	-0.4399	-0.4446	**-0.4451**	**-0.4451**	-0.395	-0.1488	**-0.4451**
Stony	**-0.4725**	**-0.4725**	**-0.4725**	**-0.4725**	-0.4711	**-0.4725**	**-0.4725**

Table 5.2: The modularity before using the local improvement algorithm.

	1	2	3	4	5	6	7
Transcription E coli	-0.4148	-0.4148	**-0.443**	-0.4342	-0.3934	-0.1835	**-0.443**
GD	**-0.2371**	-0.2048	-0.1949	-0.1967	-0.1584	-0.1829	**-0.2371**
Transcription Yeast	-0.413	-0.4322	**-0.4567**	-0.4567	-0.4033	-0.1827	**-0.4567**
Software MySQL	-0.2126	-0.2148	-0.2542	**-0.2549**	-0.1438	-0.0375	**-0.2549**
USAir 97	**-0.135**	-0.132	-0.0699	-0.0615	0.03054	-0.0931	**-0.135**
Software VTK	-0.293	-0.293	-0.3136	**-0.3136**	-0.26	-0.112	**-0.3136**
Ythan 1	-0.3004	-0.3004	-0.3034	**-0.317**	-0.1006	-0.298	**-0.317**
Software XMMS	-0.3259	-0.3291	**-0.3657**	**-0.3657**	-0.3202	-0.0731	**-0.3657**
termite Mound 3	-0.3055	-0.2918	-0.3125	**-0.3146**	-0.2922	-0.1296	**-0.3146**
Software_Abi	-0.3316	-0.3599	-0.3686	**-0.371**	-0.2958	-0.1013	**-0.371**
Termite Mound 1	**-0.3883**	-0.3596	-0.3698	-0.3787	-0.3774	-0.2463	**-0.3883**
Termite Mound 2	-0.4607	-0.4536	**-0.4679**	**-0.4679**	-0.4572	-0.1215	**-0.4679**

Table 5.3: The modularity before using the local improvement algorithm.

5.3 Local improvement with modularity

We use the idea of a local improvement method mentioned in the context of communities in 2.2.1 to test the effectiveness of various algorithms for computing the nearest bipartite graph. So, we use the local improvement method to improve the various algorithms for computing the nearest bipartite graph. Then we compute the modularity after partitioning the network using various algorithms before and after using the local improvement method. In local improvement method we are just looking at ways of moving two nodes between partition to reduce the modularity to the fact that the modularity of bipartite network is -0.5.

5.3.1 Local improvement algorithm

We use the local improvement algorithm to reduce modularity and find the biggest improvement in modularity at each step. The local improvement algorithm give us easy way to compute the effect of modularity. First we input the adjacency matrix A, the current partition $\{p, q\}$. Then we go through each nodes in the network and see if it is in the partition p or in the partition q. we start with node 1 and start with main loop. Suppose we partition

$$A = \begin{bmatrix} A_{pp} & A_{pq} \\ A_{qp} & A_{qq} \end{bmatrix},$$

into $p = \{1, 2, \ldots, n\}$ and $q = \{n+1, \ldots, n+r\}$. Then swap node 1 from p to q. Then we compute the change in modularity (ΔQ) which can be computed analytically in the following steps:

For clarity, let $\mathbf{e}^{(l)}$ be a vector of ones of length l, and $\mathbf{e}_l$ be the lth column of I. Then

$$Q_{\text{old}} = \frac{\mathbf{s}^T B \mathbf{s}}{4m} \text{ and } Q_{\text{new}} = \frac{\bar{\mathbf{s}}^T B \bar{\mathbf{s}}}{4m}$$

where

$$B = A - \frac{\mathbf{k}\mathbf{k}^T}{2m}, \quad \mathbf{s} = \begin{bmatrix} \mathbf{e}^{(n)} \\ -\mathbf{e}^{(r)} \end{bmatrix} \text{ and } \bar{\mathbf{s}} = \mathbf{s} - 2\mathbf{e}_1 \ , \mathbf{k} = A\mathbf{e}.$$

So,

$$Q_{\text{new}} = \frac{1}{4m}(\mathbf{s} - 2\mathbf{e}_1)^T B(\mathbf{s} - 2\mathbf{e}_1) = Q_{\text{old}} + \frac{1}{m}(\mathbf{e}_1^T B \mathbf{e}_1 - \mathbf{s}^T B \mathbf{e}_1).$$

Now $\mathbf{e}_1^T B \mathbf{e}_1 = a_{11} - k_1^2/(2m) = -k_1^2/(2m)$ and since

$$B\mathbf{e}^{(n+r)} = A\mathbf{e}^{(n+r)} - \frac{\mathbf{k}\mathbf{k}^T}{2m}\mathbf{e}^{(n+r)} = \mathbf{k} - \frac{2m}{2m}\mathbf{k} = 0,$$

we have

$$-\frac{1}{m}\mathbf{s}^T B \mathbf{e}_1 = \frac{2}{m}\begin{bmatrix} 0 \\ \mathbf{e}^{(r)} \end{bmatrix}^T B\mathbf{e}_1 = \frac{2}{m}\mathbf{e}_1^T \left[A - \frac{\mathbf{k}\mathbf{k}^T}{2m}\right] \begin{bmatrix} 0 \\ \mathbf{e}^{(r)} \end{bmatrix} = \frac{2}{m}k_1^q - \frac{k_1 \mathbf{k}^T}{m^2} \begin{bmatrix} 0 \\ \mathbf{e}^{(r)} \end{bmatrix} = \frac{2}{m}k_1^q - \frac{k_1 \hat{k}_q}{m^2}.$$

And so the change in modularity is given by

$$\Delta Q = \frac{-k_1^2}{2m^2} - \frac{k_1 \hat{k}_q}{m^2} + \frac{2k_1^q}{m},$$

where k_1^q is the number of edges from node 1 that end in partition q and $\hat{k}_q$ is the sum of the degrees of the nodes in partition q. W

The equivalent change can be calculated for every node very quickly and we swap the node which gives the minimum value of ΔQ (so long as it is negative). We can repeat this process until we reduce Q no further.

5.3.2 Example

In this example, we will apply the local improvement algorithm to a network with 8 nodes and 4 frustrated edges. We are going to work out the number of frustrated edges. So,initially we partition the network as in Figure 5.21 with partition $p = \begin{bmatrix} 1 & 3 & 5 & 7 \end{bmatrix}$ and partition $q = \begin{bmatrix} 2 & 4 & 6 & 8 \end{bmatrix}$. There 4 frustrated edges and the measurement of the modularity before starting the local improvement algorithm gives a value of -0.1 .
So, the principle of the local improvement algorithm is to find the best improvement we can get of modularity by moving one node from one side to another.

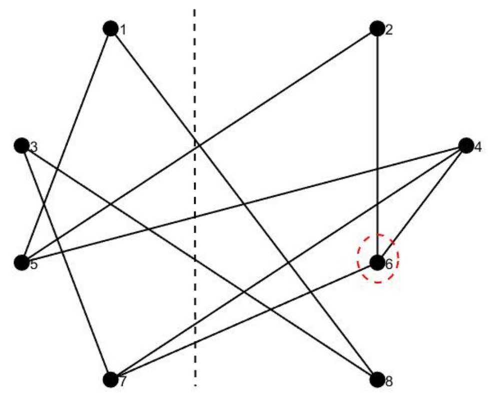


Figure 5.21: The network before start the algorithm.

Now we swap the node 6 from partition q to partition p. So the partition now is $p = \begin{bmatrix} 1 & 3 & 5 & 7 & 6 \end{bmatrix}, q = \begin{bmatrix} 2 & 4 & 8 \end{bmatrix}$. Because the $\mathbf{v} = \begin{bmatrix} -0.0200 & -0.0200 & -0.0200 & 0.0550 & 0.0550 & -0.1450 & 0.0550 & 0.1800 \end{bmatrix}$, where $\mathbf{v}$ is the change in modularity. The minimum number is -0.1450 which in node 6 and the modularity in this step is -0.2450 and we reduce the number of frustrated edges to 3 as in Figure 5.22, then we repeat the process again.

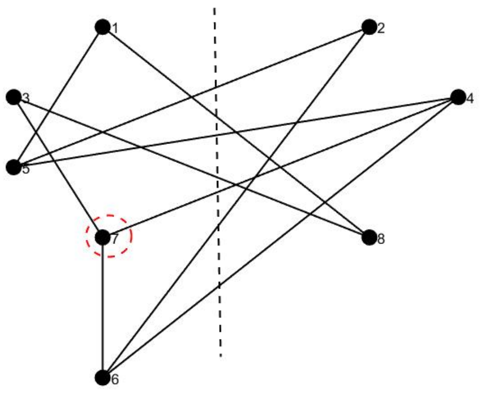


Figure 5.22: Step 1.

In this step we get the $\mathbf{v} = \begin{bmatrix} 0.0400 & 0.1200 & 0.0400 & 0.1650 & 0.1450 & 0.1450 & -0.0550 & 0.1200 \end{bmatrix}$ and the minimum number is -0.0550 which in node 7. So, we swap the node 7 from partition p to partition q. So the partition now is $p = \begin{bmatrix} 1 & 3 & 5 & 6 \end{bmatrix}, q = \begin{bmatrix} 2 & 4 & 8 & 7 \end{bmatrix}$ and the modularity in this step is -0.3000 and we reduce the number of frustrated edges to 2 as in Figure 5.23, then we repeat the process again.

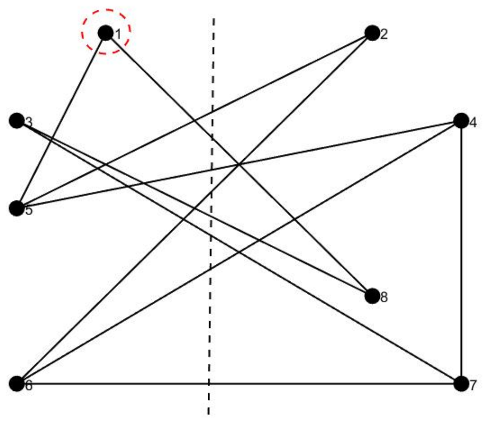


Figure 5.23: Step 2.

