STATISTICAL INFERENCE ON NETWORK DATA

BY

JINGFEI ZHANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Statistics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:
Professor Yuguo Chen, Chair
Professor Jeffrey A. Douglas
Professor John I. Marden
Professor Douglas G. Simpson
Abstract

Networks arise from modeling complex systems in various fields, such as computer science, social science, biology, psychology and finance. Understanding and analyzing networks help us better understand these complex systems and extract useful information. In this dissertation, we study problems on network sampling, network modeling and data mining on networks.

Random graphs with given vertex degrees have been widely used as a model for many real-world complex networks. However, both statistical inference and analytic study of such networks present great challenges. In Chapter 2, we propose new sequential importance sampling methods for sampling networks with a given degree sequence. These samples can be used to approximate closely the null distributions of a number of test statistics involved in such networks, and provide an accurate estimate of the total number of networks with given vertex degrees. We study the asymptotic behavior of the proposed algorithm and prove that the importance weight remains bounded as the size of the graph grows. This property guarantees that the proposed sampling algorithm can still work efficiently even for large sparse graphs. We apply our method to a range of examples to demonstrate its efficiency in real problems.

One important question for complex networks is how the network's connectivity will be affected if the network is under targeted attacks, i.e., the nodes with the most links are attacked. In Chapter 3, we found that a dolphin network is resilient to targeted attacks. To further study the resilient property, we fit an exponential random graph model to the dolphin network. The fitted model characterizes network resiliency and identifies local structures that can reproduce the global resilience property. Such a statistical model can be used to build the Internet and other networks to increase the attack tolerance of those networks.
The problem of finding densely connected subgraphs in a network has attracted a lot of recent attention. Such subgraphs are sometimes referred to as communities in social networks or molecular modules in protein networks. In Chapter 4, we propose two Monte Carlo optimization algorithms for identifying the densest subgraphs with a fixed size or with size in a given range. The new algorithms combine the idea of simulated annealing and efficient moves for the Markov chain, and both algorithms are shown to converge to the set of optimal states (densest subgraphs) with probability one. When applied to a yeast protein interaction network and a stock market graph, the algorithms identify interesting new densely connected subgraphs.

One of the most relevant features of networks representing real systems is the community structure. Detecting communities is of great importance in understanding, analyzing, organizing networks. In Chapter 5, we describe a statistical framework for modularity-based network community detection. We derive the modularity function under the proposed statistical framework, and propose a fast modularity maximization algorithm based on the eigen-spectrum of the modularity matrix. A hypothesis testing procedure is developed to determine the significance of an identified community structure. The modularity formulated under the proposed statistical framework is shown to be consistent under the degree-corrected stochastic block model framework. Several synthetic networks and real world networks are used to demonstrate the effectiveness of our method.
This thesis is dedicated to my parents who have always been loving and supportive.

Also, this thesis is dedicated to my advisor Yuguo Chen who has been a great mentor since the beginning of my studies.

Finally, this thesis is dedicated to all those who believe in the richness of learning.
# Table of Contents

List of Tables .......................................................... vii
List of Figures ........................................................... viii

Chapter 1  Background .................................................. 1
  1.1 Social Networks .................................................... 2
  1.2 Biological Networks .............................................. 2
  1.3 Technology Networks ............................................ 3

Chapter 2  Sampling for Conditional Inference on Network Data .......... 6
  2.1 Introduction ....................................................... 6
  2.2 Basic Notation ................................................... 8
  2.3 Sequential Importance Sampling ................................. 8
  2.4 Sampling Random Networks ................................... 11
  2.5 Theoretical Properties ....................................... 15
  2.6 A Refined SIS Algorithm ...................................... 18
    2.6.1 Building the Candidate Set ............................... 20
    2.6.2 Sampling Procedure ..................................... 22
  2.7 Applications and Simulations ................................ 24
    2.7.1 Counting Graphs ......................................... 25
    2.7.2 Testing Exponential Random Graph Models ............. 27
  2.8 Discussion ....................................................... 32
  2.9 Proof of the Main Results .................................... 33

Chapter 3  Exponential Random Graph Models for Networks Resilient to Targeted Attacks ................................................. 43
  3.1 Introduction ...................................................... 43
  3.2 Resilience Property of the Dolphin Network ................... 44
  3.3 Exponential Random Graph Models ............................... 47
    3.3.1 Network Statistics ....................................... 48
    3.3.2 Model Fitting ............................................. 50
  3.4 Constraints on the Network Space ............................... 52
  3.5 Model Fitting for the Dolphin Network ......................... 54
    3.5.1 Goodness of Fit Test ..................................... 57
List of Tables

2.1 Performance comparison of three Monte Carlo methods for estimating the number of graphs. ................................................................. 26

3.1 Parameter estimates and their standard errors (in parentheses) for the three ERGMs. Here * means significant at the 0.05 level. ........................................ 55
## List of Figures

1.1 Social network of 62 bottlenose dolphins in Doubtful Sound, New Zealand. The three most connected nodes are in black. ................................................................. 3
1.2 Biological network of the protein-protein interaction network in budding yeast. The size of the nodes is proportional to the links incident to them. This network is drawn using k-core decomposition. ................................................................. 4
1.3 The transmission of the Western States Power Grid of the United States. .............. 5

2.1 Illustration of a simple graph with 6 nodes and its adjacency matrix. ...................... 9
2.2 Food web of 33 types of organisms in the Chesapeake Bay. ............................. 29
2.3 Approximated distribution of the number of 5-cycles based on 1000 weighted samples from SIS-CP. The vertical line indicates the observed number of 5-cycles in the real food web. ............................................... 30

3.1 The histogram of the percentage of global efficiency changes based on 1000 random samples. The vertical line indicates the value calculated from the observed dolphin network. .................................................. 46
3.2 An illustrative example of the network. .............................................................. 49
3.3 Two networks with the same edge density but different degree sequences. .......... 53
3.4 The histogram of the degree sequence of the dolphin network. .......................... 54
3.5 Goodness-of-fit plots for model I (top), Model II (middle), and Model III (bottom). In each plot, the black solid line indicates the statistics computed from the dolphin network. The grey lines indicate the range that covers 95% of the statistics computed from 100 sampled networks. The boxplot indicates the median and the interquartile range. .................................................. 58
3.6 The histograms of the percentage of global efficiency change for samples generated from Model I (top left), Model II (top right), and Model III (bottom). The vertical line indicates the percentage of global efficiency change for the dolphin network. ..... 60
3.7 Densities of the percentage of global efficiency change after 5% targeted attacks for Model II on the space of networks with the same degree sequence as the dolphin network. Five different values for the parameter $\theta_2$ in Model II are considered: $-0.5$, $-0.25$, 0, 0.25, and 0.5. .................................................. 63
3.8 Densities of the percentage of global efficiency change after 5% targeted attacks for Model II on the space of networks with the same degree sequence as the one generated from a power-law with 100 nodes and 179 edges. Five different values for the parameter $\theta_2$ in Model II are considered: $-0.5, -0.25, 0, 0.25,$ and $0.5$.

4.1 The Erdos-Renyi graph ($n = 100, p = 0.05$) with an embedded size 10 clique. The nodes in the embedded clique are in black.

4.2 The trace plot of Algorithm 1 for finding the densest subgraph of size 10.

4.3 Trace plots of Algorithm 4.2.1 (left) and Spirin-Mirny algorithm (right) for finding the densest subgraph of size 14.

4.4 Trace plots of Algorithm 4.2.1 for finding the densest subgraph of size 35.

4.5 Histogram of the $Q$ value of size 35 subgraphs in random networks.

4.6 Trace plots of the average degree (left) and the size of the subgraph (right) for finding subgraphs ($5 \leq k \leq 50$) with the highest average degree using Algorithm 4.3.1.

4.7 Scatter plot of the average degree versus the size of the subgraph.

4.8 Trace plots of Algorithm 4.2.1 for finding the clique of size 82.

4.9 Trace plots of the average degree (left) and the size of the subgraph (right) for finding subgraphs ($100 \leq k \leq 350$) with the highest average degree using Algorithm 4.3.1.

5.1 Histogram of the maximized $\max_{me} Q_{NG}(me)$ for the 100 Erdos-Renyi graphs.

5.2 (Left) Histogram of the 100 $p_1$-values; (Right) Histogram of the 100 $p_2$-values.

5.3 Histogram of the maximized $\max_{me} Q_{NC}(me)$ for 100 graphs generated from stochastic block model with community structure.

5.4 Histogram of the maximized $Q_{MCMC}$ for the 1000 MCMC samples.

5.5 Krebs’ network of books on American politics. Liberal books are the square nodes, conservative books are the triangle nodes and neutral books are the circle nodes.
Chapter 1

Background

The study of networks can be traced back to as early as the seventeenth century when Euler laid the foundation of graph theory by proving that there is no solution to the historical problem “Seven Bridges of Königsberg” (Chartrand, 1984). Network, in its various forms, has emerged from numerous disciplines with the rapid development of technology in the new century. With the improvement of transportation system, the invention of Internet social networking platforms and the development of communication technology, it is not exaggerating to say the world is “shrinking” due to the increasing connectedness of human beings. The famous six degree separation theory (Watts, 2004) states that on average, everyone is approximately six steps away from any other person in the world. It is surprising that everyone in the world are expected to be linked through such a short chain of friendships. With the “shrinking” of our world over the past decade, there has been an explosion of interest in the study of network data across the sciences. The depth and the sophistication of network data analysis have been raised to a whole new level.

The definition of networks varies from discipline to discipline. The definition of the word “network” in Oxford English Dictionary is “a group or system of interconnected people or things”. This is a rather general concept but it emphasizes that the key concept in a network is the connectedness between its subjects. In particular, the word “network” is often used exchangeably with “graph” in mathematics. In this dissertation, we will continue with this tradition of using “network” and “graph” exchangeably. To better understand the statistical methodologies in the following chapters, some contexts from which network data arises would be helpful. In the following sections, we will discuss some of the well known research fields that give rise to network data.
1.1 Social Networks

Social networks are network structures that emerge from social studies. They have been studied extensively by researchers in sociology, psychology and anthropology. In recent years, researchers from other fields, such as engineering, business and public health also share interests in social networks. A social network usually consists of actors (such as individuals, groups or organizations) and the ties between them. A tie between a pair of actors indicates social contact (interaction) between the two units. Social networks are mostly used in social sciences to study the relationships between individuals, groups or organizations. For example, some interesting questions are “Are there communities in a social network?” “Are the interactions between actors mutual?” “What are the important factors in establishing friendships?” or “Who are the key individuals in the network?”. An example of a social network is shown in Figure 1.1. This network consists of 62 bottlenose dolphins in Doubtful Sound, New Zealand (Lusseau, 2003). Each node in Figure 1.1 represents a dolphin. Two dolphins are linked through an edge if they are seen together more often than expected. More properties of this social network will be discussed in Chapter 2 and Chapter 3.

1.2 Biological Networks

Biological networks are networks that come from biological systems. Complex biological systems can be represented and studied as networks. For example, neural systems can be modeled as networks of interactive neurons. Proteins can be represented as networks of amino acids and amino acids can be studied as networks of atoms. With the fast development in genetic studies, more and more questions are posed for large scale complex biological networks. Figure 1.2 is an protein-protein interaction network in budding yeast (Bu et al., 2003). It is plotted with large networks visualization software LaNet-vi using k-core decomposition (http://lanet-vi.soic.indiana.edu/). Each node in the graph represents a protein and a link between two protein indicates they interact with each other. This large network involves 2617 proteins and 11855 protein-protein interactions. For this biological network, some interesting research questions are “Are there densely connected subgroups
of proteins?” or “What are some patterns that appear frequently in the network?” If we can identify a densely connected subgroup, these proteins in the subgroup could potentially be one protein complex that delivers a particular biological function. If some simple patterns repeat more often than expected, these patterns could work as building blocks for this network. The architecture of such building blocks could reveal alignment rules of the proteins. More properties of this network will be discussed in Chapter 4.

1.3 Technology Networks

Technology networks is not a concept that is unfamiliar to us. Internet is a most well known example of technology networks. Typically, technology networks involve communication networks (Internet, WWW, telephone networks, cable networks, etc.), transportation networks and energy networks (electricity, gas, etc.). For example, Internet could be represented as a network where the routers are the nodes and the connection between routers are the edges. Technology networks usually are in large scale and its visual representation is an ongoing research topic. The efficiency
Figure 1.2: Biological network of the protein-protein interaction network in budding yeast. The size of the nodes is proportional to the links incident to them. This network is drawn using k-core decomposition.

of technology networks greatly impacts all aspects of our life. For example, how to increase the transmission efficiency of power grid in US while maintaining its robustness to random power plant failures. Figure 1.3 is the topology of the western states power grid of the United States (Watts and Strogatz, 1988). There are 4941 nodes and 6594 edges in this large network. Each node represents a power plant and a link between two nodes indicates transmission between the two plants. The cost efficiency and robustness of such networks are important research topics.

Although network data are expressed in different forms, they could share similar research objectives. For example, some common research objectives are network sampling, community detection in networks, network dynamics, network modeling, network motif detection or missing link prediction. There is an ever growing list of research topics in the field of network data analysis. Interested readers can refer to Goldenberg et al. (2010) and Kolaczyk (2009) for a more detailed review of network analysis.

This dissertation is organized as follow. In Chapter 2, new sequential importance sampling
methods are proposed for sampling networks with a given degree sequence. The efficiency of the new algorithms is demonstrated through simulation and real data. We also study the asymptotic behavior of the sampler. In Chapter 3, we fit an exponential random graph model to a dolphin network that is resilient to targeted attacks. The fitted model is shown to characterize the network resiliency and identifies local structures that can reproduce the global resilience property. Chapter 4 introduces simulated annealing algorithms with efficient proposal distribution that can identify dense subgroups in a network. Chapter 5 provides a statistical framework for modularity-based network community detection. Based on the proposed statistical framework, we develop a hypothesis testing procedure to test the significance of an identified community structure. We also show the modularity formulated under the proposed statistical framework is consistent under the degree-corrected stochastic block model.

Figure 1.3: The transmission of the Western States Power Grid of the United States.
Chapter 2

Sampling for Conditional Inference on Network Data

2.1 Introduction

Over the past decade, there has been an explosion of interest in the study of network data across the sciences. Network structures arise in modeling a wide variety of systems in sciences and engineering, e.g., the Internet, social networks, biological networks, and information networks. These networks have been used to analyze social interaction patterns (Borgatti et al., 2009), identify network motifs (Kashtan et al., 2004), extract densely connected molecular modules (Spirin and Mirny, 2003), and much more. In the fast growing literature on the modeling of complex networks, one of the most important classes of network models is random graphs (Bollobas, 2001). In particular, random graphs with given vertex degrees have been widely used as a model for many real-world complex networks, and this model has proved useful in understanding a variety of network properties (Holland and Leinhardt, 1981; Snijders, 1991; Newman, 2003).

In order to detect deviations from randomness in network properties, a common approach is to compare the observed network with the set of random graphs that have the same degree sequence as the observed data. From a statistical point of view, this is a hypothesis testing problem with the reference distribution for the null hypothesis chosen to be the uniform distribution over all graphs with given vertex degrees. There are several settings in which it is desirable to condition on the degree sequence in performing a hypothesis test. For applications in which the subjects are not obtained by a sampling scheme but are the only ones available to the researcher, conditioning on the vertex degrees of the network creates a probabilistic basis for a test (Lehmann, 1986). In some other applications, such as those related to the exponential random graph models, the vertex
degrees are sufficient statistics under the null hypothesis. Conditioning on the vertex degrees is a way to remove the effect of nuisance parameters on tests (Lehmann, 1986, chap. 4; Snijders, 1991).

The fixing of the degree sequence makes the problems of enumeration and of finding the distribution of a test statistic difficult. No good combinatorial methods or analytical approximations are known for deriving the distribution of test statistics when graphs with given degree sequence are uniformly distributed. However, if we can simulate graphs from the uniform distribution, we can accurately approximate the distribution of any test statistic. Several Markov chain Monte Carlo (MCMC) algorithms for generating graphs from the uniform distribution have been proposed (Roberts, 2000; Milo et al., 2002; McDonald, Smith, and Forster, 2007; Handcock et al., 2008). However, some algorithms are not known to be rapidly mixing for general degree sequences, and some algorithms can get stuck and the output is non-uniform (King, 2004). As Goldenberg et al. (2009) pointed out, “generating such exact distributions is a very tricky matter in discrete exponential families because of the need to utilize appropriate Markov bases, either explicitly as in Diaconis and Sturmfels (1998) or implicitly. It is unclear whether the proposals in this literature are in fact reaching all possible tables associated with the distribution.”

Snijders (1991) was the first to consider importance sampling in the context of graphs with fixed degree sequence. However the variation of the importance weights is often large in his method, and it sometimes generates a large proportion of invalid graphs. Blitzstein and Diaconis (2010) developed another importance sampling algorithm for generating random graphs with given degree sequence. They sample the graph edge by edge, but the proposal distribution for sampling each edge is chosen based on intuitive arguments without theoretical justifications. A different proposal distribution for sampling the graph edge by edge was proposed by Bayati, Kim, and Saberi (2010), but the performance of their algorithm is not as good as Blitzstein and Diaconis’ (2010) algorithm in our simulation studies.

Sampling from the uniform distribution over graphs with fixed degree sequence is a difficult problem. In this chapter, we propose a new sequential sampling method which uses the asymptotic approximation of Bender and Canfield (1978) to guide the sampling at each step. The graphs are generated from a distribution close to the target uniform distribution, and an important weight is
assigned to each graph. Based on the graphs and the weights, we can estimate the distribution of any test statistic and approximate the number of graphs with fixed degree sequence.

This chapter is organized as follows. Section 2.2 gives the basic notation for networks and graphs. Section 2.3 introduces the basic sequential importance sampling (SIS) methodology. Section 2.4 derives the new sampling algorithm based on the asymptotic approximation of Bender and Canfield (1978). Section 2.5 studies the asymptotic property of the SIS algorithm for large sparse graphs. Section 2.6 proposes a refined SIS method that always generates valid graphs. Section 2.7 shows some applications and numerical examples, and Section 2.8 provides concluding remarks.

2.2 Basic Notation

A network usually refers to a collection of interconnected items. Borrowing the notation from graph theory, a network $G$ can be denoted by $G = (V, E)$, where $V$ is a set of vertices (or nodes) and $E$ is a set of edges (or links). Vertices in the network represent the items under study, and edges correspond to the relationships between the items. We use $\{i, j\}$ to denote an edge between vertex $i$ and vertex $j$. The degree of a vertex is the number of edges incident to the vertex. In this chapter, we are mainly concerned with undirected graphs with no loops (both ends of an edge connect to a single vertex) or multiple edges between a pair of vertices. In other words, we only consider simple graphs. See Figure 2.1 for an example of a simple graph.

An undirected simple graph with $n$ vertices can also be represented by its adjacency matrix: an $n \times n$ binary symmetric square matrix with $t_{ij} = 1$ if there is an edge between vertex $i$ and vertex $j$. Since loops are not allowed, the adjacency matrix has zeros on its diagonal. See Figure 2.1 for an example of an adjacency matrix.

2.3 Sequential Importance Sampling

For a simple undirected graph $G$ with vertex degrees $d = (d_1, \ldots, d_n)$, it can be represented by an adjacency matrix $T$ which is an $n \times n$ symmetric zero-one table with column sums $d = (d_1, \ldots, d_n)$ and a zero diagonal. Because of the one-to-one correspondence between graphs and their adjacency
matrices, the problem of simulating random graphs with given vertex degrees is equivalent to simulating random symmetric zero-one tables with given column sums and a zero diagonal. In the following, we focus on sampling such tables.

Let $\Sigma_d$ denote the set of all $n \times n$ symmetric zero-one tables with column sums $d$ and a zero diagonal. Let $p(T) = 1/|\Sigma_d|$ be the uniform distribution over $\Sigma_d$. If we can simulate tables from an easy-to-sample proposal distribution $q(\cdot)$, where $q(T) > 0$ for all $T \in \Sigma_d$, then we have

$$E_q \left[ \frac{1_{\{T \in \Sigma_d\}}}{q(T)} \right] = \sum_{T \in \Sigma_d} \frac{1}{q(T)} q(T) = |\Sigma_d|.$$  

Hence we can estimate $|\Sigma_d|$ by

$$\hat{|\Sigma_d|} = \frac{1}{N} \sum_{i=1}^{N} \frac{1_{\{T_i \in \Sigma_d\}}}{q(T_i)}$$  \hspace{1cm} (2.1)

from $N$ independent samples $T_1, \cdots, T_N$ drawn from $q(T)$. Furthermore, if we are interested in evaluating

$$\mu = E_p f(T) = \sum_{T \in \Sigma_d} f(T) p(T),$$

we can use the weighted average

$$\hat{\mu} = \frac{\sum_{i=1}^{N} f(T_i) \frac{p(T_i)}{q(T_i)}}{\sum_{i=1}^{N} \frac{p(T_i)}{q(T_i)}} = \frac{\sum_{i=1}^{N} f(T_i) \frac{1_{\{T_i \in \Sigma_d\}}}{q(T_i)}}{\sum_{i=1}^{N} \frac{1_{\{T_i \in \Sigma_d\}}}{q(T_i)}}$$  \hspace{1cm} (2.2)

as an estimate of $\mu$. This is the standard importance sampling procedure. For example, if we let
\( f(T) = 1_{\sum_{1 \leq j < k < l \leq n} t_{jk} t_{jl} t_{kl} \geq s} \), then (2.2) estimates the proportion of graphs containing at least \( s \) triangles.

The standard error of \( \hat{\mu} \) can be simply estimated by further repeated sampling. Another way is to approximate the denominator of \( \hat{\mu} \) by the \( \delta \)-method and obtain

\[
\text{std}(\hat{\mu}) \approx \sqrt{\text{var}_q(f(T)p(T)/q(T)) + \mu^2 \text{var}_q(p(T)/q(T)) - 2\mu \text{cov}_q(f(T)p(T)/q(T), p(T)/q(T))/\sqrt{N}.}
\]

(2.3)

However, since this standard deviation is dependent on the particular function of interest, it is also useful to consider a "function-free" criterion, the effective sample size (Kong, Liu, and Wong 1994) to measure the overall efficiency of an importance sampling algorithm:

\[
\text{ESS} = \frac{N}{1 + \text{cv}^2},
\]

(2.4)

where the coefficient of variation (cv) is defined as

\[
\text{cv}^2 = \frac{\text{var}_q\{p(T)/q(T)\}}{E_q^2\{p(T)/q(T)\}},
\]

(2.5)

which is equal to \( \text{var}_q\{1_{\{T \in \Sigma_d\}}/q(T)\}/E_q^2\{1_{\{T \in \Sigma_d\}}/q(T)\} \) for the current problem. The cv\(^2 \) is simply the \( \chi^2 \)-distance between the two distributions \( p \) and \( q \); the smaller it is, the closer the two distributions are. Heuristically, the ESS measures how many i.i.d. samples are equivalent to the \( N \) weighted samples. In practice the theoretical value of the cv\(^2 \) is unknown, so its sample counterpart is used to estimate cv\(^2 \).

A central problem in importance sampling is the construction of a good proposal distribution \( q(\cdot) \). Because the target space \( \Sigma_d \) is rather complicated, it is not immediately clear what proposal distribution \( q(T) \) can be employed. One of the most useful strategies for such problems is to decompose a high dimensional problem into many low dimensional pieces, and then build up the proposal distribution sequentially. This is the key idea of sequential importance sampling (SIS).
2.4 Sampling Random Networks

The idea of SIS has been used to construct the proposal distribution for sampling networks (or adjacency matrices) with given vertex degrees (Snijders, 1991; Bayati, Kim, and Saberi, 2010; Blitzstein and Diaconis, 2010). One common feature of the available methods is the network is sampled edge by edge, or equivalently, the table is sampled cell by cell. Also the proposal distribution for each cell is often chosen based on intuitive argument without theoretical justifications.

To improve the efficiency of the current sampling algorithms, we propose a new sequential sampling method which samples the network node by node and uses the asymptotic approximation of Bender and Canfield (1978) to guide the sampling of each node. This is equivalent to sampling the adjacency matrix $T$ column by column. Denoting the configurations of the columns of $T$ by $t_1, \ldots, t_n$, we can write $q(T)$ as:

$$q(T = (t_1, \cdots, t_n)) = q(t_1)q(t_2|t_1)q(t_3|t_2,t_1) \cdots q(t_n|t_{n-1}, \cdots, t_1).$$

We start by sampling the first column of the table conditional on its marginal sum $d_1$. We need to choose $d_1$ of the $n - 1$ possible positions to put 1’s in (the first cell is restricted to be 0). Suppose the $d_1$ rows we choose are $i_1, \ldots, i_{d_1}$. Conditional on the realization of the first column, we remove the first column and the first row (by symmetry), update the column sums, and sample the first column of the remaining $(n - 1) \times (n - 1)$ subtable in a similar manner. The column sums of the subtable are updated by subtracting the respective numbers in the first row from the original column sums. This procedure is repeated recursively until all the columns are sampled.

We use $d_j^{(l)}$, $j = 1, \ldots, n - (l - 1)$, to denote the updated column sums after the first $l - 1$ columns have been sampled. So $d_j^{(1)} = d_j$ and, after sampling the positions $i_1, \ldots, i_{d_1}$ for the first column, we have

$$d_j^{(2)} = \begin{cases} 
  d_j^{(1)} - 1, & \text{if} \ j + 1 = i_k \text{ for some } 1 \leq k \leq d_1, \\
  d_j^{(1)}, & \text{otherwise.} 
\end{cases}$$

(2.6)
Note that after sampling the first \( l - 1 \) columns, the \( l \)-th column in the original table is updated to the first column of the current table.

To design a good proposal distribution for the first column \( t_1 \), we can start by writing out the true marginal distribution of \( t_1 \) under the uniform distribution over \( \Sigma_d \). Let \( v(i_1, \ldots, i_{d_1}) \) be the zero-one vector of length \( n \) which has \( i_k \)-th component equal to 1 for \( k = 1, \ldots, d_1 \), and all other components equal to 0. For the configuration of the first column \( t_1 = v(i_1, \ldots, i_{d_1}) \), we let \( d^{(2)} = (d_1^{(2)}, \ldots, d_{n-1}^{(2)}) \) be the updated column sums defined in (2.6). Then the true marginal distribution of \( t_1 \) is

\[
p(t_1 = v(i_1, \ldots, i_{d_1})) = \frac{|\Sigma_{d^{(2)}}|}{|\Sigma_d|}.
\]

The exact formula for the number of tables in \( \Sigma_d \) is not available, but asymptotic formulas for \( |\Sigma_d| \) are obtained for different kinds of degree sequences (Bender and Canfield 1978; McKay 1985; McKay and Wormald 1990, 1991). In particular, Bender and Canfield (1978) give the asymptotic number of symmetric non-negative integer matrices with bounded entries, specified and bounded column sums, and a sparse set of structural zeros. Based on Theorem 1 of Bender and Canfield (1978), we have the following corollary which gives an approximation to the number of symmetric zero-one matrices with fixed column sums \( d \) and a zero diagonal.

**Corollary 2.4.1.** For given \( d = (d_1, \ldots, d_n) \),

\[
|\Sigma_d| \sim \Delta_d = h(M) \prod_{j=1}^{n} d_j! e^{-\alpha(d)},
\]

where \( M = \sum_{j=1}^{n} d_j \), \( h(M) = M! / [(M/2)!2^{M/2}] \), and \( \alpha(d) = [\sum_{j=1}^{n} \binom{d_j}{2}]^2 / M^2 + \sum_{j=1}^{n} \binom{d_j}{2} / M \).

The proof is given in Section 2.9. This asymptotic approximation is derived assuming all column sums are bounded above by a constant \( d^* \) and \( M \to \infty \). In fact, a more precise statement for Corollary 2.4.1 is that (Bender and Canfield, 1978)

\[
\lim_{M \to \infty} \sup_{\Sigma_{d}=\{d_1,\ldots,d_n\}} \left| \frac{|\Sigma_d|}{\Delta_d} - 1 \right| = 0.
\]
By (2.7), we have the following theorem whose proof is given in Section 2.9.

**Theorem 2.4.1.** Under the uniform distribution over $\Sigma_d$, the marginal distribution of the first column $t_1$ can be approximated by

$$p(t_1 = v(i_1, \ldots, i_{d_1})) \approx \frac{\Delta d^{(2)}}{\Delta_d} \propto \left( \prod_{k=1}^{d_1} d_{i_k} \right) \exp \left\{ \frac{(\sum_{j=2}^{n} d_j^2 + d_1) (\sum_{k=1}^{d_1} d_{i_k}) - (\sum_{k=1}^{d_1} d_{i_k})^2}{(M - 2d_1)^2} \right\}. \quad (2.9)$$

The approximation error of (2.9) is $||\Sigma_d^{(2)}|/|\Sigma_d| - \Delta_d^{(2)}/\Delta_d||$. In the following corollary, whose proof is given in Section 2.9, we quantify the approximation error based on a refined approximation formula for $|\Sigma_d|$ in Theorem 4.6 of McKay (1985).

**Corollary 2.4.2.** Suppose that $d_{\text{max}} = \max_{1 \leq j \leq n} d_j = O(1)$, then

$$\frac{|\Sigma_d^{(2)}|}{|\Sigma_d|} - \frac{\Delta_d^{(2)}}{\Delta_d} = O \left( \frac{1}{M} \right) \quad (2.10)$$

as $M = \sum_{j=1}^{n} d_j \to \infty$.

Note that in the approximation (2.9), the second term in the exponent is much smaller than the first term, and two simple lower bounds and upper bounds for the second term can be obtained as follows:

$$\frac{\sum_{k=1}^{d_1} d_{i_k}^2}{(M - 2d_1)^2} \leq \frac{(\sum_{k=1}^{d_1} d_{i_k})^2}{(M - 2d_1)^2} \leq \frac{d_1 \sum_{k=1}^{d_1} d_{i_k}^2}{(M - 2d_1)^2}. \quad (2.11)$$

$$0 \leq \frac{(\sum_{k=1}^{d_1} d_{i_k})^2}{(M - 2d_1)^2} \leq \frac{(M - d_1)^2}{(M - 2d_1)^2}. \quad (2.12)$$

The first upper bound is based on the Cauchy-Schwarz inequality. The second upper bound is due to the fact that $i_k \neq 1$ for $k = 1, \ldots, d_1$ (the matrix has a zero diagonal). Hence we can further simplify (2.9) by approximating the second term in the exponent by either its lower bound or upper bound, and obtain

$$p(t_1 = v(i_1, \ldots, i_{d_1})) \approx \frac{\Delta d^{(2)}}{\Delta_d} \propto \prod_{k=1}^{d_1} d_{i_k} e^{\beta(d_{i_k})}. \quad (2.13)$$
approximately, where the expression of \( \beta(d_{ik}) \) depends on which bound is used in the approximation. For example, if the second lower bound 0 or the second upper bound \((M - d_1)^2/(M - 2d_1)^2\) is used, then (after ignoring the constant term \((M - d_1)^2/(M - 2d_1)^2\) )

\[
\beta(d_{ik}) = \frac{(\sum_{j=2}^{n} d_j^2 + d_1)d_{ik}}{(M - 2d_1)^2}. \tag{2.14}
\]

If the first lower bound or upper bound is used, then \( \beta(d_{ik}) = (\sum_{j=2}^{n} d_j^2 + d_1 - d_{ik})d_{ik}/(M - 2d_1)^2 \) or \( \beta(d_{ik}) = (\sum_{j=2}^{n} d_j^2 + d_1 - d_{ik})d_{ik}/(M - 2d_1)^2 \), respectively. In our simulation studies, we observed that these three forms of \( \beta(d_{ik}) \) performed similarly well, with the one in (2.14) being slightly better than the other two. For the rest of the chapter, we focus on the sampling algorithm based on approximation (2.13) with \( \beta(d_{ik}) \) given in (2.14).

The above derivation suggests that the true marginal distribution of \( t_1 \) under the uniform distribution over \( \Sigma_d \) can be approximated by (2.13). It is known that the conditional Poisson distribution can be written as

\[
p \left[ (Z_1, \ldots, Z_n) = v(i_1, \ldots, i_{d_1}) \mid \sum_{j=1}^{n} Z_j = d_1 \right] \propto d_1 \prod_{k=1}^{d_1} \frac{p_{ik}}{1 - p_{ik}}, \tag{2.15}
\]

where \( Z_1, \ldots, Z_n \) are independent Bernoulli trials with probability of successes \( p_1, \ldots, p_n \), respectively. Therefore, the approximation (2.13) is a conditional Poisson distribution with

\[
p_{ik} = \frac{d_{ik} e^{\beta(d_{ik})}}{[1 + d_{ik} e^{\beta(d_{ik})}]} . \tag{2.16}
\]

We adopt the drafting sampling scheme proposed by Chen, Dempster, and Liu (1994) and Chen and Liu (1997) to sample from the conditional Poisson distribution. The goal is to select \( d_1 \) units without replacement from the set \( \{1, \ldots, n\} \) with probability proportional to the product of each unit’s “weight” \( w_i = d_i e^{\beta(d_i)} \). Let \( A_k (k = 0, \ldots, d_1) \) denote the set of units selected after \( k \) draws. Then \( A_0 = \emptyset \) and \( A_{d_1} \) is the final sample. At the \( k \)th step of the drafting sampling \((k = 1, \ldots, d_1)\),
a unit \( j \in A_{k-1}^c \) is selected into the sample with probability

\[
P(j, A_{k-1}^c) = \frac{w_j R(d_1 - k, A_{k-1}^c \setminus j)}{(d_1 - k + 1) R(d_1 - k + 1, A_{k-1}^c)},
\]

where

\[
R(s, A) = \sum_{B \subseteq A, |B| = s} \left( \prod_{i \in B} w_i \right).
\]

Here \( R(s, A) \) can be computed through the recursive formula

\[
R(s, A) = R(s, A \setminus \{s\}) + w_s R(s - 1, A \setminus \{s\}).
\]

After the first column \( t_1 \) is generated by the drafting sampling algorithm, the proposal probability \( q(t_1) \) is computed using (2.15) with \( p_{i_k} \) given in (2.16). Then we remove the first column and the first row, and sample the first column of the remaining \( (n - 1) \times (n - 1) \) subtable in the same way. Our algorithm allows the columns to be sampled in any order. However, we found that in our experiments, it is usually more efficient to first sample the column with the largest column sum. So at each step of the sequential sampling algorithm, we always reorder the column sums of the subtable so that the first column of the subtable has the largest column sum. More discussion about the effect of orderings and some theoretical insights are given in Section 2.8.

