UNIVERSITY OF CALIFORNIA
Los Angeles

An Analysis of Community Detection Methods in Multi-layer Networks

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Jiaming Guo

2021

ABSTRACT OF THE THESIS

An Analysis of Community Detection Methods
in Multi-layer Networks

by

Jiaming Guo
Master of Science in Statistics
University of California, Los Angeles, 2021
Professor Arash A. Amini, Chair

The community structures commonly exist in real-world networks such as brain networks, social networks, or trade networks. Since the information of a real-world network is often captured by us with different measures of view, such real-world networks often have a multi-layer structure with different layers sharing the same community assignment. In this scenario, being able to find out the community assignment consistently will help us understand the properties and behaviors of the network so that we can exploit these networks more effectively. In this thesis, we adopt multiple methods to solve the community detection task in different scenarios and discuss the pros and cons of them by comparing the results from multiple methods. We also propose and compare some of the rank-estimation methods, which are used for solving the number of different communities in a network.

The thesis of Jiaming Guo is approved.

Oscar Madrid Padilla

Ying Nian Wu

Arash A. Amini, Committee Chair

University of California, Los Angeles

2021

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

Throughout the experiments of my research and the writing of this thesis, I have received a great deal of support and assistance.

First of all, I would like to express my deepest appreciation to my advisor, Professor Arash Amini, for the invaluable knowledge and techniques he taught me in this domain. The research training you gave me was very high-level and your critiques about my experiments were really insightful and helpful. Besides that, your rigorous attitude for research inspired me to formalize my thesis with a high academic standard.

I wish to thank the Department of Statistics and the Department of Economics for giving me financial support during my master's years, especially in the pandemic of COVID-19. Without the support from these departments, academic life and research would have been way more difficult for me.

I am also grateful to my parents, who have always been my safe heaven in this tough journey, especially when I was stressed out or skeptical about myself. To my mother - you have a profound belief in my academic capability and research potential, and you are always considerate about my situation which makes me confident to embrace the incoming challenges. To my father - your unparalleled support throughout my life is stronger than anyone else, and I would never have today's success without your dedication.

I very much appreciate the help from my friends at the Department of Statistics, Kaiwen Jiang, Dehong Xu, Tianyang Zhao, Siwei Ye, and Jialiang Geng. During the pandemic, I would never forget these weekends we spent studying, doing research, watching TV shows, and playing video games. I could not have avoided the mental struggle and loneliness during my research without your accompanying. I also want to express my special appreciation to my buddy Teng Jiang, who has been there for me with warm care and strong support throughout my master's years. You share your happiness and emotions with me and give me the power that I need to carry on.

CHAPTER 1

Introduction

The behaviors of networks are a widely studied topic in statistics during recent years, and there are multiple models proposed to simulate the random networks in the real world, including the Erdos-Renyi model [ER11], stochastic block model [MNS12], latent space model [HRH02], etc. Real-world networks often have multiple properties such as the small-world property and heavy-tailed degree distributions, among which we are interested in the community structure property as shown in [GN02]. The community structure states that the nodes in a real-world network can often be grouped into multiple sets where the connections are dense within the same group and sparser between different groups; furthermore, being able to find out community assignment of the nodes could give us important information about the nature of the network, and could help us to understand and exploit these networks more effectively.

Mathematically, with the complex networks as input, we can solve the community detection problem to find out the community assignment of all the nodes. In this thesis, different methods for the community detection task are tested under the multi-layer network setting generated by the block model. We adopt both spectrum-based methods, and spectrum-free methods (these methods are usually based on an optimization problem which aims to find out a common community factor for all the layers), then compare community detection results for different methods, and try to give some intuition and some theoretical justifications about the reason why some methods work better in certain situations.

The rest part of the thesis is organized as follows. For this chapter, we will first define

the notations of graph theory and the models we use for generating the random graphs, and then introduce the NMI (normalized mutual information) as our criterion for testing the community detection result. In Chapter 2, we focus on the methods that we test in the experiments; and we will present the experiments, results, and conclusions in Chapter 3.