In this step we get the $\mathbf{v} = \begin{bmatrix} -0.0200 & 0.1800 & 0.1800 & 0.0550 & 0.0550 & 0.2550 & 0.0550 & 0.1800 \end{bmatrix}$ and The minimum number is -0.0200 which in node 1. So, we swap the node 1 from partition p to partition q. So the partition now is $p = \begin{bmatrix} 3 & 5 & 6 \end{bmatrix}, q = \begin{bmatrix} 2 & 4 & 8 & 7 & 1 \end{bmatrix}$ and the modularity in this step is -0.3200 and we stop at this step we got 2 frustrated edges in the end as in Figure 5.24. We show in this step that we may not necessarily reduce the number frustrated of edges at each step

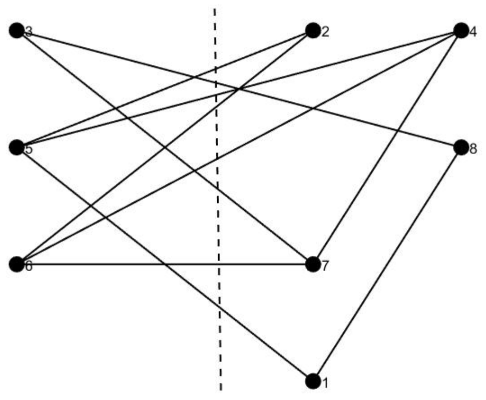


Figure 5.24: Step 3.

5.4 Experiments

In these experiments, we test real world networks with different bipartivity measures and try to find which of the measures best reflects how bipartite a network is.

5.4.1 Experiment 1

In this experiment, we are looking at 32 networks (Canton, Electronic3, Small World Citations, Transcription E coli, Centrality literature, GD, Software Digital, Transcription Yeast, ColoSpg, Little Rock, Software MySQL, USAir 97, Corporate people, Neurons C elegans, Software VTK, Ythan 1, Dolphins, PIN E coli, Software XMMS, termite Mound 3, Drugs, PIN Malaria, Software Abi, El Verde, PIN H pylori, Stony, Electronic1, Roget Thesaurus, Termite Mound 1, Electronic2, Scotch Broom and Termite Mound 2) and testing them with different bipartivity measures.

The following tables show the modularity and bipartivity measures for real networks. We have in Table 5.4 and 5.5 (6) columns:

The column n is the modularity after partitioning the network using the method we use in column n in Tables 5.2 and 5.3 with the local improvement method. The last column is the best modularity in each row.

	1	2	3	4	5	6
canton	**-0.476**	**-0.476**	**-0.476**	**-0.476**	**-0.476**	**-0.476**
Centrality_literature	-0.2232	-0.2232	-0.2118	**-0.2275**	-0.2146	**-0.2275**
ColoSpg	-0.451	-0.454	**-0.4597**	**-0.4597**	-0.451	**-0.4597**
Corporate_people	**-0.0796**	-0.0775	-0.0759	-0.0754	-0.0758	**-0.0796**
Dolphins	-0.2563	-0.2423	-0.2513	**-0.2675**	-0.2312	**-0.2675**
Drugs	**-0.18**	-0.177	-0.1745	-0.172	-0.1715	**-0.18**
El Verde	-0.3136	-0.3136	-0.3136	**-0.3142**	-0.3136	**-0.3142**
Electronic1	-0.3575	-0.3575	**-0.3735**	-0.3677	-0.3521	**-0.3735**
Electronic2	**-0.3647**	-0.3474	-0.3547	-0.3548	-0.3421	**-0.3647**
Electronic3	-0.3414	-0.3547	-0.3585	**-0.3609**	-0.3331	**-0.3609**
Little Rock	**-0.2995**	**-0.2995**	**-0.2995**	**-0.2995**	**-0.2995**	**-0.2995**
Neurons C elegans	-0.1921	-0.1893	**-0.1989**	-0.1934	-0.1559	**-0.1989**
PIN E coli	-0.1225	**-0.1235**	-0.1174	-0.1177	-0.1229	**-0.1235**
PIN Malaria	-0.2505	-0.2562	-0.2583	**-0.2652**	-0.26	**-0.2652**
PIN_H pylori	-0.3775	-0.3782	-0.351	**-0.3791**	-0.2937	**-0.3791**
Roget Thesaurus	**-0.213**	-0.2126	-0.2069	-0.2069	-0.2008	**-0.213**
Scotch Broom	**-0.3951**	**-0.3951**	-0.3945	-0.3945	**-0.3951**	**-0.3951**
Small World Citations	-0.2572	-0.2572	-0.2532	**-0.26**	-0.2304	**-0.26**
Software Digital	-0.4399	-0.4446	**-0.4451**	**-0.4451**	-0.4243	**-0.4451**
Stony	**-0.4735**	**-0.4735**	**-0.4735**	**-0.4735**	-0.4723	**-0.4735**

Table 5.4: The modularity after using the local improvement method.

	1	2	3	4	5	6
Transcription E coli	-0.43694	-0.43694	**-0.44299**	**-0.44299**	-0.42105	**-0.44299**
GD	**-0.26396**	-0.26067	-0.25279	-0.25279	-0.23863	**-0.26396**
Transcription Yeast	-0.43699	-0.4511	**-0.46063**	**-0.4606**	-0.42192	**-0.4606**
Software MySQL	-0.29768	-0.29622	-0.29935	**-0.29935**	-0.26945	**-0.29935**
USAir 97	**-0.16469**	**-0.16469**	-0.16275	-0.1256	-0.16043	**-0.16469**
Software VTK	**-0.33461**	-0.32568	-0.32833	-0.32907	-0.31812	**-0.33461**
Ythan 1	-0.32182	-0.32182	-0.31989	**-0.32207**	-0.31241	**-0.32207**
Software XMMS	-0.36132	-0.36573	**-0.3757**	**-0.3757**	-0.36016	**-0.3757**
termite Mound 3	-0.32932	-0.33076	-0.34211	**-0.34671**	-0.30665	**-0.34671**
Software_Abi	-0.3697	**-0.38835**	-0.38318	-0.38315	-0.35754	**-0.38835**
Termite Mound 1	**-0.4012**	-0.38505	-0.38319	-0.38462	-0.39064	**-0.4012**
Termite Mound 2	-0.46072	-0.45358	**-0.47143**	**-0.47143**	-0.45717	**-0.47143**

Table 5.5: The modularity after using the local improvement method.

When we compare the Tables 5.2 and 5.3 before using local improvement method with the Tables 5.4 and 5.5 after using local improvement method we see that all methods improved and the method 4 is the best.

The following table shows the time taken for each method and in the last row, we see the total of the time taken by every method. We see from the Table 5.6 that the method 3 is the best followed by the method 4.

	1	2	3	4	5
canton	0.406	0.313	0.141	0	0.031
Centrality_literature	0.438	0.344	0.016	0.188	0.047
ColoSpg	0.625	0.516	0.078	0.156	0.25
Corporate_people	18.5	29.25	17.39	21.55	23.8
Dolphins	0.359	0.578	0	0.156	0.047
Drugs	1.688	2.844	1.063	1.234	1.328
El Verde	0.516	0.609	0.063	0.094	0.109
Electronic1	0.359	0.531	0.063	0.078	0.047
Electronic2	0.5	0.719	0.172	0.141	0.156
Elecrtonic3	1.016	1.688	0.469	0.797	0.5
Little Rock	0.516	0.625	0.063	0.156	0.094
Neurons C elegans	0.547	0.578	0.125	0.297	0.219
PIN E coli	0.531	0.641	0.109	0.156	0.406
PIN Malaria	0.438	0.5	0.156	0.266	0.156
PIN_H pylori	1.094	2.125	0.688	1.172	0.984
Roget Thesaurus	3.734	7.25	3.859	4.188	3.906
Scotch Broom	0.5	0.438	0.063	0.063	0.094
Small World Citations	0.438	0.563	0.031	0.203	0.141
Software Digital	0.297	0.375	0.063	0.109	0.063
Stony	0.438	0.406	0.031	0.094	0.063
Total	32.94	50.89	24.64	31.1	32.44

Table 5.6: The speed of the algorithms in seconds.

5.4.2 Experiment 2

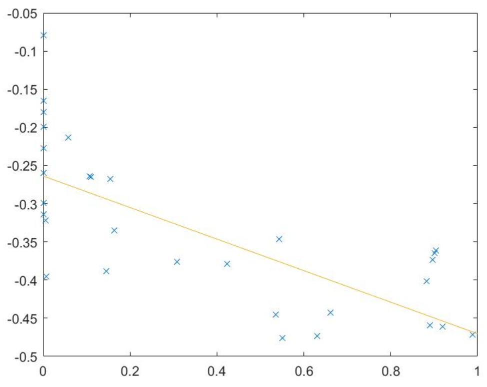
In this experiment, we are looking at 32 networks (Canton, Electronic3, Small World Citations, Transcription E coli, Centrality literature, GD, Software Digital, Transcription Yeast, ColoSpg, Little Rock, Software MySQL, USAir 97, Corporate people, Neurons C elegans, Software VTK, Ythan 1, Dolphins, PIN E coli, Software XMMS, termite Mound 3, Drugs, PIN Malaria, Software Abi, El Verde, PIN H pylori, Stony, Electronic1, Roget Thesaurus, Termite Mound 1, Electronic2, Scotch Broom and Termite Mound 2) and we compute the bipartivity measures $b_c, b_s, b_e, b_{L_N}, b_{L_s}$, LSA, and an estimate of a number of edges. The LSA (Local switching algorithm) [67] starts with randomly permuting the vertices and then partition the set of nodes into two sets X and Y with sizes $\frac{|V|}{2}, \frac{|V|}{2}$, respectively $(X \cap Y = \phi, X \cup Y = V)$. After that, the vertex u is moved from one part to another if $\mid E_u^{\text{int}}(G) \mid > \mid E_u^{\text{ext}}(G) \mid$, or $2\times \mid E_u^{\text{int}}(G) \mid > \mid E_u(G) \mid$ since $\mid E_u(G) \mid = \mid E_u^{\text{int}}(G) \mid + \mid E_u^{\text{ext}}(G) \mid$, vertex u is then marked as moved and not considered for movement in subsequent iterations. The process continues alternately between the two parts until no more movements of vertices are possible. Finally, the ratio $r^b = \frac{|E(G)_{\text{bipart}}|}{|E(G)|}$ is calculated where $\mid E(G)_{\text{bipart}} \mid$ is the number of edges remaining when edge-deleted subgraph of G becomes bipartite and $\mid E(G) \mid$ is the number of edges in G [67].