### 2.5 Theoretical Properties

As discussed in Section 2.3, a good proposal distribution is key to the efficiency of an importance sampling algorithm, and the \( \text{cv}^2 \) defined in (2.5) can be used to measure such efficiency. The \( \text{cv}^2 \) is the standardized variance of the importance weight. In this section, we study the asymptotic behavior of the importance weight as the size of the graph grows.

In the derivation of the proposal distribution (2.13), the target distribution \( |\Sigma_{d(2)}|/|\Sigma_d| \) is first approximated by \( \Delta_{d(2)}/\Delta_d \), and then \( \Delta_{d(2)}/\Delta_d \propto \Delta_{d(2)} \) is approximately proportional to our proposal distribution (2.13) with \( \beta(d_{i_k}) \) given in (2.14). This is essentially equivalent to finding an approximation \( \Lambda_{d(2)} \) to \( \Delta_{d(2)} \), and choosing our proposal distribution to be proportional to \( \Lambda_{d(2)} \). Based on the proof of Theorem 1, we can write down the explicit expression of \( \Delta_{d(2)} \) by putting
back all proportionality constants:

$$\Delta_{d^{(2)}} = \frac{e^{1/4} h(M - 2d_1)}{\prod_{j=1}^n d_j!} \left( \prod_{k=1}^{d_1} e^{\beta(d_{ik})} \right) \exp \left\{ -\frac{\left( \sum_{j=2}^n d_j^2 + d_1 \right)^2}{4(M - 2d_1)^2} - \frac{\left( \sum_{k=1}^{d_1} d_{ik} \right)^2}{(M - 2d_1)^2} \right\}. \quad (2.19)$$

If the lower bound in (2.12) is used, the approximation $\Lambda_{d^{(2)}}$ is

$$\Lambda_{d^{(2)}} = \frac{e^{1/4} h(M - 2d_1)}{\prod_{j=1}^n d_j!} \left( \prod_{k=1}^{d_1} e^{\beta(d_{ik})} \right) \exp \left\{ -\frac{\left( \sum_{j=2}^n d_j^2 + d_1 \right)^2}{4(M - 2d_1)^2} \right\}. \quad (2.20)$$

Because of the upper and lower bounds in (2.12), we have

$$\exp \left\{ -\frac{(M - d_1)^2}{(M - 2d_1)^2} \right\} \leq \frac{\Delta_{d^{(2)}}}{\Lambda_{d^{(2)}}} \leq 1. \quad (2.21)$$

It is not difficult to see that our proposal distribution (2.13) is proportional to $\Lambda_{d^{(2)}}$ because all the other terms in $\Lambda_{d^{(2)}}$ are constants for every configuration of the first column. Therefore our proposal distribution for the first column is

$$q(t_1 = v(i_1, \ldots, i_{d_1})) = \frac{\Lambda_{d^{(2)}}}{S(d)} = \frac{d^{(2)}_{d_1}}{S(d^{(2)})}, \quad (2.22)$$

where $S(d) = \sum_{d \rightarrow d^{(2)}} \Lambda_{d^{(2)}}$ is a summation over all possible ways to move from $d$ to $d^{(2)}$ after filling in the first column with an $n$-dimensional 0-1 vector with $d_1$ ones whose first cell is 0. So $S(d)$ is the normalizing constant that makes $q(t_1)$ a well defined probability. The sampling distribution for other columns can be written down in a similar way. The importance weight for a valid table is (note that $q(t_n|t_{n-1}, \ldots, t_1) = 1$)

$$w(T = (t_1, \ldots, t_n)) = \frac{p(T)}{q(T)} = \frac{1}{|\Sigma_d|} \frac{1}{q(T)} = \frac{1}{|\Sigma_d|} \frac{1}{q(t_1)} \frac{1}{q(t_2|t_1)} \cdots \frac{1}{q(t_n|t_{n-1}, \ldots, t_1)}$$

$$= \frac{1}{|\Sigma_d|} \frac{S(d)}{\Lambda_{d^{(2)}}} \frac{S(d^{(2)})}{\Lambda_{d^{(3)}}} \cdots \frac{S(d^{(n-1)})}{\Lambda_{d^{(n)}}}. \quad (2.23)$$

In the following theorem, whose proof is given in Section 2.9, we characterize the asymptotic behavior of the importance weight as the size of the graph grows.

16
Theorem 2.5.1. Suppose that \( \max_{1 \leq j \leq n} d_j = O(1) \), then the importance weight \( w(T) \) given in (2.23) is bounded as \( M = \sum_{j=1}^{n} d_j \to \infty \).

In this theorem, since the column sum (and row sum) is bounded, we have the number of columns (and rows) \( n \to \infty \) as \( M = \sum_{j=1}^{n} d_j \to \infty \). This also implies that most entries in the table will be zero since the size of the table is increasing but the row and column sums remain bounded. So this theorem characterizes the asymptotic property of the importance weights for large sparse tables or graphs.

This theorem says that the importance weight remains bounded as the number of edges (and nodes) in the graph goes to infinity. This indicates that our proposal distribution is very close to the target uniform distribution and the importance weight does not vary too much. In fact, a bounded importance weight immediately leads to a bounded variance of the importance weight, i.e., a bounded \( \text{cv}^2 \). This implies that the SIS estimator is strongly efficient in the context of rare-event simulation (Blanchet, 2009). This property guarantees that the SIS algorithm we proposed can still work efficiently even for large sparse graphs.

A bounded importance weight is also useful for studying the number of samples needed to achieve a certain accuracy of the estimator (2.1) or (2.2). By Chebyshev’s inequality, we have

\[
P(\|\Sigma_d\| - |\Sigma_d| \geq \epsilon |\Sigma_d|) \leq \frac{\text{var}(w(T))}{N \epsilon^2}. \tag{2.24}
\]

Theorem 2.5.1 implies that there exists a constant \( \kappa \) independent of \( M \) such that \( w(T) \leq \kappa \). This immediately leads to \( \text{var}(w(T)) \leq \kappa^2 \). Therefore, we need \( N \geq \epsilon^{-2} \delta^{-1} \kappa^2 \) samples to achieve \( \epsilon \)-relative precision with \( 1 - \delta \) confidence. Notice that the number of samples needed is bounded as \( M \to \infty \). Similarly based on Chebyshev’s inequality and the \( \Delta \)-method, we can show that the number of samples needed for the estimator (2.2) to achieve \( \epsilon \)-relative precision with \( 1 - \delta \) confidence is also bounded as \( M \to \infty \).
2.6 A Refined SIS Algorithm

The SIS algorithm in the last section is shown to be very efficient in the numerical studies. One minor drawback of this SIS algorithm is that it does not guarantee the validity of every sample. Sometimes the sampling cannot proceed after a few columns have been generated because no zero-one table satisfies the updated constraints on the remaining subtable. For example, suppose the degree sequence is \( (1, 1, 2, 2) \). If we happen to draw the first column as \( (0, 1, 0, 0)^T \), i.e., an edge \( \{1, 2\} \) is added between the first two nodes, then we cannot sample the third column. When this happens, a zero weight is assigned to this invalid sample. Although the percentage of invalid samples tends to be small, we develop a refined SIS algorithm in this section which guarantees that every sample is valid.

The generation of invalid samples is related to the fact that not every sequence of nonnegative integers \( (d_1, \ldots, d_k) \) corresponds to a labeled simple graph with vertex set \( \{1, \ldots, k\} \), in which vertex \( i \) has degree \( d_i \). If a finite sequence \( (d_1, \ldots, d_k) \) does correspond to a simple graph, then the sequence is called *graphical*. In order to guarantee the existence of subtables with the updated constraints, we have to first make sure that the sequence of the updated column sums is graphical.

There are several equivalent conditions to check whether a sequence is graphical (Mahadev and Peled, 1995). Probably the most famous and easy-to-check criterion is given by Erdos and Gallai (1960).

**Theorem 2.6.1.** *(Erdos-Gallai)* Let \( d_1 \geq d_2 \geq \cdots \geq d_n \) be nonnegative integers with \( \sum_{i=1}^n d_i \) even. Then \( d = (d_1, \ldots, d_n) \) is graphical if and only if

\[
\sum_{i=1}^k d_i \leq k(k-1) + \sum_{i=k+1}^n \min(k, d_i) \quad \text{for each } k \in \{1, \ldots, n\}.
\]

The \( n \) inequalities in Erdos-Gallai theorem have been simplified and refined later. In particular, Let \( m = \lvert \{j : d_j \geq j - 1\} \rvert \) be the *corrected Durfee number* of \( d \). Mahadev and Peled (1995) showed that \( d \) is graphical if and only if it satisfies the first \( m \) inequalities in (2.25). In many cases the
corrected Durfee number \(m\) is much smaller than \(n\), so the number of inequalities to check is greatly reduced.

The Erdos-Gallai theorem can be incorporated into the sequential sampling scheme in Section 2.4. We can check the Erdos-Gallai conditions after the sampling of each column, and assign zero weight to the sample if the sequence of updated column sums is not graphical. This procedure can detect invalid samples at an early stage and avoid wasting computational resources. Another way to incorporate the theorem is to use the Erdos-Gallai conditions to determine the set of configurations that are valid for the column we are sampling, and then restrict our samples to the set of valid configurations. However the description of the set of valid configurations for each column is quite complicated.

Notice that the drafting sampling method, which is used in the SIS algorithm in Section 2.4 to sample each column, actually samples the positions to put 1’s in one by one. This is equivalent to sampling the edges linked to a node one by one. Since it is much easier to determine the set of possible positions to put a 1 in (i.e., the set of possible nodes to be linked with the current node) at each step of the drafting sampling algorithm, we will incorporate the Erdos-Gallai conditions into the drafting sampling algorithm to make sure that the residual degree sequence is always graphical. This can prevent many invalid configurations. For example, for the degree sequence \((1, 1, 2, 2)\) we have considered before, the cell \(t_{21}\) would not be a possible position to put a 1 in (i.e., connecting the first two nodes is not allowed) because the residual degree sequence \((0, 0, 2, 2)\) is not graphical.

However having a graphical residual degree sequence is not sufficient to guarantee that a valid simple graph can be generated. For example, suppose the degree sequence is \((3, 4, 2, 2, 1)\). If we put a 1 in the cell \(t_{21}\) (i.e., add an edge \(\{1, 2\}\)), then the updated degree sequence is \((2, 3, 2, 2, 1)\), which is graphical. Then we may put a 1 in the cell \(t_{51}\) (i.e., add an edge \(\{1, 5\}\)), and the residual degree sequence becomes \((1, 3, 2, 2, 0)\), which is still graphical. Now the only way to realize this residual sequence is to add edges \(\{1, 2\}\), \(\{2, 3\}\), \(\{2, 4\}\), and \(\{3, 4\}\). However, edge \(\{1, 2\}\) has been added in the first step already. So the realization of the residual degree sequence \((1, 3, 2, 2, 0)\) will create multiple edges between nodes 1 and 2, which violates the constraints.

To avoid this type of invalid choices when we sample the edges for the current node, we need to
construct a “forbidden set” to keep track of the nodes that have already been linked to the current node. A link between a node in the forbidden set and the current node is called a forbidden link. When we consider whether a node can be linked to the current node, we need to make sure that the residual degree sequence corresponds to a simple graph without having forbidden links. The next section gives details on how to find the set of nodes that can be linked to the current node.

2.6.1 Building the Candidate Set

Suppose the current residual degree sequence is \( \mathbf{d} = (d_1, \ldots, d_n) \), and we are sampling edges for the first node with remaining degree \( d_1 \). The forbidden set is the set of nodes that have already been linked to node 1. Before the sampling of the next edge for node 1, we need to identify the candidate set of nodes that are allowed to be linked to node 1. We will use two theorems from Del Genio et al. (2010) to construct this candidate set. We first introduce the following definition.

**Definition 2.6.1.** For a degree sequence \( \mathbf{d} = (d_1, \ldots, d_n) \) and a forbidden set associated with node \( i \), the leftmost adjacency set of node \( i \) with degree \( d_i \) is the set of the \( d_i \) highest degree nodes that are not in the forbidden set.

The following theorem from Kim et al. (2009) and Del Genio et al. (2010) generalizes the theorem of Havel (1955) and Hakimi (1962) by taking into account forbidden links.

**Theorem 2.6.2.** Let \( \mathbf{d} = (d_1, \ldots, d_n) \) be a graphical degree sequence. Assume there is a forbidden set of nodes to which node \( i \) is not allowed to be linked. Then a simple graph avoiding the forbidden links can be constructed if and only if a simple graph can be constructed where node \( i \) is connected to all the nodes in its leftmost adjacency set.

When we sample node 1 with an associated forbidden set, Theorem 2.6.2 implies that the nodes in the leftmost adjacency set of node 1 are allowed to be linked to node 1, i.e., they are in the candidate set. For a non-forbidden node, say node \( j \), that is not in the leftmost adjacency set of node 1, we may connect node 1 with node \( j \) and the \( d_1 - 1 \) highest degree nodes in the leftmost adjacency set of node 1. After removing these \( d_1 \) links to node 1 from further consideration,
the residual degree sequence no longer contains node 1 and there are no forbidden links for the
remaining nodes. If the residual degree sequence is graphical according to Theorem 2.6.1, then
obviously node \( j \) should be in the candidate set. If the residual degree sequence does not satisfy
the conditions in Theorem 2.6.1, then based on Theorem 2.6.2, node \( j \) is not allowed to be linked
to node 1. In principle, we can check every non-forbidden node in this way to decide whether it is
in the candidate set.

Del Genio et al. (2010) proved the following theorem to further simply the construction of the
candidate set.

**Theorem 2.6.3.** Let \( d = (d_1, \ldots, d_n) \) be a degree sequence, possibly with a set of forbidden links
incident on node \( i \), and let nodes \( j \) and \( k \) be two non-forbidden nodes with degrees such that \( d_j \geq d_k \).
If the residual degree sequence obtained from \( d \) by reducing the degrees for nodes \( i \) and \( j \) by 1 is
not graphical, then the residual degree sequence obtained from \( d \) by reducing the degrees for nodes
\( i \) and \( k \) by 1 is also not graphical.

Theorem 2.6.3 implies that if a non-forbidden node with degree \( d_j \) cannot be in the candidate
set, then all non-forbidden nodes with degree less than or equal to \( d_j \) cannot be in the candidate
set either. The degrees of non-forbidden nodes that cannot be in the candidate set are referred
to as *fail-degrees* in Del Genio et al. (2010). Therefore it is enough to determine the maximum
fail-degree by checking non-forbidden nodes of decreasing or increasing degree. See Del Genio et
al. (2010) for a more efficient algorithm to determine the maximum fail-degree.

It is worth mentioning that Blitzstein and Diaconis (2010) provide another way to guarantee
that their algorithm always produces valid graphs with a given degree sequence. However their
procedure requires that the node with the smallest degree needs to be sampled first. The scheme
we introduced in this section allows us to sample the degree sequence in any order. In our simulation
studies, we found that sampling the node with the largest degree first often gives smaller \( cv^2 \) than
sampling in other orders.
2.6.2 Sampling Procedure

After we build the candidate set, we need to sample a node from the candidate set and then link it with the current node. It is shown in Section 2.4 that the marginal distribution of a column is close to a conditional Poisson distribution. The drafting sampling algorithm, which is used to sample from the conditional Poisson distribution, actually samples the nodes one by one from all the remaining nodes. Therefore it is natural to continue using drafting sampling with a slight modification to draw a node from the candidate set at each step.

To be more specific, suppose the initial degree sequence is \( d = (d_1, \ldots, d_n) \) and we are sampling edges for the first node. Assume \( k - 1 \) nodes have already been chosen to be linked to node 1. The candidate set \( C_k \) for the \( k \)th node can be built using the method from the last section. Then based on drafting sampling, a node \( j \in C_k \) is selected to be the \( k \)th node with probability

\[
P(j, C_k) = \frac{w_j R(d_1 - k, C_k \setminus j)}{(d_1 - k + 1) R(d_1 - k + 1, C_k)}.
\]

(2.26)

See Section 2.4 for the definitions of \( w_j \) and the function \( R \). In the modified drafting sampling (2.26), a node is chosen from the candidate set \( C_k \), while in the original drafting sampling (2.17), a node is chosen from the set of all the unselected nodes \( A_{k-1}^c \).

The nodes sampled by the modified drafting sampling algorithm will be linked to node 1, so this is equivalent to sampling the edges for node 1 sequentially. In drafting sampling, the probability of the final unordered list of edges is computed using (2.15), but this probability is difficult to compute for the modified drafting sampling (2.26). However if we treat the list of edges as ordered, then the probability of generating the ordered list of edges is simply the product of a sequence of probabilities like (2.26). Due to this fact, we consider the samples from the refined SIS algorithm in a larger space \( \Gamma_d \): the set of all ordered list of edges output by the refined SIS algorithm. For any \( \gamma \in \Gamma_d \), let \( T(\gamma) \) be the corresponding table in \( \Sigma_d \) (the set of all \( n \times n \) symmetric zero-one tables with column sums \( d \) and a zero diagonal). Note that \( \Sigma_d \) is equivalent to the set of simple undirected graphs with vertex degrees \( d \).
For a function \( f \) on \( \Sigma_d \), suppose we are interested in evaluating
\[
\mu = E_p f(T) = \sum_{T \in \Sigma_d} f(T)p(T)
\]

Define \( \tilde{f} \) on the larger space \( \Gamma_d \) as \( \tilde{f}(\gamma) = f(T(\gamma)) \), for \( \gamma \in \Gamma_d \). Let \( c(\gamma) \) be the number of sequences in \( \Gamma_d \) that correspond to the same table in \( \Gamma_d \), i.e.,
\[
c(\gamma) = |\{ \gamma' \in \Gamma_d : T(\gamma') = T(\gamma) \}|.
\]

Define a new distribution \( \tilde{p} \) on \( \Gamma_d \) as \( \tilde{p}(\gamma) = \frac{p(T(\gamma))}{c(\gamma)} \), for \( \gamma \in \Gamma_d \). Then we have
\[
\mu = E_p f(T) = \sum_{T \in \Sigma_d} f(T)p(T) = \sum_{\gamma \in \Gamma_d} \tilde{f}(\gamma)\tilde{p}(\gamma) = \sum_{\gamma \in \Gamma_d} \tilde{f}(\gamma)\tilde{p}(\gamma) = E_\tilde{p}\tilde{f}(\gamma). \quad (2.27)
\]

Therefore if we have a proposal distribution \( q(\gamma) \) on \( \Gamma_d \), we can still use it in importance sampling to estimate an expectation on the space \( \Sigma_d \), because such an expectation can be rewritten as an expectation on the larger space \( \Gamma_d \) and an importance sampling estimate can be written as
\[
\tilde{\mu} = \frac{\sum_{i=1}^{N} \tilde{f}(\gamma_i) \frac{p(\gamma_i)}{q(\gamma_i)}}{\sum_{i=1}^{N} \frac{p(\gamma_i)}{q(\gamma_i)}} = \frac{\sum_{i=1}^{N} f(T(\gamma_i)) \frac{p(T(\gamma_i))}{c(\gamma_i)q(\gamma_i)}}{\sum_{i=1}^{N} \frac{p(T(\gamma_i))}{c(\gamma_i)q(\gamma_i)}}, \quad (2.28)
\]
where \( \gamma_1, \ldots, \gamma_N \) are independent samples drawn from \( q(\gamma) \). If \( p(T) \) is the uniform distribution, it will be canceled out from (2.28).

For estimating the number of tables in \( \Sigma_d \), we can write
\[
|\Sigma_d| = \sum_{T \in \Sigma_d} 1 = \sum_{T \in \Sigma_d} \sum_{\gamma : T(\gamma) = T} \frac{1}{c(\gamma)} = \sum_{\gamma \in \Gamma_d} \frac{1}{c(\gamma)} = \sum_{\gamma \in \Gamma_d} \frac{1}{c(\gamma)} q(\gamma) = E_q \left[ \frac{1}{c(\gamma)q(\gamma)} \right]. \quad (2.29)
\]
Therefore it can be estimated by
\[
\tilde{|\Sigma_d|} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{c(\gamma_i)q(\gamma_i)} . \quad (2.30)
\]
The computation of $c(\gamma)$ is straightforward (Blitzstein and Diaconis, 2010). If two sequences of edges $\gamma$ and $\gamma'$ correspond to the same network, then they must choose the sequence of nodes in the same order. For the same node in $\gamma$ and $\gamma'$, they must have the same set of edges, but the edges for that node could be sampled in different orders. If we let $d^{(l)}_j, j = 1, \ldots, n - (l - 1)$, denote the residual degree sequence in decreasing order after the first $l - 1$ nodes have been sampled, then

$$c(\gamma) = \prod_{l=1}^{n} d^{(l)}_j l!.$$ (2.31)

The importance weight of the refined algorithm can be written as

$$w_r(\gamma) = \tilde{p}(\gamma) \frac{q_r(\gamma)}{q_r(\gamma) c(\gamma)}.$$ (2.32)

The following theorem compares the important weight of the refined algorithm, which we refer to as SIS-CP-refined, and the importance weight of the SIS algorithm in Section 2.4, which we refer to as SIS-CP.

**Theorem 2.6.4.** The importance weight (2.32) for SIS-CP-refined and the important weight (2.23) for SIS-CP can be ordered as

$$w_r(\gamma) \leq w(T(\gamma)), \quad \gamma \in \Gamma_d.$$ (2.33)

The proof is in Section 2.9. Combining this theorem and Theorem 2.5.1, we immediately have the following corollary for the asymptotic behavior of the importance weight for SIS-CP-refined.

**Corollary 2.6.1.** Suppose that $\max_{1 \leq j \leq n} d_j = O(1)$, then the importance weight $w_r(\gamma)$ given in (2.32) is bounded as $M = \sum_{j=1}^{n} d_j \rightarrow \infty$.

### 2.7 Applications and Simulations

In the examples in this section, we generate networks by the SIS algorithms proposed in Sections 2.4 and 2.6. We refer to the SIS algorithm in Section 2.4 as SIS-CP, and the SIS algorithm in
Section 2.6 as SIS-CP-refined. Unless noted otherwise, in the implementation of SIS-CP and SIS-CP-refined, we always select the node with the largest degree to sample. This scheme requires that we start by sampling the node with the largest degree, and after we sample all of its edges, we choose the node with the largest degree from the residual degree sequence and sample all of its edges. This procedure continues until all the nodes and edges have been sampled. Both algorithms were coded in R except the drafting sampling part which was written as a C function. All examples were run on a MacBook Pro with 2.26 GHz Intel Core 2 Duo processor.

2.7.1 Counting Graphs

The closed-form expression for the number of graphs with fixed degrees, i.e., the size of the set $\Sigma_d$, is hard to obtain. Several asymptotic methods have been developed for approximating $|\Sigma_d|$ (Bender and Canfield, 1978; McKay, 1985; McKay and Wormald, 1990, 1991). However, these formulas may not be very accurate for finite graphs, and they are often designed for special types of degree sequences. Due to the limitations of asymptotic approaches, a method for quickly estimating the number of graphs with fixed degrees is of interest.

By using formulas (2.1) and (2.30), we can estimate $|\Sigma_d|$ based on i.i.d. samples from the proposal distribution. In the following, we compare SIS-CP and SIS-CP-refined with the importance sampling algorithm proposed by Blitzstein and Diaconis (2010). Blitzstein-Diaconis algorithm always chooses the node with the smallest degree to sample its edges, and during the sampling of edges for the current node, a node $j$ is chosen to be linked to the current node with probability proportional to the residual degree of node $j$. The R code of their algorithm is available at one of the authors’ web page http://www.people.fas.harvard.edu/~blitz/Site/Research.html, and we used this code to implement Blitzstein-Diaconis algorithm. Bayati et al. (2010) propose another importance sampling algorithm to sample the graph edge by edge. We tested their algorithm on the food web data in Table 2.1, and the standard error of their estimate is $0.133 \times 10^3$ with $cv^2 = 7.469$ based on 1000 samples, which is not as good as Blitzstein-Diaconis algorithm. So we did not include Bayati et al.’s (2010) algorithm in the comparison.

We compared SIS-CP, SIS-CP-refined, and Blitzstein-Diaconis algorithm on three examples.
The first example is estimating the number of labeled 3-regular graphs (graphs with all nodes having degree 3) with 24 nodes, for which the exact answer (only the first four digits are given here) is $6.287 \times 10^{31}$ (Sloane, 2010). The second example is the Chesapeake Bay food web with 33 nodes and 71 edges, and the degree sequence is $d=(7, 8, 5, 1, 1, 2, 8, 10, 4, 2, 4, 5, 3, 6, 7, 3, 2, 7, 6, 1, 2, 9, 6, 1, 3, 4, 6, 3, 3, 3, 2, 4, 4)$. The third example is a dolphin social network with 62 nodes and 159 edges, and the degree sequence is $d=(6, 8, 4, 3, 1, 4, 6, 5, 6, 7, 5, 1, 1, 8, 12, 7, 6, 9, 7, 4, 9, 6, 1, 3, 6, 3, 3, 5, 5, 9, 5, 1, 3, 10, 5, 1, 7, 11, 8, 2, 8, 5, 6, 7, 4, 11, 2, 6, 1, 2, 7, 10, 4, 2, 7, 2, 2, 9, 1, 5, 1, 3)$. See Sections 2.7.2 and 2.7.2 for more information about the food web and dolphin network.

<table>
<thead>
<tr>
<th>Method</th>
<th>Estimated number of graphs</th>
<th>$cv^2$</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(a) 3-regular graph with 24 nodes</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blitzstein-Diaconis</td>
<td>$(6.2714 \pm 0.0777) \times 10^{31}$</td>
<td>0.1536</td>
<td>95.3</td>
</tr>
<tr>
<td>SIS-CP</td>
<td>$(6.3065 \pm 0.0123) \times 10^{31}$</td>
<td>0.0038</td>
<td>12.3</td>
</tr>
<tr>
<td>SIS-CP-refined</td>
<td>$(6.2837 \pm 0.0118) \times 10^{31}$</td>
<td>0.0035</td>
<td>35.2</td>
</tr>
<tr>
<td><strong>(b) Food web data</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blitzstein-Diaconis</td>
<td>$(1.5508 \pm 0.0738) \times 10^{57}$</td>
<td>2.2680</td>
<td>304.9</td>
</tr>
<tr>
<td>SIS-CP</td>
<td>$(1.5447 \pm 0.0030) \times 10^{57}$</td>
<td>0.0037</td>
<td>20.0</td>
</tr>
<tr>
<td>SIS-CP-refined</td>
<td>$(1.5464 \pm 0.0030) \times 10^{57}$</td>
<td>0.0037</td>
<td>93.4</td>
</tr>
<tr>
<td><strong>(c) Dolphin network</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blitzstein-Diaconis</td>
<td>$(1.8497 \pm 0.0844) \times 10^{167}$</td>
<td>2.0822</td>
<td>2096.9</td>
</tr>
<tr>
<td>SIS-CP</td>
<td>$(1.8258 \pm 0.0018) \times 10^{167}$</td>
<td>0.0010</td>
<td>55.0</td>
</tr>
<tr>
<td>SIS-CP-refined</td>
<td>$(1.8288 \pm 0.0018) \times 10^{167}$</td>
<td>0.0010</td>
<td>373.1</td>
</tr>
</tbody>
</table>

Table 2.1: Performance comparison of three Monte Carlo methods for estimating the number of graphs.

The simulation results, based on 1000 importance samples for the three examples, are summarized in Table 2.1. The number after the “±” sign is the standard error. The results for the 3-regular graph with 24 nodes show that all three methods give good approximation to the true value, but the accuracy level of the estimates are quiet different. In all three examples, SIS-CP and SIS-CP-refined have very similar performance. Both of them outperform Blitzstein-Diaconis algorithm, and the improvement is more significant when the degrees of the nodes vary a lot (such as the food web and the dolphin network). This can be seen both from the standard error and
For the dolphin network data, the standard error of Blitzstein-Diaconis algorithm is about 47 times larger than that of SIS-CP, which means Blitzstein-Diaconis algorithm needs $47^2 = 2209$ times more samples in order to produce the same standard error as SIS-CP for this example.

In the above comparisons, all three methods used 1000 importance samples in each example, but the comparisons do not take into account the computation time. In the simulations, it appears that SIS-CP is the fastest and Blitzstein-Diaconis algorithm is the slowest among the three. SIS-CP-refined takes a little longer time than SIS-CP because of the construction of the candidate set to guarantee every sample is valid. However it is worth pointing out that the comparison of the computation time is not totally fair here, because Blitzstein-Diaconis algorithm is coded in R while the two SIS-CP based algorithms have the main function coded in R and the drafting sampling part coded in C.

All samples generated by SIS-CP-refined and Blitzstein-Diaconis algorithm are guaranteed to be valid. SIS-CP is the only one among the three that may produce invalid samples, but surprisingly SIS-CP did not generate any invalid graphs from the above simulation with 1000 samples in all three examples. Part of the reason is because we always select the node with the largest degree to sample. For example, if the food web degree sequence is sampled according to its original order given in this section, SIS-CP will generate 4.6% invalid graphs based on 1000 samples. We also noticed that for the degree sequence $d = (3, 3, 3, 3, 3, 7, 7, 7, 7, 7)$, even if we always sample the node with the largest degree, SIS-CP still generates about 60% invalid samples. This is the case that SIS-CP-refined, which produces no invalid samples, can have an much bigger advantage over SIS-CP. Indeed for this degree sequence, SIS-CP-refined gives the smallest standard error and $\text{cv}^2$ based on 1000 samples when the node with the smallest degree is sampled first. Therefore SIS-CP-refined can be useful especially when the percentage of invalid samples is high for SIS-CP.

2.7.2 Testing Exponential Random Graph Models

The exponential random graph model (ERGM) (also known as the $p^*$ model) is one of the most widely used models for social network analysis (Wasserman and Pattison, 1996; Snijders et al., 2006). It is a generalization of the Markov random graph model. The ERGM specifies a probability
distribution on the space of all graphs with \( n \) vertices:

\[
P_{\theta}(G) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^{k} \theta_i u_i(G) \right), \tag{2.34}
\]

where \( Z(\theta) \) is the normalizing constant, \( \theta = (\theta_1, \ldots, \theta_k) \) is the parameter, and the statistics \( u_i(G) \) are counts of graph structures. When the degree sequence is the main feature of interest, then \( u_i(G) \) can be chosen to be the degree of the \( i \)th node \( d_i(G) \) which leads to the following model

\[
P_{\theta}(G) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^{n} \theta_i d_i(G) \right). \tag{2.35}
\]

In order to test the goodness of fit of model (2.35) to the observed data, we can condition on the degree sequence \( (d_1(G), \ldots, d_n(G)) \), which are sufficient statistics for the parameters \( \theta_1, \ldots, \theta_n \). The conditional distribution of \( G \) given its degree sequence \( (d_1(G), \ldots, d_n(G)) \) is uniform over all graphs with this degree sequence. By using the sequential sampling methods we proposed, we can accurately approximate the distribution of any test statistic under the null hypothesis of a uniform distribution conditional on the degree sequence. This frees us to choose a test statistic with good power against the alternatives of interest without worrying about how to find its distribution. Another advantage of this method is that we can approximate the exact distribution of the test statistic for graphs of any size. Most available test statistics are based on asymptotic approximations which are often unreliable for graphs of moderate size.

Another way to test model (2.35) is to put it in a larger family of distributions by adding one more count of graph structure (e.g., the number of triangles) in the exponent (Holland and Leinhardt, 1981). In this case, conditioning on the degree sequence gives the uniformly most powerful unbiased test (Lehmann, 1986; Snijders, 1991).

**Food web data**

Figure 2.2 is the network of the food web for the Chesapeake Bay ecosystem in the summer (Baird and Ulanowicz, 1989). There are 33 types of organisms, each one is represented by a node, and 71
links among them. A link between two nodes means one predates on the other. This is a simplified graph without directed links or loops and it was analyzed in Blitzstein and Diaconis (2010). They considered using the clustering coefficient and $k$-cycles (for $3 \leq k \leq 6$) as test statistics. A cycle is a simple path from a node $x$ to $x$ with no other repeated nodes or edges in the middle. For example $x \rightarrow y \rightarrow z \rightarrow x$ is a 3-cycle. Here we follow Blitzstein and Diaconis (2010) to look at 5-cycles in the graph.

Figure 2.2: Food web of 33 types of organisms in the Chesapeake Bay.

We used SIS-CP to generate 1000 random graphs with the same degree sequence as the food web, and counted the number of 5-cycles in each random graph. The distribution of the number of 5-cycles under the uniform distribution over $\Sigma_d$ can be approximated by the histogram (based on the weighted samples) in Figure 2.3. The number of 5-cycles in the real food web is 153. Using the 1000 weighted samples, we estimated the probability $P(\# \text{ of 5-cycles} \leq 153)$ as $0.051 \pm 0.007$,
which took 4.4 minutes. Based on the same number of samples, SIS-CP-refined took 5.8 minutes to give a very similar estimate and standard error as SIS-CP, while Blitzstein-Diaconis algorithm took 9.4 minutes to give the estimate 0.064 with a larger standard error 0.018. These all indicate that the observed number of 5-cycles in the food web data is quiet low, and further investigation on this phenomenon would be useful.

![Histogram](image)

Figure 2.3: Approximated distribution of the number of 5-cycles based on 1000 weighted samples from SIS-CP. The vertical line indicates the observed number of 5-cycles in the real food web.

The following MCMC algorithm provides an alternative way to sample from the uniform distribution over $\Sigma_d$ and estimate related probabilities (Blitzstein and Diaconis, 2010). At each step of the MCMC algorithm, randomly choose two edges $\{x, y\}$ and $\{u, v\}$ from the current graph $G$ with distinct nodes $x, y, u, v$. If there are no edges between the pair $x$ and $u$ and the pair $y$ and $v$, then the Markov chain moves to a new state $G'$ constructed by replacing the edges $\{x, y\}$ and $\{u, v\}$ from the current graph $G$ by two new edges $\{x, u\}$ and $\{y, v\}$; otherwise, the Markov chain stays at the current graph $G$. The MCMC algorithm took about 70 minutes to generate 10,000 samples and estimated the probability $P(\# \text{ of 5-cycles} \leq 153)$ as 0.069±0.007 (using 7,500 samples
as burn-in). It seems MCMC needs more samples and takes longer time to obtain an estimate with the same accuracy as SIS-CP and SIS-CP-refined.

