DETECTING COMMUNITIES IN COMPLEX UNIPARTITE AND BIPARTITE NETWORKS BY MAXIMIZING THE MODULARITY

A Dissertation

Presented to

the Faculty of the Department of Physics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Santiago Treviño III

December 2013

DETECTING COMMUNITIES IN COMPLEX UNIPARTITE AND BIPARTITE NETWORKS BY MAXIMIZING THE MODULARITY

Santiago Treviño III

APPROVED:

Dr. Kevin E. Bassler, Chairman
Dept. of Physics

Dr. Gemunu Gunaratne
Dept. of Physics

Dr. Lowell Wood
Dept. of Physics

Dr. Wu-Pei Su
Dept. of Physics

Dr. Tim Cooper
Dept. of Biology and Biochemistry

Dr. Gábor Balázsi
Dept. of Systems Biology
M.D. Anderson Cancer Center
The University of Texas

Dean, College of Natural Sciences and Mathematics

First, I am sincerely grateful to my dissertation advisor, Dr. Kevin E. Bassler, for his guidance and support throughout my studies. I would also like to thank Dr. Bogdan Danila for his discussions, advice, and unwavering advocacy of Linux. I would like to thank Dr. Lowell Wood for his advice and discussions both on and off the topic of physics. I also gratefully acknowledge the advice and support received in my research from Dr. Charo I. Del Genio, Dr. Tim Cooper, Dr. Gábor Balázsi, Dr. Gemunu Gunaratne, and Dr. Wu-Pei Su. I appreciate the encouragement and discussions from the members of my research group Amy Nyberg and Shabnam Hossein, who now know more about my research than they would like. I would especially like to thank my wife, Shelley, who I told, "Don't worry, I'm just going to get my Master's degree," for her love, support, and encouragement. Finally, I would like to thank my family for instilling in me the importance of education through their love, support, and example.

DETECTING COMMUNITIES IN COMPLEX UNIPARTITE AND BIPARTITE NETWORKS BY MAXIMIZING THE MODULARITY

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Physics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Santiago Treviño III

December 2013

Abstract

This dissertation develops and improves methods to detect the modular structure of complex unipartite and bipartite networks using the method of modularity maximization, in which one seeks to find the partition of nodes that maximizes a quality function known as the modularity. Finding partitions that maximize modularity has been proven to be NP-hard and therefore, fast, approximate algorithms are developed to find high quality partitions.

First, novel methods are developed using an ensemble approach to detect and analyze multiple high quality modularity partitions to identify modular and hierarchical structure in networks. These methods are applied to the genetic regulatory network of *Escherichia coli* and a set of functional communities are identified and used to predict candidate regulatory interactions.

The problem of community detection in bipartite networks is then studied and a new bipartite modularity, the Information Preserving (IP) modularity, is introduced. The IP modularity detects communities in bipartite networks using a unimode projection that includes more information about the structure of the bipartite network in the projection than previous methods, and is applicable when the links in the bipartite network are weighted. The IP modularity is shown to detect meaningful structure in both a model network and the real-world metabolic network of *E. coli*.

Finally, the leading-eigenvalue method with final tuning is adapted to find the partition that maximizes bipartite modularity and an agglomerative step that merges communities is added to the algorithm. This step extends the effective applicability of the leading-eigenvalue method of modularity maximization to networks of tens of thousands of nodes in size. The methods developed in this dissertation significantly enhance the science of uncovering the structure in complex networks and are applicable to study a wide range of biological, medical, social and physical networks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Complex networks and graphs have increasingly become an important area of research as well as a tool for analysis in a wide range of disciplines [10, 11, 12]. A graph or network is simply a set of nodes and a set of the links that connect them. The study of graphs is said to have begun in 1736 when Leonard Euler analyzed the Königsberg bridge problem [13].

The Prussian city of Königsberg was built on opposite banks of the river Pregal which runs through the city where it splits and forms an island named Kneiphof [14]. There are seven bridges that connect each side of the river and the island. A question was posed whether an individual beginning at any point in the city is able to cross each bridge once and only once. Euler showed such a walk is not possible by reducing the problem to a list of the land masses and the bridges that connect them. Graph theory initially focused on small networks such as this one and has grown into a rich subject involving primarily the study of simple graphs such as regular graphs, latices, and random networks [15, 16].

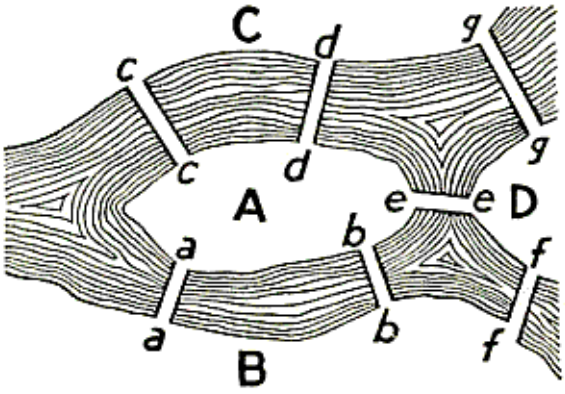
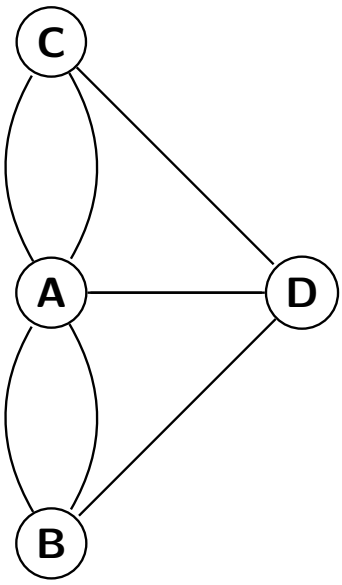


FIGURE 98. *Geographic Map: The Königsberg Bridges.*

(a)



(b)

Figure 1.1: **The Königsberg bridge**. This figure gives (a) a geographic representation of the city of Königsberg with each land mass labeled with an uppercase letter and the seven bridges connecting them whose ends are labeled with lowercase letters. This figure is from Ref. [1]. (b) a graph representation with each landmass represented by a node and each bridge represented by a link that connects two nodes.

Erdös and Rényi introduced and extensively studied the properties of a random graph model in which each pair of N nodes in a network are connected with probability p [17, 18, 19]. One of the most prominent features of the model is the emergence of a "giant component" in the network. It was found that when $pN < 1$ most of the nodes in the network are found in a collection of small clusters (whose size is of order $ln(N)$). However, as pN is increased, an abrupt change in the structure of the network occurs at $pN = 1$, at which value most (approximately $N^{2/3}$) of the nodes in the network are found in a single connected component. Additionally, it was found that, for all values of p and N, the number of connections adjacent to a node is Poisson distributed. This implies there exists a "typical" node in the network that is, on average, connected to pN other nodes.

Erdös and Rényi initially studied a random graph with the assumption that real-world graphs would behave similarly [20]. It was not until the 20th century, with the availability of and ability to obtain large network data sets, that real-world networks were found to have properties and topologies that are neither random nor regular but that can be described as complex [21]. Barabási and Albert found that in networks as diverse as the World Wide Web, actor collaboration networks, and the electrical power-grid of the western United States there is not a "typical" node in the network [22]. Unlike the the network of Erdös and Rényi they found that the distribution of the number of nodes a single node interacts with is power-law distributed. There exists in the same network many nodes with a few connections and a few nodes with a very large number of connections, but no connectivity is typical. Graphs exhibiting this power-law connection distribution are thus termed "scale free."

The Barabási Albert (BA) model was introduced at the same time to explain the

power-law degree distribution seen in real-world networks [20]. The two properties of the model, necessary to exhibit a scale-free topology, are growth and preferential attachment [20, 12]. The model begins with m_o nodes in the network and the network is grown by adding one node at each time step. Each time a node is added to the network it connects to the existing nodes with $m \leq m_o$ links. The probability of it connecting to a node i, that is currently in the network, is given by $k_i / \sum_j k_j$ where k_i is the total number of links connected to node i and $\sum_j$ sums over all current nodes in the network. Thus, a node added to the network is more likely (or prefers) to connect to a node with a large number of connections. The resulting graph exhibits a scale-free topology with a degree exponent of three [20].

The discovery that real-world networks exhibit behavior markedly different than random graphs fueled a renewed interest in the field of complex networks, particularly among physicists [23, 24, 25, 26]. It is now known that common properties a complex network may exhibit include a power-law degree distribution [27, 28, 29], sparsity [30], and community structure [31, 32, 33]. Community structure, the propensity of nodes in a network to form highly connected groups, is an area of extensive research [34].

Modular, or community, structure has been found in networks as diverse as the World-Wide Web (WWW) [35], the Internet [36], social networks [37, 38], food webs [39, 40], metabolic networks [41], and sexual contacts [42]. The fact that nodes in a network are found to be organized into modules is important, not only in its ability to reveal the large-scale structure of a network, but because these modules often correspond to functional units. For example, communities in the WWW have been found to be web pages organized by topic [35] and metabolic network communities have been found to relate to pathways and cycles [41, 43, 5]. In experimental science and exploratory research, detecting communities can aid in

uncovering an initial structure for the nodes of a network. These modules can then be used in making hypotheses about the properties and function of unclassified nodes in the network [5]. Additionally, the role a node plays is sometimes affected by its position within a community, with nodes at the boundary of communities both acting as "gate keepers" of information flow in social networks [44, 45, 46] and having a high conservation rate across species in metabolic networks [41]. Modular structure has even been found to affect dynamical processes on networks such as synchronization, percolation, and the spreading of epidemics [47, 48]. For all these reasons, detecting communities continues to be an area of extensive research.

Detecting modular structure is a non-trivial problem because a complex network is not simply a single homogeneous structure. A number of techniques, discussed later in this chapter, have been developed in the science community to detect modules in networks [34, 49, 33]. These techniques include hierarchical clustering [50, 51, 52, 53], partitional clustering [54, 55, 56], and spectral methods [57, 58]. The type of method this work focuses on is to define a quality function called the modularity that quantifies how "modular" a particular grouping of nodes is for a particular network. Modularity functions may be defined for a network depending on distinct features of the network. For example, a modularity function defined for networks where only one type of node exists, a unipartite network, is known as modularity "Q" [2]. Detecting communities then reduces to finding the grouping of nodes, called a partition, that maximizes the modularity function chosen.

Maximizing (or minimizing) a modularity function is related to a broad group of problems known as optimization problems [59]. For example, in physics, a well studied optimization problem is to find the ground state of a spin glass by minimizing it's Hamiltonian. A model was introduced by Edwards and Anderson that describes the

spin glass using an Ising model with a randomly distributed magnetic coupling [60]. The Hamiltonian of the EA model is written as,

$$H = -\sum_{<i,j>} J_{ij}\sigma_i\sigma_j - B\sum_i \sigma_i, \tag{1.1}$$

where σ_i is value of the spin at lattice i, J_{ij} is the strength of the magnetic coupling between spins, B is the value of the magnetic field, and $< i, j >$ indicates the sum runs over nearest neighbor spins. The value of J_{ij} is selected randomly from a probability distribution, for example, it can be selected randomly to be either 1 or -1. Finding the ground state of this spin glass is difficult due to the combination of competing ferromagnetic and anti-ferromagnetic Ising spins which results in many metastable states. For example, a spin may have a nearest neighbor whose coupling strength is equal to 1 and, at the same time, a nearest neighbor whose coupling strength is equal to -1. Therefore, it is impossible to choose a spin state that satisfies each neighbor's preference simultaneously. As will be discussed below in section 1.7.3, finding the partition that maximizes modularity Q has been shown to be equivalent to minimizing the Hamiltonian of a generalized Ising model, the Potts model [61]. Therefore results and optimization techniques in community detection may have broad application in understanding optimization problems from other areas of science.

Finding the grouping of nodes that maximizes modularity is a particularly hard problem to solve. A way to classify how difficult a problem is is by determining the time complexity of the algorithm that finds its solution. The time complexity of an algorithm is the worst-case estimate of how the time required to complete the algorithm scales with the size of the system. For example, one solution is to simply survey all the possible combinations of nodes in the network and choose the grouping

giving the largest modularity. The number of different possible groupings of N nodes is given by the Bell number [62], which grows faster than exponentially with N. Thus as the size of the network grows it quickly becomes impossible to computationally survey all the possible groupings of nodes in the network.

Problems whose solution runs in polynomial time using a deterministic Turing machine are said to belong to the class P [63, 64]. Similarly, if a problem can be solved by an algorithm that runs in polynomial time on a non-deterministic Turing machine the problem belongs to the class NP [63]. Unfortunately, finding the partition that maximizes modularity Q has been proven to be NP-hard which means it is at least as difficult as the hardest problems in NP [65]. Therefore, finding the exact solution that maximizes a modularity function for a particular network may be an effectively impossible computational task, and approximate algorithms must be used that balance finding high quality partitions with their computational complexity.

This dissertation develops and improves fast, approximate methods to study and detect communities in complex unipartite and bipartite networks. The remainder of the chapter begins with a mathematical description of the terms and concepts that will be used throughout this dissertation. Properties of the community structure in real-world networks are then discussed with example networks from the literature. An overview of the current methods used in network science to detect communities are discussed ending with a description of modularity Q. Modularity Q, its properties, and methods to maximize the modularity are reviewed. This includes a discussion of current methods to extend modularity definitions to bipartite networks. Finally, an outline of the remainder of the dissertation is given at the end of this chapter.

1.1 Network definitions

A graph $G(V, E)$ or network is simply the set V of N nodes or vertices and the list E of m edges or links that connect them [66, 67]. An example network of 6 nodes connected by 8 links is represented in Fig. 1.2a. This is an example of a *simple graph*, a graph where each pair of nodes is connected by at most one link and that contains no *self-loops*, links connecting a node to itself. A matrix representation of the connections in a network is called the *adjacency matrix* $\mathbf{A}$, where the element of the matrix $A_{i,j}$ is equal to the value of the link or links in the graph between node i and node j. The adjacency matrix for the network in Fig. 1.2a is

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}. \tag{1.2}$$

The *degree* d_i of node i is defined as the number of edges adjacent to it [66]. In terms of the adjacency matrix the degree of node i is given by $d_i = \sum_j A_{ij}$. The vector $\mathbf{d}$, whose element d_i represents the degree of node i in the network of Fig. 1.2a

is

$$\mathbf{d} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 2 \\ 2 \\ 3 \end{bmatrix}. \tag{1.3}$$

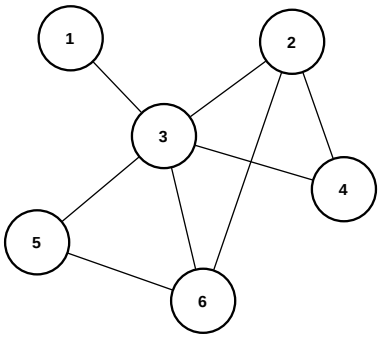
As discussed thus far, the only information a link represents is if a pair of nodes are connected. However, links may carry additional information [68]. A *multi-graph* is a graph where each pair of nodes may be connected by more than one link. Such node connections are called *multi-links* with additional links indicating a stronger connection. An example of a multi-graph is shown in Figure 1.2b, with three links connecting node 5 to 6, and two links connecting both node 1 to 3 and node 2 to 6. A multi-link between nodes i and j in the adjacency matrix is represented by an integer value equal to the number of links. A generalization of this idea is a *weighted* graph whose links take on any real value. Negative links in this case could represent, in the context of a social graph, animosity [68]. Finally, links in a *directed* graph point from node i to node j to represent interaction in one direction. For example, hyper-links, the word, phrase, or image that allow users to click on and jump to a web page in the WWW, point from web page A to web page B. However, there is not necessarily a hyper-link from web page B back to web page A. In a directed graph a link from node i to node j is again represented by the element A_{ij} in the adjacency matrix. In this case, the adjacency matrix is not symmetric and two different types of degrees can be defined for a node. The number of out going links adjacent to node i is defined as the out-degree and the number of incoming links adjacent to node i is

defined as the in-degree.

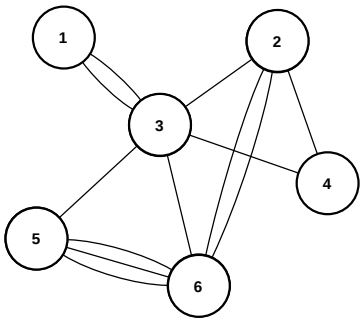
A graph can also be classified by the number of node types that it contains [66]. Two common types are *unipartite* graphs, in which there is only one type of node such as in Figures 1.2a and 1.2b, and *bipartite* graphs in which there are two types of nodes. A bipartite network has an additional property where each link in the network only connects a node from each type. For example, Fig. 1.2c is a weighted bipartite graph where the two types of nodes are represented by the shape of the node with circles representing type one and squares representing type two. Additionally, the links in this graph are weighted with the increasing width representing increasing weight. Thus, the strongest connection represented in this graph is between node 4 and 6. A generalization of a bipartite graph is a *multipartite* graph which contains many type of nodes where links connect nodes of different type and a link never exists between two nodes of the same type.

1.1.1 Network partitions

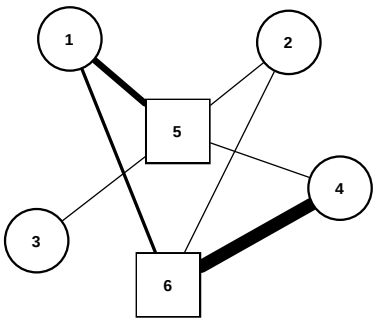
The objective of community detection methods is to assign each of the N nodes of the network to one or in some cases a number of k communities that defines a network partition $P(G)$. For a set of k communities $C = \{C_1, C_2, .., C_k\}$ a community detection method that assigns each node into multiple communities is known as an over-lapping community detection method [5]. The partition $P(G)$ can then be represented by an $N \times k$ matrix with element P_{ij} equal to the amount of membership node i is assigned to community C_j. Often, P_{ij} is normalized so that $\sum_{j=1}^{k} P_{ij} = 1$ for each node i [69]. Alternatively, in other community detection methods each node is only assigned to one community in which case $P(G)$ is a vector of length N with



(a)



(b)



(c)

Figure 1.2: **Network Examples**. This figure gives examples of (a) a simple graph; a graph whose link values are binary and that contains no self-loops, (b) a multi-graph; a graph that may contain any integer number of links between nodes , and (c) a weighted bipartite graph; a graph in which there are two types of nodes and each weighted link in the network only connects a node from each type. The weight of a link is indicated by the line thickness.

P_i equal to an integer, 1 through k, representing the community number that node i is assigned to.

Additionally, a community detection method may not output one community partitioning but a set of nested partitions organized into a dendrogram [50, 51, 52, 53]. A dendrogram is a graph showing the clusters of a set of objects, with the following classes of vertices [70]:

1. vertices of degree 1, corresponding to objects;
2. vertices of degree greater than 2, called intermediate objects;
3. only one vertex of degree 2, called the root object.

Figure 1.3 gives an example of such a dendrogram in which there are 12 nodes. At the bottom level of the dendrogram are the N nodes (objects) of the network each beginning in a single community. Pairs of nodes are merged together into the same community at each horizontal level beginning from the bottom up where each merge is represented by an intermediate object. The vertical axis corresponds to the similarity value at which the corresponding pair of nodes are merged with lower values on the axis representing stronger similarity. Similarity measures are further discussed in section 1.4.1. The horizontal line at the top of the dendrogram represents the root object (imagine a vertex at the center of the line) and corresponds to all of the nodes in the network partitioned into one community. Then, each level of the dendrogram represents a particular community partition that is chosen by making a horizontal cut in the dendrogram. For example, if the cut represented by the horizontal red line in Fig 1.3 is chosen the resulting partition will contain 4 communities. Numbering the nodes in order from left to right the community

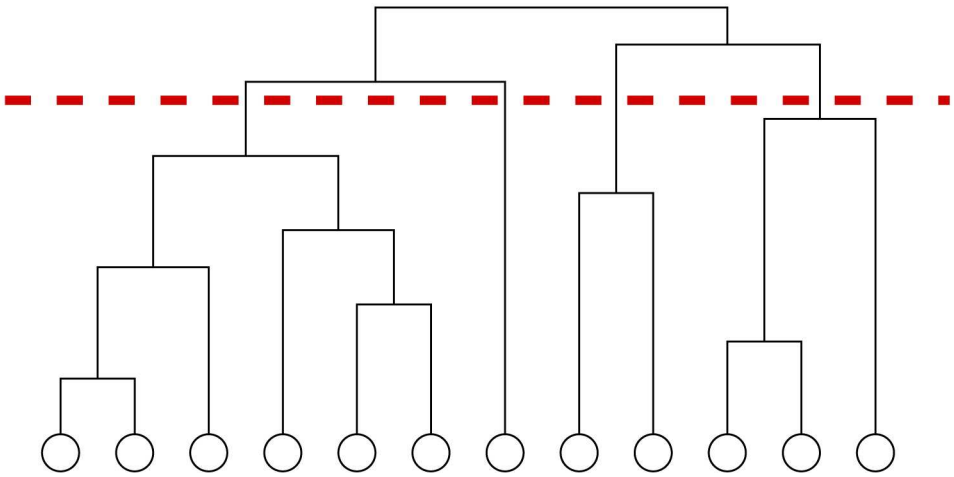


Figure 1.3: **Dendrogram**. Hierarchical methods produce a partitional structure called a dendrogram. Each node in the network begins at the bottom level of the dendrogram in its own community called a singleton cluster. Pairs of nodes are merged at each level and a partition is chosen by making a horizontal cut in the dendrogram indicated here by the red dotted line. The vertical axis corresponds to the similarity value at which the pair of nodes were merged with lower values on the axis representing stronger similarity. Figure from Ref. [2]

partition is then $P = \{1, 1, 1, 1, 1, 1, 2, 3, 3, 4, 4, 4\}$.

1.2 Biological networks

Biological functions and processes often result from the interactions between molecules. Broadly speaking there are two approaches involving network biology currently in the literature [71, 72, 73]. A "bottoms up" approach involves building small networks of molecules in an attempt to model small biological circuits [74, 75]. This may involve for example, the modeling of small genetic circuits using differential equations to determine decay rates, production rates, and interaction strengths. A "top down" approach, on the other hand, focuses on using high-throughput "omics" data to create and study the overall structure of a biological network [76]. A top down network representation focuses on the large-scale interactions within a biological system and can give novel insight into its processes and functions. The following discussion will focus on this top down approach in which a network representation is regularly used. Commonly studied networks include protein-protein interaction networks, genetic regulatory networks, and metabolic networks [77]. The following focuses on the description of the interactions of a biological system at the gene level and separately at the metabolic level. General properties of cellular networks is then discussed.

1.2.1 Genetic networks

In a genetic network the nodes of the network represent genes and links represent the direct or indirect regulatory interactions between them. A simplified description

of the gene regulatory process is given in the, so called, *central dogma of molecular biology* [78, 79]. According to the dogma, all of the genetic information in a cell is stored in the cell's DNA and is encoded using four nucleotides which contain the bases adenine, guanine, cytosine, and thymine. Although the DNA carries all of the information in the cell, the majority of its structure, functions, and processes are carried out by proteins. A protein is synthesized by the cell in a two step process (Fig. 1.4). First, the double stranded DNA is transcribed into a single stranded messenger RNA (mRNA) molecule by an RNA polymerase. The mRNA is composed of the same nucleotide bases as in DNA except thymine is replaced by the nucleotide uracil. The mRNA is then translated into a protein by a ribosome. Regulation can occur when one or more regulatory proteins bind to the DNA and affect the transcription rate of the RNA polymerase and thus the amount of gene "expressed." The amount of a gene expressed for a specific experimental condition can be measured using DNA microarrays.

DNA microarrays are able to measure the expression of tens of thousands of genes in a single experiment [80, 81]. The principle behind DNA microarray experiments is DNA hybridization where nucleotide sequences in DNA specifically pair with one another through hydrogen bonds between nucleotide base pairs so that adenine pairs with thymine and guanine pairs with cytosine [82]. DNA microarrays consist of ordered spots each filled with multiple copies of complementary DNA(cDNA) whose nucleotide sequence is specific to one gene called the probe. RNA from a sample is fluorescently labeled by reverse transcription and these fluorescent targets are allowed to hybridize to the probes on the array. The amount of transcript present is found by measuring the intensity of the fluorescent signal generated using laser excitation. Additionally, a "reference" sample labeled with a different fluorophore

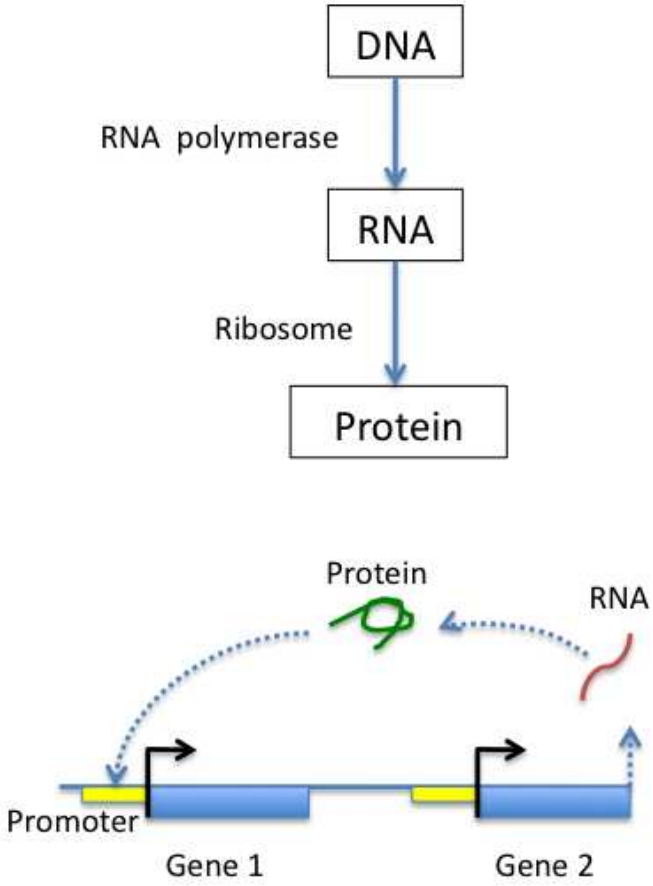


Figure 1.4: **The central dogma of microbiology**. The central dogma of microbiology describes the flow of genetic information in a cell. The DNA is transcribed into RNA by an RNA polymerase which is translated into a protein by the ribosome. The transcription rate of the RNA polymerase can be affected when proteins bind to a site on the DNA called the promoter. In this figure the promoter is labeled yellow, the segment of DNA translated is labeled blue, and the start site of translation is labeled by the black arrow.

may be incorporated into the array [83, 84]. This acts as an internal control by granting the ability to compare relative abundance between the test and reference sample.

The increasing availability of gene expression data has spurred development of a number of approaches that aim to determine the underlying structure of the transcriptional regulatory network [85, 86, 87, 88, 89, 90, 91, 92]. Most of these techniques fall into the broad categories of correlation-based methods, information-theoretic methods, Bayesian network predictions, or methods based on dynamical models. These approaches generally infer regulatory links between the nodes (genes) of the network on the basis of the level of correlation in their transcriptional response to a series of environmental and genetic perturbations. The strength of the links is either weighted by the correlation value, or is unweighted and the links are assumed to exist only if the correlation exceeds a threshold value. Once the links are assigned, the network becomes well defined. However, variation in the application of each method can produce differences in the link weight between pairs of nodes. Additionally, if the threshold for placing links is varied even slightly there can be significant differences in the network structure inferred from a given data set [93]. In Chapter 3, novel methods to infer regulatory interactions between genes that address these problems are introduced.

1.2.2 Metabolic networks

A metabolic network represents the chemical reactions in an organism by a set of nodes representing an enzyme or metabolite and a set of links that connect an enzyme to a metabolite if that enzyme catalyzes a reaction the metabolite is involved in [94,

95]. Metabolic chemical reactions involve the management of energy and materials within the cell and in general involve the breaking down or building up of complicated molecules. The set of chemical reactions that breakdown (build up) molecules are referred to as catabolic (anabolic) pathways. A chemical reaction can be represented in equation form by

$$\text{Substrate(s)} \xrightarrow{\text{Enzyme}} \text{Product(s)}, \tag{1.4}$$

whose constituents are the substrate(s), product(s), and enzymes. The chemical substrate(s) and product(s) are known as metabolites and are the input and outputs of the reaction. Also involved in metabolic reactions are enzymes which catalyze the reaction by binding to specific metabolites called the substrate(s). The formation of this enzyme-substrate complex provides a means to lower the activation energy and effect the speed of the reaction through various mechanisms such as providing a conducive micro-environment for the reaction as well as providing a template for orientation of the substrate [95]. The majority of metabolic reactions are reversible with an enzyme able to the catalyze the reaction in either direction.

An example of a metabolic reaction involving the hydrolysis of sucrose to glucose and fructose is given by the equation

$$\text{Sucrose} + \text{H}_2\text{O} \xrightarrow{\text{Sucrase}} \text{Glucose} + \text{Fructose}. \tag{1.5}$$

The enzyme that catalyzes this reaction is called Sucrase. Sucrose binds to the Sucrase protein at a specific site on the enzyme called an active site. The enzyme-substrate complex facilitates the reaction until the substrate is converted to products and Glucose and Fructose are released. Metabolic networks are examples of bipartite networks with one set of nodes representing the metabolites and the other the enzymes. An example of Eqn. 1.5 represented as a bipartite network is given in

Fig. 1.5 where a link connects an enzyme to a metabolite if that enzyme catalyzes a reaction in which the metabolite is involved. Other representations exist, which will be discussed below, that project the bipartite network on to a unipartite network with only one set of nodes represented in the projection. For example, nodes may represent only metabolites with links connecting substrates to products.

Constructing a metabolic network involves a multi-step process to determine the chemical components and transformations in an organism [96, 97]. Initially, annotated genomes are used to identify metabolic enzymes and how these enzymes interact to catalyze reactions. Annotated genomes are found in databases such as EcoCyc [98], SYG (*Saccharomyces* Genome Database) [99], and CyGD (Comprehensive Yeast Genome Database) [100]. Next, those reactions catalyzed by enzymes are automatically and manually curated from databases such as KEGG [101], Metacyc [102], and Transport DB [103] as well as other sources such as research literature and comparative genomics. This is followed by the network's conversion to a computational model, such as a stoichiometric representation [104], which facilitates further evaluation of the model such as its ability to support growth and the known synthesis of amino-acids. Throughout this process metabolic pathways that are incomplete or missing are filled in or added to the network. In Chapter 4, the metabolic network of *Escherichia coli* is studied to compare, test, and validate bipartite modularities.

1.2.3 Biological network properties

A general feature of biological networks is a scale-free topology. Studies involving metabolic networks [105, 106], gene networks [107, 108], and protein-protein networks [109] have all found scale-free topologies. As discussed, the BA model

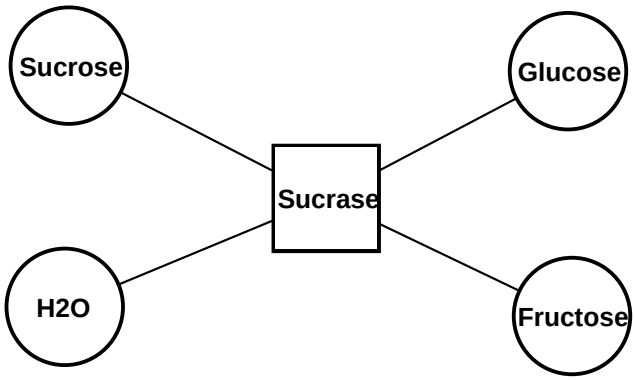


Figure 1.5: **Bipartite representation of a metabolic reaction**. A bipartite network representation of Eqn. 1.5 with metabolites represented as circle shaped nodes and enzymes represented as square shaped nodes. A link connects an enzyme to a metabolite if the enzyme catalyzes a reaction the metabolite is involved in.

indicates that a scale-free topology stems from two basic mechanisms; growth and preferential attachment. This suggests that the hubs in cellular networks may be some of the oldest constituents of the network from an evolutionary standpoint. For example, in metabolic networks coenzyme A, NAD and GTP are among the hub metabolites [106] which are remnants of the RNA world [110]. Other mechanisms have been suggested to produce scale-free topology in networks such as the biological mechanism of gene duplication [111, 112, 113, 114, 115, 116, 117, 118], when one or several genes are randomly copied twice in the genome during cell division. Additionally, the efficiency of transport and flow processes across networks has been tied to the emergence of a scale-free topology [119, 120].

Cellular systems are intrinsically modular. Groups of molecules can interact depending on their spatial location in the cell, temporally depending on the current stage of a cell cycle, as well as chemically for example during signal transduction [121]. There have been many analyses of the modular structure in cellular networks [121, 122, 75]. Modules have been found to correspond to discrete functional units in protein-protein networks [123, 124], gene-regulatory networks [125, 126], as well as metabolic networks [127]. These modules can be conserved during evolution [128, 129]. Additionally, biological networks have been found to be arranged hierarchically [130, 131, 127]. This suggests there is not a set of unique modules that define a biological network but modules that work in both isolation and combination at various levels of the hierarchy. A significant portion of this work focuses on developing and improving methods to infer the modular organization of genetic and metabolic networks. Methods that take into account the variability in microarray data, genetic network reconstruction, and modular hierarchical network properties will be studied in Chapter 2. In Chapter 4, the problem of community detection

in metabolic networks will be studied specifically addressing the bipartite nature of the network. However, a discussion of real-world networks is not complete with out examples of the application of network theory in other areas of science.

1.3 Other real-world networks

1.3.1 Social networks

Networks representing the social interactions between people are also of interest [132, 133]. Community detection has its roots in the social sciences. Each of us is a member of a social community with, for example, family members, co-workers, and friends. A widely used and studied example of a social network that exhibits community structure is Zachary's karate club [134]. Zachary studied the social interactions of members of a university karate club over three years. These interactions are represented as a network shown in Fig. 1.7. Each of the 34 nodes in the network represent a member of the group. A link is drawn between two nodes if the corresponding members were observed to interact outside of the club activities. This graph exhibits clear community structure with two groups each centered around nodes with a large number of links. A dispute between the club president and instructor caused the karate club to split, represented by the dotted line, into two separate organizations. The ability to predict the club fissure based solely on the knowledge of the links in the network is a standard test for community detection algorithms.

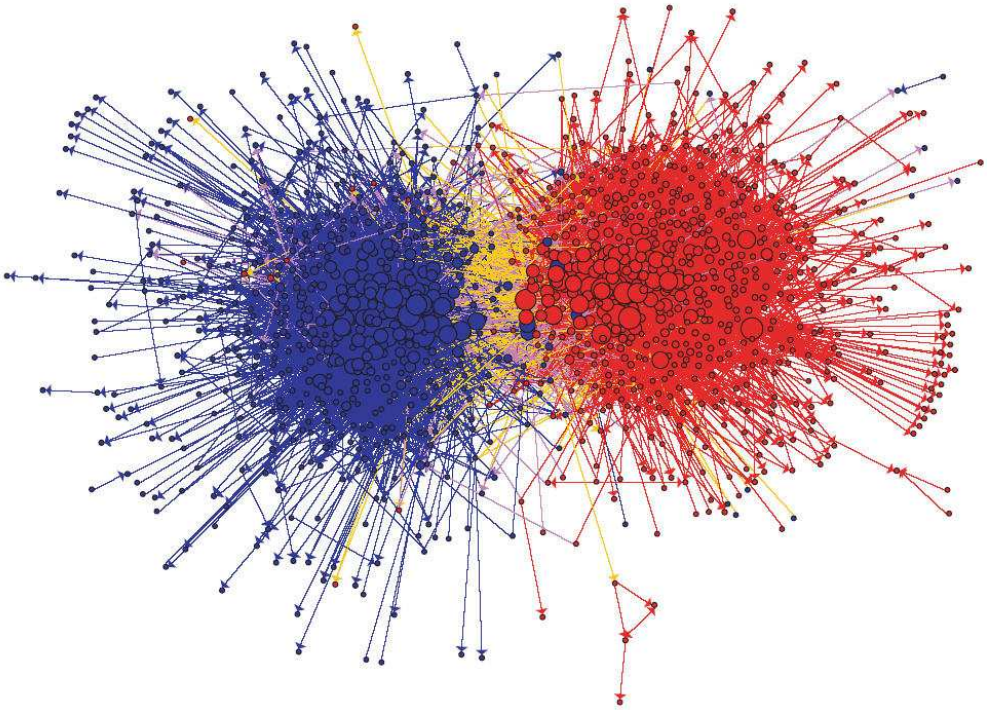


Figure 1.6: **An information network**. A network of hyperlinks between political web blogs in 2005. The two large communities can be classified by political orientation as liberal (blue) for the cluster on the left and conservative (red) on the right. Yellow links indicate a hyperlink between a conservative and liberal web blog. Additionally, the size of the node corresponds to the number of web-pages that link to it. Figure from Ref. [3]

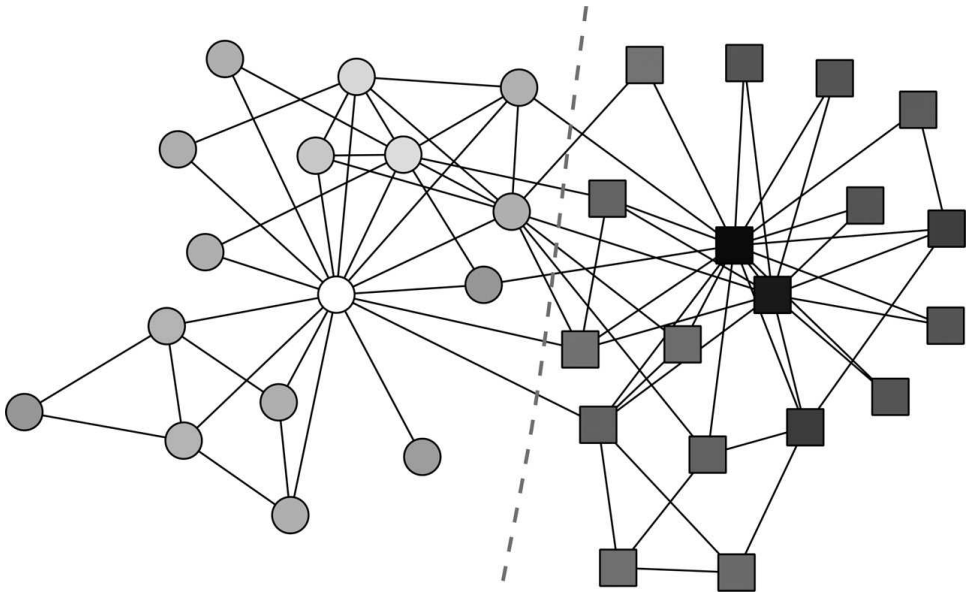


Figure 1.7: **Zachary's karate club network**. The nodes represent 34 members of a university karate club group and a link is placed between a pair of nodes if they were observed to interact outside of the club. A dispute within the club caused it to split represented by the dotted line forming two separate communities. Community membership is indicated by the shape of the node and the gray scale represents strength of membership with the black(white) corresponding to the strongest membership of the group on the right(left). This figure is from Ref. [4].

1.3.2 Information networks

The World Wide Web (WWW) is an example of an information network in which the nodes represent a web page and a *directed* link points from page A to page B if there exists a hyperlink from page A to page B. Alternatively, one may study the structure of the Internet with the nodes representing routers and computers and links the physical interactions (cables) between them. [135, 136, 137]. Fig. 1.6 represents a WWW network of hyperlinks between 1,224 political web blogs in 2005 [3]. From the figure, it is clear there are two large communities whose political orientation was classified as liberal for the blue cluster on the left and conservative for the red on the right. Often a link between two web pages is assumed to represent subject similarity and the link direction is ignored. The topology of the world wide web has been well studied [138, 139, 140, 141] with one of the most obvious features being the immense size of the network.

1.4 Methods to detect communities

While there are many methods to detect communities [34, 33, 142], the objective of a community or clustering algorithm is generally the same. Given a list of the N nodes and m connections between these nodes, community detection methods seek to assign each node of the network into a set of communities. Methods then fall under one of two broad categories, hierarchical clustering or partitional clustering, based on the number and structure of the partition output. Hierarchical clustering results in a tree of nested partitions while partitional clustering results in a single partition of the network that seeks to maximize or minimize a quality function. Additionally,

methods exist that assign a node to multiple communities which is known as "fuzzy clustering" [143, 144, 145, 69] or overlap methods [5].

In general, there is always a trade off between the quality of the partitions an algorithm is able to find and the length of time required to complete the algorithm. The time complexity of an algorithm, represented as $O(N^{\alpha}\ m^{\beta})$, is the worst-case estimate of how the time required to complete the algorithm scales with the number of nodes in the network, N , as well as the number of links, m. One would like an algorithm to scale with the lowest exponents of α and β possible. For example, very large networks, with nodes on the order of millions and links on the order of billions, are only possible to analyze with algorithms that scale as O(N) or O(m). Due to the difficulty of the problem one often seeks a fast, accurate approximate method.

1.4.1 Hierarchical clustering

Hierarchical clustering [50, 51, 52, 53] begins by assigning a distance value between each pair of N nodes in the network based on a "similarity" value [146]. For example, if the nodes in a network are each attributed n properties then two nodes, X and Y, can be treated as data-points in n-space. Then given $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$ one could use a norm such as the Euclidean distance,

$$d_{AB} = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}. \tag{1.6}$$

Alternatively, similarity measures exist based on structural equivalence such as the similarity of adjacency values between vertices [132, 45]. The resulting $N \times N$ similarity matrix is then used to create a tree of nested partitions (Fig. 1.3). The most common method to create this dendrogram is *agglomerative* hierarchical clustering

where each node in the network begins as its own community, called a *singleton* cluster, placed at the bottom of the dendrogram. Each pair or group of clusters with the smallest similarity value are merged into a community. This merger is represented as a branch on the dendrogram with a height corresponding to the similarity value. The similarity matrix is then reduced in dimension and a new similarity value between the new merged cluster and all other existing clusters is calculated. There are many ways to assign this new distance value, however the most common methods are are called single linkage, average linkage, and complete linkage clustering. This process is repeated iteratively until all nodes are merged into a single community at the top of the dendrogram. Alternatively, in *divisive* hierarchical clustering the nodes begin in one community and are iteratively divided. In either case the result is a hierarchical tree of nested partitions that must be cut to obtain a specific partition of the network. This cut is often chosen when there is a large jump in the height of cluster mergings.

Hierarchical clustering is advantageous for the fact that a prior knowledge of the number and size of the communities in the network is not needed. In general the time complexity of hierarchical clustering algorithms run in $\mathrm{O}(N^3)$ but there are efficient agglomerative algorithms that run in $\mathrm{O}(N^2)$ [147, 148]. However, the output of the algorithm is always a hierarchically nested set of partitions which does not depend on whether the network is itself hierarchically structured. It is not necessarily obvious which cut gives the best partition of the network. Furthermore, traditional similarity measures such as the Euclidean distance require the network to be embedded in an n-dimensional space which is not necessarily straight-forward when the only node properties are its connections.

1.4.2 Partitional clustering

Partitional clustering is based on maximizing or minimizing a quality function and results in a single partition of the network. These include clustering methods such as graph partitioning [54, 55] and k-means clustering [56]. In graph partitioning one seeks to divide the nodes in a network into g groups of predefined size by minimizing the number of links between the g groups, called the cut size. A commonly used method to minimize the cut size is to initially bisect the graph using the Kernighan-Lin algorithm [149]. Each community is then iteratively bisected. K-means clustering requires the network to be embedded in n-dimensional space. To partition the network into k groups, k "centroids" are distributed in the n space and nodes are assigned to the centroids so as to minimize the total intra-cluster distance defined as

$$\sum_{i=1}^{k} \sum_{\mathbf{x_j} \in \mathbf{S_i}} ||\mathbf{x_j} - \mathbf{c_i}||^{\mathbf{2}}, \tag{1.7}$$

where S_i is the subset of points of the i^{th} cluster, $\mathbf{x_i}$ is the position vector of node i, and $\mathbf{c_j}$ is the position vector of centroid j. To minimize intra-cluster distance a commonly used algorithm is Lloyd's algorithm [150]. Lloyd's algorithm begins by distributing the k centroids as far as possible from each other within the n-dimensional space and assigning each node to the nearest centroid which defines k clusters. Next, the center of mass for each cluster is calculated and this becomes the new position of its centroid. The nodes are then reassigned to the nearest centroid. This procedure is iterated until the position of each centroid is stable and the cluster assignments do not change. The k-means algorithm has a time complexity of O(N) however the solution depends on the initial positions of the centroids [142]. Therefore, often one runs the algorithms many times with varying initial conditions choosing the solution with the smallest intra-cluster distance [34].