We use the algorithm which gives the minimum modularity in Table 5.7 then we compute the bipartivity measures and LSA. Also, we find the Correlation Coefficient between the minimum modularity and bipartivity measures in Table 5.9 to see the relationship between these measures.

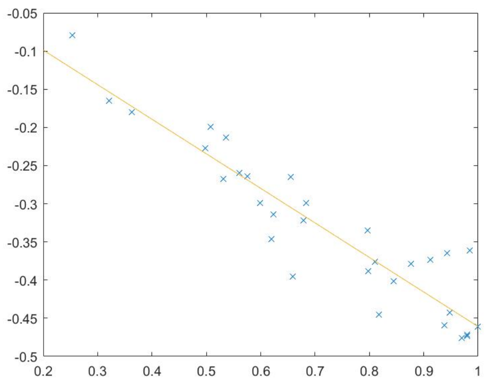
	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA	frustrated E
canton	-0.47596	0.550983	0.969526	0.973126	0.734844	0.970547	0.976	19
Elecrtonic3	-0.36085	0.903425	0.984032	0.797314	0.912128	0.956006	0.8413	166
Small World Citations	-0.26003	0.000118	0.561003	0.597586	0.529584	0.660274	0.7445	400
Transcription E coli	-0.44299	0.661264	0.94794	0.914474	0.959516	0.978644	0.9364	39
Centrality_literature	-0.2275	0.0001	0.498308	0.683524	0.582047	0.613348	0.7194	194
GD	-0.26396	0.106902	0.575269	0.737008	0.895869	0.946116	0.7606	167
Software Digital	-0.44507	0.534761	0.817721	0.939394	0.95286	0.97403	0.9394	12
Transcription Yeast	-0.46063	0.918998	0.998868	0.911488	0.980551	0.989678	0.9313	94
ColoSpg	-0.45972	0.890182	0.938028	0.948127	0.995246	0.99756	0.951	18
Little Rock	-0.29949	0.000	0.599063	0.788836	0.511177	0.820148	0.7984	488
Software MySQL	-0.29935	0.001227	0.682953	0.709308	0.973751	0.989478	0.7931	1218
USAir 97	-0.16469	0.000	0.320908	0.629351	0.658826	0.722713	0.6627	788
Corporate_people	-0.07957	0.000	0.253447	0.496014	0.74503	0.753424	0.5764	5816
Neurons C elegans	-0.19894	0.000	0.508202	0.641156	0.534967	0.575818	0.6898	708
Software VTK	-0.33461	0.163029	0.797047	0.792189	0.938648	0.964823	0.8246	282
Ythan 1	-0.32207	0.004799	0.678286	0.799325	0.523225	0.747642	0.8212	119
Dolphins	-0.26747	0.154829	0.53117	0.72956	0.654247	0.713769	0.7547	43
PIN E coli	-0.1235	0	0.2408	0.6175	0.9258	0.9528	0.6227	1312
Software XMMS	-0.3757	0.308081	0.80956	0.825749	0.977193	0.988851	0.8629	314
termite Mound 3	-0.34671	0.542647	0.620194	0.805492	0.912995	0.958258	0.8284	85
Drugs	-0.18003	0.0001	0.362991	0.593936	0.846119	0.883264	0.6754	817
PIN Malaria	-0.26517	0.109428	0.655404	0.713576	0.650219	0.754015	0.7467	173
Software_Abi	-0.38835	0.14448	0.797758	0.831297	0.990739	0.995233	0.8703	290
El Verde	-0.31424	0.000	0.623571	0.811675	0.594033	0.748337	0.8124	271
PIN_H pylori	-0.37908	0.423129	0.876634	0.86533	0.858333	0.912228	0.8775	188
Stony	-0.47352	0.630128	0.979657	0.972289	0.715769	0.97181	0.9735	23
Electronic1	-0.37352	0.896729	0.912019	0.835979	0.904729	0.95014	0.8571	31
Roget Thesaurus	-0.21299	0.057644	0.535572	0.670604	0.860142	0.896855	0.7115	1199
Termite Mound 1	-0.4012	0.883001	0.844908	0.887574	0.935432	0.964056	0.9009	76
Electronic2	-0.36466	0.9005	0.943149	0.807018	0.909259	0.953746	0.8622	77
Scotch Broom	-0.39509	0.007431	0.659237	0.811475	0.878037	0.931493	0.8361	69
Termite Mound 2	-0.47143	0.987768	0.979899	0.960714	0.982756	0.990977	0.9607	11

Table 5.7: The minimum modularity and bipartivity measures.

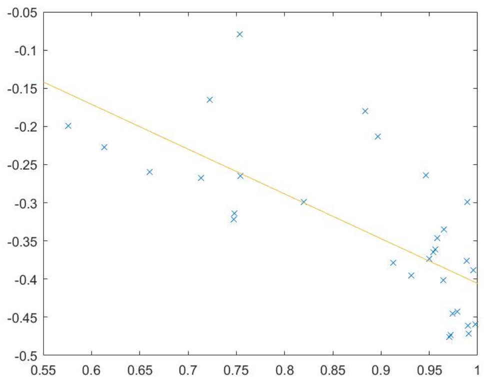
Then, we compute the Correlation Coefficient between the minimum modularity with bipartivity measures.
The following figures show the Correlation Coefficient between the minimum modularity with bipartivity measures.



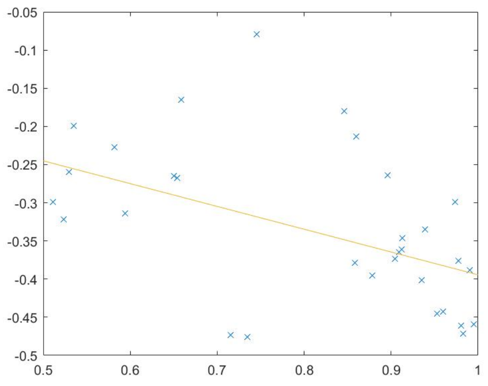
(a) b_e and modularity.



(b) b_λ and modularity..

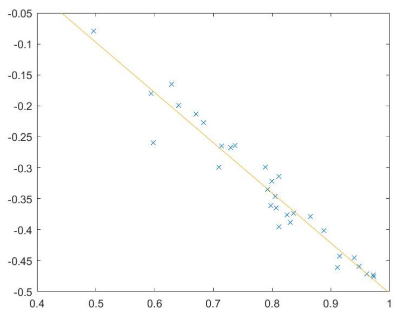


(c) b_{L_n} and modularity.

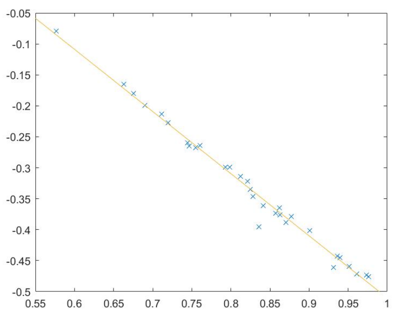


(d) b_{L_s} and modularity.

Figure 5.25: The correlation coefficient between modularity and b_e, b_λ, b_{L_n}, b_{L_s}.



(a) b_c and modularity.



(b) LSA and modularity.

Figure 5.26: The correlation coefficient between modularity and LSA, b_c.

The following Table 5.8 is the Correlation Coefficient between modularity and bipartivity measures for all 32 networks.

	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA
min	1	-0.73461	-0.93588	-0.9665	-0.4296	-0.62799	-0.99422
b_e	-0.73461	1	0.812837	0.724643	0.554774	0.60213	0.7427
b_λ	-0.93588	0.812837	1	0.890526	0.442844	0.606632	0.93782
b_c	-0.9665	0.724643	0.890526	1	0.401519	0.625114	0.976133
b_{L_s}	-0.4296	0.554774	0.442844	0.401519	1	0.880682	0.415021
b_{L_n}	-0.62799	0.60213	0.606632	0.625114	0.880682	1	0.628495
LSA	-0.99422	0.7427	0.93782	0.976133	0.415021	0.628495	1

Table 5.8: The Correlation Coefficient between modularity and bipartivity measures.

The following Table 5.9 is the Correlation Coefficient between modularity and bipartivity measures for networks which has modularity less than -0.35.

	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA
min	1	-0.20927	-0.40864	-0.95084	0.202146	-0.51683	-0.95275
b_e	-0.20927	1	0.795374	0.276937	0.144789	0.218053	0.314467
b_λ	-0.40864	0.795374	1	0.450376	-0.16772	0.289237	0.52746
b_c	-0.95084	0.276937	0.450376	1	-0.26345	0.431381	0.985435
b_{L_s}	0.202146	0.144789	-0.16772	-0.26345	1	0.435932	-0.24611
b_{L_n}	-0.51683	0.218053	0.289237	0.431381	0.435932	1	0.5108
LSA	-0.95275	0.314467	0.52746	0.985435	-0.24611	0.5108	1

Table 5.9: The Correlation Coefficient between modularity and bipartivity measures.

Then, we compute the modularity and bipartivity measures for other group of real world networks (Luft, Ryan, Easy, British, Turkish, Berlin, France, Scandinavian, KLM, Alitalia, Swiss, Iberia, Norwegian, Austrian, Flybe, Wizz, Portugal, Brussels, Finnair, LOT Polish, Vueling, Nostrum, Lingus, German, Pegasus, Netjets, Holland, Niki, Sun-Express, Aegean, Czech, European, Malev, Baltic, Wideroe, TNT and Olympic) in the Table 5.10 and we put the smallest value in each row in bold.