**Dolphin social network**

For complex networks, such as the Internet, social networks, and power grids, one important question is how the network's connectivity will be affected if some nodes are attacked and removed from the network (Albert, Jeong, and Barabási, 2000; Lusseau, 2003). Two types of attacks are of particular interest. One is random attacks, which mean the nodes are attacked at random. The other is targeted attacks, which mean the nodes with the highest degrees are attacked. It is concluded in Albert et al. (2000) that the Internet and the World Wide Web (WWW) are quiet robust to random attacks but are highly vulnerable to targeted attacks. Interestingly Lusseau (2003) finds that in the network of bottlenose dolphins in a community at Doubtful Sound, New Zealand, the information can still be transferred well between the dolphins even when the community is under targeted attacks. In this section, we carry out a statistical test on the resilience property of the dolphin network.

Figure 1.1 is the social network of 62 bottlenose dolphins that Lusseau (2003) has studied. This particular network is also mentioned in Chapter 1. Two dolphins are linked if they were seen together more frequent than expected. So an edge means these two individuals are closely associated. To measure the interconnectedness of a network, Albert et al. (2000) and Lusseau (2003) use the diameter $\sigma$, which is the average length of the shortest paths between any pair of nodes in the network. A small $\sigma$ indicates that information can be passed quickly within the individuals of the network. It is believed that the human society have a diameter of about six (Milgram, 1967). The diameter of the dolphin social network in Figure 1.1 is 3.36.

To study the robustness of a network against targeted attacks, we can look at the change in diameter when a small percentage of the most connected nodes are removed. For example, the diameter of the Internet more than triples when 2.5% of the most connected nodes are removed (Albert et al., 2000). For the Doubtful Sound dolphin network, surprisingly, the diameter only increases by 5.78% when three individuals (about 5% of the dolphin community) with the most
links are eliminated.

To test the statistical significance of such a small change in diameter for the dolphin network under targeted attacks, we used SIS-CP to generate 1000 random graphs with the same degree sequence as the dolphin network, and computed the change in diameter after removing the same three most connected nodes. Some of the random graphs break into a few isolated parts after the targeted attacks, and their corresponding diameter is treated as infinity. Based on 1000 samples, the probability of seeing 5.78% or less increase in diameter is estimated to be 0.015 ± 0.004, which took about 1 minute. This indicates that the dolphin network behaves very differently from random networks with the same degree sequence. The dolphin network is formed in a way that is resilient to targeted attacks, while random networks tend to have a large increase in diameter or break into isolated fragments. The resilience property of the dolphin network allows the dolphin community to stay cohesive even when 5% of its members with most associates are removed. Further study of the resilience property would be welcome. In particular, if we can build the Internet and the WWW with this property, it will increase the attack tolerance of those networks. In Chapter 3, we will focus on studying the resilience property of networks using exponential random graph models.

2.8 Discussion

We developed two sequential importance sampling strategies for sampling networks with a fixed degree sequence. The first strategy SIS-CP samples the network node by node and uses the conditional Poisson sampling as the proposal distribution. The second strategy SIS-CP-refined builds upon the first strategy and guarantees that the sampling procedure always produces a valid network. SIS-CP is usually faster than SIS-CP-refined, but it may produce some invalid tables. Both algorithms can generate samples very close to the target uniform distribution, and they compare favorably with other existing Monte Carlo algorithms.

In both SIS approaches, we can obtain a rough estimate even with just a few samples and then we can keep going until the required precision level is reached. The same set of samples can be used to estimate \( p \)-values of the conditional inference and the total number of graphs with fixed degrees.
simultaneously. The asymptotic approximation of Bender and Canfield (1978) provides a theoretical justification of the new proposal distributions. The asymptotic behavior of the importance weight shows that the proposed algorithm is still very efficient for large sparse graphs.

Different orderings of the degree sequence can affect the efficiency of both SIS algorithms. In our examples, we found that selecting the node with the largest degree to sample often works the best. One possible explanation is in the proof of Theorem 2.5.1. Note that the term $C$ in (2.47) tends to be small if the first node has the largest degree, and this will lead to a small upper bound for the importance weight. More theoretical analysis of the effect of orderings will be useful. For the food web data, if we always sample the node with the smallest degree, the estimate is $(1.641 \pm 0.077) \times 10^57$ with 68.8% invalid networks based on 1000 samples from SIS-CP, and the $cv^2$ is 2.223. This result is obviously not as good as the one based on sampling the node with the largest degree at each step (see Table 2.1). If we sample the node based on the original order of the food web data (given in Section 2.7.1), the performance is somewhere in between the other two orders. It gives an estimate of $(1.547 \pm 0.015) \times 10^57$ with 4.6% invalid networks based on 1000 samples from SIS-CP, and the $cv^2$ is 0.095. The only exception we have seen is in the application of SIS-CP-refined to the degree sequence $d = (3, 3, 3, 3, 7, 7, 7, 7)$, in which sampling the nodes with the smallest degree is actually doing better than sampling the node with the largest degree. It is an advantage of our algorithms to allow the degree sequence to be sampled in any order.

2.9 Proof of the Main Results

Proof of Corollary 2.4.1

Let $(m_{ij})_{n \times n}$ be an $n \times n$ symmetric zero-one matrix, where $m_{ij} = 0$ means the $(i,j)$th position is a structural zero. Bender and Canfield’s (1978) Theorem 1 states that the number of $n \times n$ symmetric matrices over $[0,t]$ with column sums $d = (d_1, \ldots, d_n)$ and the same set of structural
zeros as \((m_{ij})_{n \times n}\) can be approximated by

\[ h(M, \delta)e^{a-b}/\prod_{j=1}^{n} d_j!, \quad (2.36) \]

where \(M = \sum_{j=1}^{n} d_j, \delta = \sum_{m_{i=0}} d_i, \epsilon = -1\) if \(t = 1\) and \(+1\) if \(t > 1, a = \left(\sum_{j=1}^{n} \frac{(d_j)}{2}/M\right)^2 + \sum_{m_{i=0}} \frac{(d_j)}{2}/M, b = \left(\sum_{m_{i=0}, i<j} d_i d_j + \sum_{n=1}^{n} \frac{(d_j)}{2}\right)/M, h(M, \delta) = \sum \left(M-\delta\right) c_{M-j}, \text{ and } c_j = j!/(j/2) \cdot 2^{j/2}\) if \(j\) is even and \(0\) if \(j\) is odd. For the set \(\Sigma_d\), the matrix \((m_{ij})_{n \times n}\) has 0 on its diagonal and 1 elsewhere, \(\epsilon\) is \(-1\) because \(t = 1, a\) is simplified to \(\sum_{j=1}^{n} \frac{(d_j)}{2}/M^2, b\) is simplified to \(\sum_{j=1}^{n} \frac{(d_j)}{2}/M, \delta\) equals to \(\sum_{j=1}^{n} d_j = M, \text{ and } h(M, \delta) = c_M = M!/[(M/2)!2^{M/2}]\). Plugging these simplifications into expression (2.36) leads to Corollary 2.4.1.

**Proof of Theorem 2.4.1**

Under the uniform distribution over \(\Sigma_d\), the marginal distribution of the first column \(t_1\) is \(p(t_1 = v(i_1, \ldots, i_{d_1})) = \frac{|\Sigma_d|}{|\Sigma_d|}.\) Using the asymptotic approximation for \(|\Sigma_d|\) and \(|\Sigma_d|\) in Corollary 2.4.1, we have \(p(t_1 = v(i_1, \ldots, i_{d_1})) = \frac{|\Sigma_d|}{|\Sigma_d|} \approx \Delta_{d(2)}/\Delta_d \propto \Delta_{d(2)},\) and

\[
\Delta_{d(2)} = \frac{h(M-2d_1)}{\prod_{j=1}^{n} d_j(2)!} \exp \left\{ -\left(\frac{\sum_{j=1}^{n-1} \frac{(d_j)}{2}}{M-2d_1}\right)^2 - \frac{\sum_{j=1}^{n-1} \frac{(d_j)}{2}}{M-2d_1} \right\}
\]

\[
\propto \left(\prod_{k=1}^{d_1} d_{ik}\right) \exp \left\{ -\left(\frac{\sum_{j=1}^{n-1} \frac{(d_j)}{2}}{M-2d_1} + \frac{1}{2}\right)^2 \right\}
\]

\[
= \left(\prod_{k=1}^{d_1} d_{ik}\right) \exp \left\{ -\left(\frac{\sum_{j=2}^{n} d_j^2 - 2\sum_{k=1}^{d_1} d_{ik} d_1}{2(M-2d_1)}\right)^2 \right\}
\]

\[
\propto \left(\prod_{k=1}^{d_1} d_{ik}\right) \exp \left\{ \frac{(\sum_{j=2}^{n} d_j^2 + d_1)(\sum_{k=1}^{d_1} d_{ik}) - (\sum_{k=1}^{d_1} d_{ik})^2}{(M-2d_1)^2} \right\}
\]

The theorem is thus proved.
Proof of Corollary 2.4.2

McKay’s (1985) Theorem 4.6 implies that \(|\Sigma_d|\) is uniformly \([h(M)/\prod_{j=1}^n d_j]! \exp\{-\alpha(d) + O(\bar{d}^2/M)\}\) as \(n \to \infty\), where \(h(M)\) and \(\alpha(d)\) are defined in Corollary 2.4.1, and \(\bar{d} = 2 + d_{\max}(1.5d_{\max} + 1)\). The conditions needed for this theorem are \(d_{\max} \geq 1\) and \(\bar{d} \leq \epsilon_1 M\), where \(\epsilon_1 < 1/3\). Since we require \(d_{\max}\) to be bounded in this corollary, the two conditions in McKay’s theorem are satisfied, and also \(O(\bar{d}^2/M)\) becomes \(O(1/M)\). Now we have

\[
\left| \frac{|\Sigma_d|}{\Delta_d} - 1 \right| = \left| \exp \left\{ O \left( \frac{1}{M} \right) \right\} - 1 \right| = O \left( \frac{1}{M} \right). \tag{2.37}
\]

The approximation error can be written as

\[
\left| \frac{|\Sigma_d(2)|}{|\Sigma_d|} - \frac{\Delta_d(2)}{\Delta_d} \right| = \left| \frac{\Delta_d(2)}{|\Sigma_d|} \left( \frac{|\Sigma_d(2)|}{\Delta_d(2)} - 1 \right) - \frac{\Delta_d(2)}{\Delta_d} \left( \frac{|\Sigma_d|}{\Delta_d} - 1 \right) \right| \leq \frac{\Delta_d(2)}{|\Sigma_d|} \left( \left| \frac{|\Sigma_d(2)|}{\Delta_d(2)} - 1 \right| + \left| \frac{|\Sigma_d|}{\Delta_d} - 1 \right| \right).
\]

Combining (2.37) and the fact that \(\Delta_d(2)/|\Sigma_d| = (\Delta_d(2)/|\Sigma_d(2)|) \cdot (|\Sigma_d(2)|/|\Sigma_d|) \leq \Delta_d(2)/|\Sigma_d(2)| \leq 2\) for \(M\) sufficiently large, we have \(\|\Sigma_d(2)/|\Sigma_d| - \Delta_d(2)/\Delta_d\| = O(1/M)\) as \(M \to \infty\).

Proof of Theorem 2.5.1

The importance weight in (2.23) can be rewritten as

\[
w(T) = \frac{p(T)}{q(T)} = \frac{1}{|\Sigma_d|} \frac{S(d)}{\Lambda_d(2)} \cdots \frac{S(d^{(n-1)})}{\Lambda_d^{(n)}} = \frac{\Lambda_d}{|\Sigma_d|} S(d) / \Lambda_d \cdots \frac{S(d^{(n-1)})}{\Lambda_d^{(n-1)}} \frac{1}{\Lambda_d^{(n)}} = \frac{\Lambda_d}{|\Sigma_d|} \tilde{R}_d \frac{1}{\Lambda_d^{(n)}}, \tag{2.38}
\]

where \(\tilde{R}_d = R_1 \times R_2 \times \cdots \times R_{n-1}\) and \(R_j = S(d^{(j)})/\Lambda_{d^{(j)}}\) for \(j = 1, \ldots, n-1\).

The first term \(\Lambda_d/|\Sigma_d| = (\Delta_d/|\Sigma_d|) \cdot (\Delta_d/\Delta_d)\). It is known from Bender and Canfield (1978) that \(\Delta_d/|\Sigma_d| = O(1)\) (see also Corollary 1). The bounds in (2.21) implies that \(\Lambda_d/\Delta_d = O(1)\) as \(M \to \infty\). Therefore \(\Lambda_d/|\Sigma_d| = (\Delta_d/|\Sigma_d|) \cdot (\Lambda_d/\Delta_d) = O(1)\). The last term \(1/\Lambda_d^{(n)}\) is bounded since \(d^{(n)} = 0\) and \(d^{(n-1)}\) can only be \((1,1)\) or \((0,0)\). So we only need to show that \(\tilde{R}_d\) is bounded as \(M \to \infty\). We will first analyze the asymptotic behavior of \(R_1\) as \(M \to \infty\). Then we will prove that the product of the \(n-1\) terms in \(\tilde{R}_d\) is bounded as \(M \to \infty\).
We first define a few useful notations. For a vector \( s = (s_1, \ldots, s_n) \) of length \( n \), we define, for any positive integer \( k \), \([s^k]_1 = \sum_{j=1}^n s_j^k\), \( s! = s_1! \cdots s_n!\), and \( \xi(s) = (\sum_{j=1}^n s_j^2)^2/[4(\sum_{j=1}^n s_j)^2] \). we know

\[
R_1 = \frac{S(d)}{\Lambda_d} = \frac{\sum_{d \to d^{(2)}} \Lambda_{d^{(2)}}}{\Lambda_d} = \frac{\sum_{d \to d^{(2)}} \Lambda_{d^{(2)}}}{\Delta_d} \cdot \frac{\Delta_d}{\Lambda_d}.
\]

(2.39)

Since \( \Delta_d/\Lambda_d \leq 1 \) based on (2.21), we will focus on the term \( (\sum_{d \to d^{(2)}} \Lambda_{d^{(2)}})/\Delta_d \). Define \( \zeta = (\sum_{j=1}^n d_j^2 - d_1^2 + d_1)/[2(\sum_{j=1}^n d_j - 2d_1)^2] \). Then \( \beta(d_{ik}) \) defined in (2.14) can be written as \( \beta(d_{ik}) = (\sum_{j=2}^n d_j^2 + d_1)d_{ik}/(M - 2d_1)^2 = 2\zeta d_{ik} \), and \( \xi(d) \) can be written as

\[
\xi(d) = \frac{(\sum_{j=1}^n d_j^2)^2}{4(\sum_{j=1}^n d_j)^2} = \frac{\sum_{j=1}^n d_j^2}{4(\sum_{j=1}^n d_j)^2}[d^2]_1 = \left( \frac{\zeta}{2} \left( 1 - \frac{2d_1}{d_1^2} \right)^2 + \frac{d_1^2 - d_1}{4[d_1^2]} \right)[d^2]_1.
\]

Therefore

\[
\sum_{d \to d^{(2)}} \frac{\Lambda_{d^{(2)}}}{\Delta_d} = \sum_{d \to d^{(2)}} \frac{h(M - 2d_1)}{h(M)} \left( \prod_{k=1}^{d_1} d_{ik} \right) \left( e^{\sum_{k=1}^{d_1} 2\zeta d_{ik}} \right) e^{-\zeta ([d_1]^2 - d_1^2 + d_1)} + \xi(d)
\]

(2.40)

In order to study the upper bound for \( \sum_{d \to d^{(2)}} \prod_{k=1}^{d_1} d_{ik} e^{2\zeta d_{ik}} \), we first define \( d_1 \) i.i.d. random variables \( J_1, \ldots, J_{d_1} \) with distribution \( P(J_1 = j) = d_j \exp(2\zeta d_j)/z, \ j = 2, \ldots, n \), where \( z = \sum_{j=2}^n d_j \exp(2\zeta d_j) \). Let \( A = \{J_i \neq J_j : i \neq j\} \) be the event that all the \( J_i \)'s are different.

Then

\[
\sum_{d \to d^{(2)}} \prod_{k=1}^{d_1} d_{ik} e^{2\zeta d_{ik}} = \frac{z^{d_1}}{d_1!} P(A).
\]

(2.41)

We first analyze \( z \). We have

\[
z = \sum_{j=2}^n d_j \left( 1 + 2\zeta d_j + (2\zeta)^2 \frac{d_j^2}{2!} + \cdots \right) \leq ([d_1^2]_1 - d_1) \exp \left( \frac{2\zeta ([d_1^2]_1 - d_1^2)}{|d_1| - d_1} + O\left( \frac{\zeta^2 ([d_1^2]_1 - d_1^2)}{|d_1| - d_1} \right) \right).
\]

(2.42)
Now we estimate \( P(A) \) using the inclusion-exclusion principle.

\[
P(A) \leq 1 - \left( \frac{d_1}{2} \right) \left( \frac{1}{z^2} \sum_{j=2}^{n} d_j^2 \exp(4\zeta d_j) + \frac{3}{1} \right) \left( \frac{d_1}{3} \right) \left( \frac{1}{z^3} \sum_{j=2}^{n} d_j^3 \exp(6\zeta d_j) + \frac{4}{2} \right) \left( \frac{d_1}{4} \right) \left( \sum_{j=2}^{n} d_j^4 \exp(4\zeta d_j) \right)^2.
\]

We analyze each term on the right hand side separately. Under our assumption, we have

\[
\left( \frac{d_1}{2} \right) \left( \frac{1}{z^2} \sum_{j=2}^{n} d_j^2 \exp(4\zeta d_j) \right) = \left( \frac{d_1}{2} \right) \left( \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} \right) \left( 1 + 4\zeta \frac{[d^3]_1 - d_1^3}{d^2_1 - d_1^2} + \frac{4\zeta^2}{2(d)_1 - d_1} \right)
\]

\[
= \left( \frac{d_1}{2} \right) \left( \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} \right) \left( 1 + 4\zeta \frac{[d^3]_1 - d_1^3}{d^2_1 - d_1^2} + \frac{4\zeta^2}{2(d)_1 - d_1} \right)
\]

\[
= \left( \frac{d_1}{2} \right) \left( \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} \right) \left( 1 + 4\zeta \frac{[d^3]_1 - d_1^3}{d^2_1 - d_1^2} + \frac{4\zeta^2}{2(d)_1 - d_1} \right)
\]

Notice that

\[
\frac{[d^3]_1 - d_1^3}{(d)_1 - d_1^2} - \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} = \sum_{2 \leq i < j} d_id_j(d_i - d_j)^2 \geq 0.
\]

Therefore, we have

\[
\left( \frac{d_1}{2} \right) \left( \frac{1}{z^2} \sum_{j=2}^{n} d_j^2 \exp(4\zeta d_j) \right) \leq \left( \frac{d_1}{2} \right) \left( \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} \right) \left( 1 + O \left( \frac{[d^4]_1 - d_1^4}{d^2_1 - d_1^2} \right) \right)
\]

\[
= \left( \frac{d_1}{2} \right) \left( \frac{[d^2]_1 - d_1^2}{(d)_1 - d_1} \right) + O \left( \frac{[d^4]_1}{d^2_1} \right).
\]

We can also find an upper bound as

\[
\left( \frac{d_1}{2} \right) \left( \frac{1}{z^3} \sum_{j=2}^{n} d_j^3 \exp(6\zeta d_j) \right) \leq \left( \frac{d_1}{3} \right) \left( \frac{[d^3]_1 - d_1^3}{(d)_1 - d_1} \right) \left( 1 + 6\zeta \frac{[d^4]_1 - d_1^4}{d^3_1 - d_1^3} + O \left( \frac{[d^5]_1}{d^3_1} \right) \right).
\]

Using the same technique, it is easy to get

\[
\left( \frac{d_1}{3} \right) \left( \frac{1}{z^3} \sum_{j=2}^{n} d_j^3 \exp(6\zeta d_j) \right) \leq \left( \frac{d_1}{3} \right) \left( \frac{[d^3]_1 - d_1^3}{(d)_1 - d_1} \right) \left( 1 + 6\zeta \frac{[d^4]_1 - d_1^4}{d^3_1 - d_1^3} + O \left( \frac{[d^5]_1}{d^3_1} \right) \right).
\]

37
As a consequence,

\[
P(A) \leq 1 - \left( \frac{d_1}{2} \right) \left[ \frac{d_1^2 - d_1^2}{(d_1 - d_1)^2} \right]^3 + 3 \left( \frac{d_1}{3} \right) \left[ \frac{d_1^3 - d_1^3}{(d_1 - d_1)^3} \right] \left( 1 + 6 \zeta \left( \frac{d_1^4 - d_1^4}{(d_1^3 - d_1^3)} \right) \right) + O\left( \frac{d_1^2}{d_1^2} \right)
\]

\[
+ 6 \left( \frac{d_1}{4} \right) \left[ \frac{d_1^2 - d_1^2}{(d_1 - d_1)^2} \right]^2 \left( 1 + 4 \zeta \left( \frac{d_1^3 - d_1^3}{d_1^2 - d_1^2} \right) \right) + O\left( \frac{d_1^4}{d_1^4} \right)
\]

\[
= 1 - \left( \frac{d_1}{2} \right) \left[ \frac{d_1^2 - d_1^2}{(d_1 - d_1)^2} \right] + O\left( \frac{d_1^3}{d_1^3} \right) \leq \exp \left( - \left( \frac{d_1}{2} \right) \left[ \frac{d_1^2 - d_1^2}{(d_1 - d_1)^2} \right] \right) + O\left( \left( \frac{d_1^3}{d_1^3} \right) \right) \quad (2.46)
\]

Now combining (2.40), (2.41), (2.42) and (2.46), we have

\[
\sum_{d \to d^{(2)}} \frac{A_d^{(2)}}{\Delta_d} = \frac{h(M - 2d_1)}{h(M)} \left( \frac{z_{d_1}^1}{d_1!} P(A) \right) \exp \left\{ \left( \frac{\zeta}{2} + \frac{[d_1^2]^1}{4[d_1^2]} \right) (d_1^2 - d_1) - \frac{2\zeta d_1 [d_1^2]^1}{[d_1]} + \frac{2\zeta d_1^2 [d_1^2]^1}{[d_1]^2} \right\}
\]

\[
\leq \frac{h(M - 2d_1)}{h(M)} \left( \frac{[d_1^2]^1}{[d_1] - d_1} \right) \exp \left( \left( \frac{\zeta}{2} + \frac{[d_1^2]^1}{4[d_1^2]} \right) (d_1^2 - d_1) - \frac{2\zeta d_1 [d_1^2]^1}{[d_1]} + \frac{2\zeta d_1^2 [d_1^2]^1}{[d_1]^2} \right) \exp \left\{ \left( \frac{\zeta}{2} + \frac{[d_1^2]^1}{4[d_1^2]} \right) (d_1^2 - d_1) - \frac{2\zeta d_1 [d_1^2]^1}{[d_1]} + \frac{2\zeta d_1^2 [d_1^2]^1}{[d_1]^2} \right\}
\]

\[
= \exp\{A + B + C + D\},
\]

where

\[
A = \log \left( \frac{h(M - 2d_1)}{h(M)} \right) \left( \frac{[d_1^2]^1}{[d_1] - d_1} \right), \quad B = \left( \frac{\zeta}{2} + \frac{[d_1^2]^1}{4[d_1^2]} \right) (d_1^2 - d_1) - \left( \frac{d_1}{2} \right) \left( \frac{[d_1^2]^1}{[d_1] - d_1} \right),
\]

\[
C = \frac{2\zeta d_1 [d_1^2]^1}{[d_1] - d_1} - \frac{2\zeta d_1^2 [d_1^2]^1}{[d_1]} \quad \text{and} \quad D = \frac{2\zeta d_1^2 [d_1^2]^1}{[d_1]^2} + \frac{[d_1^2]^1}{4[d_1^2]} + O\left( \frac{[d_1^3]^1}{[d_1]^2} \right) + O\left( \frac{\zeta^2 d_1 [d_1^3]^1}{[d_1] - d_1} \right).
\]

We analyze each term separately. From Theorem 2 of Bender and Canfield (1978), we have \( h(M) \sim \sqrt{2} (M/e)^{M/2} \). In other words, \( \lim_{M \to \infty} \left| h(M) / [\sqrt{2} (M/e)^{M/2}] - 1 \right| = 0 \). Therefore we can write \( A \) as

\[
A = \log \left( \frac{\sqrt{2} (M - 2d_1)/e^{(M-2d_1)/2}}{\sqrt{2} (M/e)^{M/2}} \right) \left( \frac{[d_1]^1 - d_1}{d_1!} \right) \frac{h(M - 2d_1)}{h(M)} \sqrt{2} (M/e)^{M/2} \]

\[
= d_1 + \left( \frac{[d_1]^1}{2} \right) \log \left( \frac{[d_1]^1 - 2d_1}{[d_1]^1} \right) - d_1 \log \left( \frac{[d_1]^1 - 2d_1}{[d_1]^1 - d_1} \right) - \log d_1! + o(1)
\]

\[
= -O\left( \frac{d_1^2}{2[d_1]^1} \right) - d_1 \log \left( \frac{[d_1]^1 - 2d_1}{[d_1]^1 - d_1} \right) - \log d_1! + o(1).
\]
Obviously $A \leq 0$ when $M = [d]_1 \to \infty$. Next we analyze $B$. We have

\[
B \leq (d_1^2 - d_1) \left( \frac{\sum_{j=1}^{n} d_j^2}{4(\sum_{j=1}^{n} d_j - 2d_1)^2} + \frac{[d_1^2]_1}{4(\sum_{j=1}^{n} d_j - 2d_1)^2} - \frac{[d_1^2]_1 - d_1^2}{2([d_1^2]_1 - d_1)^2} \right) = (d_1^2 - d_1) \left( \frac{d_1(2[d_1^2]_1 - 3d_1)[d_1^2]_1}{2([d_1^2]_1 - d_1)^2} + \frac{d_1^2}{2([d_1^2]_1 - d_1)^2} \right) = O \left( \frac{1}{[d_1^2]_1} \right).
\]

For $C$ we have,

\[
C = 2\zeta d_1 \left( \frac{[d_1^2]_1 - d_1^2}{[d_1^2]_1 - d_1} - \frac{[d_1^2]_1}{[d_1^2]_1} \right) = 2\zeta d_1 \frac{d_1([d_1^2]_1 - d_1[1_d])}{([d_1^2]_1 - d_1)[d_1]} = O \left( \frac{1}{[d_1^2]_1} \right),
\]

because $\zeta = O \left( 1/[d_1^2]_1 \right)$. When $d_1$ is the largest column sum, we have

\[
C = 2\zeta d_1 \frac{d_1 \sum_{j=1}^{n} d_j(d_j - d_1)}{([d_1^2]_1 - d_1)[d_1]} \leq 0. \quad (2.47)
\]

For $D$, it is not hard to see that $D = O \left( 1/[d_1^2]_1 \right)$ because $\zeta = O \left( 1/[d_1^2]_1 \right)$.

From (2.39), we know (note that $\Delta_d/\Lambda_d \leq 1$, and $A \leq 0$ when $M = [d]_1 \to \infty$)

\[
\log(R_1) = \log \left( \frac{\sum_{d \to d(2)} \Lambda_d}{\Delta_d} \right) + \log \left( \frac{\Delta_d}{\Lambda_d} \right) \leq \log \left( \frac{\sum_{d \to d(2)} \Lambda_d}{\Delta_d} \right) = A + B + C + D \leq B + C + D = O \left( \frac{1}{[d_1^2]_1} \right).
\]

Therefore, there exists a constant $\rho > 0$ and an integer $N_\rho > 0$, such that for any $j$ with $[d^{(j)}]_1 \geq N_\rho$, we have

\[
R_j \leq \exp \left( \rho \frac{1}{[d^{(j)}]_1^2} \right). \quad (2.48)
\]

Define $\max_j [d^{(j)}]_1 \geq N_\rho$ as $J_\rho$. Then for $j > J_\rho$, we have $[d^{(j)}]_1 < N_\rho$. This indicates that after filling in the $J_\rho$-th column of the table, there are only a finite number of 1’s to fill in the subtable, and this procedure can be done in a finite number of steps. Therefore

\[
R_{J_\rho + 1} \times R_{J_\rho + 2} \times \cdots \times R_{n-1} = \prod_{j=J_\rho+1}^{n-1} \frac{\sum_{d \to d^{(j+1)}} \Lambda_d}{\Lambda_{d^{(j)}}}.
\]

39
is a finite number. Therefore, there exist $\kappa_0$ such that

$$R_{\mu+1} \times R_{\mu+2} \times \cdots \times R_{n-1} \leq \kappa_0 \exp \left( \rho \sum_{j=\mu+1}^{n-1} \frac{1}{|d(j)|^2} \right).$$

(2.49)

Combining (2.48) and (2.49), we have

$$\tilde{R}_d = R_1 \times R_2 \times \cdots \times R_{n-1} \leq \kappa_0 \exp \left( \rho \sum_{j=1}^{n-1} \frac{1}{|d(j)|^2} \right) \leq \kappa_0 \exp \left( \rho \sum_{j=1}^{\infty} \frac{1}{j^2} \right) = \kappa_0 \exp \left( \frac{\rho \pi^2}{6} \right).$$

The last inequality holds because $[d(j)]_1 - [d(j+1)]_1 \geq 2$ for $d_1^{(j)} \neq 0$. Since $\tilde{R}_d$ is bounded as $M \to \infty$, it follows from (2.38) that the importance weight $w(T)$ is bounded as $M \to \infty$.

**Proof of Theorem 2.6.4**

We first extend SIS-CP to the larger space $\Gamma_d$ and denote the corresponding importance weight as $\tilde{w}(\gamma)$, and then we show that $w(T(\gamma)) = \tilde{w}(\gamma) \geq w_r(\gamma)$.

The drafting sampling method in SIS-CP samples the positions to put 1’s in one by one to obtain an ordered list of positions in each column. The SIS-CP on the space $\Sigma_d$ treats the list of positions as unordered and computes the probability using (2.15). When we extend SIS-CP to the larger space $\Gamma_d$, we can simply treat the list of positions from drafting sampling as ordered, and compute the corresponding sampling probability as the product of a sequence of probabilities in (2.17). For example, the sampling probability of an ordered list of positions $(i_1, \ldots, i_{d_1})$ in the first column is

$$\prod_{k=1}^{d_1} P(i_k, A_{k-1}^c) = \prod_{k=1}^{d_1} \frac{w_{i_k} R(d_1 - k, A_{k-1}^c \setminus i_k)}{(d_1 - k + 1) R(d_1 - k + 1, A_{k-1}^c)} = \frac{\prod_{k=1}^{d_1} w_{i_k}}{d_1! R(d_1, A_0^c)}. \quad (2.50)$$

Note that this probability is the same for all $d_1!$ possible orders of the list. This property is true for other columns as well. Therefore, when we extend SIS-CP to $\Gamma_d$, the sampling distribution is
\( \tilde{q}(\gamma) = q(T(\gamma))/c(\gamma), \gamma \in \Gamma_d \), where \( c(\gamma) \) is defined in (2.31). The importance weight is

\[
\tilde{w}(\gamma) = \frac{\tilde{p}(\gamma)}{\tilde{q}(\gamma)} = \frac{p(T(\gamma))/c(\gamma)}{q(T(\gamma))/c(\gamma)} = w(T(\gamma)).
\] (2.51)

So we just need to show \( w_r(\gamma) \leq \tilde{w}(\gamma) \).

Since \( w_r(\gamma) = \tilde{p}(\gamma)/q_r(\gamma) \) and \( \tilde{w}(\gamma) = \tilde{p}(\gamma)/\tilde{q}(\gamma) \), it is enough to show \( q_r(\gamma) \geq \tilde{q}(\gamma) \) for \( \gamma \in \Gamma_d \). Notice that \( q_r(\gamma) \) and \( \tilde{q}(\gamma) \) are a product of a sequence of probabilities in (2.26) and (2.17) respectively. The number of terms in \( q_r(\gamma) \) and \( \tilde{q}(\gamma) \) are the same, and each term is the probability of selecting a position from the candidate set \( C_k \) or from the set of all the unselected positions \( \tilde{A}^c_{k-1} \). Here \( C_k \subset A^c_{k-1} \). If we can show that every term in \( q_r(\gamma) \) is greater than or equal to the corresponding term in \( \tilde{q}(\gamma) \), then we immediately have \( q_r(\gamma) \geq \tilde{q}(\gamma) \).

Without loss of generality, let us look at the term for sampling the \( k \)-th position of the first column, and we want to show

\[
P(j, C_k) = \frac{w_j R(d_1 - k, C_k \setminus j)}{(d_1 - k + 1)R(d_1 - k + 1, C_k)} \geq \frac{w_j R(d_1 - k, A^c_{k-1} \setminus j)}{(d_1 - k + 1)R(d_1 - k + 1, A^c_{k-1})} = P(j, A^c_{k-1}),
\] (2.52)

Based on the definition of \( R(s, A) \) in (2.18), we have \( R(d_1 - k + 1, A^c_{k-1}) = w_j R(d_1 - k, A^c_{k-1} \setminus j) + R(d_1 - k + 1, A^c_{k-1} \setminus j) \). Using this formula, we can rewrite \( P(j, C_k) \) and \( P(j, A^c_{k-1}) \) as

\[
\frac{w_j}{(d_1 - k + 1) \left( w_j + \frac{R(d_1 - k + 1, C_k \setminus j)}{R(d_1 - k, C_k \setminus j)} \right)} \quad \text{and} \quad \frac{w_j}{(d_1 - k + 1) \left( w_j + \frac{R(d_1 - k + 1, A^c_{k-1} \setminus j)}{R(d_1 - k, A^c_{k-1} \setminus j)} \right)},
\] (2.53)

respectively. Therefore in order to show (2.52), we only need to show

\[
R(d_1 - k + 1, C_k \setminus j)R(d_1 - k, A^c_{k-1} \setminus j) \leq R(d_1 - k + 1, A^c_{k-1} \setminus j)R(d_1 - k, C_k \setminus j).
\] (2.54)

Each term in the right hand side of (2.54) has the form \( w_{i_1}^2 \cdots w_{i_r}^2 w_{i_{r+1}} w_{i_{r+2}} \cdots w_{i_{2d_1-2k+1-1}} \),

where \( w_{i_1}, \ldots, w_{i_r} \) are the terms chosen by both \( R(d_1 - k + 1, A^c_{k-1} \setminus j) \) and \( R(d_1 - k, C_k \setminus j) \), \( w_{i_{r+1}}, \ldots, w_{i_{2r}} \) are the terms from \( R(d_1 - k + 1, A^c_{k-1} \setminus j) \) that are in \( A^c_{k-1} \) but not in \( C_k \), and \( w_{i_{2r+1}}, \ldots, w_{i_{2d_1-2k+1-1}} \) are in \( C_k \) and \( d_1 - k + 1 - l - r \) of them are from \( R(d_1 - k + 1, A^c_{k-1} \setminus j) \) and the other \( d_1 - k - l \).
are from $R(d_1 - k, C_k \setminus j)$. This specific term appears $\binom{2d_1 - 2k - 2l - r + 1}{d_1 - k - l + 1}$ times in the right hand side of (2.54). It can be derived in the same way that this term appears $\binom{2d_1 - 2k - 2l - r + 1}{d_1 - k - l + 1}$ times in the left hand side of (2.54). Since

$$\frac{\binom{2d_1 - 2k - 2l - r + 1}{d_1 - k - l + 1}}{\binom{2d_1 - 2k - 2l - r + 1}{d_1 - k - l}} = \frac{d_1 - k - l - r + 1}{d_1 - k - l + 1} \leq 1,$$

the inequality in (2.54) holds, which implies $w_r(\gamma) \leq w(T(\gamma))$ based on the aforementioned arguments.
Chapter 3

Exponential Random Graph Models for Networks Resilient to Targeted Attacks

3.1 Introduction

In complex networks, such as the Internet and social networks, sometimes a number of nodes may be removed from the network due to the attack from outside or the failure of the nodes. One example is from the security breaches in the cyber space. When the nodes of the network are attacked or compromised, one way to prevent the attack from percolating through other nodes in the network is to disconnect the compromised nodes from the network. The separation of the nodes under attack can avoid further damage on these nodes and allow for investigation of security breaches. The removal of some of the nodes will certainly affect the communication within the network. Therefore one important question is how the network’s connectivity will be affected after some nodes are attacked and removed from the network.