Partitional clustering is advantageous because algorithms typically have lower time complexities than hierarchical clustering algorithms. Additionally, this method defines a quality function that quantifies the quality of a partition and allows for the comparison of multiple partitions. A search for the partition that optimizes a quality function can be computationally prohibitive and often multiple runs of an algorithm are necessary to find a solution. Additionally, some partitioning methods such as k-means and graph partitioning are not ideal for community detection in networks due to the requirement that the number of communities must be an input of the algorithm. However, with its ability to quantify and explicitly incorporate a community definition into its quality function, partitional clustering has the advantage of being able to specifically tailor a quality function for community detection in complex networks. A quality function that has been introduced for this purpose will be discussed in section 1.4.5.

1.4.3 Spectral clustering

Spectral clustering methods include any method or technique that uses the eigenvectors of a matrix, such as a similarity matrix, to partition nodes. Historically, spectral clustering was developed in graph partitioning algorithms for parallel computing in which a computation involved N tasks with M communications between the tasks [57, 58]. For computations involving two processors, one would like divide the N tasks (nodes) between the two processors so as to minimize the number of communications (links) between processors called the cut size, R. One can represent R in equation form as

$$R = \frac{1}{2} \sum_{i,j} A_{ij} \delta_{P(i)P(j)}, \tag{1.8}$$

where A_{ij} is the adjacency matrix, P(i) is processor node i is assigned to, and $\delta_{P(i)P(j)}$ is the Kronecker delta function equal to 1 if $P(i)$ is equal to $P(j)$ and 0 otherwise. Then, by defining a vector $\mathbf{s}$ of length N whose element s_i is 1 if task i belongs to processor one or -1 if task i belongs to processor two, the Kronecker delta function can be written as $\delta_{P(i)P(j)} = \frac{1}{2}(1 - s_i s_j)$. This can be used to write R as

$$R = \frac{1}{4}\sum_{ij}(1 - s_i s_j)A_{ij} = \frac{1}{4}\sum_{ij} s_i s_j(k_i\delta_{ij} - A_{ij}), \tag{1.9}$$

where the relation, $\sum_{ij} A_{ij} = \sum_i k_i = \sum_i s_i^2 k_i = \sum_{ij} s_i s_j k_i \delta_{ij}$, was used and k_i is the degree of node i. Equation 1.9 can be written in matrix form as

$$R = \frac{1}{4}\mathbf{s^T L s}, \tag{1.10}$$

where $\mathbf{L} = \mathbf{k_i}\delta_{\mathbf{ij}} - \mathbf{A_{ij}}$ is known as the Laplacian matrix. One can use the eigenvectors of the Laplacian matrix $\mathbf{v_i}$ to minimize R by writing $\mathbf{s}$ as a linear combination of these eigenvectors, $\mathbf{s} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{a_i v_i}$ so that Eqn. 1.10 becomes

$$R = \sum_i a_i^2 \lambda_i, \tag{1.11}$$

where λ_i is the eigenvalue of $\mathbf{L}$ corresponding to $\mathbf{v_i}$. Then, to minimize R one can make $\mathbf{s}$ as close to parallel to the eigenvalue v_1 of $\mathbf{L}$ as possible corresponding to the smallest eigenvalue λ_1. Because $s_i \in \{-1, 1\}$ one can make the assignment

$$s_i = \begin{cases} 1 & \text{if } u_i \geq 0 \\ -1 & \text{if } u_i < 0 \end{cases}. \tag{1.12}$$

Unfortunately, the smallest eigenvalue of the Laplacian is 0 with corresponding eigenvector $v_1 = \{1, 1, 1, ..., 1\}$. This places all tasks into one processor which is trivial. A solution to this problem is to fix the number of tasks allocated to each processor and use the next lowest eigenvalue of the Laplacian [151]. For applications

requiring further division of tasks or nodes, one can then recursively bisect each of the two components to further divide the network. However, minimizing the cut size based on the Laplacian is not suited to community detection in networks due to the requirement that the number of communities and group sizes be an input of the method. Fortunately, as will be discussed, spectral methods can be adapted to modularity matrices whose properties are better suited for community detection.

1.4.4 Other methods

As discussed, methods exist that assign nodes into multiple communities. A well known method to find overlapping communities is the k-clique percolation method by Palla *et al.* [5]. To find communities a k-clique is defined as a set of k nodes that are completely connected. Two k-cliques are defined as being adjacent if they share $k-1$ nodes. Then a k-clique community is the union of all k-cliques that can be reached through adjacent k-cliques. For example, in Fig. 1.8 an example network with k-clique communities when $k = 4$ is shown with a number of nodes belonging to more than one community. One of the fastest algorithms to find k-clique communities is the Sequential Clique Percolation algorithm [152] whose time complexity is network dependent scaling linearly with the number of k-cliques. A k-clique can sometimes be a rather strict criterion for community detection with many nodes that only have single connections not belonging to any k-clique communities. Additionally, there is no criterion that requires connections between nonadjacent k-cliques in a k-clique community. This may not necessarily conform to the notion of a community as a group of densely connected nodes. Nevertheless, k-clique percolation continues to be a standard method when considering overlapping communities.

A novel method by Raghaven *et al.* [153] finds communities in a network by propagating "labels" through the network. The method begins by assigning each of the N nodes in the network a unique label representing a community. At each time step a node is chosen randomly and it is assigned the label currently possessed by the majority of its neighbors. If there is not a majority, a label is assigned to the node randomly from the labels constituting the tie. This is repeated until the label assignments are stable, defined as the point where each node in the network has the same label as the majority of its neighbors. The communities are then defined by grouping nodes together with the same label. The algorithm's time complexity is very fast scaling as O(m). Initially, the authors sought a community method that required no optimization or defined quality function but only required the structure of the network. It has since been shown that the label propagation method is equivalent to finding the local energy minima for the Hamiltonian of a q-state Potts model given by

$$H = -\sum_{ij} A_{ij}\delta_{\sigma_i\sigma_j}, \tag{1.13}$$

where A_{ij} is the adjacency matrix and σ_i is the label of node i. The number of such minima is larger than the number of nodes in the network [154]. The global minima is in fact the trivial solution of all nodes carrying the same label [155] and therefore is not an ideal objective function. There have been modifications to the label propagation algorithm [155]. For example, by assigning a strength to each label at its origin node which decreases as the label propagates [156].

A number of methods focus on statistical inference-based models that fit a set of parameters to observed network data [157, 158, 159, 160]. For example Ball *et al.* [161] have develop a method to detect overlapping modules that seeks to partition

the links of a network into K communities called "colors". Given K colors, Θ_{iz} is defined as the tendency of vertex i to have edges of color z. $\Theta_{iz}\Theta_{jz}$ is then the expected number of edges of color z that connect node i to node j which is expected to be Poisson distributed. To partition the links into colors, the method seeks to find the parameters Θ that maximize the probability of generating a graph G with adjacency elements A_{ij} where the probability of generation such a graph is given by

$$P(G|\Theta) = \prod_{i<j} \frac{(\sum_z \Theta_{iz}\Theta_{jz})^{A_{ij}}}{A_{ij}!} \times \prod_i \frac{(\frac{1}{2}\sum_z \Theta_{iz}\Theta_{iz})^{A_{ii}/2}}{A_{ii}/2!}, \tag{1.14}$$

where $\sum_z \Theta_{iz}\Theta_{jz}$ is the expected number of links between node i and j of any color and the factor of 1/2 occurs in the last product of the equation to account for over-counting when considering self-loops. The authors show that maximizing Eqn. 1.14 is equivalent to simultaneously solving the two equations,

$$q_{ij}(z) = \frac{\Theta_{iz}\Theta_{jz}}{\sum_z \Theta_{iz}\Theta_{jz}} \tag{1.15}$$

and

$$\Theta_{iz} = \frac{\sum_j A_{ij} q_{ij}(z)}{\sqrt{\sum_{ij} A_{ij} q_{ij}(z)}}, \tag{1.16}$$

which can be done beginning with random initial conditions using iteration. $q_{ij}(z)$ is the probability that an edge between node i and j has color z and satisfies the normalization condition $\sum_z q_{ij}(z) = 1$. When these parameters are determined, node i is then assigned to community z by the fraction of its adjacent links assigned to community z. In this way a node can be assigned to more than one community based on the community assignment of the links adjacent to it. The complexity of the algorithm is $O(mK)$ and typically converges quickly *able* to partition networks of up to 10^6 nodes. Additionally, the algorithm performs well on computer generated

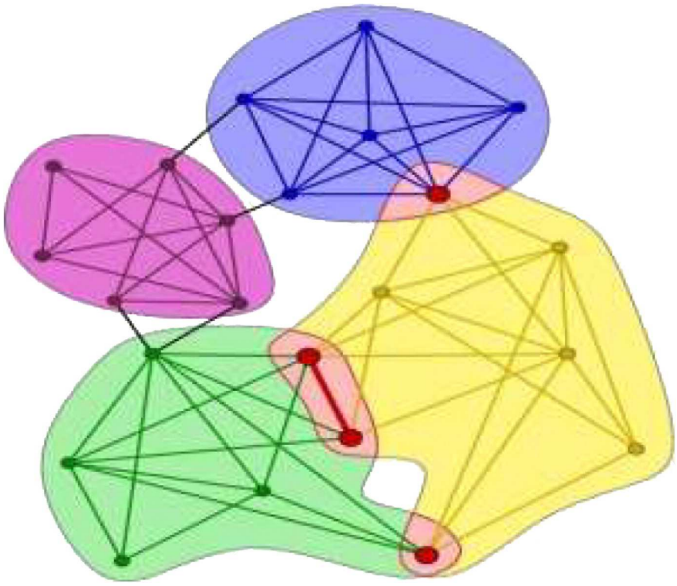


Figure 1.8: **Overlapping communities**.An example network partitioned using the k-clique percolation method where $k = 4$. Communities are shaded by color and nodes belonging to more than one community are labeled red. From Ref. [5]

networks in which the number of communities is known. Unfortunately, for real-world networks the number of communities is not known beforehand and this limits the applicability of these types of methods.

A recent hierarchical clustering algorithm by Seemann *et al.* [162] has been introduced to classify cancer subtypes based on high-throughput genomics using persistent homology [163]. The algorithm begins by selecting a subset of genes to be used to bisect a set of patients. First, the genes with the smallest "non-dimensionalized standard deviation", indicating a higher likelihood of membership in a distinct distribution, are selected for a patient clustering criterion. The non-dimensionalized standard deviation σ is given by

$$\sigma = \frac{\sqrt{Var(X)}}{\langle |X - \langle X \rangle| \rangle}, \tag{1.17}$$

where $\langle X \rangle$ represents the expectation value of a random variable X and $Var(X) = \langle X^2 \rangle - \langle X \rangle^2$. To further refine this subset each gene is then represented by an N dimensional point cloud using its expression level for each of the N patients. Then the proximity, $d(X_n, X_m)$ between genes X_n and X_m is defined as, $d(X_n, X_m) = 1 - C(X_n, X_m)$, where $C(X_n, X_m)$ is the Pearson correlation coefficient. A "simplicial complex" [163] is defined by defining a radius r_c and connecting any two points X_n and X_m for which $d(X_n, X_m) < r_c$. Then, genes that are members of the disjoint simplicial complexes that are stable for a wide range of r_c are chosen as the final clustering criterion. Once these M genes are selected the patients are represented as an M dimensional cloud and stable simplicial complexes are again used to bisect the patient set. These two steps can be repeated to further divide the network. Additionally, it is possible to profile new patients into the current patient subclasses (partitions) by defining the boundary of a patient subclass as the largest

mean proximity between current members of a subclass. Then, if a new patients mean proximity to members of a subclass is smaller than the boundary a patient is assigned to that subclass. This method presents novel techniques for both reducing the dimensionality of bi-sectioning criterion and combining persistent homology with hierarchical clustering and has the same advantages and disadvantages of other hierarchical clustering methods.

1.4.5 Modularity maximization

In 2002, Newman and Girven introduced a new community detection method based on the idea of link "betweenness [37]." The original method can be classified as a divisive hierarchical method which seeks to identify the links in a network that connect communities and remove them. Edge betweenness is the number of shortest paths between all vertices that run along the edge. The idea is that if two communities are only separated by a few edges then those edges should contain a high number of shortest paths between pairs of nodes from the two communities. The algorithm begins by calculating the betweenness of all edges and the edge with the largest betweenness is removed. This is done using the algorithm of Newman [164]. A new edge betweenness for all remaining edges is recalculated and the process continues until all links have been removed. The result is a dendrogram from which a partition is selected. The algorithm has a running time of $O(m^2N)$ and is shown to work well on generated networks with known community structure as well as real world networks. However, it still suffers from the fact that one must choose a cut in the dendrogram from the N set of nested partitions.

Newman and Girven then introduced a quality function known as"modularity

Q [2] to choose a cut. The modularity quality function explicitly contains the definition of a community, compares each community in a partition to a null model, does not require as an input the number of communities in a network, and measures quantitatively the strength of a partition. It is for the these reasons this work focuses on maximizing the quality function known as the modularity.

1.5 Modularity

To detect communities in unipartite networks, a quality function is needed that takes into account properties of the network being studied as well as incorporates a community definition. If a community is a highly connected group of nodes, one would expect there to be more links within a community than between communities. Girvin and Newman introduced modularity "Q" to quantify this idea [4] adding the constraint that there should be more links within a community than one would expect for a network with the same degree sequence and randomly distributed links. The modularity is defined as

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{1.18}$$

where A_{ij} is the element of the Adjacency matrix, m is the total number of links in the network and is given by $m = \sum_{ij} A_{ij}/2$, k_i is the degree of node i, c_i is the community to which node i is assigned, and $\delta(c_i, c_j)$ is the Kronecker delta function. The more modular a network is, the larger Q will be with a maximum Q equal to 1 and minimum equal to $-\frac{1}{2}$ [65].

One can gain more insight into this definition by rewriting Q in a different yet

equivalent form. Consider each contribution to the modularity by community c,

$$\frac{\sum_{ij\in c} A_{ij}}{2m} - \frac{\sum_{ij\in c} k_i k_j}{(2m)^2}. \tag{1.19}$$

Let l_c represent the number of links in community c and d_c represent the total degree of the nodes in community c. Then $\sum_{ij\in c} A_{ij} = 2l_c$ and $\sum_{ij\in c} k_i k_j = d_c^2$ so the modularity may be written as

$$Q = \sum_{c=1}^{C} \left[\frac{l_c}{m} - \left(\frac{d_c}{2m} \right)^2 \right], \tag{1.20}$$

where C is the total number of communities in the network. In this form, it is clear the modularity is a measure of the fraction of links in each community, l_c/m, above its expected fraction of links, $(d_c/2m)^2$.

If a network contains weighted edges, Eq. 1.18 is easily generalized [68]. The element A_{ij} represents the weight of the link between node i and j. Then the degree of the node k is defined as the total weight adjacent to node i and is given by $k = \sum_j A_{ij}$. The total amount of weight in the graph is given by $m = \frac{1}{2}\sum_{ij} A_{ij}$. With these new definitions the form of modularity Q for weighted networks remains unchanged and is given by Eqn. 1.18. However, it is non-trivial to generalize modularity Q to bipartite networks where two types of nodes exist.

1.6 Bipartite modularity

A *bipartite* network is a network composed of two types of nodes, X and Y, and links that are only found to connect a member of each type. Bipartite networks include some social [165, 166], collaboriation [167, 168, 169, 170], and biological networks [171, 172, 173]. In the definition of modularity Q given in the previous

section, there is a random expectation that links will exist between any two nodes in a network even if the network is bipartite. A consequence of this expectation is easily seen if one considers a particular partition of a bipartite network. Consider an unweighted bipartite network with p nodes of type one, q nodes of type two, and m total links with modularity Q written in the form given by Eqn 1.20. If one chooses to partition a bipartite network into two communities with all of the nodes of type X in community 1 and all of the nodes of type Y in community 2 for each community, $l_c = 0$ and $d_c = m$. Therefore, the modularity Q for any bipartite network partitioned in this way gives

$$\begin{aligned} Q &= \left[\frac{0}{m} - \left(\frac{m}{2m}\right)^2\right] + \left[\frac{0}{m} - \left(\frac{m}{2m}\right)^2\right] \\ &= -\frac{m^2}{4m^2} - \frac{m^2}{4m^2} \\ &= -\frac{1}{2}, \end{aligned} \tag{1.21}$$

the absolute minimum value of modularity Q. Modularity Q expects each of the communities to contain 1/4 of the total links in the network which will never occur due to the bipartivity. Therefore, one cannot simply maximize the modularity Q of a bipartite network as nodes of the same type partitioned in a community together will appear to have a smaller number of connections than expected effecting the result.

An additional issue exists when detecting communities as there are two different methodologies that can be used for partitioning bipartite networks. The first method is to study each side of the bipartite network separately by creating a projection of one side of the network. This has the advantage of showing directly the relationships between nodes of one type. For example, consider the bipartite network in Fig 1.9a, where there are 5 nodes of type X, 4 nodes of type Y, and 10 links. A simple projection of the network is illustrated in Fig 1.9b, where each pair of projected

nodes are connected by a link if they have at least one common neighbor of the opposite type. It is immediately clear that node 1 has a relationship solely with node 2 in the network. It is also clear that information in the bipartite network has been lost as all connected nodes in the projection share one link while, for example, in the bipartite network node 1 and 2 share two neighbors of type Y while node 2 and 5 share only one neighbor of type Y. An alternative projection keeps this information and connects neighbors with more than one link according to the number of neighbors of the opposite type they have in common. A multi-link projection of this type is shown in Fig 1.9c.

Additional projections have been used to preserve additional bipartite information. Newman considered this problem in the context of a bipartite co-authorship network where one type of nodes are scientists and the other type of nodes are scientific papers with a link between each scientist and the papers he or she co-authored [164]. Newman created a projection of the scientist side of the network by assigning a weight between scientist i and j using the formula

$$w_{ij} = \sum_k \frac{\delta_i^k \delta_i^k}{n_k - 1}, \tag{1.22}$$

where k runs over all papers, δ_i^k is 1 if scientist i is a coauthor of paper k and 0 otherwise, and n_k is the degree of paper k. Thus, Newman considered two scientists co-authorship in a paper as stronger if they were the sole co-authors weighting each term in the sum by $n_k - 1$. Consider the application of this projection in Fig. 1.9c where two links connect both pairs of nodes (1,2) and (2,3) indicating they each share two Y type neighbors. The two neighbors of pair (1,2), nodes 6 and 7, are solely connected to nodes 1 and 2. Pair (2,3) also share two neighbors however one of their neighbors, node 4, is connected to a total of four nodes. Using Newman's

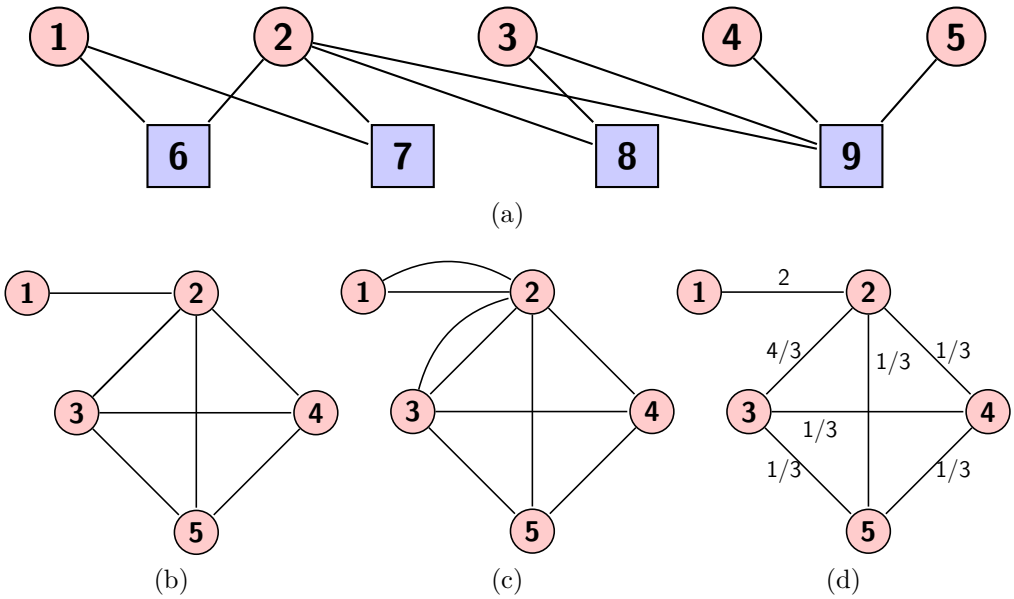


Figure 1.9: **Unipartite projection from a bipartite network**. In figure (a) an example bipartite graph is given consisting of 5 nodes of type X and 4 nodes of type Y. A method to analyze bipartite networks is to project one side of the network onto a unipartite graph. The simplest example of a projection of the type X side of the network is shown in (b) where a link indicates the pair of nodes share at least one neighbor. A multi-graph projection is shown in projection (c) with each link representing a common neighbor between the pair of nodes. A weighted projection is shown in projection (d) where the link weight between a pair of nodes is given by Eqn. 1.22.

projection, as shown in Fig. 1.9d, preserves this information and the strength of node pair (1,2) is distinguished from (2,3). Although, this projection has been used to study the community structure of collaboration networks [164, 174] it has not been widely adapted to other bipartite networks. A projection similar to this one is studied in Chapter 3.

A modularity method has been applied to multi-link projected networks by Guimera *et al.* [175] where each side of the bipartite networks community structure is analyzed separately. For each projection a link is placed between two nodes of the same type for every neighbor of the opposite type they are both connected to. Communities are then detected by maximizing modularity in the multi-link projection. This solves the problem of applying modularity Q to bipartite networks as in the multi-link projection only nodes of one type exist and are expected to be connected. Although this method is shown to find significant community structure in unweighted networks, the multi-link projection is not easily extended when the bipartite network is weighted. For example, in Fig. 1.10 node 1 and 2 are both connected to the same 2 neighbors (3 and 4), yet each is connected to 3 and 4 with a different weight. Thus, the number of "shared" neighbors between node 1 and 2 is not clear. The problem of extending the method of bipartite community detection using unipartite projections from a bipartite network when the links of the bipartite network are weighted is addressed in Chapter 3.

The second methodology analyzes a bipartite network directly resulting in a bi-clustered partition in which the two types of nodes are partitioned together in communities. This has the advantage that all of the information from the bipartite network is incorporated into the community structure. However, the bi-clustered partitioning is constrained to have the same number of communities on both sides.

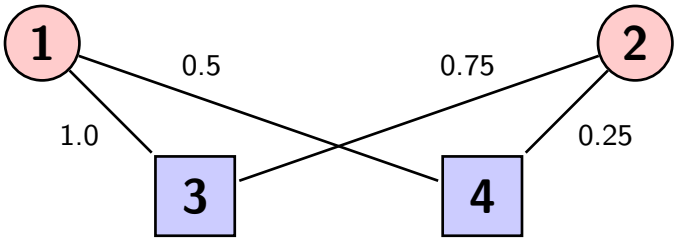


Figure 1.10: **A weighted bipartite example** In this example bipartite network the links in the bipartite network are weighted. A multi-graph projection is not readily applicable as the number of neighbors between nodes 1 and 2 is not clearly defined with weighted links.

.

Barber [176] has adapted the modularity Q definition to this type of bipartite clustering by defining all modularity elements between nodes of the same type equal to zero. This solves the problem in the modularity Q definition in which there is a random expectation that links exist between any two nodes in the network without sacrificing any loss of information as in a method involving a unipartite projection from a bipartite network. Barber's modularity also generalizes easily when considering a bipartite network with weighted links. In Chapter 3, the problem of community detection in weighted bipartite networks is addressed. Returning now to modularity Q, its properties and modularity maximizing methods are further discussed.

1.7 Criticisms of modularity

Thus far the advantages of using modularity Q to detect communities in complex networks have been discussed. This section analyzes the challenges of applying modularity maximization to community detection in networks. First, the topology of modularity Q is analyzed which has a direct impact on the ability to find the maximum modularity partition. Next, the inability of modularity Q to find overlapping communities is discussed. This is followed by a discussion of the ability to solely rely on the value of the modularity as a measure of modular significance. Finally, the so-called resolution limit in modularity Q is analyzed that may effect its ability to partition nodes into communities with a large variation in link density. These criticisms are important as they serve to illuminate the scope of modularity maximization in real-world applications.

1.7.1 The modularity landscape

Finding the partition to maximize modularity Q has been shown to be NP-hard [65, 177]. In the next section, a number of heuristic techniques are discussed that attempt to find high modularity partitions. Of interest then, is the topological landscape of high modularity partitions. Good *et al.* [6] have found that networks with modular structure exhibit a global peak around the optimal partition. However, they found this "peak" may be more similar in shape to a plateau. As a quantitative example they consider the case of k sparsely interconnected groups of nodes with equal densities $d_i \approx 2m/k$. Consider the change in modularity from merging two communities. Let us assume there are two modules i and j and that each contain l_i and l_j links and have degree d_i and d_j, respectively. Then the change in modularity from merging the two modules is given by

$$\begin{aligned}
\Delta Q &= Q_{merged} - Q_{separate} \\
&= \frac{l_i + l_{ij} + l_j}{m} - \left(\frac{d_i + d_j}{2m}\right)^2 - \left[\frac{l_i}{m} + \frac{l_j}{m} - \left(\frac{d_i}{2m}\right)^2 - \left(\frac{d_j}{2m}\right)^2\right] \\
&= \frac{l_{ij}}{m} + \frac{1}{(2m)^2}\left(d_i^2 + d_j^2 - (d_i + d_j)^2\right) \\
\Delta Q &= \frac{l_{ij}}{m} - \frac{2d_i d_j}{(2m)^2}, \qquad (1.23)
\end{aligned}$$

where l_{ij} is the number of links connecting module i and module j. Then the lower bound of the change in modularity when merging two communities in the above example is given when there are no links connecting module i and j ($l_{ij} = 0$). Then,

$$\Delta Q = -\frac{8m^2}{(2mk)^2} = \frac{2}{k^2}, \qquad (1.24)$$

which for example, is -0.005 for $k = 20$ and tends to zero as for larger k. Therefore, there can exist many partitions with modularity close to Q_{max}. Additionally, Good

et al. surveyed the modularity landscape of a number of synthetic and real world networks using a Simulated Annealing (SA) technique (described in Section 1.8.1). An example of such a modularity landscape for the metabolic network of the spirochaete *Treponema pallidum* is shown in Fig. 1.11. Here, the amount of difference in the community assignment between partitions was quantified using an information theoretic measure called the variation of information (VI) [178]. To visualize this difference between all partitions, the VI was was embedded in a 2-dimensional x-y coordinate system using a curvilinear component analysis algorithm [179] with the corresponding modularity score given by the z-coordinate. The inset shows the peak of the modularity landscape which displays a large number of partitions with similar modularity values yet structurally different community assignment. In Chapter 2, novel methods are introduced that take advantage of this property by obtaining the community structure from the ensemble of high modularity partitions.

1.7.2 Overlapping communities

As modularity Q is defined, each node is partitioned into a single unique community. However, in many real-world applications a node may have multiple roles in the network. As previously discussed, in social networks an individual could be, at the same time, a member of a family community, a work community, and a community for a particular hobby. Additionally, in biological networks such as genetic regulatory networks genes may participate in multiple functional communities. As maximizing modularity Q is strictly a partitional method many of the overlapping community methods previously discussed were motivated by modularity Q's inability to detect

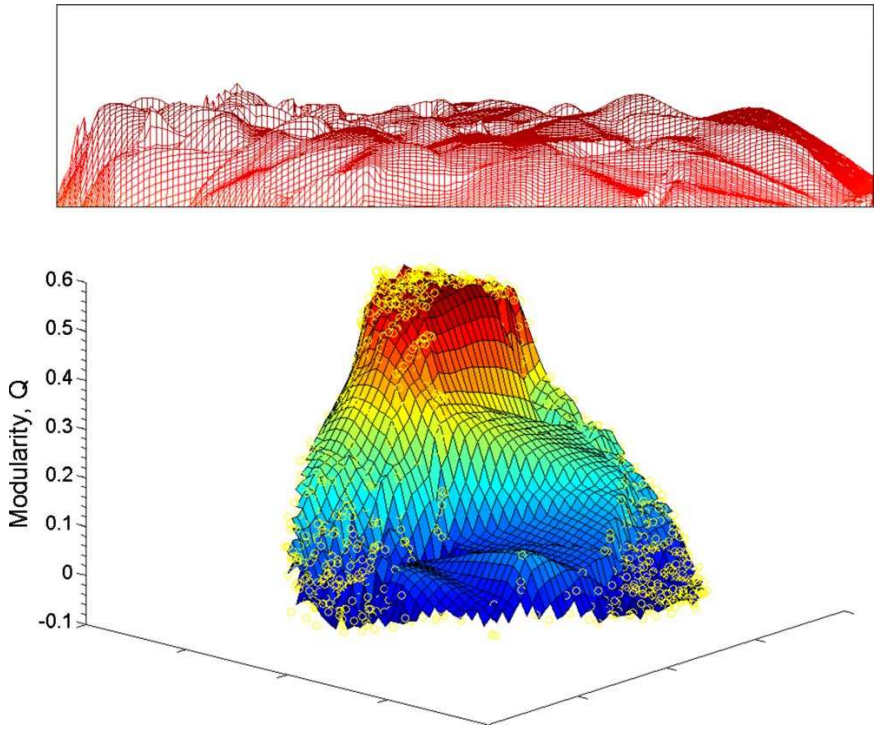


Figure 1.11: **The modularity landscape.** The modularity landscape for the metabolic network of the spirochaete *Treponema pallidum* found by [6]. Multiple partitions were found using a Simulated Annealing technique to maximize modularity Q and either stopped randomly or at a local maximum. Each of the found partitions is embedded in a 2-dimensional x-y coordinate system with the corresponding modularity score given by the z-coordinate. The inset shows the peak of the modularity landscape which displays a large number of partitions with similar modularity values.

this type of structure [161, 5]. In Chapter 2, the analysis of the community structure from the ensemble of high modularity partitions, which as discussed addresses the topology of the modularity landscape, additionally allows a construction of an overlapping community partition.

1.7.3 The significance of a partition

One of the criticisms of some community detection methods is that for a given dataset a partition is always found. How then does one determine if the found partition is significant or merely an artifact of the method? In the same manner, how does one evaluate a method that finds communities?

If the communities in a dataset are known *a priori*, a method can be evaluated based on its ability to recover the structure. One such computer-generated standard for unipartite networks is the planted l-partition model [180]. This model begins with l communities each containing n nodes. Nodes in the same community are connected with probability p_{in} and nodes in different communities are connected with probability p_{out}. The average degree, $\langle k \rangle$, of a node is therefore, $\langle k \rangle = (n-1)p_{in} + n(l-1)p_{out} = k_{intra} + k_{inter}$ where $k_{intra} \equiv (n-1)p_{in}$ is defined as the intra-community degree and $k_{inter} \equiv n(l-1)p_{out}$ is defined as the inter-community degree. For a given n, l and $\langle k \rangle$ the model begins with $k_{inter} = 0$ and the l communities are disconnected and therefore well defined. As k_{inter} is increased the number of inter-community links increases until $p_{in} = p_{out} = \langle k \rangle / (nl-1)$ at which point the network is completely random. Then, the quality of a community detection method can be evaluated by how well it recovers the l communities as a function of k_{inter}. Girven and Newman considered a special case of the l-partition model with $l = 4$, $n = 32$

and $\langle k \rangle = 16$ which has become a standard benchmark [37]. In Chapter 4, a bipartite l-partition model is constructed and used to compare bipartite modularities.

A completely random graph is not expected to contain significant community structure. Strikingly, it is exactly this type of random graph, an Erdös-Rényi graph, that can give large values of modularity Q. This is due to fluctuations in the distribution of links in the network [181, 61, 182]. The modularity Q values of Erdös-Rényi networks have been well studied by Reichardt and Bornholdt [61, 182]. They express the community detection problem as a Hamiltonian for a q-state Potts spin glass given by

$$H(\sigma) = -\sum_{i<j} (A_{ij} - \gamma p_{ij})\, \delta(\sigma_i, \sigma_j), \tag{1.25}$$

where σ_i represents the spin or community of node i, q is the number of possible spin indices, A_{ij} is the element of the adjacency matrix between node i and j, p_{ij} is the probability that a link exists between node i and node j, normalized, so that $\sum_{i \neq j} p_{ij} = 2M$, and γ a parameter that controls the amount of energy that a link contributes compared to a non-link. This Hamiltonian therefore represents a spin glass with couplings $J_{ij} = A_{ij} - \gamma p_{ij}$ that are ferromagnetic between connected nodes and antiferromagnetic between unconnected nodes.

When $\gamma = 1$ and $p_{ij} = \frac{k_i k_j}{2m}$, the Newman Modularity can be written as $Q = -\frac{1}{M} H(\sigma)$. Additionally, for $\gamma = 1$ the mean of the coupling is 0, and therefore, one expects zero magnetization in the ground state and an equipartition of the network. Equipartitions of random graphs have been previously studied by Fu and Anderson [183] and by Kanter and Sompolinsky [184] who use the replica method [185] to derive an analytic expression for the ground state of the spin glass Hamiltonian in the large N limit and assuming p $\sim$ O(1). For a network of q equipartitions the ground

state Hamiltonian $H(\sigma) = -N^{\frac{3}{2}} J \frac{U(q)}{q}$ where $U(q)$ is the ground state energy and J is the variance of the coupling constant which, assuming $p_{ij} = p$ for an Erdös-Rényi network and $\gamma = 1$, is given by $J^2 = p(1-p)$ [184, 61]. The maximum of $\frac{U(q)}{q}$ is equal to 0.485 when $q = 5$ [61] and with $M = pN^2/2$ Reichardt and Bornholdt give the expected modularity Q for an ER graph as

$$\langle Q_{ER} \rangle = 0.97 \sqrt{\frac{1-p}{pN}}. \tag{1.26}$$

Additionally, Reichardt and Bornholdt derive similar results by successive recursive bi-partitioning until Q no longer increases [61] using similar results from Fu and Anderson [183]. In this case, after b successive recursive bipartition the modularity is given by

$$Q(b) = \frac{2^b - 1}{2^b} - \frac{1}{\langle k \rangle} \sum_{t=1}^{b} \langle k_{out}, t \rangle, \tag{1.27}$$

where $\langle k \rangle$ is the average degree of the entire network and $\langle k_{out,t} \rangle$ is the average number of external links gained by a node after partition t. $\langle k_{out} \rangle$ is calculated using $\frac{pN - U(2)\sqrt{pN(1-p)}}{2}$ [61]. Improvement of these methods, including empirical finite size corrections, is discussed in Chapter 3.

If Erdös-Rényi networks are able to be partitioned with high modularity Q then for a network partition to truly be modular one would expect it have a modularity Q larger than expected for an Erdös-Rényi network. For a network with N nodes and m links, the modularity value can be compared to the modularity distribution of an ensemble of Erdös-Rényi networks with N nodes that are connected with probability $p = 2m/(N(N-1))$. For such an ensemble of Erdös-Rényi networks with average modularity $\langle Q_{ER} \rangle$ and standard deviation σ_{ER}, the z-score of Q can be measured

by [34]

$$z = \frac{Q - \langle Q_{ER} \rangle}{\sigma_{ER}}. \tag{1.28}$$

The z-score measures the effect size of the network's modularity Q value which is simply a standardized measure of the difference between it and the average modularity value of the ensemble of Erdös-Rényi networks. Typically, z-scores larger than 0.5 indicate a strong effect size [186]. In Chapter 3, bipartite modularity is similar studied and an analytic derivation of the expected bipartite modularity and standard deviation is derived for random bipartite networks. This allows a better measure of a partitions modular strength using the z-score.

1.7.4 The resolution limit

As discussed above, the modularity function measures the fraction of links in a community above its expectation for each community in the partition. The expectation assumes that it is possible for each node in the network to be connected to any other node. As a consequence, two modules in a network may be merged in the maximum modularity partition if the number of links which connect them is larger than expected which in some cases occurs via a single link. Fortunato *et al.* [187] was the first to describe this phenomenon and show that a community is more likely to consist of a substructure of modules if the community's number of links, l_c, satisfies the inequality, $l_c < \sqrt{2m}$. A more intuitive explanation was given by Good *et al.* [6] as follows. As shown in the last section (Eqn. 1.23) the change in modularity from merging two communities is given by

$$\Delta Q = \frac{l_{ij}}{m} - \frac{2d_i d_j}{(2m)^2}. \tag{1.29}$$

These two modules will be merged in the maximum modularity partition when $\Delta Q > 0$ which is given when

$$l_{ij} > d_i d_j / 2m \equiv \langle l_{ij} \rangle. \tag{1.30}$$

This shows two modules are merged when the number of links between them, l_{ij}, is greater than the expected number $\langle l_{ij} \rangle$. As the number of links in the network, m, grows, the expected number of links between the two modules decreases and can eventually equal less than 1. Thus, the resolution limit can pose a greater problem for unweighted networks in which the smallest link value is 1 [6]. For example, if there is a single link between the two modules and the rest of the network, and it is assumed that the two modules contain the same number of links, l ,a merge will occur, according to Eqn. 1.30 when

$$\begin{aligned} 1 &> \frac{(2l+2)^2}{2m} \\ \sqrt{2m} &> 2(l+1) \\ l &< \sqrt{m/2} - 1. \end{aligned} \tag{1.31}$$

Then, the merged module will contain $2l + 1 \equiv l_m$ links and will contain this substructure when

$$\begin{aligned} 2l+1 &< 2(\sqrt{m/2} - 1) + 1 \\ l_m &< \sqrt{2m} - 1 \approx \sqrt{2m}, \end{aligned} \tag{1.32}$$

recovering the results of Fortunato *et al.* Thus, the modularity function looks for communities with a typical density of links, $\sqrt{2m}$, whose size grows with the number of links in the network.

Solutions to the resolution problem have been proposed most notably by Arenas *et al.* [188] and Berry *et al.* [189] . The solution of Arenas *et al.* involves adding

self-loops to all nodes in the network with weight r. This changes the inequality in Eqn. 1.31 to

$$l < \sqrt{\frac{m}{2} + \frac{Nr}{4}} - \frac{nr}{2} - 1, \tag{1.33}$$

where n is the number of nodes in the module. Thus the network can be surveyed at different resolution levels by tuning the parameter r. However the best choice of r is not clear for a particular network without *a priori* knowledge of its structure. Nevertheless, this method has still been used to explore interesting features of a network such as the stability of partitions with respect to r. Berry *et al.* has shown for a weighted network a community may not be resolved if

$$l < \sqrt{\frac{W\epsilon}{2}} - \epsilon, \tag{1.34}$$

where l is the sum of all link weights in the module, W is the sum of all link weights in the network and ϵ is the largest inter-community link. Berry *et al.* then proposed a re-weighting scheme that locally increases the link weight of intra-community edges and decreases the link weight of inter-community edges, ϵ. Thus, ϵ changes locally alleviating the problem of choosing a particular resolution parameter as in the case of Arenas *et al.*

1.8 Algorithms to maximize the modularity

A partitional community detection algorithm seeks to find the partition that maximizes or minimizes a quality function. If modularity Q helps one choose a good partition in a dendrogram, then it is thought that finding the partition that maximizes modularity Q should give the "best" partition. Unfortunately, finding the partition that maximizes modularity Q has been proven to be an NP-hard problem [65, 177].

However, there exist approximate algorithms that have been developed to find high modularity partitions.

1.8.1 Simulated annealing

Perhaps the algorithm with the ability to find a partition closest to the "true" modularity maximum is simulated annealing [181]. Simulated annealing is a Metropolis Monte-Carlo algorithm that begins by placing nodes in a random partition at an initial temperature T. There is a list of moves, which can be either global or local, that are chosen at random. Global moves include merging two communities while local moves include moving a node into a random community. If a move gives a positive increase ΔQ it is always accepted. However, if it decreases the modularity it is accepted with a probability $\exp(\beta \Delta Q)$ where $\beta \propto 1/T$. After a number of moves the temperature is lowered by ΔT, called the cooling schedule, set by the user. This process is iterated until the partition reaches a stable state which can give a good approximation of the maximum of modularity Q. Simulated annealing is similarly used by Reichardt and Bornholdt [61] to minimize the Modularity Q equivalent Hamiltonian in the q-state Potts spin glass of Eqn. 1.25.

In principle, as the length of the simulated annealing cooling schedule increases the probability of finding the global Q maximum goes to one [190]. However, this can greatly increase the computational time of the algorithm the complexity of which cannot be estimated due to the dependence on the various parameters chosen. Simulated annealing's applicability is therefore ideal for smaller networks.

1.8.2 Greedy algorithms

Newman was the first to devise an algorithm to maximize the modularity [191]. He used an agglomerative hierarchical technique where each node in the network begins in one of N singleton cluster represented by the lowest level in a dendrogram. At each step in the algorithm, the two clusters that give the largest increase in modularity ΔQ when merged are joined together in a single cluster constituting a new level in the dendrogram. This continues until all nodes are in a single community. Thus, the total number of partitions in the dendrogram is N and one chooses the partition giving the largest modularity. By using structures for sparse matrices the algorithm can be used on very large graphs [192] with a complexity of $\mathrm{O}((m+N)N)$ or $\mathrm{O}(N^2)$ for a sparse graph. Modifications to the algorithm have been proposed that can increase the quality of the partition by normalizing ΔQ to favor smaller [193] or equal community sizes [194]. Additionally improvements have been found by beginning with a different initial configuration of nodes into clusters [195, 196, 197] as well as adding tuning steps to the algorithm [198] that perform a local search for modularity improvement. For example, a complete greedy tuning algorithm considers moving all vertices into any cluster (and a newly created one) and makes the move with the largest modularity increase [198]. This is repeated until there are no longer any moves that give an increase in modularity. Additionally, Noack and Rotta[198] considered moving groups of nodes rather than individual ones into other clusters which again improved performance. Although greedy algorithms are fast, other methods that will be discussed below find higher quality solutions.

1.8.3 Extremal optimization

Extremal Optimization (EO) was first used by Duch and Arenas [199] to maximize the modularity and involves assigning a fitness value to each node representing its contribution to the modularity for a given partition. The fitness value represents the contribution to the modularity of node i normalized by its degree, and is defined as

$$\lambda_i = \frac{1}{k_i} \sum_j \left(A_{ij} - \frac{k_j}{2m} \right) \delta(c_i, c_j), \tag{1.35}$$

where A_{ij} is the element of the Adjacency matrix, m is the total number of links in the network, k_i is the degree of node i, c_i is the community to which node i is assigned, and $\delta(c_i, c_j)$ is the Kronecker delta function. The nodes are initially assigned into two communities of equal size. The nodes are ordered beginning with smallest fitness value and a node with rank r is selected with probablilty $r^{-\tau}$ and moved to the opposite community. τ is a variable that must be tuned. The fitness is recalculated for the resulting partition and the process continues until Q cannot be improved. At this point, the two communities become disconnected components and the algorithm is repeated for each component. EO was an attempt to find quality partitions such as in simulated annealing while reducing the time requirement of the algorithm. EO has a complexity of $O(N^2 \log(N))$. However, algorithms that iteratively bisect the network are known to introduce bias [9].

1.8.4 The leading eigenvalue method

A spectral algorithm to maximize the modularity was developed by Newman that uses the eigenspectrum of the modularity matrix to iteratively divide the network into communities [151, 4]. This is similar to the spectral partitioning method of

section 1.4.3. One begins with the definition of modularity Q,

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{1.36}$$

where A_{ij} is the element of the Adjacency matrix, m is the total number of links in the network and is given by $m = \sum_{ij} A_{ij}/2$, k_i is the degree of node i, c_i is the community to which node i is assigned, and $\delta(c_i, c_j)$ is the Kronecker delta function.