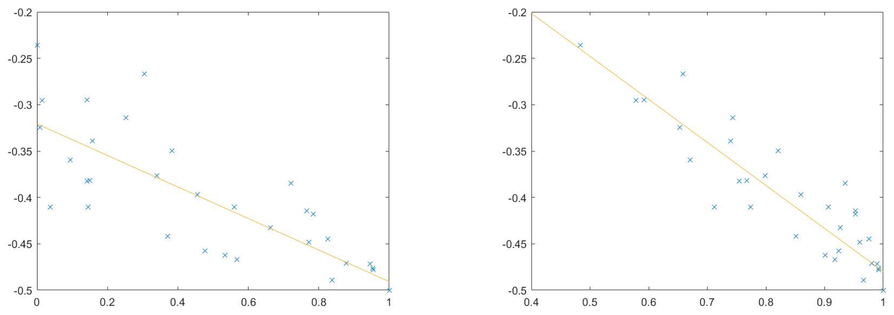
	1	2	3	4	5	6	7	8	9	10
Luft	-0.3722	-0.4058	-0.3722	-0.4058	-0.4064	-0.4064	-0.4017	**-0.4101**	-0.0972	**-0.4101**
Ryan	-0.1945	-0.2238	-0.1914	-0.2173	-0.1695	-0.2238	-0.2205	**-0.2362**	-0.1109	-0.2321
Easy	-0.2851	**-0.3249**	-0.2851	**-0.3249**	-0.3036	**-0.3249**	-0.3179	**-0.3249**	-0.1805	-0.3079
British	**-0.5**	**-0.5**	-0.485	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	-0.485	**-0.5**
Turkish	**-0.4577**	**-0.4577**	**-0.4577**	**-0.4577**	**-0.4577**	**-0.4577**	**-0.4577**	**-0.4577**	-0.3173	**-0.4577**
Berlin	-0.2616	-0.2826	-0.2616	-0.2826	-0.2538	-0.2836	**-0.2956**	**-0.2956**	-0.1702	-0.2831
France	-0.4575	-0.4575	-0.4575	-0.4575	**-0.471**	**-0.471**	**-0.471**	**-0.471**	-0.4575	-0.4575
Scandinavian	-0.3742	-0.3742	-0.3742	-0.3742	-0.3819	-0.3819	-0.3729	-0.3729	-0.1506	**-0.3822**
KLM	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	-0.1171	**-0.5**
Alitalia	-0.2537	-0.363	-0.2537	-0.363	-0.3515	**-0.3818**	**-0.3818**	**-0.3818**	-0.1445	-0.363
Swiss	-0.3868	**-0.4672**	-0.3868	**-0.4672**	**-0.4672**	**-0.4672**	**-0.4672**	**-0.4672**	0	-0.3068
Iberia	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**	**-0.4718**
Norwegian	-0.2831	**-0.3393**	-0.2831	**-0.3393**	-0.3277	**-0.3393**	-0.3277	**-0.3393**	-0.1718	**-0.3393**
Austrian	-0.4201	-0.4314	-0.4201	-0.4314	**-0.4448**	**-0.4448**	**-0.4448**	**-0.4448**	-0.4201	-0.4314
Flybe	**-0.3594**	**-0.3594**	**-0.3594**	**-0.3594**	**-0.3594**	**-0.3594**	**-0.3594**	**-0.3594**	-0.2139	**-0.3594**
Wizz	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**	**-0.4892**
Portugal	**-0.4623**	**-0.4623**	**-0.4623**	**-0.4623**	**-0.4623**	**-0.4623**	**-0.4623**	**-0.4623**	-0.2992	**-0.4623**
Brussels	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	0	**-0.5**
Finnair	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**	**-0.4765**
LOT Polish	-0.3056	**-0.4106**	-0.3056	**-0.4106**	-0.3605	**-0.4106**	-0.3605	**-0.4106**	-0.3056	**-0.4106**
Vueling	-0.2329	-0.3095	-0.2329	-0.3095	**-0.3141**	**-0.3141**	**-0.3141**	**-0.3141**	-0.2329	-0.3095
Nostrum	-0.3845	**-0.4147**	-0.3845	**-0.4147**	-0.3908	-0.4135	-0.3908	-0.4135	-0.3845	**-0.4147**
Lingus	-0.4316	**-0.4484**	-0.4316	**-0.4484**	-0.4316	**-0.4484**	-0.4316	**-0.4484**	-0.4316	**-0.4484**
German	-0.4104	-0.4255	0.02941	**-0.4421**	-0.2982	-0.4255	-0.4255	-0.4255	-0.2599	-0.4255
Pegasus	-0.2878	-0.33	-0.2878	-0.33	**-0.3971**	**-0.3971**	**-0.3971**	**-0.3971**	-0.2878	-0.33
Netjets	-0.2646	-0.2945	-0.2682	-0.2946	-0.2669	**-0.2948**	-0.1945	-0.2833	-0.1561	-0.2613
Holland	**-0.3847**	**-0.3847**	**-0.3847**	**-0.3847**	**-0.3847**	**-0.3847**	**-0.3847**	**-0.3847**	-0.3786	-0.3786
Niki	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**
SunExpress	**-0.4104**	**-0.4104**	**-0.4104**	**-0.4104**	**-0.4104**	**-0.4104**	**-0.4104**	**-0.4104**	-0.1965	-0.2517
Aegean	**-0.3498**	**-0.3498**	**-0.3498**	**-0.3498**	**-0.3498**	**-0.3498**	**-0.3498**	**-0.3498**	-0.2237	**-0.3498**
Czech	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	-0.144	**-0.5**
European	-0.3539	-0.3539	-0.3539	-0.3539	**-0.3768**	**-0.3768**	**-0.3768**	**-0.3768**	-0.2942	-0.3539
Malev	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	**-0.5**	0	**-0.5**
Baltic	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**	**-0.478**
Wideroe	-0.1343	-0.2558	-0.1728	**-0.2667**	-0.1956	-0.2556	-0.1956	-0.2556	-0.1467	**-0.2667**
TNT	-0.4038	**-0.4182**	-0.4038	**-0.4182**	-0.4038	**-0.4182**	-0.4038	**-0.4182**	-0.4038	**-0.4182**
Olympic	-0.3505	**-0.4327**	-0.3505	**-0.4327**	**-0.4327**	**-0.4327**	**-0.4327**	**-0.4327**	-0.3505	**-0.4327**

Table 5.10: The Modularity of airline networks.

	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA
Luft	-0.4101	0.0382	0.7737	0.8689	0.7205	0.8818	0.9057
Ryan	-0.2362	0.0001	0.4836	0.6922	0.58	0.651	0.7238
Easy	-0.3249	0.0083	0.6528	0.7752	0.6323	0.7766	0.8176
British	-0.5	1	1	1	1	1	1
Turkish	-0.4577	0.4772	0.9243	0.9576	0.892	0.9484	0.9576
Berlin	-0.2956	0.0147	0.5786	0.75	0.6244	0.7254	0.788
France	-0.471	0.8773	0.9805	0.9565	0.9343	0.968	0.9565
Scandinavian	-0.3822	0.1419	0.754	0.8727	0.7675	0.8766	0.8727
KLM	-0.5	1	1	1	1	1	1
Alitalia	-0.3818	0.1496	0.767	0.7204	0.7316	0.8484	0.8602
Swiss	-0.4672	0.5681	0.9173	0.8833	0.8879	0.9421	0.9667
Iberia	-0.4718	0.9461	0.9893	0.9714	0.9629	0.9802	0.9714
Norwegian	-0.3393	0.1582	0.7393	0.7701	0.6949	0.7989	0.8276
Austrian	-0.4448	0.8256	0.9758	0.9126	0.9126	0.9514	0.9306
Flybe	-0.3594	0.0939	0.6703	0.8586	0.6373	0.8023	0.8586
Wizz	-0.4892	0.8379	0.9661	0.9891	0.9369	0.9828	0.9891
Portugal	-0.4623	0.5333	0.901	0.9623	0.8949	0.9471	0.9623
Brussels	-0.5	1	1	1	1	1	1
Finnair	-0.4765	0.9538	0.992	0.9762	0.969	0.9836	0.9762
LOT Polish	-0.4106	0.5601	0.9064	0.7818	0.8408	0.9035	0.9091
Vueling	-0.3141	0.253	0.7437	0.6825	0.7379	0.8198	0.7778
Nostrum	-0.4147	0.7661	0.9528	0.8841	0.8912	0.9508	0.913
Lingus	-0.4484	0.7722	0.9599	0.931	0.8977	0.9463	0.9483
German	-0.4421	0.3704	0.851	0.9254	1	1	0.9254
Pegasus	-0.3971	0.4557	0.8591	0.7586	0.782	0.8626	0.8621
Netjets	-0.2948	0.1418	0.5924	0.7611	1	1	0.7944
Holland	-0.3847	0.7217	0.9355	0.8772	0.8001	0.8993	0.8772
Niki	-0.5	1	1	1	1	1	1
SunExpress	-0.4104	0.1446	0.712	0.9104	0.7454	0.8704	0.9104
Aegean	-0.3498	0.3833	0.8206	0.8491	0.787	0.868	0.8491
Czech	-0.5	1	1	1	1	1	1
European	-0.3768	0.3408	0.7987	0.8493	0.7952	0.8657	0.8493
Malev	-0.5	1	1	1	1	1	1
Baltic	-0.478	0.9551	0.9923	0.9778	0.9711	0.9848	0.9778
Wideroe	-0.2667	0.3055	0.6583	0.6222	0.7061	0.8435	0.7222
TNT	-0.4182	0.7848	0.9523	0.9016	0.8932	0.9401	0.918
Olympic	-0.4327	0.6631	0.9264	0.8372	0.8839	0.9331	0.8837

Table 5.11: The minimum modularity and bipartivity measures

Then we compute the Correlation Coefficient between b_e and modularity is -0.8317 and the Correlation Coefficient between b_λ and modularity is -0.9182.



(a) b_e and modularity. **(b)** b_λ and modularity.

Figure 5.27: The correlation Coefficient between modularity and b_l, b_e.

The following Table 5.12 is the Correlation Coefficient between modularity and bipartivity measures for all 37 airline networks.

	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA
min	1	-0.8317	-0.91819	-0.91404	-0.81252	-0.83949	-0.98819
be	-0.8317	1	0.926691	0.764336	0.840815	0.802489	0.812458
bl	-0.91819	0.926691	1	0.800575	0.809804	0.829564	0.88998
bc	-0.91404	0.764336	0.800575	1	0.76699	0.778942	0.942856
bs	-0.81252	0.840815	0.809804	0.76699	1	0.968017	0.802921
bn	-0.83949	0.802489	0.829564	0.778942	0.968017	1	0.827835
LSA	-0.98819	0.812458	0.88998	0.942856	0.802921	0.827835	1

Table 5.12: The correlation Coefficient between modularity and bipartivity measures for airline networks.