Two types of attacks have been studied in the literature: random attacks and targeted attacks. Random attacks mean the nodes are attacked at random, and targeted attacks mean the nodes with the highest degrees (i.e., with the most links) are attacked. It is well known that the Internet and the World Wide Web (WWW) are quite robust to random attacks, but are highly vulnerable to targeted attacks (Albert et al., 2000). If 2.5% of the most connected nodes in the Internet are removed, then the diameter (average length of the shortest paths between any pair of nodes) of the Internet more than triples (Albert et al., 2000). Surprisingly Lusseau (2003) found that for the network of 62 bottlenose dolphins in a community at Doubtful Sound, New Zealand, the diameter only increases by 5.78% when 5% of the dolphins with the most links are removed. Using the efficient sequential sampling algorithm developed in Chapter 2, we concluded that such a small
change in diameter is statistically significant. This indicates that the dolphin network is formed in a particular way (instead of randomly linked with each other) that is resilient to targeted attacks.

There has been an increasing interest in the study of targeted attacks on real-world networks. Many real-world networks are extremely vulnerable to the removal of the most connected nodes (Albert et al., 1999). The most connected nodes play an important role in maintaining the network’s connectivity. The malfunction of the key nodes would result in a significant loss of the information carrying ability of the network. It is important to understand how to design networks that are robust to such attacks.

Most of the existing approaches for studying attack tolerance rely on analyzing one or two statistics of the network, such as the diameter, global efficiency, local efficiency, clustering coefficient, or the size of the largest connected cluster (Albert et al., 2000; Crucitti et al., 2003; Schneider et al., 2011). In order to better understand the resilience property of the dolphin network and find statistical models that characterize network resiliency, we fit the dolphin network with the exponential random graph model (ERGM), which is one of the most widely used models for social network analysis (Wasserman and Pattison, 1996; Robins et al., 2007a). The ERGM involves a set of local structures of the network, so fitting an ERGM can help us understand what kind of local features can contribute to the global resilience property. Such a statistical model can also be used to build the Internet and other networks to increase the attack tolerance of those networks.

This chapter is organized as follows. Section 3.2 introduces the global efficiency measure and explains the resilience property of the dolphin network. Section 3.3 reviews the ERGM and model fitting procedure. Section 3.4 discusses the constraints we put on the network space. Section 3.5 fits the ERGM to the dolphin network. Section 3.6 studies the attack tolerance of the fitted model, and Section 3.7 provides concluding remarks.

### 3.2 Resilience Property of the Dolphin Network

When we discuss the attack tolerance of the Internet, the World Wide Web and the bottlenose dolphins in Section 3.1, we used the change of the diameter after the attack as the measure. The
diameter of a network is the average of the minimum distances between any pair of nodes in the network. It is a well-studied, important network metric because it is one of the metrics that characterize the small world property of networks. However, the diameter is not well defined for networks that are not connected. This is problematic because in practice some networks consist of several isolated fragments or become disconnected after attacks. In that case, it is often up to the researchers to redefine diameters.

Recently another measure called global efficiency is proposed to characterize the small world property of networks (Latora and Marchiori, 2001). For a network $G$ with $n$ nodes, its global efficiency is defined as

$$E(G) = \frac{1}{n(n-1)} \sum_{i \neq j \in G} \frac{1}{d_{ij}},$$  \hspace{1cm} (3.1)$$

where $d_{ij}$ is the length of the shortest path between nodes $i$ and $j$. The global efficiency is closely related to the diameter because the diameter is the average of $d_{ij}$ instead of $1/d_{ij}$. If nodes $i$ and $j$ are disconnected, then $d_{ij} = \infty$ and $1/d_{ij} = 0$, so global efficiency is well defined for disconnected networks as well. The global efficiency is always between 0 and 1, with $E(G) = 0$ for an empty graph with no edges between its nodes and $E(G) = 1$ for a complete graph with all the $n(n-1)/2$ possible edges.

The global efficiency is shown to be a better measure than the diameter for describing the global properties of complex networks, especially when a large number of nodes are removed (Crucitti et al., 2003). Therefore we will use the percentage of global efficiency change after the attack to measure network resilience. It is shown in Crucitti et al. (2003) that scale-free networks are extremely vulnerable to targeted attacks in terms of the global efficiency. We looked at the global efficiency change for two real data sets. The first is the Internet router-level network based on the ITDK0304 skitter data between April 21 and May 8 of 2003. The data is available at the web page of the Cooperative Association for Internet Data Analysis (http://www.caida.org/tools/measurement/skitter/router_topology/). This network contains 192,244 nodes and 609,066 undirected edges. After removing 2.5% most connected nodes, the global efficiency reduced from 0.1501 to 0.0696, which is a decrease of 53.63%. The second data is a subset of the WWW containing 325,729 nodes and
1,090,108 undirected edges (Albert et al., 1999). After a 2.5% targeted attack, the global efficiency reduced from 0.1535 to 0.0189, which is a decrease of 87.69%. Such vulnerability to targeted attacks is also observed in simulated scale-free networks similar to the Internet and the WWW (Crucitti et al., 2003).

![Histogram of global efficiency changes](image)

Figure 3.1: The histogram of the percentage of global efficiency changes based on 1000 random samples. The vertical line indicates the value calculated from the observed dolphin network.

The bottleneck dolphin network has 62 nodes and 159 edges with a global efficiency of 0.3792. After removing three most connected individuals (about 5% of the community), the global efficiency becomes 0.3585 which only decreases by 5.459%. This is a very small change comparing to the behavior of other complex real world networks. It also shows that under the global efficiency measure, the dolphin network is still resilient to targeted attacks. To test the statistical significance of this small change in global efficiency, we compared the dolphin network with random networks having the same degree sequence as the dolphin network. Totally 1000 random networks were generated using the sequential importance sampling algorithm developed in Chapter 1, and for each network the percentage of global efficiency change is computed after the removal of the three most connected nodes. The histogram of the 1000 values of the percentage of global efficiency change is given in Figure 3.1, and the probability of having a change of global efficiency less than
or equal to 5.459% is estimated to be 0.0152 with standard error 0.0039. This shows that the dolphin network is formed in a way that has very high attack tolerance comparing to other random networks with the same degree sequence. In this chapter, we fit a statistical model to the dolphin network to understand its resilience property.

### 3.3 Exponential Random Graph Models

A network (or graph) $G$ with $n$ vertices (or nodes) $V$ and a set of edges (or links) $E$ can be represented by its adjacency matrix $y$, where $y_{ij} = 1$ if there is an edge from node $i$ to node $j$ and 0 otherwise. The degree of a node is the number of edges incident to the node. We use $\{i, j\}$ to denote an edge between node $i$ and node $j$. In this chapter, we are mainly concerned with simple undirected graphs (no loops or multiple edges) because the dolphin network is of this type. Therefore $y$ is an $n \times n$ symmetric 0-1 matrix with a zero diagonal.

The exponential random graph model (ERGM) specifies a probability distribution on the space $\mathcal{Y}$ of all graphs under consideration

$$
P_\theta(Y = y) = \frac{\exp\{\theta^T g(y)\}}{\kappa(\theta)},
$$

where $\kappa(\theta)$ is the normalizing constant, $\theta = (\theta_1, \ldots, \theta_p)$ is the parameter, and the statistics $g(y) = (g_1(y), \ldots, g_p(y))$ are counts of graph structures or features of the network. Sometimes $g(y)$ can also incorporate additional covariates of the network. The normalizing constant $\kappa(\theta)$ usually cannot be computed explicitly even for a moderate size graph because it involves the summation over all $y \in \mathcal{Y}$.

ERGMs have been used extensively in the study of networks (Wasserman and Pattison, 1996; Robins et al., 2007a, 2007b). The statistics $g(y)$ often include a set of local structures of the networks. Some local rules can describe the transitivity of the network, such as the ratio of the number of triangles to the number of two-stars. Some can provide information on how well the network conveys information, such as the total number of edges. To fit an ERGM, we need to
identify a subset of local measures that can concisely summarize the global property of a network. A well fitted ERGM can help us understand how the global structure can be reproduced by local metrics and how local rules could affect the global property of a network. We can also use the fitted ERGM to design networks with certain properties.

3.3.1 Network Statistics

Although basic local structures, such as star counts, triangle counts, and the degree distribution, are traditional candidates for the local measures (Frank and Strauss, 1986), it is pointed out in Snijders et al. (2006) that including such basic terms could result in a probability model which concentrates its mass at either the full graph or the empty graph. This so called “degeneracy” phenomenon makes it very difficult to have reasonable parameter estimation, and places a serious barrier between specifying a reasonable ERGM and making reliable parameter estimation. However, the degeneracy issue is caused not by the ERGM itself, but by the network statistics chosen to be included in the model (Snijders et al., 2006). Hunter (2007) discussed three new network metrics: geometrically weighted degree (GWD), geometrically weighted edgewise shared partner (GWESP), and geometrically weighted dyadwise shared partner (GWDSP). These new statistics not only help avoid the degeneracy problem, but also provide insight on network structures from a different perspective. The definitions of these statistics are given below.

For a network with $n$ nodes and an $n \times n$ adjacency matrix $y$, let $D_i(y)$ be the number of nodes in $y$ with $i$ edges. Then $D_i(y)$, $i = 0, \ldots, n-1$, are the degree distribution of $y$, and they satisfy the linear constraint $D_0(y) + \cdots + D_{n-1}(y) = n$. For a given $i$, let $EP_i(y)$ be the number of edges $\{k, l\}$ such that nodes $k$ and $l$ are linked through an edge (i.e., nodes $k$ and $l$ are neighbors) and they share exactly $i$ partners in common (i.e., there are exactly $i$ nodes that are linked to both nodes $k$ and $l$). Then $EP_i(y)$, $i = 0, \ldots, n-2$, are the edgewise shared partner distribution of $y$, and the sum $EP_0(y) + \cdots + EP_{n-2}(y)$ equals the total number of edges in the graph. For a given $i$, let $DP_i(y)$ be the number of dyads $(k, l)$ such that nodes $k$ and $l$ share exactly $i$ partners in common. Here the dyad $k$ and $l$ do not need to be neighbors of each other. Then $DP_i(y)$, $i = 0, \ldots, n-2$, are the dyadwise shared partner distribution of $y$, and the sum $DP_0(y) + \cdots + DP_{n-2}(y)$ equals the total number
of dyads in the graph. For a given $i$, define the non-edgewise shared partner $NSP_i(y)$ as $DP_i(y) - EP_i(y)$, which equals the number of dyads in the network that are not connected but share exactly $i$ partners in common. For example, in Figure 3.2, the degree distribution is $(D_0(y), \ldots, D_4(y)) = (0, 0, 1, 4, 0)$, the edgewise shared partner distribution is $(EP_0(y), \ldots, EP_3(y)) = (2, 4, 1, 0)$, the dyadwise shard partner distribution is $(DP_0(y), \ldots, DP_3(y)) = (2, 4, 3, 1)$, and the non-edgewise shared partner distribution is $(NSP_0(y), \ldots, NSP_3(y)) = (0, 0, 2, 1)$.

Figure 3.2: An illustrative example of the network.

Based on the above terms, the statistics $GWD$, $GWESP$, $GWDSP$ and geometrically weighted non-edgewise shared partner ($GWNSP$) are defined as:

\[
GWD = e^{\tau_1} \sum_{i=1}^{n-1} \{1 - (1 - e^{-\tau_1})^i\} D_i(y),
\]

\[
GWESP = e^{\tau_2} \sum_{i=1}^{n-2} \{1 - (1 - e^{-\tau_2})^i\} EP_i(y),
\]

\[
GWDSP = e^{\tau_3} \sum_{i=1}^{n-2} \{1 - (1 - e^{-\tau_3})^i\} DP_i(y),
\]

\[
GWNSP = e^{\tau_4} \sum_{i=1}^{n-2} \{1 - (1 - e^{-\tau_4})^i\} NSP_i(y).
\]

Here $\tau_i \geq 0$, $i = 1, \ldots, 4$, are decay parameters. When $\tau_2 = \tau_3 = \tau_4$, it is easy to see that $GWDSP = GWESP + GWNSP$. The intuition behind the four geometrically weighted metrics is
to constrain the effect of higher order terms in the summation and control the degeneracy problem. For example, the number of triangles is a problematic term in the ERGM as we discussed before. For a complete graph with 6 nodes (every pair of nodes are connected), there are 20 triangles. If we remove one edge from the graph, the number of triangles will decrease from 20 to 16. This is a big change for the deletion of a single edge. For large networks, the number of triangles is even more sensitive to the removal of edges. Based on the definition of the edgewise shared partner distribution, we can write the number of triangles as \( \frac{1}{3} \sum_{i=1}^{n-2} iEP_i(y) \). In this expression, the coefficient for \( EP_i(y) \) is \( i/3 \) which increases linearly with the index \( i \), while in the GWESP, the coefficient for \( EP_i(y) \) is \( e^{-\tau^2} \{1 - (1 - e^{-\tau^2})^i\} \) which is bounded above by \( e^{-\tau^2} \) for all \( i \). Compared to the number of triangles, the GWESP puts less weight on higher order terms and is therefore less sensitive to small changes in the graph. As explained in Snijders et al. (2006), a model with these geometrically weighted metrics can avoid the model degeneracy problem and capture the higher order dependency structure in the network. More details about these four metrics can be found in Snijders et al. (2006) and Hunter (2007).

### 3.3.2 Model Fitting

We consider estimating the parameters in the ERGM by the maximum likelihood method. Because the analytical form of the maximum likelihood estimate (MLE) is not available for the ERGM, finding the MLE is normally done with either Markov chain Monte Carlo maximum likelihood estimation (MCMCMLE) (Geyer and Thompson, 1992; Snijders, 2002) or maximum pseudo-likelihood estimation (MPLE) (Frank and Strauss, 1986; Strauss and Ikeda, 1990). Although the MPLE is easier to implement, it assumes independence of all dyads in the network and therefore can produce unreliable estimates. In this chapter, we use MCMCMLE to make inference on the ERGM.

The idea of the MCMCMLE method can be summarized as follows. From (3.2), we have the log-likelihood function

\[
l(\theta) = \theta^T g(y) - \log \kappa(\theta).
\]  

(3.7)

Directly maximizing \( l(\theta) \) involves the intractable normalizing constant \( \kappa(\theta) \). Therefore, we instead
consider the following log-ratio of likelihoods

\[ l(\theta) - l(\theta_0) = (\theta - \theta_0)^T g(y) - \log \left( \frac{\kappa(\theta)}{\kappa(\theta_0)} \right) \],

\( (3.8) \)

where \( \theta_0 \) is an arbitrarily chosen parameter value. Notice that

\[ \frac{\kappa(\theta)}{\kappa(\theta_0)} = \sum_{y^* \in Y} \exp\{ (\theta - \theta_0)^T g(y^*) \} \frac{\exp\{ \theta_0^T g(y^*) \}}{\kappa(\theta_0)} = E_{\theta_0}[\exp\{ (\theta - \theta_0)^T g(y^*) \}]. \]

\( (3.9) \)

So if we can draw \( m \) random samples \( y^*_1, \ldots, y^*_m \) from \( P_{\theta_0} \), say through a Markov chain, we can approximate \( \kappa(\theta)/\kappa(\theta_0) \) by the sample mean

\[ \frac{1}{m} \sum_{i=1}^{m} \exp\{ (\theta - \theta_0)^T g(y^*_i) \}. \]

\( (3.10) \)

Then the log-ratio of likelihoods can be approximated by

\[ l(\theta) - l(\theta_0) \approx (\theta - \theta_0)^T g(y) - \log \left( \frac{1}{m} \sum_{i=1}^{m} \exp\{ (\theta - \theta_0)^T g(y^*_i) \} \right). \]

\( (3.11) \)

By maximizing (3.11), we can obtain an approximation to the MLE. The convergence property of the approximate MLE is discussed in Geyer and Thompson (1992).

The algorithm for implementing MCMCMLE can be summarized as follows.

1. Pick a starting value \( \theta_0 \).

2. Draw random samples from \( P_{\theta_0} \) through a Markov chain, and based on these samples, maximize (3.11) using the Newton-Raphson method.

3. Set the maximizer in Step 2 as \( \theta_0 \) and go back to Step 2, or stop if the desired number of iterations is reached.

In Step (2), we need to construct a Markov Chain with \( P_{\theta_0}(Y = y) \) as the stationary distribution. This can be realized by the Metropolis-Hastings algorithm. We may start the Markov chain with the observed network, and then propose a move (or an adjustment) of the current network at each
step and use the Metropolis-Hastings rule to decide whether the Markov chain will go to the new network or stay at the current network. One requirement is that the Markov chain should be able to reach every network \( y \in \mathcal{Y} \). For example, if \( \mathcal{Y} \) is all simple undirected graphs with \( n \) nodes, the move could be adding an edge between two unconnected nodes or removing an existing edge between two nodes. If \( \mathcal{Y} \) contains all simple undirected graphs with a given degree sequence, Blitzstein and Diaconis (2010) mentioned the following moves: Choose two edges \( \{u_1, v_1\} \) and \( \{u_2, v_2\} \) on four different nodes, and replace them by two new edges \( \{u_1, u_2\} \) and \( \{v_1, v_2\} \) if these two edges do not exist yet. In this chapter, fitting and simulating from the ERGMs are done through the R package “ergm” (version 2.43) (Hunter et al., 2008b).

3.4 Constraints on the Network Space

In this section, we specify the space \( \mathcal{Y} \) of all networks of interest in the ERGM (3.2). Usually the number of nodes \( n \) is fixed. In that case, the number of edges in the network plays an important role in network resilience. In general, if we add more edges to the network, it will increase the global efficiency of the network. An extreme case is the complete graph whose global efficiency is 1 before and after targeted attacks, so it is most resilient to targeted attacks. However the complete graph is not of interest to us here.

If we fix both the number of edges and the number of nodes in the network to control for the effect of edge density, it seems networks with evenly distributed degrees tend to have high attack tolerance. For example, both graphs in Figure 3.3 have 6 nodes and 5 edges, but the graph on the left has more evenly distributed degrees than the one on the right. The global efficiency for the graph on the right will immediately decrease to 0 after removing one most connected node, while the global efficiency for the graph on the left will stay positive even after removing a couple of most connected nodes. Clearly the graph on the left has higher attack tolerance than the right one. The importance of the degree distribution in attack tolerance is also discussed in Albert et al. (2000). They showed that a network from the Erdős-Rényi model (Erdős and Rényi, 1960), in which the expected degree of each node is the same, tends to have high tolerance to targeted attacks. On the
other hand, for some scale-free networks with inhomogeneous power-law degree distribution, they are vulnerable to targeted attacks. To control for both the effect of edge density and the effect of degree variation, we fix the degree sequence \((d_1, \ldots, d_n)\) in this chapter. Of course this implies that the number of nodes and the number of edges are fixed as well. So the space \(\mathcal{Y}\) consists of all networks with the same degree sequence as the observed network.

![Figure 3.3: Two networks with the same edge density but different degree sequences.](image)

Fixing the degree sequence has been considered in the literature for various reasons. Schneider et al. (2011) argued that in practice we cannot keep adding edges to increase the robustness of the network because the cost of adding links between every pair of nodes is too expensive in the context of power grids or the Internet. They also assumed that changing the node degree can be much more expensive than changing the links between nodes. In some other situations, fixing the degree sequence may create a basis for exact inference because they are sufficient statistics for the unknown parameters (Chen, 2007). This is sometimes related to random graphs with given degrees which have been used to model complex networks. Another reason to fix the degree sequence is the degree of a node may reflect certain inherent characteristics of an individual, such as the capacity of a machine or the friendliness of a person. These characteristics may not be changeable, and we may need to model the network with these quantities fixed.

The network of 62 bottlenose dolphins has low edge density and unevenly distributed degrees \(d=(6, 8, 4, 3, 1, 4, 6, 5, 6, 7, 5, 1, 1, 8, 12, 7, 6, 9, 7, 4, 9, 6, 1, 3, 6, 3, 3, 5, 5, 9, 5, 1, 3, 10, 5, 1, 7, 11, 8, 2, 8, 5, 6, 7, 4, 11, 2, 6, 1, 2, 7, 10, 4, 2, 7, 2, 2, 9, 1, 5, 1, 3)\), see Figure 3.4.
for the histogram of the degree sequence. Conditioning on the degree sequence allows us to make conditional inference on how and to what extent configurations of local rules could affect attack tolerance. The set of all graphs with the same degree sequence as the dolphin network is the space $\mathcal{Y}$ for the ERGM (3.2). The space $\mathcal{Y}$ is still enormous, containing about $1.826 \times 10^{167}$ networks, as were discussed in Section 2.7.1. We hope to find an appropriate model for the dolphin network under these constraints.

![Figure 3.4: The histogram of the degree sequence of the dolphin network.](image)

### 3.5 Model Fitting for the Dolphin Network

In this section, we fit ERGMs to the network of 62 bottlenose dolphins. There are a list of network statistics that can be potential candidates for $g(y)$ in the ERGM (3.2). However since the degree sequence is fixed, metrics such as the number of edges, the number of nodes with degree $k$, the number of two-paths, $k$-star counts, and the GWD all become fixed numbers. So we mainly look at the following four statistics that are not constants in the set $\mathcal{Y}$: $k$-cycle counts, GWESP, GWDSP, and GWNSP.

In our study of the model fitting, we found that there is no particular advantage to choose different decay parameter $\tau$ for the three geometrically weighted metrics. When $\tau$ is chosen to be the same, we have $\text{GWESP} + \text{GWNSP} = \text{GWDSP}$, and this linear relation implies there is no
need to consider GWDSP. Based on the definition of edgewise shared partners and dyadwise shared partners, we can write the number of cycles as

\[
\text{number of 3-cycle} = \frac{1}{3} \sum_{i=1}^{n-2} iEP_i(y),
\]

\[
\text{number of 4-cycle} = \frac{1}{2} \sum_{i=2}^{n-2} \binom{i}{2} DP_i(y).
\]

Since GWESP and GWDSP are also weighted sums of \( EP_i(y) \) and \( DP_i(y) \), there is a subtle connection between \( k \)-cycle counts and geometrically weighted metrics. We found through model selection that when GWESP and GWNSP are included, adding \( k \)-cycle counts does not improve the fitting of the model. Therefore only two metrics GWESP and GWNSP will be considered for \( g(y) \). That leads to three possible ERGMs with the exponent \( \theta^T g(y) \) being \( \theta_1 \cdot \text{GWESP} \) (Model I), \( \theta_2 \cdot \text{GWNSP} \) (Model II) and \( \theta_1 \cdot \text{GWESP} + \theta_2 \cdot \text{GWNSP} \) (Model III), respectively.

We fitted these three models to the observed dolphin network. We found that for the decay parameter \( \tau \) ranging from 0.1 to 0.5, MCMCMLE gave similar estimates for parameters \( \theta_1 \) and \( \theta_2 \). The approximate AIC (Akaike information criterion) values for fitted models with different \( \tau \) are also similar with \( \tau \) around 0.4 being slightly better than others. Therefore we fix \( \tau = 0.4 \) in the model fitting. Table 3.1 gives the estimates of the parameters for the three models, and each model is fitted and diagnosed with R package “ergm”.

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Model I (GWESP)</th>
<th>Model II (GWNSP)</th>
<th>Model III (GWESP and GWNSP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \theta_1 )</td>
<td>1.468 (0.129)*</td>
<td>-</td>
<td>0.058 (0.112)</td>
</tr>
<tr>
<td>( \theta_2 )</td>
<td>-</td>
<td>-0.313 (0.033)*</td>
<td>-0.421 (0.015)*</td>
</tr>
</tbody>
</table>

Table 3.1: Parameter estimates and their standard errors (in parentheses) for the three ERGMs. Here * means significant at the 0.05 level.

The estimate of \( \theta_1 \), the coefficient for GWESP, is positive for both Models I and III. This indicates that two neighboring individuals are encouraged to share partners. Consider the hypothetical situation that the number of shared partners for a pair of neighboring nodes with \( k \) shared partners is increased to \( k + 1 \), and assume this only results in a change of \( (EP_k, EP_{k+1}) \) to \( (EP_k - 1, EP_{k+1} + 1) \) and all other \( EP_i \) and \( NSP_i \) are not affected. This assumption is difficult
to satisfy in practice because the increase of the shared partner for one neighboring pair typically will affect the edgewise shared partner distribution as well as the non-edgewise shared partner distribution. However, studying the probability change under this seemingly unrealistic assumption can provide some insight on what kind of networks the model favors. Let \( P_{before} \) and \( P_{after} \) denote the probability of the network before and after the change \((EP_k, EP_{k+1}) \rightarrow (EP_k - 1, EP_{k+1} + 1)\) occurs. We have

\[
\frac{P_{after}}{P_{before}} = \frac{\exp\{\theta_1 e^\tau[(EP_k - 1)(1 - \rho^k) + (EP_{k+1} + 1)(1 - \rho^{k+1})]\}}{\exp\{\theta_1 e^\tau[EP_k(1 - \rho^k) + EP_{k+1}(1 - \rho^{k+1})]\}} = \exp\{\theta_1 \rho^k\}, \tag{3.14}
\]

where \( \rho = 1 - e^{-\tau} \). Therefore the change of \((EP_k, EP_{k+1})\) to \((EP_k - 1, EP_{k+1} + 1)\) will result in a log probability change of \(\theta_1 \rho^k\). Since \(\theta_1\) is positive in our models, we can see that the probability increases when a neighboring pair obtain one more shared partner, but the additional gain in probability due to the increase of one shared partner decreases as the number of shared partners \(k\) increases.

The estimate of \(\theta_2\), the coefficient for GWNSP, is negative for both Models II and III. This indicates that if two individuals are not connected, the model discourages them to have shared partners. In other words, two individuals are discouraged to have distance two. Interestingly, similar properties have been discussed for the brain network, i.e., two nodes have a direct connection if needed, but otherwise prefer a longer path between them to maintain efficiency or stability of the network (Simpson et al., 2011). We can also look at how the probability of a network changes when a pair of unconnected nodes increase their shared partner count by one, assuming this does not affect other terms in the model. Similar analysis as (3.14) suggests that the reduction in probability due to the increase of one shared partner decreases as the number of shared partners \(k\) increases.

In Model III, the estimate of \(\theta_1\) is small and it is not significant at the 0.05 level. This shows that the GWESP is a less important term than the GWNSP.
3.5.1 Goodness of Fit Test

To select an appropriate ERGM from the three models under consideration, traditional criteria that involve the likelihood function have their limitations because the intractable normalizing constant \( \kappa(\theta) \) cannot be computed directly and some approximation will be necessary. It is also hard to use the traditional criteria to answer the central question in fitting ERGMs, i.e., can the global structures be reproduced by the local rules? To emphasize this special aspect of ERGM fitting, Hunter et al. (2008a) proposed to simulate a number of samples from the fitted model and compare the values of a set of network statistics in the observed network to those calculated from sampled networks. If the comparison shows that one or more of the observed network statistics are not typical, it indicates that the model does not fit well.

The set of network statistics used in the comparison should characterize different aspects of network structures. Hunter et al. (2008a) proposed using the degree distribution, the minimum geodesic distance distribution, and the edgewise shared partner distribution as the statistics. Since the degree distribution is fixed in our network space, it is not of interest to consider that in our study. The minimum geodesic distance for any pair of nodes is the length of the shortest path connecting them. It is one of the most important metrics of networks and many useful characteristic metrics, such as the diameter and vertex betweenness, are calculated based on the minimum geodesic distance. The edgewise shared partner can quantify the clustering of the network and give triangle counts and other high order metrics. In this section, we select models based on Hunter et al.’s (2008a) graphical goodness of fit method using the minimum geodesic distance distribution and the edgewise shared partner distribution.

We generated 100 samples from each fitted model and the goodness of fit plots for each model are given in Figure 3.5. We can see that for Model III, the observed network statistics always fall in the 95% confidence intervals formed by the simulated networks, but that is not the case for Models I and II. In terms of the minimum geodesic distance, both Models I and II overestimate the number of dyads with minimum distances 2 and 3, but underestimate the number of dyads with minimum distance 5, 6, 7, etc. This shows that comparing to the observed network, the distance
Figure 3.5: Goodness-of-fit plots for model I (top), Model II (middle), and Model III (bottom). In each plot, the black solid line indicates the statistics computed from the dolphin network. The grey lines indicate the range that covers 95% of the statistics computed from 100 sampled networks. The boxplot indicates the median and the interquartile range.
between a pair of nodes tends to be shorter in the simulated networks from Models I and II. In terms of edgewise shared partners, Model I underestimates $EP_0(y)$ which denotes the number of neighboring pairs that share no partners in common. Because the sum of $EP_1(y)$ equals the total number of edges which is a fixed constant here, we can see that comparing to the observed network, more neighboring pairs in the networks generated from Model I share common partners. On the contrary, Model II overestimates $EP_0(y)$. Both Models I and II seem to overestimate $EP_1(y)$ which denotes the number of neighboring pairs that share one partner in common. Therefore, based on the goodness of fit plots, Model III has the best fit among the three.

### 3.6 Attack Tolerance of the Fitted Models

In this section, we study the attack tolerance of the samples from three fitted models. For each model, we generated 5,000 samples from the model and computed the percentage of global efficiency change for each sample under the same targeted attack (removing three most connected individuals). The histogram of the percentage of global efficiency change for samples generated from each model is given in Figure 3.6.

The global efficiency for the dolphin network decreases 5.459% after the 5% targeted attack. Based on the 5,000 samples from Model III, the probability of seeing less than or equal to 5.459% global efficiency change is estimated to be 0.6398 with standard error 0.0152. This indicates that Model III does capture the resilience property of the dolphin network. Comparing with the random networks shown in Figure 3.1, we can see that samples from Model III are more resilient to targeted attacks than random networks. The samples from Model I, however, estimated the probability of seeing less than or equal to 5.459% global efficiency change to be 0.0116 with standard error 0.0034. This shows Model I does not capture the resilience property of the dolphin network. For Model II, the estimate for the same probability is 0.4054 with standard error 0.0155. This indicates that Model II also does pretty well in terms of capturing the resilience property of the dolphin network. Both Models II and III share the statistic GWNSP, and the simulation shows that the GWNSP is important for reproducing the resilience property of the dolphin network.
The following argument provides some connection between the GWNSP and the resilience property. We start by rewriting the expression of the global efficiency as

$$E(G) = \frac{1}{n(n-1)} \sum_{i \neq j \in G} \frac{1}{d_{ij}} = \frac{2}{n(n-1)} \sum_{k=1}^{n-1} \frac{s_k}{k} \triangleq c \sum_{k=1}^{n-1} \frac{s_k}{k}, \quad (3.15)$$

where $s_k$ is the number of dyads $(i,j)$ in the network with $d_{ij} = k$, and $c = 2/n(n-1)$. After a
targeted attack of $l$ most connected nodes, the global efficiency of the remaining graph $\tilde{G}$ becomes

$$E(\tilde{G}) = \frac{1}{(n-l)(n-l-1)} \sum_{i \neq j \in \tilde{G}} \frac{1}{d_{ij}} = \frac{2}{(n-l)(n-l-1)} \sum_{k=1}^{n-l-1} \frac{s_k}{k} \triangleq \tilde{c} \sum_{k=1}^{n-l-1} \frac{s_k}{k}, \quad (3.16)$$

where $s_k$ is the number of dyads $(i,j)$ in $\tilde{G}$ with $d_{ij} = k$, and $\tilde{c} = 2/(n-l)(n-l-1)$. If $\tilde{G}$ is connected, a simple lower bound for $E(\tilde{G})$ is

$$E(\tilde{G}) \geq \tilde{c}s_1 + \tilde{c} \sum_{k=2}^{n-l-1} \frac{s_k}{n-l-1} = \tilde{c}s_1 + \tilde{c} \left( \frac{n-l-1}{2} - s_1 \right). \quad (3.17)$$

If $\tilde{G}$ is not connected, then $E(\tilde{G})$ can be simply bounded below by $\tilde{c}s_1$. Assume that $\tilde{G}$ is connected (the argument for the disconnected case is similar). Then the change of the global efficiency after the attack is

$$E(G) - E(\tilde{G}) = c \sum_{k=1}^{n-1} \frac{s_k}{k} - \tilde{c} \sum_{k=1}^{n-l-1} \frac{s_k}{k} \leq cs_1 - \tilde{c}s_1 + c \sum_{k=2}^{n-1} \frac{s_k}{k} - \tilde{c} \left( \frac{n-l-1}{2} - s_1 \right). \quad (3.18)$$

Since $s_1$ and $\tilde{s}_1$ equal to the total number of edges in $G$ and $\tilde{G}$ respectively, they are fixed numbers. The last term in the upper bound is also fixed. Therefore the only term that may vary is $\sum_{k=2}^{n-1} s_k/k$. Since $\sum_{k=1}^{n-1} s_k$ equals to the total number of dyads in $G$ which is a fixed number, so $\sum_{k=1}^{n-1} s_k - s_1$ is fixed as well. If we decrease $s_2$ and increase $s_3, \ldots, s_{n-1}$ correspondingly, then $\sum_{k=2}^{n-1} s_k/k$ would decrease because the leading term $s_2$ has the largest coefficient $1/2$.