1.8.4.1 Initial Division

To initially divide the network into two groups, one creates a vector $\mathbf{s}$ whose element s_i is 1 if node i is assigned to the first community or -1 if node i is assigned to the second. Then, $\delta(c_i, c_j)$ can be written as $\delta(c_i, c_j) = \frac{1}{2}(s_i s_j + 1)$ which is substituted into Eqn. 1.36 giving

$$\begin{aligned}
Q &= \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \frac{1}{2}(s_i s_j + 1) \\
&= \frac{1}{4m} \left[\sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j + \sum_{ij} A_{ij} - \sum_{ij} \frac{k_i k_j}{2m} \right] \\
&= \frac{1}{4m} \left[\sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j + 2m - \frac{4m^2}{2m} \right] \\
&= \frac{1}{4m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j \\
Q &= \frac{1}{4m} \mathbf{s^T B s},
\end{aligned} \tag{1.37}$$

where $\mathbf{B}$ is the modularity matrix with elements $B_{ij} = A_{ij} - k_i k_j / 2m$. Writing $\mathbf{s}$ as a linear combination of the eigenvectors $\mathbf{u}$ of $\mathbf{B}$, $\mathbf{s} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{a_i u_i}$, the modularity may be written

$$
\begin{aligned}
Q &= \frac{1}{4m}\sum_{i=1}^{N} a_i \mathbf{u}_i^T \mathbf{B} \sum_{j=1}^{N} a_j \mathbf{u}_j \\
&= \frac{1}{4m}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j \mathbf{u}_i^T \mathbf{B}\mathbf{u_j} \\
&= \frac{1}{4m}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j \mathbf{u}_i^T \mathbf{u_j}\beta_j \\
&= \frac{1}{4m}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j \delta_{ij}\beta_j \\
Q &= \frac{1}{4m}\sum_{i=1}^{N} a_i^2 \beta_i, \qquad (1.38)
\end{aligned}
$$

where β_i is the eigenvalue of $\mathbf{B}$ with eigenvector $\mathbf{u_i}$. Then the division of the network corresponding to the largest value of Q is that which places the most weight a_i in the terms corresponding to the largest positive eigenvalues. Therefore, the nodes are assigned to the two groups using the eigenvector $\mathbf{u_i^1}$ corresponding to the largest positive eigenvalue β_i^1. Because $s_i \in \{-1, 1\}$, one can make the assignment

$$
s_i = \begin{cases} 1 & \text{if } u_i^1 \geq 0 \\ -1 & \text{if } u_i^1 < 0 \end{cases}. \qquad (1.39)
$$

Note that there still exists for the modularity Q matrix a zero eigenvalue corresponding to the eigenvector $\mathbf{u} = \{1, 1, 1, ..., 1\}$. In the spectral clustering example of section 1.4.3 this lead to problems as one sought to minimize the cut size using the smallest eigenvalue of the Laplacian. The smallest eigenvalue of the Laplacian is *always* zero which lead to the trivial partition of all nodes in one community. However, the largest eigenvalue of the modularity Q matrix is *only* zero when the largest possible change in modularity Q is zero with all nodes assigned one community. This is advantageous as it can serve as a criterion to stop division.

The eigenvector corresponding to the largest eigenvalue can be found using the power method. However, issues can exist as the power method finds the eigenvector corresponding to the largest absolute eigenvalue so one must make sure the eigenvalue found is positive. Additionally, the power method may fail to converge if there exists for the largest eigenvalue, β_1, an equal in magnitude but opposite in sign eigenvalue, β_2, where $\beta_2 = -\beta_1$. In this case, one may shift the matrix by a small value α and find the eigenvector corresponding to the largest eigenvalue of $\mathbf{B} + \alpha\mathbf{I}$ [200]. This will shift all eigenvalues by α allowing convergence.

1.8.4.2 A Kernighan-Lin tuning step

The assignment using the eigenvalue $\mathbf{u_i^1}$ serves as the initial guess for the bisection and a variant of the Kernighan-Lin (K-L) algorithm [149] is used to improve the assignment. The K-L algorithm is a tuning algorithm that, as discussed, began as a method for graph partitioning. The variant of the K-L algorithm begins by moving each node from it's current community into the other to see if there is an improvement in modularity.

To find this improvement, begin by assuming the initial guess for the split is contained in vector $\mathbf{s}$. s_i is assigned a value of 1 or -1 whether the node i is in the first or second community, respectfully. Then the modularity for this guess can be written

$$Q = \sum_{i,j} B_{ij} s_i s_j, \tag{1.40}$$

where for the moment, the factor of $1/4m$ is ignored. Consider moving node k then

one can rewrite this equation as

$$Q = \sum_{i,j \neq k} B_{ij} s_i s_j + s_k \sum_i B_{ik} s_i + s_k \sum_i B_{ki} s_i - B_{kk} s_k^2. \tag{1.41}$$

Now, let $B_{ik} = B_{ki}$, and because $s_k^2 = 1$,

$$Q = \sum_{i,j \neq k} B_{ij} s_i s_j + 2 s_k \sum_i B_{ik} s_i - B_{kk}. \tag{1.42}$$

If node k moved then s_k becomes s'_k, and the new modularity is

$$Q' = \sum_{i,j \neq k} B_{ij} s_i s_j + 2 s'_k \sum_i B_{ki} s_i - B_{kk}. \tag{1.43}$$

Then, the change in modularity is given by

$$\delta Q = 2(s'_k - s_k) \sum_{i \neq k} B_{ki} s_i. \tag{1.44}$$

Now consider both cases where k is initially assigned into either the first or second community. If $s_k = 1$ then $s_{k'} - s_k = -1 - 1 = -2$. If $s_k = -1$ then $s_{k'} - s_k = 1 - (-1) = 2$. Then, for both cases $(s_{k'} - s_k) = -2 s_k$, and adding back in the factor of $1/4m$,

$$\delta Q = -\frac{1}{m} s_k \sum_{i \neq k} B_{ki} s_i. \tag{1.45}$$

Then, the K-L tuning step is as follows:

1. Initialize Q_{total}, defined as the improvement using the K-L tuning step, to zero.
2. Set the initial configuration of vector $\boldsymbol{s}$ to $\boldsymbol{s_{initial}}$.
3. consider moving all nodes by computing, δQ using Eqn. 1.45 for all k.
4. Pick the node giving the largest $\delta Q \equiv$ node K. If there is more then one node with the largest δQ choose one of these randomly.

5. Add δQ to Q_{total}.

6. Set $s_K = -s_K$ for node K

7. If Q_{total} is the largest or equal to the largest found so far store $\boldsymbol{s}$.

8. Repeat steps 3. through 7. only considering nodes which have not been moved until all nodes have been moved once.

9. If the largest intermediate Q_{total} is positive, then repeat from step 1 using $\boldsymbol{s}$ as the initial guess or if there is more than one intermediate $\boldsymbol{s}$ stored with this Q_{total} randomly choose between them. If Q_{total} is negative end the algorithm returning the initial configuration vector $\boldsymbol{s_{initial}}$.

If the largest eigenvalue is zero or, alternatively, the Kernighan-Lin algorithm fails to find an assignment that produces an increase in modularity when splitting the community, the algorithm stops.

1.8.4.3 Subsequent divisions

To divide the network into more than 2 communities one can iteratively bisect each community as long as the modularity increases. However, each community cannot simply be treated as a separate graph using Eqn. 1.37. There is a correction term to the modularity matrix after the initial splitting. Consider writing the definition of modularity Q as

$$Q = \frac{1}{2m} \sum_{k}^{N_c} \sum_{i,j \in C_k} B_{ij}, \tag{1.46}$$

where N_c is the number of communities C. If community C_{k_o} is split into 2 communities, $C_{k_o,1}$ and $C_{k_o,2}$, the new modularity Q' can be written as

$$\begin{aligned} Q' &= \frac{1}{2m}\left[\sum_{\substack{k=1\\k\neq k_o}}^{N_c}\sum_{i,j\in C_k} B_{ij} + \sum_{i,j\in C_{k_o,1}} B_{ij} + \sum_{i,j\in C_{k_o,2}} B_{ij}\right] \\ Q' &= \frac{1}{2m}\left[\sum_{\substack{k=1\\k\neq k_o}}^{N_c}\sum_{i,j\in C_k} B_{ij} + \sum_{i,j\in C_{k_o,1}} B_{ij} + \sum_{i,j\in C_{k_o,2}} B_{ij} + \sum_{i,j\in C_{k_o}} B_{ij} - \sum_{i,j\in C_{k_o}} B_{ij}\right], \end{aligned}$$

where zero has been added into the equation in the last two terms. This allows one to write

$$\begin{aligned} Q' &= \frac{1}{2m}\left[\sum_{k=1}^{N_c}\sum_{i,j\in C_k} B_{ij} + \sum_{i,j\in C_{k_o,1}} B_{ij} + \sum_{i,j\in C_{k_o,2}} B_{ij} - \sum_{i,j\in C_{k_o}} B_{ij}\right] \\ Q' &= Q + \frac{1}{2m}\left[\sum_{i,j\in C_{k_o,1}} B_{ij} + \sum_{i,j\in C_{k_o,2}} B_{ij} - \sum_{i,j\in C_{k_o}} B_{ij}\right] \qquad (1.47) \\ \Delta Q &= \frac{1}{2m}\left[\sum_{i,j\in C_{k_o,1}} B_{ij} + \sum_{i,j\in C_{k_o,2}} B_{ij} - \sum_{i,j\in C_{k_o}} B_{ij}\right], \qquad (1.48) \end{aligned}$$

where Q was substituted into Eqn. 1.47 using Eqn. 1.46. Now the same method is used as during the initial division creating a vector $\mathbf{s}$ whose element s_i is 1 if node i is assigned to the first community, $C_{k_o,1}$, or -1 if node i is assigned to the second community, $C_{k_o,2}$. This allows the first two terms in Eqn. 1.48 to be combined giving

$$\begin{aligned} \Delta Q &= \frac{1}{2m}\left[\frac{1}{2}\sum_{i,j\in C_{k_o}} B_{ij}(s_i s_j + 1) - \sum_{i,j\in C_{k_o}} B_{ij}\right] \\ &= \frac{1}{2m}\left[\frac{1}{2}\sum_{i,j\in C_{k_o}} B_{ij} s_i s_j - \frac{1}{2}\sum_{i,j\in C_{k_o}} B_{ij}\right] \\ \Delta Q &= \frac{1}{4m}\left[\sum_{i,j\in C_{k_o}} B_{ij} s_i s_j - \sum_{i,j\in C_{k_o}} B_{ij}\right]. \qquad (1.49) \end{aligned}$$

Finally, one uses the relation,

$$\sum_{i,j\in C_{k_o}} \delta_{ij} s_i s_j \sum_{l\in C_{k_o}} B_{il} = \sum_{i\in C_{k_o}} s_i^2 \sum_{l\in C_{k_o}} B_{il} = \sum_{i,l\in C_{k_o}} B_{il}, \tag{1.50}$$

to write,

$$\begin{aligned}
\Delta Q &= \frac{1}{4m}\left[\sum_{i,j\in C_{k_o}} B_{ij}s_i s_j - \sum_{i,j\in C_{k_o}} B_{ij}\right] \\
&= \frac{1}{4m}\left[\sum_{i,j\in C_{k_o}} B_{ij}s_i s_j - \sum_{i,j\in C_{k_o}} \delta_{ij} s_i s_j \sum_{l\in C_{k_o}} B_{il}\right] \\
&= \frac{1}{4m}\sum_{i,j\in C_{k_o}}\left[B_{ij} - \delta_{ij}\sum_{l\in C_{k_o}} B_{il}\right] s_i s_j \\
\Delta Q &= \frac{1}{4m}\mathbf{s^T B^{(k_o)} s},
\end{aligned} \tag{1.51}$$

where $\mathbf{B^{(k_o)}}$ has elements $B_{ij}^{(k_o)} = B_{ij} - \delta_{ij}\sum_{l\in C_{k_o}} B_{il}$. To maximize the change in modularity, ΔQ, when bisecting a community k_o after the initial division one can assign elements of s_i using the eigenvalue corresponding to the largest eigenvalue of $\mathbf{B^{(k_o)}}$. The Kernighan-Lin algorithm can again be combined to improve the division. Communities are then iteratively bisectioned as long as the modularity increases. Additionally, note that Eqn. 1.51 is the generalized form of Eqn. 1.37 and can be used to make the initial split as the term $\sum_{l\in C_{k_o}} B_{il} = 0$ when all nodes are in a single community.

1.8.4.4 Final tuning

Recently, Sun *et al.* [9] have shown that iteratively bisecting a network creates a bias in the resulting community structure found. Consider, after the initial bisection of the network each community forms two disjoint subsets. Each subset of nodes will only be divided in the subsequent iterations of the algorithm so that once two

nodes are separated into different communities they are never again found together in the same community. Dividing the network in this way creates "hard partitions" in the algorithm. Sun *et al.* found a manifestation of these hard partitions in the distribution of the community sizes in an ensemble of Erdös-Rényi networks. Peaks in the community size distributions exist of size $\mathrm{N}/2^x$ for certain values of integer x which depends on the number of nodes in the network and the connection probability. Furthermore, if instead the network is trisected at each step of the spectral partitioning process peaks exist of size $\mathrm{N}/3^x$ for values of integer x. To amend this bias Sun *et al.* added an additional tuning step called Final Tuning in the spectral partitioning algorithm of Newman and Girven in which one considers the change in modularity from moving each node from its current community to all existing communities or forming its own community.

To find the change in modularity Q when node l begins in community C_x and is moved to community C_y begin with modularity Q written as

$$Q = \frac{1}{2m} \sum_{k}^{N_c} \sum_{i,j \in C_k} B_{ij}, \tag{1.52}$$

where N_c is the number of communities C. Initially, Q can be written as

$$Q = \sum_{\substack{k=1 \\ k \neq x,y}}^{N_c} \sum_{i,j \in C_k} B_{ij} + \underbrace{\sum_{i,j \in C_x} B_{ij}}_{(I)} + \underbrace{\sum_{i,j \in C_y} B_{ij}}_{(II)}. \tag{1.53}$$

With l moved from C_x to C_y the modularity becomes

$$Q' = \sum_{\substack{k=1 \\ k \neq x,y}}^{N_c} \sum_{i,j \in C_k} B_{ij} + \underbrace{\sum_{i,j \in C_x \setminus l} B_{ij}}_{(III)} + \underbrace{\sum_{i,j \in C_y \cup l} B_{ij}}_{(IV)}, \tag{1.54}$$

then

$$\delta Q_l = [(III) - (I) + (IV) - (II)]. \tag{1.55}$$

Note that

$$\begin{aligned}(III) &= (I) - \sum_{j \in C_x} B_{lj} - \sum_{i \in C_x} B_{il} + B_{ll} \\ (IV) &= (II) + \sum_{j \in C_y} B_{lj} + \sum_{i \in C_y} B_{il} + B_{ll}.\end{aligned}$$

Therefore, the change in modularity is given by

$$\delta Q_l = \sum_{i \in C_y} B_{il} + \sum_{j \in C_y} B_{lj} - \sum_{i \in C_x} B_{il} - \sum_{j \in C_x} B_{lj} + 2B_{ll}. \tag{1.56}$$

For a symmetric matrix,

$$\delta Q_l = 2 \left(\sum_{i \in C_y} B_{il} - \sum_{i \in C_x} B_{il} + B_{ll} \right). \tag{1.57}$$

For each iteration of the algorithm after the initial division, after all existing communities have been split with the Kernighan-Lin step, final tuning is performed.

Then, the final tuning step is as follows:

1. Initialize Q_{total}, defined as the improvement using the final tuning step, to zero.
2. Set the initial partition of the network vector $\boldsymbol{P(G)}$ to $\boldsymbol{P(G)_{initial}}$.
3. Consider moving all nodes from its current community to all other existing communities or into a community of its own by computing δQ using Eqn. 1.57.
4. Pick the move giving the largest δQ. If there is more then one move with the largest δQ choose one of these randomly.
5. Add δQ to Q_{total}.
6. Fix the community assignment for the node moved updating $\boldsymbol{P(G)}$.

7. If Q_{total} is the largest or equal to the largest found so far, store $\boldsymbol{P(G)}$.

8. Repeat steps 3. through 7. only considering nodes which have not been moved until all nodes have been moved once.

9. If the largest intermediate Q_{total} is positive, then repeat from step 1 using $\boldsymbol{P(G)}$ as the initial guess or if there is more than one intermediate $\boldsymbol{P(G)}$ stored with this Q_{total} randomly choose between them. If Q_{total} is negative end the algorithm returning the initial partition $\boldsymbol{P(G)}$.

Adding this tuning step after each round of bisections, when combined with spectral partitioning, finds the largest value of Q for any algorithm for networks up to a few thousand nodes in size [9]. The algorithm has a complexity of $\mathrm{O}(N^2 \log N)$ with an ideal trade off between quality and complexity.

Recently, a K-L variant algorithm similar to final tuning has been applied to modularity maximization. Sobolevsky *et al.* [201] consider moving nodes between two communities in the network with the destination community possibly empty. Then every pair of communities in the network is updated in this fashion until the modularity is no longer improved. This is also an improvement over the Newman's spectral partitioning algorithm with K-L tuning.

Adapting the spectral partitioning algorithm with final tuning to maximize bipartite modularities is a non-trivial problem that is addressed in Chapter 3. Additionally, a new tuning step is introduced that improves the modularity maximizing performance of this algorithm without increasing its complexity.

1.9 Dissertation organization

In the previous sections, the problem of community detection in networks was introduced, focusing on the properties and methods of modularity maximization. The remainder of this dissertation focuses on both the application of modularity maximization methods in biological networks and the development of methods and applications of bipartite modularity maximization in both weighted and unweighted bipartite networks.

In Chapter 2, methods for exploring the hierarchical organization of genetic regulatory networks that robustly detect core functional communities are presented. The methods are tested and their validity demonstrated, by applying them to *Escherichia coli* genetic expression data, finding a hierarchy of functionally relevant communities and then comparing those communities to the known *E. coli* functional groups. Examples are then given of how these methods can be used to infer regulatory interactions between genes.

Chapter 3 introduces a new definition of bipartite modularity, defined as the Information Preserving (IP) modularity, that extends the ability to partition each side of bipartite networks separately when the links in the bipartite network are weighted. The method is meant to preserve more information about the bipartite network in its unipartite projection. Additionally the spectral method of modularity maximization is adapted to bipartite modularities and a new tuning step is added that merges communities. Adding this tuning step to the spectral method described in section 1.8.4 for unipartite networks extends the current algorithms ability to find the partition with the largest value of Q for any algorithm in networks of up to *tens* of thousands of nodes in size. Finally, the expected value and standard deviation

of IP modularity for random Erdös-Rényi networks is then analytically derived so that a Z-score comparison of the modularity values of real-world networks can be computed.

In Chapter 4, the ability of the bipartite modularities described in Chapter 3 to recover the known structure of a bipartite model with unweighted and weighted links is studied. When compared to the Guimera *et al.* bipartite modularity the IP modularity performs better at recovering the known structure. Furthermore, a real world bipartite network, the metabolic network of *E. coli*, is used as both an example and a benchmark of the applicability of the method.

The final chapter is a summary of this dissertation.

Chapter 2

Robust Detection of Hierarchical Communities from *Escherichia coli* Gene Expression Data

2.1 Introduction

The following chapter is an edited version of a work published in PloS Computational Biology [202]. Gene regulation networks represent the set of regulatory interactions between all genes of an organism. These networks can contribute to our understanding of the development of organisms and how they integrate internal and external signals to coordinate gene expression responses [203, 85]. Moreover, knowledge of gene regulation networks allows communities of closely interacting genes to be identified. Once identified, such communities are an important resource for developing hypotheses for the function of uncharacterized genes and can provide insight into

patterns of regulatory network evolution and function [88, 204, 205, 206, 86, 207]. Examining the relationships between communities can also reveal a hierarchical set of interactions, which is thought to be a fundamental organizing principle in many biological systems [208, 209, 171]. For all these reasons, determining gene regulation networks and their functional organization remains a major goal of systems biology.

As mentioned in Chapter 1, approaches to determine the underlying structure of the transcriptional regulatory network generally infer links between genes of the network based on correlation in their transcriptional response to a series of environmental and genetic perturbations. [85, 86, 87, 88, 89, 90, 91, 92]. Identification of groups of interacting node (gene) communities poses an additional challenge. Communities can be identified using computational methods developed in network science [210]. These methods include hierarchical clustering [211, 212, 213], clique based clustering [214, 215, 216, 217], core-pheriphery [218, 219, 220], K means clustering [221], principal component analysis [222, 223], label propagation [153, 224], statistical mechanical approaches [225, 145] , and modularity maximization methods [2, 4, 226, 227, 9]. Often these algorithms agglomerate or divide the nodes of a network into groups based on either the links of the network or the strength of the correlation value between pairs of nodes. However, certain algorithm parameters, such as the number of groups, are often required as user inputs and can become increasingly difficult to predict as the size and complexity of the network grows. In addition, there can be considerable variability in the community detection process due to approximations and stochastic elements of the computational algorithms.

In this chapter, methods are presented for determining the hierarchical organization of genetic regulatory networks and for detecting functional communities of genes that are robust to variability in both gene expression data and community detection

parameters. A recently developed community detection method [9] is applied to regulation networks inferred from a compendium of *E. coli* expression profiles using the context likelihood of relatedness (CLR) algorithm [88]. This method uses the mutual information in the data sequence for pairs of genes to construct a weighted "Z-score matrix" that describes the relatedness of each gene pair. This weighted Z-score matrix is converted to a network by choosing a threshold value. Any element of the Z-score matrix above the chosen threshold value is converted to an unweighted link and any element of the matrix below the chosen threshold value is removed. We choose threshold values of 2.0, 4.0, and 6.0 which is below, at, and above the critical threshold value for the which the network is no longer fully connected. Then, to identify communities in each network, a recently developed community detection algorithm [9] that partitions the network so as to maximize its modularity, is run multiple times. The modularity maximizing algorithm used by this method, when applied to a series of widely studied networks, produces the partitioning with the largest modularity of any known fast algorithm for networks up to a few thousand nodes in size [9].

As mentioned above, there is variability in the community detection process. Indeed, numerous network partitions can give modularities close to the maximum and these partitions can be structurally diverse [6]. Rather than treat this property as a disadvantage, the following methods use the stochasticity to find correlations between the ensemble of runs of the community detection algorithm. A core community is then defined, as those nodes that are consistently assigned to the same community over multiple partitions of the network. This ensemble analysis of partitionings to find correlations between different sets of network partitions, combined with varying the threshold value used to create a network, enables the investigation of relationships

between communities at different threshold values. Community relationships are defined as hierarchical if communities at a higher threshold value are contained within communities at a lower threshold value. These methods not only allow one to find the hierarchical organization of communities within the network, but also to determine if a network is, in fact, hierarchical – a feature that is not forced upon the network by the method.

Comparisons of independent gene expression experiments often find considerable inter- and even intra-experiment variation, which can amplify stochastic aspects of the community detection process [228, 229, 230]. While variation can be minimized by standardizing the platform and analysis pipeline used, the low-replication common to many gene expression studies, means that the variance of each individual gene expression estimate is typically quite high. To investigate the effects of experimental noise on the ability to assign genes to core communities, artificial data sets are constructed with various levels of experimental noise. At each noise value, multiple runs of the community detection process are performed, allowing the determination of the sensitivity of core community structure to realistic levels of expression variation. It is found that increasing the value of expression noise had a similar effect to increasing the relatedness cutoff value used to create the network. Noise decreases the size of the core communities, leaving only the most strongly related genes as consistent members, but does not tend to assign genes into new core communities. Biological relevance in the communities found by these methods is tested by determining whether they significantly enrich for gene ontology (GO) terms identified in *E. coli*. It is found that, in many cases, there are statistically significant matches between a core community and GO term, indicating that communities are likely to

be biologically relevant. Thus, the methods presented to investigate genetic regulatory networks and to determine the hierarchy of their functional communities appear robust to the variability in the community detection process and to the existence of experimental noise.

2.2 Inferring gene interactions from expression data

In the following work, *E. coli* expression data downloaded from the Many Microbe Microarrays Database (M^{3D}) version 4, build 5 [231] is analyzed. This build consists of a compendium of expression profiles from 730 different experiments reporting expression of 4,298 *E. coli* MG1655 genes. These experiments report the effect on gene expression of 380 different perturbations, of which 152 were repeated at least three times. Experiments include environmental perturbations such as pH levels, growth phase, presence of antibiotics, temperature, growth media, and oxygen concentration, as well as genetic perturbations. For each gene the data from the various experiments were normalized to account for varying detection efficiencies and differences in labeling. The values then reported are the $\log_2$ of the normalized expression intensity.

2.2.1 The context likelihood of relatedness method

To identify interactions between genes the context likelihood of relatedness (CLR) algorithm [88] is applied. Generally, network inference is difficult because of bias from uneven condition sampling, upstream regulation, and inter-laboratory variations in

microarray results. The CLR algorithm attempts to mitigate these difficulties by increasing the contrast between the physical interactions and the indirect relationships by taking the context of each interaction and relationship into account. Links are assigned based on the mutual information in gene expression patterns, which, unlike simple correlation methods, can accommodate non-linear relationships between pairwise gene expression patterns. Although some other algorithms offer higher precision in terms of recovering known regulatory links [232], CLR is attractive for allowing identification of indirect links that might serve to strengthen relationships between genes within co-regulated communities. Note, however, two limitations of networks derived from the underlying data set and CLR approach. First, the expression experiments are not considered as time series, which could give information as to the direction of regulatory interactions [233]. Second, combinatorial regulatory interactions, for example, in which two or more regulator genes must be active to regulate a target gene is not considered.

Implementation of the CLR algorithm begins by calculating the mutual information in the expression data for each pair of genes. This is done by treating the data for each gene as a discrete random variable, so that every pair of genes X and Y is assumed to have expression levels x_i and y_i for each experiment $i = 1, 2, 3, \ldots$. The mutual information $I(X, Y)$ in the expression of X and Y is

$$I(X, Y) = \sum_{i,j} p(x_i, y_j) \ \log \frac{p(x_i, y_j)}{p(x_i)\ p(x_j)}, \tag{2.1}$$

where $p(x_i)$ and $p(y_j)$ are the marginal probability distributions that the expression level of X is x_i and of Y is y_j, respectively, and $p(x_i, y_j)$ is the joint probability distribution that, simultaneously, the expression levels of X and Y are x_i and y_j,

respectively. These discrete probability distributions are calculated from the continuous expression data using B-spline smoothing and discretization. Rather than assign an expression value to one bin, as in classical binning, the B-spline functions allow an expression value to be assigned to multiple bins to account for fluctuations in biological and measurement noise. This is sometimes referred to as "fuzzy binning" [234]. For N genes, this calculation results in an $N \times N$ symmetric matrix of mutual information values. Here, to calculate the probability distributions for *E. coli* we use 10 discrete bins and a third-order B-spline function. The results do vary slightly if the number of bins used or the order of the B-spline function is changed. However, the results vary slowly with these parameters and do not change any principle conclusions.

Mutual information between a gene pair can be due to random background effects, or a regulatory relationship. To distinguish the relevant mutual information from its background, the CLR algorithm compares each mutual information value $I(X, Y)$, to the distribution of the mutual information values between gene X and all other genes $\{I(X, Y); \forall Y\}$, and separately, to the distribution of the mutual information values between gene Y and all other genes $\{I(X, Y); \forall X\}$. The distributions are assumed to be normal and a Z-score value, Z_x and Z_y, is assigned to $I(X, Y)$ for distribution X and Y, respectively. The Z-score value of $I(X, Y)$ compared to a normal distribution i, with a mean μ_i and standard deviation σ_i, is given by

$$Z_i = \frac{I(X, Y) - \mu_i}{\sigma_i} \,. \tag{2.2}$$

Any value of Z_x or Z_y less than zero is set to zero as negative Z-score values represent a mutual information value below the mean and thus below the background. Finally,

the relatedness value between gene X and gene Y is defined as

$$f(X,Y) = \sqrt{Z_X^2 + Z_Y^2}. \tag{2.3}$$

For N genes, this calculation results in an $N \times N$ symmetric matrix of relatedness values.

The resulting CLR relatedness matrix can be used to define a network with weighted links between genes. In principle this network can be analyzed to find its community structure. However, doing so would not allow an exploration of hierarchical community organization. Instead, we apply a threshold value of relatedness, $f_{\min}$, above which a regulatory interaction is inferred.

2.2.2 Defining a network

Once the matrix of relatedness values is calculated, a network of regulatory interactions is inferred by placing links between every pair of genes whose relatedness value exceeds some threshold, $f_{\min}$. For a given $f_{\min}$ value, this procedure results in a defined interaction network. The threshold value $f_{\min}$ that is chosen has considerable effect on the network that is created and on its community structure. The distribution of relatedness value, f, of pairs of genes is shown in Fig. 2.1A. Clearly, increasing the cutoff value significantly reduces the number of links in the network. At $f_{\min} = 2$ all 4,297 genes are in the largest connected component and therefore the network is fully connected (Fig. 2.1B). At approximately $f_{\min} = 4$, the inferred network begins to break up and at $f_{\min} = 6$, the size of the largest connected component is substantially reduced and a number of isolated components exist. Thus, $f_{\min} = 4$ is approximately the critical value at which the network remains largely intact as one connected network. In the work below networks inferred from $f_{\min}$ values of 2,

4 and 6 are considered. These values correspond to points on, and at either side of the critical threshold value.

2.3 Identifying communities and their hierarchical organization

A recently developed extension of the leading eigenvalue method is used to determine the community structure of the inferred *E. coli* regulatory network [4]. This method aims to identify a partitioning of nodes into a disjoint set that maximizes network modularity. In order to solve this problem, the leading eigenvalue method combined with final tuning [9] is used. Final tuning improves the approximate solution given by the leading eigenvalue method by removing constraints that bias the results. For widely studied example networks with up to a few thousand nodes, the size of the genetic network of *E. coli* used in this analysis, combining final tuning with the leading eigenvalue method has been demonstrated to produce network partitionings with the largest $Q_{\max}$ of any known method [9].

Community detection algorithms contain stochastic elements that can cause different runs to give different partitionings. Indeed, partitionings of the same network can be structurally diverse, despite having similar modularity scores [6]. This property is exploited by analyzing an ensemble of partitionings and measuring their correlations. This allows one to both find the pairs of genes that are most often grouped together and examine the family of community structures that can result from a modularity maximization.

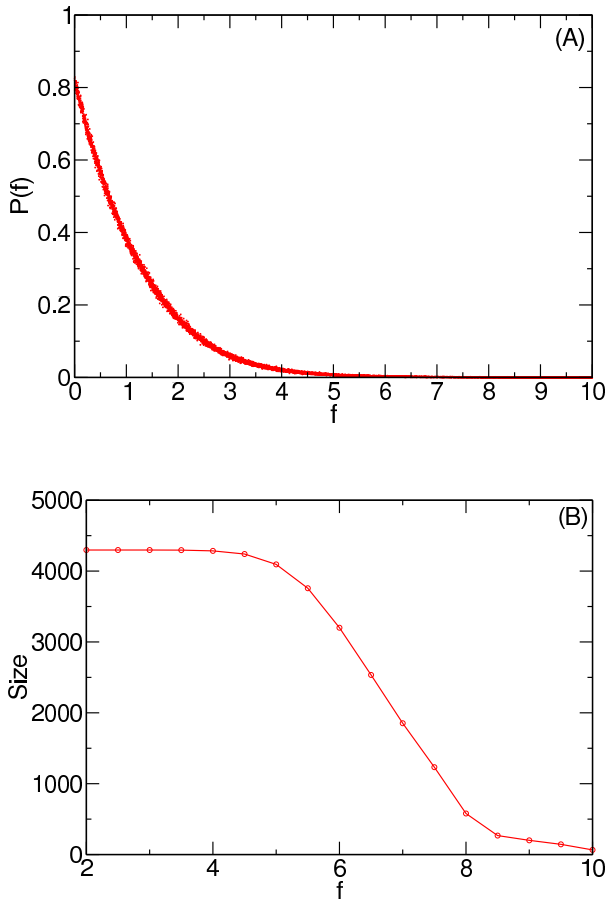


Figure 2.1: **Distribution of gene relatedness and network size in the *E. coli* CLR network.** (A) Probability distribution of relatedness values, f, between pairs of genes in *E. coli* calculated using the CLR algorithm and the full M^{3D} dataset. (B) Size of the largest connected component for relatedness value, f. At small values of f_{min} the network is fully connected but begins to break up into multiple disconnected components at a critical value of approximately $f_{min} = 4$.

At a particular f_{min} value, which defines a unique network, the community detection algorithm is run 10 times, generating a correlation matrix where each element represents the proportion of times gene X and gene Y are found in the same community. We define sets of genes that are always found in the same community as a "core community". We performed this procedure for $f_{min} = 2, 4$ and 6, which, as discussed above, give networks that are supercritical, critical, and subcritical, respectively.

2.3.1 Modularity results

At $f_{\min} = 2$ there are only six communities, while at $f_{\min} = 6$ there is a mode of 965 communities with the largest consisting of 417 genes. This is consistent with the finding that at small values of $f_{\min}$ the network is fully connected, while at large values the network breaks up into a large number of small disconnected parts. At intermediate values of the threshold, where the network begins to break up, the community structure is complex, consisting of a broad distribution of different sized communities. Interestingly, as $f_{\min}$ increases so does the value of the maximum modularity found, $Q_{\max}$. At $f_{\min} = 2$, $Q_{\max} \approx 0.37$ indicating that the network structure is not particularly modular, while at $f_{\min} = 6$, $Q_{\max} \approx 0.85$ indicating that the network structure is highly modular.

Additionally, it important to note the community detection algorithm is run on the entire network of 4297 nodes at all three threshold values, regardless of whether the resulting network is fully connected. At $f_{\min} = 2$, this is not particularly important as the graph is fully connected. However, at $f_{\min} = 4$, the network is composed of 11 disconnected components the largest of which contains 4285 nodes and 9 of which are isolated nodes. Similarly, at $f_{\min} = 6$, the network is composed of 904

isolated components the largest of which contains 3201 nodes and 783 which are isolated nodes. In all cases, community detection results in all isolated nodes partitioned into singleton communities. Similarly, nodes are never partitioned together in the same community if they are members of different isolated components.

2.3.2 Statistical analysis of ensembles of network partitions

The multicolor matrix correlation plot of Fig. 2.2 simultaneously shows the statistical correlations in the modular relationships between pairs of genes at supercritical, critical, and subcritical threshold values. First, single color, blue red and green, matrix correlation plots corresponding to $f_{\min}$ values of 2, 4, and 6, respectively, are created. Note that the order of genes used in a matrix correlation plot is arbitrary. However, by judiciously choosing an ordering, modular relationships become more apparent. The genes in each of these single color correlation plots are then simultaneously reordered as follows. First, the genes were ordered so that all of the genes in the same community at $f_{\min} = 2$ are listed together, according to the size of the community, beginning with the largest and ending with the smallest. Next, the genes in each of those communities are reordered such that the subset of those genes that comprise the largest community at $f_{\min} = 4$ are listed first, followed by those in the next largest such community, *etc.*, until all genes within the $f_{\min} = 2$ community have been listed. Then each of the genes within a $f_{\min} = 4$ core community that are within an $f_{\min} = 2$ community are again reordered. The genes in each of those communities are reordered such that the subset of those genes that comprise the largest core community at $f_{\min} = 6$ are listed first, followed by those in the next largest such community, *etc.*, until all genes within the $f_{\min} = 4$ core community that

are within an $f_{\min} = 2$ community have been listed. Finally, the three single color correlation plots are combined into the multicolor plot shown in Fig. 2.2, where each matrix element of the resulting plot has an RGB color that simultaneously indicates its correlations in the modular structure at each of the three $f_{\min}$ values.

2.3.3 The hierarchical structure of the network

As shown in Fig. 2.2, substantial differences in the community structure of the networks inferred at different $f_{\min}$ values are found. As $f_{\min}$ is increased, links that connect weakly related genes are removed from the network, which can cause genes to switch communities, and communities to merge or divide. Analysis of these changes lead to two conclusions. First, there is a basic community structure that is robustly determined such that many pairs of genes remain in the same community at all three $f_{\min}$ values, indicated by the block diagonal white elements. That is, there is a basic community structure that is invariant with respect to adding or subtracting links between weakly related genes. Second, community structure is hierarchical. To see this, note that at $f_{min} = 2$ the community structure consists of six large communities, indicated by the blue blocks, while at higher values it begins to break up into smaller communities. More importantly, the relationship between communities at different $f_{\min}$ values indicates that the structure of the network is largely hierarchical. A hierarchical structure is revealed when a community breaks up into subcommunities as $f_{\min}$ increases. If the *E. coli* regulatory network was completely hierarchical, one would see only block diagonal elements consisting of large blue blocks that break up into purple then white sub-blocks as $f_{\min}$ is increased. Communities at one value of $f_{\min}$ that are subcommunities of the same community at a smaller $f_{\min}$ value

are therefore hierarchically closer to each other than ones that remain in different communities at the smaller $f_{\min}$ value. Fig. 2.2 indicates that the inferred *E. coli* regulatory network has a largely but not completely hierarchical structure. This is apparent from the large fraction of the blue blocks ($f_{\min} = 2$ communities) that contain on diagonal purple and white blocks ($f_{\min} = 4$ and 6, respectively). However, there are some red off diagonal blocks that indicate a non-hierarchical ordering as $f_{\min}$ is increased from 2 to 4. Furthermore, although the purple $f_{\min} = 4$ blocks largely break up into white blocks as $f_{\min}$ is increased to 6, there are some off diagonal cyan and green blocks that indicate non-hierarchical ordering. About 68% of the core community matrix elements at $f_{\min} = 4$ were hierarchically in core communities at $f_{\min} = 2$, and about 80% of the core community matrix elements at $f_{\min} = 6$ were hierarchically in core communities at $f_{\min} = 4$.

2.4 Community structure is robust to experimental noise

Given the relatively high experimental variation and low replication typical of gene expression measurements, it is of practical interest to determine whether inferred community structure is robust to this source of noise. To explore the effects of experimental noise, the community structure is found in artificial datasets created to mimic the actual data with various levels of experimental noise. To generate these datasets, a restricted set of the actual data is considered consisting of the 152 experiments that were repeated at least three times in the M^{3D} database. For each of the 152 experiments, the mean $m(X)$ and standard error $\sigma(X)$ of the expression

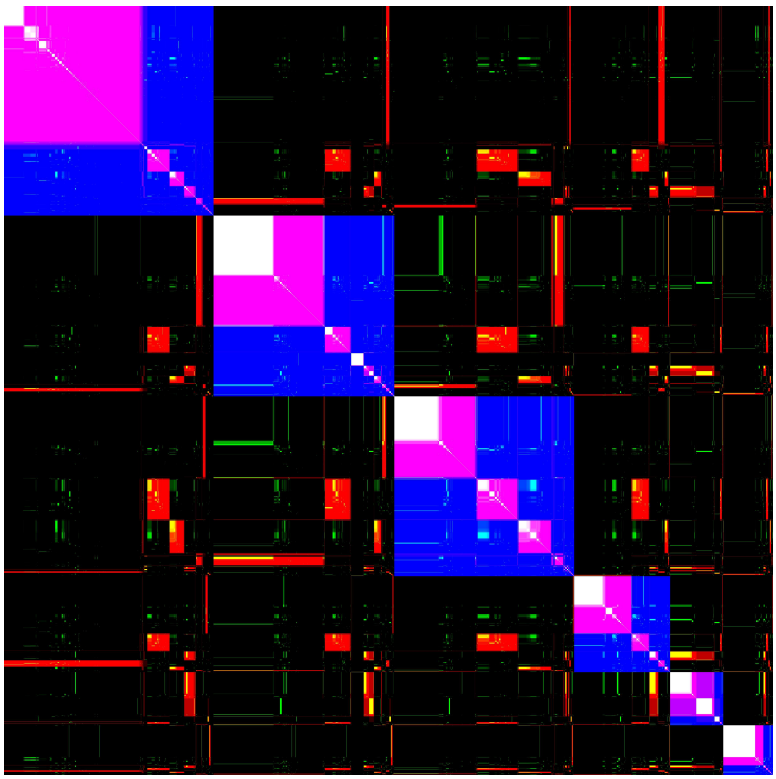


Figure 2.2: **Correlation matrix.** Correlation matrix showing community structure found in the *E. coli* network with relatedness threshold values $f_{min} = 2, 4$ and 6. Genes are ordered in the same sequence along the x and y axes beginning in the upper left corner, and this ordering is the same for all three relatedness values (gene order is given in SI). The matrix element in the position (X, Y) is colored blue, red, or green if genes X and Y are in the same community at threshold values 2, 4 or 6, respectively. The density of the color indicates the strength of the correlation in the partitionings of the pair of genes. For example, considering the correlation between a pair of genes in the 10 replicate partitionings performed on the $f_{min} = 4$ network, dark and light red indicates that the pair of genes are always and rarely found to be in the same community, respectively. The red, green and blue colors corresponding to $f_{min} = 2, 4$ and 6 thresholds, respectively, are combined to indicate the correlations of each pair of genes at all three threshold values. Thus, the color of the matrix element in the position (X, Y) is white if genes X and Y are in the same community at all three threshold values. It is purple (yellow) if the two genes are in the same community at thresholds 2 and 4 (4 and 6), but not at threshold 6 (2) and it is black if the two genes are not in the same community at any of the three threshold values.

level of each gene X is calculated. Assuming a normal distribution of error, artificial data is then generated for an artificial experiment by randomly choosing a value for the expression of each gene X from a Gaussian distribution with mean $m(X)$ and standard deviation $c\sigma(X)$, where c is a positive constant. The amount of noise in the artificial data can be adjusted by varying c with $c = 0$ recreating the original data set. Artificial data sets were generated at values of c ranging from 0 to 4. For each value of c, ensembles of 20 different artificial data sets were constructed and then analyzed. Crucially, these data sets considered each gene and experiment independently, thereby preserving any inherent differences between different gene's expression variability.

2.4.1 Statistical analysis of ensembles of noisy network partitions

For each noisy data set, the CLR algorithm is used to infer a regulation network at an $f_{\min}$ value of 2, and the community structure was determined with the methods described above. For each dataset, 10 different community partitionings were obtained, giving a total of 200 partitionings for each value of c. Fig. 2.3 shows a series of correlation matrix plots for the community structure found for the partitioning ensembles for $c = 0, 0.5, 1, 2$ and 4

The gray scale plots of Fig. 2.3 are the matrix correlation plots for the statistical ensemble analysis of the 200 partitionings constructed from the artificial noisy data for each noise value c. The gray scale of each matrix element in the plots corresponds to the fraction of pairs of partitionings in which the corresponding pairs of genes are found to be in the same community. The order of genes in Fig. 2.3A is such that

all of the genes in the largest core community are arbitrarily listed first, followed by a similar list of the genes in the second, third, fourth, and fifth largest core communities. Note that when $c = 0$ all genes are in one of the five core communities and therefore this list contains all genes. In Fig. 2.3:B, C, D, and E, the genes in each of the 5 core communities at $c = 0$ have been reordered, but the order of the genes with respect to these core communities has been preserved. That is, in each of these subfigures, all genes that are in the i^{th} largest core community at $c = 0$ are always listed before any genes in the j^{th} largest core community at $c = 0$ if $i < j$. In each subfigure, the genes within a $c = 0$ core community have been reordered such that the subset of those genes that comprise the largest core community at the c value corresponding to the subgraph are listed first, followed by those in the next largest such core community, *etc.*, until all genes within the $c = 0$ core community has been listed. Note that, some genes may be isolated in their own core community with this method.

The degree of noise clearly has a major impact on community structure. Nevertheless, except at $c = 4$, there exist robustly determined core communities. In addition this analysis revealed two important results. First, as the noise level c increased, a large proportion of the genes in a core community are partitioned into sub communities but genes rarely switch out of their $c = 0$ core communities. This is similar to what happens when the threshold value for creating the network was increased (Fig. 2.2). Second, with one exception, the number of nodes included in each core community decreased as c was increased (Fig. 2.4A). Noise acts mainly conservatively, decreasing the size of core communities, rather than causing association of genes into new communities.