1.1 Background and Notations

1.1.1 Graphs

In graph theory, a graph (or network) with n nodes is formally defined by $G = \langle V, E \rangle$, where $V = \{1, 2, \cdots, n\}$ is the vertex set and E is the edge set. In this thesis, we mainly discuss the simple undirected graph, which contains no duplicate edges between a pair of nodes, and no loops (an edge from some vertex back to itself); furthermore, the edges are unweighted in a simple undirected graph. The edge set of a simple random graph G can be expressed as $E = \{(i, j) \,|\, i, j \in V, i \neq j\}$.

An alternative approach to representing a given graph $G = \langle V, E \rangle$ is to use the matrix-type statistics. For a simple random graph G, the widely-used adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is defined as $A_{ij} = \mathbf{1}\{(i, j) \in E\}$. $\mathbf{A}$ is symmetric, and the diagonal elements of $\mathbf{A}$ are all zero. Based on the adjacency matrix, we can further define the degree of a node i as $d_i = \#\{(i, j) \in E\} = \sum_{j=1}^{n} A_{ij}$, which is used to measure how "connected" node i is with the other nodes in the graph. Then, we can define the degree matrix as $\mathbf{D} = \mathrm{diag}\{d_1, d_2, \cdots, d_n\} \in \mathbb{R}^{n \times n}$.

With the statistics defined above, we can express the graph Laplacian as

$$\mathbf{L} = \mathbf{A} - \mathbf{D},$$

which is a powerful matrix representation of the graph as mentioned in [CG97]. The eigenvalues and eigenvectors of graph Laplacian are informative; in fact, the Laplacian always has an eigenvalue 0 with the corresponding eigenvalue $\mathbf{1}_n$, given the fact that $\mathbf{A}\mathbf{1}_n = (d_1, d_2, \cdots, d_n)^\top = \mathbf{D}\mathbf{1}_n$. If the graph is further divided into two or more separate com-

munities, that is, $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & 0 \\ 0 & \mathbf{A}_2 \end{pmatrix}$ with $\mathbf{A}_1 \in \mathbb{R}^{m \times m}$, then the Laplacian also has a second eigenvector as $(\overbrace{1, \cdots, 1}^{m}, \underbrace{0, \cdots, 0}_{n-m})^\top$. As a result, the eigenvectors of graph Laplacian contain community-related information, which is the intuition of the spectral clustering method (to be introduced in Chapter 2). We can also define the normalized graph Laplacian as $\mathbf{L}^{\text{norm}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. It can be shown that $\mathbf{L}^{\text{norm}}$ and $\mathbf{L}$ have the same eigenvalues when all nodes have a positive degree.

1.1.2 Multi-Layer Stochastic Block Model (MLSBM)

The idea of the stochastic block model (SBM) is first proposed in [HLL83], which is named in reference to an older *non-stochastic block model* widely used in social science. Compared to the famous Erdos-Renyi model [ER11], SBM is more realistic yet still fairly explainable, since all the edges in the model are not as homogeneous as those in the Erdos-Renyi model. In this chapter, we extend the basic SBM to the multi-layer setting with the following modifications.

For a stochastic block graph model with multiple layers (MLSBM), we assume there are L layers in total and there are n same nodes across all layers. We further assume there exist K distinct communities, namely $\{1, 2, \cdots, K\}$; for a given layer l, the community label assignment matrix is given in the one-hot form by $\mathbf{Z}^{(l)} \in \mathbb{R}^{n \times K}$, where the i-th row of $\mathbf{Z}^{(l)}$ is the community label vector for node i at layer l, that is,

$$Z_{ik}^{(l)} = \begin{cases} 1, & \text{if node } i \text{ belongs to community } k \text{ at layer } l; \\ 0, & \text{otherwise.} \end{cases} \tag{1.1}$$

The community label assignment can also be expressed in the vector form as $\mathbf{z} \in \mathbb{R}^n$, where the community label of node i is given by the quantity z_i.

We further denote $\mathbf{B}^{(l)} \in \mathbb{R}^{K \times K}$ as the connectivity matrix at layer l. Here, $B_{pq}^{(l)} = B_{qp}^{(l)} \in [0, 1]$ is the mutual probability that a pair of nodes in community p and q form an edge.