There is a subtle connection between $s_2$ and the GWNSP. Notice that the GWNSP is defined as $e^{\tau_4} \sum_{i=1}^{n-2} (1-(1-e^{-\tau_4})^i) \text{NSP}_i(y)$. As $i$ increases, the coefficient $1-(1-e^{-\tau_4})^i$ becomes close to 1. If we replace this coefficient by 1, we have an approximation to the GWNSP as $e^{\tau_4} \sum_{i=1}^{n-2} \text{NSP}_i(y) = e^{\tau_4}s_2$. In other words, the GWNSP is approximately proportional to $s_2$. The term GWNSP is in Models II and III. For example, Model II takes the form of $P(Y=y) \propto \exp\{\theta_2 \cdot \text{GWNSP}\}$. When $\theta_2 < 0$, the model discourages networks with large value of the GWNSP. Since the GWNSP is approximately proportional to $s_2$, Model II also tends to discourage networks with large value of $s_2$. As we argued earlier, a small $s_2$ will lead to a small $\sum_{k=2}^{n-1} s_k/k$ and a small upper bound of
the global efficiency change in (3.18). Although a small upper bound does not necessarily mean a small global efficiency change, it still sheds light on why Models II and III with $\theta_2 < 0$ for the GWNSP tend to favor networks resilient to targeted attacks.

Because the term GWNSP seems to play an important role in producing networks with high attack tolerance, in the next simulation study, we look at how the attack tolerance of the model changes as we vary the coefficient $\theta_2$ for GWNSP. Consider Model II

$$
P(Y = y) \propto \exp(\theta_2 \cdot \text{GWNSP}), \quad y \in \mathcal{Y},
$$

where the space $\mathcal{Y}$ consists of all networks with the same degree sequence as the dolphin network. We chose $\tau = 0.5$ and five different values for $\theta_2$: $-0.5$, $-0.25$, $0$, $0.25$, and $0.5$. For each $\theta_2$, we used the R package “ergm” to generate 1,000 samples from the corresponding model. These samples are from an MCMC output with 50,000 burn-in steps and every thousandth sample in the chain is kept for inference. To study the attack tolerance of these five models, we computed the percentage of the global efficiency change after 5% targeted attacks for the 1,000 samples from each model, and then used kernel density estimation to obtain the plots in Figure 3.7. We can see that the density curve gradually shifts to the right as $\theta_2$ increases, which indicates that the attack tolerance gradually decreases as $\theta_2$ increases. Another interesting observation is that some of the 1,000 sampled networks are disconnected (before the attack), and the percentage of disconnected networks is $0.611$, $0.302$, $0.1$, $0.017$, and $0.008$ for samples from the five models, which again shows a decreasing pattern as $\theta_2$ increases.

We did the same simulation study for another degree sequence following the power-law distribution $P(k) \propto k^{-2.5}$ with 100 nodes and 179 edges. We used the same $\tau$ and $\theta_2$ values as before, generated 1,000 samples from each model using the “ergm” package, and computed the percentage of global efficiency change after 5% targeted attacks. Figure 3.8 shows the same pattern as Figure 3.7, i.e., the density curve gradually shifts to the right as $\theta_2$ increases. We did not observe disconnected networks from models with $\theta_2 = 0.25$ and $0.5$. The percentage of disconnected networks from models with $\theta_2 = -0.5$, $-0.25$, and $0$ are $0.109$, $0.032$, and $0.003$, respectively. Again the
percentage of disconnected networks decreases as $\theta_2$ increases.

The simulation results suggest that if we need to build a network that is resilient to targeted attacks and the degree sequence is already given, we may sample a network from Model II with a negative parameter value $\theta_2$. Or we may run an MCMC algorithm with Model II as the stationary distribution, and then pick a network from the MCMC output that has the highest attack tolerance. The idea of simulated annealing can be used as well. This could be useful for building the Internet, the WWW, or some other networks to achieve high attack tolerance. Models with small $\theta_2$ seem to have high attack tolerance, but they also tend to generate some disconnected graphs. If connectivity is a basic requirement, we can focus on the sampled networks that are connected.
Figure 3.8: Densities of the percentage of global efficiency change after 5% targeted attacks for Model II on the space of networks with the same degree sequence as the one generated from a power-law with 100 nodes and 179 edges. Five different values for the parameter $\theta_2$ in Model II are considered: $-0.5$, $-0.25$, $0$, $0.25$, and $0.5$.

### 3.7 Discussion

In this chapter, we fit ERGMs to a dolphin network to study its resilience to targeted attacks. To control for the effect of edge density and degree variation and focus on how the nodes are connected with each other to make the network resilient, we consider networks having the same degree sequence as the dolphin network. The local structures we identified that play an important role in the resilience property are GWNSP and GWESP, with GWNSP being the most important one. The samples generated from the fitted model show that the model captures the resilience property and fits the dolphin network well. Such a statistical model can be used to build the Internet and other networks with the same resilience property.

The attack tolerance is measured by the percentage of global efficiency change in our study. The conclusion still holds when the absolute change of global efficiency is used as the measure. Most figures, such as Figures 3.1 and 3.6, are similar under these two measures.
Chapter 4

Monte Carlo Algorithms for Identifying Densely Connected Subgraphs

4.1 Introduction

Network analysis can help us better understand complex systems, and it is becoming increasingly important in various fields, including social science, biology, computer science, psychology and finance. The identification of densely connected subgraphs is one important aspect of network analysis. Densely connected subgraphs have been discovered and studied in biological systems (Everett et al., 2006; Spirin and Mirny, 2003), social networks (Wasserman and Faust, 1994), the World Wide Web (Flake et al., 2000; Dourisboure et al., 2007) and many more. In protein-protein networks or protein-DNA networks, densely connected subgroups of proteins/genes have been identified as functional modules (Gavin et al., 2002), and they function together in important biological processes such as signal transduction and cell-fate regulation (Spirin and Mirny, 2003). In social networks, members of a densely connected subgroup (also referred to as a cohesive subgroup in social science) could share the same religion, social status, or interest in sports or politics. Identifying and studying the architecture of these densely connected subgroups are of particular interest to researchers.

A densely connected subgraph usually refers to a group of vertices in a network that are highly connected within themselves. Members within a densely connected subgraph have more direct, more frequent and stronger contact with each other (Wasserman and Faust, 1994). Cliques are obviously the densest graphs because there is a link between every pair of nodes in a clique. Since cliques do not always exist in a network, dense subgraphs are usually defined through relaxations of cliques, which are often referred to as quasi-cliques. There are several ways to define quasi-cliques,
and different algorithms have been proposed to find dense subgraphs for each definition.

We introduce the notation from graph theory which will be used to define quasi-cliques. A network $G$ can be denoted as $G(V, E)$, where $V$ is a set of $n$ nodes and $E$ is a set of edges. We use $d_G = (d_1, d_2, \ldots, d_n)$ to denote the degree sequence of $G$. If there is an edge between node $i$ and node $j$, it is denoted as $\{i, j\}$. In this chapter, we are concerned with undirected graphs with no self loops or multiple edges.

A widely adopted definition for quasi-clique is defined through a relaxation of the edge density of a clique. The edge density of a graph $G(V, E)$ is defined as

$$Q(G) = \frac{2m}{n(n-1)}, \quad (4.1)$$

where $m$ is the number of edges among the $n$ nodes. The parameter $Q$ characterizes how densely the graph is connected. It is obvious that $0 \leq Q \leq 1$. A fully connected graph (clique) has $Q = 1$ and an empty graph (null graph) has $Q = 0$. If a graph has an edge density $\gamma$, then it is called a $\gamma$-quasi-clique.

Abello et al. (2002) proposed a greedy randomized adaptive search procedure to search for the largest quasi-cliques with edge density greater than or equal to $\gamma$ (also referred to as maximal $\gamma$-quasi-cliques). In their algorithm, the original graph is first decomposed into several small components, and a multi-start local search algorithm is performed within each component. The initial decomposition is a crucial step of the algorithm. However, the decomposition of graphs is a difficult problem, especially for large graphs. The maximal $\gamma$-quasi-clique found through local search is also likely to be a local optimum (Lee et al., 2010). Spirin and Mirny (2003) proposed a stochastic search algorithm to identify densely connected subgroups in protein-protein interaction networks, and several dense protein modules were discovered. They also proposed a super-paramagnetic clustering method using the spin-model. Bu et al. (2003) introduced a spectral analysis method based on a similarity matrix to find dense subgraphs in the protein-protein interaction network. Recently, a branch-and-bound approach is proposed by Pajouh et al. (2012) to search for the largest quasi-cliques with density $\gamma$. The problem of finding the maximal $\gamma$-quasi-cliques is examined from a
mathematical perspective in Pattillo et al. (2013), and some analytical upper bounds on the size of the maximum quasi-cliques are established.

Another commonly used definition for quasi-clique is defined through a relaxation of the degree sequence of a clique. In this definition, a graph with degree sequence \((d_1, d_2, \ldots, d_n)\) is a \(\gamma\)-quasi-clique if

\[
\gamma = \min\left(\frac{d_1}{n-1}, \frac{d_2}{n-1}, \ldots, \frac{d_n}{n-1}\right). \tag{4.2}
\]

It is obvious that \(0 \leq \gamma \leq 1\), and a clique has \(\gamma = 1\) and a graph with an isolated node has \(\gamma = 0\). Liu and Wong (2008) proposed a novel algorithm to search for maximal \(\gamma\)-quasi-cliques. Their algorithm combines several pruning techniques built within the depth-first search framework. Wang et al. (2008) proposed a mining technique that combines theoretical bounds, the graph traversal technique and visualization cues. Brunato et al. (2008) extended the well-known algorithms on finding the largest cliques to finding maximal \(\gamma\)-quasi-cliques. The computation time of such heuristic algorithms is usually a concern and largely depends on the structure (e.g., sparsity or degree distribution) of the graph (Lee et al., 2010).

In this chapter, we propose two algorithms for identifying the densest subgraphs with a fixed size \(k\) or with size in a given range \([k_{\min}, k_{\max}]\). Both algorithms combine the idea of simulated annealing and efficient moves for the Markov chain. Moreover, our algorithms work for different definitions of dense subgraphs. In Sections 4.2 and 4.3, we introduce the two algorithms. In Section 4.4, we give the convergence properties of the proposed algorithms. In Section 4.5, we demonstrate the performance of the algorithms through simulation study and real data analysis. In Section 4.6, we conclude with some discussions.

### 4.2 Identifying Densest Subgraphs with Fixed Size

We first consider the problem of finding the densest subgraphs with a given size \(k\). We use the edge density \(Q\) in (4.1) to illustrate the proposed algorithm although our method works for other definitions of dense subgraphs. Denote \(C_k = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}\) as a subgraph of \(G\) with \(k\) nodes.

Finding the densest subgraphs with size \(k\) is equivalent to finding size \(k\) subgraphs that have the
highest density $Q$, i.e., solving the following optimization problem:

$$\text{argmax}_{C_k \subset G} Q(C_k).$$  \hspace{1cm} (4.3)

For a graph $G$ with $n$ nodes, the number of subgraphs with size $k$ is $\binom{n}{k}$, which could be a large number even for a moderate $n$. A large proportion of the subgraphs are disconnected, especially for sparse graphs. However, disconnected subgraphs are not of interest to us, because the purpose of searching for dense subgraphs is to find a group of nodes that are highly interactive with each other, and it is natural to focus only on connected subgraphs. Therefore the state space of the optimization problem in (4.3) can be reduced to

$$S_k = \{C_k \mid C_k \text{ is a size } k \text{ connected subgraph of } G(V,E)\}.$$  

Furthermore, we also assume that the graph $G(V,E)$ itself is connected. If the graph $G$ is disconnected into several isolated components, we can just focus on each component separately.

Deciding if a size $n$ graph has cliques with at least $k$ nodes is one of Karp’s 21 NP-complete problems (Karp, 1972). It is, therefore, not hard to see that the optimization problem of finding the largest clique in a graph is also NP-complete (since the optimization problem and the decision problem can be reduced to each other in polynomial time). Although the definition of quasi-clique is not unified, the time complexity for finding such subgraphs generally remains NP-complete (Lee et al., 2010). In real networks with a large number of vertices, such as the protein-protein interaction network, performing an exhaustive search for dense subgraphs, even combined with efficient pruning techniques, can become very impractical.

Spirin and Mirny (2003) proposed a Markov chain Monte Carlo (MCMC) algorithm that seeks to maximize $Q$ over subgraphs with size $k$. For a subgraph $C_k = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$ with $k$ nodes, denote $L(C_k) = \sum_{j=1}^{k} \sum_{l=1}^{k} L_{i_j i_l}$, where $L_{ij}$ is the shortest path between node $i$ and node $j$ in graph $G$. Spirin and Mirny’s algorithm starts with a set of $k$ nodes $C_k$. At every step, a node $u_0$ is randomly selected from $C_k$ and a node $u_1$ is randomly selected from the neighbors of $C_k \setminus u_0$ in $G$.  

68
Define $C'_k = C_k \cup u_1 \setminus u_0$ and replace $C_k$ by $C'_k$ with probability

$$p = \begin{cases} 
1 & \text{if } L(C'_k) \leq L(C_k), \\
\exp\left(-\frac{L(C'_k) - L(C_k)}{T}\right) & \text{if } L(C'_k) > L(C_k),
\end{cases}$$

(4.4)

where the temperature parameter $T$ is suggested to be chosen as $T = k$ in Spirin and Mirny (2003).

Every ninth step, an attempt is made to replace a node in $C_k$ with a node that is not connected to $C_k$. After a predetermined number of steps, the graph with the smallest $L_0$ is recorded as the maximizer of $Q$. The algorithm performs well in identifying small dense subgraphs ($k \leq 7$). However, it has trouble locating dense subgraphs of moderate sizes, even with the suggested temperature (see Figure 4.3).

In this section, we propose a simulated annealing algorithm for identifying size $k$ subgraphs with the largest density $Q$. In particular, we design a proposal distribution that aims at increasing the efficiency of the simulated annealing algorithm. A simulated annealing algorithm usually starts with a high temperature which ensures freedom in the transition between different states. Then according to a specific cooling schedule, the temperature gradually decreases to a very small value, which constrains the Markov chain moves within a small range of the objective value. At a fixed temperature $T$ in the simulated annealing algorithm, our proposal distribution for the Markov chain consists of two types of moves, the local move and the global move.

Suppose the current state of the Markov chain is $C_k$. In the **local move**, we first randomly select a node $u_1$ from the neighbors of $C_k$ and randomly select a node $u_0 \in C_k$ whose removal will not disconnect $C_k \cup u_1$. Denote $C'_k = C_k \cup u_1 \setminus u_0$ as the proposed state. A node whose removal will disconnect the remaining graph is called a cut vertex. Since a connected simple graph ($n \geq 2$) has at least two non-cut vertices (Clark and Holton, 1991), it is guaranteed that we have at least one candidate for $u_0$.

In the **global move**, we “grow” a size $k$ connected subgraph from a selected node. We first randomly select a node $v_1 \notin C_k$, where $C_k$ is the current state of the Markov chain. Then we randomly selected a node $v_i$ from the neighbors of $\{v_1, v_2, \ldots, v_{i-1}\}$, for $i = 2, \ldots, k$, until we reach
a subgraph of size $k$. Denote $C_k' = \{v_1, v_2, \ldots, v_k\}$ as the proposed state.

The details of our simulated annealing algorithm can be summarized as follows:

**Algorithm 4.2.1 (Identifying the densest subgraph of size $k$).**

*Input: Cooling schedule $T_i$, $i = 1, 2, \ldots$, with $T_1 > T_2 > \ldots$, and the number of iterations $M$ at each temperature.*

1. At step $t = 1$, select a size $k$ connected subgraph $C_{k,1}$ and set the temperature level $i = 1$.
2. Set $T = T_i$.
3. (a) At step $t$, suppose the current state is $C_{k,t}$. With probability $\alpha$, propose a **local move** and with probability $1 - \alpha$ propose a **global move**. Denote the proposed state as $C_k'$.  
   (b) Calculate the acceptance probability

   $$p = \begin{cases} 
   1 & \text{if } Q(C_k') \geq Q(C_{k,t}) , \\
   \exp \left( \frac{Q(C_k') - Q(C_{k,t})}{T} \right) & \text{if } Q(C_k') < Q(C_{k,t}) . 
   \end{cases} \quad (4.5)$$

   (c) With probability $p$, set $C_{k,t+1} = C_k'$ and with probability $1 - p$, set $C_{k,t+1} = C_{k,t}$.
   (d) Set $t = t + 1$.
4. If $t < iM$, go to step 3. If $t = iM$, set $i = i + 1$ and go to step 2.
5. Stop after $Q$ converges or the algorithm reaches a predetermined number of steps.

Here $M$ is the number of iterations at each temperature. Theoretical studies have suggested that the number of moves required to guarantee a solution that is arbitrarily close to the optimum will typically be exponential in problem size (Mitra et al., 1986; Aarts and Van Laarhoven, 1985). It is, however, not practical to have such a large number of iterations, and the theories do not lead to generic decisions on how to choose $M$ in practice. Nevertheless, they do provide some insights as to what should be taken into consideration when choosing $M$. In practice, the choice of $M$ should be problem specific, depending on the structure of the graph $G$ and the size $k$ of the subgraph we are looking for. Most real world networks are sparse, i.e., $m = O(n)$, where $m$ is the total number of edges and $n$ is the total number of nodes in the graph. When identifying dense subgraphs of
moderate sizes (<1000) within a sparse graph with \( n \) nodes, we suggest choosing \( M \in [n, 2n] \). However, in practice, \( M \) could be smaller depending on the size of the problem. In general, \( M \) should increase with the size and density of the graph \( G \) and the size \( k \) of the subgraphs we are interested in.

In Step 3(a), the proportion of local moves and global moves is controlled by the parameter \( \alpha, 0 \leq \alpha \leq 1 \). The local moves aim at exploring the neighbors of the current state \( C_{k,t} \). If the current state is in the “good region” (i.e., the neighborhood of the subgraphs with the highest \( Q \) value) and the temperature is low, the Markov chain can quickly converge to the subgraph with the highest \( Q \) through local moves. However, it may take a considerable amount of steps via the local moves for the Markov chain to explore \( S_k \) and find the “good region”. Moreover, in a local move, we propose a new connected subgraph by replacing only one node in the current subgraph by another node. Due to the nature of this proposal distribution, the Markov chain could be trapped in a local mode before it reaches the “good region”. In order for the Markov chain to better explore the sample space and identify the “good region”, we introduced the global move. The global move allows the sampler to jump out of the local mode, and it also shortens the time needed to identify the “good region”.

In practice, the parameter \( \alpha \) should be chosen to strike a balance between the local and global moves. If \( \alpha \) is too small, then the sampler “jumps around” too often, and the algorithm can not fully explore the neighborhood of the current subgraph. If \( \alpha \) is too large, then the algorithm can not fully take advantage of the global move, and the Markov chain could be trapped in a local mode before it fully explores the sample space. In our studies, we find that \( \alpha \in [0.75, 0.9] \) usually achieves a reasonable balance between these two types of moves. In all the examples in Section 4.5, we set \( \alpha = 0.9 \).

To assess the convergence of the density function \( Q \) in Step 5, we monitor the value of \( Q \) at every step of the algorithm. If after running \( \tau \) steps at temperature \( T_i \), the value of \( Q \) stays the same for the next \( 2M - \tau \) steps (\( M - \tau \) steps at temperature \( T_i \) and \( M \) steps at temperature \( T_{i+1} \)),
then $Q$ is considered to have converged. In other words,

$$Q(C_k, (i-1)M+\tau+j) = Q(C_k, (i-1)M+\tau), \text{ for } j = 1, \ldots, 2M - \tau.$$  

The algorithm may also stop after reaching a predetermined number of steps. In that case, the subgraph that has the largest recorded $Q$ value is the output of the algorithm.

The convergence property of Algorithm 1 is given in Section 4.4. It is worth mentioning that Algorithm 4.2.1 also works for other definitions of quasi-cliques by modifying the optimization objective $Q$ accordingly. For example, We can use the algorithm to find dense subgraphs defined by (4.2). We can also use the algorithm to find subgraphs with the smallest $L(C_k)$ or the smallest diameter.

### 4.3 Identifying Densest Subgraphs with Size Bounds

In some applications, the size of the densest subgraph may not be known. Instead, we may only know that we are interested in the densest subgraphs with size in a given range. To handle such cases, we propose an algorithm that allows the Markov chain to explore subgraphs of different sizes.

When the state space consists of subgraphs with different sizes, the edge density $Q$ is no longer an appropriate optimization objective because of the following lemma.

**Lemma 4.3.1.** Let $G$ be a graph with $n$ nodes, and $G_1$ be a subgraph of $G$ with $n-1$ nodes. Then the following inequality holds:

$$\max_{G_1 \subseteq G} Q(G_1) \geq Q(G). \quad (4.6)$$

See Section 4.7 for the proof. The lemma indicates that the optimization objective $Q$ implicitly favors smaller subgraphs. Here we use an alternative criterion based on the average degree

$$R(C_k) = m/k \quad (4.7)$$
as the optimization objective, where \( m \) is the total number of edges among the \( k \) nodes in subgraph \( C_k \). The average degree \( R \) is another way to define dense subgraphs. When the size \( k \) of the subgraph is fixed, maximizing the average degree \( R \) is equivalent to maximizing the edge density \( Q \), but when \( k \) varies, \( R \) no longer favors smaller graphs.

Maximizing the average degree \( R \) with size bounds is a problem that has been well studied in computer science (often referred to as the dense subgraph problem). Large sparse networks, such as the World Wide Web, generally have a low average degree over the entire network. However, there are usually local regions that contain far more links than their fair share. Finding such local regions can be used to identify web communities, link spam, and so on (Feige et al., 1997; Anderson and Chellapilla, 2009). Two variations of the problem of finding the subgraph with maximum average degree has been considered. One is finding the densest at least size \( k \) subgraph (referred to as \( dalks \) in computer science), and the other is finding the densest at most size \( k \) subgraph (referred to as \( damks \) in computer science).

In this section, we propose an algorithm that can find the densest subgraph with at least \( k_{\min} \) nodes and at most \( k_{\max} \) nodes. Define the new optimization objective as

\[
\arg\max_{C_k \in S_{k_{\min},k_{\max}}} R(C_k),
\]

where \( S_{k_{\min},k_{\max}} = \{ C_k \mid C_k \text{ is a size } k \text{ connected subgraph of } G(V,E), k \in [k_{\min},k_{\max}] \} \) is the state space. Finding \( dalks \) is a special case of the optimization problem (4.8) if \( k_{\min} \) is set to the minimum size and \( k_{\max} \) is set to the total number of nodes \( n \) in \( G \). Similarly, finding \( damks \) is also a special case of (4.8) if \( k_{\min} = 1 \) and \( k_{\max} \) is set to the maximum number of nodes. Here is our simulated annealing algorithm that maximizes \( R(C_k) \) for \( C_k \in S_{k_{\min},k_{\max}} \).

**Algorithm 4.3.1** (Identifying the densest subgraph with size \( k \in [k_{\min},k_{\max}] \)).

*Input: Cooling schedule \( T_i \), \( i = 1,2,\ldots \), with \( T_1 > T_2 > \ldots \), and the number of iterations \( M \) at each temperature.*

1. At step \( t = 1 \), select a connected size \( k_0 \) subgraph \( C_{k_0,1} \) such that \( k_{\min} \leq k_0 \leq k_{\max} \) and set
the temperature level \( i = 1 \).

2. Set \( T = T_1 \).

3. At step \( t \), suppose the current state is \( C_{k,t} \) with size \( k_t \).

   (a) With probability \( \beta_1 > 0 \), implement the following:
   
   If \( k_t > k_{\min} \), randomly select a node \( u_0 \in C_k \) whose removal will not disconnect \( C_{k,t} \).
   Denote \( C_{k,*}^t = C_{k_0,t} \setminus u_0 \) as the proposed state with size \( k^* = k_t - 1 \), and go to step 4. If \( k_t = k_{\min} \), go to step 3.

   (b) With probability \( \beta_2 > 0 \), implement the following:
   
   If \( k_t < k_{\max} \), randomly select a node \( u_0 \) from the neighbors of \( C_{k,t} \). Denote \( C_{k,*}^t = C_{k_t,t} \cup u_0 \) as the proposed state with \( k^* = k_t + 1 \), and go to step 4. If \( k_t = k_{\max} \), go to step 3.

   (c) With probability \( 1 - \beta_1 - \beta_2 > 0 \), implement the following:
   
   With probability \( \alpha \), propose a local move, and with probability \( 1 - \alpha \), propose a global move. Denote the proposed state as \( C_{k,*}^t \) with \( k^* = k_t \).

4. (a) Calculate the acceptance probability

   \[
   p = \begin{cases} 
   1 & \text{if } R(C_{k,*}^t) \geq R(C_{k,t}) , \\
   \exp \left( \frac{R(C_{k,*}^t) - R(C_{k,t})}{T} \right) & \text{if } R(C_{k,*}^t) < R(C_{k,t}). 
   \end{cases} 
   \]

   (4.9)

   (b) With probability \( p \), set \( k_{t+1} = k^* \) and \( C_{k_{t+1},t+1} = C_{k,*}^t \). With probability \( 1 - p \), set \( k_{t+1} = k_t \)
   and \( C_{k_{t+1},t+1} = C_{k,t} \).

   (c) Set \( t = t + 1 \).

5. If \( t < iM \), go to step 3. If \( t = iM \), set \( i = i + 1 \) and go to step 2.

6. Stop after \( R \) converges or the algorithm reaches a predetermined number of steps.

Here \( M \) is the number of iterations at each temperature. In practice, the choice of \( M \) should be problem specific, depending on the structure of the graph \( G \) and the size range \([k_{\min}, k_{\max}]\) of the subgraphs. When identifying dense subgraphs of moderate sizes (\( k_{\max} < 1000 \)) within a sparse graph with \( n \) nodes, we suggest choosing \( M \in [n, 5n] \). However, in practice, \( M \) could be smaller.
depending on the size of the problem. In general, $M$ should increase with the size and density of the graph $G$ and the size range of the subgraphs we are interested in.

Algorithm 4.3.1 allows us to examine subgraphs of different sizes. In Step 3, the parameter $\beta_1$ ($\beta_2$) reflects the proportion of moves that will decrease (increase) the size of the subgraph by one. If $\beta_1$ and $\beta_2$ are too small, then it takes the Markov Chain a long time to explore subgraphs of all sizes within the given range, especially if the specified size range is large. If $\beta_1$ and $\beta_2$ are too large, then the Markov chain can not fully explore the subgraphs with a given size. Moreover, the global moves are proposed with probability $(1 - \beta_1 - \beta_2)(1 - \alpha)$. If $\beta_1$ and $\beta_2$ are too large, the Markov chain can not fully take advantage of the global moves. Therefore, the parameters should be chosen to strike a balance between different kind of moves. In general, we recommend choosing $\beta_1 \in [0.2, 0.3]$, $\beta_2 \in [0.2, 0.3]$ and $\alpha \in [0.75, 0.9]$ in Algorithm 2. In our studies, we set $\beta_1 = 0.25$, $\beta_2 = 0.25$ and $\alpha = 0.9$.

All moves in Step 3 guarantee the connectedness of the subgraph. We could use the trace plot of the algorithm to learn the density of subgraphs of different sizes (see Figure 4.7). This could also be used as a preliminary step of identifying dense regions when we have no prior knowledge of the network (Anderson and Chellapilla, 2009). The convergence property of Algorithm 2 is given in Section 4.4.

### 4.4 Convergence Properties

In this section, we discuss the convergence property of the two proposed simulated annealing algorithms. Algorithm 1 specifies a Markov chain on the set $S_k$ of connected size $k$ subgraphs. The following lemma, whose proof is given in Section 4.7, shows that the Markov chain can reach every state in $S_k$.

**Lemma 4.4.1.** The Markov chain on $S_k$ specified by Algorithm 4.2.1 is irreducible.

The simulated annealing algorithm in Algorithm 4.2.1 can be viewed as an inhomogeneous Markov process whose transition probability depends on the temperature $T$. We show below that
the inhomogeneous Markov process in Algorithm 4.2.1 converges to the set of optimal states (states with the global maximum density $Q$ value) with probability one if the temperature is lowered slowly.

**Theorem 4.4.1** (Convergence of Algorithm 4.2.1). *In Algorithm 4.2.1, if $\lim_{i \to \infty} T_i = 0$ and

$$
\sum_{i=1}^{\infty} e^{-1/T_i} = \infty, \tag{4.10}
$$

then we have $\lim_{t \to \infty} P(C_{k,t} \in S_k^0) = 1$, where $S_k^0 = \{ C_k^* \in S_k : Q(C_k) \leq Q(C_k^*) \text{ for all } C_k \in S_k \}$ is the optimal set.*

The proof, which follows the results in Hajek (1988), is given in Section 4.7. Theorem 5.3.1 shows that Algorithm 4.2.1 eventually converges to the set of states with globally maximum density $Q$. Condition (4.10) requires the temperature in the algorithm to cool down slowly. For example, if $T_i$ assumes the logarithmic cooling schedule

$$
T_i = \frac{1}{\log(i + 1)}, \text{ for } i = 1, 2, \ldots, \tag{4.11}
$$

then condition (4.10) is satisfied.

Similarly we can show the irreducibility and convergence of Algorithm 2.

**Lemma 4.4.2.** *The Markov chain on $S_{k_{\text{min}},k_{\text{max}}}$ specified in Algorithm 4.3.1 is irreducible.*

See Section 4.7 for the proof.

**Theorem 4.4.2** (Convergence of Algorithm 4.3.1). *In Algorithm 4.3.1, if $\lim_{i \to \infty} T_i = 0$ and

$$
\sum_{i=1}^{\infty} e^{-\frac{k_{\text{max}}-1}{2T_i}} = \infty, \tag{4.12}
$$

then we have $\lim_{t \to \infty} P(C_{k,t} \in S_{k_{\text{min}},k_{\text{max}}}^0) = 1$, where $S_{k_{\text{min}},k_{\text{max}}}^0 = \{ C_k^* \in S_{k_{\text{min}},k_{\text{max}}} : R(C_k) \leq R(C_k^*) \text{ for all } C_k \in S_{k_{\text{min}},k_{\text{max}}} \}$ is the optimal set.*

See Section 4.7 for the proof. Theorem 5.5.1 shows that the Markov chain in Algorithm 4.3.1 also converges to the set of optimal states with probability one. Condition (4.12) also requires the
temperature in the algorithm to cool down slowly.

4.5 Applications and Simulations

In this section, we apply the proposed algorithms to a simulation study on the planted clique problem and two real data examples: one is the yeast protein interaction network, and the other is the stock market graph. All examples were coded in R and run on a MacBook Pro with 2.26 GHz Intel Core 2 Duo processor.

In the implementation of the proposed simulated annealing algorithms, the cooling schedules in Theorem 1 and Theorem 2 could be slow and require a long computation time. In practice, the temperature should cool down sufficiently slowly, and the Markov chain should also spend enough time at low temperatures to ensure that the region around a local optimum is fully explored (Dowsland and Thompson, 2012). With these considerations, we adopt one of the most widely used geometric cooling schedule in the examples (Nourani and Andresen, 1998). The geometric cooling scheme sets the temperature $T_i$ according to

$$T_i = T_0 \times \left( \frac{T_K}{T_0} \right)^{i/K} ,$$

(4.13)

where $T_0$ is the initial temperature, $K$ is the number of cooling steps and $T_K$ is the final temperature.

The initial temperature $T_0$ in the simulated annealing algorithm should be high enough to allow free movement through the sample space. A reasonable initial temperature could be estimated from the data. For example, we can calculate the maximum change in the objective function over a given number of Markov chain moves. This can then be used to estimate the temperature at which a move that decreases the objective function value could be accepted with a reasonable probability.

For the number of cooling steps $K$, if it is too large, then the temperature decreases very slowly and the computation could be very time-consuming. If $K$ is too small, then the temperature decreases very fast and the Markov chain may converge to a local optimum. Choosing a proper $K$ requires a good understanding of the problem. It depends on the structure of the graph, the size of
the subgraphs and the initial temperature. In general, $K$ should increase with size and density of the graph, the size of the subgraphs and the initial temperature. For Algorithm 1, we recommend choosing $K \geq 20$, and for Algorithm 2, we recommend choosing $K \geq 50$. In practice, $K$ could be smaller depending on the problem.

In theory, the final temperature $T_K$ in the simulated annealing algorithms should be set to zero (Dowsland and Thompson, 2012). However, in practice, a frozen state where no further moves are possible is often reached before the temperature reaches zero. Therefore, the final temperature in the cooling schedule is usually set to a very small value. If a final temperature is not low enough for the Markov chain to reach the frozen state, we can lower the final temperature further. For Algorithm 1, we recommend choosing $T_K \in [10^{-5}, 10^{-3}]$, and for Algorithm 2, we recommend $T_K \in [10^{-3}, 10^{-1}]$.

4.5.1 The Planted Clique Problem

In the classical planted clique problem, we are given a graph $G$ whose edges are generated by starting with an Erdos-Renyi random graph, and then “planting” a clique (adding edges to make a clique) on $k$ vertices (Brubaker and Vempala, 2009). The planted clique problem was introduced as a potential variant of the extensively studied problem of finding the largest clique in a random graph (Jerrum, 1992). In this section, we apply Algorithm 1 to the planted clique problem.

We first generated an Erdos-Renyi random graph with $n = 100$ nodes and each edge is generated with probability $p = 0.05$. Then we embedded a size 10 clique in the graph. Figure 4.1 shows the graph we generated. Even though $p$ is as small as 0.05, it is still hard to visually identify the embedded clique in the graph. We applied Algorithm 1 to search for the densest subgraph of size 10 in Figure 4.1. The algorithm starts with the temperature $T_1$ (set $T_0 = 1$) and gradually cools down to $T_K = 0.001$ in $K = 50$ steps with a cooling schedule of $T_i = T_{i-1}/K$, $i = 1, 2, \ldots, K$. At each temperature $T_i$, the number of moves is set to $M = 200$. The probability of proposing a local move is set to $\alpha = 0.9$. After 10,000 steps, Algorithm 1 identified the embedded size 10 clique. Figure 4.2 shows the trace plot of the density $Q$.