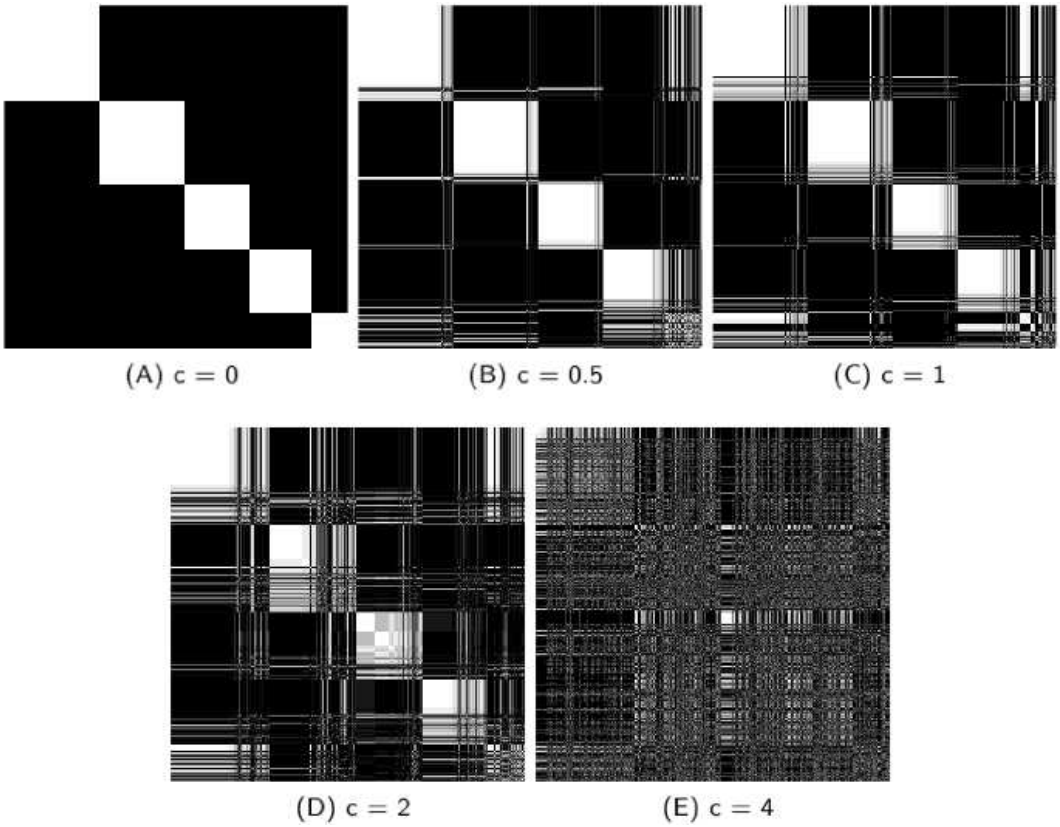


Figure 2.3: **Change in core community structure as noise is increased from c = 0 to c = 4.** The gray scale value of each element indicates the fraction of times the two genes occurred in the same community over replicate community partitionings. If the element is white (black) the two genes were always (never) found in the same community. At each noise value there are clearly white diagonal blocks indicating sets of genes that are always found in the same community, which we refer to as core communities. Note that, the five core communities at $c = 0$ (Figure 2.3A) are in the same order in Figure 2.3:B, C, D, and E. Within each of the five core communities of Figure 2.3A , the node order is allowed to change in Figure 2.3:B, C, D, and E in order to display the largest subcommunity first.

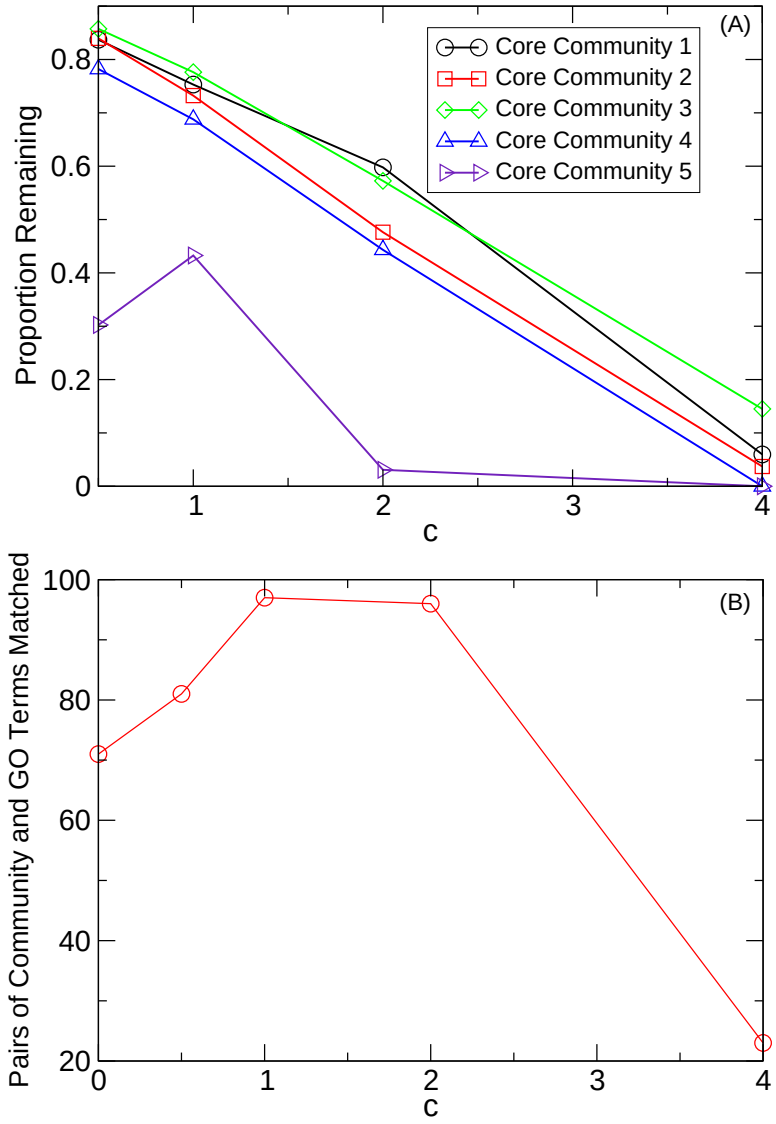


Figure 2.4: **The effect of noise on core community structure and GO term enrichment.** (A) Proportion of $c = 0$ core community nodes that remain in a core community. (B) The number of significant GO term enrichments as a function of noise level c for networks constructed with $f_{min} = 2$. If a GO term is enriched by more than one community, each enrichment is counted separately.

2.5 Communities enrich for functionally related genes

It have thus far been demonstrated that the computational methods presented can robustly identify a community structure in the *E. coli* regulatory network. An important remaining question is whether this structure is biologically relevant. To test this, the simple expectation is examined that genes in the same operon, and that therefore share at least one promoter control region, will tend to group together in the same community. Even using the very stringent requirement that all genes within an operon be in the same community and not accounting for the presence of secondary promoters that are internal to the operon and might act to decouple operon regulation, it is found that genes within an operon are much more likely to group together that expected by chance (Permutation test, $p < 0.001$)(Figure 2.5). For example, given the number and size of communities found at $f_{\min} = 6$, approximately 1% of operons remain together if individual genes are assigned to communities randomly, compared to $> 45\%$ in the community assignments determined by the final tuning algorithm.

2.5.1 Hypergeometric tests

Next a test was performed to determine whether the community structure inferred by our method groups genes with similar biological functions. To account for the resolution limit [187], only core communities larger than 10 genes were considered. Then, these core communities are compared to terms of the gene ontology identified in *E. coli* [235, 236, 237], using a hypergeometric test with Benjamini-Hochberg

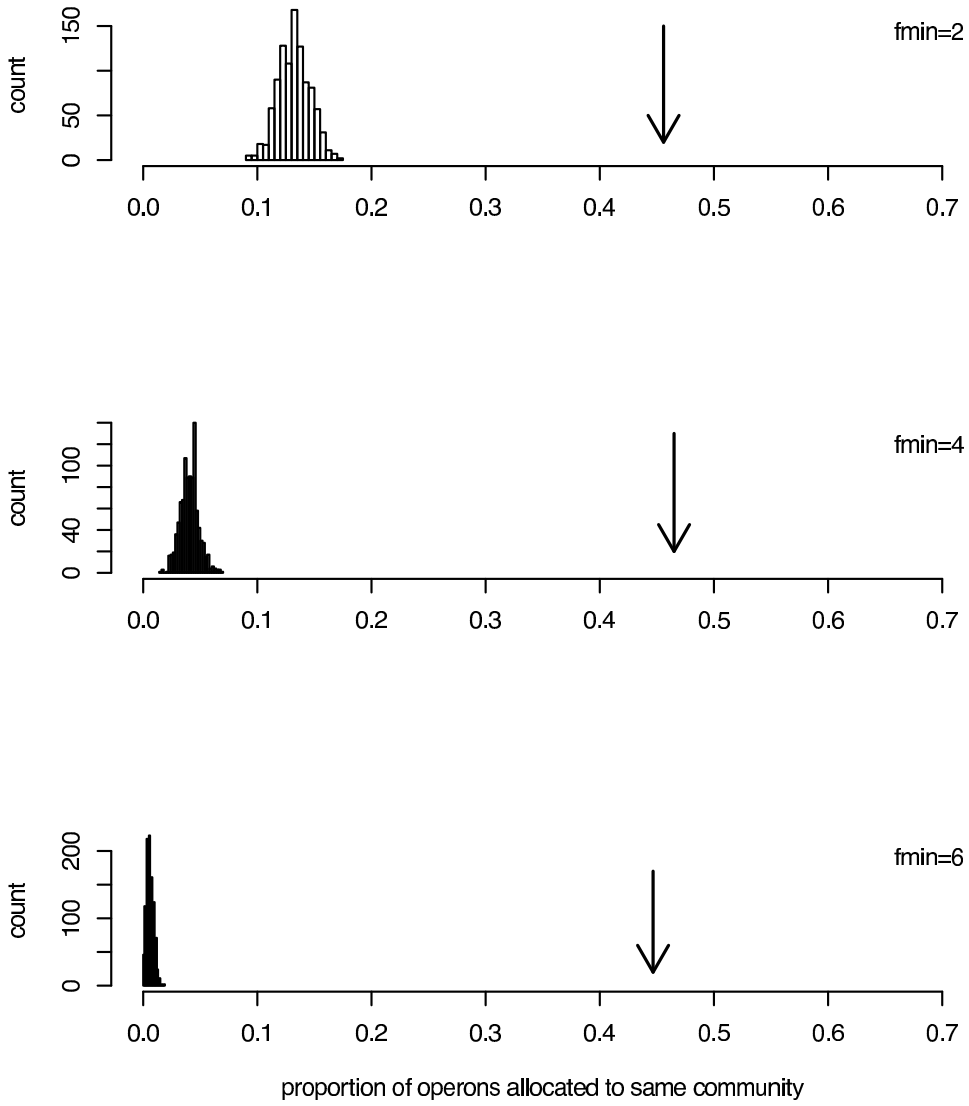


Figure 2.5: **Operon retention.** The fraction of 544 operons (comprising 2172 genes) identified in the *E. coli* genome where all genes in the operon were assigned to the same final tuning community was determined at $f_{min} = 2, 4$ and 6 (indicated by arrows). These actual values were compared to 1000 random distributions of the same set of genes to empty community sets of the same size and number as were present in the final tuning partitionings (histograms). In all cases, actual operon retention proportions were much greater than in any of the 1000 randomly distributed sets, indicating that they were very unlikely to occur by chance and therefore that the final tuning community partitionings effectively group genes in the same operon to the same community.

correction. The hypergeometric test calculates the probability that a community of size n has k genes in common with a GO term of size m in a network with N total genes. For random groupings this probability is

$$P = \frac{\begin{pmatrix} m \\ k \end{pmatrix} \begin{pmatrix} N-m \\ n-k \end{pmatrix}}{\begin{pmatrix} N \\ n \end{pmatrix}}. \tag{2.4}$$

If a community and a GO term are found to have an overlap that is unlikely to occur by chance (a low P value) then their relationship is likely to be relevant. Note that a low P value can occur if the number of genes in common, k, is either greater than or less than expected by chance. For a hypergeometric distribution the expected number of matches is given by mn/N. Only the "positive" enrichments for which $k > mn/N$ are classified as relevant.

To control for false discoveries due to multiple comparisons, the P values obtained are corrected using Eq. 2.4 with the Benjamini-Hochberg (BH) procedure [238]. The BH procedure is implemented as follows. For a given core community, the P values obtained by comparing it to the M GO terms are ordered in a list such that they are increasing, $P_1 \leq P_2 \leq \ldots \leq P_M$. The corrected P values are then taken to be MP_r/r, where r is the rank, or position on the ordered list, of the P value. Then, as is commonly accepted, the relationship between a community and a GO term is judged to be relevant if their corrected P value is less than 0.05.

147, 239 and 288 statistically significant matches are found between core communities and GO terms for communities identified at $f_{\min}$ values of 2, 4 and 6, respectively. Tables 2.1 and 2.2 detail these results for the 25 most enriched relationships found at $f_{\min} = 4$ and $f_{\min} = 6$, respectively. Note that many genes

are described by multiple GO terms; *e.g.*, the gene $flgM$ is a member of all terms in the GO hierarchy: 'flagellin-based flagellum basal body, rod' $\rightarrow$ 'flagellin-based flagellum' $\rightarrow$ 'flagellum' so not all enrichments are independent. Nevertheless, the network partitioning results in communities that significantly enrich for many GO terms, suggesting that the gene groupings are biologically meaningful.

Figure 2.4B shows the number of statistically significant GO term enrichments as a function of noise level, c. Interestingly, enrichment peaks at a noise level of $c = 1$, which corresponds to the artificial data with noise level consistent with that of the experimental data. This is presumably due to the fact that the mean expression values found from the experimental data are estimates, so that a noise value of $c = 0$ will give a precise, but not necessarily accurate estimate of gene expression. As discussed above, increasing the noise in the artificial datasets causes the size of the core communities to decrease. Interestingly, the $c = 0$ core community that dissolves the quickest, core community 5 (numbered beginning in the upper left hand corner of Figure 2.3A), contributes only one significant GO term enrichment at $c = 0$. Finally, note that there are some differences in the identity of core communities when the restricted set of 152 experiments is compared to those generated using the full experimental data (at $f_{\min} = 2$). Nevertheless, as mentioned in Ref. [88], the CLR algorithm can produce nearly equivalent results as the full data set when a small, yet diverse set of expression profiles is chosen. This fact highlights the importance of judiciously choosing experimental conditions when the data set is small.

Table 2.1: The 25 most relevant relationships found for $f_{min} = 4$ without noise.

P value	GO term	Com size	GO size	In com	Description
8.41e-42	9288	72	24	24	bacterial-type flagellum
9.57e-39	6826	53	37	25	iron ion transport
8.22e-38	1539	72	28	24	ciliary or flagellar motility
3.67e-35	6412	826	101	79	translation
6.51e-34	3735	826	56	54	structural constituent of ribosome
3.08e-31	3723	826	105	77	RNA binding
1.73e-29	6935	72	22	19	chemotaxis
4.30e-29	3774	72	17	17	motor activity
5.38e-29	9425	72	17	17	bacterial-type flagellum basal body
2.06e-25	19861	72	15	15	flagellum
5.61e-25	5506	53	210	31	iron ion binding
3.72e-24	19843	826	42	40	rRNA binding
6.98e-23	6811	53	79	22	ion transport
6.99e-22	30529	826	36	35	ribonucleoprotein complex
1.72e-21	5840	826	38	36	ribosome
6.62e-21	8652	247	62	32	cellular amino acid biosynthetic process
4.11e-17	5506	139	210	39	iron ion binding
6.66e-16	9055	139	116	29	electron carrier activity
7.30e-15	51539	139	98	26	4 iron, 4 sulfur cluster binding
8.22e-15	15453	300	15	15	oxidoreduction-driven active transmembrane transporter activity light-driven active transmembrane transporter activity
1.85e-13	6865	247	70	27	amino acid transport
6.13e-13	45272	300	13	13	plasma membrane respiratory chain complex I
9.19e-13	30964	300	13	13	NADH dehydrogenase complex
1.97e-12	9060	300	21	16	aerobic respiration
2.15e-12	5515	826	875	251	protein binding calmodulin binding

Table 2.2: The 25 most relevant relationships found for $f_{min} = 6$ without noise.

P value	GO term	Com size	GO size	In com	Description
1.07E-78	3735	107	56	51	structural constituent of ribosome
3.09E-68	6412	107	101	57	translation
6.45E-57	19843	107	42	38	rRNA binding
7.99E-53	5840	107	38	35	ribosome
8.37E-47	30529	107	36	32	ribonucleoprotein complex
5.60E-42	9288	71	24	24	bacterial-type flagellum
2.84E-40	3723	107	105	42	RNA binding
5.49E-38	1539	71	28	24	ciliary or flagellar motility
5.24E-34	6826	63	37	24	iron ion transport
6.54E-34	45272	14	13	13	plasma membrane respiratory chain complex I
1.31E-33	30964	14	13	13	NADH dehydrogenase complex
9.34E-31	3954	14	12	12	NADH dehydrogenase activity
1.27E-29	6935	71	22	19	chemotaxis
3.29E-29	3774	71	17	17	motor activity
4.11E-29	9425	71	17	17	bacterial-type flagellum basal body
3.18E-28	15453	14	15	12	oxidoreduction-driven active transmembrane transporter activity light-driven active transmembrane transporter activity
4.56E-28	9432	56	23	18	SOS response
1.04E-26	48038	14	18	12	quinone binding ubiquinone binding
1.63E-25	19861	71	15	15	flagellum
1.23E-24	9408	30	26	15	response to heat
1.66E-24	8652	192	62	32	cellular amino acid biosynthetic process
2.61E-24	9061	14	40	13	anaerobic respiration
4.12E-23	22627	107	17	16	cytosolic small ribosomal subunit
5.56E-21	15986	11	11	9	ATP synthesis coupled proton transport
7.23E-20	46933	11	8	8	" hydrogen ion transporting ATP synthase activity, rotational mechanism "

2.6 Inferring candidate regulatory interactions

Partitioning of regulatory networks into communities of genes with similar responses to genetic and environmental perturbations can be used to identify candidate new regulatory interactions between genes. To this end, the communities that most significantly enriched for a GO Term at $f_{\min} = 4$ and $f_{\min} = 6$ are considered, and the relatedness network among the genes within each community are compared to the subnetwork of known regulatory interactions involving these genes presented in RegulonDB [239]. What follows are simply two examples; however, the results contain a wealth of other gene communities whose interactions can be analyzed in a similar manner.

The community with the most significant GO term enrichment at $f_{\min} = 4$ contains 72 genes, including all 24 genes in the GO term for bacterial-type flagellum (Table 2.3). Because of their co-regulation, the remaining 48 genes in this community are implicated as having some relevance for the development, function or control of the *E. coli* flagellum. Indeed, of these genes, many have recognized roles in environmental sensing and signal transduction, functions that are physiologically upstream of flagellum control. An additional 11 genes in the community do not have any annotated function, but two of them, *ycgR* and *yhjH*, contain domains that are consistent with flagellum related activity and five of them (*yjdA yjdZ ynjH ycgR* and *yhjH*) are annotated as being regulated by at least one of the two characterized regulators present in the community (*flhDC* and the flagellum sigma factor, *fliA*) [240, 239]. One further unannotated gene, *ymdA*, is connected to *flhDC* only in the CLR network, and is therefore a candidate for being connected to flagellum regulation as well as having a role in flagellum function. The pattern of connections

in this community also serves to highlight the difference between the RegulonDB (direct regulatory links) and CLR (co-regulation) networks. Ten operons are identified that interact with FlhDC in the CLR but not the RegulonDB network. These interactions might represent previously unknown direct interactions, but are probably best explained as indirect interactions mediated through their direct regulation by FliA, which is regulated by FlhDC (Fig. 2.6).

At $f_{\min} = 6$, the community with the most significant functional enrichment contains 107 genes, including 51 of 56 genes annotated as being structural components of the ribosome (Table 2.4). This very significant enrichment suggests that the 15 genes present in the community that do not have any annotated function might also be involved in translational processes. The most striking aspect of this community, however, is that it contains only one recognized regulator, *fis*, which, as annotated in the regulonDB database, is involved in only a very small fraction of the inferred regulatory interactions (Fig. 2.7). Moreover, no recognized transcription factor serves to indirectly connect regulation of more than three of the community operons and no sigma factor is unique to this community. These observations suggest the presence of some other regulatory factor that is in common to some or all of the genes in the community. One candidate for this factor is ppGpp, a small molecule which, in association with DskA, is known to affect regulation of many ribosome associated genes by decreasing the stability of the RNA polymerase open complex [241]. Indeed, a recent study directly examined the effect of ppGpp on nine of the 51 primary promoters present in the community. In all cases, ppGpp was shown to affect promoter activity in at least one of the tested conditions and a comparison of global gene expression profiles of bacteria that differed in ppGpp levels, found that a further twelve promoters in the community differed in expression by at least 2-fold in response to

ppGpp [7, 8]. Together, these results suggest the remaining 30 promoters in the community as candidates to also be affected by ppGpp.

2.7 Conclusion

Unsupervised methods have been presented for determining communities of co-regulated genes and their hierarchical organization based on expression data profiles collected under a variety of environmental and genetic perturbations. These methods combined the CLR algorithm and a tunable threshold value to infer the underlying regulatory network. A statistical ensemble analysis of the network partitionings that result from a recently developed community detection algorithm was used to determine the network's community structure. Applying these methods to *E. coli* expression data three key results were obtained. i). Regulatory communities in *E. coli* were largely hierarchical so that the effect of increasing (decreasing) the $f_{\min}$ threshold was largely simply to split (combine) the communities found. ii) The structure of the inferred regulatory network was robust to relatively high experimental noise. iii) The regulatory communities found significantly enrich for functionally related gene groupings. These findings are discussed in turn.

The technique used applies a threshold to determine whether mutual information between the expression responses of two genes is sufficient to infer a connecting regulatory link. The value of this threshold influences the size and unity of the inferred network. However, the network structure is relatively invariant to the addition or removal of links between more weakly related genes. There at least two broad mechanisms that might cause genes to be weakly connected in the network. First,

Table 2.3: Genes in the community at $f_{min} = 4$ that enrich GO:9288 bacterial-type flagellum.

Genes in the GO Term	Genes not in GO Term
cheZ, flgB, flgC, flgE, flgF, flgG, flgH, flgI	aer, cheA, cheB, cheR, cheW, cheY, flgA, flgD, flgM, flgN, flhA, flhB, flhC, flhD, flhE, fliA
flgJ, flgK, flgL, fliC, fliD, fliE, fliF, fliG	fliI, fliK, fliL, fliP, fliT, fliZ, flxA, intG, modA, modB, modC, motA, motB, qseB, rcsA, tap
fliH, fliJ, fliM, fliN, fliO, fliQ, fliR, fliS	tar, trg, tsr, uhpT, ves, ybjM, ycgR, yecR, yedN, yfdY, yhjH, yjcZ, yjdA, ykfB, ymdA, ynjH

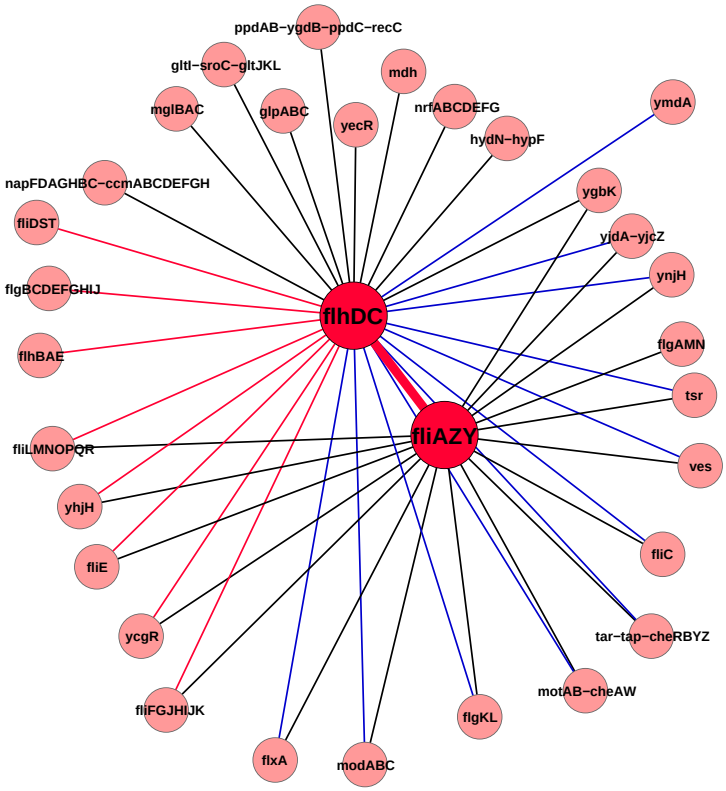


Figure 2.6: **Regulatory links from *flhDC* and *fliA* in the $f_{min} = 4$ community that significantly enriches for flagellum associated genes.** Genes are organized into operons as annotated by RegulonDB. Black, blue and red lines indicate regulatory interactions that are annotated in RegulonDB, inferred in the CLR network or both, respectively. For simplicity, only links from FlhDC and to targets of these links from *fliA* are shown. Many of the interactions that are found in the CLR network are not present in RegulonDB (blue lines). These interactions are candidates for indicating unrecognized regulatory interactions between FlhDC and the target genes. However, in most cases these interactions can be explained through the action of FlhDC on the sigma factor encoded by *fliA* (thick red line), which does directly affect all but one of the target genes. This point underlines the difference between the CLR network, which includes direct and indirect regulatory interactions, and the direct transcriptional network as annotated in RegulonDB. Note the CLR connection between FlhDC and the target gene *ymdA* cannot be explained through any known indirect interaction and is, therefore, a candidate for representing a new direct interaction.

Table 2.4: Genes in the community at $f_{min} = 6$ that enriches GO:3735 structural constituent of ribosome

Genes in the GO Term	Genes not in GO Term
rplA, rplB, rplC, rplD, rplE, rplF, rplI, rplJ, rplK, rplL, rplM, rplN, rplO, rplP, rplQ, rplR, rplS, rplU, rplV, rplW, rplX, rplY, rpmA, rpmB, rpmC, rpmD, rpmE, rpmG, rpmH, rpmJ, rpsA, rpsB, rpsC, rpsD, rpsE, rpsF, rpsG, rpsH, rpsI, rpsJ, rpsK, rpsL, rpsM, rpsN, rpsO, rpsP, rpsQ, rpsR, rpsS, rpsT, rpsU,	cdsA, cmk, dnaG, dusB, efp, fis, fusA, gidB, gmk, infB, ispU, lpxB, mnmG, mrdA, murA, nusA, nusG, obgE, parE, ppa, prfC, priB, pyrH, queA, rbfA, rho, rimM, rlmN, rnhB, rnpA, rpoA, rpoZ, secE, secG, secY, speA, speB, tff, tig, trmA, trmD, trmI, truB, truC, tsf, typA, yadB, yggN, ygiQ, yhbC, yhbE, yhbY, yidC, yidD, yqcC

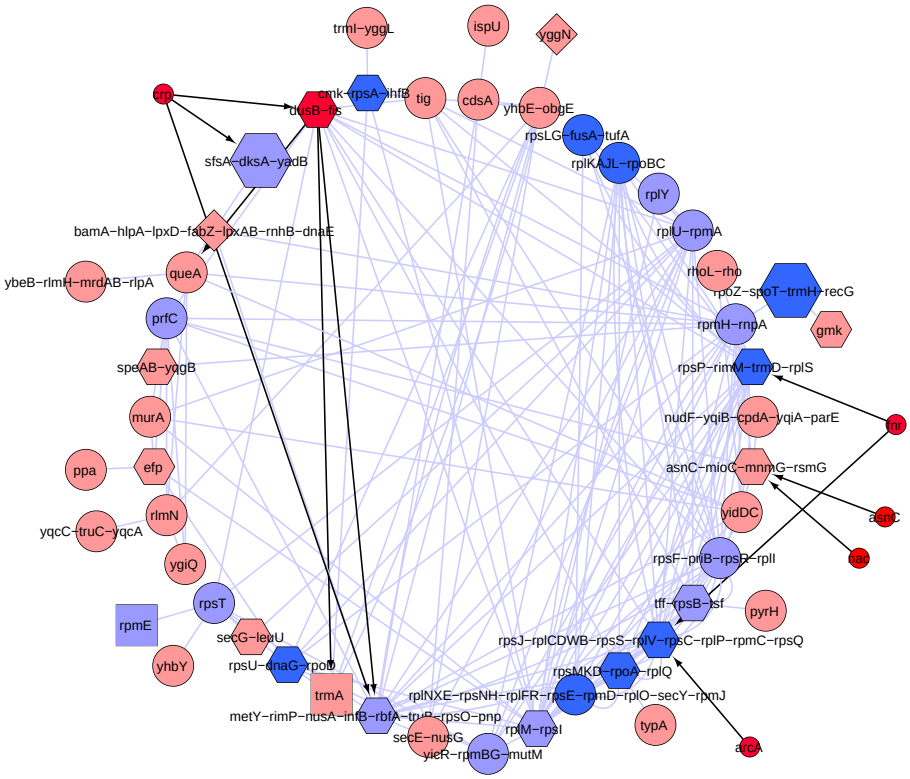


Figure 2.7: **Links connecting operons in the $f_{min} = 6$ community that enriches for genes involved in ribosome structure.** CLR links are in light blue, RegulonDB links are in black. Small symbols are genes that are not in the community, but are regulators of genes that are in the community and are therefore candidates for mediating indirect interactions between community genes. Symbol shape and color indicate attributes as follows: red, transcription factors; dark blue, ppGpp regulated promoter by direct assay [7]; light blue, ppGpp regulated translation related promoter by microarray [8]; pink, other; hexagon, $\sigma 70$ promoter; diamond, $\sigma 24$ promoter; square, $\sigma 32$ promoter; circle, unknown sigma factor. Note that very few interactions observed in the CLR network can be explained by the direct interactions annotated in RegulonDB. The high proportion of ppGpp sensitive promoters among operons contained in the community suggests this molecule as a good candidate for regulating the remaining interactions. The network layout was determined by the circular layout option in Cytoscape 2.8.1, no particular significance should be attached to operons being outside the main circle.

the relevant molecular interactions may exert weak expression control on the regulated gene. Second, the regulatory interactions might be environmentally dependent, being active in only a subset of the experimental conditions. Comparison of communities present in regulatory networks obtained at increasingly stringent thresholds indicates that the regulatory network is largely hierarchical such that large communities present in the low threshold network tended to split into smaller sub-groups of strongly related genes as the threshold was increased. By contrast, increasing the threshold causes relatively few genes to associate in new communities that were not subsets of the original communities.

Relatively high experimental noise is of considerable concern in analysis of gene expression data. Indeed, even small differences in preparation and sample growth conditions, or in the exact platform and analysis procedure used, can manifest as substantial differences in gene expression estimates [228, 229, 230, 242]. To address the influence of experimental noise on the ability identify regulatory interactions and communities, datasets were generated with different noise levels, calculated independently across experiments and genes. Comparing communities identified in networks inferred from these data sets, it was found that not only are the predictions for the functional communities robust against noise up to double that seen in the original empirical dataset, but that the effects of experimental noise are mainly conservative. That is, experimental noise reduced the size of core regulatory communities but did not tend to create new communities.

For the purpose of identifying functional communities in a biological network it was useful to study the community structure of different networks constructed with a range of relatedness threshold values. At large threshold values, the nodes in each of the small disconnected pieces were highly related. These small groups

provided the most statistically significant enrichments for GO terms and thus best identify biologically relevant communities. However, as the threshold value used to construct a network is reduced, the community sizes tended to increase. These enlarged communities include other nodes that may also be relevantly related to the core communities found at higher threshold values. Because of these competing considerations, if only one threshold value is to be chosen for which to make biological comparisons, it is suggested that the critical threshold value should be used, which for *E. coli* is approximately $f_{\min} = 4$. Choosing the critical value will not only balance the above two considerations, but as discussed earlier, also gives the most statistically complex distribution of community structure.

The usefulness of the methods presented are multifold. First, the functional community predictions of the methods can be used to refine existing knowledge of the functional relationships of genes in well known organisms such as *E. coli*. That is, the overlap of the core communities with the *E. coli* GO terms was not exact, suggesting that the additional genes in the core communities that enrich a particular GO term may themselves be candidates for genes that should be included in that term of the gene ontology. In this way, the predictions of these methods can be used to suggest new experiments to refine our understanding of the *E. coli* regulatory system. It has been explicitly demonstrated how this can be done by analyzing two of the communities found with the methods that significantly enrich GO terms and predicting previously unknown regulatory interactions. Furthermore the methods can readily be applied to expression data for other, less well studied, organisms, and to other types of biological data, to identify functional communities in their networks. The predictions from the unsupervised methods will be particularly useful, for making initial approximate predictions for the functional communities

and their organization of less well known organisms. Additionally, it should be noted that the methods have been applied to expression data based on an arbitrary variety of experimental and genetic perturbations. However, the methods could instead be applied to more targeted sets of expression data. For example, data based on particular types of environmental perturbations, particular types of genetic knockouts, with cells in a particular stage of the cell cycle, or with cells in a particular developmental stage of a multi-cellular organism. By examining more targeted data of these sorts, the dynamics of particular functional communities can be explored.

Chapter 3

Maximizing the Modularity in Bipartite Networks

Community detection is an important tool when studying the large-scale structure and function of complex networks. For unipartite networks, such as the *E. coli* regulatory network studied in Chapter 2, maximizing the modularity Q [2] provides an effective method to investigate its structure. However, an important class of complex networks that have a wide range of real-world applications is bipartite networks. In a bipartite network, there are two sets of nodes where each node is only connected to nodes from the other set. Additionally, links in a bipartite network may be weighted by a real value to represent the relative strength of connection between different pairs of nodes.

Many real-world complex networks that occur have a natural bipartite topology. In the social sciences, bipartite graphs commonly arise in the study of affiliation or

collaboration networks [165, 166, 167, 168, 169, 170]. In scientific collaboration networks, the two sets of nodes represent authors and papers with a link representing authorship. Newman, for example, studied scientific collaboration networks from databases such as MEDLINE, the Los Alamos e-Print Archive, and NCSTRL [167]. These bipartite networks exhibit some properties similar to those of unipartite networks. For example, the distribution of the number of papers a scientist writes and the number of collaborators of a scientist both exhibit a power-law form and the networks are highly clustered. Additionally, some biological networks naturally have a bipartite structure [171, 172, 173]. In the human disease network studied by Goh *et al.* [243], one set of nodes represents genetic diseases and the other all known disease genes in the human genome. Then, links represent a known association between a mutation in the gene and a genetic disease. Goh *et al.* studied separately the "human disease network" and the "disease gene" network by projecting each side of the network onto a unipartite graph. Interestingly, by analyzing the structure of each graph, it was found that the majority of disease encoding genes are nonessential and that genes associated with the same disorder share common functional characteristics.

As previously discussed in Section 1.6, modularity Q, defined for unipartite networks, is not suited to detect communities in bipartite networks as there is a random expectation that links will exist between any two nodes in a network. As a consequence, methods have been developed to detect communities in bipartite networks that broadly fall under two avenues of analysis. Either one biclusters the nodes of a bipartite network, defined as partitioning both types of nodes into the same community, or one projects each type of node, separately, onto a unipartite network and detects communities in the unipartite network. In the latter case, each projected network is composed of nodes of one type and therefore communities contain only

nodes of one type. If a projection method is used, a unipartite network is defined, and communities can be detected using methods discussed in Chapter 1. Additionally, a modification of the modularity for bipartite networks using a projection has been introduced by Guimera *et al.* [175] and will be discussed below. Alternatively, for a biclustering analysis some of the methods in Chapter 1 such as edge clustering [244], clique percolation [245, 246], and modularity [176] have been generalized to detect communities in bipartite networks by biclustering. As discussed in Chapter 1, modularity quality functions are particularity suited to the problem of community detection as they, in general, explicitly contain the definition of a community, compare each community in a partition to a null model, do not require as an input the number of communities in a network, and measure quantitatively the strength of a partition. Therefore, this chapter studies the problem of modularity maximization in bipartite networks when the links can be either unweighted or weighted using both methods of bipartite analysis.

First, a biclustering modularity introduced by Barber [176] is defined and described. This method is a straight-forward extension of Newman's modularity Q and generalizes easily to weighted links. Next, a modularity by Guimera *et al.* [175] is studied that projects each side of the network onto a unipartite network involving only nodes of one type using a multi-link projection. Unfortunately, this method does not extend to weighted bipartite networks. Therefore, a new bipartite modularity is introduced, called the Information Preserving (IP) modularity, which extends the ability to partition each side of bipartite networks separately when the links in the bipartite network are weighted. Additionally, this method preserves more information about the bipartite network in its unipartite projection than does Guimera's modularity. Next, the leading eigenvalue method of modularity maximization is adapted

for each bipartite modularity and a new tuning step is added that merges communities. A bipartite Erdös-Rényi type network is defined as a network with P nodes of type 1 connected to Q nodes of type 2 with probability p. We then show how each tuning step significantly improves the partitions found when maximizing each modularity in an ensemble of bipartite Erdös-Rényi type networks and real world bipartite networks.

Finally, in the last section of this chapter, the expected value and standard deviation of IP modularity for bipartite Erdös-Rényi type networks is analytically derived so that a Z-score comparison of the modularity values of real-world networks can be computed. Calculating the Z-score allows the significance of the modularity of the partitioning to be quantified.

3.1 Bipartite modularity

As mentioned in the previous section, there is a fundamental issue for bipartite networks when choosing a quality function to maximize. Either both types of nodes can be included in the same community defining a biclustering analysis or each side of the network can be partitioned separately using projection methodologies. To address this issue, two standard definitions have been introduced.

3.1.1 Barber modularity

Bipartite modularity, as defined by Barber [176], partitions nodes from each set into the same community, which is called biclustering. For a network of N nodes, q of

the first type and p of the second type, the modularity is defined as

$$Q^B = \frac{1}{2m} \sum_{i,j} B_{ij} \delta_{C(i),C(j).} \tag{3.1}$$

$C(i)(C(j))$ is the community to which node $i(j)$ belongs, m is the total number of links in the network and δ is the Kronecker delta function. $\mathbf{B}$ has the form

$$\mathbf{B} = \begin{bmatrix} \mathbf{O}_{p\times p} & \tilde{\mathbf{B}}_{p\times q} \\ \\ \tilde{\mathbf{B}}^T_{q\times p} & \mathbf{O}_{q\times q} \end{bmatrix}, \tag{3.2}$$

where $\tilde{B}_{ij}$ is a $p \times q$ matrix where node i belongs to the p nodes of the first type and node j belongs to the q nodes of the second type. Then, $\tilde{B}_{ij} = A_{ij} - k_i d_j/(m)$, A_{ij} is the adjacency matrix, and $k_i(d_j)$ is the degree of node $i(j)$. Barber's definition of bipartite modularity is similar to Newman's unipartite definition. However, Barber's definition accounts for the fact that in a bipartite network one would never expect nodes of the same set to be connected. Elements of the modularity matrix that correspond to pairs of nodes in the same set are therefore 0.

3.1.1.1 Weighted links

There are advantages to biclustering a bipartite network in this way. With both types of nodes in the same community it is easier to conceive why nodes of the first type were included together in a community. They are connected through the nodes of the second type in the same community. Additionally, Barber's modularity does not require a projection in which case information about the bipartite network is lost. Furthermore, Barber's bipartite modularity is easily generalized if the links between the nodes in the network are weighted. For positive, real valued elements A_{ij} of the adjacency matrix, the degree of node i, k_i, is given by $k_i = \sum_j A_{ij}$, the total weight

connected to node i. Similarly, m is simply sum of all link weights in the network, $m = \sum_{ij} A_{ij}/2$.

3.1.2 Guimera modularity

One can also partition each set of nodes in a bipartite network separately. Bipartite modularity, as defined by Guimera *et al.* [175], labels the set to be partitioned the actor set and the other the team set. For a network of q actors and p teams, the modularity for the actor set is defined as

$$Q^G = \sum_{i,j}^{q} \left(\frac{c_{ij}}{\sum_{a}^{p} m_a \left(m_a - 1\right)} - \frac{t_i t_j}{\left(\sum_{a}^{p} m_a\right)^2} \right) \delta_{C(i),C(j)}, \tag{3.3}$$

where $t_i(t_j)$ is the degree of actor $i(j)$, c_{ij} is the number of times actor i and actor j are found to be in the same team, and m_a is the degree of team a. Note that $B_{ii} = 0$ is defined to be 0 for all i.

Guimera's definition of bipartite modularity projects each set of nodes in the bipartite network, separately, onto a unipartite multigraph [175]. Then each projected network can be partitioned separately with the advantage that each set of nodes can be partitioned into a different number of communities. To better understand this modularity function, consider that each community, s, in the Guimera modularity makes a contribution to the total modularity of

$$\frac{\sum_{i \neq j \in s} c_{ij}}{\sum_{a} m_a \left(m_a - 1\right)} - \frac{\sum_{i \neq j \in s} t_i t_j}{\left(\sum_{a} m_a\right)^2}. \tag{3.4}$$

Consider now each team in the bipartite network with degree m_a. Each actor connected to this team will be connected to $m_a - 1$ actors in the multigraph projection. Therefore, each team contributes $m_a(m_a - 1)/2$ links to the multigraph projection and the total number of these links, L_{mp}, is given by

$$L_{mp} = \frac{\sum_a m_a (m_a - 1)}{2}. \tag{3.5}$$

Additionally, $\sum_{i \neq j \in s} c_{ij}$ is two times the number of inner module links, l_{mp}, in the multigraph projection. The first term in Eqn. 3.4 can then be written as

$$\frac{\sum\limits_{i \neq j \in s} c_{ij}}{\sum\limits_a m_a (m_a - 1)} = \frac{2l_{mp}}{2L_{mp}} = \frac{l_{mp}}{L_{mp}}. \tag{3.6}$$

Similarly, the total number of links, L, in the original bipartite networks is given by $L = \sum_a m_a$. Next, rewrite the term $\sum_{i \neq j \in s} t_i t_j$ as

$$\sum_{i \neq j \in s} t_i t_j = \sum_{ij \in s} t_i t_j - \sum_{i \in s} t_i^2, \tag{3.7}$$

and define $\sum_{iji \in s} t_i t_j = d_s^2$ where, d_s is the total degree of community s in the bipartite network. Then, the each community contribution to the modularity can be written as

$$\frac{l_{mp}}{L_{mp}} - \left[\left(\frac{d_s}{L} \right)^2 - \frac{\sum_{i \in s} t_i^2}{(L)^2} \right]. \tag{3.8}$$

Thus, the Guimera modularity compares the fraction of inner-module links of the multigraph projection, l_{mp}/L_{mp}, to the expected fraction of links of the same module in the bipartite network, $(d_s/L)^2$ minus a correction term, $\frac{\sum_{i \in s} t_i^2}{(L)^2}$, that subtracts the expected fraction of links due to self loops.

3.1.2.1 Weighted links

Unfortunately, the Guimera modularity is ill defined when the links in the bipartite network are weighted. As discussed in section 1.6, the problem is due to the term c_{ij} of Eqn. 3.3 which is defined as the number of times actor i and actor j are found to belong to the same team k. Consider again Fig. 1.10 where the link between an actor and a team is weighted. Then the number of times actor i and actor j are considered in the same team k is not easily generalized in the case of weighted links because each actors membership in team k, A_{ik} and A_{jk}, can take on any real value and are not necessarily equal. For example, possible generalizations of c_{ij} could involve the product or, separately, the sum, of A_{ik} and A_{jk}. Therefore, a projection is needed which is defined when the links in the bipartite network are both weighted and unweighted. This problem is further studied in Section 3.1.3.

3.1.2.2 Singleton clusters

In this section, we will analyze a phenomenon that occurs when finding the partition that maximizes the bipartite modularity defined by Guimera *et al.* in which there are many nodes that are partitioned separately into individual communities or clusters of size one. We will refer to such clusters as "singleton clusters." An important question is if this phenomenon is a direct consequence of this particular definition of the Guimera modularity or if it is dependent on the algorithm used to maximize the modularity.

To analyze the occurrence of singleton clusters, we will consider the bipartite network of Fig. 3.1. For this example network, there are two team nodes and $X + Y + Z$ actor nodes. There are X nodes only connected to the team on the

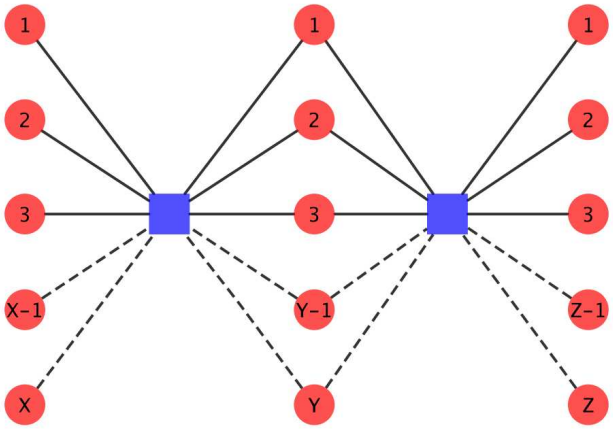


Figure 3.1: **An example bipartite network.** The network contains two (blue square) team nodes and X +Y +Z (red circle) actor nodes. There are X nodes only connected to the team on the left, Y nodes connected to both teams and Z nodes only connected to the team on the right.

left, Y nodes connected to both teams and Z nodes only connected to the team on the right. For such a network, singleton clusters often appear when the X nodes are partitioned into one community, the Z nodes are partitioned into one community, and there are Y singleton clusters. For this to occur, the contribution to the modularity when the Y nodes are partitioned into one community must be smaller than the contribution of the Y singleton clusters. The contribution to the modularity for these partitions are now analyzed.