The following Table 5.13 is the Correlation Coefficient between modularity and bipartivity measures for networks which has modularity less than -0.35.

	min	b_e	b_λ	b_c	b_{L_s}	b_{L_n}	LSA
min	1	-0.80722	-0.86326	-0.85437	-0.64564	-0.66694	-0.97865
be	-0.80722	1	0.936242	0.718058	0.761893	0.742369	0.79186
bl	-0.86326	0.936242	1	0.691193	0.638563	0.6333	0.824602
bc	-0.85437	0.718058	0.691193	1	0.633957	0.682356	0.893477
bs	-0.64564	0.761893	0.638563	0.633957	1	0.984779	0.652197
bn	-0.66694	0.742369	0.6333	0.682356	0.984779	1	0.688325
LSA	-0.97865	0.79186	0.824602	0.893477	0.652197	0.688325	1

Table 5.13: The correlation Coefficient between modularity and bipartivity measures for airline networks.

5.5 Conclusion

In this chapter, we see the different ways of characterizing bipartivity lead to different algorithms for finding a way of partition. Our experiments with algorithms for finding a good partition into anti-communities show that simple spectral algorithms, based on the analysis of a single eigenvector, can perform as well (or better) than some of the more complex techniques that have been proposed in the literature, and that post-processing with local improvement can really pay dividends. The local improvement technique is similar to that introduced by Kernighan and Lin [47] but takes advantage of the fact that we do not need to balance the size of bipartitions, we simply need to reduce the number of frustrated edges. These algorithms can also give a relative cheap approximation of m_d, the number of frustrated edges, since we can get an upper bound by counting the intra-community edges in the bipartition we calculate for the cost of one eigenvector computation.

We show that method 3, which uses the signless Laplacian and finds the eigenvector associated with the smallest eigenvalue is faster than other methods in terms of time, and method 4, which uses the normalised Laplacian and finds the eigenvector associated with the biggest eigenvalue, is the best in terms of giving the best modularity, which is very closely correlated to m_d.

Chapter 6

Conclusions and Future Work

In this work, our aim was to examine some definitions and methods for analysing graph partitioning problems including clustering coefficients, the graph Laplacian including its eigenvalues and eigenvectors, the normalized Laplacian, and the signless Laplacian. We also introduced a selection of real-world test networks with which to compare our results. The networks were selected from a variety of fields of study such as brain networks, informational networks, biological networks, social and economic networks, technological networks, and software networks to ensure a wide array of network types were examined.

To attempt to enhance the area of community detection, we first had to review the existing literature on the subject. To this end, we examined the Stochastic Block Model, and various algorithms to partition such as local improvement methods and spectral partitioning. We examined various methods of assessing the quality of partitioning using Newman and Girvan's general expression of modularity. Indices such as the Performance Index and the Davies-Bouldin Validation Index study clusters as well. We determined that modularity was an effective method of measuring the quality of a partition among other validation indices contained within the literature. Along with modularity, similarity measures also needed to be obtained in order to judge network quality effectively. The Pearson Coefficient, the Manhattan norm, the Euclidean norm, and the Infinity norm were also surveyed. Examining centrality to determine node importance with respect to the rest of the graph also plays an important role in accurately and usefully partitioning graphs. Using similarity measures to find communities using linkage approaches was also reviewed. Assessing communicability between nodes that can be considered either off or on is another way of detecting communities as they are usually found closer together.

We identified some issues not just with using modularity but some other individual measures for community detection and that based on some of the drawbacks of each measure, using a combination of measures to assess quality is the best approach. More recently, one optimisation approach, the Louvain method, offers several advantages which serve to address the issue with resolution limits. In addition, we found that dynamical process such as vertex similarity or the map equation approach can also be used to detect communities. The two final methods discussed for detecting communities are the spectral bipartization method and the b_1 measure. The corollary of detecting communities would obviously seem to be detecting anti-communities, that is, areas of networks that have a lower density of connections. Conceptually, detecting anti-communities and bipartitions are analogous. Adjacent to this idea is finding the anti-modularity, an idea posed by Chen et al. which suggests that anti-communities may not be optimally partitioned with existing detection methods.

The use of random graph networks to provide test models for the various measures and approaches is dependent on how well the random graphs reflect the properties of real-world graphs. Since improving these networks was part of our aim (making random models more realistic), we used the Erdős–Rényi model, scale-free networks, and subsequently experimented with adding triangles on fragments. Also, we compared some more complex models including the BTER model and ones that use the degree distribution more closely and then concluded that there were improvements in the accuracy of random networks reflecting real world networks. Furthermore, we concluded that focusing specifically on adding triangles was more effective than adding other types of fragments and if we added triangles, we affect many other types of fragments and sometimes we are very successful in bringing everything in line. In this study, we examined ten real networks by computing 15 fragments. For our experiment we showed that BTER is the best but it takes more time than Erdős–Rényi model and scale-free model.

Rather than simply identifying graphs as bipartite or non-bipartite, measuring to what degree a graph is bipartite or characterizing bipartivity is a good way of addressing inefficiencies within networks. We reviewed characterizations of bipartivity in real-world networks and random networks to demonstrate the importance of this area. We proved 12 equivalent characterisations of bipartivity. We defined eight measures for networks to measure bipartivity and showed the different bipartivity measures in real-world networks and random networks give us different results for the same case. Also, we showed that different measures can have different behavior in practice and in theory. In theory, we showed that two of these measures could exactly do the opposite thing, with one going to 1 and the other going to 0. So, it is important to see what happens with these bipartivity measures on real world networks. For this, we examined the real-world networks by splitting our networks using the signs of the eigenvector of the most negative eigenvalue of the adjacency matrix. We then randomly add back the edges removed in forming these bipartitions and look at how the bipartivity measures evolve and showed that there are groups of real worlds that have different behavior of bipartivity measures. In each group we see a different type of behaviour. Also, we examined whether a network is nearly bipartite or not by modularity and we examined different types of random graphs like random trees, Tree generated by Prüfer sequences we showed that b_λ and b_{L_N} have same behaviour and b_e and b_{L_s} have same behaviour and goes down steeply and very fast. Also, we examined the Fullerene Graphs where all measures have the same behaviour.

Experiments with 6 different algorithms in order to detect anti-communities so as to divide graphs into communities with as few connections as possible between them were conducted. Furthermore, the effectiveness of each algorithm was also tested using a local improvement method with modularity. Finally, these methods were tested with 32 real-world networks and the results were compared. A second experiment was run with the same real-world networks and 37 airline networks to compare minimum modularity and bipartivity measures in order to compare minimum modularity and bipartivity measures. Overall, using the signless Laplacian with the smallest eigenvectors and eigenvalues was the fastest while using the normalised Laplacian with the largest eigenvectors and eigenvalues resulted in the best modularity.

In the future, I will try to extend the work to approximate multipartivity but this requires a new theory of characterizations of multipartivity. Approximate multipartivity is common, for example, in food networks where the different partitions represent different trophic levels (from pure predators to prey and vegetation, we can also try to see if we can use multiple bipartivity measures on the same graph to get additional insight (e.g. core-periphery structure appears to have high b_λ but low b_e. That is we aim to expand the idea of local and global bipartivity we have observed in some our theoretical examples to real world networks and our experiments give evidence that the phenomenon is prevalent in real examples.

Also, I will try to improve random models by gaining an understanding of when adding triangles works. Our experiment in Chapter 3 showed that we could be very successful in reproducing the graphlet statistics of real world networks. But so far we have been unable to classify the circumstances that lead to this success. If we can distinguish between cases we could provide much more reliable simulations and possibly get a better understanding of when graphlets are the best tool for interpreting complex network behaviour.

Appendix A

MATLAB code for Chapters 4 and 5

1. Taking random networks ER and adding triangles (ERplusT.m).
2. Taking random networks BA and adding triangles (BAplusT.m).
3. Random trees with different size (treeA.m).
4. Tree generated by Prüfer sequences with different size (treeB.m).
5. Random tree with different measures after adding edges (treeL.m).
6. Trees generated by Prüfer sequences with different measures after adding edges (PtreeL.m).
7. Reintroducing edges to random graphs with different measures (Randombipartite.m).
8. Computing the bipartivity measures for real-world graphs after add back edges removed (bipreal2.m).
9. Finding the best bipartition (bestbip.m).