We repeated the above procedure 100 times by generating 100 Erdos-Renyi graphs and embed-
finding a size 10 clique in each of the graphs. Algorithm 1 identified the clique in all 100 cases. With the suggested temperature setting ($T = 10$) and 50,000 Markov chain moves, Spirin and Mirny’s (2003) algorithm can only find the embedded clique 1 out of the 100 times.

Figure 4.1: The Erdos-Renyi graph ($n = 100, p = 0.05$) with an embedded size 10 clique. The nodes in the embedded clique are in black.

Figure 4.2: The trace plot of Algorithm 1 for finding the densest subgraph of size 10.
4.5.2 Yeast Protein Interaction Network

Large-scale biological experiments have provided biological networks such as protein-protein interaction networks, protein-DNA interaction networks and metabolic networks. In these large biological networks, numerous motifs that consist three to four nodes were discovered. These motifs, as a biological functional module, regulate feedback and feed-forward loops in cells (Lee et al., 2002). However, large-scale biological processes such as signal transduction, cell-fate regulation, transcription and translation, involve more than five but fewer than hundreds of proteins (Spirin and Mirny, 2003). These proteins are often densely connected and work as one functional module. Due to the size of the network, the discovery of those densely connected subnetworks remains a difficult problem.

Dense subgraphs with fixed size

The yeast protein-protein network is a simple undirected network with 3,992 nodes and 6,500 edges. Each node represents one protein and an edge between two nodes indicates that there is an interaction between the two proteins. The network is not connected. There are several small groups of nodes (size ≤ 5) that are fragmented from the main body of the network. The main body is a connected network consists of 3,669 nodes and 6,316 edges. In the following analysis, we will focus on dense subgraph discovery in the main body of 3,669 nodes.

In Spirin and Mirny (2003), they analyzed the yeast protein-protein network and identified more than 50 statistically significant protein clusters with size ranging from 4 to 35. In particular, using exhaustive search, they found that the largest clique has 14 nodes. We applied Algorithm 1 to search for the densest subgraph with size 14. Algorithm 1 starts with the temperature $T_1$ and gradually cools down to $T_K = 0.001$ in $K = 15$ steps with a cooling schedule of $T_i = T_1^{i/K}$, $i = 1, 2, \ldots, K$. At each temperature $T_i$, the number of moves is set to $M = 2,000$. The probability of proposing a local move $\alpha$ is set to 0.9. Algorithm 1 quickly identified the size 14 clique in less than 30,000 Markov chain moves (see Figure 4.3). The computation time is 57.78 seconds. Spirin and Mirny’s (2003) algorithm cannot find the size 14 clique even after 300,000 Markov chain moves (see...
Figure 4.3), although the effective temperature $T$ was set to 14 as suggested in Spirin and Mirny (2003). The densest cluster found in 300,000 steps by Spirin-Mirny algorithm has $Q = 0.54$, which is much smaller than the $Q$ value of a clique ($Q = 1$). The 300,000 moves took 148.62 seconds.

Among all the dense subgraphs that were identified in Spirin and Mirny (2003), the largest one is of size 35 and has $Q = 0.119$. It is mentioned in Spirin and Mirny (2003) that this is a “new” module that has not been studied in experiments. We applied Algorithm 1 to search for the densest subgraph with size 35. The temperature starts with $T_1$ and gradually cools down to $T_K = 0.0001$ in $K = 25$ steps with a cooling schedule of $T_i = T_K^{i/K}$, $i = 1, 2, \ldots, K$. At each temperate $T_i$, the number of moves is set to $M = 5,000$. The probability of proposing a local move $\alpha$ is set to 0.9. After 125,000 steps, we found that the densest subgraph of size 35 has $Q = 0.2992$ (see Figure 4.4). The members of this subgraph is given in Section 4.8. This subgraph is more dense than the subgraph reported in Spirin and Mirny (2003), and it may be of interest for scientists to study this subgraph in experiments.
Figure 4.4: Trace plots of Algorithm 4.2.1 for finding the densest subgraph of size 35.

Statistical significance

To test the statistical significance of a densely connected subgraphs, Spirin and Mirny (2003) compared the observed network with random networks having the same degree sequence as the observed network. In other words, the reference distribution for the null hypothesis is chosen to be the uniform distribution over all networks that preserve the number of interactions of each node. To generate random networks with the given degree sequence, we implemented the following MCMC algorithm. We start with the original graph $G$. At each step of the Markov chain, two edges $\{x, y\}$ and $\{u, v\}$ with distinct nodes $x$, $y$, $u$, $v$ are chosen randomly from the current graph. If neither $x$ and $u$ nor $y$ and $v$ is linked, then a new graph can be constructed by replacing the edges $\{x, y\}$ and $\{u, v\}$ from the current graph by two new edges $\{x, u\}$ and $\{y, v\}$, and the Markov chain moves to the new state. Otherwise, the Markov chain stays at the current graph.

To test the statistical significance of the dense subgraph with 35 nodes identified by Algorithm 4.2.1 (see Section 4.8 for the subgraph), we implemented the above MCMC algorithm and kept one sample in every 5,000 steps for inference. For each random network we kept, we computed the $Q$ value for the same subgraph with 35 nodes. Figure 4.5 is the histogram of the $Q$ values computed for the size 35 subgraph on all the 100,000 random networks we obtained. It is evident that
the observed subgraph with $Q = 0.2992$ in protein-protein interaction network is highly significant with $p$-value approximately equal to 0.

![Histogram of Q for 100,000 random graphs](image)

**Figure 4.5:** Histogram of the $Q$ value of size 35 subgraphs in random networks.

It is worth mentioning that based on the trace plot of $Q$ from Algorithm 4.2.1, we could also locate some local maxima. The subgraphs that achieve the local maximum $Q$ could be interesting to study as well. For example, when we searched for the densest subgraph of size 21 in the yeast network, one of the local maxima in the trace plot has $Q = 0.3571$, and this subgraph is highly significant with $p$-value approximately equal to 0. This group of 21 proteins (see Section 4.8 for the members) could potentially be a new “module” that carry out a certain biological process.

**Dense subgraphs with size bounds**

In this section, we apply Algorithm 4.3.1 to search for dense subgraphs with the highest average degree $R$ in the yeast protein-protein interaction network. We set the size bounds of the subgraphs to $5 \leq k \leq 50$. The temperature starts with $T_1$ and gradually cools down to $T_K = 0.1$ in $K = 20$ steps with a cooling schedule of $T_i = 10 \times (\frac{T_K}{10})^{i/K}$, $i = 1, 2, \ldots, K$. At each temperature $T_i$, the number of moves is set to $M = 5,000$. The probability of proposing a local move $\alpha$ is set to 0.9. At each step, the probability of proposing an increase or decrease in size is $\beta_1 = \beta_2 = 0.25$. After 100,000 moves, the subgraph ($5 \leq k \leq 50$) with the highest average degree is identified to be the same size 14 clique that was discussed in Section 4.5.2 (see Figure 4.6). Figure 4.7 is the scatter plot of the average degree $R$ against the size of the subgraph $k$ for the states that the Markov
chain has visited. We can see that the densest subgraphs with size between 10 and 16 have higher average degrees.

Figure 4.6: Trace plots of the average degree (left) and the size of the subgraph (right) for finding subgraphs $(5 \leq k \leq 50)$ with the highest average degree using Algorithm 4.3.1.

### 4.5.3 Mining Stock Market Graphs

A natural graph representation of the stock market is based on the cross correlations of price fluctuations (Boginski et al, 2006). In a stock market graph, each financial instrument is represented by a vertex, and two vertices are linked by an edge if the correlation coefficient of the logarithm of daily return of the two instruments calculated over a certain period of time exceeds a specified threshold $\theta$.

The stock market graph we analyzed has 5,700 nodes and 50,025 edges. It is constructed based on the return prices from October 20, 2008 to October 15, 2010 (502 consecutive days) in the American stock market consisting of NASDAQ, AMEX and BYSE. This stock market data is also analyzed in Budai and Jallo (2011). The correlation threshold is set to $\theta = 0.5$, which describes high correlations between a pair of stocks (Boginski et al, 2006).
Dense subgraphs with fixed size

Cliques in the market graph represent classes of stocks whose price fluctuations exhibit similar behavior over time. In the US market, the cliques are usually found to be within specific industries (or sectors) while in the Swedish market, they are usually found to be around some of the largest companies (Budai and Jallo, 2011). Using Algorithm 4.2.1, the largest clique we found is of size 82 (see Section 4.8 for the members). The algorithm starts with the temperature $T_1$ and gradually cools down to $T_K = 0.00001$ in $K = 50$ steps with a cooling schedule of $T_i = T_{i+1}^{i/K}, i = 1, 2, \ldots, K$. At each temperature $T_i$, the number of moves is set to $M = 1,000$. The probability of proposing a local move $\alpha$ is set to 0.9. See Figure 4.8 for the trace plot of Algorithm 4.2.1. Using Spirin and Mirny’s (2003) algorithm, the densest subgraph identified in 500,000 steps has edge density $Q = 0.5194$ (temperature set as the suggested value $T = 82$), and the clique was not found.
Bolginski et al. (2006) considered the market graph constructed based on the return prices from September 24, 1988 to September 15, 2000 (500 consecutive days) in the American stock market with correlation threshold set to $\theta = 0.5$. They found that the largest clique in the market graph is of size 18. The increase in the size of the largest clique from 18 to 82 indicates that more and more stocks in the US market are affecting the behavior of the others and there is a trend of globalization of the stock market. This phenomenon is also referred to as the “globalization hypothesis” in Bolginski et al. (2006). This clique with 82 nodes consists mostly of stocks from the material sector and the technology sector. These two sectors were both significantly affected by the economic downturn in 2008. However, it is not obvious why these two sectors exhibit highly similar behavior from October 20, 2008 to October 15, 2010. This phenomenon calls for further study.

**Dense subgraphs with size bounds**

We also used Algorithm 4.3.1 to search for dense regions in the market graph (subgraphs with the highest average degree $R$). We set the size bounds of the subgraphs to $100 \leq k \leq 350$. The

Figure 4.8: Trace plots of Algorithm 4.2.1 for finding the clique of size 82.
algorithm starts with temperature $T_1$ and gradually cools down to $T_K = 0.001$ in $K = 50$ steps with a cooling schedule of $T_i = 10 \times \left(\frac{T_K}{10}\right)^{i/K}$, $i = 1, 2, \ldots, K$. At each temperature $T_i$, the number of moves is set to $M = 2,000$. The probability of proposing a local move $\alpha$ is set to 0.9. At each step, the probability of proposing an increase or decrease in size is $\beta_1 = \beta_2 = 0.25$. After 100,000 moves, the subgraph with the highest average degree is identified (see Figure 4.9). It is a subgraph with 338 nodes (see Section 4.8 for the members) and the average degree is 179.01. This dense region includes mostly stocks from the material sector, the financial sector, the health care sector, the technology sector, the industry sector and the conglomerates sector. It shows that these industries are greatly influenced by each other. The size and diversity of this dense region in the market graph also show that nowadays, more and more stocks are affected by the behavior of the others.

Figure 4.9: Trace plots of the average degree (left) and the size of the subgraph (right) for finding subgraphs ($100 \leq k \leq 350$) with the highest average degree using Algorithm 4.3.1.
4.6 Discussions

In this chapter, we propose two simulated annealing algorithms for identifying dense subgraphs: one for subgraphs with a fixed size and the other one for subgraphs with size in a given range. Both algorithms are shown to converge to the set of optimal states (densest subgraphs) with probability one. When there are multiple optimal states, the algorithms may converge to any one of them. One may run the algorithms multiple times with different starting points to identify different states in the optimal set.

Although we focus on simple unweighted graphs in this chapter, the proposed algorithms can be easily extended to weighted graphs. For a weighted graph $G(V, E)$, each node $i$ has a weight $w_i$. One interesting problem in weighted graphs is to find the maximum weight connected subgraph (Hochbaum et al., 1994; Dilkina et al., 2010). It is a network design problem and has applications in various fields such as conservation planning, forestry, systems biology, computer vision, and communication network design. Finding the maximum weight connected subgraph with size $k$ is the same as solving the problem of

$$\arg \max_{C_k \in S} \sum_{i \in C_k} w_i.$$  \hspace{1cm} (4.14)

Algorithm 4.2.1 and Algorithm 4.3.1 can still be applied to search for the solution by using (4.14) as the optimization objective.

4.7 Proof of the Main Results

Proof of Lemma 4.3.1

Suppose the degree sequence of $G$ is $d = (d_1, d_2, \ldots, d_n)$. Without loss of generality, assume $d$ is in non-decreasing order. Then

$$Q(G) = \frac{d_1 + d_2 + \cdots + d_n}{n(n - 1)}$$
and
\[
\max_{G_1 \subset G} Q(G_1) = \frac{d_2 + \cdots + d_n - d_1}{(n-1)(n-2)}.
\]

When \(d_1 = 0\), it is obvious that \(\max_{G_1 \subset G} Q(G_1) \geq Q(G)\). When \(d_1 \geq 1\), we have
\[
\frac{\max_{G_1 \subset G} Q(G_1)}{Q(G)} = \frac{d_2 + \cdots + d_n - d_1}{d_1 + d_2 + \cdots + d_n} \times \frac{n}{n-2} \\
= \left(1 - \frac{2d_1}{d_1 + d_2 + \cdots + d_n}\right) \times \frac{n}{n-2} \\
\geq \left(1 - \frac{2}{n}\right) \times \frac{n}{n-2} = 1.
\]

Proof of Lemma 4.4.1

To prove the irreducibility of the Markov chain, it is enough to show that

a) every \(C'_k \in S_k\) can be reached from any \(C_k \in S_k\) through one global move;

b) every \(C'_k \in S_k\) can be reached from any \(C_k \in S_k\) through a sequence of local moves.

To prove a), pick any node \(v_1\) that is in \(C'_k\) but not in \(C_k\). It is known that every connected graph with finite nodes has a spanning tree (Diestel, 2006), and any node in an undirected tree can be the root node. Therefore, we can find a spanning tree of \(C'_k\) and take \(v_1\) as the root node. In this way, \(C'_k\) can be reached from \(C_k\) in one global move.

To prove b), we use the inductive method. When \(k = 2\), \(C_2 = \{v_1, v_2\}\) and \(C'_2 = \{v'_1, v'_2\}\) are two edges in graph \(G(V,E)\). If \(C_2\) and \(C'_2\) have one common node, then \(C'_2\) can be reached from \(C_2\) in one local move. If they do not have overlapping nodes, then one can find the shortest path connecting \((v_1, v_2)\) and \((v'_1, v'_2)\) (because \(G\) is connected). Then \(C'_2\) can be reached from \(C_2\) through a sequence of moves along the shortest path. Now assuming that any size \(k\) connected subgraph \(C'_k\) can be reached from any size \(k\) connected subgraph \(C_k\), we need to show that any size \(k + 1\) connected subgraph \(C'_{k+1}\) can be reached from any size \(k + 1\) connected subgraph \(C_{k+1}\). First there is a size \(k\) connected subgraph \(C_k \subset C_{k+1}\) and a size \(k\) connected subgraph \(C'_k \subset C'_{k+1}\). By the assumption, \(C'_k\) can be reached from \(C_k\) through a sequence of local moves. Denote the path generated by the local moves as \((C^{(1)}_k, C^{(2)}_k, \ldots, C^{(t)}_k)\). Let \(v_{k+1} = C_{k+1} \setminus C_k\) and \(C^{(1)}_k = C_k \cup v^{(1)} \setminus u^{(1)}\) (by the definition
of local moves), then $C_{k+1}^{(1)} = C_k \cup v^{(1)}$ and $C_{k+1}$ only differ by one pair of nodes $v^{(1)}$ and $v_{k+1}$. Therefore $C_{k+1}^{(1)}$ can be reached from $C_{k+1}$ through one local move. By a similar argument, we can show that $C_{k+1}^{(1)}$ can be reached from $C_{k+1}$ through the path $(C_{k+1}^{(1)}, v_{k+1}^{(2)}, \ldots, v_{k+1}^{(l)})$, where $C_{k+1}^{(i)}$, $i \geq 2$, is defined in a similar way as $C_{k+1}^{(1)}$. Therefore, the Markov chain is irreducible.

**Proof of Theorem 5.3.1**

We first review Hajek’s (1988) theorem on the convergence of the simulated annealing algorithm in a general setting, and then we show the convergence of Algorithm 4.2.1. Assume the state space $S = \{1, 2, \ldots, q\}$ is finite (which is required in Hajek’s theorem), and each element $i \in S$ is associated with a weight $c_i$. The goal of the simulated annealing algorithm is to find the elements in the optimal set $S^0 = \{i^* \in S \mid c_i \leq c_i^* \text{ for all } i \in S\}$. Let $Z = \{Z(i, j)\}^q_{i,j=1}$ be a stochastic $q \times q$ matrix and $\sum_{j=1}^{q} Z(i, j) = 1$. Let $\{X_t\}$ be a discrete inhomogeneous Markov chain on $S$ whose one step transition probability at each step $t$ is

$$p_t(X_{t+1} = j | X_t = i) = \begin{cases} Z(i, j) & \text{if } c_i \leq c_j, i \neq j \\ Z(i, j) f_{\eta(t)}(c_i, c_j) & \text{if } c_i > c_j, \\ 1 - \sum_{m \neq i} p_t(m | i) & \text{if } i = j, \end{cases} \tag{4.15}$$

where $f_{\eta(t)}(c_i, c_j) = \exp\{(c_j - c_i)/\eta(t)\}$ and $\eta(t)$ is the temperature at step $t$. Hajek’s (1988) theorem needs the following definitions.

**Definition 4.7.1.** We say $j$ is reachable at height $E$ from $i$ if $i = j$ and $c_i \leq E$, or if there is a sequence of elements $i = i_0, i_1, i_2, \ldots, i_l = j$ such that $Z(i_k, i_{k+1}) > 0$ for $0 \leq k \leq l - 1$ and $c(i_k) \leq E$ for $0 \leq k \leq l$.

Based on the stochastic matrix $Z$, we define the neighborhood graph $G$ whose vertices are elements in $S$ and edges are those pairs $(i, j)$ satisfying $i = j$ or $Z(i, j) > 0$.

**Definition 4.7.2.** We say $G$ has weak reversibility if the following is true: for any real number $E$ and two states $x$ and $y$ in $G$, $x$ is reachable at height $E$ from $y$ if and only if $y$ is reachable at height $E$ from $x$. 

90
Hajek (1988) showed the following necessary and sufficient conditions for the convergence of simulated annealing algorithms.

**Theorem 4.7.1.** (Theorem 1 in Hajek, 1988) Let $q^*$ be the maximum of all the local but non-global maximums. If the following conditions are satisfied

(a) $\eta(1) \geq \eta(2) \geq \cdots$ and $\lim_{t \to \infty} \eta(t) = 0$,
(b) $G$ is irreducible,
(c) $G$ satisfies weak reversibility,

then

$$\lim_{t \to \infty} P(X_t \in S^0) = 1$$ (4.16)

if and only if

$$\sum_{i=1}^{\infty} \exp(-q^*/\eta(i)) = \infty.$$ (4.17)

In Algorithm 4.2.1, the state space $S_k$ is finite and each state is a connected size $k$ subgraph. The weight for each state is its edge density $Q$. The simulated annealing algorithm is designed to identify the subgraph with the largest edge density (i.e., the state with the largest weight). The stochastic matrix $Z = \alpha Z_L + (1 - \alpha) Z_G$, where $Z_L$ is the transition matrix for local moves and $Z_G$ is the transition matrix for global moves. In particular, $Z_L(i,j)$ and $Z_G(i,j)$ are the probability of proposing state $j$ from state $i$ in one local move and one global move, respectively. The temperature in Algorithm 4.2.1 is $\eta(t) = T\lceil t/M \rceil$. At each step $t$, $Z(i,j)$ is the proposal probability and $\min\{1, f_{\eta(t)}(c_i, c_j)\}$ is the acceptance probability. Therefore, the transition matrix for the inhomogeneous Markov chain at step $t$ is $p_t$ in (4.15).

Based on $Z$, we define its neighborhood graph $G(Z)$. We need to show that the neighborhood graph $G(Z)$ is weak reversible. To prove this, we first show that for $C_{k,r}, C_{k,s} \in S_k$, if $Z(C_{k,r}, C_{k,s}) > 0$ then $Z(C_{k,s}, C_{k,r}) > 0$. We know that if $Z(C_{k,r}, C_{k,s}) > 0$, then $C_{k,s}$ can be reached from $C_{k,r}$ through one global move or one local move. For the case that they are linked through one global move, it is obvious that $C_{k,s}$ can be linked to any $C_k \in S_k$ in one global move. Therefore $Z(C_{k,s}, C_{k,r}) > 0$. For
the case that they are linked through one local move, then $C_{k,s} = C_{k,r} \cup u_1 \setminus u_0$ where $u_1$ is linked to $C_{k,r}$ and $u_0$ is not a cut-vertex of $C_{k,r} \cup u_1$. Since $C_{k,s} \cup u_0 = C_{k,r} \cup u_1$ and both $C_{k,s} \cup u_0$ and $C_{k,r} \cup u_1$ are connected, we know $u_0$ is linked to $C_{k,s}$. Moreover, the removal of $u_1$ from $C_{k,s} \cup u_0$ will not fragment $C_{k,s} \cup u_0$ because $C_{k,s} \cup u_0 \setminus u_1 = C_{k,r}$ and $C_{k,r}$ is a connected subgraph. Therefore $C_{k,r} = C_{k,s} \cup u_0 \setminus u_1$ can be reached from $C_{k,s}$ in one local move and $Z(C_{k,s}, C_{k,r}) > 0$. Now we show $G(Z)$ is weak reversible. If $C_{k,r}$ is reachable at height $E$ from $C_{k,s}$, then there exist $i_0, i_1, i_2, \ldots, i_l$ such that $C_{i_0} = C_{k,r}$, $C_{i_l} = C_{k,s}$, $Z(C_{i_j}, C_{i_{j+1}}) > 0$ for $0 \leq j \leq l - 1$, and $Q(C_{i_j}) \leq E$ for $0 \leq j \leq l$. Based on the argument above, we also have $Z(C_{i_{j+1}}, C_{i_j}) > 0$ for $0 \leq j \leq l - 1$. Therefore $C_{k,s}$ is reachable at height $E$ from $C_{k,r}$. So the neighborhood graph $G(Z)$ is weak reversible.

We just showed that condition (c) of Theorem 5.5.1 is satisfied. Lemma 4.4.1 implies that condition (b) of Theorem 5.5.1 is also satisfied. Condition (a) is obviously satisfied because $T_1 > T_2 > \cdots$ and $\lim_{i \to \infty} T_i = 0$. The last condition we need is (4.17). Note that

$$\sum_{i=1}^{\infty} \exp(-q^*/\eta(i)) \geq \sum_{i=1}^{\infty} \exp(-1/T_i) = \infty$$

(4.18)

because the edge density of any graph $Q \leq 1$ and $\eta(i) = T_{\lceil i/M \rceil} \geq T_i$. Hence, Theorem 5.3.1 is proved.

**Proof of Lemma 4.4.2**

From Lemma 4.4.1, we know two subgraphs with the same size $k$ can be reached from each other. Therefore, it is sufficient to show that for $k_0 > 0$,

a) Every size $k + k_0$ connected subgraph $C'_{k+k_0}$ can be reached from any size $k$ connected subgraph $C_k$;

b) Every size $k - k_0$ connected subgraph $C'_{k-k_0}$ can be reached from any size $k$ connected subgraph $C_k$.

To prove a), we first notice that there is a size $k$ connected subgraph $C'_k \subset C'_{k+k_0}$. From Lemma 4.4.1, we know that the subgraph $C'_k$ can be reached from $C_k$. We can then move from $C'_k$
to $C_{k+k_0}$ in $k_0$ steps by adding one node at a time. Thus, we showed that $C_{k+k_0}$ can be reached from $C_k$.

To prove b), we first obtain a size $k - k_0$ connected subgraph $C_{k-k_0} \subset C_k$ by deleting non-cut vertices one at a time. From Lemma 4.4.1, we know that the subgraph $C_{k-k_0}'$ can be reached from $C_{k-k_0}$. Thus we showed that $C_{k-k_0}'$ can be reached from $C_k$.

**Proof of Theorem 5.5.1**

In Algorithm 4.3.1, the state space $S_{k_{\min}, k_{\max}}$ is finite, and each state is a connected subgraph with size $k \in [k_{\min}, k_{\max}]$. Each state $C_k$ is associated with its weight $R(C_k)$, and our goal is to find the state with the maximum weight. Let $Y = \{Y(i,j)\}_{i,j=1}^{\kappa_2}$ be the $\kappa_2 \times \kappa_2$ transition matrix, where $\kappa_2 = |S_{k_{\min}, k_{\max}}|$ and $Y(i,j)$ denotes the probability of proposing state $j$ from state $i$. We have $\sum_{j=1}^{\kappa_2} Y(i,j) = 1$. The temperature in Algorithm 4.3.1 is $T(t) = T_{[t/M]}$. Similar to the proof of Theorem 5.3.1, we have that the transition matrix for the inhomogeneous Markov chain at step $t$ can be written as $p_t$ in (4.15).

Based on $Y$, we define the neighborhood graph $G(Y)$ whose vertices are elements in $S_{k_{\min}, k_{\max}}$ and edges are those pairs $(i,j)$ satisfying $i = j$ or $Y(i,j) > 0$. To show the weak reversibility of $G(Y)$, we first show that

- **a)** if $Y(C_{k,s}, C_{k+1,s}) > 0$, then $Y(C_{k+1,s}, C_{k,s}) > 0$ for $C_{k,s}, C_{k+1,s} \in S_{k_{\min}, k_{\max}}$;
- **b)** if $Y(C_{k,s}, C_{k-1,s}) > 0$, then $Y(C_{k-1,s}, C_{k,s}) > 0$ for $C_{k,s}, C_{k-1,s} \in S_{k_{\min}, k_{\max}}$.

To prove a), notice that if $Y(C_{k,s}, C_{k+1,s}) > 0$, then $C_{k+1,s} = C_{k,s} \cup u_0$, where $u_0$ is a connected with $C_{k,s}$. Therefore, $C_{k,s} = C_{k+1,s} \setminus u_0$ and $u_0$ is not a cut-vertex of $C_{k,s}$, which implies $Y(C_{k+1,s}, C_{k,s}) > 0$. To prove b), notice that if $Y(C_{k,s}, C_{k-1,s}) > 0$, then $C_{k-1,s} = C_{k,s} \setminus u_0$, where $u_0$ is not a cut-vertex of $C_{k,s}$. Therefore, $C_{k,s} = C_{k-1,s} \cup u_0$ and $u_0$ is a neighbor of $C_{k-1,s}$, which implies that $Y(C_{k-1,s}, C_{k,s}) > 0$.

Now we show the weak reversibility of $G(Y)$. If $C_{k+k_0}$ is reachable at height $E$ from $C_k$, then we can find a path in $G(Y)$ so that $C_{k+k_0}$ can be reached from $C_k$ and for each subgraph along the path, its $R$ value is less than or equal to $E$. From a) and b) above and Lemma 4.4.1, we know
that along the same path, $C_k$ can also be reached from $C_{k+k_0}$. Therefore, $C_k$ is reachable at height $E$ from $C_{k+k_0}$. Using similar arguments, we can show that if $C_{k-k_0}$ is reachable at height $E$ from $C_k$, then $C_k$ is reachable at height $E$ from $C_{k-k_0}$. Together with the weak reversibility argument for subgraphs of the same size in Theorem 5.3.1, we have $G(Y)$ is weak reversible.

We just showed that condition (c) of Theorem 4.7.1 is satisfied. Lemma 4.4.2 implies that condition (b) of Theorem 4.7.1 is also satisfied. Condition (a) is obviously satisfied because $T_1 > T_2 > \cdots$ and $\lim_{i \to \infty} T_i = 0$. The last condition we need is (16). Since the average degree $R \leq (k_{\text{max}} - 1)/2$ and $\eta(i) = T_{\lceil i/M \rceil} \geq T_i$, we have

$$\sum_{i=1}^{\infty} \exp \left( -\frac{q^*}{\eta(i)} \right) \geq \sum_{i=1}^{\infty} \exp \left( -\frac{k_{\text{max}} - 1}{2T_i} \right) = \infty.$$  \hspace{1cm} (4.19)

Hence, Theorem 5.5.1 is proved.

4.8 Results in Section 4.5

Members of Subgraphs of a Yeast Protein Network

The annotation is available at the National Center for Biotechnology Information (http://www.ncbi.nlm.nih.gov/protein).

a. The denest subgraph with size 35, and its edge density $Q = 0.2992$.


b. A dense subgraph with size 21, and its edge density $Q = 0.3571$.

YBR193C YBR254C YDR108W YDR246W YDR308C YDR407C YDR472W YER022W YGL025C YGL127C YGR166W YHR041C YHR058C YIL021W YKR068C YML077W YMR218C YOL135C YOR115C YOR174W
Members of Subgraphs of a Stock Market Graph

a. A clique of size 82 in the stock market graph.

ONEQ PRFZ ATI JOYG FLS X PX BTU OXY CMI CNX FCX DE FLR BUCY BHP ADRE CLF MEE WLT FMC RS NBL JEC PPG CAT CEO NE CVX CM QQQQ PBR SCCO TWD FTI NOV NUE APA MUR RDS-A COP BBL DVN CNQ CRK EOG HES OII SLB AGU EQT SCHN HP ACI ANR CAM OIS UNT XEC BHI DRQ PDE TOT XOM ACWX GOOG IBM AAPL SU MOS APD E PTR LZ OMG ECA RDC RIG UPL COG DO SM

b. A dense subgraph of 338 stocks in the stock market graph.

AAPL ACI ACO ACWI ACWX ADRE AGU AKS ALB ALL AMB AMG AMN AMP AMSC AMX ANR AP APA APC APD APH ARE ARJ ASTE ATI ATR ATW AVB AVY AXE AXP BBG BBL BBT BEN BFS BGC BHI BHP BLK BLL BMO BNS BOH BOKF BRE BTU BUCY BXP BYU CAM CAT CBE CBSH CCL CEO CF CFR CHDN CHK CLB CLC CLF CLI CLR CM CME CMI CNI CNQ CNX COF COG COL COP CP CPT CRK CRS CS CSX CTBI CUK CWEI CWX CXY CYN DB DCI DD DE DEL DIS DLR DO DOV DRI DRQ DVN E ECA EGN EGP ELS EME EMN EMR ENR EOG EPR EQR ERT ERF ESS ESV ETN EV FAST FCX FFIN FLR FLS FMC FO FRT FST FTI FWT GBL GDI GEF GLF GMT GNK GOOG GR GS GWW HAL HBHC HCN HCP HDB HES HIW HME HOE HON HOT HP HR HSC HUB-B IBB IBM ICE IEX IMO IR ITR I TT ITW JEC JEF JLL JOYG JPM JRCC KALU KDN KIM KMT KRC L LAQ LECO LII LRY LUFK LUK LZ MA MAA MAC MAN MAR MCF MDC MEE MHK MICC MLI MLM MMM MOS MRO MS MSM MT MTB MTX MUR NBL NBR NDSN NE NEU NFX NHP NIHD NKE NNN NOV NSC NTRS NUE NXY NYX O OFC OII OIS OMG ONEQ OSQ OXY PBR PCAR PCH PCL PCT PDE PFF PFG PH PJC PKX PLL PNC PNR POT PPG PRA PRFZ PRK PRU PSA PTR PVH PX PXD PXP QQQQ R RBC RDC RDS-A RDS-B REG RIG RIF RL ROK ROP RRC RS RY RYN SBAC SCCO SCHN SF SFG SHAW SI SIAL SKT SLB SLG SM SNA SNH SPG SPN SPW SSL SSS SU SWK SWN TCO TD TDD TEF TEX TIF TMK TNB TOT TROW TS TXI TYC UMBF UNM UNT UPL UPS URS UTX WAB WABC VALE WBK WCC VFC WFC WFT WHR WLL WLT VM C VMi VNO WRC WRE WSO VTR WTS WYNX X XCO XEC ZEAS
Chapter 5

A Statistical Framework for Modularity-based Network Community Detection

5.1 Introduction

Networks have been the focus of much recent attention, since they describe a multitude of complex systems found in various fields, including biology, social science, information technology, finance and many others. Networks are build upon nodes and the edges (or interactions) between them. For example, social networks consist of individuals and the interactions between the individuals, such as friendship, collaboration or similar personal interest. The World Wide Web describes the webpages and their linking patterns. A stock market network models the stocks and their synchronized price fluctuations over time.

Real world networks often display high level of local inhomogeneity, with high edge density (or concentration of edges) within certain groups of nodes and low edge density between these groups. This feature is often referred to as the “community structure” (Fortunato, 2010). Communities occur in many network systems in social science, biology, political science, economics, computer science, etc. In the protein-protein network, communities are groups of proteins that carry specific functions in the cell (Chen and Yuan, 2006). In the World Wide Web, communities correspond to groups of pages that are related to the same or similar topics (Dourisboure et al., 2007).

Studying the community structures can help us better understand networks, since nodes in the same community usually share common properties. For example, the members in a Facebook friendship community usually share similar demographic attributes or personal interests (Yang et al., 2013). Moreover, different communities sometimes exhibit significantly different network properties, which makes studying them at the level of the entire network inappropriate (Newman,
Community detection can also have concrete applications. For example, identifying communities of customers with similar interests in the purchase relationship network of an online retail store can help setting up more efficient recommendation systems (Reddy et al., 2002).

Due to the great importance of finding community structures in networks, there has been a great amount of work on this topic from fields such as computer science, physics, statistics and sociology (Agrawal and Kempe, 2008; Reichardt an Bornholdt, 2006; Newman and Girvan, 2004; Snijders and Nowicki, 1997). Detecting communities in a network is not a trivial task. The number of possible partitions of the network is usually very large, especially when the number and the sizes of the communities are in general unknown.

In this chapter, the community detection techniques aim at partitioning the network into several non-overlapping segments. Therefore, the term community detection and network partition will be used interchangeably. There are other type of techniques that can detect overlapping communities. See Fortunato (2010) for a review on this topic.