If the Y nodes are partitioned into a single community together, then each pair of nodes would have two teams in common and therefore contain two links between them in the multi-graph projection. Therefore, the number of inner-module links, l_{mp}, would be equal to a fully connected network of Y nodes with link weight two and given by

$$l_{mp} = Y(Y-1). \tag{3.9}$$

Similarly, the total number of links in the multi-graph projection would be

$$L_{mp} = (X+Y)(X+Y-1) + (Y+Z)(Y+Z-1). \tag{3.10}$$

The total degree of this module in the bipartite network is given by $d_s = 2Y$, the total number of links in the bipartite network is $L = X + 2Y + Z$ and the correction term is $\sum_i t_i^2 = 4Y$. Therefore, for Y nodes in a community the contribution to the modularity is given by

$$\begin{aligned} &\frac{Y(Y-1)}{(X+Y)(X+Y-1)+(Y+Z)(Y+Z-1)} \\ &- \left[\frac{(2Y)^2 - 4Y}{(X+2Y+Z)^2}\right]. \end{aligned} \tag{3.11}$$

If instead, each actor connected to both teams is partitioned into a singleton cluster, then for each of these Y singleton clusters, $l_{mp} = 0$, $d_s = 2$, and $\sum_i t_i^2 = 4$.

Therefore, the first term is zero and the contribution for Y singleton clusters is given by

$$-Y\left[\frac{(2)^2-4}{(X+2Y+Z)^2}\right]=-Y\left[\frac{(4-4)}{(X+2Y+Z)^2}\right]=0. \tag{3.12}$$

Therefore, there is no penalty to the modularity for partitioning this set of nodes into singleton clusters and the condition for singleton clusters to occur is if the contribution of Y nodes into a single community is negative. This is exactly when this partition is no longer a community according to the definition of Guimera *et al.*

Then, the condition to find singleton clusters for the example network of Fig. 3.1 is when the modularity contribution of Y singleton clusters is greater than the modularity contribution of the Y nodes in a single community

$$\begin{aligned}&\frac{Y(Y-1)}{(X+Y)(X+Y-1)+(Y+Z)(Y+Z-1)}\\&-\left[\frac{(2Y)^2-4Y}{(X+2Y+Z)^2}\right]<0.\end{aligned} \tag{3.13}$$

Under the assumptions $X \geq 1$, $Y \geq 1$, $Z \geq 1$, and $Z \geq X$, the inequality of Eqn. 3.13 is satisfied when

$$\begin{aligned}&Z>5\\&X<1+Z-\sqrt{5+4Z}\\&Y<\frac{1}{4}(-2X+X^2-2Z-2XZ+Z^2).\end{aligned} \tag{3.14}$$

When these conditions are satisfied the modularity for the partition with Y singleton clusters, X nodes in a community, and Z nodes in a community will always be the maximum modularity possible for any partition in this example network. The

modularity is then given by

$$\begin{aligned} M &= \frac{X(X-1)}{2L_{pn}} - \frac{[X^2-X]}{L^2} \\ &\quad + \frac{Z(Z-1)}{2L_{pn}} - \frac{[Z^2-Z]}{L^2}. \end{aligned} \tag{3.15}$$

The surface $Y(X,Z) = \frac{1}{4}(-2X + X^2 - 2Z - 2XZ + Z^2)$ is plotted in Fig. 3.2 for $1 < X < 15$ and $5 < Z < 40$. If, for any point Y(X,Z), the number of nodes Y is below this surface, then singleton clusters will exist for these values. From the surface, we can see that for particular values of X and Z the Guimera modularity gives singleton clusters for a large range of Y values. For example, when $X = 5$ and $Z = 15$, singleton clusters exist for $1 < Y < 15$. Figure 3.3 shows the partition found when maximizing the modularity for $X = 5$, $Y = 14$, and $Z = 15$. The colors represent different communities and there are indeed 14 singleton clusters. The modularity for this partition is $M = 0.099480$. When $X = 5$, $Y = 15$, and $Z = 15$ the condition for singleton clusters is no longer satisfied and the X and Y nodes, or the Z and Y nodes, are merged into one community. As the difference between the size of X and Z increases, so does the allowed range of Y. For example, when $X = 5$ and $Z = 30$, singleton clusters exist for $1 < Y < 139$.

In conclusion, the existence of singleton clusters is due to the community definition for the bipartite modularity as defined by Guimera *et al.* It is a direct consequence of defining $B_{ii} = 0$ which does not penalize the partition of nodes into singleton clusters. The existence is therefore independent of the chosen algorithm to find the partition that maximizes this modularity.

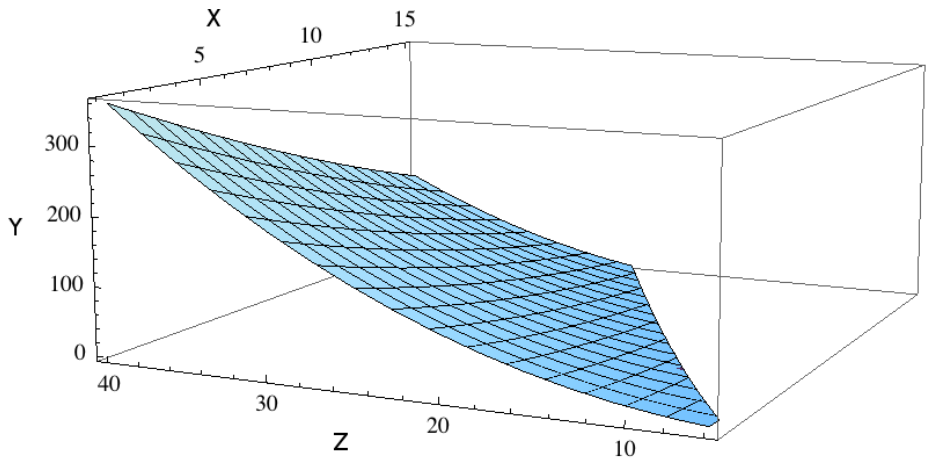


Figure 3.2: **The $\mathbf{Q^G}$ singleton conditions.** A portion of the surface $Y = \frac{1}{4}(-2X + X^2 - 2Z - 2XZ + Z^2)$. For any point $Y(X, Z)$ below the surface singleton clusters will exist in the maximum modularity partitions for Q^G.

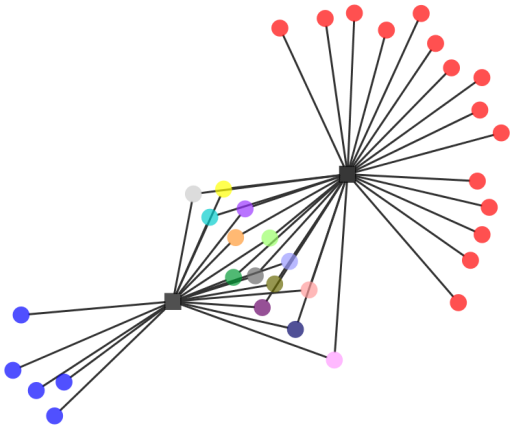


Figure 3.3: **The $\mathbf{Q^G}$ singleton clusters.** The maximum modularity partition for the example network in Fig. 3.1 when $X = 5$, $Y = 14$ and $Z = 15$ when maximizing the Q^G modularity. The colors represent different communities and there exist 14 singleton clusters.

3.1.3 Information preserving modularity

To detect communities in bipartite networks, a projection is needed that: i) generalizes to the case when links in the bipartite network are weighted, ii) preserves information about the bipartite network, and iii) does not produce singleton clusters. A projection that satisfies condition ii has been introduced by Newman [164] and was described in Section 1.6. Newman created a projection of the scientist side of scientific collaboration networks by assigning a weight between scientist i and j using the formula

$$w_{ij} = \sum_k \frac{\delta_i^k \delta_j^k}{n_k - 1}, \tag{3.16}$$

where k runs over all papers, δ_i^k is 1 if scientist i is a co-author of paper k and 0 otherwise, and n_k is the degree of paper k. Thus, Newman considered two scientists co-authorship in a paper as stronger if they were the sole co-authors weighting each term in the sum by $n_k - 1$. As was shown in Fig. 1.9, this projection preserves more information than the multilink projection used in the Guimera modularity. Unfortunately, this projection assumes, in both the terms in the numerator and denominator of the summation, that the links in the bipartite network are not weighted. The delta functions δ_i^k and δ_j^k are defined to be either zero or one. Additionally, if two scientists are the sole authors of a paper the lowest possible degree for a paper, n_k, is two. Thus the denominator normalizes the degree of each paper by subtracting one so that the smallest possible value in the denominator of the summand is one. If the links in a bipartite network are weighted, the possible degree of a paper, n_k, extends to any real value greater than zero in which case subtracting one would result in a negative term in the summand when $0 < n_k < 1$ and zero if $n_k = 1$. For these reasons, Newman's projection is not suitable for weighted bipartite networks and so

a new modularity method is now introduced.

Assume again that the set of p nodes in the bipartite network to be partitioned is the "actor" set and the other set of q nodes is the "team" set. The IP modularity detects communities in bipartite networks by creating a weighted unipartite projection from the nodes in the actor set and detect communities in the weighted projection. The projected link weight itself contains information about the bipartite nature of the network. For each pair of actors (i, j), $i \neq j$, the new link weight, w_{ij}, in the weighted unipartite projection is given by

$$w_{ij} = \sum_{k=1}^{q} \frac{2r_{ik}r_{jk}}{m_k}, \tag{3.17}$$

where $r_{ik}(r_{jk})$ is the link weight in the bipartite network between node $i(j)$ and team k and m_k is the degree of team k. By dividing each term in the summand by the team degree, a higher link strength between actors is assigned when there is mutual membership in a team with a small degree. Note the link weight w_{ii} is defined to be 0.

Communities in the weighted unipartite projection are then detected by finding the partition that maximizes the modularity given by

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta_{C(i),C(j).} \tag{3.18}$$

Here, A_{ij} is the element of the adjacency matrix corresponding to actor i and j and is given by w_{ij}. The variable $k_i(k_j)$ is the degree of node i(j) where $k_i = \sum_j w_{ij}$, $C(i)(C(j))$ is the community to which node $i(j)$ belongs, $m = \sum_{ij} w_{ij}/2$ is the total number of links in the projected network and δ is the Kronecker delta function. The IP modularity is defined for weighted links but is applicable for unweighted bipartite networks as well as $r_{ik} \in \{0, 1\}$.

In addition to its natural extension to weighted links, the projected link weight w_{ij} carries more information about the original bipartite network than the multi-link projection used in the Guimera modularity. For example, consider the bipartite network in Fig 3.4a, where there are 5 nodes of type X, 4 nodes of type Y, and 10 links. The projection of type X using the Guimera modularity method is shown in Fig 3.4b. This projection distinguishes node pairs (1,2) and (2,3) from the rest of the network as each pair has 2 nodes of type Y in common. However, the two neighbors of pair (1,2), nodes 6 and 7, are solely connected to nodes 1 and 2. Pair (2,3) also shares two neighbors; however, one of their neighbors, node 4, is connected to a total of four nodes. Using the IP modularity projection, as shown in Fig. 3.4c, preserves this information and the strength of node pair (1,2) is distinguished from (2,3).

3.1.3.1 Singleton clusters

To study the occurrence of singleton clusters using the IP modularity, again consider the network of Fig 3.1. As shown in section 1.5, the contribution of each community in the projected network to the IP modularity can be written as

$$\frac{l_s}{L} - \left(\frac{d_s}{2L}\right)^2, \tag{3.19}$$

where, l_s is the total number of links contained in community s, d_s is the total degree of community s, and L is the total number of links in the projected network. For a weighted network the "degree" of each node is equivalent to the sum of each link weight adjacent to the node. It follows that L is the sum of all link weights in the projected network, l_s is the sum of all projected link weights in community s, and d_s is the sum of each node degree in community s.

Using Eqn. 3.19 and Eqn. 3.17, the total number of links L in the projected

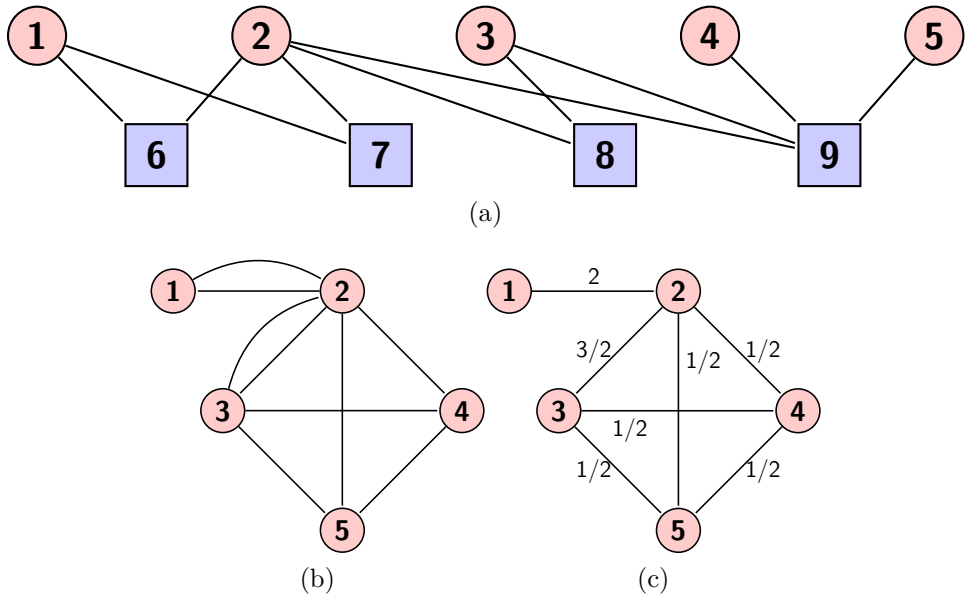


Figure 3.4: **Bipartite projections**. (a) a bipartite graph consisting of 5 nodes of type X and 4 nodes of type Y. (b) the Guimera modularity multi-graph projection of (a), with each link representing a common neighbor between the pair of nodes. (c) the IP modularity weighted projection of (a), where the link weight between a pair of nodes is given by Eqn. 3.17.

network is given by

$$\begin{aligned} L = \quad & \frac{X(X-1)}{X+Y} + \frac{2XY}{X+Y} \\ & + Y\left(Y-1\right)\left(\frac{1}{X+Y} + \frac{1}{Y+Z}\right) \\ & + \frac{2YZ}{Y+Z} + \frac{Z(Z-1)}{Y+Z}. \end{aligned} \tag{3.20}$$

The contribution to the IP Modularity when either the X nodes are partitioned in a single community or the Z nodes are partitioned in a single community has the same form and is given by

$$M1_{W \in X,Z} = \frac{W(W-1)}{(W+Y)L} - \frac{\left[\frac{2(W(W-1)+2WY}{W+Y}\right]^2}{(2L)^2}. \tag{3.21}$$

Similarly, the contribution to the IP modularity when Y nodes are partitioned together in a single community is given by

$$\begin{aligned} M1_Y = & \frac{Y(Y-1)\left[\frac{1}{X+Y} + \frac{1}{Y+Z}\right]}{L} - \\ & \frac{\left[\frac{2XY}{X+Y} + 2Y(Y-1)\left[\frac{1}{X+Y} + \frac{1}{Y+Z}\right] + \frac{2YZ}{Y+Z}\right]^2}{(2L)^2}. \end{aligned} \tag{3.22}$$

Alternatively, if the Y nodes each form their own community the contribution to the IP modularity of Y singleton clusters is given by

$$M_{SC} = -Y\frac{\left[\frac{2(Y-1)}{X+Y} + \frac{2}{Y+Z} + \frac{2Z}{Y+Z} + \frac{2Y}{X+Y}\right]^2}{(2L)^2}.$$

Finally, the contribution to the IP modularity when all Y nodes form a single community with either the X or Z nodes, $M2_{Y,W \in X,Z}$, is given by

$$\begin{aligned} M2_{Y,W \in X,Z} = & \frac{Y(Y-1)\left[\frac{1}{X+Y} + \frac{1}{Y+Z}\right] + \frac{2*Y*W}{W+Y} + \frac{W(W-1)}{W+Y}}{L} - \\ & \frac{\left[\frac{2XY}{X+Y} + 2Y(Y-1)\left[\frac{1}{X+Y} + \frac{1}{Y+Z}\right] + \frac{2YZ}{Y+Z} + \frac{2W(W-1)}{W+Y} + \frac{2WY}{W+Y}\right]^2}{(2L)^2}. \end{aligned}$$

Comparing the above modularities there are conditions for which the modularity contribution of Y singleton clusters is larger than all Y nodes in a single community ($M_{SC} > M1_Y$). However, it is never the case that the maximum modularity will be found when either the Y nodes are in a community together or there are Y singleton clusters. Therefore, singleton clusters do not exist in the IP maximum modularity partition. The maximum modularity is always found when either community Y is found to be merged with community X or community Z. The modularity and conditions for each case is given by

$$M_{max} = \begin{cases} M1_X + M2_{Y,Z} & \text{if } Z < X \\ M1_Z + M2_{Y,X} & \text{if } X < Z. \end{cases} \tag{3.23}$$

Thus, for this example network, Y is always merged with whichever set of nodes is smaller X or Z. If $X = Z$ the modularity is equivalent by merging the Y nodes with either the X nodes or Z nodes. For the example in Fig. 3.3, when $X = 5$, $Z = 15$, and $Y = 14$, then Y is merged with community X giving the partition shown in Fig. 3.5. The modularity for this partition is $M_{max} = 0.116591$.

In conclusion, for the example network analyzed, a condition does not exist for which the partition giving the maximum IP modularity contains singleton clusters. This is due to two properties of IP modularity. The first is that although w_{ii} is defined to be zero, the corresponding element of the IP modularity matrix is not defined to be zero. Thus, singleton clusters are penalized in the IP modularity definition. Additionally, with the division of the team degree in the definition of w_{ij} the link weight connecting one of the Y nodes to an X node or Z node in the IP projection will be different unless $Z = Y$. This was not the case in the Guimera modularity as each (X,Y) and (Y,Z) pair of nodes shared one team and both link

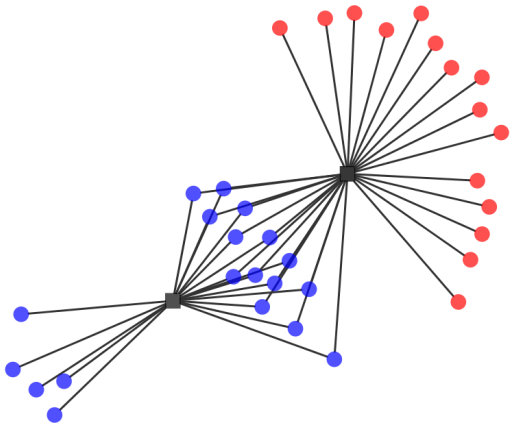


Figure 3.5: **The $\mathbf{Q^{IP}}$ maximum modularity partition.**The maximum modularity partition for the example network in Fig. 3.1 when $X = 5$, $Y = 14$ and $Z = 15$ when maximizing the Q^{IP} modularity. Singleton clusters do not exist and the X and Y nodes are partitioned into one community, shown in blue.

weights were therefore equal in the projection.

Thus, the IP modularity described above generalizes to the case when links in the bipartite network are weighted, preserves more information about the bipartite network than the Guimera modularity, and does not produce singleton clusters. In chapter 4, the ability for the IP modularity to recover the known structure of communities in both generated and real-world networks will be validated. Now the problem of finding the partition that maximizes the bipartite modularities described above is addressed.

3.2 Maximizing bipartite modularity

To detect communities in bipartite networks by maximizing the modularity, a variation of the leading eigenvector method [4, 151] to maximize modularity Q (see section 1.8.4) is now improved and adapted to the Barber, Guimera, and IP modularities discussed in the previous section. This method recursively divides the network according to the elements of the eigenvector corresponding to the largest positive eigenvalue of the modularity matrix. In the following discussion, the Barber, Guimera, and IP modularities will be refereed to as Q^B, Q^G, and Q^{IP}, respectively, when modifications of the algorithm are specific to a particular modularity method. To begin, consider a modularity matrix $\mathbf{B}$ with elements B_{ij} and modularity given by

$$Q = \sum_{ij} B_{ij}\delta_{C(i),C(j)}, \tag{3.24}$$

where $C(i)(C(j))$ is the community to which node $i(j)$ belongs.

3.2.1 The initial division

One begins by initially bisecting the network into two communities. The modularity equation can be rewritten by considering a vector $\mathbf{s}$ whose element $s_i = 1$ if node i is assigned to the first community or $s_i = -1$ if node i is assigned to the second. With $\delta_{C(i),C(j)} = \frac{(s_i s_j + 1)}{2}$, for the general modularity matrix with elements B_{ij}, the modularity is given by

$$Q = \frac{1}{2} \sum_{i,j} B_{ij} s_i s_j. \tag{3.25}$$

Writing $\mathbf{s}$ as a linear combination of the normalized eigenvectors $\mathbf{u}$ of $\mathbf{B}$, $\mathbf{s} = \sum_{i=1}^{N} a_i \mathbf{u_i}$ where a_i are constants and given by $a_i = \mathbf{u}_i^T \mathbf{s}$, the modularity, as previously derived, becomes

$$Q = \frac{1}{2} \sum_{i=1}^{N} a_i^2 \beta_i, \tag{3.26}$$

where β_i is the eigenvalue of $\mathbf{B}$ with eigenvector $\mathbf{u_i}$. From this equation, it is clear that maximizing Q is equivalent to choosing the quantities a_i^2 so as to place as much possible weight in the terms that correspond to the largest positive eigenvalues of $\mathbf{B}$. Then, the vector $\mathbf{s}$ giving the largest value of modularity would be one that is parallel to vector $\mathbf{u_1}$ corresponding to the largest positive eigenvalue, β_1, of $\mathbf{B}$. However, the elements of $\mathbf{s}$ are constrained to be either 0 or 1. Therefore, if the element u_i^1 of the eigenvector $\mathbf{u_1}$ is positive s_i is assigned a value of 1 and if the element u_i^1 is negative then $s_i = -1$. For $\mathbf{B}$ defined by Q^G and Q^{IP}, finding the eigenvector is straightforward and is found using the power method. However, there is a problem for the modularity matrix of Q^B.

For the modularity matrix of Q^B, there is an eigenvalue $-\lambda$ for every eigenvalue λ. As first discussed by Barber [176], the eigenvalue equation $\mathbf{B}\mathbf{x_i} = \lambda_i \mathbf{x_i}$ defined in

the Barber modularity of Eqn. 3.28 can be written as

$$\begin{bmatrix} \mathbf{O}_{p\times p} & \tilde{\mathbf{B}}_{p\times q} \\ \\ \tilde{\mathbf{B}}^T_{q\times p} & \mathbf{O}_{q\times q} \end{bmatrix} \begin{bmatrix} \mathbf{u_i} \\ \\ \mathbf{v_i} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{B}}\mathbf{v_i} \\ \\ \tilde{\mathbf{B}}^T\mathbf{u_i} \end{bmatrix} = \lambda_i \begin{bmatrix} \mathbf{u_i} \\ \\ \mathbf{v_i} \end{bmatrix}, \tag{3.27}$$

where $\mathbf{u_i}$ is a $p \times 1$ vector and $\mathbf{v_i}$ is a $q \times 1$ vector. However, one can also create a vector so that

$$\begin{bmatrix} \mathbf{O}_{p\times p} & \tilde{\mathbf{B}}_{p\times q} \\ \\ \tilde{\mathbf{B}}^T_{q\times p} & \mathbf{O}_{q\times q} \end{bmatrix} \begin{bmatrix} \mathbf{u_i} \\ \\ -\mathbf{v_i} \end{bmatrix} = \begin{bmatrix} -\tilde{\mathbf{B}}\mathbf{v_i} \\ \\ \tilde{\mathbf{B}}^T\mathbf{u_i} \end{bmatrix} = -\lambda_i \begin{bmatrix} \mathbf{u_i} \\ \\ -\mathbf{v_i} \end{bmatrix}, \tag{3.28}$$

which shows for each eigenvalue λ_i there exists an eigenvalue $-\lambda_i$. Additionally, $\tilde{\mathbf{B}}$ is generally not a square matrix and is not symmetric. Therefore, it is possible for the left and right eigenvalues to be complex. To address these issues, we consider the singular values of $\tilde{\mathbf{B}}$. The singular values of an $m \times n$ matrix $\mathbf{A}$ are the square roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$, which are all real and non-negative [200]. The algorithm begins by dividing the network into two communities by finding the left and right singular vectors of $\tilde{\mathbf{B}}$. If $p < q$, then we find the singular vector $\mathbf{u_1}$ corresponding to the largest eigenvalue σ_1 of $\tilde{\mathbf{B}}\tilde{\mathbf{B}}^T$.The eigenvector $\mathbf{u_1}$ is found using the power method and has p elements corresponding to the the first p elements of $\mathbf{s}$. The elements of vector $\mathbf{s}$ are assigned a value of 1 or -1 according to the sign of the corresponding element in $\mathbf{u_1}$. The next q elements of $\mathbf{s}$ are assigned using the singular vector $\mathbf{v_1}$ given by

$$\mathbf{v_1} = \frac{\mathbf{1}}{\sigma_\mathbf{1}}\tilde{\mathbf{B}}^\mathbf{T}\mathbf{u_1}. \tag{3.29}$$

If $q < p$, elements $p+1$ to $p+q$ of $\mathbf{s}$ are assigned first by finding the singular vector $\mathbf{v_1}$ corresponding to the largest eigenvalue σ_1 of $\tilde{\mathbf{B}}^T\tilde{\mathbf{B}}$, then $\mathbf{u_1}$ is given by, $\mathbf{u_1} = \frac{\mathbf{1}}{\sigma_\mathbf{1}}\tilde{\mathbf{B}}\mathbf{v_1}$.

The modularity of this division can often be improved using a variant of the Kernighan-Lin tuning algorithm [149]. The algorithm begins by moving each node from its current community into the other to see if there is an improvement in modularity. The change in modularity, δQ_k, from moving node k from its current community to the other is given by

$$\delta Q_k = -4 s_i \sum_{i \neq k} B_{ki} s_i. \tag{3.30}$$

Then, the Kernighan-Lin (K-L) tuning step is as follows:

1. Initialize Q_{total}, defined as the improvement using the K-L tuning step, to zero.
2. Set the initial configuration of vector $\boldsymbol{s}$ to $\boldsymbol{s_{initial}}$, where the element $s_i = 1$ if node i is assigned to the first community or $s_i = -1$ is assigned to the second community.
3. Consider moving each node k from its current community to the opposite community by computing, δQ using Eqn. 3.30.
4. Pick the node k giving the largest δQ, even if δQ is negative, and call this node K. If there is more than one node with the largest δQ choose one of these randomly.
5. Add δQ to Q_{total}.
6. Set the element $s_K = -s_K$ for node K.
7. If Q_{total} is the largest or equal to the largest found so far store $\boldsymbol{s}$ and its corresponding Q_{total}.
8. Repeat steps 3. through 7. only considering nodes which have not been moved until all nodes have been moved once.

9. If the largest intermediate Q_{total} is positive then repeat from step 1 using the corresponding $\boldsymbol{s}$ as the initial guess, or if there is more than one intermediate $\boldsymbol{s}$ stored with this Q_{total} randomly choose between them. If Q_{total} is negative end the algorithm returning the initial configuration vector $\boldsymbol{s_{initial}}$.

If the Kernighan-Lin algorithm fails to find an assignment that produces an increase in modularity when splitting the community, the algorithm stops.

3.2.2 Subsequent divisions

To continue dividing a community, C_k, after the initial division one must choose $\mathbf{s}$ to maximize

$$\delta Q_{C_k} = \frac{1}{2}\left(\sum_{i,j\in C_k} B_{ij}s_i s_j - \sum_{i,j\in C_k} B_{ij}\right). \tag{3.31}$$

For the bi-clustering modularity matrix of Q^B, again due to the property that there is an eigenvalue $-\lambda$ for every eigenvalue λ, one chooses $\mathbf{s}$ to maximize the first term, $\sum_{i,j\in C_k} B_{ij}s_i s_j$, by considering the singular values of $\tilde{\mathbf{B}}^k = \{\tilde{B_{ij}}|i,j \in C_k\}$. Then, one must check that the division truly gives a positive increase in modularity by subtracting the correction term, $\sum_{i,j\in C_k} B_{ij}$. For the modularity matrix of Q^G and Q^{IP}, δQ_{C_k} may be simplified to give

$$\delta Q_{C_k} = \frac{1}{2}\sum_{i,j\in C_k} B^k_{ij}s_i s_j, \tag{3.32}$$

where, $B^k_{ij} = B_{ij} - \delta_{ij}\sum_{l\in C_k} B_{il}$. Again, each division is improved by the Kernighan-Lin step and this process continues iteratively until the modularity can no longer be increased. However, as was shown by Sun *et al.* [9] in the leading eigenvalue algorithm for maximizing modularity Q in unipartite networks, a bias is introduced when continually bisecting the network in this way [9].

3.2.3 Final tuning

After the initial bisection of the network, each community forms two disjoint subsets. Each subset of nodes will only be divided in the subsequent iterations of the algorithm so that once two nodes are separated into different communities they are never again found together in the same community. Dividing the network in this way creates hard partitions and a bias in the algorithm.

A manifestation of these hard partitions can be seen when considering the distribution of the community sizes in an ensemble of bipartite Erdös-Rényi type networks. The distribution of the community sizes for an ensemble of bipartite Erdös-Rényi type network partitions is shown in Fig. 3.6. Each network contains 200 actors and 200 teams with the probability of an actor-team connection of $p = 0.02$. The network was bi-clustered using the modularity of Barber *et al.*, Q^B. The black curve (circles) is the community size distribution for partitions in which the modularity was found by continually bisecting the network and applying the Kernighan-Lin tuning step. There are two peaks corresponding to sizes of $N/2^3$ and $N/2^4$ where N is the size of the network ($N = 400$). The bias is manifested in the fact that the peaks occur at community sizes equal to N divided by a power of two.

Fig. 3.7 and Fig. 3.8 give the distribution of the community sizes for the same ensemble using the modularity of Guimera, Q^G, and the IP modularity , Q^{IP}, respectively. For each network the actor set was partitioned. For both figures, the black curve (circles) is the community size distribution for partitions in which the modularity was found by continually bisecting the network and applying the Kernighan-Lin tuning step. There are two peaks corresponding to sizes of $N/2^2$ and $N/2^3$ where N

is the size of the actor set ($N = 200$). Therefore, for all definitions of bipartite modularity the bisection bias is manifested in the fact that the peaks occur at community sizes equal to N divided by a power of two.

To amend the bias, a modified Kernighan-Lin step, defined as "final tuning", is applied that was first introduced by Sun *et al.* [9] for the unipartite modularity of Newman and Girvan. Here, the algorithm is described when considering any modularity matrix $\mathbf{B}$. After the initial division of the network with Kernighan-Lin tuning, consider the change in modularity from moving each node from its current community to all existing communities or forming its own community. In the case when node l begins in community C_x and is moved to community C_y, then the change in modularity is given by

$$\delta Q_l = \sum_{i \in C_y} B_{il} + \sum_{j \in C_y} B_{lj} - \sum_{i \in C_x} B_{il} - \sum_{j \in C_x} B_{lj} + 2B_{ll}. \tag{3.33}$$

This equation is applicable when considering the modularity matrix $\tilde{\mathbf{B}}$ but can be simplified when the modularity matrix is symmetric as in Q^G and Q^{IP}. For a symmetric matrix

$$\delta Q_l = 2 \left(\sum_{i \in C_y} B_{il} - \sum_{i \in C_x} B_{il} + B_{ll} \right). \tag{3.34}$$

Then, the final tuning step is as follows:

1. Initialize Q_{total}, defined as the improvement using the final tuning step, to zero.

2. Set the initial partition of the network vector $\boldsymbol{P(G)}$ to $\boldsymbol{P(G)_{initial}}$, where the element P_i is equal to the current community assignment of node i.

3. Consider moving each node l from its current community, to all other existing communities, or into a community of its own by computing δQ using Eqn. 3.33.

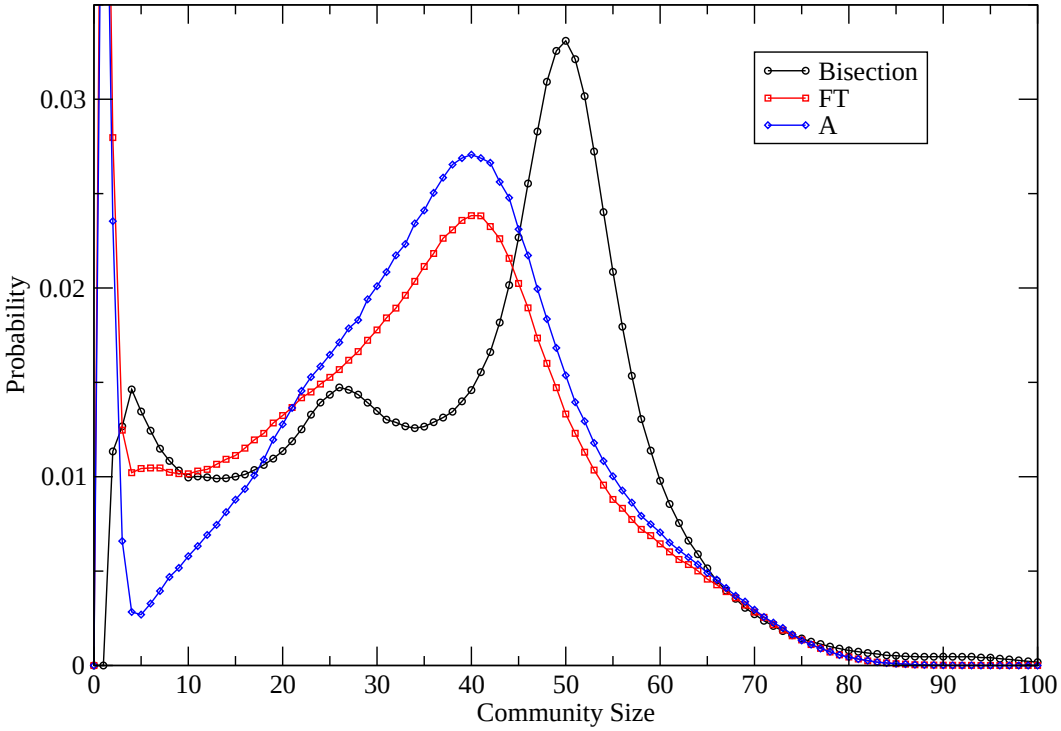


Figure 3.6: **Community size distribution found by maximizing Barber's modularity for an ensemble of bipartite Erdös-Rényi type networks with 200 nodes of each type and probability of connection $\mathbf{p = 0.02}$.** Each network was partitioned by maximizing Q^B using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

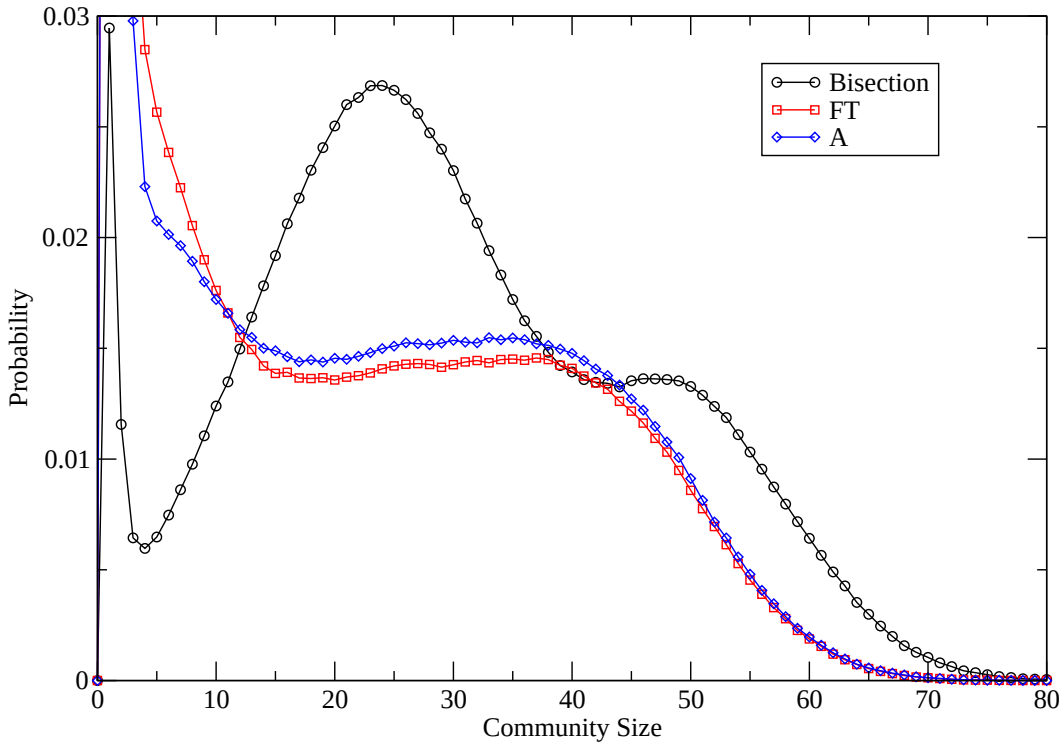


Figure 3.7: **Community size distribution found by maximizing Guimera's modularity for an ensemble of bipartite Erdös-Rényi type networks, 200 of the first type and 200 of the second, with probability of connection $\mathbf{p = 0.02}$.** For each network the actors were partitioned by maximizing Q^G using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

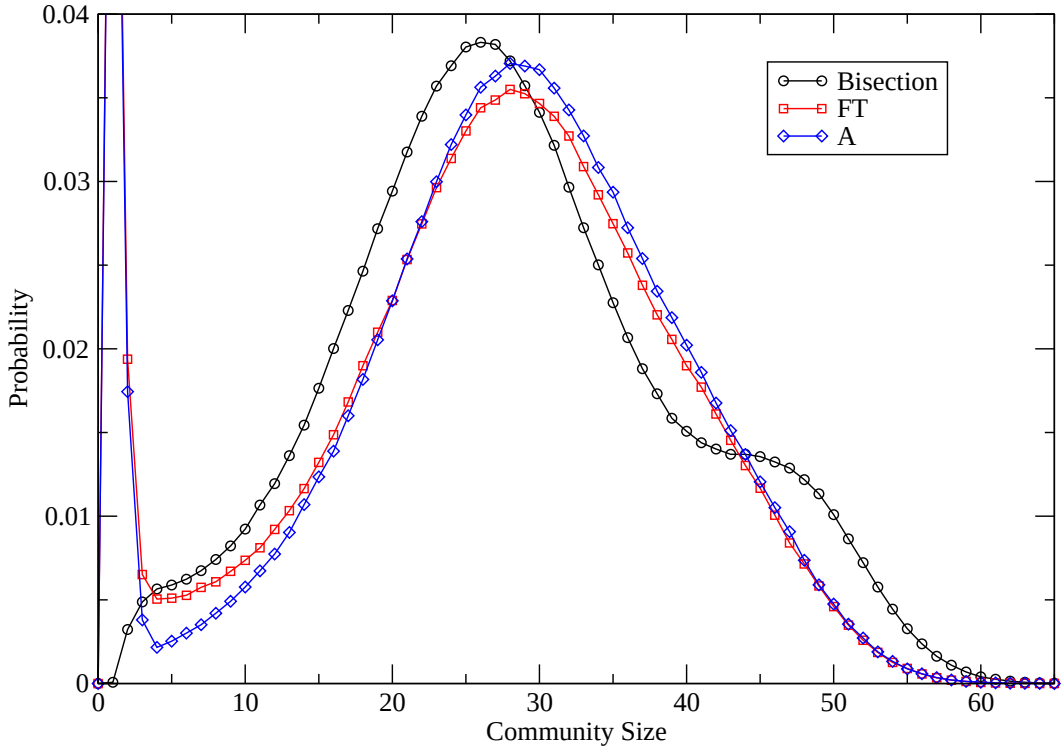


Figure 3.8: **Community size distribution found by maximizing the IP modularity for an ensemble of bipartite Erdös-Rényi type networks, 200 of the first type and 200 of the second, with probability of connection p = 0.02.** For each network, actors were partitioned by maximizing Q^{IP} using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

4. Pick the move giving the largest δQ. If there is more then one move with the largest δQ, choose one of these randomly.

5. Add δQ to Q_{total}.

6. Fix the community assignment for the node moved by updating $\boldsymbol{P(G)}$.

7. If Q_{total} is the largest or equal to the largest found so far store $\boldsymbol{P(G)}$ and Q_{total}.

8. Repeat steps 3 through 7 only considering nodes which have not been moved, until all nodes have been moved once.

9. If the largest intermediate Q_{total} is positive then repeat from step 1 using the corresponding $\boldsymbol{P(G)}$ as the initial guess or, if there is more than one intermediate $\boldsymbol{P(G)}$ stored with this Q_{total}, randomly choose between them. If Q_{total} is negative, end the algorithm returning the initial partition $\boldsymbol{P(G)}$.

To demonstrate the usefulness of final tuning with an analytic example, consider the simple bipartite network of Fig. 3.9 in which there are 9 actors, 5 teams and 15 links. If this network is bi-clustered by bisecting, then the first division produces the partition of Fig. 3.9a. The next and final step when bisecting is to divide the community of 5 actors and 3 teams resulting in the partition of Fig. 3.9b. This partition has a modularity value of $Q^B = 114/225$. However, this is not the partition with the maximum modularity. Applying final tuning produces the partition of Fig. 3.9c in which an actor from the first community has been moved to the middle community. The modularity value for this partition is $Q^B = 24/45$.

Similarly, final tuning improves the results when projecting a bipartite network onto a unipartite multigraph using the Q^G modularity. Consider the bipartite network of Fig. 3.10a in which there are 9 actors, 5 teams and 17 links. The unipartite

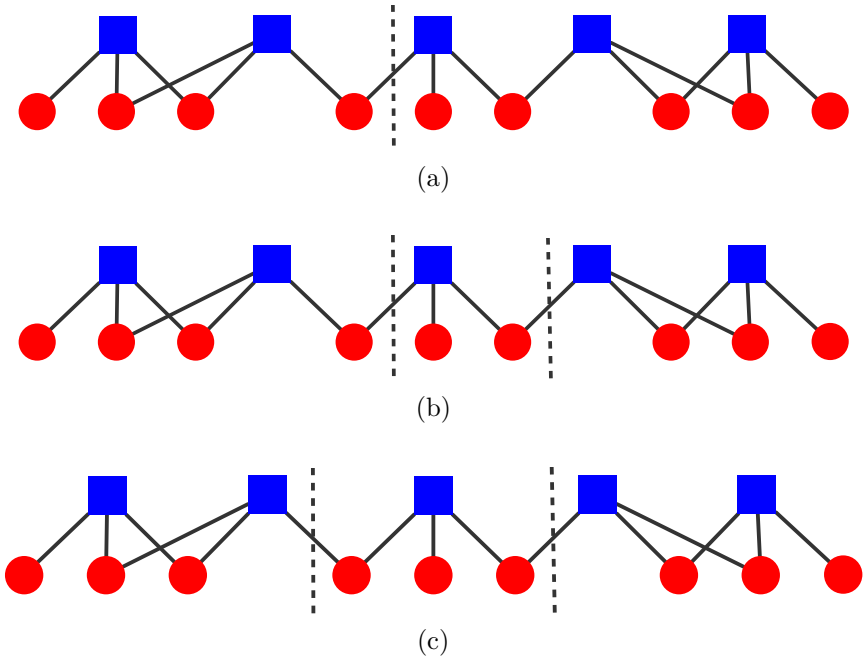


Figure 3.9: **Biclustering partitions of a simple bipartite network** (a) after one bisection, (b) giving the final result with bisectioning and (c) giving the optimal partitioning that maximizes modularity Q^B. The optimal partitioning is found when bisectioning is combined with final tuning.