ERplusT.m

```
function [A,t] = ERplusT(n, m, tr, fl)
% Take an ER network with n nodes and m edges and rewire so that it has
% around tr triangles.
warning off
```

```
% Optional flag to report statistics.
if nargin < 4, fl = 0; end


A=sparse(n,n);
if 2*m > n*(n-1), disp('Too many edges requested.'), return, end
% Generate the ER network (upper-triangular part only)
% This method seems faster than built-in routines and guarantees an
% exact number of edges.
c = 0;
while c < m
    % M random edges. Avoid loops. Repeat until we have m unique edges.
    M = m-c;
    i = randi(n,M,1)-1; j = mod(i+randi(n-1,M,1),n);
    % Make sure i < j.
    ij = sort([i j],2)+1;
    % Check for duplicate edges.
    A = A + sparse(ij(:,1),ij(:,2),1,n,n);
    c = nnz(A);
end


% Built-in: A = triu(sprandsym(n,2*m/n/(n-1))>0,1);


A = A > 0;%, plot(graph(A+A')), pause
% Find edges that lie in existing triangles
B = A.*(A+A')^2;%, pause
% Initial number of triangles
TR = sum(full(sum(B)))/3; trg = tr - TR;


if fl, fprintf('Initial graph has %d triangles.\n',TR), end


if trg <= 0
```

```
    k = 0;
else
    k = 1;
end


ct = 0;


while k && ct < 5
    k = 0; ct = ct+1;
    % Find edges that aren't part of triangles
    C = A-(B>0); [x,y] = find(C);
    % Remove trg of these edges, if there are that many.
    pote = length(x);
    if pote < trg
       de = 1:pote;
    else
        re = randperm(pote); de = re(1:trg);
    end
    A = A - sparse(x(de),y(de),1,n,n);


    mxeadd = min([pote trg]);
    % Find P2s that aren't in triangles and identify missing edges
        A2 = triu((A+A')^2,1);
        D = A2 - A2.*A;
        [xn,yn] = find(D);
        me = length(xn);
        % If there aren't enough, put back some of the removed edges.
        % Then it's worth another iteration.
        if me < mxeadd
            k = 1;
            xx = unique([xn yn;x y],'rows');
            xn = xx(:,1); yn = xx(:,2);
            me = length(xn);
        end
            re = randperm(me); re = re(1:mxeadd);
            xn = xn(re); yn = yn(re);
        A = A + sparse(xn,yn,1,n,n);
    %plot(graph(A+A'));pause
    if k % Recalculate values needed for another iteration.
        B = A.*(A+A')^2; TRa = sum(full(sum(B)))/3;
        trg = tr - TRa; if trg <=0, k = 0; end
    end

end

if fl || nargout==2
    t = sum(full(sum(A.*(A^2))));
end
if fl
    fprintf('Final graph has %d triangles.\n', t)
end

A = A + A';
```

```
BAplusT.m
function [A,t] = BAplusT(n, d, tr, fl)
% Take a BA network with n nodes and minimum degree d edges and rewire so that
    it has
% around tr triangles. Uses pref.m (courtesy of Higham and Taylor).
warning off
% Optional flag to report statistics.
```

```
if nargin < 4, fl = 0; end

A = triu(pref(n,d),0); % Upper tirangular portion of a BA network.

% Find edges that lie in existing triangles
B = A.*(A+A')^2;%, pause
% Initial number of triangles
TR = sum(full(sum(B)))/3; trg = tr - TR;

if fl, fprintf('Initial graph has %d triangles.\n',TR), end

if trg <= 0
    k = 0;
else
    k = 1;
end

ct = 0;

while k && ct < 5
    k = 0; ct = ct+1;
    % Find edges that aren't part of triangles
    C = A-(B>0); [x,y] = find(C);
    % Remove trg of these edges, if there are that many.
    pote = length(x);
    if pote < trg
       de = 1:pote;
    else
        re = randperm(pote); de = re(1:trg);
    end
    A = A - sparse(x(de),y(de),1,n,n);
    mxeadd = min([pote trg]);
    % Find P2s that aren't in triangles and identify missing edges
        A2 = triu((A+A')^2,1);
        D = A2 - A2.*A;
        [xn,yn] = find(D);
        me = length(xn);
        % If there aren't enough, put back some of the removed edges.
        % Then it's worth another iteration.
        if me < mxeadd
            k = 1;
            xx = unique([xn yn;x y],'rows');
            xn = xx(:,1); yn = xx(:,2);
            me = length(xn);
        end
            re = randperm(me); re = re(1:mxeadd);
            xn = xn(re); yn = yn(re);
        A = A + sparse(xn,yn,1,n,n);
    %plot(graph(A+A'));pause
    if k % Recalculate values needed for another iteration.
        B = A.*(A+A')^2; TRa = sum(full(sum(B)))/3;
        trg = tr - TRa; if trg <=0, k = 0; end
    end


end


if fl || nargout==2
    t = sum(full(sum(A.*(A^2))));
end
if fl
    fprintf('Final graph has %d triangles.\n', t)
```

```
end

A = A + A';

treeA.m

n = 1000;

i = 2:n; j = zeros(1,n-1); % A tree has n-1 edges

% In this loop we will connect node k+1 to one of the existing nodes at random.

k = zeros(n-1,1);

for x = 1:50



for h=1:n-1

j(h) = randi(h); end;

% Put 1s in the adjacency matrix at the prescribed coordinates.

A = sparse(i,j,1,n,n); A = A + A';

k = k+ hist(A*ones(n,1),1:n-1)';
end

dis = k/sum(k);
```

```
maxdeg = max(find(k));

plot(1:maxdeg,dis(1:maxdeg))

plot(graph(A))
```

treeB.m

```
% convert prufer sequence to tree

n = 500;

p = randi(n,1,n-2);

t = zeros(n-1,2);

d = ones(n,1);


  for i = 1:n-2

    d(p(i)) = d(p(i)) + 1;

  end


  for i = 1:n-2

    x = find(d==1,1);

    y = p(i);

    d(x) = d(x) - 1; d(y) = d(y) - 1;

    t(i,:) = [x; y];

  end

   t(n-1,:) = find(d==1);


k = zeros(n-1,1);


for x=1:50;

    A=sparse(t(:,1),t(:,2),1,n,n); A = A+A';




k = k+ hist(A*ones(n,1),1:n-1)';
```

```
end

dis = k/sum(k);

maxdeg = max(find(k));
plot(1:maxdeg,dis(1:maxdeg))

plot(graph(A))

treeL.m

n = 500;

i = 2:n; j = zeros(1,n-1); % A tree has n-1 edges
% In this loop we will connect node k+1 to one of the existing nodes at random.
k = zeros(n-1,1);

for x = 1:50



for h=1:n-1


j(h) = randi(h); end;


% Put 1s in the adjacency matrix at the prescribed coordinates.


A = sparse(i,j,1,n,n); A = A + A';
k = k+ hist(A*ones(n,1),1:n-1)';
end


% Some edges to add
```

```
add  = 3000;

i1 = randi(n,add,1)-1; i2 = mod(i1+randi(n-1,add,1),n);

i1 = i1 + 1; i2 = i2 + 1;


 bE = zeros(add,1);

 bL = zeros(add,1);

 bLn= zeros(add,1);

 bLs= zeros(add,1);

C = full(A);



for h = 1: add

    % Add the edge

    C(i1(h),i2(h)) = 1; C(i2(h),i1(h)) = 1;

    % Compute the new bipartivity measures

    n=size(C,1);

k=C*ones(n,1);

Ls=diag(k)+C;

d=1./sqrt(k);

Ln=eye(n)-diag(d)*C*diag(d);

    eval = eig(C);

    bL(h) = abs(min(eval))/max(eval);

    bE(h) = sum(exp(-eval))/sum(exp(eval));

    bLs(h)=1/(1+min(eig(Ls)));

    bLn(h)=abs(1-max(eig(Ln)));

end

figure(2)

plot(1:add,bL,1:add,bE,1:add,bLn,1:add,bLs)
```

```
PtreeL.m

% convert prufer sequence to tree

n = 500;

p = randi(n,1,n-2);

t = zeros(n-1,2);

d = ones(n,1);


  for i = 1:n-2

    d(p(i)) = d(p(i)) + 1;

  end


  for i = 1:n-2

    x = find(d==1,1);

    y = p(i);

    d(x) = d(x) - 1; d(y) = d(y) - 1;

    t(i,:) = [x; y];

  end

  t(n-1,:) = find(d==1);


k = zeros(n-1,1);


for x=1:50;

    A=sparse(t(:,1),t(:,2),1,n,n); A = A+A';




k = k+ hist(A*ones(n,1),1:n-1)';


m=sum(k)/2;

end


% Some edges to add
```

```
add  = 5000;

i1 = randi(n,add,1)-1; i2 = mod(i1+randi(n-1,add,1),n);

i1 = i1 + 1; i2 = i2 + 1;

% compute the bipqrtivity

 bE = zeros(add,1);

 bL = zeros(add,1);

 bLn= zeros(add,1);

 bLs= zeros(add,1);


C = full(A);

for h = 1: add

    % Add the edge

    C(i1(h),i2(h)) = 1; C(i2(h),i1(h)) = 1;

    % Compute the new bipartivity measures

    eval = eig(C);

   k =C*ones(n,1);

Ls=diag(k)+C;

d=1./sqrt(k);

Ln=eye(n)-diag(d)*C*diag(d);

    eval = eig(C);

    bL(h) = abs(min(eval))/max(eval);

    bE(h) = sum(exp(-eval))/sum(exp(eval));

    bLs(h)=1/(1+min(eig(Ls)));

    bLn(h)=abs(1-max(eig(Ln)));

end

figure(2)

plot(1:add,bL,1:add,bE,1:add,bLn,1:add,bLs)


figure(1)

 plot(graph(A))
```

Randombipartite.m

```
n1 = 250; n2 = 250; p =.3;

B = rand(n1,n2)<p;

A = [zeros(n1) B; B' zeros(n2)];

n = size(A,1);

[V,D]=eigs(A,1,-n); % Most negative eigenvalue

% Find two partitions

P1=find(V>0);P2=find(V<0);

% B will be adjacency matrix of the bipartite graph

C = A; C(P1,P1)=0; C(P2,P2)=0;




% You can see the bipartition with spy. Not obvious from the graph.


figure(1)

spy(C([P1;P2],[P1;P2]))

figure(2)

plot(graph(C([P1;P2],[P1;P2])))


% Now add some edges to destroy bipartivity and measure the change in b_l

% and b_e

p1 = length(P1); p2 = length(P2);

% We'll add this many edges


add = 50;

bE = zeros(add,1);

bL = zeros(add,1);

bLn= zeros(add,1);

bLs= zeros(add,1);
```

```
bALL = zeros(add,4);

for  trials = 1:10

C=A;
for i = 1: add
    % Flip a coin to decide whether to add edges to nodes in P1 or P2
    if rand < .5
        x = randperm(p1); edge = P1(x(1:2));
    else
        x = randperm(p2); edge = P2(x(1:2));
    end
    % Add the edge
    C(edge(1),edge(2)) = 1; C(edge(2),edge(1)) = 1;
    % Compute the new bipartivity measures
    eval = eig(C);
     k =C*ones(n1+n2,1);
     d=1./sqrt(k);
     Ls=diag(k)+C;
     Ln=eye(n1+n2)-diag(d)*C*diag(d);
    bL(i) = abs(min(eval))/max(eval);
    bE(i) = sum(exp(-eval))/sum(exp(eval));
    bLs(i)=1/(1+min(eig(Ls)));
    bLn(i)=abs(1-max(eig(Ln)));
end

bALL = bALL + [bL bE bLs bLn];
end
bALL = bALL/10;
figure(3)
```

```
plot(1:add,bALL(:,1),1:add,bALL(:,2),1:add,bALL(:,3),1:add,bALL(:,4))
```