The community detection approaches can be loosely divided into two classes. One major class of the community detection approaches involves maximizing some quality functions over all possible partitions of the network. Such approaches include the well-studied cut models (Flake et al., 2000), spectral clustering (Shi and Malik, 2000), modularity maximization (Newman and Girvan, 2004) and so on. Another major class of techniques are model-based approaches, i.e., fitting probabilistic models to the networks with community structures. This class includes the stochastic block model (Nowicki and Snijders, 2001; Bickel and Chen, 2009), the degree-corrected stochastic block model (Karrer and Newman, 2011), the mixed membership model (Airoldi et al., 2008) and the multivariate latent variable model (Handcock et al., 2007), etc. It is worth mentioning that, from an algorithmic perspective, many model-based approaches also lead to maximizing certain criterions, such as maximizing the profile likelihood over all possible partitions (Bickel and Chen, 2009; Zhao et al., 2012).

Among all the community detection approaches, modularity maximization is one of the most popular approaches (Fortunato, 2010). In Newman and Girvan (2004), a quality function called modularity was proposed for measuring the quality of a partition on a network. The authors
suggested that a large Newman-Girvan modularity value indicates a good partition. Subsequent works have shown empirically that partitions that maximize the modularity function often identify interesting community structures in real networks (Newman, 2004, 2006a, 2006b; Clauset et al., 2004; Chen and Yuan, 2006).

However, the network communities found by maximizing the Newman-Girvan modularity function should be judged carefully. Since the null model in the Newman-Girvan modularity function lacks a solid statistical basis, it can be difficult to determine the statistical significance of the community structure obtained from maximizing the modularity function. It has been shown that some random graphs with no community structures may have partitions with large modularity values (Guimera et al., 2004; Reichardt and Bornholdt, 2006). To better understand the results of modularity-based community detection, it is necessary to test the statistical significance of the maximized network modularity function.

In this chapter, we re-examine the null model in the Newman-Girvan modularity function and provide a statistical framework for the modularity-based community detection. We derive the modularity function under the proposed statistical framework, and propose a fast modularity maximization algorithm based on the eigen-spectrum of the modularity matrix. Based upon the proposed statistical framework, we will introduce a hypothesis testing procedure to determine the significance of the partitions obtained from maximizing the modularity function. We show that the modularity formulated under the proposed statistical framework is consistent under the degree-corrected stochastic block model framework.

The rest of the chapter is organized as follows. Section 5.2 discusses the Newman-Girvan modularity function and its connection to community structure in networks. Section 5.3 introduces the statistical framework for modularity-based community detection. Under the proposed framework, a hypothesis testing procedure is proposed for testing the significance of an identified community structure. We also discuss the connection of the proposed statistical framework to the degree-corrected stochastic block model. Section 5.4 discusses a fast modularity maximization algorithm. Section 5.5 shows the consistency of the modularity function derived under the proposed statistical framework. Section 5.6 uses synthetic networks and real world networks to demonstrate the
effectiveness of our method. Section 5.7 provides some concluding remarks.

5.2 Modularity and Community Structure

A network (or graph) \( G(V, E) \) with a set of \( n \) nodes \( V \) and a set of edges \( E \) can be represented by its adjacency matrix \( A \), where \( A_{ij} = 1 \) if there is a link from node \( i \) to node \( j \) and 0 otherwise. Node degree \( d_i \) is the number of edges connected to node \( i \). In this chapter, we are mainly concerned with simple graphs (undirected graphs with no self-loops or multiple edges), since most undirected real world networks are of this type. For simple graphs, the adjacency matrix \( A \) is a symmetric 0-1 matrix with a zero diagonal. The column sums of \( A \) are the same as the degree sequence \( d = (d_1, \ldots, d_n) \) of \( G(V, E) \). Moreover, \( m = \sum_{i<j} A_{ij} \) equals the total number of edges in \( G \).

In Newman and Girvan (2004), the authors proposed a community detection technique that relies solely on the betweenness of the edges in the network. The betweenness of an edge is defined as the number of shortest paths between all pair of nodes that pass along this edge. An edge is expected to have high betweenness if it lies between two communities, since many shortest paths linking nodes from the two different communities will go through this edge. The network will break into different fragments that correspond to communities in the original network, if the edges between them are removed.

Based on this observation, a hierarchical algorithm is proposed in Newman and Girvan (2004), in which edges with the highest betweenness are removed recursively until the network breaks down from one community of \( n \) nodes into \( n \) communities of one node. This whole process can be represented by a dendrogram showing various possible partitions of the network. To determine which partition is optimal, the authors defined a quality measure \( Q_{NG} \) referred to as the modularity.

Given a graph \( G(V, E) \) with \( n \) nodes and community assignment \( e = (e_1, \ldots, e_n) \), where \( e_i \in \{1, \ldots, K\} \) is the community that node \( i \) belongs to. The Newman-Girvan modularity \( Q_{NG} \) is defined as

\[
Q_{NG}(e, G) = \frac{1}{2m} \sum_{i,j} [A_{ij} - E(A_{ij})] \delta(e_i, e_j),
\]  

(5.1)
where $\delta(r,s) = 1$ if $r = s$ and 0 otherwise. Here $E(A_{ij})$ is the expected number of edges between node $i$ and node $j$ under the null model, in which edges are placed at random with no community structures (Newman and Girvan, 2004). The modularity function measures the “discrepancy” between the observed number of edges within the communities and the expected number of edges within the communities under the null model. If the number of edges inside the communities is close to the expected value, $Q_{NG}$ is close to 0. When $Q_{NG}$ approaches 1, it indicates strong community structure. In Newman and Girvan (2004), the partition in the dendrogram that has the largest $Q_{NG}$ value is output as the community structure.

The null model in the Newman-Girvan modularity is, however, not clearly defined. Instead, $E(A_{ij})$ is calculated based on some intuitively sensible constraints. In a detailed description of the modularity formulation in Newman (2006b), the author discussed the importance of preserving the observed degree sequence in the null model. The author proposed setting the expected node degree $E(d_i)$ equals to the observed node degree $d_i$ in the null model, i.e.,

$$\sum_j E(A_{ij}) = d_i. \quad (5.2)$$

Moreover, the author pointed out that edges should be placed entirely at random in the null model. The probability that two nodes are placed at the ends of an edge should only depend on the degrees of the nodes, i.e.,

$$E(A_{ij}) = f(d_i)f(d_j), \quad (5.3)$$

for some function $f(\cdot)$ (Newman, 2006b). With some simple calculation, it is easy to show that with constraints (5.2) and (5.3), we have

$$E(A_{ij}) = \frac{d_i d_j}{2m}. \quad (5.4)$$

Thus the expectation $E(A_{ij})$ is calculated without clearly specifying the null model. It is also worth mentioning that, in (5.4) the expected number of edges $E(A_{ij})$ can be larger than one and the expected number of self links $E(A_{ii})$ can be larger than zero. In other words, multiple edges
and self-loops are allowed in the null model of Newman-Girvan modularity.

Having such a measure of network partitions naturally leads to a new class of approaches in community detection. Rather than just using the Newman-Grivan modularity function $Q_{NG}$ as a measure of the quality of a partition, one can instead directly try to find the partition that maximizes it. Brandes et al. (2008) showed that finding the partition that maximizes the modularity function for a given graph is NP-complete. Numerous heuristic approaches have been proposed to find partitions that maximizes the Newman-Girvan modularity function (Newman, 2006a; Agrawal and Kempe, 2008; Reichardt and Bornholdt, 2006; Clauset et al., 2004; Massen and Doye, 2005; Wang et al., 2008). Finding the community assignment that maximizes the modularity function is still an active research topic.

One remaining issue in the modularity-based community detection approaches is the interpretation of the partition obtained by maximizing $Q_{NG}$. Test of hypotheses can not be performed on the community structure obtained from maximizing the modularity function, for the null model in the Newman-Girvan modularity function lacks of a solid statistical basis. In Newman and Girvan (2004), the authors pointed out that networks with strong community structure typically have maximum modularity value $\max_e Q_{NG}(e, G) \in [0.3, 0.7]$, which is widely used as a general rule of thumb for subsequent works on modularity-based community detection. However, in general, a large Newman-Girvan modularity value does not necessarily indicate a good partition. Random graphs from the Erdős-Rényi model are not supposed to have community structures. Because the probability of having an edge between any pair of nodes is the same and every node is treated equally. Still, Erdős-Rényi graphs could have partitions that has large Newman-Girvan modularity values (Guimera et al., 2004; Reichardt and Bornholdt, 2006).

From the definition of modularity, a network should only be considered to have community structure if its maximized modularity value is significantly larger than the maximized modularity value of graphs from the null model. To decide the significance of the community structures in a network, it requires a well defined statistical framework. In the next section, we will describe a statistical framework for the modularity-based community detection.
5.3 A Statistical Framework for Modularity

Given a graph $G(V, E)$ with $n$ nodes and degree sequence $d = (d_1, \ldots, d_n)$, the null model in the definition of modularity measure is chosen to be a random graph model with no community structure. It specifies a distribution $p(\cdot)$ over the sample space $\Sigma$.

Intuitively, the graphs in the sample space $\Sigma$ should share some basic structural properties with $G$. In real world networks, the distribution of edges are highly inhomogeneous, with both global inhomogeneity, i.e., many vertices with small degrees and a few vertices with large degrees, and local inhomogeneity, i.e., high concentration of edges within certain groups of nodes and low concentration of edges between these groups (Fortunato, 2010). To study the local inhomogeneity, it is important to control for the global inhomogeneity. In other words, in order to study the community structure, it is important to control for the degree sequence. It is therefore, desirable to preserve the observed degree sequence in the null model.

In the formulation of the Newman-Girvan’s modularity, constraint (5.2) also aims at preserving the degree sequence in the null model. Therefore, for the sample space in our statistical framework, we fix the degree distribution of graphs from the null model at $d$ and denote the sample space as $\Sigma_d$. Since most real world networks do not contain self-loops nor multiple edges, i.e., $A_{ij} \in \{0, 1\}$ and $A_{ii} = 0$, it is desirable to preserve such properties in the sample space of the null model (Massen and Doye, 2005; Calferi et al., 2010). In the following analysis, we define the sample space $\Sigma_d$ as the set of all simple graphs with degree sequence $d = (d_1, \ldots, d_n)$.

Given the degree sequence, it is reasonable to assume that there is no preference for any configuration and every graph from the sample space is equally likely to occur, i.e.,

$$p(g) = \frac{1}{|\Sigma_d|}, \text{ for } g \in \Sigma_d. \quad (5.5)$$

Here $|\Sigma_d|$ is the total number of graphs in $\Sigma_d$. Section 5.3.2 discusses another motivation for this
null model. With the null model in (5.5), our modularity function is defined as

$$Q(e, G) = \frac{1}{2m} \sum_{i,j} [A_{ij} - E_{p, \Sigma_d}(A_{ij})] \delta(e_i, e_j).$$

(5.6)

Here the expectation $E_{p, \Sigma_d}(\cdot)$ is taken with respect to $p(\cdot)$ on $\Sigma_d$.

To calculate $P_{ij} = E_{p, \Sigma_d}(A_{ij})$, it is easy to see that

$$P_{ij} = \frac{|\Sigma_d|_{A_{ij}=1}}{|\Sigma_d|} = 1 - \frac{|\Sigma_d|_{A_{ij}=0}}{|\Sigma_d|}.$$  

(5.7)

Here $|\Sigma_d|_{A_{ij}=1}$ is the total number of simple graphs with degree sequence $d$ and have an edge between node $i$ and node $j$. Similarly, $|\Sigma_d|_{A_{ij}=0}$ is the total number of simple graphs with degree sequence $d$ and have no edge between node $i$ and node $j$. The next theorem gives an approximation to $P_{ij}$ under the asymptotic setting.

**Theorem 5.3.1.** Given a simple graph $G(V, E)$ with degree distribution $d = (d_1, \ldots, d_n)$. Denote $m = \frac{1}{2} \sum_{i=1}^n d_i$ as the total number of edges in $G(V, E)$. Assuming $\max_i d_i = o(m^{1/4})$, we have

$$\lim_{n \to \infty} \sup_{\bigcup_{d=(d_1, \ldots, d_n)} |\Sigma_d|} |P_{ij} - (1 - e^{-\frac{d_i d_j}{2m}})| = 0.$$  

(5.8)

See Section 5.8 for the proof. In the theorem, the condition $\max_i d_i = o(m^{1/4})$ describes the rate that the node degree increases. Since it is true that $\max_i d_i \geq m/n$, this condition also implies that $\max_i d_i = o(n^{1/3})$, which describes the sparsity as the graph size grows. Moreover, with the condition $\max_i d_i = o(m^{1/4})$, it is easy to see that $\max_{i,j} \frac{d_i d_j}{2m} \to 0$. By $\lim_{x \to 0} 1 - e^x = x$, we have the following result.

**Corollary 5.3.1.** Given a simple graph $G(V, E)$ with degree distribution $d = (d_1, \ldots, d_n)$. Denote
$m$ as the total number of edges. Assuming $\max_i d_i = o(m^{1/4})$, we have

$$
\lim_{n \to \infty} \sup_{d=\{d_1, \ldots, d_n\}} \left| \frac{1}{\sum_{d_1}^{\infty} d_d} \right| P_{ij} - \frac{d_id_j}{2m} = 0. 
$$

(5.9)

Under the asymptotic setting, the probability of having an edge between node $i$ and node $j$ is well approximated by the expected number of edges formulated in Newman and Girvan (2004).

When the graph is of moderate size, the approximation in (5.8) or (5.9) may not be satisfactory. In this case, we can approximate the linking probability $P_{ij}$ using Monte Carlo simulation. With $N$ samples $g_1, \ldots, g_N$ from the null distribution in (5.5), we can estimate $P_{ij}$ by

$$
P_{ij} = \sum_{l=1}^{N} A_{ij}^{(l)}/N, 
$$

(5.10)

where $A_{ij}^{(l)}$ is the adjacency matrix of $g_l$. The following MCMC algorithm provides an easy way to sample from the uniform distribution over $\Sigma_d$. Denote $\{i, j\}$ as an edge between node $i$ and node $j$. At each step of the MCMC algorithm, randomly choose two edges $\{x, y\}$ and $\{u, v\}$ from the current graph $g$ with distinct nodes $x, y, u, v$. If there are no edges between the pair $x$ and $u$ and the pair $y$ and $v$, then the Markov chain moves to a new state $g'$ constructed by replacing the edges $\{x, y\}$ and $\{u, v\}$ from the current graph $g$ by two new edges $\{x, u\}$ and $\{y, v\}$; otherwise, the Markov chain stays at the current graph $g$. A sample $g_i$ is taken after every $N_0$ MCMC steps.

Uniformly generate graphs with fixed degree sequence is a well studied problem. One of the most straightforward approach is the aforementioned MCMC algorithm, which is often referred to as the “rewiring” method. There are other more sophisticated sequential importance sampling schemes for estimating $P_{ij}$ (Blitzstein and Diaconis, 2010; Zhang and Chen, 2013).

5.3.1 Testing the Significance of Community Structure

In this section, we introduce a hypothesis testing procedure built under the introduced statistical framework that can test the significance of an identified structure.
Given a graph $G(V, E)$ and a community assignment $e^* = (e_1^*, \ldots, e_n^*)$, the statistical significance of the partition is

$$P[Q(e^*, G) \leq \max_e Q(e, g)],$$

(5.11)

for $g$ from the null model in (5.5). A small $p$-value indicates that graphs from the null model are very unlikely to have a maximized modularity value as large as $Q(e^*, G)$, and the partition $e^*$ we obtained from $G$ is statistically significant. A straightforward way to estimate the $p$-value in (5.11) is to generate samples $g_1, \ldots, g_N$ using the MCMC algorithm and then approximate (5.11) with

$$\frac{1}{N} \sum_{i=1}^{N} I \left( Q(e^*, G) \leq \max_e Q(e, g_i) \right).$$

(5.12)

### 5.3.2 Connection to Degree-Corrected Stochastic Block Model

The stochastic block models is one of the most widely used models for networks with communities. Consider a graph $G(V, E)$ with $n$ nodes and adjacency matrix $A$. Under the stochastic block model, there are $K$ classes (or blocks) and each node must belong to only one of the classes. Let $c = (c_1, \ldots, c_n)$ denote the true community labels where $c_i$ is the community that node $i$ belongs to. Under the stochastic block model, each $A_{ij}$ is an independent Bernoulli random variable, with

$$E[A_{ij} | c] = W_{c_i c_j},$$

(5.13)

where $W$ is the matrix of linking probabilities, i.e., $W_{rs}$ is the probability that a node in block $r$ is linked to a node in block $s$.

The stochastic block model can produce a large variety of networks. When fitted to an observed network, the stochastic block model can uncover the underlying block (or community) structure. However, the stochastic block model has limitations in its application. For every block (or community) in the model, the nodes are assumed to be homogeneous, i.e., within the same block all nodes are considered to be equivalent. This is problematic since the model does not allow hub nodes (nodes that have significantly more links than the others), which are frequently observed in
real networks. Fitting a stochastic block model to networks that have highly inhomogeneous degree distributions can lead to inaccurate results (Karrer and Newman, 2011).

To address this problem, Karrer and Newman (2011) proposed to add parameters that account for the degree correction in the stochastic block model. The new model is referred to as the degree corrected stochastic block model. The authors show empirically that when there is heterogeneity present in the degree distribution, the degree-corrected stochastic block model fits better than the standard stochastic block model.

In Peng and Carvalho (2013), a generalized degree-corrected stochastic block model with $K$ blocks is defined as

$$A_{ij} \sim Bernoulli(q^{-1}(\theta_i + \theta_j + Z_{c_i c_j})),$$

(5.14)

where $q(.)$ is a link function, $Z_{c_i c_j}$ reflects the linking probability between block $i$ and block $j$, and $\theta = (\theta_1, \ldots, \theta_n)$ is a vector of node specific parameters. With a logit link, the likelihood of the degree-corrected block model can be written as

$$P(G|\theta, Z) = \prod_{i<j} [\text{logit}^{-1}(\theta_i + \theta_j + Z_{c_i c_j})]^{A_{ij}} [1 - \text{logit}^{-1}(\theta_i + \theta_j + Z_{c_i c_j})]^{1-A_{ij}}. \quad (5.15)$$

If we consider the degree-corrected stochastic block model as the underlying model, the null model is naturally taken to be a degree-corrected stochastic block model with only one block, i.e., no block structures in the network. We then have

$$A_{ij} \sim Bernoulli(\text{logit}^{-1}(\theta_i + \theta_j + Z)),$$

(5.16)

Since there is only one block, $Z$ matrix degenerates to a single parameter. This is essentially the well-studied logistic linear model (or $\beta$-model) for network data (Holland and Leinhardt, 1981; Chatterjee et al., 2001; Park and Newman, 2004; Blitzstein and Diaconis, 2011). The likelihood in (5.15) can be simplified to

$$P(G|\theta) = \frac{e^{\sum_{i} Z_{d_i}/2 + \sum_{i} \theta_i d_i}}{\prod_{i<j} (1 + e^{\theta_i + \theta_j + Z})}. \quad (5.17)$$

106
We can see the degree sequence $d = (d_1, \ldots, d_n)$ is the sufficient statistic for parameters $\theta = (\theta_1, \ldots, \theta_n)$. Conditioning on $d$, $P(G|\theta, d)$ becomes uniform distribution, formalizing the null model intuition that all graphs exhibiting the same degree sequence be considered as equally likely. Therefore, conditioning on the degree sequence, the null model of the degree-corrected stochastic block model becomes the proposed null model in (5.5). When making inference from the null model of the degree corrected stochastic block model, fixing the degree sequence can remove the nuisance parameters and create a basis for exact inference where accurate p-values can be obtained (Lehmann, 1986).

5.4 Modularity Maximization

In the section, we discuss modularity maximization techniques for finding the partition that maximizes the modularity function, i.e.,

$$\hat{e} = \arg \max_{e=(e_1, \ldots, e_n)} \max_{e_i \in \{1, \ldots, n\}} Q(e, G). \quad (5.18)$$

Modularity based community detection techniques are among the most popular approaches in detecting communities in networks (Fortunato, 2010). Existing approaches for maximizing the modulation function come from various fields, such as computer science, physics, sociology and statistics. Some are fast techniques that can be applied to large graphs, but may not find good approximations to the optimum of the modularity function (Clauset et al., 2004; Newman, 2004). Some are accurate methods that find good approximations to the optimum but are limited to graphs of moderate sizes (Massen and Doye, 2005; Guimera et al, 2004; Agrawal and Kempe, 2008). Some algorithms achieve a good balance between accuracy and complexity (Newman, 2006a; Wang et al., 2008). See Chapter VI in Fortunato (2010) for a thorough review.

In this chapter, we consider two modularity maximization approaches. For small graphs with no more than a hundred nodes, we use the linear programming approach proposed in Agrawal and Kempe (2008). The authors suggested maximizing the modularity function within the framework
of mathematical programming. For graph of moderate sizes, the algorithm runs fairly fast and has more accurate results compared to approaches designed for large graphs. For larger graphs, we propose a fast spectral clustering algorithm, which is a simplification of the algorithm discussed in Newman (2006b).

To simplify the notation, we define an assignment matrix $B_{n \times K}$, which is a 0-1 matrix with $B_{ij} = 1$ if the $i$-th node is in the $j$-th community and $B_{ij} = 0$ otherwise. Each row of $B$ sums to unity, and the columns $b_1, \ldots, b_K$ of $B$ are mutually orthogonal. Maximizing modularity in (5.18) can therefore be expressed as

$$\max_B \{\text{Trace}(B^TMB)\} \text{ s.t. } \text{Trace}(B^TB) = n, \tag{5.19}$$

where $M = A - E_{p, \Sigma_d}(A)$ is the modularity matrix. Newman (2006b) pointed out the intimate relationship between community structures and the eigen-spectrum of the Newman-Girvan modularity matrix. Here we extend that relationship to the community structure and the eigen-spectrum of the modularity matrix $M = A - E_{p, \Sigma_d}(A)$.

Denote the eigenvalues of $M$ as $\lambda_1, \ldots, \lambda_n$ and the corresponding normalized pairwise orthogonal eigenvectors $v_1, \ldots, v_n$. Without loss of generality, assume $\lambda_1 \geq \ldots \geq \lambda_n$. Denote $\kappa = \sum_{i=1}^{n} I(\lambda_i > 0)$ as the number of positive eigenvalues. We have

$$Q = \text{Trace}(B^TMB) = \sum_{i=1}^{n} \sum_{k=1}^{K} \lambda_i (v_i^T b_k)^2. \tag{5.20}$$

Maximizing the modularity is equivalent to choosing $K - 1$ orthogonal columns $b_1, \ldots, b_{K-1}$ such that the summation in (5.20) is maximized. Since $v_1, \ldots, v_n$ form an orthonormal basis of a $n$-dimensional vector space, we have

$$b_k = \sum_{i=1}^{n} \alpha_{ki} v_i, \text{ for } k = 1, \ldots, K,$$

where

$$\alpha_{ki} = v_i^T b_k.$$
Therefore, we have
\[ Q = \sum_{i=1}^{n} \sum_{k=1}^{K} \lambda_i (v_i^T b_k)^2 = \sum_{i=1}^{n} \lambda_i \left( \sum_{k=1}^{K} a_{ki}^2 \right). \]  \hspace{1cm} (5.21)

This shows that the major contribution to the modularity value comes from the projection of columns \( b_1, \ldots, b_K \) onto the subspace spanned by the leading eigenvectors. For a partition that achieves large \( Q \), vectors \( b_1, \ldots, b_K \) necessarily have large projections onto the leading eigenvectors with positive eigenvalues. Newman(2006b) showed that if we do not have the binary constraint on the entries in \( B \), \( Q \) will be maximized when \( b_k \) is proportional to \( v_k, k = 1, \ldots, K - 1 \), and when we have as many orthogonal columns in \( B \) as there are positive eigenvalues, i.e., \( K = \kappa + 1 \). However, the entries in \( B \) are binary. With this constraint, the number of positive eigenvalues \( \kappa \) becomes an upper bound for \( K - 1 \).

When the graph is large, we can consider dropping the terms in (5.20) that are proportional to the \( n - K + 1 \) smallest of the eigenvalues \( \lambda_i \) (Newman, 2006b; Ng et al., 2001; Wang et al., 2008). Consider the \( K - 1 \) largest positive eigenvalues \( \lambda_1, \ldots, \lambda_{K-1} \) and form a diagonal matrix \( \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_{K-1}) \). Choose their corresponding eigenvectors \( v_1, \ldots, v_{K-1} \) and form a matrix \( V = (v_1|\ldots|v_{K-1}) \). Then we have

\[
\max_B Q = \max_B \sum_{i=1}^{n} \sum_{k=1}^{K} \lambda_i (v_i^T b_k)^2 \approx \max_B \sum_{i=1}^{K-1} \sum_{k=1}^{K} \lambda_i (v_i^T b_k)^2 \\
= \max_B (\text{Trace}(B^T V \Lambda V^T B)) = \max_B \left( \text{Trace}(B^T V \frac{1}{2} \Lambda \frac{1}{2} V^T B) \right) \\
= \max_B \left( \text{Trace}(V^T B \Lambda \frac{1}{2} V^T B) \right) = \max_B \sum_{k=1}^{K} ||w_k||^2,
\]

where \( w_k = \sum_i y_i I(B_{ik}=1) \) and \( y_i \) is the \( i \)-th row of matrix \( VA \frac{1}{2} \). The modularity maximization is now a problem of grouping vectors \( y_i \) into \( K \) groups such that the magnitude of vector \( u_k \) is maximized. One simple approach would be applying the \( k \)-means clustering on normalized vectors \( y_1, \ldots, y_n \) (Ng et al., 2001).

Here is a summary of our approach of detecting up to \( K \leq \kappa + 1 \) communities in the graph \( G(V, E) \).
1. Find the modularity matrix $M$, its eigenvalues $\lambda_1, \ldots, \lambda_n$ and their corresponding normalized eigenvectors $v_1, \ldots, v_n$.

2. For each value $k$, $2 \leq k \leq K$:
   a. Construct the corresponding diagonal matrix $\Lambda$ and the eigenvector matrix $V$. Calculate $Y = V\Lambda^{1/2}$.
   b. Perform $k$-means clustering on the normalized rows $y_i$ of matrix $Y$.
   c. With the membership output from the $k$-means clustering, calculate modularity function $Q_k$.

3. Output the $k$ that has the largest $Q_k$ and its corresponding community labels.

It is worth mentioning that in step 2(b), more sophisticated approaches can be used. This type of problems are referred to as the vector partitioning problems, i.e., grouping vectors $y_i$ into $K$ groups such that the magnitude of vector $w_k$ is maximized (Alpert et al., 1999).

5.5 Consistency Under the Degree-Corrected Stochastic Block Model

The notion of consistency for a community detection criterion is first defined in Bickel and Chen (2009). A community detection criterion $Q$ is defined to be strongly consistent if $\hat{c} = \arg \max_e Q(e, G)$ satisfies

$$P(\hat{c} = e) \to 1 \text{ as } n \to \infty,$$

(5.22)

where $e$ is the true community labels for the nodes. In Zhao et al. (2012), the authors defined weak consistency for a community detection criterion $Q$ as

$$\forall \epsilon > 0 \quad P \left[ \left( \frac{1}{n} \sum_{i=1}^{n} 1(\hat{c}_i \neq c_i) \right) < \epsilon \right] \to 1 \text{ as } n \to \infty.$$  

(5.23)
When the community detection criteria are invariant under permutations of the community labels, there are identifiability issues with these definitions of consistency. Zhao et al. (2012) suggested replacing \( \hat{c} = c \) with \( \hat{c} \) and \( c \) belongs to the same equivalent class under permutation. It is worth mentioning that in both type of consistency, the number of communities \( K \) is assumed to be a known quantity. Therefore, all the modularity consistency properties are shown for

\[
\hat{c} = \arg \max_{e=(e_1, \ldots, e_n), \quad e_i \in \{1, \ldots, K\}} Q(e, G)
\]  

(5.24)

Under the proposed framework in Section 5.3, assuming the graph satisfy the sparsity condition in Corollary 5.3.1, we detect communities by finding the maximizer of the modularity function, i.e.,

\[
\hat{c} = \arg \max_{e=(e_1, \ldots, e_n), \quad e_i \in \{1, \ldots, K\}} \frac{1}{2m} \sum_{i,j} (A_{ij} - \frac{d_i d_j}{2m}) \delta(e_i, e_j).
\]  

(5.25)

In this section, we show that (5.25) is consistent under the degree corrected stochastic block model framework. The degree-corrected stochastic block model is defined as follows:

1. Each node \( i \) has a pair of latent variables \((c_i, \theta_i)\), where \( c_i \in [1, \ldots, K]\) and \( \theta_i \) takes values in \( x_1 \leq \ldots \leq x_N \). For identifiability, assume \( E[\logit(\theta_i)] = 1/2 \).

2. The marginal distribution of \( c = (c_1, \ldots, c_n) \) follows a multinomial distribution with parameters \((\pi_1, \ldots, \pi_K)\).

3. Edges \( A_{ij} \) are independent Bernoulli random variable with

\[
\logit(E[A_{ij} | c, \theta]) = \theta_{c_i} + \theta_{c_j} + Z_{c_ic_j},
\]

where \( Z \) is a symmetric \( K \times K \) matrix.

Parameter \( Z_{ij} \) capture the within and between community linking probabilities. Self loops are allowed in the model for simplicity, i.e., \( E(A_{ii}) > 0 \). This makes no difference in the results compared to the setting where diagonal items are forced to be zero. However, allowing self loops
makes the notations much simpler. It is also worth mentioning that, the standard stochastic block model is a special case of the degree-corrected stochastic block model. One simply needs to set $\theta_i$ to zero.

Define $\Pi_{K \times M}$ to be the joint distribution of $(c_i, \theta_i)$, i.e., $P(c_i = a, \theta_i = x_u) = \Pi_{au}$. Here we do not assume that $m\theta$ and $c$ are independent, since it is possible that the community label and the node specific random variables are not independent in the model. Follow Bickel and Chen (2009) and Zhao et al. (2012), we define the expected degree $\lambda_n = n\rho_n$, where $\rho_n = P(A_{ij} = 1) \to 0$. When $\lambda_n \to \infty$, we naturally have $n \to \infty$.

Next we state our results on the consistency of the proposed modularity function under the degree-corrected stochastic block model.

**Theorem 5.5.1.** Under the degree-corrected stochastic block model, if for all $a \neq b$, the parameters satisfy

$$\logit^{-1}(x_u + x_v + Z_{ab}) < \frac{1}{H_0} \left( \sum_{b'v'} \logit^{-1}(x_{u'} + x_{v'} + Z_{ab'})\Pi_{b'v'} \right) \left( \sum_{a'u'} \logit^{-1}(x_{u'} + x_{v'} + Z_{aa'})\Pi_{a'u'} \right)$$

and

$$\logit^{-1}(x_u + x_v + Z_{aa}) > \frac{1}{H_0} \left( \sum_{b'v'} \logit^{-1}(x_{u'} + x_{v'} + Z_{ab'})\Pi_{b'v'} \right)^2$$

(5.26)

(5.27)

Here $H_0 = \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\Pi_{av}\Pi_{bv}$. Then the proposed modularity $Q(e, G)$ in (5.25) is strongly consistent when $\frac{\lambda_n}{\log n} \to \infty$ and weakly consistent when $\lambda_n \to \infty$.

See Appendix B for the proof. The constraints (5.26) and (5.27) on the parameters in Theorem 5.5.1 essentially requires that links are more likely to establish within communities rather than between communities. The consistency result suggests that if the graphs are from a degree corrected stochastic block model with $K$ communities, then the community labels obtained from maximizing the modularity function $Q$ will be close to the true community labels as the number of nodes goes to infinity. By setting $\theta$ to zero in Theorem 5.5.1, we immediately have

**Corollary 5.5.1.** Under the stochastic block model, if the parameters satisfy

$$\logit^{-1}(Z_{ab}) < \frac{1}{H_0} \left( \sum_{b'} \logit^{-1}(Z_{ab'})\pi_{b'} \right) \left( \sum_{a'} \logit^{-1}(Z_{a'b})\pi_{a'} \right)$$

(5.28)
and

\[ \logit^{-1}(Z_{aa}) > \frac{1}{H_0} \left( \sum_{b'} \logit^{-1}(Z_{ab'}) \pi_{b'} \right)^2 \]  \hspace{1cm} (5.29)

Here \( H_0 = \sum_{ab} \logit^{-1}(Z_{ab}) \pi_a \pi_b \). Then the proposed modularity \( Q(e, G) \) in (5.25) is strongly consistent when \( \frac{\lambda_n}{\log n} \to \infty \) and weakly consistent when \( \lambda_n \to \infty \).

5.6 Numerical Examples

In this section, we denote the Newman-Girvan modularity function as \( Q_{NG} \). The modularity function calculated using the approximation in Theorem 5.3.1 will be referred to as the asymptotically approximated modularity, denoted as \( Q_{asym} \). Moreover, we refer to the modularity function approximated using MCMC algorithm as the MCMC approximated modularity, denoted as \( Q_{MCMC} \).

The linear programming algorithm in Agrawal and Kempe (2008) is coded in C++ and implemented in a CPLEX environment. The fast modularity maximization algorithm introduced in Section 3.2 is coded and implemented in R. All examples were run on a MacBook Pro with 2.26 GHz Intel Core 2 Duo processor.

5.6.1 Erdos-Renyi Random Graphs

For many real world networks, a high modularity value usually indicates a strong community structure. However, this is not true in general. It has been shown that random graphs could have partitions with large modularity values (Guimera et al., 2004; Reichardt an Bornholdt, 2006). To interpret the results from modularity-based community detection, it is necessary to test the significance of the maximized modularity value. In this section, we look at the maximized modularity function of graphs from the Erdos-Renyi random graph model and demonstrate the use of the hypothesis testing procedure proposed in Section 5.3.1.

In an Erdos-Renyi graph, edges are established independently between each pair of nodes with equal probability \( p \). There should be no community structures in an Erdos-Renyi graph, since all nodes are treated equally. However, graphs generated from the Erdos-Renyi model can have large modularity values. Figure 5.1 is the histogram of \( \max_e Q_{NG}(e, G) \) of 100 Erdos-Renyi graphs with
$n = 100$ and $p = 0.05$. The average of the 100 maximum modularity $\max_e Q_{NG}(e, G)$ is 0.399, the minimum of the 100 maximum modularity is 0.350 and the maximum of the 100 maximum modularity is 0.451. Based on the general rule of thumb, all 100 Erdos-Renyi graphs are considered to have strong community structure since they all have $\max_e Q_{NG}(e, G) > 0.3$.

Figure 5.1: Histogram of the maximized $\max_e Q_{NG}(e)$ for the 100 Erdos-Renyi graphs.