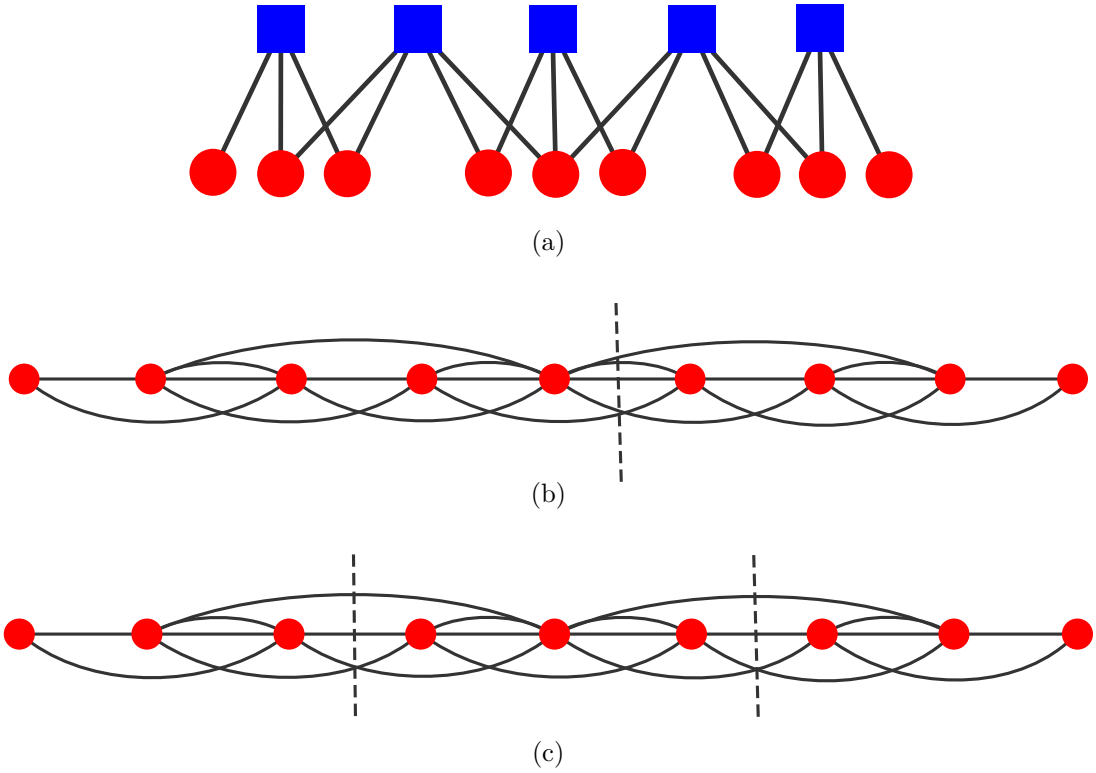
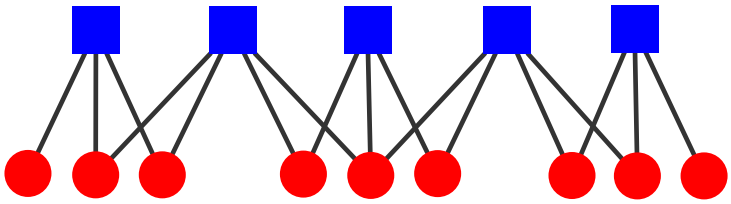


Figure 3.10: **The maximization of modularity $\mathbf{Q^G}$** where (a) is the initial bipartite network, (b) is the partitioned unipartite projection of the red actor set using bisectioning, and (c) is the optimal partitioning of the unipartite projection of the red actor set using bisectioning combined with final tuning.

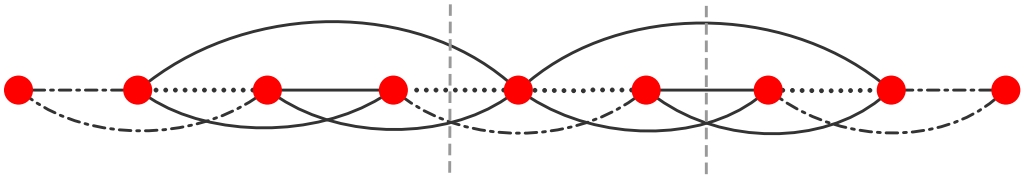
multigraph projection of the actor set shown in Fig. 3.10b and 3.10c using Q^G. Each link between two actors represents a team through which they are connected. If this network is partitioned by bisecting then the resulting partition is given in Fig. 3.10b. The partition has a modularity value of $Q^G = 2230/6069$ and further dividing either community will result in a lower modularity. However, applying final tuning results in the partition of Fig. 3.10c. This partition has three communities and a modularity value of $Q^G = 2413/6069$.

Final tuning improves the results when projecting the same bipartite network, used in the previous section, onto a unipartite multigraph using the modularity Q^{IP}. Consider the bipartite network of Fig. 3.11a in which there are 9 actors, 5 teams and 17 links. The unipartite multigraph projection of the actor set shown in Fig. 3.11b and 3.11c using Q^{IP}. Each link between two actors is weighted using Eqn. 3.17. If this network is partitioned by bisecting then the resulting partition is given in Fig. 3.11b. The partition has a modularity value of $Q^{IP} = 2645/10368$ and further dividing either community will result in a lower modularity.Applying final tuning produces the partition of Fig. 3.11c in which an actor from the first community has been moved to the middle community. The modularity value for this partition is $Q^{IP} = 31/96$.

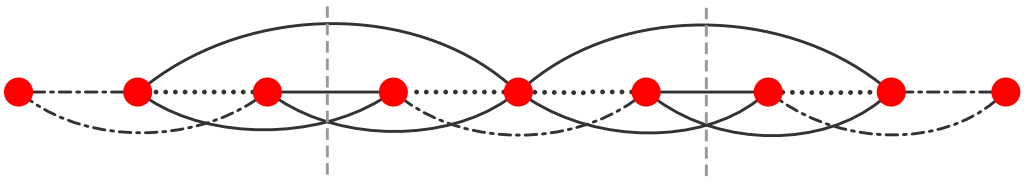
Consider again the ensemble of bipartite Erdös-Rényi type networks. The distribution of community sizes when final tuning step is added to the algorithm is given by the red curves for Q^B, Q^G, and Q^{IP}, in Fig. 3.6, Fig. 3.7, and Fig. 3.8, respectively. For all modularity definitions, adding the final tuning step results in a single peak indicating the bias has now been removed.



(a)



(b)



(c)

Figure 3.11: **The maximization of modularity $\mathbf{Q^{IP}}$** where (a) is the initial bipartite network, (b) is the partitioned unipartite projection of the red actor set using bisectioning, and (c) is the optimal partitioning of the unipartite projection of the red actor set using bisectioning combined with final tuning. In the unipartite projection solid links represent weight $\frac{1}{4}$, dotted links represent weight $\frac{7}{12}$, and dash-dotted links represent weight $\frac{1}{3}$.

3.2.4 An agglomeration step

In both the Kernighan-Lin and Final tuning algorithms the ability to find the maximum modularity partition is improved by performing local searches in modularity space. That is, both of the previous tuning algorithms move individual nodes to improve the current partition. In this section, a tuning step is introduced to the algorithm that performs a global search in the modularity space by considering moving entire communities of nodes in the current partition. A global search may be able to find partitions in the modularity space that a local search would be unable to. For example, if two communities exist in the current iteration of the algorithm that would give a larger modularity score when merged, a local search may never find this partition as the modularity penalty of moving, individually, each node from its current community to the other, would be too large.

To find the change in modularity Q when community C_x and community C_y are merged, begin with the definition of a general modularity function Q written as

$$Q = \sum_{\substack{k=1 \\ k \neq x,y}}^{N_c} \sum_{i,j \in C_k} B_{ij} + \sum_{i,j \in C_x} B_{ij} + \sum_{i,j \in C_y} B_{ij}, \tag{3.35}$$

where N_c is the total number of communities. If community C_x and community C_y are merged to form community C_z the new modularity is written simply as

$$Q' = \sum_{\substack{k=1 \\ k \neq z}}^{N_c'} \sum_{i,j \in C_k} B_{ij} + \sum_{i,j \in C_z} B_{ij}, \tag{3.36}$$

where $N_c' = N_c - 1$. In terms of C_x and C_y the contribution to the modularity of community C_z can be written as

$$\sum_{i,j \in C_z} B_{ij} = \sum_{i,j \in C_x} B_{ij} + \sum_{i,j \in C_y} B_{ij} + 2 \sum_{i \in C_x} \sum_{j \in C_y} B_{ij}. \tag{3.37}$$

Therefore, the change in modularity $\delta Q = Q' - Q$ from merging community C_x and C_y is given by

$$\delta Q = 2 \sum_{i \in C_x} \sum_{j \in C_y} B_{ij}. \tag{3.38}$$

.

For each round of bisection with Kernighan-Lin after final tuning is performed the following agglomerative tuning step is performed:

1. Initialize Q_{total}, defined as the improvement using the final tuning step, to zero.
2. Set the initial partition of the network vector $\boldsymbol{P(G)}$ to $\boldsymbol{P(G)_{initial}}$, where the element P_i is equal to the current community assignment of node i.
3. Compute the change in modularity δQ for the agglomeration of all possible pairs of communities using Eqn 3.38.
4. Merge the pair of communities giving the largest δQ and update the merger in $\boldsymbol{P(G)}$.
5. Add δQ to Q_{total}.
6. Record Q_{total} up to this point and the corresponding $\boldsymbol{P(G)}$.
7. Repeat from step 3 until you are left with only one community containing all N nodes.
8. Find the configuration with the largest change in modularity.
9. If the largest intermediate Q_{total} is non-negative, assign the communities to the corresponding intermediate configuration $\boldsymbol{P(G)}$. If there are multiple configurations with the same positive Q_{total}, choose the configuration with the least

number of communities. If the largest Q_{total} is negative assign the communities to the initial partition $\boldsymbol{P(G)_{initial}}$.

Note that when considering multiple configurations with the same change in modularity, rather than choose a random configuration, the configuration with the least number of communities is chosen. This choice serves as a "correction" to the current partition of nodes that the bisection with Kernighan-Lin with final tuning step has missed.

3.2.5 The complete algorithm

To incorporate all of the tuning steps described above to maximize bipartite modularity:

1. Initialize the network with all N nodes partitioned into one community of size N. Note this step assumes that for the Q^G and Q^{IP} modularities the network of N nodes is the projected network.

2. Attempt to divide each of the existing communities using the leading eigenvalue method as described in Sections 3.2.1 and 3.2.2 with the Kernighan-Lin algorithm described in Section 3.2.1.

3. Perform the final tuning algorithm as described in Section 3.2.3.

4. Peform the agglomeration algorithm as described in Section 3.2.4.

5. Repeat steps 2-4 until no further improvement of the modularity is achieved.

3.3 Numerical results

3.3.0.1 Biparite Erdös-Rényi type network

When partitioning a bipartite Erdös-Rényi type network using bipartite modularity, final tuning removes the bias created by hard partitioning when dividing the network, as shown in Fig. 3.6, Fig. 3.7, and Fig. 3.8. Fig. 3.12 shows the corresponding distributions of the modularity for the same set of ensemble of networks when partitioning a network using Q^B. The average of the bisectioning only peak, shown in black is 0.516618 ± 0.000018, with standard deviation 0.012853 ± 0.000011. When applying final tuning the average of the bisectioning with FT peak, shown in red is 0.528079 $\pm$ 0.000018 with standard deviations 0.012686 ± 0.000011. The distribution curve where final tuning is applied is shifted to the right indicating an improvement in modularity. Similarly, when adding the agglomeration tuning step, the average of the corresponding peak, shown in blue is 0.529426 ± 0.000018 with standard deviation 0.012832 ± 0.000011 again indicating a further improvement.

For the Guimera modularity, again final tuning improves the average modularity as shown in Fig. 3.13. The average of the bisectioning only peak, shown in black and the bisectioning with final tuning peak, shown in red is 0.320294 ± 0.000023 and 0.332522 ± 0.000023, respectively with standard deviations 0.015933 ± 0.000014 and 0.015974 ± 0.000014, respectively. When applying agglomeration the average marginally improves shown by the blue peak is 0.332896 ± 0.000023 with standard deviation 0.016009 ± 0.000014.

Finally, for the IP modularity, both tuning steps again improve the average modularity as shown in Fig. 3.14. The average of the bisectioning only peak, shown

in black and the bisectioning with final tuning peak, shown in red is 0.301142 ± 0.000019 and 0.312340 ± 0.000019, respectively with standard deviations 0.013569 ± 0.000012 and 0.013635 ± 0.000012, respectively. When applying agglomeration the average modularity marginally improves. The average of the agglomeration peak, shown in blue is 0.312791 ± 0.000019 with standard deviation 0.013698 ± 0.000012. Interestingly, the agglomeration tuning step has a greater improvement on the average modularity for Q^B. This may be due to size of the projected networks of Q^G and Q^{IP} being half the size of the biclustered networks of Q^B. With more nodes, and therefore a larger number of possible partitions in the network, the agglomeration step is expected to have a greater impact on modularity maximization and this is shown to be true in the next section.

3.3.0.2 Real-world networks

Thus far, it has been shown that for an ensemble of bipartite Erdös-Rényi type networks, final tuning and agglomeration improves the modularity values found for all definitions of bipartite modularity. Applying final tuning and agglomeration to a number of real-world networks also improves the modularity compared to bisecting without final tuning as shown in Table 3.1, Table 3.2, and Table 3.3. The networks are a social network of southern women in the 1930's [165], the largest component of the Scotland corporate interlock network [247], and the largest component of a network of the administrative elite in The Netherlands [248]. Due to stochastic elements in the community detection algorithm, such as when a move is chosen randomly among the multiple moves that result in the same increase in modularity, hundreds of analyses for each network were run and the partition with maximum modularity is reported. There are, however, cases in which adding final tuning and agglomeration

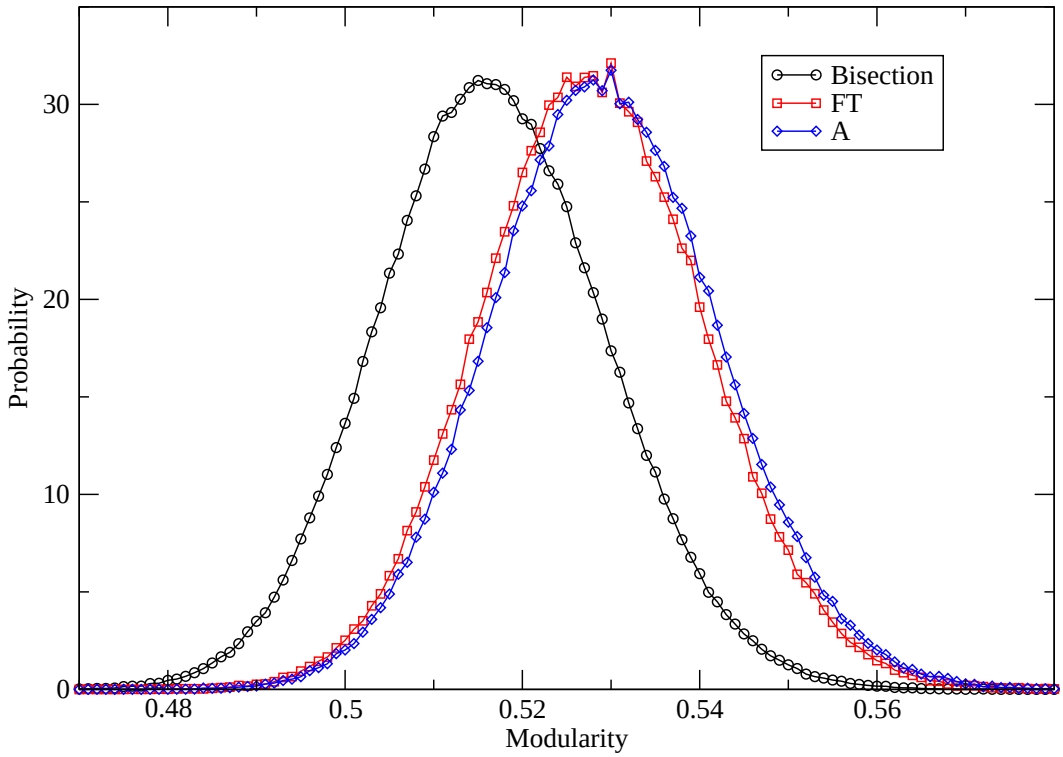


Figure 3.12: **Q^B modularity distribution for an ensemble of bipartite Erdös-Rényi type networks.** There are 200 nodes of each type and probability of connection $p = 0.02$. Each network was partitioned by maximizing Q^B using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

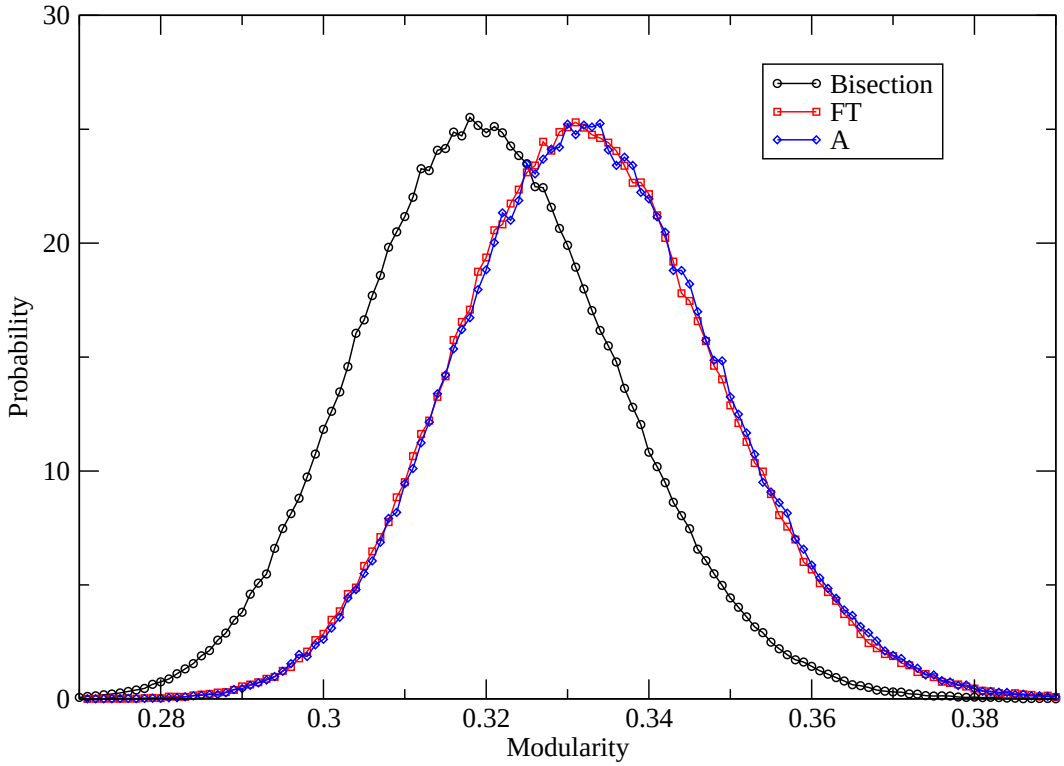


Figure 3.13: **Q^G modularity distribution for an ensemble of bipartite Erdös-Rényi type networks.** There are 200 nodes of each type with probability of connection $p = 0.02$. For each network the 200 node set of the first type was partitioned by maximizing Q^G using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

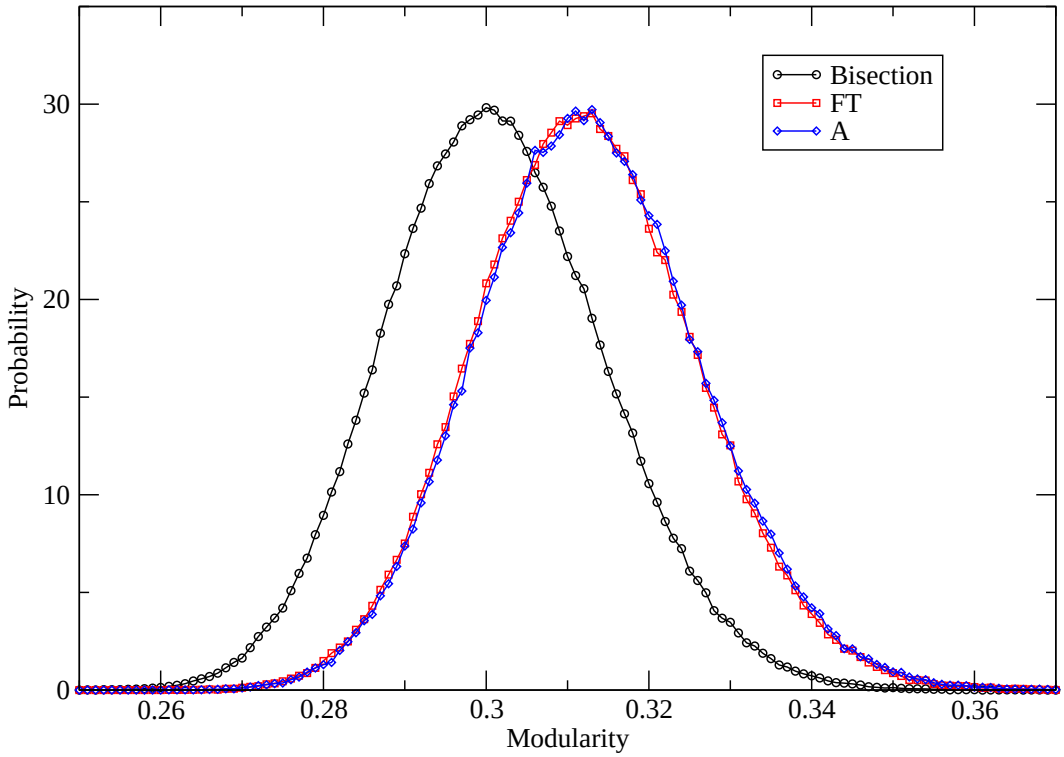


Figure 3.14: **Q^{IP} modularity distribution for an ensemble of bipartite Erdös-Rényi type networks.** There are 200 nodes of each type with probability of connection $p = 0.02$. For each network the 200 node set of the first type was partitioned by maximizing Q^{IP} using the bisectioning algorithm, (black circles), the bisectioning algorithm combined with final tuning, (red squares), and the bisectioning algorithm combined with final tuning and agglomeration (blue triangles).

Q^B					
Network	Actors	Teams	wo/ft	ft	a
Southern	18	14	0.340866	0.345158	0.345158
Scotland	131	86	0.703660	0.709366	0.709432
Dutch	1794	585	0.776806	0.783532	0.801707

Table 3.1: Comparison of the maximum modularity Q^B found for real world bipartite networks using bisectioning without final tuning (wo/ft) , bisectioning with final tuning (ft), and bisectioning with final tuning and agglomeration (a).

does not improve the maximum modularity found when combined with bisectioning. For example in Table 3.2, for the Southern women's network, adding final tuning does not improves the maximum modularity found. Similarly in Table 3.1, adding the agglomeration step does not improve the maximum modularity found when using final tuning. For these cases, the partitioning with maximum modularity is found without the addition of final tuning and agglomeration. However, as the example networks grow in size the addition of final tuning and agglomeration is expected to improve the maximum modularity found. This is also the case for unipartite networks as shown in Table 3.4. Maximum modularity values for networks beginning with the network of size 453 are improved when adding the final tuning step as shown in the last column.

In this section, a leading eigenvalue algorithm to find the partition that maximizes bipartite modularity has been developed. Regardless of which bipartite modularity is maximized, finding the network partition that maximizes the modularity by continually bisecting the network creates a bias in the algorithm due to hard partitioning. Applying a modified Kernighan-Lin step called final-tuning amends the bias and improves the modularity values found. Additionally, an agglomeration step in the algorithm, that considers merging pairs of communities, further improves the results by performing a global search of the modularity space.

Q^G				
Network	Actors	wo/ft	ft	a
Southern	18	0.216875	0.216875	0.216875
Scotland	131	0.611385	0.615863	0.616002
Dutch	1794	0.804407	0.807471	0.819560
Network	Teams	wo/ft	ft	a
Southern	14	0.278773	0.278773	0.278773
Scotland	86	0.612727	0.616884	0.618095
Dutch	585	0.561673	0.568781	0.571242

Table 3.2: Comparison of the maximum modularity Q^G for each type of nodes in the bipartite network found using bisectioning without final tuning (wo/ft) , bisectioning with final tuning (ft), and bisectioning with final tuning and agglomeration (a).

Q^{IP}				
Network	Actors	wo/ft	ft	a
Southern	18	0.2440909	0.2440909	0.2440909
Scotland	131	0.5597220	0.5603517	0.5603517
Dutch	1794	0.7037658	0.7074518	0.7158929
Network	Teams	wo/ft	ft	a
Southern	14	0.168122	0.168122	0.168122
Scotland	86	0.557789	0.557789	0.557789
Dutch	585	0.524446	0.536349	0.540076

Table 3.3: Comparison of the maximum modularity Q^{IP} for each type of nodes in the bipartite network found using bisectioning without final tuning (wo/ft) , bisectioning with final tuning (ft), and bisectioning with final tuning and agglomeration (a).

	Q		
Network	Size	ft*	a
Karate	34	0.420	0.419790
Jazz musicians	198	0.445	0.445144
Metabolic	453	0.452	0.452647
E-mail	1133	0.580	0.582368
Key signing	10680	0.867	0.883528
Physicists	27519	0.737	0.746743

Table 3.4: Comparison of the maximum modularity Q found for real world unipartite networks using bisectioning with final tuning (ft) and bisectioning with final tuning and agglomeration (a). *Values as reported by Sun *et al.* [9].

3.4 Characterizing effect-size and significance of the results

As discussed in Section 1.7.3, Reichardt and Bornhold [182, 61, 249] derived an approximation for the expected value of modularity Q, for an ensemble of Erdös-Rényi graphs with N nodes and link probability p, given by the equation

$$\langle Q \rangle = 0.97\sqrt{\frac{1-p}{Np}}. \tag{3.39}$$

This equation was derived with the assumptions that $N \gg 1$ and $p \sim 1$. In other words, this equation is expected to hold for large dense networks. However, in many real-world systems, networks are typically sparse [30], and often their size is only few tens of nodes [134, 250, 251]. Therefore, to ensure the applicability of Eqn 3.39, it is necessary to find appropriate scaling corrections. Finding such corrections analytically is a difficult problem, thus, here we find a suitable correction numerically.

For small system sizes, it has been found that values of the modularity measured in simulations are lower than those predicted by Eqn 3.39 [252]. For larger systems, the equation is effectively in agreement with simulations, expect for values of p in the vicinity of the giant component transition [252]. This suggests the correction we need is twofold, consisting of a multiplicative piece to scale down the prediction for small systems, and a second additive piece to account for the case of sparse networks. Thus, an *ansatz* for the corrected form is

$$\langle Q \rangle \to C_1 \cdot 0.97\sqrt{\frac{1-p}{Np}} + C_2. \tag{3.40}$$

Simulation results seem to quickly approach the prediction of Eqn. 3.39 with increasing system size [252]. Therefore, assume that C_1 is of the form

$$C_1 = 1 - \lambda e^{-\frac{N}{N_o}}.$$

Recent work [252] has shown empirically that fitting these two parameters with the high-p tail of the results yields

$$\begin{aligned} \lambda &= \frac{7}{5} \\ N_o &= 50. \end{aligned}$$

Therefore, the multiplicative correction is

$$C_1 = 1 - \frac{7}{5} e^{-\frac{N}{50}}. \tag{3.41}$$

For the additive piece of the correction there is a clear dependence on p and N, we start by assuming the general form

$$C_2 = C_o p^{\alpha} (1-p)^{\beta} N^{\gamma}, \tag{3.42}$$

where the exponents α, β, and γ may depend on N. Recent work [252] has shown empirically that fitting these parameters yields

$$\begin{aligned} C_o &= 1 \\ \alpha &= -\frac{1}{6} \log\left(\frac{2}{5}N\right) \\ \beta &= \frac{5}{4} \\ \gamma &= -\frac{6}{5} + \frac{13}{15} e^{-\frac{N}{100}}. \end{aligned}$$

Then, to obtain the corrected expression for the expected maximum modularity the previous parameters are substituted into Eqn. 3.42, and using Eqn. 3.41 with

Eqn. 3.40:

$$\langle Q \rangle = \left(1 - \frac{7}{5} e^{-\frac{N}{50}}\right) 0.97 \sqrt{\frac{1-p}{Np}} + p^{-\frac{1}{6}\log\left(\frac{2}{5}N\right)} (1-p)^{\frac{5}{4}} N^{-\frac{6}{5}+\frac{13}{15}e^{-\frac{N}{100}}}. \tag{3.43}$$

To compute the Z-score of a given modularity score for a particular network we need to be able to express the variance of the modularity in the null model choice. To do so, we first use Eqn. 3.39 to find the expected form of the variance, using propagation of uncertainties. Equation 3.39 was originally derived assuming the number of nodes N and edges M are fixed, rather than N and p. Therefore, in finding an equation for the variance of $\langle Q \rangle$, p cannot be considered a constant. Then,

$$\sigma^2_{\langle Q \rangle} = (\partial_p \langle Q \rangle)^2 \sigma_p^2. \tag{3.44}$$

With M fixed, one can write $p = \frac{2M}{N^2}$, hence

$$\sigma_p^2 = (\partial_M p)^2 \sigma_M^2 = \frac{4}{N^4} \sigma_M^2. \tag{3.45}$$

As M is binomially distributed, its variance is

$$\sigma_M^2 = \frac{N^2}{2} p(1-p). \tag{3.46}$$

Substituting Eqn. 3.46 into Eqn. 3.45 yields

$$\sigma_p^2 = \frac{2}{N} p(1-p). \tag{3.47}$$

Finally, substituting Eqn. 3.47 into Eqn. 3.44 one obtains

$$\sigma^2_{\langle Q \rangle} = \frac{0.97^2}{2} \frac{1}{N^3 p^2}. \tag{3.48}$$

Once more, numerical simulations indicate that the actual variance deviates from the theoretical prediction [252]. Thus, also in this case we need to find a correction.

The deviation of the measured variances from those predicted by means of Eqn. 3.48 rapidly increases with the size of the network, apparently converging towards a constant. Therefore, we postulate that the correction C' to Eqn. 3.48 is multiplicative and has the form

$$C' = C'_o - e^{-\epsilon(N-N_o)}. \tag{3.49}$$

A fit of these parameters gives $C'_o = 2$, $\epsilon = \frac{1}{30}$ and $N_o = 10$. Thus, the final expression for the variance for an ensemble of Erdös-Rényi graphs is

$$\sigma^2_{\langle Q \rangle} = \left(15 - e^{-(N-10)/30}\right) \frac{0.97^2}{2} \frac{1}{N^3 p^2}. \tag{3.50}$$

To determine p for a particular network of size N one simply computes p using

$$p = \frac{2L}{N(N-1)}, \tag{3.51}$$

where L is the number of links in the network. With Eqn. 3.51, Eqn. 3.43, and Eqn. 3.50 the Z-score of a given modularity Q score for a particular network can now be computed.

Next, we will modify the previous results to compute the Z-score of a given modularity Q^{IP} score for a particular bipartite network containing N_A actor nodes, N_T team nodes, and L' links. For this particular bipartite network with L' links an equivalent bipartite Erdos-Reyni type network will be connected with probability p' given by

$$p' = \frac{L'}{N_A N_T}. \tag{3.52}$$

To detect communities using the Q^{IP} modularity, the N_A actor nodes are then projected onto a weighted unipartite network using the weight between an actor i and actor j defined in Eqn. 3.17. As this projected network is now unipartite, we can

compute the equivalent Erdös-Rényi p value in this projected network of N_A nodes by computing the average weight $\langle w_{ij} \rangle$ between actor i and actor j. The average degree of team k,$\langle m_k \rangle = p' N_A$ and therefore

$$\langle w_{ij} \rangle = \sum_k \frac{2(p')^2}{p' N_A} = \frac{2p' N_T}{N_A}. \tag{3.53}$$

Therefore, the average number of links L or, equivalently, the total link weight, in the projected network of N_A actors is given by

$$L = \frac{N_A(N_A - 1)}{2} \cdot \frac{2p' N_T}{N_A}, \tag{3.54}$$

and therefore, the equivalent p is given by

$$p = \frac{2p' N_T}{N_A}. \tag{3.55}$$

Then, an analytic expression for the average maximum $\langle Q^{IP} \rangle$ modularity and variance $\sigma^2_{\langle Q^{IP} \rangle}$ can be given by substituting

$$N \to N_A \tag{3.56}$$

$$p \to \frac{2p' N_T}{N_A} \tag{3.57}$$

into Eqn. 3.43 and Eqn. 3.50, giving

$$\begin{aligned} \langle Q^{IP} \rangle &= \left(1 - \frac{7}{5} e^{-\frac{N_A}{50}}\right) 0.97 \sqrt{\frac{1 - \frac{2p' N_T}{N_A}}{2p' N_T}} \\ &+ \left(\frac{2p' N_T}{N_A}\right)^{-\frac{1}{6}\log\left(\frac{2}{5}N_A\right)} \left(1 - \frac{2p' N_T}{N_A}\right)^{\frac{5}{4}} N_A^{-\frac{6}{5} + \frac{13}{15} e^{-\frac{N_A}{100}}} \end{aligned} \tag{3.58}$$

$$\sigma^2_{\langle Q^{IP} \rangle} = \left(15 - e^{-(N_A - 10)/30}\right) \frac{0.97^2}{2} \frac{1}{N_A (2p' N_T)^2}. \tag{3.59}$$

Then for a particular network with an IP modularity score of Q^{IP}, the Z-score is given by

$$z = \frac{Q^{IP} - \langle Q^{IP} \rangle}{\sigma_{\langle Q^{IP} \rangle}}. \tag{3.60}$$

3.5 Conclusion

This chapter introduced and discussed methods that maximize modularity for detecting communities in bipartite networks when the links are both weighted and unweighted. There are two general methods to detect communities in bipartite networks. One can choose to bicluster both types of nodes into one community or create two separate unipartite projections and detect communities in each projected network. To bicluster a network, Barber introduced a generalization of Newman's modularity Q that nullifies the expectation that two nodes of the same type in a bipartite network are connected. This method generalizes easily when the links in the network are weighted. To detect communities by projecting a bipartite network into a unipartite network, a standard method currently used is the method introduced by Guimera. Unfortunately, this method does not generalize when the links in the bipartite network are weighted. Additionally, it was shown there is a problem in the definition of the projected modularity, which allows nodes to be partitioned into singleton clusters without a modularity penalty. Therefore, the Information Preserving bipartite modularity was introduced that easily generalizes to a weighted bipartite network, does not suffer from the singleton cluster problem, and ,additionally, preserves more information about the bipartite network in the unipartite projection.

To find the partition that maximizes the modularity, the leading eigenvalue method with final tuning was generalized to maximize modularity in bipartite networks. Additionally, a novel agglomerative tuning step was introduced that considers iteratively merging communities to improve the modularity. This tuning step further improves the modularity, with a significant increase over final tuning for larger networks, when tested on both random bipartite Erdös-Rényi type networks and real

world networks. For unipartite networks, this extends the current algorithm's ability to find the partition with the largest value of modularity Q for any algorithm in networks of up to at least *tens* of thousands of nodes in size. Finally, the expected value and standard deviation of IP modularity for bipartite Erdös-Rényi type networks was then analytically derived. A Z-score comparison of the modularity values of real-world networks can now be computed.

It remains to be seen whether the IP modularity does in fact find meaningful partitions when detecting communities in networks. In the next chapter, the IP modularity is validated by testing its ability to recover the known structure of a bipartite model with unweighted and weighted links. Furthermore, the metabolic network of *E. coli* is used as a real-world example of the applicability of the method.

Chapter 4

Validation and Application of the Information Preserving Modularity

4.1 Introduction

In Chapter 3, two different methods were discussed to find communities in bipartite networks. The first method is biclustering in which both types of node in the bipartite network are partitioned into the same community. The most common modularity definition used to bicluster nodes is the modularity of Barber [176], applicable when the links in a bipartite network are either weighted are unweighted. The other method that was discussed was unipartite projection in which each type of node in the bipartite network may be projected onto a unipartite network. Communities are then detected in the unipartite projection. The most common modularity definition

used for the unipartite projection method is the modularity of Guimera *et al.* [175]. It was shown in Chapter 3 that the Guimera *et al.* modularity suffers from the fact that it is not easily generalized to the case in which the links in the bipartite network are weighted and, additionally, there is a problem in which nodes are likely to be assigned into singleton clusters. Therefore, a new bipartite modularity was introduced, the Information Preserving modularity, whose projection preserves more of the bipartite information, generalizes to weighted links, and whose modularity measure does not suffer from the problem of finding singleton clusters.

In this chapter, the Information Preserving modularity's ability to recover meaningful community structure is validated in two ways. First, the IP modularity is tested by using it to find the structure in a bipartite network model that is constructed *a priori* with a community structure. Then, the effectiveness of IP modularity is compared to that of Guimera's and Barber's by using them to recover the assigned partition when the links in the generated network are unweighted and weighted, respectively. Additionally, the real-world metabolic network of *Escherichia coli* is used as an example application and a validation of the use of IP modularity. To this end, the IP, Barber, and Guimera modularity methods are used to detect communities in the metabolic network and these communities are tested for biological relevance in their ability to significantly enrich for gene ontology (GO) terms and metabolites grouped by their Metabolic reaction classification.

4.2 Bipartite planted partition model

As discussed in Chapter 1, the planted l partition model [180] is a standard method to assess the performance of community detection methods in unipartite networks. The

model tests how well a community detection method is able to recover l communities each containing n assigned nodes as noise is added to the network. A similar model is needed to test community detection methods in bipartite networks. One is now introduced.

4.2.1 The model

To compare methods to find bipartite communities, a bipartite model based on the unipartite planted partition model [180] is now constructed. A bipartite network contains two types of nodes and for the following discussion nodes of the first type are called the "actor" set and nodes of the second type are called the "team" set. Then the model begins with l communities each containing a actors and t teams. Actors are connected to teams in the same community with probability p_{in} and connected to teams in different communities with probability p_{out}. The average actor degree, $\langle k_a \rangle$, and average team degree, $\langle k_t \rangle$, are given by

$$\begin{aligned} \langle k_a \rangle &= t\,p_{in} + (l-1)\,t\,p_{out} \\ \langle k_t \rangle &= a\,p_{in} + (l-1)\,a\,p_{out}. \end{aligned} \tag{4.1}$$

For the majority of bipartite community detection methods that will be evaluated only one set of nodes in the bipartite network is partitioned. In this case, the set in the planted partition model that is partitioned is chosen to be the actor set. Then, for each actor, $k_{intra} \equiv t\,p_{in}$ is defined as the average intra-community degree and $k_{inter} \equiv (l-1)\,t\,p_{out}$ is defined as the average inter-community degree of each actor.

For a given a, t, l, and $\langle k_a \rangle$, if $k_{inter} = 0$, then the l communities are well defined with no inter-community links. When k_{inter} is increased, the number of inter-community links increases until $p_{in} = p_{out} = \langle k_a \rangle / (t\,l)$ at which point the

network is a completely random, Erdös-Rényi type bipartite network. This occurs for $k_{inter} = (l-1)\langle k_a \rangle / l \equiv k_{max}$ and, thus, depends on both the number of communities and the average actor degree. Therefore, to evaluate community detection methods, for example as $\langle k_a \rangle$ increases or decreases, define $k_{norm} \equiv k_{inter}/k_{max}$. The network is composed of l disconnected communities when $k_{norm} = 0$ and is completely random when $k_norm = 1$. As in the unipartite case, the quality of a community detection method is evaluated by how well it recovers the l communities of a nodes as a function of k_{inter}, or equivalently of k_{norm}.

4.2.2 Partition quality

To measure the performance of a community detection method, the amount of overlap between the planted partition of the l communities of a nodes and the partition found by a community detection method must be described quantitatively. For this purpose, the mutual information between found and known partitions can be used [253]. The normalized mutual information I between partitions A and B is defined as

$$I_{AB} = \frac{-2\sum_{i=1}^{N_M^A}\sum_{j=1}^{N_M^B} n_{ij}^{AB} \ln\left(\frac{n_{ij}^{AB} N}{n_i^A n_j^B}\right)}{\sum_{i=1}^{N_M^A} n_i^A \ln\left(\frac{n_i^A}{N}\right) + \sum_{j=1}^{N_M^B} n_j^B \ln\left(\frac{n_j^B}{N}\right)}, \tag{4.2}$$

where N is the total number of actors in the network, N_m^A is the number of communities in partition A, n_i^A is the number of nodes in community i for partition A, n_{ij}^{AB} is the number of nodes in community i of partition A and community j of partition B. The normalized mutual information is then 1 if the two partitions are identical and 0 if there is no correlation.

To further elucidate the behavior of the mutual information, I_{AB}, perhaps it's

simplest to consider a unipartite model. Consider an initial partition, P_o, in which N nodes have been assigned to 4 communities of equal size. For example, for a partition of $N = 20$ nodes, a possible P_o is $P_o = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4]$. Now, consider a separate partition P_r that, initially, is equal to P_o. When $P_r = P_o$, the mutual information between the two partitions is equal to 1. However, as P_r changes the mutual information should decrease. To see exactly how this decrease behaves as the difference between P_r and P_o increases, consider Fig. 4.1. In Fig. 4.1 the ensemble-averaged mutual information between P_o and P_r is plotted as a function of the fraction of nodes that are randomly reassigned in P_r, for various sizes of N. As more of the nodes in P_r are randomly assigned, the amount of overlap between the original partition P_o and P_r decreases, and I_{AB} decreases toward 0. This is also true as the network increases in size. Note that I_{AB} is not exactly zero when all the nodes have been randomly assigned because the random assignment has been restricted to one of the four existing communities. The mutual information would continue to decrease if the nodes are randomly assigned to more than four communities.

4.2.3 Unweighted links

To test the ability to detect communities in unweighted bipartite networks, the Information Preserving (IP) modularity and the Guimera (G) modularity are now compared using the bipartite l-partition model. In Fig. 4.2a there are 4 communities each containing 50 actors and 12 teams. In Fig. 4.2a, one can see when the average actor degree, $\langle k_a \rangle$, is equal to the number of teams the G and IP modularity curves overlap. The planted partition is particularly easy to find because when $\langle k_a \rangle$ is equal to the number of teams, each community begins, at $k_{norm} = 0$, fully connected with

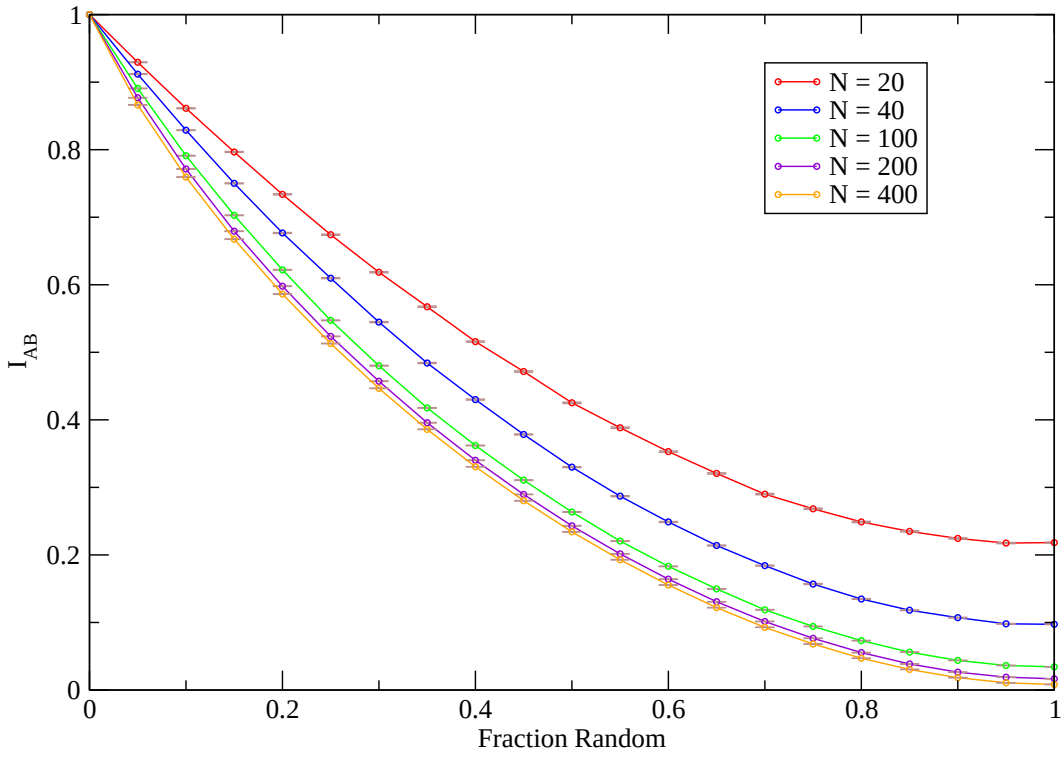
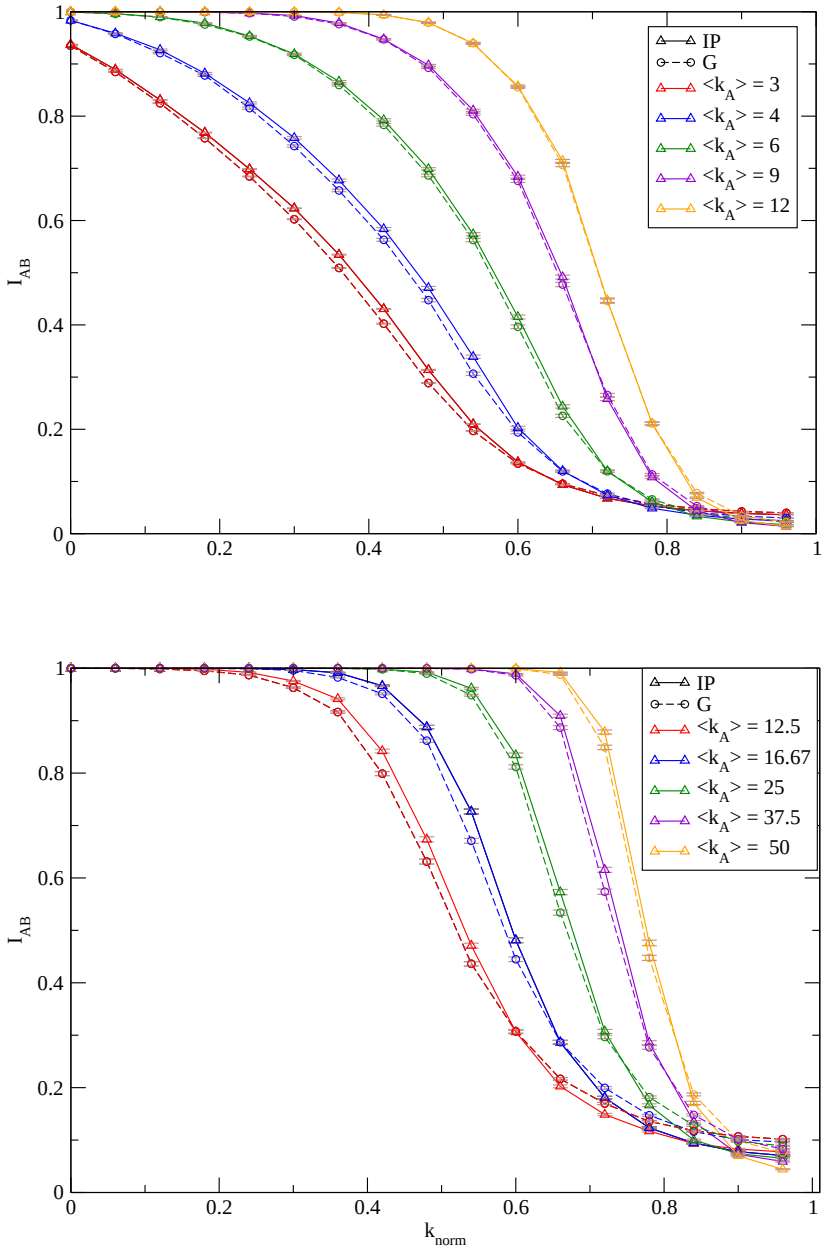


Figure 4.1: **The behavior of the mutual information, $\mathbf{I_{AB}}$.** The ensemble-averaged mutual information between an initial partition P_o and a partition P_r is plotted as a function of the fraction of nodes that are randomly reassigned in the P_r partition. The number of nodes, N, in the partition is given in the legend and the size of the P_r ensemble used in the calculation is 10,000.