bipreal2.m

```
% compute the bipartivity for real networks and add edges
D=load('Dolphins.txt'); A=edgeL2adj(D);
[u,v]=eig(A);
diag(v);
bpv=u(:,1);
p1=find(bpv>0);p2=find(bpv<=0);
B=A;B(p1,p1)=0;B(p2,p2)=0;
figure(1)
plot(graph(B));
C=triu(A-B);
[i,j]=find(C);

add =length(i);
 bE = zeros(add,1);
 bL = zeros(add,1);
 bLn= zeros(add,1);
 bLs= zeros(add,1);

bAll = zeros(add,4);
% We'll repeatedly add the edges back to the bipartite graph
% and then plot the average measures.

numtrials = 10;
for trials = 1:numtrials
    p = randperm(add);
    i = i(p); j = j(p);
    F = full(B);
```

```
for k = 1: add
    % Add the edge
    F(i(k),j(k)) = 1; F(j(k),i(k)) = 1;
    % Compute the new bipartivity measures
    eval = eig(F);
    n = size(F,1);
    K =F*ones(n,1);
     d=1./sqrt(K);
     Ls=diag(K)+F;
     Ln=eye(n)-diag(d)*F*diag(d);
    bL(k) = abs(min(eval))/max(eval);
    bE(k) = sum(exp(-eval))/sum(exp(eval));
    bLs(k)=1/(1+min(eig(Ls)));
    bLn(k)=abs(1-max(eig(Ln)));
end
    bAll=bAll+[bL bE bLs bLn];
end

bAll = bAll/numtrials;

plot(1:add,bAll(:,1),1:add,bAll(:,2),1:add,bAll(:,3),1:add,bAll(:,4))
```

```
bestbip.m
C=load('Centrality_literature.txt');
A=edgeL2adj(C);A = A + A'; A = A>0; A = A - diag(diag(A));
tr=trace(A^3)/6;m=sum(sum(A))/2;n=size(A,1);
G=expm(-A);
B = G>0; B = B - diag(diag(B));
L = diag(sum(B)) - B;
```

```
[u,v]=eigs(L,2,0);

v(2,2)

conncomp(graph(B))

p=find(u(:,2)>0);q=find(u(:,2)<=0);

spy(A([p;q],[p;q]))

subplot(2,2,2);spy(A([p;q],[p;q]))

[u,v]=eigs(A,1,-1000);

p=find(u(:,1)>0);q=find(u(:,1)<=0);

sum(sum(A(p,p)))+sum(sum(A(q,q)))

subplot(2,2,1);spy(A([q; p],[q; p]));

N=A+diag(sum(A));

eig(N);

[u,v]=eigs(N,1,0);

p=find(u(:,1)>0);q=find(u(:,1)<=0);

subplot(2,2,3);spy(A([q;p],[q;p]))

D=diag(sum(A));

D=diag(1./sqrt(sum(A)));

M=eye(n)-D*A*D;

[u,v]=eigs(M,1);

eig(M);

p=find(u(:,1)>0);q=find(u(:,1)<=0);

sum(sum(A(p,p)))+sum(sum(A(q,q)))

subplot(2,2,4);spy(A([p;q],[p;q]))
```

Bibliography

[1] Jimi Adams, James Moody, and Martina Morris. "Sex, drugs, and race: how behaviors differentially contribute to the sexually transmitted infection risk network structure". In: *American journal of public health* 103.2 (2013), pp. 322–329.

[2] Réka Albert and Albert-László Barabási. "Statistical mechanics of complex networks". In: *Reviews of modern physics* 74.1 (2002), p. 47.

[3] Azhar Aleidan and Philip A Knight. "Spectral techniques for measuring bipartivity and producing partitions". In: *Journal of Complex Networks* 11.4 (2023), cnad026.

[4] Konstantin Andreev and Harald Racke. "Balanced graph partitioning". In: *Theory of Computing Systems* 39.6 (2006), pp. 929–939.

[5] Alex Arenas, Alberto Fernandez, and Sergio Gomez. "Analysis of the structure of complex networks at different resolution levels". In: *New journal of physics* 10.5 (2008), p. 053039.

[6] Vladimir Batagelj and Andrej Mrvar. "Layouts for GD01 graph-drawing competition". In: (2001). URL: `http://vlado.fmf.uni-lj.si/pub/gd/01/report.pdf` (visited on 01/05/2023).

[7] Alexander Bertrand and Marc Moonen. "Distributed computation of the Fiedler vector with application to topology inference in ad hoc networks". In: *Signal Processing* 93.5 (2013), pp. 1106–1117.

[8] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[9] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. “On modularity clustering”. In: *IEEE transactions on knowledge and data engineering* 20.2 (2007), pp. 172–188.

[10] Franc Brglez, David Bryan, and Krzysztof Kozminski. “Combinational profiles of sequential benchmark circuits”. In: *IEEE International Symposium on Circuits and Systems*. IEEE. 1989, pp. 1929–1934.

[11] Sergey Brin and Lawrence Page. “The anatomy of a large-scale hypertextual web search engine”. In: *Computer networks and ISDN systems* 30.1-7 (1998), pp. 107–117.

[12] Gareth Butland, José Manuel Peregrın-Alvarez, Joyce Li, Wehong Yang, Xiaochun Yang, Andrei Starostine, Dawn Richards, Bryan Beattie, Nevan Krogan, Michael Davey, et al. “Interaction network containing conserved and essential protein complexes in Escherichia coli”. In: *Nature* 433.7025 (2005), pp. 531–537.

[13] Ling Chen, Qiang Yu, and Bolun Chen. “Anti-modularity and anti-community detecting in complex networks”. In: *Information Sciences* 275 (2014), pp. 293–313.

[14] Fan Chung and Ron Graham. *Spectral graph theory*. 92. American Mathematical Soc., 1997.

[15] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. “Finding community structure in very large networks”. In: *Physical review E* 70.6 (2004), p. 066111.

[16] Vittoria Colizza, Romualdo Pastor-Satorras, and Alessandro Vespignani. “Reaction–diffusion processes and metapopulation models in heterogeneous networks”. In: *Nature Physics* 3.4 (2007), pp. 276–282.

[17] Anna Concas, Silvia Noschese, Lothar Reichel, and Giuseppe Rodriguez. “A spectral method for bipartizing a network and detecting a large anti-community”. In: *Journal of Computational and Applied Mathematics* 373 (2020), p. 112306.

[18] Dragoš Cvetković, Peter Rowlinson, and Slobodan K Simić. “Signless Laplacians of finite graphs”. In: *Linear Algebra and its applications* 423.1 (2007), pp. 155–171.

[19] Kinkar Ch Das. “On conjectures involving second largest signless Laplacian eigenvalue of graphs”. In: *Linear Algebra and its Applications* 432.11 (2010), pp. 3018–3029.

[20] Gerald F Davis, Mina Yoo, and Wayne E Baker. “The small world of the American corporate elite, 1982-2001”. In: *Strategic organization* 1.3 (2003), pp. 301–326.

[21] Tomislav Došlić and Damir Vukičević. “Computing the bipartite edge frustration of fullerene graphs”. In: *Discrete Applied Mathematics* 155.10 (2007), pp. 1294–1301.

[22] Ernesto Estrada. *The structure of complex networks: theory and applications.* 2012.

[23] Ernesto Estrada and Jesús Gómez-Gardeñes. “Network bipartivity and the transportation efficiency of european passenger airlines”. In: *Physica D: Nonlinear Phenomena* 323 (2016), pp. 57–63.

[24] Ernesto Estrada and Naomichi Hatano. “Communicability in complex networks”. In: *Physical Review E* 77.3 (2008), p. 036111.

[25] Ernesto Estrada and Philip A Knight. *A first course in network theory.* Oxford University Press, USA, 2015.

[26] Ernesto Estrada and Juan A Rodríguez-Velázquez. “Spectral measures of bipartivity in complex networks”. In: *Physical Review E* 72.4 (2005), p. 046105.

[27] Dario Fasino and Francesco Tudisco. “A modularity based spectral method for simultaneous community and anti-community detection”. In: *Linear Algebra and its Applications* 542 (2018), pp. 605–623.

[28] Miroslav Fiedler. “A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory”. In: *Czechoslovak mathematical journal* 25.4 (1975), pp. 619–633.

[29] Josh A Firth and Ben C Sheldon. “Experimental manipulation of avian social structure reveals segregation is carried over across contexts”. In: *Proceedings of the Royal Society B: Biological Sciences* 282.1802 (2015), p. 20142350.

[30] Santo Fortunato. “Community detection in graphs”. In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[31] Santo Fortunato and Marc Barthelemy. “Resolution limit in community detection”. In: *Proceedings of the national academy of sciences* 104.1 (2007), pp. 36–41.

[32] Santo Fortunato and Darko Hric. “Community detection in networks: A user guide”. In: *Physics reports* 659 (2016), pp. 1–44.

[33] Jens Gottlieb, Bryant A Julstrom, Günther R Raidl, and Franz Rothlauf. “Prüfer numbers: A poor representation of spanning trees for evolutionary search”. In: *Proceedings of the Genetic and Evolutionary Computation Conference*. Vol. 343. 2001, p. 350.

[34] Jonathan L Gross and Jay Yellen. *Handbook of graph theory*. CRC press, 2003.

[35] Roger Guimera, Marta Sales-Pardo, and Luís A Nunes Amaral. “Modularity from fluctuations in random graphs and complex networks”. In: *Physical Review E* 70.2 (2004), p. 025101.

[36] Shin-Kap Han. “The other ride of Paul Revere: The brokerage role in the making of the American revolution”. In: *Mobilization: An international quarterly* 14.2 (2009), pp. 143–162.

[37] Karl Havens. “Scale and structure in natural food webs”. In: *Science* 257.5073 (1992), pp. 1107–1109.

[38] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. “Stochastic blockmodels: First steps”. In: *Social networks* 5.2 (1983), pp. 109–137.

[39] Petter Holme, Fredrik Liljeros, Christofer R Edling, and Beom Jun Kim. “Network bipartivity”. In: *Physical Review E* 68.5 (2003), p. 056107.

[40] Yanqing Hu, Hongbin Chen, Peng Zhang, Menghui Li, Zengru Di, and Ying Fan. “Comparative definition of community and corresponding identifying algorithm”. In: *Physical Review E* 78.2 (2008), p. 026121.