This example shows that using the Newman-Girvan modularity and following the general rule of thumb may lead to false conclusions. To better understand the community detection results, it is necessary to perform the following test of hypothesis. We randomly generate 100 Erdos-Renyi graphs $G_1, \ldots, G_{100}$ with $n = 100$ and $p = 0.05$. For each graph $G_i$, we calculate its maximized modularity value $\max_e Q_{asym}(e, G_i)$ and $\max_e Q_{MCMC}(e, G_i)$. To perform the hypothesis testing, for each graph $G_i$, we generate 1000 MCMC samples $g_1^{(i)}, \ldots, g_{1000}^{(i)}$ from the null model in (5.5). Then we estimate the $p$-values using

$$p_1^{(i)} = \frac{1}{1000} \sum_{j=1}^{1000} I \left( \max_e Q_{asym}(e, G_i) \leq \max_e Q_{asym}(e, g_j^{(i)}) \right)$$

and

$$p_2^{(i)} = \frac{1}{1000} \sum_{j=1}^{1000} I \left( \max_e Q_{MCMC}(e, G_i) \leq \max_e Q_{MCMC}(e, g_j^{(i)}) \right).$$

Figure 5.2 are the histograms for the 100 $p_1$-values and 100 $p_2$-values. Under the null model, the
$p$-values are roughly uniformly distributed.

![Histograms showing $p_1$-values and $p_2$-values.](image)

Figure 5.2: (Left) Histogram of the 100 $p_1$-values; (Right) Histogram of the 100 $p_2$-values.

If we set the significance level at 0.05, Type I error is estimated to be 0.03 and 0.07 respectively for the two tests of hypothesis. This indicates that the level of the proposed hypothesis testing procedure is controlled. This example shows that simply following the general rule of thumb could lead to false conclusions, and the test of hypothesis is needed in order to decide the significance of an identified community structure.

### 5.6.2 Synthetic Modular Networks

In this section, we test our community detection procedure on networks that are known to have community structures. We generate graphs from the standard stochastic block model with $n = 1000$ nodes and three blocks with sizes 200, 300 and 500 respectively. The linking probability matrix for the stochastic block model is set to $B = \begin{pmatrix} 0.5 & 0.1 & 0.1 \\ 0.1 & 0.3 & 0.1 \\ 0.1 & 0.1 & 0.2 \end{pmatrix}$. In this model, links are more likely to be established within the blocks and less likely to be established between the blocks, and graphs generated from this model should have community structure.

Figure 5.3 is the histogram of $\max_e Q_{NG}(e, G)$ of 100 graphs generated from the stochastic block model. The maximum of the 100 maximum modularity is 0.263. Based on the general
rule of thumb, none of the 100 graphs are considered to have strong community structure since \( \max_e Q_{NG}(e,G) < 0.3 \). This example also shows that following the general rule of thumb could lead to false conclusions. To test the significance of the identified community structures, we need to perform hypothesis testing.

![Histogram](image)

**Figure 5.3**: Histogram of the maximized \( \max_e Q_{NG}(e) \) for 100 graphs generated from stochastic block model with community structure.

In this simulation study, we generated 100 graphs \( G_1, \ldots, G_{100} \) from the stochastic block model. Since the graph is large, we will only study the asymptotically approximated modularity function \( Q_{asym} \). For each graph \( G_i \), we calculate its maximized modularity value \( \max_e Q_{asym}(e,G_i) \). To perform the hypothesis testing, for each graph \( G_i \), we generate 1000 MCMC samples \( g_1^{(i)}, \ldots, g_{1000}^{(i)} \) from the null model in (5.5) and estimate the \( p \)-value.

For all the 100 graphs, the \( p \)-values of the identified community structure are estimated to be 0, which suggest that all the 100 graphs are considered to have strong community structures. This indicates that the proposed hypothesis testing procedure has high power. Moreover, for all the 100 graphs, the community assignment has a misclassification rate of 0. This agrees with the
consistency results in section 5.5.

5.6.3 Krebs’ Network of Books on American Politics

The Krebs’ network of books on American politics has 105 vertices and 441 edges. Each node represents a book on US politics that is sold by the online bookseller Amazon.com. Each edge between a pair of nodes represents the frequent co-purchasing of the two books by the same buyers, which is indicated by the “customers who bought this book also bought these other books” feature on Amazon.com. Newman (2006a) provided a classification of these 105 books as liberal (l), conservative (c) or neutral (n) based on a reading of the descriptions and reviews of the books posted on Amazon.com.

This network is of moderate size and we will only use the MCMC approximated modularity function $Q_{\text{MCMC}}$. The modularity function $Q_{\text{MCMC}}$ is calculated based on 1000 MCMC samples, each taken after 1000 MCMC moves.

Using the modularity maximization algorithm by Agrawal and Kempe (2008), $Q_{\text{MCMC}}$ is maximized at $K = 5$ with $\max Q_{\text{MCMC}} = 0.535$. After we obtain the partition, it is necessary to perform the test of hypothesis to decide its significance. Figure 5.4 is the histograms of the maximized $Q_{\text{MCMC}}$ of the 1000 MCMC samples from the null model. The $p$-value is estimated to be 0, which indicates the community structure we identified in the network is significant.

Among the 5 identified communities in Figure 5.5, two large communities are obviously the liberal community and the conservative community. Based on the members in the communities, the three smaller communities are roughly neutral, neutral conservative and conservative. One interesting observation is that the smaller conservative community on the rightmost is almost only connected to the big conservative community, and has almost no connections to the liberal communities and neutral communities. This indicates that if customers buy books from this small conservative community, it is very unlikely that they will buy books from the liberal community or the neutral community. Some examples of the books in this more extreme conservative community are “Useful Idiots: How Liberals Got It Wrong in the Cold War and Still Blame America First” by Mona Charen, “The Right Man: The Surprise Presidency of George W. Bush” by David Frum.
Figure 5.4: Histogram of the maximized $Q_{MCMC}$ for the 1000 MCMC samples.

and “The Savage Nation: Saving America from the Liberal Assault on Our Borders, Language and Culture” by Michael Savage.

Figure 5.5: Krebs’ network of books on American politics. Liberal books are the square nodes, conservative books are the triangle nodes and neutral books are the circle nodes.
5.7 Discussion

In this chapter, we provide a statistical framework for the modularity function. Based on the proposed statistical framework, we introduce a significance testing procedure for the obtained community structure. The proposed modularity function and statistical testing procedure perform well with both simulated networks and real networks. We also show that under the degree-corrected stochastic block model setting, the proposed modularity function is consistent as a community detection criterion.

It is worth mentioning that modularity function $Q$ can have negative values. A negative modularity value indicates the within communities number of edges are less than its expected value under the null model. In fact, a partition with large negative modularity suggests the existence of multipartite structure (Newman, 2006b). To detect the multipartite structure in the network, one can try to minimize the modularity function. The statistical framework proposed in the chapter can also be used to test the significance of a detected multipartite structure.

5.8 Proof of the Main Results

Proof of Theorem 5.3.1

Let $(D_{ij})_{n \times n}$ be an $n \times n$ symmetric zero-one matrix, where $D_{ij} = 0$ means the $(i,j)$th position is a structural zero. Define $\lambda = \sum_{i=1}^{n} d_i (d_i - 1)/(4m)$, $\gamma = \sum_{i<j=0} d_i d_j / 2m$ and $d_{max} = \max_i d_i$. Also define $\tilde{d} = 2 + d_{max}(1.5d_{max} + x_{max} + 1)$, where $x_{max}$ is the maximum of the degree sequence of $(D_{ij})_{n \times n}$. Theorem 4.6 in Mckay (1985) state that

**Theorem.** Suppose $d_{max} \geq 1$ and $\tilde{d} \leq \epsilon_1 m$, where $\epsilon_1 < 2/3$. Then as $n \to \infty$, the number of simple graphs with degree sequence $d$ is uniformly

$$
\frac{(2m)!}{(m)!(2m)^{\frac{n}{2}}} \exp\{-\lambda - \lambda^2 - \gamma + O(\tilde{d}^2/m)\},
$$

as $n \to \infty$. 

119
The conditions needed for this theorem are $d_{\text{max}} \geq 1$ and $\tilde{d} \leq \epsilon_1 m$, where $\epsilon_1 < 2/3$. With $d_{\text{max}} = o(m^{1/4})$, the two conditions in Mckay’s theorem are satisfied and also $O(d^2/m)$ becomes $o(1)$. For the set $\Sigma_d$ of $n \times n$ symmetric zero-one tables with column sums $d$ and a zero diagonal, the matrix $(D_{ij})_{n \times n}$ has 0 on its diagonal and 1 elsewhere and $\gamma = 0$. Plugging these simplifications into expression (5.32) leads to

$$|\Sigma_d| \sim \frac{(2m)!}{(m)!2^m m! \prod_{i=1}^{m} d_i!} \exp\{-\lambda - \lambda^2 + o(1)\}. \tag{5.33}$$

For the set $\Sigma_d(i,j)=0$ of $n \times n$ symmetric zero-one tables with column sums $d$, a zero diagonal and structure zeros on the $(i,j)$th entry, the matrix $(D_{ij})_{n \times n}$ has 0 on its diagonal, $D_{ij} = D_{ji} = 0$ and 1 elsewhere. Again, plugging these simplifications into expression (5.32) leads to

$$|\Sigma_d(i,j)=0| \sim \frac{(2m)!}{(m)!2^m m! \prod_{i=1}^{m} d_i!} \exp\{-\lambda - \lambda^2 - \frac{d_i d_j}{2m} + o(1)\}. \tag{5.34}$$

Plugging the results in (5.33) and (5.34) to (5.7), we have

$$\lim_{m \to \infty} \sup_{\bigcup_{n=1}^{m} \Sigma_d} |P_{ij} - \left(1 - e^{-\frac{d_i d_j}{m}}\right)| = 0.$$

Proof of Theorem 5.5.1

First we will formalize the notations that will be used in the proof. For any label $e = (e_1, \ldots, e_n)$, define $O_{K \times K}(e)$ by

$$O_{kl}(e) = \sum_{ij} A_{ij} I\{e_i = k, e_j = l\},$$

and define

$$O_k(e) = \sum_{l} O_{kl}(e).$$

For $k \neq l$, $O_{kl}$ is the number of edges between block $k$ and block $l$, and $O_{kk}$ is the number of edges within block $k$ (we shall often suppress the argument $e$ for brevity).
Define array \( \hat{S}_{K \times K \times M} \), \( \hat{V}_{K \times K \times M} \) and \( \hat{\Pi}_{K \times M} \) as
\[
\hat{S}_{kau}(e) = \frac{1}{n} \sum_{i} I(e_{i} = k, c_{i} = a, \theta_{i} = x_{u}),
\]
\[
\hat{V}_{kau}(e) = \frac{\sum_{i=1}^{n} I(e_{i} = k, c_{i} = a, \theta_{i} = x_{u})}{\sum_{i=1}^{n} I(c_{i} = a, \theta_{i} = x_{u})},
\]
\[
\hat{\Pi}_{au} = \frac{1}{n} \sum_{i=1}^{n} I(c_{i} = a, \theta_{i} = x_{u}).
\]
Roughly speaking, \( \hat{S} \) can be thought of as the empirical joint distribution of \( e, c \) and \( m \theta \) while \( \hat{V} \) can be thought of the as the marginal distribution of \( e \) and \( \hat{\Pi} \) is the empirical joint distribution of \( c \) and \( m \theta \).

Before we proceed to the proof, we first state a lemma.

**Lemma 5.8.1.** We have two matrices \( P_{K \times K} \) and \( S_{K \times K \times N} \) and \( S_{K \times K \times N} \) has nonnegative entries. Define \( K \times K \) matrix \( S^{u} \) as \( S^{u}_{ij} = S^{u}_{ji} \). Assume that
a) \( P \) is symmetric;
b) \( P \) does not have two identical columns;
c) there exist at least one nonzero entry in each column of \( S^{u} \) for \( u = 1, \ldots, N \);
d) for \( 1 \leq a, b, k, l \leq K \) and \( 1 \leq u, v \leq N \), \( P_{kl} = P_{ab} \) whenever \( S_{kau}S_{lbv} > 0 \).

Then \( S^{u} \) are diagonal matrices or permutations of diagonal matrices for \( u = 1, \ldots, N \).

The proof of the lemma can be found in the last part of Section 5.8. Define \( \mu_{n} = n^{2} \rho_{n} \), we will show Theorem 5.5.1 in four steps:

**Step 1:** First we will show that the modularity function in (5.25) can be written in the form of 
\[
F\left(\frac{O(e)}{\mu_{n}}\right),
\]
for some function \( F(\cdot) \).

**Step 2:** Show that \( F\left(\frac{O(e)}{\mu_{n}}\right) \) is uniformly close to its population version.

**Step 3:** Show weak consistency by showing that there exist \( \delta_{n} \rightarrow 0 \), such that
\[
P\left(\max_{e:||V(e)-V(e)||_{1} \geq \delta_{n}} F\left(\frac{O(e)}{\mu_{n}}\right) < F\left(\frac{O(e)}{\mu_{n}}\right)\right) \rightarrow 1, \text{ as } \lambda_{n} \rightarrow \infty.
\]
Here \( ||S||_1 = \sum_{kau} |S_{kau}| \).

**Step 4**: Show that

\[
P \left( \max_{e: e \neq c} F \left( \frac{O(e)}{\mu_n} \right) < F \left( \frac{O(c)}{\mu_n} \right) \right) \to 1,
\]

when \( \frac{\lambda_n}{\log n} \to \infty \), which implies strong consistency.

**Details of Step 1:**

The modularity in (5.25) can also be written as

\[
Q(e) = \sum_k \left( \frac{O_{kk}}{2m} - \frac{O_k^2}{(2m)^2} \right),
\]

since it is true that

\[
\frac{O_k^2}{2m} = \frac{\sum_i (d_i I(e_i = k))^2}{2m} = \sum_{ij} \frac{d_i d_j I(e_i = k, e_j = k)}{2m}.
\]

Moreover, define

\[
F(O) = \sum_k \left( \frac{O_{kk}}{\sum_k O_{kh}} - \left( \frac{\sum_l O_{kl}}{\sum_k O_{kh}} \right)^2 \right).
\]

We have

\[
F \left( \frac{O(e)}{\mu_n} \right) = \sum_k \left( \frac{O_{kk}}{2m} - \frac{O_k^2}{(2m)^2} \right) \quad (5.35)
\]

**Details of Step 2:**
Define $H_{kl}(R) = \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})R_{kau}R_{lbv}$, we have

$$\frac{1}{\mu_n} E(O_{kl} \mid \mathbf{c}, \mathbf{\theta}) = \frac{1}{\mu_n} \sum_{ij} \sum_{abuv} E(A_{ij} I(e_i = k, c_i = a, \theta_i = x_u)I(e_j = l, c_j = b, \theta_j = x_v) \mid \mathbf{c}, \mathbf{\theta})$$

$$= \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\hat{S}_{kau}(\mathbf{e})\hat{S}_{lbv}(\mathbf{e}) = H_{kl}(\hat{S}(\mathbf{e})).$$

Define

$$\hat{T}_{kl}(\mathbf{e}) = \frac{1}{\mu_n} E(O_{kl}(\mathbf{e}) \mid \mathbf{c}, \mathbf{\theta}).$$

We have

$$\hat{T}_{kl}(\mathbf{e}) = \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\hat{S}_{kau}(\mathbf{e})\hat{S}_{lbv}(\mathbf{e})$$

$$= \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\hat{V}_{kau}(\mathbf{e})\hat{\Pi}_{au}\hat{V}_{lbv}(\mathbf{e})\hat{\Pi}_{bv}.$$ 

Replace $\hat{\Pi}$ by $\Pi$, we define

$$T_{kl}(\mathbf{e}) = \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\hat{V}_{kau}(\mathbf{e})\Pi_{au}\hat{V}_{lbv}(\mathbf{e})\Pi_{bv}.$$ 

To show $F\left(\frac{O(\mathbf{e})}{\mu_n}\right)$ is uniformly close to its population version, we show that there exist $\epsilon_n \to 0$, such that

$$P(\max_{\mathbf{e}} \left| F\left(\frac{O(\mathbf{e})}{\mu_n}\right) - F(T(\mathbf{e}))\right| < \epsilon_n) \to 1 \quad \text{as} \quad \lambda_n \to \infty. \quad (5.36)$$

We have

$$\left| F\left(\frac{O(\mathbf{e})}{\mu_n}\right) - F(T(\mathbf{e}))\right| \leq \left| F\left(\frac{O(\mathbf{e})}{\mu_n}\right) - F(\hat{T}(\mathbf{e}))\right| + \left| F(\hat{T}(\mathbf{e})) - F(T(\mathbf{e}))\right|.$$ 

Since $F(.)$ is Lipschitz in all its arguments, we have

$$\left| F(\hat{T}(\mathbf{e})) - F(T(\mathbf{e}))\right| \leq M_1 ||\hat{T}(\mathbf{e}) - T(\mathbf{e})||_\infty. \quad (5.37)$$

123
Here $||X||_{\infty} = \max_{kl} |X|$. Further,

$$|\hat{T}(e) - T(e)| = \left| \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})\hat{V}_{kau}(e)\hat{V}_{lbv}(e)(\hat{\Pi}_{au}\hat{\Pi}_{bv} - \Pi_{au}\Pi_{bv}) \right| \leq \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab})|\hat{\Pi}_{au}\hat{\Pi}_{bv} - \Pi_{au}\Pi_{bv}|$$

Since $\hat{\Pi}_{au} \rightarrow_p \Pi_{au}$ for all $a, u$, we have the lefthand side of (5.37) converges to 0 in probability uniformly over all $e$ as $\lambda_n \rightarrow \infty$. Next, we have

$$\left| F \left( \frac{O(e)}{\mu_n} \right) - F(\hat{T}(e)) \right| \leq M_1 \frac{||O(e) - \mu_n\hat{T}(e)||_{\infty}}{\mu_n^{\frac{2}{2}}} \frac{\omega^2}{\mu_n^{\frac{2}{2}}} \cdot \frac{2}{\omega/3} \quad (5.38)$$

Since the $A_{ij}$ in $O_{kl}(e)$ are independent and $|A_{ij}| \leq 1$, according to Bernstein’s inequality, we have

$$P(|O_{kl}(e) - \mu_n\hat{T}(e)| > \omega|c, \theta) \leq 2 \exp \left( - \frac{\omega^2/2}{\text{var}(O_{kl}|c, \theta) + 2\omega/3} \right) \quad (5.39)$$

Since $\text{var}(O_{kl}|c, \theta) \leq 2n^2 \text{max}_{ij} \text{var}(A_{ij}|c, \theta) \leq 2n^2 \rho_n \text{max}_{uvab} \logit^{-1}(x_u + x_v + Z_{ab})$.

Define $\tau = \text{max}_{uvab} \logit^{-1}(x_u + x_v + Z_{ab})$. Let $\omega = \epsilon^2 \rho_n$, for $\epsilon < 3\tau$, we have

$$P\left( \frac{O_{kl}(e)}{\mu_n} - \hat{T}(e) > \epsilon|c, \theta \right) \leq 2 \exp \left( - \frac{\epsilon^2 \rho_n^2}{8n^2 \rho_n \tau} \right) = 2 \exp \left( - \frac{1}{8\tau} \epsilon^2 \rho_n \right)$$

We have the lefthand side of (5.38) converge to 0 in probability uniformly if $\lambda_n \rightarrow \infty$. Hence, we have shown (5.36).

**Details of Step 3:**

Show that there exist $\delta_n \rightarrow 0$, such that

$$P \left( \max_{e:||\hat{V}(e) - \hat{V}(c)||_1 \geq \delta_n} F \left( \frac{O(e)}{\mu_n} \right) < F \left( \frac{O(c)}{\mu_n} \right) \right) \rightarrow 1, \quad \text{as} \quad \lambda_n \rightarrow \infty.$$
We will first show that $F(H(S))$ is uniquely maximized over \( \{S : S \geq 0, \sum_k S_{kau} = \Pi_{au}\} \) by \( S = D \) where \( D = \Pi_{au}E_{ka} \). Here \( S \) is the limit of \( \hat{S} \) and \( E \) is any row permutation of the \( K \times K \) identity matrix. The matrix \( E \) is for when there is permutation equivalent class. For simplicity, in the following proof, we will assume \( E \) is the identity matrix itself.

If \( Q(e) \) is maximized by the true label \( c \), then \( F(H(S)) \) should be maximized by the true assignment \( S = D \). Since \( \sum_k \hat{S}(e) \to \Pi_{au} \), the limit \( S \) must satisfy that \( \sum_k S_{kau} = \Pi_{au} \). Therefore, we only need to consider maximizer of \( F(H(S)) \) under the constraint \( \{S : S \geq 0, \sum_k S_{kau} = \Pi_{au}\} \).

Define \( H_0 = \sum_{kl} H_{kl} \) and \( H_k = \sum_l H_{kl} \), we have

\[
F(H) = \sum_k \left( \frac{H_{kk}}{H_0} - \frac{H_k^2}{H_0^2} \right).
\]

Define

\[
\triangle_{kl} = \begin{cases} 
1 & \text{for } k = l, \\
-1 & \text{for } k \neq l.
\end{cases}
\]

Use the equally that

\[
\sum_k \left( \frac{H_{kk}}{H_0} - \frac{H_k^2}{H_0^2} \right) + \sum_{k \neq l} \left( \frac{H_{kl}}{H_0} - \frac{H_k H_l}{H_0^2} \right) = 0,
\]
we have

\[
F(H(S)) = \frac{1}{2H_0} \sum_{kl} \Delta_{kl} \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab}) S_{kau} S_{lbv} \\
- \frac{1}{2H_0^2} \sum_{kl} \Delta_{kl} \left[ \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab}) \Pi_{bu} S_{kau} \right] \left[ \sum_{abuv} \logit^{-1}(x_u + x_v + Z_{ab}) \Pi_{au} S_{lbv} \right] \\
= \frac{1}{2H_0} \sum_{kl \ ab} \Delta_{kl} \sum_{uv} \logit^{-1}(x_u + x_v + Z_{ab}) S_{kau} S_{lbv} \\
- \frac{1}{2H_0^2} \sum_{kl \ ab} \Delta_{kl} \sum_{uv} \logit^{-1}(x_u + x_v + Z_{ab}) \logit^{-1}(x_u' + x_v' + Z_{a'b'}) \Pi_{bu} \Pi_{a'u'} S_{kau} S_{lbv'} \\
= \frac{1}{2H_0} \sum_{kl \ ab} \Delta_{kl} S_{kau} S_{lbv'} \left[ \logit^{-1}(x_u + x_v + Z_{ab}) \logit^{-1}(x_u' + x_v' + Z_{a'b'}) \Pi_{bu} \Pi_{a'u'} \right] \\
- \frac{1}{2H_0^2} \sum_{kl \ ab} \Delta_{kl} S_{kau} S_{lbv'} \logit^{-1}(x_u + x_v + Z_{ab}) \logit^{-1}(x_u' + x_v' + Z_{a'b'}) \Pi_{bu} \Pi_{a'u'} \\
\leq \frac{1}{2H_0} \sum_{kl \ ab} \Delta_{ab'} S_{kau} S_{lbv'} \left[ \logit^{-1}(x_u + x_v + Z_{ab}) \logit^{-1}(x_u' + x_v' + Z_{a'b'}) \Pi_{bu} \Pi_{a'u'} \right] \\
- \frac{1}{2H_0^2} \sum_{kl \ ab} \Delta_{ab'} \sum_{uv} \logit^{-1}(x_u + x_v + Z_{ab}) \logit^{-1}(x_u' + x_v' + Z_{a'b'}) \Pi_{bu} \Pi_{a'u'} \Pi_{au} \Pi_{a'u'} \\
= F(H(\mathbb{D}))
\]

The inequality is true because of conditions (5.26) and (5.27). Now we need to show that \( \mathbb{D} \) is the unique maximizer of \( F(H(S)) \). The inequality holds only if \( \Delta_{kl} = \Delta_{ab'} \) when \( S_{kau} S_{lbv'} > 0 \). This follows the results in Lemma 5.8.1, since \( \Delta \) is symmetric and does not have two identical columns.

Since \( F(H(S)) \) is uniquely maximized by \( \mathbb{D} \), by Lipschitz condition, there exist \( \delta_n \rightarrow 0 \) such that

\[
F(T(c)) - F(T(e)) > 2\epsilon_n \quad \text{for} \quad ||\hat{V}(e) - \hat{V}(c)||_1 > \delta_n. \quad (5.40)
\]

126
Thus, with (5.36), we have

\[
P\left(\max_{e: ||\hat{V}(e) - \hat{V}(c)||_1 \geq \delta_n} F\left(\frac{O(e)}{\mu_n}\right) < F\left(\frac{O(c)}{\mu_n}\right)\right) \\
\geq P\left(\max_{e: ||\hat{V}(e) - \hat{V}(c)||_1 \geq \delta_n} F\left(\frac{O(e)}{\mu_n}\right) - \max_{e: ||\hat{V}(e) - \hat{V}(c)||_1 \geq \delta_n} F(T(e)) < \epsilon_n, \right.
\]

This implies that

\[
P(||\hat{\hat{V}}(c) - \hat{V}(c)||_1 \leq \delta_n) \to 1,
\]

where \(\hat{c} = \arg \max_e F\left(\frac{O(e)}{\mu_n}\right)\) is the estimator. Since

\[
\frac{1}{n} |e - c| = \frac{1}{n} \sum_{i=1}^{n} I(c_i \neq \hat{e}_i) = \sum_{au} \Pi_{au}(1 - \hat{V}_{au}(e)) \\
\leq \sum_{au} (1 - \hat{V}_{au}(e)) = \frac{1}{2} \left( \sum_{au} (1 - \hat{V}_{au}(e)) + \sum_{au} \sum_{k \neq a} \hat{V}_{ka}(e) \right) \\
= \frac{1}{2} ||\hat{V}(e) - \Pi||_1 = \frac{1}{2} ||\hat{V}(e) - \hat{V}(e)||_1
\]

We have established the weak consistency of the estimator.

**Details of Step 4:**

To show

\[
P\left(\max_{e: e \neq c} F\left(\frac{O(e)}{\mu_n}\right) < F\left(\frac{O(c)}{\mu_n}\right)\right) \to 1, \text{ as } \lambda_n \to \infty.
\]

We only need to show that there exist \(\delta_n \to 0\), such that

\[
P\left(\max_{e: 0 < ||\hat{V}(e) - \hat{V}(c)||_1 \leq \delta_n} F\left(\frac{O(e)}{\mu_n}\right) < F\left(\frac{O(c)}{\mu_n}\right)\right) \to 1, \text{ as } \lambda_n \to \infty. \quad (5.41)
\]
Since from the results in Step 3, we already have there exist $\delta_n \to 0$, such that

$$P\left(\max_{e:||\hat{V}(e) - V(c)||_1 \geq \delta_n} F\left(\frac{O(e)}{\mu_n}\right) < F\left(\frac{O(c)}{\mu_n}\right)\right) \to 1, \text{ as } \lambda_n \to \infty.$$  

With the Lipschitz properties of $F$, we have

$$F\left(\frac{O(e)}{\mu_n}\right) - F\left(\frac{O(c)}{\mu_n}\right) = F(\hat{T}(e)) - F(\hat{T}(c)) + \Delta(e, c), \quad (5.42)$$

where $|\Delta(e, c)| \leq M\left(\frac{O(e)}{\mu_n} - \hat{T}(e)\right) - \left(\frac{O(c)}{\mu_n} - \hat{T}(c)\right) ||\infty$. By the continuity of $F$ in the neighborhood $||\hat{V}(e) - \hat{V}(c)||_1 \leq \delta_n$, we have

$$F(\hat{T}(e)) - F(\hat{T}(c)) \leq -C||\hat{V}(e) - \hat{V}(c)||_1 + o(||\hat{V}(e) - \hat{V}(c)||_1).$$

Since the derivative of $F$ is continuous with respect to $\hat{V}(e)$ in the neighborhood $||\hat{V}(e) - \hat{V}(c)||_1 \leq \delta_n$, there exist a $\delta^*$ such that

$$F(\hat{T}(e)) - F(\hat{T}(c)) \leq -\frac{C}{2}||\hat{V}(e) - \hat{V}(c)||_1 + o(||\hat{V}(e) - \hat{V}(c)||_1), \quad (5.43)$$

for $||\hat{\Pi} - \Pi||_\infty \leq \delta^*$. With (5.42) and (5.43), to show that

$$P\left(\max_{e:0 < ||\hat{V}(e) - V(c)||_1 \leq \delta_n} F\left(\frac{O(e)}{\mu_n}\right) < F\left(\frac{O(c)}{\mu_n}\right)\right) \to 1, \text{ as } \lambda_n \to \infty,$$

it is sufficient to show

$$P(\max_{\{e \neq c\}} |\Delta(e, c)| \leq C||\hat{V}(e) - \hat{V}(c)||_1/4) \to 1.$$  

Since we have

$$\frac{1}{n}|e - c| \leq \frac{1}{2}||\hat{V}(e) - \hat{V}(c)||_1,$$
for each \( m \geq 1 \), we have

\[
P \left( \max_{|e - c| = m} |\Delta(e, c)| > C \| \hat{V}(e) - \hat{V}(c) \|_1 / 4 \right) \\
\leq P \left( \max_{|e - c| \leq m} \left\| \left( \frac{O(e)}{\mu_n} - \hat{T}(e) \right) - \left( \frac{O(c)}{\mu_n} - \hat{T}(c) \right) \right\|_\infty > \frac{Cm}{2Mn} \right) = r_1
\]

If \( e_1 \neq c_1, \ldots, e_m \neq c_m, e_{m+1} = c_{m+1}, \ldots, e_n = c_n \),

\[
O_{kl}(e) - O_{kl}(c) = \sum_{i=1}^{m} [A_{ii} I(e_i = k, e_i = l) - A_{ii} I(c_i = k, c_i = l)] \\
+ 2 \sum_{i<j}^{m} [A_{ij} I(e_i = k, e_j = l) - A_{ij} I(c_i = k, c_j = l)] \\
+ 2 \sum_{i=1}^{m} \sum_{j=m+1}^{n} [A_{ij} I(e_i = k, e_j = l) - A_{ij} I(c_i = k, c_j = l)].
\]

Therefore, we have

\[
\text{var}(O_{kl}(e) - O_{kl}(c)|c, \theta) \leq \rho_n \tau (m + 2m(m-1) + 4m(n-m)) \\
\leq 4mn\rho_n \tau.
\]

With Bernstein’s inequality, we have for \( \epsilon \geq 6\tau m/n \),

\[
P \left( \left\| \left( \frac{O(e)}{\mu_n} - \hat{T}(e) \right) - \left( \frac{O(c)}{\mu_n} - \hat{T}(c) \right) \right\| \geq \epsilon|c, \theta \right) \leq 2 \exp \left( -\frac{(\epsilon n^2 \rho_n)^2/2}{4mn\rho_n \tau + 2\epsilon n^2 \rho_n / 3} \right) \\
\leq 2 \exp \left( -\frac{3}{8} \epsilon n^2 \rho_n \right).
\]

For \( \epsilon < 6\tau m/n \),

\[
P \left( \left\| \left( \frac{O(e)}{\mu_n} - \hat{T}(e) \right) - \left( \frac{O(c)}{\mu_n} - \hat{T}(c) \right) \right\| \geq \epsilon|c, \theta \right) \leq 2 \exp \left( -\frac{(\epsilon n^2 \rho_n)^2/2}{4mn\rho_n \tau + 2\epsilon n^2 \rho_n / 3} \right) \\
\leq 2 \exp \left( -\frac{n}{16m\tau} \epsilon^2 n^2 \rho_n \right).
\]
Denote $\alpha = C/2M$, if $\alpha \geq 6\tau$, then

$$r_1 \leq 2K^{m+2}n^m \exp\left(-\alpha \frac{3m}{8n}n^2\rho_n\right) = 2K^2[K\exp(n - 3\alpha n\rho_n/8)]^m.$$ 

If $\alpha < 6\tau$, then

$$r_1 \leq 2K^{m+2}n^m \exp\left(-\alpha^2 \frac{m}{16\tau n}n^2\rho_n\right) = 2K^2[K\exp(n - \alpha^2 n\rho_n/16\tau)]^m.$$ 

In both cases, $\lambda_n / \log n \to \infty$. We have

$$P(\text{max}_{e \neq c} |\Delta(e, c)| \leq C||\hat{\Delta}(e) - \hat{\Delta}(c)||_1/4)$$

$$= \sum_{m=1}^{\infty} P(\text{max}_{|e-c|=m} |\Delta(e, c)| > C||\hat{\Delta}(e) - \hat{\Delta}(c)||_1/4) \to 0.$$ 

Thus we have established the strong consistency.

**Proof of Lemma 5.8.1**

The proof is similar to Lemma 3.2 in Bickle and Chen (2009). There are two possibilities for $S$ and we prove the lemma in the following two steps.

1) There exist permutations of the rows and columns of $S^u$ for $u = 1, \ldots, N$, such that the diagonal items are all positive after permutation. In this case, if for any $u_i$, $S^{u_i}$ is not a diagonal matrix. Then there must exist $k \neq a$ such that $S_{kau_i} > 0$. This gives that $S_{kau_1}S_{bbu} > 0$ for $b = 1, \ldots, K$ and $v = 1, \ldots, N$. By d), we have $P_{bb} = P_{ab}$ for $b = 1, \ldots, K$. This contracts with b). Thus $S^u$, $u = 1, \ldots, N$ must be diagonal matrices or permutations of diagonal matrices.

2) If there does not exist such permutations, we can always permute rows and columns of $S^u$, such that for some $m^u$, $S_{iju} = 0$ for $1 \leq i, j \leq m^u$, and $S_{bbu} > 0$ for $b > m^u$. Note that there exist at least one nonzero entry in each of the first $m^u$ columns, i.e., $S_{kiv} > 0$ for $i = 1, \ldots, m^u$ and $k_i \in \{m^u + 1, \ldots, K\}$. So $S_{k_iu}S_{bbu} > 0$ and $S_{k_iu}S_{kj} > 0$ for $i, j = 1, \ldots, m^u$ and $b > m^u$. By d),

130
we have for $i, j = 1, \ldots, m^u$ and $b > m^u$,

$$P_{k_ik_j} = P_{t^2}, \text{ and } P_{k_ib} = P_{ib}. \quad (5.44)$$

With symmetry a), we have for

$$P_{1i} = P_{i1} = P_{k_ik_1} = P_{ik_1} = P_{k_1i}.$$ 

By (5.44), we also have $P_{1b} = P_{k_1b}$ for $b > m^u$. Therefore, the 1-st row and $k_1$-th row of $P$ are identical. This contradicts with b). The conclusion follows.
References