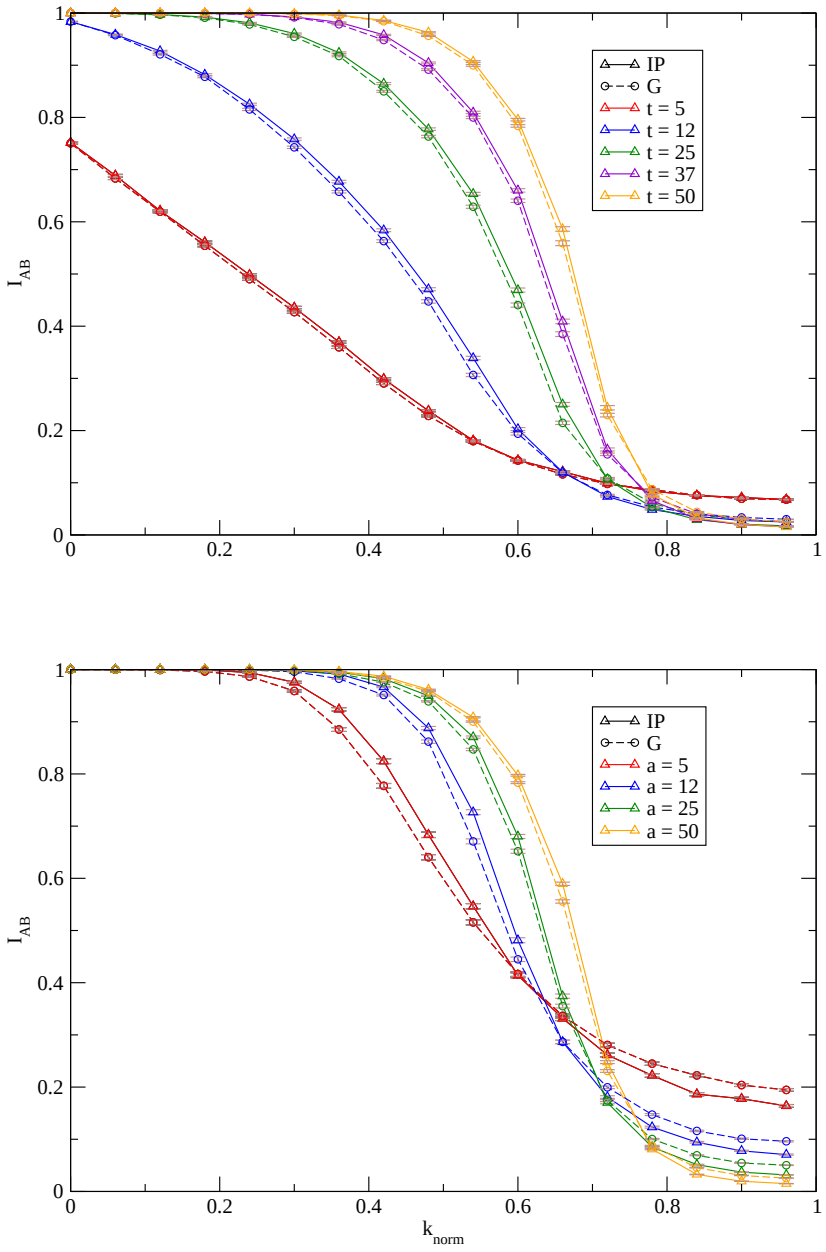
each actor connected to all teams in its assigned community. However, as $\langle k_a \rangle$ decreases, and there is therefore less information about the planted partition, the IP modularity recovers a larger amount of the planted partition as indicated by a larger average I_{AB}. In Fig. 4.2b the same analysis is performed as in Fig. 4.2a but for the opposite set of nodes in the bipartite network which contains $a = 12$ nodes per community. Both modularities are able to recover a larger amount of the planted partition for larger values of k_{norm} when compared to the results of Fig. 4.2a. This is due to the average degree, $\langle k \rangle$, being larger. So, for example, when each side of the network is on average connected to $\frac{1}{2}$ of the nodes of the opposite type the average degree when $a = 50$ is 6 while the average degree when $a = 12$ is 25. For most cases, the IP modularity finds a higher average I_{AB} than the Guimera modularity. There is a point however, when the Guimera I_{AB} curve crosses the IP I_{AB} curve and is higher for some values of k_{norm}. This crossover is most evident in Fig 4.2b. However, the crossover occurs for large values of k_{norm} and small values of I_{AB} indicating a large amount of inter-community links and a low amount of overlap with the planted partition.

To investigate if these results change as the ratio of teams to actors in each community changes, models were generated with 4 communities, 50 actors, and $\langle k_a \rangle = \frac{1}{3}t$. The number of teams in each community was varied, and the results are shown in Fig. 4.3a. From the figure, the IP modularity consistently performs better than the G modularity with the closest performance pertaining to the extreme case when there are very few teams to actors (ratio of 0.1). In Fig. 4.3b, the opposite side of the bipartite network shown in Fig. 4.3a is partitioned. Again, for most cases, the IP modularity finds a higher average I_{AB} than the Guimera modularity with a crossover occurring for large values of k_{norm} and small values of I_{AB}.



(b)

Figure 4.2: **Bipartite planted partition comparison for unweighted links and varying degree**. For each plot in the graph, at each value of k_{norm}, 1000 models of 4 communities are generated and for each model the modularity maximizing algorithm is run once and the average I_{AB} is calculated for the resulting partitions when (a) $a = 50$ and $t = 12$ and when (b) $a = 12$ and $t = 50$. The average degree of each node a is indicated for each plot in the legend.



(b)

Figure 4.3: **Bipartite planted partition comparison for unweighted links and varying set size**. For each plot in the graphs, at each value of k_{norm}, 1000 models of 4 communities are generated and for each model the modularity maximizing algorithm is run once and the average I_{AB} is calculated for the resulting partitions when (a) $a = 50$, t varies, and $\langle k_a \rangle = \frac{1}{3} \cdot t$ and when (b) a varies, $t = 50$, and $\langle k_a \rangle = \frac{50}{3}$. The size of the varied set is indicated for each plot in the legend.

4.2.4 Weighted links

To test the performance of the IP modularity for a weighted network consider the case in the bipartite l-planted partition model when $p_{in} = p_{out}$. For these values, the network is a completely random, Erdös-Rényi type bipartite network and is therefore ideal for adding a weighted constraint to the links. To do so, assume that if a link occurs in the bipartite network between an actor and team of the same community, a weight is assigned randomly from a uniform distribution of mean $\langle w_{intra} \rangle$ and width 0.5. Furthermore, assume that if a link occurs between two nodes in different communities a weight is assigned randomly from a uniform distribution of mean $\langle w_{inter} \rangle$ and width 0.5. Additionally, let $\langle w_{intra} \rangle + \langle w_{inter} \rangle = 2.0$. Then the quality of a community detection method can be evaluated by how well it recovers the l communities of a nodes as a function of $\langle w_{inter} \rangle$. For the following analysis, $\langle w_{inter} \rangle$ ranges from 0.25 to 1.0. Initially, when $\langle w_{inter} \rangle = 0.25$ any link between communities has an average value of 0.25 and a link within a community has an average value of 1.75, and the communities are well defined. At the end of the range with $\langle w_{inter} \rangle = \langle w_{intra} \rangle = 1.00$ the network becomes completely random.

To study the performance of the IP modularity, we can only compare it to that of Barber (B) because the Guimera modularity does not generalize to weighted links. The B modularity biclusters a network into communities, therefore when computing the normalized mutual information of a B modularity partition, only the overlap of the actor set is considered. In Fig. 4.4a, there are 4 communities, each containing 50 actors and 12 teams, and the average actor degree is varied. As the average degree decreases, there is less information about the planted partition and the mutual information decreases. From the figure, in each case, the B modularity is able to recover

more of the planted partition than the IP modularity. However, this is perhaps not surprising because the IP modularity creates a weighted projection which loses some information in the original bipartite network. Nevertheless, the IP modularity is able to recover a large amount of the planted partition structure. In Fig. 4.4b, the same analysis is performed as in Fig. 4.4a but for the opposite set of nodes in the bipartite network which contains $a = 12$ nodes per community. Again, the B modularity is able to recover a higher amount of the planted partition with a wider gap between the IP and B average mutual information than in the case of Fig. 4.4a. When projecting the smaller set of nodes in the bipartite network, much more information must be projected into a smaller number of nodes. This indicates there may be a limit to the projection method when studying the smaller side of a bipartite network and this limit is now studied further.

To investigate the behavior of the IP modularity as the ratio of teams to actors in each community changes, models were generated with 4 communities, 50 actors, and $\langle k_a \rangle = \frac{3}{2}t$. The number of teams in each community was varied and the results are shown in Fig. 4.5a. Just as in Fig. 4.4a, the B modularity has a higher mutual information but the gap between the IP and B modularities is consistent, which suggests that the projection is losing the same amount of information regardless of the team size. However, this is not the case when the smaller side of the network becomes the actor set. In Fig. 4.5b, the opposite side of the network is partitioned and one can see that as the size of the actor side decreases the gap between the mutual information curves of the B and IP modularity increases. The team side in the case stays the same at $t = 50$, but as the actor side decreases the projection must condense information about the 50 node side into a smaller number of actor nodes. As Fig. 4.5b shows, the mutual information curve for the IP modularity decreases quite rapidly

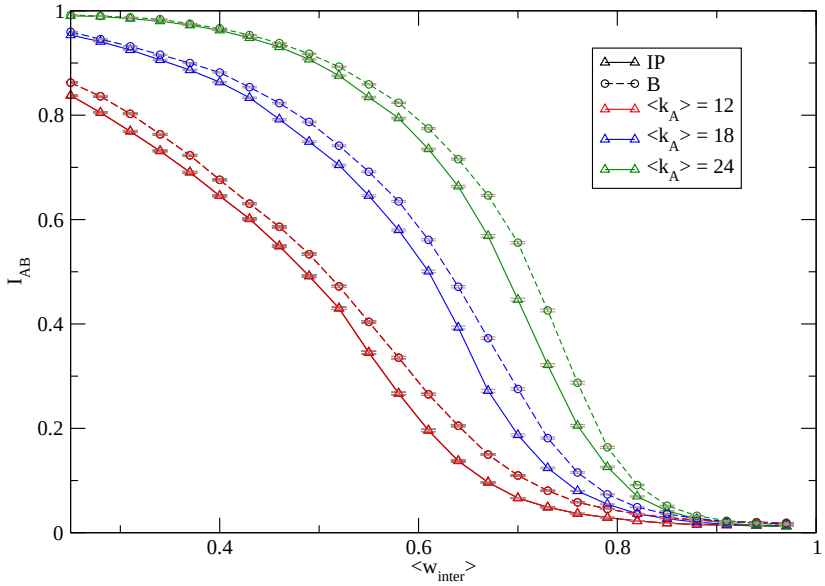
when the actor side of the network is about 10% the size of the team side of the bipartite network. Nevertheless, the IP modularity is able to find a large amount of the planted partition when the links in the bipartite network are weighted. Note that this does not suggest one should always use the Barber modularity. The bipartite planted partition model introduced here for comparison, biclusters the nodes in the network, with each type of node constrained to the same number of communities. This is exactly the type of partition the Barber modularity seeks to find. Therefore, it is reasonable that for this model the Barber modularity finds a higher amount of the planted partition. In the next section, the bipartite metabolic network of *Escherichia coli* is used to further compare bipartite modularity when partitioning real world networks.

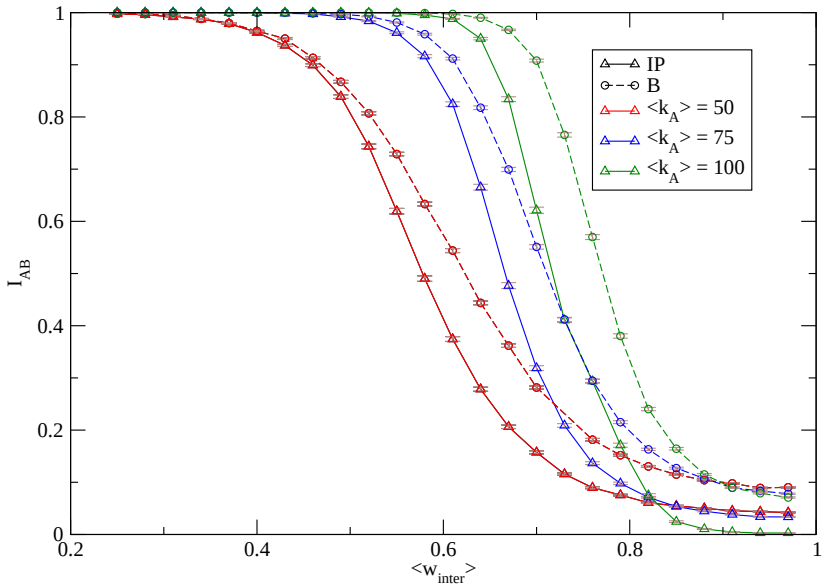
4.3 The *Escherichia coli* metabolic network

As seen in the previous section, the IP modularity performs well at recovering the partition of a model bipartite network where a known partition is defined *a priori*. However, in real world networks, communities are usually not known beforehand. Therefore, in this section, the IP modularity is applied to the real world bipartite metabolic network of *E. coli* as both a test and an example application.

4.3.1 Data

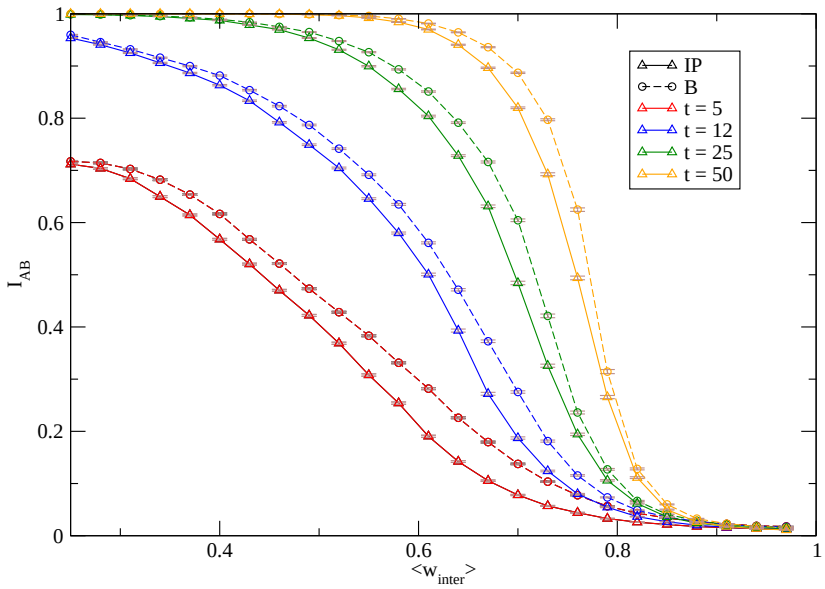
For the following analysis, a genome-scale reconstruction of *E. coli* metabolism is studied. This reconstruction accounts for 1366 genes, 2251 metabolic reactions, and 1136 metabolites of the laboratory strain *E.coli* K-12 MG1655 [254]. Then, to create

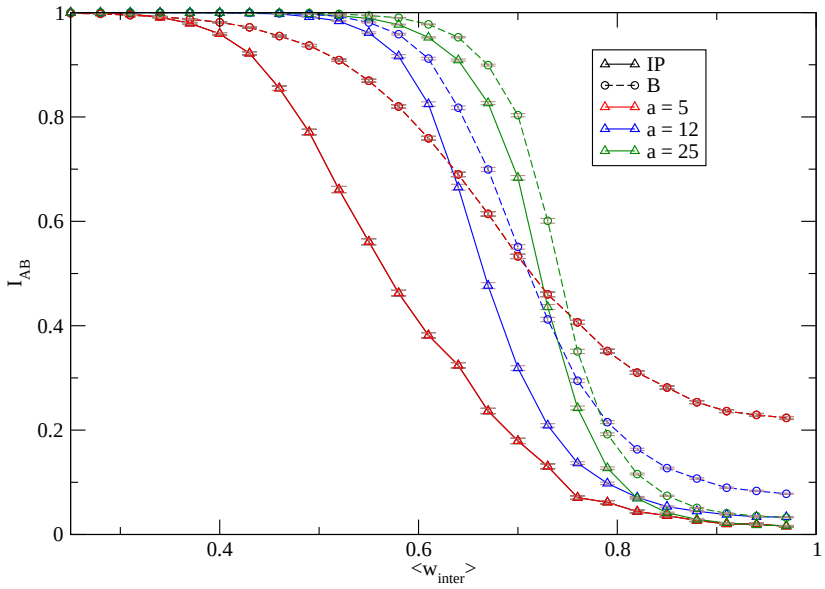




(b)

Figure 4.4: **Bipartite planted partition comparison for weighted links and varying degree**. For each plot in the graphs, at each value of k_{norm}, 1000 models of 4 communities are generated and for each model the modularity maximizing algorithm is run once and the average I_{AB} is calculated for the resulting partitions when (a) $a = 50$ and $t = 12$ and when (b) $a = 12$ and $t = 50$. The average degree of each node a is indicated for each plot in the legend.





(b)

Figure 4.5: **Bipartite planted partition comparison for weighted links and varying set size**. For each plot in the graph, at each value of k_{norm}, 1000 models of 4 communities are generated and for each model the modularity maximizing algorithm is run once and the average I_{AB} is calculated for the resulting partitions when (a) $a = 50$, t varies, and $\langle k_a \rangle = \frac{3}{2} \cdot t$ and when (b) a varies, $t = 50$, and $\langle k_a \rangle = 75$. The size of the varied set is indicated for each plot in the legend.

a network, each gene, representing an enzyme, and metabolite are represented as a node. A link is placed between a metabolite and gene if the gene's corresponding enzyme is involved in catalyzing a reaction involving the metabolite. To simplify the network representation, all sink and transport equations are ignored as well as the direction of the reactions. This naturally creates an unweighted undirected bipartite network with two types of nodes representing the enzymes and metabolites. With the transport and sink equations removed the resulting bipartite network is composed of 1062 genes and 1094 metabolites.

4.3.2 Detecting communities

To detect communities in the metabolic network, the same leading eigenvalue method with final tuning [9] and agglomeration used in Section 4.2 is applied to maximize the Barber, Guimera, and IP modularity. Additionally, variants of the analysis methods developed in Chapter 2 for unipartite networks are used in this bipartite analysis. The community detection algorithm for each modularity is run 1000 times and the ensemble of partitions is analyzed. Maximizing the Barber modularity results in co-clustered communities of both enzyme and metabolite nodes. For the modularities of both the Guimera and IP, a unipartite projection from the bipartite network is created for each type of node in the network and each network is then clustered separately. Creating a unipartite projection results in a separate enzyme network and metabolite network for the Guimera and IP modularity methods. As previously discussed in Chapter 2, community detection algorithms contain stochastic elements that can result in different partitions for multiple runs. This property is again exploited, by analyzing the ensemble of 1000 partitions and finding each core

community of nodes, which is defined as all pairs of nodes that are always found together in the same community for the entire ensemble.

The resulting modularity values found for the ensemble of partitions using all three modularity methods is described in Table 4.1. For each method, the average modularity, its standard deviation and the maximum modularity are given for the ensemble of 1000 partitions. As discussed in Section 1.7.3, a better measure of a partitions modular strength is given by the z-score. Therefore, the corresponding z-score, its standard deviation, and maximum z-score are also computed. The z-score was computed using Eqn. 1.28, by finding the average modularity and standard deviation of a 250,000 ensemble of random bipartite Erdös-Rényi type networks with 1062 nodes of type 1, 1094 nodes of type 2 and the same average number of links as the *E.Coli* metabolic network. For the 1062 node side of the random bipartite Erdös-Rényi type network, the ensemble averaged IP modularity was 0.124384 with an analytic prediction using Eqn. 3.58 of 0.230350. For the 1094 node side of the random bipartite Erdös-Rényi type network, the ensemble averaged IP modularity was 0.125389 with an analytic prediction using Eqn. 3.58 of 0.219799. For all modularities, the average z-score of the partitions found was large indicating a high modular strength. Analyzing the ensemble of partitions for the Barber method there is a mode of 17 communities with the largest community consisting of 923 nodes. For the enzyme side of the metabolic network there is a mode of 67 and 12 communities for the Guimera and IP modularity, respectively, with the largest community consisting of 671 and 311 genes, respectively. For the metabolite side of the metabolic network, there is a mode of 29 and 12 communities for the Guimera and IP modularity, respectively, with the largest community consisting of 913 and 997 metabolites, respectively.

	Q					
	Avg Mod	SD	Max Mod	Avg Z	SD	Max Z
Barber	0.609534	0.003449	0.616410	77.842342	0.898230	79.633225
Guimera						
Enzy	0.407850	0.0	0.407850	123.747782	0.0	123.747782
Met	0.561262	0.000828	0.562958	191.188834	0.365799	191.776834
IP						
Enzy	0.478747	0.000029	0.478748	171.404750	0.014062	171.405195
Met	0.441709	0.002060	0.444633	151.734026	0.988340	153.136815

Table 4.1: Comparison of the modularity Q and z-score found for the *Escherichia coli* metabolic network using the Barber modularity, Guimera modularity, and the IP modularity. For each bipartite modularity, the resulting ensemble of 1000 partition's average modularity (Avg Mod) and its standard deviation (SD), maximum modularity (Max Mod), average Z-score (Avg Z) and its standard deviation (SD), and the maximum Z-score is reported.

4.3.3 Enrichment

4.3.3.1 Enzyme communities

To analyze and compare the community detection results for each modularity method, each gene core community is compared to known biological functional communities in *E. coli* using the gene ontology identified in *E. coli* [235, 236, 237] using a hypergeometric test with Benjamini-Hochberg correction [238] as described in Section 2.5.1. The hypergeometric test with Benjamini-Hochberg correction results in a P value that is the probability of the overlap between a core community and GO term occurring randomly.Then, P-values lower than 0.05 are considered statistically significant. We limit the study to account only for communities of size 5 or greater. For the co-clustered communities of the Barber method the metabolites are removed from each community during the analysis so that each "core community" is only composed of genes.

4.3.3.2 Metabolite communities

In the genome-scale reconstruction of *E. coli* metabolism being studied, each reaction is classified by a metabolic subtype such as Nucleotide Salvage Pathway, Glycerophospholipid Metabolism, Cofactor and Prosthetic Group Biosynthesis, Nitrogen Metabolism and Cysteine Metabolism [254]. Previously, it has been shown that metabolites are likely to be grouped into communities whose members share a common subtype [41]. To test for this, the metabolites in the dataset are assigned to groups whose constituents are involved in reactions of the same metabolic subtype. Note that this means a metabolite may be assigned to more than one group if it is

involved in multiple disparate reactions. Then each metabolic community of size 5 or larger is tested for significant overlap with a metabolic subtype group using a hypergeometric test with Benjamini-Hochberg correction as described in Section 2.5.1. For the co-clustered communities of the Barber method the genes are removed from each community during the analysis so that each "core community" is only composed of metabolites.

4.3.4 A comparison of the 3 bipartite modularities

For communities identified using the Barber, Guimera and IP modularity method, 152, 127 and 167 statistically significant matches are found between gene core communities and GO terms, respectively. Not only are there more statistically significant matches using the IP modularity method, but the most significant enrichments, measured by smallest P value, were found using the IP modularity. Tables 4.2, 4.3, and 4.4 detail the results for the 20 most significant enrichments for the Barber, Guimera, and IP modularity, respectively. The most significant gene core community has a P value of 1.34e-34 and 2.05e-23 for the Barber and Guimera modularity methods, respectively. The Barber community with the most significant functional enrichment contains 78 genes, including 33 of 39 genes in the GO Term for anaerobic respiration. The Guimera community with the most significant functional enrichment contains 43 genes, including 20 of 24 genes involved in the translation process. The most significant gene core community for the IP modularity has a P value of 3.22e-78. This community contains 234 genes, including 141 of 170 genes involved in ATP binding. Furthermore, there is one other gene core community contributing a

smaller P value of 3.37e-40 using the IP modularity than the most significant community found using each of the other two methods. Additionally, Tables 4.5, 4.6, and 4.7 give the single most significant enrichment for each gene core community using the Barber, Guimera, and IP modularity, respectively.

For the metabolite communities, there are 16, 33, and 30 statistically significant enrichments of metabolic subtypes using the Barber, Guimera, and IP modularity method, respectively. Table 4.9 displays the metabolite community enrichments found using the Barber modularity and Tables 4.8 and 4.10 detail the results for the 20 most significant metabolite community enrichments for the Guimera, and IP modularity, respectively. Interestingly, although the Guimera and IP modularity have a similar number of significant enrichments, the most significant is found using the Barber modularity. Nevertheless, the 30 significant enrichments provides further confirmation that the IP modularity is able to find biologically meaningful communities. Additionally, Tables 4.11, 4.12, and 4.13 give the single most significant enrichment for each metabolite core community using the Barber, Guimera, and IP modularity, respectively. With these results, to detect communities in bipartite networks, we recommend that both the Barber modularity and IP modularity be used. These modularity definitions are each representative of the two methodologies of bipartite network analysis and can be applied to both weighted and unweighted bipartite networks.

4.4 Conclusion

In this chapter the ability of using IP modularity to find meaningful partitions was tested and validated using both a computational model and a real-world network.

First, a model network was created whose partitioning is defined, a planted partition, and the ability of the IP modularity to recover the partition was tested as noise was added to the network. When the links in the model network were unweighted, the IP modularity was able to recover more of the planted partition when compared to the Guimera modularity. Additionally, weights were added to the links in the bipartite network and the IP modularity was compared to the Barber modularity's ability to recover the planted partition. In this case, the Barber modularity was able to recover a larger amount of the planted partition. However, the IP modularity creates a unipartite projection from the bipartite network and therefore some of the information of the bipartite network must be lost. Nevertheless, the IP modularity was able to recover a large amount of the planted partition. Furthermore, the bipartite planted partition model biclusters the nodes in the network, with each type of node constrained to the same number of communities. This is exactly the type of partition the Barber modularity seeks to find. Therefore, it is reasonable that for this model the Barber modularity finds a higher amount of the planted partition.

As an example application to a real-world network, the IP modularity was applied to the metabolic network of *E. coli* [254]. An unweighted undirected bipartite network is created from a genome-scale reconstruction by connecting an enzyme and metabolite if that particular enzyme was known to catalyze a reaction involving the metabolite. Next, communities were detected in the network using the Barber, Guimera, and IP modularity and for each modularity an ensemble of 1000 partitions were analyzed. To quantify the modular strength the z-score of the resulting partitions for each bipartite modularity was computed. "Core community" of genes and separately "core community of metabolites" were created by finding those nodes

who were always found together in the same community for the ensemble of partitions. These core communities were then tested for biological significance by testing whether the gene core communities and metabolite core communities significantly enrich for gene ontology (GO) terms and Metabolic reaction classification, respectively. The IP modularity found the highest number of total significant enrichments with 197 followed by the Barber modularity with 168 and finally the Guimera modularity with 160. With these results, the IP modularity is shown to find meaningful community structure in a bipartite network. Furthermore, the ability to partition bipartite networks by creating a unipartite projection from the bipartite network has been extended to weighted networks using the IP modularity.

Table 4.2: The 20 most relevant gene relationships found for the Barber modularity.

P value	GO term	Com size	GO size	In com	Description
1.34e-31	9061	78	39	33	anaerobic respiration
1.70e-25	9055	78	71	38	electron carrier activity
3.86e-20	5506	78	111	41	iron ion binding
9.74e-20	22900	78	59	31	electron transport chain
3.01e-17	15986	11	9	9	ATP synthesis coupled proton transport
7.70e-17	8654	26	13	12	phospholipid biosynthetic process
1.70e-16	5886	78	192	48	plasma membrane integral to plasma membrane
1.76e-15	9252	18	37	14	peptidoglycan biosynthetic process
5.30e-15	46933	11	8	8	" hydrogen ion transporting ATP synthase activity, rotational mechanism
7.85e-15	55114	78	248	52	oxidation reduction
1.33e-14	8658	18	9	9	penicillin binding
5.66e-14	16491	78	165	42	oxidoreductase activity
7.48e-14	6810	78	113	35	transport
1.70e-13	45272	78	13	13	plasma membrane respiratory chain complex I
1.78e-13	51539	78	61	26	" 4 iron, 4 sulfur cluster binding
1.89e-13	30964	78	13	13	NADH dehydrogenase complex
9.31e-13	46961	11	7	7	" proton-transporting ATPase activity, rotational mechanism
9.36e-13	9002	18	8	8	serine-type D-Ala-D-Ala carboxypeptidase activity
2.25e-12	3954	78	12	12	NADH dehydrogenase activity
1.16e-10	6631	8	13	7	fatty acid metabolic process

Table 4.3: The 20 most relevant gene relationships found for the Guimera modularity.

P value	GO term	Com size	GO size	In com	Description
2.05e-23	6412	43	24	20	translation
5.87e-17	4812	43	15	14	aminoacyl-tRNA ligase activity
1.76e-15	9252	18	37	14	peptidoglycan biosynthetic process
1.22e-13	166	43	176	31	nucleotide binding
2.04e-13	6418	43	14	12	tRNA aminoacylation for protein translation
1.40e-12	9002	18	8	8	serine-type D-Ala-D-Ala carboxypeptidase activity
8.35e-12	8658	18	9	8	penicillin binding
1.58e-11	16874	43	32	15	ligase activity ligase activity, forming carbon-oxygen bonds
4.86e-11	5524	43	170	28	ATP binding
1.23e-10	30170	20	49	13	pyridoxal phosphate binding
1.00e-09	8360	18	29	10	regulation of cell shape
3.84e-09	9263	14	6	6	deoxyribonucleotide biosynthetic process
2.23e-08	55114	671	248	200	oxidation reduction
5.67e-08	15293	10	5	5	symporter activity
1.57e-07	15035	14	9	6	protein disulfide oxidoreductase activity
2.38e-07	46930	5	4	4	pore complex
3.86e-07	16837	18	5	5	carbon-oxygen lyase activity, acting on polysaccharides
1.16e-06	104	32	6	6	succinate dehydrogenase activity
1.29e-06	43169	17	9	6	cation binding
1.72e-06	6631	6	13	5	fatty acid metabolic process

Table 4.4: The 20 most relevant gene relationships found for the IP modularity.

P value	GO term	Com size	GO size	In com	Description
3.22e-78	5524	234	170	141	ATP binding
1.76e-68	166	234	176	137	nucleotide binding
3.37e-40	9055	120	71	55	electron carrier activity
2.63e-31	22900	120	59	45	electron transport chain
2.31e-26	9252	50	37	26	peptidoglycan biosynthetic process
1.04e-25	5886	120	192	72	plasma membrane integral to plasma membrane
8.67e-25	9061	120	39	33	anaerobic respiration
3.34e-24	5506	120	111	54	iron ion binding
7.04e-24	55114	120	248	79	oxidation reduction
5.40e-21	51539	120	61	38	4 iron, 4 sulfur cluster binding
9.56e-21	6810	120	113	51	transport
6.47e-18	8137	120	20	20	NADH dehydrogenase (ubiquinone) activity
7.18e-18	9060	120	20	20	aerobic respiration
9.93e-18	16491	120	165	58	oxidoreductase activity
1.36e-17	16874	234	32	31	ligase activity ligase activity, forming carbon-oxygen bonds
1.39e-15	8360	50	29	18	regulation of cell shape
8.01e-14	30170	296	49	41	pyridoxal phosphate binding
9.37e-14	48038	120	18	17	quinone binding ubiquinone binding
1.49e-13	19439	15	11	9	aromatic compound catabolic process
3.34e-13	7047	50	27	16	cellular cell wall organization oncogenesis

Table 4.5: The single most relevant gene relationship for each core community found for the Barber modularity.

P value	GO term	Com size	GO size	In com	Description
1.34e-31	9061	78	39	33	anaerobic respiration
3.01e-17	15986	11	9	9	ATP synthesis coupled proton transport
7.70e-17	8654	26	13	12	phospholipid biosynthetic process
1.76e-15	9252	18	37	14	peptidoglycan biosynthetic process
1.16e-10	6631	8	13	7	fatty acid metabolic process
1.87e-09	16226	10	7	6	iron-sulfur cluster assembly
1.02e-08	9263	16	6	6	deoxyribonucleotide biosynthetic process
7.14e-07	9486	6	4	4	cytochrome bo3 ubiquinol oxidase activity
7.14e-07	8745	6	4	4	N-acetylmuramoyl-L-alanine amidase activity
2.99e-05	43169	5	9	4	cation binding
4.33e-05	8253	16	4	4	5'-nucleotidase activity
1.26e-04	8556	5	3	3	potassium-transporting ATPase activity
1.52e-04	16491	11	165	10	oxidoreductase activity
1.68e-04	9401	5	4	3	phosphoenolpyruvate-dependent sugar phosphotransferase system
2.50e-03	16887	6	6	3	ATPase activity endodeoxyribonuclease activity, producing 5'-phosphomonoesters
2.23e-02	8942	5	2	2	nitrite reductase [NAD(P)H] activity
3.34e-02	16961	6	2	2	mitochondrion inheritance

Table 4.6: The single most relevant gene relationship for each core community found for the Guimera modularity.

P value	GO term	Com size	GO size	In com	Description
2.05e-23	6412	43	24	20	translation
1.76e-15	9252	18	37	14	peptidoglycan biosynthetic process
1.23e-10	30170	20	49	13	pyridoxal phosphate binding
3.84e-09	9263	14	6	6	deoxyribonucleotide biosynthetic process
2.23e-08	55114	671	248	200	oxidation reduction
5.67e-08	15293	10	5	5	symporter activity
2.38e-07	46930	5	4	4	pore complex
1.16e-06	104	32	6	6	succinate dehydrogenase activity
1.29e-06	43169	17	9	6	cation binding
1.72e-06	6631	6	13	5	fatty acid metabolic process
6.43e-06	8415	7	31	6	acyltransferase activity
1.18e-05	8253	12	4	4	5'-nucleotidase activity
1.73e-05	8415	55	31	12	acyltransferase activity
9.23e-05	43171	19	4	4	peptide catabolic process
1.69e-04	8654	5	13	4	phospholipid biosynthetic process
3.01e-04	4332	25	4	4	fructose-bisphosphate aldolase activity
1.16e-03	8654	7	13	4	phospholipid biosynthetic process
3.60e-03	8742	13	3	3	L-ribulose-phosphate 4-epimerase activity
6.99e-03	8168	5	8	3	methyltransferase activity
4.92e-02	9103	6	45	4	lipopolysaccharide biosynthetic process

Table 4.7: The single most relevant gene relationship for each core community found for the IP modularity.

P value	GO term	Com size	GO size	In com	Description
3.22e-78	5524	234	170	141	ATP binding
3.37e-40	9055	120	71	55	electron carrier activity
2.31e-26	9252	50	37	26	peptidoglycan biosynthetic process
8.01e-14	30170	296	49	41	pyridoxal phosphate binding
1.49e-13	19439	15	11	9	aromatic compound catabolic process
4.08e-13	8654	47	13	12	phospholipid biosynthetic process
7.40e-13	6631	9	13	8	fatty acid metabolic process
4.32e-11	43169	19	9	8	cation binding
1.82e-10	45454	34	13	10	cell redox homeostasis
2.73e-07	16226	43	7	7	iron-sulfur cluster assembly
2.09e-05	287	116	129	35	magnesium ion binding
1.17e-02	5975	75	45	12	carbohydrate metabolic process

Table 4.8: The relevant metabolite relationships found for the Barber modularity.

P value	Com num	Com size	GO size	In com	Subtype
6.60e-74	145	92	158	86	Glycerophospholipid Metabolism
1.46e-21	216	18	17	13	Murein Biosynthesis
4.68e-20	100	29	103	24	Nucleotide Salvage Pathway
4.12e-14	216	18	35	12	Murein Recycling
3.81e-12	35	17	50	12	Oxidative Phosphorylation
2.43e-08	77	17	255	16	Cofactor and Prosthetic Group Biosynthesis
4.28e-05	35	17	30	6	Nitrogen Metabolism
5.68e-05	80	9	103	7	Nucleotide Salvage Pathway
2.84e-04	82	15	103	8	Nucleotide Salvage Pathway
4.24e-04	35	17	28	5	Citric Acid Cycle
1.09e-03	35	17	58	6	Unassigned
1.33e-02	224	7	223	6	Alternate Carbon Metabolism
1.40e-02	82	15	38	4	Cysteine Metabolism
3.51e-02	35	17	52	4	Inorganic Ion Transport and Metabolism
3.94e-02	35	17	51	4	Purine and Pyrimidine Biosynthesis
4.87e-02	82	15	58	4	Unassigned

Table 4.9: The 20 most relevant metabolite relationships found for the Guimera modularity.

P value	Com num	Com size	GO size	In com	Subtype
1.09e-53	95	126	158	88	Glycerophospholipid Metabolism
1.06e-32	25	58	103	42	Nucleotide Salvage Pathway
1.57e-22	21	41	35	21	Murein Recycling
5.38e-20	21	41	17	15	Murein Biosynthesis
1.54e-08	13	37	23	10	Histidine Metabolism
5.27e-07	13	37	64	13	Arginine and Proline Metabolism
5.30e-07	13	37	44	11	Tyrosine, Tryptophan, and Phenylalanine Metabolism
1.49e-06	13	37	30	9	Valine, Leucine, and Isoleucine Metabolism
1.71e-06	105	19	223	15	Alternate Carbon Metabolism
2.26e-06	99	6	30	5	Nitrogen Metabolism
8.33e-05	15	75	255	35	Cofactor and Prosthetic Group Biosynthesis
9.69e-05	19	28	255	18	Cofactor and Prosthetic Group Biosynthesis
1.57e-04	5	12	223	10	Alternate Carbon Metabolism
2.08e-04	15	75	14	7	Methylglyoxal Metabolism
2.66e-04	1	61	51	12	Purine and Pyrimidine Biosynthesis
6.53e-04	1	61	50	11	Oxidative Phosphorylation
6.72e-04	19	28	28	6	Methionine Metabolism
1.11e-03	95	126	143	30	Cell Envelope Biosynthesis
3.25e-03	1	61	52	10	Inorganic Ion Transport and Metabolism
4.22e-03	13	37	13	4	Glutamate Metabolism
5.05e-03	21	41	98	11	Lipopolysaccharide Biosynthesis / Recycling
1.17e-02	15	75	13	5	Glyoxylate Metabolism
1.25e-02	15	75	52	10	Inorganic Ion Transport and Metabolism
1.29e-02	13	37	18	4	Alanine and Aspartate Metabolism

Table 4.10: The 20 most relevant metabolite relationships found for the IP modularity.

P value	Com num	Com size	GO size	In com	Subtype
5.95e-49	24	78	103	58	Nucleotide Salvage Pathway
1.93e-23	14	137	68	43	tRNA Charging
3.63e-19	22	47	35	20	Murein Recycling
6.30e-19	22	47	17	15	Murein Biosynthesis
4.87e-16	18	103	223	58	Alternate Carbon Metabolism
1.30e-15	3	43	223	34	Alternate Carbon Metabolism
4.67e-12	18	103	50	24	Oxidative Phosphorylation
7.13e-11	5	21	255	20	Cofactor and Prosthetic Group Biosynthesis
2.12e-09	26	31	23	10	Histidine Metabolism
6.46e-09	18	103	30	16	Nitrogen Metabolism
1.65e-07	22	47	98	18	Lipopolysaccharide Biosynthesis / Recycling
5.46e-07	26	31	64	12	Arginine and Proline Metabolism
4.20e-06	28	10	98	8	Lipopolysaccharide Biosynthesis / Recycling
4.52e-06	3	43	21	8	Pentose Phosphate Pathway
6.19e-06	3	43	28	9	Glycolysis/Gluconeogenesis
1.71e-05	18	103	28	12	Citric Acid Cycle
1.13e-04	1	11	52	6	Inorganic Ion Transport and Metabolism
3.12e-04	18	103	13	7	Glyoxylate Metabolism
1.72e-03	18	103	58	14	Unassigned
3.53e-03	26	31	13	4	Glutamate Metabolism
4.06e-03	14	137	255	47	Cofactor and Prosthetic Group Biosynthesis
4.82e-03	16	8	44	4	Tyrosine, Tryptophan, and Phenylalanine Metabolism
8.02e-03	18	103	22	7	Pyruvate Metabolism
1.08e-02	16	8	30	3	Valine, Leucine, and Isoleucine Metabolism
1.32e-02	16	8	68	4	tRNA Charging

Table 4.11: The single most relevant metabolite relationship for each core community found for the Barber modularity.

P value	Com num	Com size	GO size	In com	Subtype
6.60e-74	145	92	158	86	Glycerophospholipid Metabolism
1.46e-21	216	18	17	13	Murein Biosynthesis
4.68e-20	100	29	103	24	Nucleotide Salvage Pathway
3.81e-12	35	17	50	12	Oxidative Phosphorylation
2.43e-08	77	17	255	16	Cofactor and Prosthetic Group Biosynthesis
5.68e-05	80	9	103	7	Nucleotide Salvage Pathway
2.84e-04	82	15	103	8	Nucleotide Salvage Pathway
1.33e-02	224	7	223	6	Alternate Carbon Metabolism

Table 4.12: The single most relevant metabolite relationship for each core community found for the Guimera modularity.

P value	Com num	Com size	GO size	In com	Subtype
1.09e-53	95	126	158	88	Glycerophospholipid Metabolism
1.06e-32	25	58	103	42	Nucleotide Salvage Pathway
1.58e-22	21	41	35	21	Murein Recycling
1.55e-08	13	37	23	10	Histidine Metabolism
1.71e-06	105	19	223	15	Alternate Carbon Metabolism
2.26e-06	99	6	30	5	Nitrogen Metabolism
8.33e-05	15	75	255	35	Cofactor and Prosthetic Group Biosynthesis
9.69e-05	19	28	255	18	Cofactor and Prosthetic Group Biosynthesis
1.57e-04	5	12	223	10	Alternate Carbon Metabolism
2.66e-04	1	61	51	12	Purine and Pyrimidine Biosynthesis
1.69e-02	9	26	255	14	Cofactor and Prosthetic Group Biosynthesis

Table 4.13: The single most relevant metabolite relationship for each core community found for the IP modularity.