[41] Norman P Hummon, Patrick Doreian, and Linton C Freeman. “Analyzing the structure of the centrality-productivity literature created between 1948 and 1979”. In: *Knowledge* 11.4 (1990), pp. 459–480.

[42] Mark Huxham, S Beaney, and Dave Raffaelli. “Do parasites reduce the chances of triangulation in a real food web?” In: *Oikos* (1996), pp. 284–300.

[43] Thomas C. Ings, José M. Montoya, Jordi Bascompte, Nico Blüthgen, Lee Brown, Carsten F. Dormann, François Edwards, David Figueroa, Ute Jacob, J. Iwan Jones, Rasmus B. Lauridsen, Mark E. Ledger, Hannah M. Lewis, Jens M. Olesen, F.J. Frank Van Veen, Phil H. Warren, and Guy Woodward. “Ecological networks–beyond food webs”. In: *Journal of Animal Ecology* 78.1 (2009), pp. 253–269.

[44] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. “What’s in a crowd? Analysis of face-to-face behavioral networks”. In: *Journal of theoretical biology* 271.1 (2011), pp. 166–180.

[45] Kathryn James. “Six degrees of information seeking: Stanley Milgram and the small world of the library”. In: *The Journal of Academic Librarianship* 32.5 (2006), pp. 527–532.

[46] Lucas GS Jeub, Prakash Balachandran, Mason A Porter, Peter J Mucha, and Michael W Mahoney. “Think locally, act locally: Detection of small, medium-sized, and large communities in large networks”. In: *Physical Review E* 91.1 (2015), p. 012821.

[47] Brian W Kernighan and Shen Lin. “An efficient heuristic procedure for partitioning graphs”. In: *The Bell system technical journal* 49.2 (1970), pp. 291–307.

[48] Steve Kirkland and Debdas Paul. “Bipartite subgraphs and the signless Laplacian matrix”. In: *Applicable Analysis and Discrete Mathematics* (2011), pp. 1–13.

[49] Jon M Kleinberg et al. “Authoritative sources in a hyperlinked environment.” In: *SODA*. Vol. 98. Citeseer. 1998, pp. 668–677.

[50] Valdis Krebs. “Proxy Networks. Analyzing One Network to Reveal Another”. In: *Bulletin de méthodologie sociologique. Bulletin of sociological methodology* 79 (2003), pp. 61–70.

[51] Jérôme Kunegis. “Exploiting the structure of bipartite graphs for algebraic and spectral graph theory applications”. In: *Internet Mathematics* 11.3 (2015), pp. 201–321.

[52] Jérôme Kunegis. “Handbook of Network Analysis [KONECT–the Koblenz Network Collection]”. In: *arXiv preprint arXiv:1402.5500* (2014).

[53] Sebastian Lackner, Andreas Spitz, Matthias Weidemüller, and Michael Gertz. “Efficient anti-community detection in complex networks”. In: *Proceedings of the 30th International Conference on Scientific and Statistical Database Management.* 2018, pp. 1–12.

[54] Douglas J LaCount, Marissa Vignali, Rakesh Chettier, Amit Phansalkar, Russell Bell, Jay R Hesselberth, Lori W Schoenfeld, Irene Ota, Sudhir Sahasrabudhe, Cornelia Kurschner, et al. “A protein interaction network of the malaria parasite Plasmodium falciparum”. In: *Nature* 438.7064 (2005), pp. 103–107.

[55] Andrea Lancichinetti and Santo Fortunato. “Limits of modularity maximization in community detection”. In: *Physical review E* 84.6 (2011), p. 066122.

[56] Chung-Yen Lin, Chia-Ling Chen, Chi-Shiang Cho, Li-Ming Wang, Chia-Ming Chang, Pao-Yang Chen, Chen-Zen Lo, and Chao A Hsiung. “hp-DPI: Helicobacter pylori database of protein interactomes—embracing experimental and inferred interactions”. In: *Bioinformatics* 21.7 (2005), pp. 1288–1290.

[57] Francois Lorrain and Harrison C White. “Structural equivalence of individuals in social networks”. In: *The Journal of mathematical sociology* 1.1 (1971), pp. 49–80.

[58] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. “The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations”. In: *Behavioral Ecology and Sociobiology* 54.4 (2003), pp. 396–405.

[59] J Memmott, ND Martinez, and JE Cohen. “Predators, parasitoids and pathogens: species richness, trophic generality and body sizes in a natural food web”. In: *Journal of Animal Ecology* 69.1 (2000), pp. 1–15.

[60] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon. “Superfamilies of evolved and designed networks”. In: *Science* 303.5663 (2004), pp. 1538–1542.

[61] Carlo Morselli. *Inside criminal networks*. Vol. 8. Springer, 2009.

[62] Christopher R Myers. “Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs”. In: *Physical review E* 68.4 (2003), p. 046116.

[63] Mark EJ Newman. “Finding community structure in networks using the eigenvectors of matrices”. In: *Physical review E* 74.3 (2006), p. 036104.

[64] Mark EJ Newman and Michelle Girvan. “Finding and evaluating community structure in networks”. In: *Physical review E* 69.2 (2004), p. 026113.

[65] Mark EJ Newman and Gesine Reinert. “Estimating the number of communities in a network”. In: *Physical review letters* 117.7 (2016), p. 078301.

[66] Andrew Ng, Michael Jordan, and Yair Weiss. “On spectral clustering: Analysis and an algorithm”. In: *Advances in neural information processing systems* 14 (2001), pp. 849–856.

[67] Debdas Paul and Dragan Stevanović. “Eigenvector-based identification of bipartite subgraphs”. In: *Discrete Applied Mathematics* 269 (2019), pp. 146–158.

[68] Georgios A Pavlopoulos, Panagiota I Kontou, Athanasia Pavlopoulou, Costas Bouyioukos, Evripides Markou, and Pantelis G Bagos. “Bipartite graphs in systems biology and medicine: a survey of methods and applications”. In: *GigaScience* 7.4 (2018), giy014.

[69] Tiago P Peixoto. “Bayesian stochastic blockmodeling”. In: *Advances in network clustering and blockmodeling* (2019), pp. 289–332.

[70] Tiago P Peixoto. “Hierarchical block structures and high-resolution model selection in large networks”. In: *Physical Review X* 4.1 (2014), p. 011047.

[71] Andrea Perna, Sergi Valverde, Jacques Gautrais, Christian Jost, Ricard Solé, Pascale Kuntz, and Guy Theraulaz. “Topological efficiency in three-dimensional gallery networks of termite nests”. In: *Physica A: Statistical Mechanics and its Applications* 387.24 (2008), pp. 6235–6244.

[72] Pascal Pons and Matthieu Latapy. “Computing communities in large networks using random walks”. In: *International symposium on computer and information sciences*. Springer. 2005, pp. 284–293.

[73] John J Potterat, L Phillips-Plummer, Stephen Q Muth, RB Rothenberg, DE Woodhouse, TS Maldonado-Long, HP Zimmerman, and JB Muth. “Risk network structure in the early epidemic phase of HIV transmission in Colorado Springs”. In: *Sexually transmitted infections* 78.suppl 1 (2002), pp. i159–i163.

[74] Heinz Prüfer. “Neuer beweis eines satzes über permutationen”. In: *Arch. Math. Phys* 27.1918 (1918), pp. 742–744.

[75] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi. “Defining and identifying communities in networks”. In: *Proceedings of the national academy of sciences* 101.9 (2004), pp. 2658–2663.

[76] Douglas P Reagan and Robert B Waide. *The food web of a tropical rain forest.* University of Chicago Press, 1996.

[77] Comandur Seshadhri, Tamara G Kolda, and Ali Pinar. "Community structure and scale-free collections of Erdős-Rényi graphs". In: *Physical Review E* 85.5 (2012), p. 056109.

[78] Shai S Shen-Orr, Ron Milo, Shmoolik Mangan, and Uri Alon. "Network motifs in the transcriptional regulation network of Escherichia coli". In: *Nature genetics* 31.1 (2002), pp. 64–68.

[79] Ross M Thompson and Angus R Mcintosh. "Disturbance, resource supply, and food-web architecture in streams". In: *Ecology Letters* 1 (1998), pp. 200–209.

[80] Amanda L Traud, Peter J Mucha, and Mason A Porter. "Social structure of Facebook networks". In: *Physica A: Statistical Mechanics and its Applications* 391.16 (2012), pp. 4165–4180.

[81] Jeffrey Travers and Stanley Milgram. "An experimental study of the small world problem". In: *Social networks*. Elsevier, 1977, pp. 179–197.

[82] Piet Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, 2010.

[83] Lav R Varshney, Beth L Chen, Eric Paniagua, David H Hall, and Dmitri B Chklovskii. "Structural properties of the Caenorhabditis elegans neuronal network". In: *PLoS computational biology* 7.2 (2011), e1001066.

[84] Olivier J Walther and Dimitris Christopoulos. "Islamic terrorism and the Malian rebellion". In: *Terrorism and Political Violence* 27.3 (2015), pp. 497–519.

[85] Gaoxia Wang, Yi Shen, and Ming Ouyang. "A vector partitioning approach to detecting community structure in complex networks". In: *Computers & Mathematics with Applications* 55.12 (2008), pp. 2746–2752.

[86] Wayne W Zachary. "An information flow model for conflict and fission in small groups". In: *Journal of anthropological research* 33.4 (1977), pp. 452–473.

[87] Pan Zhang and Cristopher Moore. "Scalable detection of statistically significant communities and hierarchies, using message passing for modularity". In: *Proceedings of the National Academy of Sciences* 111.51 (2014), pp. 18144–18149.

[88] Jiajing Zhu, Yongguo Liu, Yun Zhang, Xiaofeng Liu, Yonghua Xiao, Shidong Wang, and Xindong Wu. "Exploring anti-community structure in networks with application to incompatibility of traditional Chinese medicine". In: *Physica A: Statistical Mechanics and its Applications* 486 (2017), pp. 31–43.

[89] Philipp Zumstein. "Comparison of spectral methods through the adjacency matrix and the Laplacian of a graph". In: *TH Diploma, ETH Zürich* (2005).