P value	Com num	Com size	GO size	In com	Subtype
5.95e-49	24	78	103	58	Nucleotide Salvage Pathway
1.93e-23	14	137	68	43	tRNA Charging
3.63e-19	22	47	35	20	Murein Recycling
4.87e-16	18	103	223	58	Alternate Carbon Metabolism
1.30e-15	3	43	223	34	Alternate Carbon Metabolism
7.13e-11	5	21	255	20	Cofactor and Prosthetic Group Biosynthesis
2.12e-09	26	31	23	10	Histidine Metabolism
4.20e-06	28	10	98	8	Lipopolysaccharide Biosynthesis / Recycling
1.13e-04	1	11	52	6	Inorganic Ion Transport and Metabolism
4.82e-03	16	8	44	4	Tyrosine, Tryptophan, and Phenylalanine Metabolism
3.38e-02	20	10	255	7	Cofactor and Prosthetic Group Biosynthesis

Chapter 5

Conclusions

This dissertation focused on developing fast, accurate, approximate methods for detecting communities in unipartite and bipartite networks by finding partitions that maximize modularity. Modularity [2] is a function that quantitatively measures how modular a particular partition is for a particular network. New algorithms and methods were proposed to find partitions that maximize modularity in unipartite and bipartite networks when the links in the network are both weighted and unweighted. Additionally, novel methods were developed to address known drawbacks to finding communities with modularity maximization. The usefulness of the algorithms and methods developed were demonstrated in applications to real-world biological networks. In this final chapter of the dissertation these methods, applications, and results are briefly summarized.

In Chapter 2, novel methods were presented to robustly detect co-regulated and functionally enriched gene communities and demonstrate their application and validity for *Escherichia coli* gene expression data. A weighted network of gene interactions

was identified from the expression data using the context likelihood of relatedness (CLR) method [88]. The network of weighted interactions was converted to an unweighted network by choosing a threshold value. Any weighted link above the chosen threshold value was converted to an unweighted link and any weighted link below the chosen threshold value was removed. A recently developed community detection algorithm [9] was run multiple times for threshold values below, at, and above the critical threshold value for the which the unweighted network was no longer fully connected. The ensemble of community detection results were analyzed to find those genes that were always found together in the same community, defined as a core community. Additionally, studying how the ensemble of communities changed as the threshold value increased allowed the identification and study of the network hierarchy. The core communities found at each threshold value significantly enriched for gene ontology (GO) terms [235, 236, 237], consistent with them representing biologically meaningful groups. Further, analysis of the most significantly enriched communities identified several candidate new regulatory interactions. The robustness of our methods was demonstrated by showing that a core set of functional communities is reliably found when artificial noise, modeling experimental noise, is added to the data. It was found that noise mainly acts conservatively, increasing the relatedness required for a network link to be reliably assigned and decreasing the size of the core communities, rather than causing association of genes into new communities.

In Chapter 3, the problem of community detection in bipartite networks when the links are both weighted and unweighted links was studied. Bipartite networks are uniquely structured networks that contain two types of nodes and links connecting pairs of nodes that consist of one node of each type. Two methods of analysis exist

to find the community structure of bipartite networks, biclustering and single mode projection, with modularity functions commensurate to each method. The method of studying bipartite networks by biclustering, partitions both types of nodes into the same community. This has the advantage that all of the information from the bipartite network is incorporated into the community structure. The two sets of nodes are therefore constrained to contain the same number of communities. The method of single mode projection studies each type of node in the bipartite network separately by creating a unipartite projection. This has the advantage of showing directly the relationships between nodes of one type and allows each type of node to be partitioned into a different number of communities. However, some amount of information about the bipartite structure of the network is always lost in the projection. This chapter introduced a new definition of bipartite modularity called the Information Preserving (IP) modularity that uses the method of unipartite projection. The IP modularity, unlike any other known modularity, can be used to partition weight bipartite networks using a single mode projection. The IP modularity was compared to two existing definitions of modularity, the Barber modularity [176] and Guimera modularity [175], that use the method of biclustering and unipartite projection, respectively. The IP modularity improves on the current methods of unipartite projection in its ability to find communities when the weighted links in the bipartite network are weighted and in preserving more of the bipartite information in its projection. Then the leading-eigenvalue method [4, 151] with final tuning [9] was adapted to find the partition that maximizes the modularity using any of the bipartite modularity definitions described above. An additional tuning step, that agglomerates communities was added to the algorithm that extends and improves

the performance of the leading-eigenvalue method of modularity maximization, allowing networks with, at least, tens of thousands of nodes to be analyzed. Finally, the expected value and standard deviation of IP modularity for random Erdös-Rényi networks was analytically derived so that a Z-score comparison of the modularity values of real-world networks can be computed. This allows a better measure of the statistical significance of the partition.

Finally, in Chapter 4, finding community structure by maximizing Information Preserving modularity is tested and validated in two ways. First, an ensemble of model bipartite networks, whose partition is *a priori* defined, is used to test the IP modularity's ability to recover the known partition as noise is added to the network. For these model bipartite networks whose links are unweighted, the IP modularity outperforms the Guimera modularity usually recovering more of the known partition when either side of the network is clustered. Additionally, the IP modularity is tested when the links in the model bipartite network are weighted and compared to results from maximizing the Barber modularity. The IP modularity recovers a large amount of the known partition as noise is added to the bipartite networks but slightly less than the Barber modularity. However, this is reasonable as the Barber modularity does not create a projection and seeks to bicluster nodes into what are precisely the type of communities defined in the model. Furthermore, this analysis validates the IP modularity's ability to extend the method of detecting communities by creating a unipartite projection from a bipartite network to weighted bipartite networks. Finally, all three modularities are applied to detect communities in the bipartite metabolic network of *Escherichia coli*. For each modularity method, an ensemble of network partitions is analyzed and the core communities are tested for biological enrichment. The IP modularity finds the largest number of biologically significant

enrichments with a large number of enzyme core communities that significantly enrich for gene ontology (GO) terms and metabolite core communities that significantly enrich for Metabolic reaction classification.

The novel methods developed in this dissertation, most notably, the ensemble approach to detecting modular and hierarchical structure in networks and the development of the IP modularity to find communities in weighted bipartite networks, will significantly enhance the science of uncovering the structure in networks. Furthermore, algorithmic advances presented in this dissertation, will allow the methods to be used on much larger networks and datasets, extending the applicability of our methods to a larger class of real world problems. We have demonstrated their applicability to genetic and metabolic networks of *Escherichia coli* but they will also be useful for studying a wide range of biological, medical, social, and physical networks.

Bibliography

[1] Maurice Kraitchik. *Mathematical Recreations*. M.W. Norton, 1942.

[2] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, Feb 2004.

[3] Lada A Adamic and Natalie Glance. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, pages 36–43. ACM, 2005.

[4] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[5] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[6] Benjamin H. Good, Yves-Alexandre de Montjoye, and Aaron Clauset. Performance of modularity maximization in practical contexts. *Phys. Rev. E*, 81:046106, Apr 2010.

[7] JJ Lemke, P Sanchez-Vazquez, HL Burgos, G Hedberg, W Ross, and RL Gourse. Direct regulation of *Escherichia coli* ribosomal protein promoters by the transcription factors ppGpp and DskA. *Proc Natl Acad Sci USA*, 108:5712–5717, 2011.

[8] Matthew F. Traxler, Sean M. Summers, Huyen-Tran Nguyen, Vineetha M. Zacharia, G. Aaron Hightower, Joel T. Smith, and Tyrrell Conway. The global, ppGpp-mediated stringent response to amino acid starvation in *Escherichia coli*. *Mol Microbiol*, 68(5):1128–1148, 2008.

[9] Y. Sun, B. Danila, K. Josi, and K. E. Bassler. Improved community structure detection using a modified fine-tuning strategy. *EPL (Europhysics Letters)*, 86(2):28004, 2009.

[10] M.E.J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[11] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4):175–308, 2006.

[12] R. Albert and A.L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47, 2002.

[13] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[14] Gerald L Alexanderson. About the cover: Euler and konigsberg's bridges: A historical view. *Bulletin of the American Mathematical Society*, 43(4):567, 2006.

[15] Béla Bollobás. *Modern Graph Theory*. Springer Verlag, 1998.

[16] Reinhard Diestel. Graph theory. *Graduate Texts in Mathematics*, 173:3, 2005.

[17] Paul Erdös and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5:17–61, 1960.

[18] P ERDÖS and A RÉNYI. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[19] Paul Erdös and Alfred Rényi. On the strength of connectedness of a random graph. *Acta Mathematica Hungarica*, 12(1):261–267, 1961.

[20] Albert-László Barabási and Eric Bonabeau. Scale-free networks. *Scientific American*, pages 60–69, May 2003.

[21] D J Watts and S H Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.

[22] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[23] Fan RK Chung and Linyuan Lu. *Complex Graphs and Networks*. Number 107. Amer Mathematical Society, 2006.

[24] Guido Caldarelli and Alessandro Vespignani. Large scale structure and dynamics of complex networks: From information technology to finance and natural science (complex systems and interdisciplinary science). 2007.

[25] Stefan Bornholdt, Heinz Georg Schuster, and John Wiley. *Handbook of Graphs and Networks*, volume 2. Wiley Online Library, 2003.

[26] Mark Newman. *Networks: an Introduction*. Oxford University Press, Inc., 2010.

[27] Derek J. de Solla Price. Networks of scientific papers. *Science*, 149(3683):510–515, 1965.

[28] S. Redner. How popular is your paper? an empirical study of the citation distribution. *The European Physical Journal B - Condensed Matter and Complex Systems*, 4:131–134, 1998.

[29] Per O. Seglen. The skewness of science. *Journal of the American Society for Information Science*, 43(9):628–638, 1992.

[30] C.I. Del Genio, T. Gross, and K.E. Bassler. All scale-free networks are sparse. *Physical Review Letters*, 107(17):178701, 2011.

[31] Albert Daz-Guilera, Jordi Duch, Alex Arenas, and Leon Danon. *Community Structure Identification*, chapter 6, pages 93–114.

[32] Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[33] Mason A Porter, Jukka-Pekka Onnela, and Peter J Mucha. Communities in networks. *Notices of the AMS*, 56(9):1082–1097, 2009.

[34] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[35] Gary William Flake, Steve Lawrence, C Lee Giles, and Frans M Coetzee. Self-organization and identification of web communities. *Computer*, 35(3):66–70, 2002.

[36] Kasper Astrup Eriksen, Ingve Simonsen, Sergei Maslov, and Kim Sneppen. Modularity and extreme edges of the internet. *Physical Review Letters*, 90(14):148701, 2003.

[37] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[38] David Lusseau and Mark EJ Newman. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(Suppl 6):S477–S481, 2004.

[39] Stuart L Pimm. The structure of food webs. *Theoretical Population Biology*, 16(2):144–158, 1979.

[40] Ann E Krause, Kenneth A Frank, Doran M Mason, Robert E Ulanowicz, and William W Taylor. Compartments revealed in food-web structure. *Nature*, 426:282–285, 2003.

[41] Roger Guimera and Luis A Nunes Amaral. Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, 2005.

[42] Geoffrey P Garnett, James P Hughes, Roy M Anderson, Bradley P Stoner, Sevgi O Aral, William L Whittington, H HUNTER HANDSFIELD, and King K Holmes. Sexual mixing patterns of patients attending sexually transmitted diseases clinics. *Sexually Transmitted Diseases*, 23(3):248–257, 1996.

[43] M Huss, P Holme, et al. Currency and commodity metabolites: their identification and relation to the modularity of metabolic networks. *IET systems biology*, 1(5):280, 2007.

[44] Mark S Granovetter. The strength of weak ties. *American Journal of Sociology*, pages 1360–1380, 1973.

[45] Ronald S Burt. Positions in networks. *Social Forces*, 55(1):93–122, 1976.

[46] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[47] Juan G Restrepo, Edward Ott, and Brian R Hunt. Characterizing the dynamical importance of network nodes and links. *Physical Review Letters*, 97(9):94102, 2006.

[48] Alex Arenas, Albert Díaz-Guilera, and Conrad J Pérez-Vicente. Synchronization reveals topological scales in complex networks. *Physical Review Letters*, 96(11):114102, 2006.

[49] MEJ Newman. Communities, modules and large-scale structure in networks. *Nature Physics*, 8(1):25–31, 2011.

[50] Peter HA Sneath, Robert R Sokal, et al. *Numerical Taxonomy. The Principles and Practice of Numerical Classification.* W H Freeman & Co, 1973.

[51] Benjamin King. Step-wise clustering procedures. *Journal of the American Statistical Association*, 62(317):86–101, 1967.

[52] Fionn Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983.

[53] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

[54] Alex Pothen. Graph partitioning algorithms with applications to scientific computing. *ICASE LaRC Interdisciplinary Series in Science and Engineering*, 4:323–368, 1997.

[55] Earl R Barnes. An algorithm for partitioning the nodes of a graph. *SIAM Journal on Algebraic Discrete Methods*, 3(4):541–550, 1982.

[56] James MacQueen et al. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1(281-297):14, 1967.

[57] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305, 1973.

[58] Alex Pothen, Horst D Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[59] Philip E Gill, Walter Murray, and Margaret H Wright. *Practical Optimization*. Academic Press, 1981.

[60] Samuel Frederick Edwards and Phil W Anderson. Theory of spin glasses. *Journal of Physics F: Metal Physics*, 5(5):965, 1975.

[61] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[62] E. T. Bell. Exponential numbers. *Amer. Math. Monthly*, 41:411–419, 1934.

[63] Michael Sipser. *Introduction to the Theory of Computation*, volume 27. Thomson Course Technology Boston, MA, 2006.

[64] Alan M Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(2):230–265, 1936.

[65] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20(2):172–188, 2008.

[66] John Harris, Jeffry L Hirst, and Michael J Mossinghoff. *Combinatorics and Graph Theory*. Springer Verlag, 2008.

[67] Béla Bollobás. *Graph Theory*, volume 62. North Holland, 1982.

[68] Mark E J Newman. Analysis of weighted networks. *Physical Review E*, 70(5):056131, 2004.

[69] Shihua Zhang, Rui-Sheng Wang, and Xiang-Sun Zhang. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications*, 374(1):483–490, 2007.

[70] Guillermo Restrepo and José L Villaveces. From trees (dendrograms and consensus trees) to topology. *Croatica Chemica Acta*, 78(2):275–281, 2005.

[71] Hiroaki Kitano. Systems biology: a brief overview. *Science*, 295(5560):1662–1664, 2002.

[72] Frank J Bruggeman and Hans V Westerhoff. The nature of systems biology. *Trends in Microbiology*, 15(1):45–50, 2007.

[73] Jason A Papin, Jennifer L Reed, Bernhard O Palsson, et al. Hierarchical thinking in network biology: the unbiased modularization of biochemical networks. *Trends in Biochemical Sciences*, 29(12):641, 2004.

[74] Jeff Hasty, David McMillen, and James J Collins. Engineered gene circuits. *Nature*, 420(6912):224–230, 2002.

[75] Michael E Wall, William S Hlavacek, and Michael A Savageau. Design of gene circuits: lessons from bacteria. *Nature Reviews Genetics*, 5(1):34–42, 2004.

[76] Hong Li. Systems genetics in -omics era: current and future development. *Theory in Biosciences*, 132(1):1–16, 2013.

[77] Björn H Junker and Falk Schreiber. *Analysis of Biological Networks*, volume 2. Wiley-Interscience, 2008.

[78] Francis Crick et al. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[79] Sarah Leavitt and DeWitt Stetten Jr. *Deciphering the Genetic Code: Marshall Nirenberg*. Office of NIH History, 2010.

[80] David J Duggan, Michael Bittner, Yidong Chen, Paul Meltzer, and Jeffrey M Trent. Expression profiling using cDNA microarrays. *Nature Genetics*, 21:10–14, 1999.

[81] Robert J Lipshutz, Stephen PA Fodor, Thomas R Gingeras, and David J Lockhart. High density synthetic oligonucleotide arrays. *Nature Genetics*, 21:20–24, 1999.

[82] Edwin Southern, Kalim Mir, and Mikhail Shchepinov. Molecular interactions on microarrays. *Nature Genetics*, 21:5–9, 1999.

[83] Dari Shalon, Stephen J Smith, and Patrick O Brown. A DNA microarray system for analyzing complex dna samples using two-color fluorescent probe hybridization. *Genome Research*, 6(7):639–645, 1996.

[84] Thomas Tang, Nicolas François, Annie Glatigny, Nicolas Agier, Marie-Hélène Mucchielli, Lawrence Aggerbeck, and Hervé Delacroix. Expression ratio evaluation in two-colour microarray experiments is significantly improved by correcting image misalignment. *Bioinformatics*, 23(20):2686–2691, 2007.

[85] Eric H. Davidson, Jonathoan P. Rast, Paola Oliveri, Andrew Ransick, Cristina Calestani, Chiou-Hwa Yuh, and Takua Minokawa. A genomic regulatory network for development. *Science*, 295:1669–1678, 2002.

[86] E Segal, M Shapira, A Regev, D Pe'er, D Botstein, D Koller, and N Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet*, 34:166–176, 2003.

[87] N Friedman and D Pe'er M Linial, I Nachman. Using Bayesian networks to analyze expression data. *J Comp Biol*, 7:601, 2000.

[88] Jeremiah J. Faith, Boris Hayete, Joshua T. Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J. Collins, and Timothy S. Gardner. Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles. *PLoS Biol*, 5:e8, 2007.

[89] N Friedman. Inferring cellular networks using probabilistic graphical models. *Science*, 303:799–805, 2004.

[90] T S Gardner, D Bernardo, D Lorenz, and P Mendes. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301:102–105, 2003.

[91] Diogo F T Veiga, Bhaskar Dutta, and Gábor Balázsi. Network inference and network response identification: moving genome-scale data to the next level of biological discovery. *Mol BioSys*, 6:469–480, 2010.

[92] Gad Getz, Erel Levine, and Eytan Domany. Coupled two-way clustering analysis of gene microarray data. *Proc Natl Acad Sci USA*, 97:12079–12084, 2000.

[93] Daniel Marbach, Robert J. Prill, Thomas Schaffter, Claudio Mattiussi, Dario Floreano, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proc Natl Acad Sci USA*, 107:6286–6291, 2010.

[94] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell*. Garland Science, 2007.

[95] Neil A Campbell and Jane B Reece. *Biology, (2008).*, volume 98. Benjamin Cumming's Publishing Company., 1984.

[96] Adam M Feist, Markus J Herrgård, Ines Thiele, Jennie L Reed, and Bernhard Ø Palsson. Reconstruction of biochemical networks in microorganisms. *Nature Reviews Microbiology*, 7(2):129–143, 2008.

[97] Jennifer L Reed, Iman Famili, Ines Thiele, and Bernhard O Palsson. Towards multidimensional genome annotation. *Nature Reviews Genetics*, 7(2):130–141, 2006.

[98] Ingrid M Keseler, Julio Collado-Vides, Alberto Santos-Zavaleta, Martin Peralta-Gil, Socorro Gama-Castro, Luis Muñiz-Rascado, César Bonavides-Martinez, Suzanne Paley, Markus Krummenacker, Tomer Altman, et al. Ecocyc: a comprehensive database of escherichia coli biology. *Nucleic Acids Research*, 39(suppl 1):D583–D590, 2011.

[99] J Michael Cherry, Eurie L Hong, Craig Amundsen, Rama Balakrishnan, Gail Binkley, Esther T Chan, Karen R Christie, Maria C Costanzo, Selina S Dwight, Stacia R Engel, et al. Saccharomyces genome database: the genomics resource of budding yeast. *Nucleic Acids Research*, 40(D1):D700–D705, 2012.

[100] U Güldener, M Münsterkötter, G Kastenmüller, N Strack, Jacques van Helden, Cl Lemer, J Richelles, SJ Wodak, J Garcia-Martinez, JE Perez-Ortin, et al. Cygd: the comprehensive yeast genome database. *Nucleic Acids Research*, 33(suppl 1):D364–D368, 2005.

[101] Minoru Kanehisa, Susumu Goto, Yoko Sato, Miho Furumichi, and Mao Tanabe. Kegg for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Research*, 40(D1):D109–D114, 2012.

[102] Ron Caspi, Tomer Altman, Kate Dreher, Carol A Fulcher, Pallavi Subhraveti, Ingrid M Keseler, Anamika Kothari, Markus Krummenacker, Mario Latendresse, Lukas A Mueller, et al. The metacyc database of metabolic pathways and enzymes and the biocyc collection of pathway/genome databases. *Nucleic Acids Research*, 40(D1):D742–D753, 2012.

[103] Qinghu Ren, Kaixi Chen, and Ian T Paulsen. Transportdb: a comprehensive database resource for cytoplasmic membrane transport systems and outer membrane channels. *Nucleic Acids Research*, 35(suppl 1):D274–D279, 2007.

[104] Jeffrey D Orth, Ines Thiele, and Bernhard Ø Palsson. What is flux balance analysis? *Nature Biotechnology*, 28(3):245–248, 2010.

[105] Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltan N Oltvai, and A-L Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[106] Andreas Wagner and David A Fell. The small world inside large metabolic networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1478):1803–1810, 2001.

[107] David E Featherstone and Kendal Broadie. Wrestling with pleiotropy: genomic and topological analysis of the yeast gene expression network. *Bioessays*, 24(3):267–274, 2002.

[108] Himanshu Agrawal. Extreme self-organization in networks constructed from gene expression data. *Physical Review Letters*, 89(26):268702, 2002.

[109] Hawoong Jeong, Sean P Mason, A-L Barabási, and Zoltan N Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.

[110] Steven A Benner, Andrew D Ellington, and Andreas Tauer. Modern metabolism as a palimpsest of the rna world. *Proceedings of the National Academy of Sciences*, 86(18):7054–7058, 1989.

[111] Ralf Mrowka, Andreas Patzak, Hanspeter Herzel, and Dirk Holste. Sequence-related human proteins cluster by degree of evolutionary conservation. *Phys. Rev. E*, 70:051908, Nov 2004.

[112] Eli Eisenberg and Erez Y Levanon. Preferential attachment in the protein network evolution. *Physical Review Letters*, 91(13):138701, 2003.

[113] J. Kim, P. L. Krapivsky, B. Kahng, and S. Redner. Infinite-order percolation and giant fluctuations in a protein interaction network. *Phys. Rev. E*, 66:055101, Nov 2002.

[114] Alexei Vázquez, Alessandro Flammini, Amos Maritan, and Alessandro Vespignani. Modeling of protein interaction networks. *Complexus*, 1(1):38–44, 2002.

[115] Andrey Rzhetsky and Shawn M Gomez. Birth of scale-free molecular networks and the number of distinct DNA and protein domains per genome. *Bioinformatics*, 17(10):988–996, 2001.

[116] Ashish Bhan, David J Galas, and T Gregory Dewey. A duplication growth model of gene expression networks. *Bioinformatics*, 18(11):1486–1493, 2002.

[117] Andreas Wagner. How the global structure of protein interaction networks evolves. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 270(1514):457–466, 2003.

[118] Romualdo Pastor-Satorras, Eric Smith, Ricard V Solé, et al. Evolving protein interaction networks through gene duplication. *Journal of Theoretical Biology*, 222(2):199–210, 2003.

[119] Zoltán Toroczkai and Kevin E Bassler. Network dynamics: Jamming is limited in scale-free systems. *Nature*, 428(6984):716–716, 2004.

[120] Zoltan Toroczkai, Balázs Kozma, Kevin E Bassler, NW Hengartner, and G Korniss. Gradient networks. *Journal of Physics A: Mathematical and Theoretical*, 41(15):155103, 2008.

[121] Leland H Hartwell, John J Hopfield, Stanislas Leibler, and Andrew W Murray. From molecular to modular cell biology. *Nature*, 402:C47–C52, 1999.

[122] Uri Alon. Biological networks: the tinkerer as an engineer. *Science*, 301(5641):1866–1867, 2003.

[123] Jose B Pereira-Leal, Anton J Enright, and Christos A Ouzounis. Detection of functional modules from protein interaction networks. *PROTEINS: Structure, Function, and Bioinformatics*, 54(1):49–57, 2004.

[124] Alexander W Rives and Timothy Galitski. Modular organization of cellular networks. *Proceedings of the National Academy of Sciences*, 100(3):1128–1133, 2003.

[125] Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166–176, 2003.

[126] Jan Ihmels, Gilgi Friedlander, Sven Bergmann, Ofer Sarig, Yaniv Ziv, and Naama Barkai. Revealing modular organization in the yeast transcriptional network. *Nature genetics*, 31(4):370–377, 2002.

[127] Erzsébet Ravasz, Anna Lisa Somera, Dale A Mongru, Zoltán N Oltvai, and A-L Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, 2002.

[128] Christine Vogel, Sarah A Teichmann, and Jose Pereira-Leal. The relationship between domain duplication and recombination. *Journal of Molecular Biology*, 346(1):355–365, 2005.

[129] Jose B Pereira-Leal, Emmanuel D Levy, and Sarah A Teichmann. The origins and evolution of functional modules: lessons from protein complexes. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 361(1467):507–517, 2006.

[130] Loic Giot, Joel S Bader, C Brouwer, Amitabha Chaudhuri, Bing Kuang, Y Li, YL Hao, CE Ooi, Brian Godwin, E Vitols, et al. A protein interaction map of drosophila melanogaster. *Science*, 302(5651):1727–1736, 2003.

[131] Soon-Hyung Yook, Zoltán N Oltvai, and Albert-László Barabási. Functional and topological characterization of protein interaction networks. *Proteomics*, 4(4):928–942, 2004.

[132] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*, volume 8. Cambridge University Press, 1994.

[133] John Scott. *Social Network Analysis*. SAGE Publications Limited, 2012.

[134] Wayne W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):pp. 452–473, 1977.

[135] Andre Broido et al. Internet topology: Connectivity of ip graphs. In *ITCom 2001: International Symposium on the Convergence of IT and Communications*, pages 172–187. International Society for Optics and Photonics, 2001.

[136] Qian Chen, Hyunseok Chang, Ramesh Govindan, and Sugih Jamin. The origin of power laws in internet topologies revisited. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 608–617. IEEE, 2002.

[137] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM Computer Communication Review*, volume 29, pages 251–262. ACM, 1999.

[138] Albert Reka, Hawoong Jeong, and Albert-Laszlo Barabasi. Diameter of the world wide web. *Nature*, 401(9):130–131, 1999.

[139] Albert-László Barabási, Réka Albert, and Hawoong Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69–77, 2000.

[140] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. *Computer Networks*, 33(1):309–320, 2000.

[141] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, Dandapani Sivakumar, Andrew Tompkins, and Eli Upfal. The web as a graph. *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 1–10, 2000.

[142] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999.

[143] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.

[144] James C Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, 1981.

[145] Jörg Reichardt and Stefan Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Phys. Rev. Lett.*, 93:218701, Nov 2004.

[146] Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, pages 58–64, 2000.

[147] Robin Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[148] Daniel Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977.

[149] B W Kernighan and S Lin. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal*, 49:291, 1970.

[150] Stuart Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[151] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, Sep 2006.

[152] Jussi M Kumpula, Mikko Kivelä, Kimmo Kaski, and Jari Saramäki. Sequential algorithm for fast clique percolation. *Physical Review E*, 78(2):026109, 2008.

[153] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E*, 76:036106, Sep 2007.

[154] Gergely Tibély and János Kertész. On the equivalence of the label propagation method of community detection and a Potts model approach. *Physica A: Statistical Mechanics and its Applications*, 387(19):4982–4984, 2008.

[155] Michael J Barber and John W Clark. Detecting network communities by propagating labels under constraints. *Physical Review E*, 80(2):026129, 2009.

[156] Ian XY Leung, Pan Hui, Pietro Liò, and Jon Crowcroft. Towards real-time community detection in large networks. *Physical Review E*, 79(6):066107, 2009.

[157] MB Hastings. Community detection as an inference problem. *Physical Review E*, 74(3 Pt 2):035102.

[158] Mark EJ Newman and Elizabeth A Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences*, 104(23):9564–9569, 2007.

[159] Hugo Zanghi, Christophe Ambroise, and Vincent Miele. Fast online graph clustering via erdős–rényi mixture. *Pattern Recognition*, 41(12):3592–3599, 2008.

[160] Jake M Hofman and Chris H Wiggins. Bayesian approach to network modularity. *Physical Review Letters*, 100(25):258701, 2008.

[161] Brian Ball, Brian Karrer, and MEJ Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3):036103, 2011.

[162] Lars Seemann, Jason Shulman, and Gemunu H Gunaratne. A robust topology-based algorithm for gene expression profiling. *ISRN Bioinformatics*, 2012:381023, 2012.

[163] Danijela Horak, Slobodan Maletić, and Milan Rajković. Persistent homology of complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03034, 2009.

[164] Mark EJ Newman. Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality. *Physical Review E*, 64(1):016132, 2001.

[165] A. Davis, B. B. Gardner, and M. R. Gardner. *Deep South.* University of Chicago Press, Chicago, 1970.

[166] Patrick Doreian, Vladimir Batagelj, and Anuška Ferligoj. Generalized blockmodeling of two-mode network data. *Social Networks*, 26(1):29–53, 2004.

[167] Mark EJ Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[168] Katy Börner, Jeegar T Maru, and Robert L Goldstone. The simultaneous evolution of author and paper networks. *Proceedings of the National Academy of Sciences*, 101(Suppl 1):5266–5273, 2004.

[169] Michael J Barber, Margarida Faria, Ludwig Streit, and Oleg Strogan. Searching for communities in bipartite networks. *arXiv preprint arXiv:0803.2854*, 2008.

[170] Roger Guimera, Brian Uzzi, Jarrett Spiro, and Luis A Nunes Amaral. Team assembly mechanisms determine collaboration network structure and team performance. *Science*, 308(5722):697–702, 2005.

[171] Albert-László Barabási and Zoltán N. Oltvai. Network biology: understanding the cell's functional organization. *Nat Rev Genet*, 5:101, 2004.

[172] Sergei Maslov and Kim Sneppen. Specificity and stability in topology of protein networks. *Science Signalling*, 296(5569):910, 2002.

[173] Jordi Bascompte and Pedro Jordano. Plant-animal mutualistic networks: the architecture of biodiversity. *Annu. Rev. Ecol. Evol. Syst.*, 38:567–593, 2007.

[174] Pablo M Gleiser. How to become a superhero. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(09):P09020, 2007.

[175] Roger Guimerà, Marta Sales-Pardo, and Luís A. Nunes Amaral. Module identification in bipartite and directed networks. *Phys. Rev. E*, 76:036102, Sep 2007.

[176] Michael J. Barber. Modularity and community detection in bipartite networks. *Phys. Rev. E*, 76:066102, Dec 2007.

[177] Atsushi Miyauchi and Noriyoshi Sukegawa. Maximizing Barber's bipartite modularity is also hard. *arXiv preprint arXiv:1310.4656*, 2013.

[178] Marina Meilă. Comparing clusterings by the variation of information. In *Learning Theory and Kernel Machines*, pages 173–187. Springer, 2003.

[179] John John Aldo Lee and Michel Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

[180] Anne Condon and Richard M Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms*, 18(2):116–140, 2001.

[181] Roger Guimera, Marta Sales-Pardo, and Luís A Nunes Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101, 2004.

[182] Jörg Reichardt and Stefan Bornholdt. Partitioning and modularity of graphs with arbitrary degree distribution. *Physical Review E*, 76(1):015102, 2007.

[183] Yaotian Fu and Philip W Anderson. Application of statistical mechanics to np-complete problems in combinatorial optimisation. *Journal of Physics A: Mathematical and General*, 19(9):1605, 1986.

[184] I Kanter and H Sompolinsky. Graph optimisation problems and the potts glass. *Journal of Physics A: Mathematical and General*, 20(11):L673, 1987.

[185] S F Edwards and P W Anderson. Theory of spin glasses. *Journal of Physics F: Metal Physics*, 5(5):965, 1975.

[186] Jack Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Routledge Academic, 1988.

[187] Santo Fortunato and Marc Barthlemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[188] A Arenas, A Fernndez, and S Gmez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[189] Jonathan W. Berry, Bruce Hendrickson, Randall A. LaViolette, and Cynthia A. Phillips. Tolerating the community detection resolution limit with edge weighting. *Phys. Rev. E*, 83:056119, May 2011.

[190] Vincent Granville, Mirko Krivánek, and J-P Rasson. Simulated annealing: A proof of convergence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(6):652–656, 1994.

[191] Mark E J Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.

[192] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[193] Leon Danon, Albert Díaz-Guilera, and Alex Arenas. The effect of size heterogeneity on community identification in complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(11):P11010, 2006.

[194] Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks. *arXiv preprint cs/0702048*, 2007.

[195] Josep M Pujol, Javier Béjar, and Jordi Delgado. Clustering algorithm for determining community structure in large networks. *Physical Review E*, 74(1):016107, 2006.

[196] Haifeng Du, Marcus W Feldman, Shuzhuo Li, and Xiaoyi Jin. An algorithm for detecting community structure of social networks based on prior knowledge and modularity. *Complexity*, 12(3):53–60, 2007.

[197] Biao Xiang, En-Hong Chen, and Tao Zhou. Finding community structure based on subgraph similarity. *Complex Networks*, 207:73–81, 2009.

[198] Andreas Noack and Randolf Rotta. Multi-level algorithms for modularity clustering. *Experimental Algorithms*, 5526:257–268, 2009.

[199] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104, 2005.

[200] David Poole. *Linear Algebra: A Modern Introduction*. Thomson Brooks/Cole, 2010.

[201] Stanislav Sobolevsky, Riccardo Campari, Alexander Belyi, and Carlo Ratti. A general optimization technique for high quality community detection in complex networks. *arXiv preprint arXiv:1308.3508*, 2013.

[202] Santiago Treviño, Yudong Sun, Tim F Cooper, and Kevin E Bassler. Robust detection of hierarchical communities from *Escherichia coli* gene expression data. *PLoS computational biology*, 8(2):e1002391, 2012.

[203] Gábor Balázsi, A P Heath, L Shi, and M L Gennaro. The temporal response of the *Mycobacterium tuberculosis* gene regulatory network during growth arrest. *Mol Sys Biol*, 4:225, 2008.

[204] H-W Ma, J Buer, and A-P Zeng. Hierarchical structure and modules in the *Escherichia coli* transcriptional regulatory network revealed by a new top-down approach. *BMC Bioinformatics*, 5:199, 2004.

[205] Tim F Cooper, S K Remold, R E Lenski, and D Schneider. Expression profiles reveal parallel evolution of epistatic interactions involving the CRP regulon in *Escherichia coli*. *PLoS Genetics*, 4(2):e35, 2008.

[206] Z Shi, CK Derrow, and B Zhang. Co-expression module analysis reveals biological processes, genomic gain, and regulatory mechanisms associated with breast cancer progression. *BMC Sys Biol*, 4:74, 2010.

[207] E Bonnet, M Tatari, A Michoel, K Marchal, G Berx, and Y Van de Peer. Module network inference from a cancer gene expression data set identifies microRNA regulated modules. *PLoS One*, 5:e10162, 2010.

[208] A Beyer, C Workman, J Hollunder, D Radke, U Moller, T Wilhelm, and T Ideker. Integrated assessment and predication of transcription factor binding. *PLoS Comp Biol*, 2:e70, 2006.

[209] E Ravasz, A L Somera, D A Mongru, Z N Oltvai, and A. L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297:1551–1555, 2002.

[210] Santo Fortunato. Community detection in graphs. *Phys Rep*, 486:75–174, 2010.

[211] X Wen, S Fuhrman, G S. Michaels, D B. Carr, S Smith, and et al. Large-scale temporal gene expression mapping of central nervous system development. *Proc Natl Acad Sci USA*, 95:334, 1998.

[212] M B Eisen, P T Spellman, P O Brown, and D Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci USA*, 95:14863, 1998.

[213] J N Weinstein, T G Myers, P M O'Connor, S H Friend, A J Fornace, and et al. An information-intensive approach to the molecular pharmacology of cancer. *Science*, 275:343, 1997.

[214] S. B. Seidman. Network structure and minimum degree. *Soc. Netw.*, 5:269–287, 1983.

[215] S. B. Seidman and B. L. Foster. A graph theoretic generalization of the clique concept. *J. Math. Sociol.*, 6:139–154, 1978.

[216] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.

[217] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446:664–667, 2007.

[218] Feng Luo, Bo Li, Xiu-Feng Wan, and Richard Scheuermann. Core and periphery structures in protein interaction networks. *BMC Bioinformatics*, 10(Suppl 4):S8, 2009.

[219] S. P. Borgatii and M. G. Everett. Models of core/periphery structures. *Social Networks*, 21:375–395, 1999.

[220] S. P. Borgatii and M. G. Everett. Peripheries of cohesive subsets. *Social Networks*, 21:397–407, 1999.

[221] S Tavazoie, J D Hughes, M J Campbell, R J Cho, and G M Church. Systematic determination of genetic network architecture. *Nat Genet*, 22:281, 1999.

[222] S Raychaudhuri, J M Stuart, and R B Altman. Principal components analysis to summarize microarray experiments: application to sporulation time series. *Pacific Symposium on Biocomputing*, page 455, 2000.

[223] O Alter, P Brown, and D Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97:10101, 2000.

[224] Qiming Lu, G. Korniss, and Boleslaw Szymanski. The naming game in social networks: community formation and consensus engineering. *Journal of Economic Interaction and Coordination*, 4(2):221–235, November 2009.

[225] Marcelo Blatt, Shai Wiseman, and Eytan Domany. Superparamagnetic clustering of data. *Phys. Rev. Lett.*, 76:3251–3254, Apr 1996.

[226] Roger Guimerà, Marta Sales-Pardo, and Luís A. Nunes Amaral. Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E*, 70:025101, Aug 2004.

[227] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72:027104, Aug 2005.

[228] JJ Chen, H-M Hsueh, RR Delongchamp, C-J Lin, and C-A Tsa. Reproducibility of microarray data: a further analysis of microarray quality control (MAQC) data. *BMC Bioinformatics*, 8:412, 2007.

[229] KA Baggerly, KR Coombes, and ES Neeley. Run batch effects potentially compromise the usefulness of genomic signatures for ovarian cancer. *J Clin Oncology*, 26:1186–1187, 2008.

[230] DL Duewer, WD Jones, LH Reid, and M Salit. Learning from microarray interlaboratory studies: measures of precision for gene expression. *BMC Genet*, 10:153, 2009.

[231] J J Faith, M E Driscoll, V A Fusaro, E J Cosgrove, B Hayete, and et al. Many microbe microarrays database: uniformly normalized affymetrix compendia with structured experimental metadata. *Nucleic Acids Res*, 36:D866, 2008.

[232] H Zare, D Sangurdekar, P Srivastava, M Kaveh, and A Khodursky. Reconstruction of *Escherichia coli* transcriptional regulatory networks via regulon-based associations. *Science*, 297:39, 2009.

[233] Ka Yee Yeung, Kenneth M. Dombek, Kenneth Lo, John E. Mittler, Jun Zhu, Eric E. Schadt, Roger E. Bumgarner, and Adrian E. Raftery. Construction of regulatory networks using expression time-series data of a genotyped population. *Proceedings of the National Academy of Sciences*, 108(48):19436–19441, 2011.

[234] Carsten O Daub, Ralf Steuer, Joachim Selbig, and Sebastian Kloska. Estimating mutual information using B-spline functions - an improved similarity measure for analyzing gene expression data. *BMC Bioinformatics*, 5:118, 2004.

[235] GO-Consortium. Gene Ontology: tool for the unification of biology. *Nat Genet*, 25:25–29, 2000.

[236] Peter D. Karp, Monica Riley, Milton Saier, Ian T. Paulsen, Julio Collado-Vides, Suzanne M. Paley, Alida Pellegrini-Toole, César Bonavides, and Socorro Gama-Castro. The EcoCyc database. *Nucleic Acids Res*, 30(1):56–58, 2002.

[237] J Hubble, J Demeter, H Jin, M Mao, M Nitzberg, TB Reddy, F Wymore, ZK Zachariah, G Sherlock, and CA Bell. Implementation of genepattern within the stanford microarray database. *Nucleic Acids Res*, 37(Database Issue):D898–901, 2009.

[238] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *J R Stat Soc B (Methodological)*, 57:289–300, 1995.

[239] Socorro Gama-Castro, Heladia Salgado, Martin Peralta-Gil, Alberto Santos-Zavaleta, Luis Muiz-Rascado, Hilda Solano-Lira, Vernica Jimenez-Jacinto, Verena Weiss, Jair S. Garca-Sotelo, Alejandra Lpez-Fuentes, Liliana Porrn-Sotelo, Shirley Alquicira-Hernndez, Alejandra Medina-Rivera, Irma Martnez-Flores, Kevin Alquicira-Hernndez, Ruth Martnez-Adame, Csar Bonavides-Martnez, Juan Miranda-Ros, Araceli M. Huerta, Alfredo Mendoza-Vargas, Leonardo Collado-Torres, Blanca Taboada, Leticia Vega-Alvarado, Maricela Olvera, Leticia Olvera, Ricardo Grande, Enrique Morett, and Julio Collado-Vides. RegulonDB version 7.0: transcriptional regulation of *Escherichia coli* K-12 integrated within genetic sensory response units (gensor units). *Nucleic Acids Res*, 39:D98 – 105, 2010.

[240] Monica Riley, Takashi Abe, Martha B. Arnaud, Mary K.B. Berlyn, Frederick R. Blattner, Roy R. Chaudhuri, Jeremy D. Glasner, Takashi Horiuchi, Ingrid M. Keseler, Takehide Kosuge, Hirotada Mori, Nicole T. Perna, Guy Plunkett, Kenneth E. Rudd, Margrethe H. Serres, Gavin H. Thomas, Nicholas R. Thomson, David Wishart, and Barry L. Wanner. *Escherichia coli* k-12: a cooperatively developed annotation snapshot2005. *Nucleic Acids Res*, 34(1):1–9, 2006.

[241] Melanie M. Barker, Tamas Gaal, Cathleen A. Josaitis, and Richard L. Gourse. Mechanism of regulation of transcription initiation by ppGpp. i. effects of ppGpp on transcription initiation in vivo and in vitro. *J Mol Biol*, 305(4):673 – 688, 2001.

[242] Rafael A Irizarry, Daniel Warren, Forrest Spencer, Irene F Kim, Shyam Biswal, Bryan C Frank, Edward Gabrielson, Joe G N Garcia, Joel Geoghegan, Gregory Germino, Constance Griffin, Sara C Hilmer, Eric Hoffman, Anne E Jedlicka, Ernest Kawasaki, Francisco Martinez-Murillo, Laura Morsberger, Hannah Lee, David Petersen, John Quackenbush, Alan Scott, Michael Wilson, Yanqin Yang, Shui Qing Ye, and Wayne Yu. Multiple-laboratory comparison of microarray platforms. *Nat Meth*, 2:345–349, 2005.

[243] Kwang-Il Goh, Michael E Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-Laszlo Barabasi. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, 2007.

[244] Peng Zhang, Jinliang Wang, Xiaojia Li, Menghui Li, Zengru Di, and Ying Fan. Clustering coefficient and community structure of bipartite networks. *Physica A: Statistical Mechanics and its Applications*, 387(27):6869–6875, 2008.

[245] Sune Lehmann, Martin Schwartz, and Lars Kai Hansen. Biclique communities. *Physical Review E*, 78(1):016108, 2008.

[246] Nan Du, Bai Wang, Bin Wu, and Yi Wang. Overlapping community detection in bipartite networks. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 1, pages 176–179. IEEE, 2008.

[247] J. Scott and M. Hughes. *The Anatomy of Scottish Capital: Scottish Companies and Scottish Capital, 1900-1979*. Croom Helm, London, 1980.

[248] Wilco Dekker and Ben van Raaij. *De elite. De Volkskrant Top 200 van invloedrijkste Nederlanders*. Meulenhoff, 2006.

[249] Jörg Reichardt and Stefan Bornholdt. When are networks truly modular? *Physica D: Nonlinear Phenomena*, 224(1):20–26, 2006.

[250] 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. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.

[251] Pablo M Gleiser and Leon Danon. Community structure in jazz. *Advances in Complex Systems*, 6(04):565–573, 2003.

[252] C.I. Del Genio and K.E. Bassler. Unpublished.

[253] Leon Danon, Albert Diaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008, 2005.

[254] Jeffrey D Orth, Tom M Conrad, Jessica Na, Joshua A Lerman, Hojung Nam, Adam M Feist, and Bernhard O Palsson. A comprehensive genome-scale reconstruction of *Escherichia coli* metabolism-2011. *Mol Syst Biol*, 7(535), 2011.