With this model, whether there is an edge between node i and node j at layer l follows a Bernouli distribution, $A_{ij}^{(l)} \sim \text{Bern}(B_{z_i z_j}^{(l)})$. We can also re-express this Multi-Layer Stochastic Block Model (MLSBM) in the matrix form as

$$\mathbb{E}(\mathbf{A}^{(l)}) = \mathbf{Z}^{(l)}\mathbf{B}^{(l)}{\mathbf{Z}^{(l)}}^{\top}$$

for $l = 1, 2, \cdots, L$. The aim of (consensus) community detection problem on MLSBM is formalized as solving the community label assignment matrix $\mathbf{Z}^{(l)}$ for all layers based on the adjacency matrices $\mathbf{A}^{(l)}$.

A simplified version of this model assumes that the same label matrix $\mathbf{Z}$ is shared across all layers. Hence, the MLSBM degenerates to the following form:

$$\mathbb{E}(\mathbf{A}^{(l)}) = \mathbf{Z}\mathbf{B}^{(l)}\mathbf{Z}^{\top} \tag{1.2}$$

for $l = 1, 2, \cdots, L$. The aim is still to solve the *true* community label assignment matrix $\mathbf{Z}$. We will mainly focus on (1.2) in the *Methods* chapter.

1.1.3 Degree-Corrected (Multi-Layer) Stochastic Block Model (DCSBM)

In the MLSBM (1.2), since the nodes within a certain community are exchangeable, they have exactly the same expected degrees which could be very different from the real-world setting. A solution to this issue is to use the Degree-Corrected (Multi-Layer) Stochastic Block Model (DCSBM) as in [KN11], which gives each node a different degree-correction parameter to modify the actual degrees.

Formally, the DCSBM is defined by $A_{ij}^{(l)} \sim \text{Bern}(\theta_i\theta_j B_{z_i z_j}^{(l)})$, where each $\theta_i > 0$ is the degree-correction factor associated with node i showing its individual tendency of forming ties. Alternatively, if we denote $\mathbf{\Theta} = \text{diag}\{\theta_1, \cdots, \theta_n\} \in \mathbb{R}^n$ as the degree-correction matrix, then the DCSBM can be expressed in the matrix form similar to (1.2):

$$\mathbb{E}(\mathbf{A}^{(l)}) = \mathbf{\Theta}\mathbf{Z}\mathbf{B}^{(l)}\mathbf{Z}^{\top}\mathbf{\Theta}. \tag{1.3}$$

Both (1.2) and (1.3) are studied in this thesis in order to compare the performance of different methods under various conditions. Furthermore, one can also introduce a parameter $\rho > 0$ to both of the models in order to control the sparsity for the whole layer, that is, $\mathbb{E}(\mathbf{A}^{(l)}) = \rho\mathbf{\Theta}\mathbf{Z}\mathbf{B}^{(l)}\mathbf{Z}^\top\mathbf{\Theta}$. An estimation to the degree of node i at layer l is further given by

$$d_i = \sum_{j \neq i} \mathbb{E}(A_{ij}) = \sum_{j \neq i} \rho, \theta_i \theta_j B^{(l)}_{z_i z_j} \tag{1.4}$$

which can be controlled as the independent variable in the *Methods* chapter.

1.2 Normalized Mutual Information (NMI)

Having simulated network data generated by the models introduced above, one may use normalized mutual information (NMI) as the criterion to quantify the clustering performance of different methods in the experiments. The concept of mutual information is widely used in the domain of information theory as in [Cov99], but we have made some slight modifications in order to fit our label matching scenario. Given the predicted label as $\mathbf{z}$ and true label as $\mathbf{y}$ in vector form, the NMI is formally defined by

$$\mathrm{NMI}(\mathbf{z}, \mathbf{y}) = \frac{2 \times I(\mathbf{z}; \mathbf{y})}{H(\mathbf{z}) + H(\mathbf{y})}. \tag{1.5}$$

Here, $H(\cdot)$ is the entropy of a distribution defined as $H(\mathbf{z}) = -\sum_{k=1}^{K} \Pr(z = k) \log \Pr(z = k)$, where $\Pr(z = k)$ is the frequency of nodes assigned with community k in a given label $\mathbf{z}$. On the other hand, $I(\cdot;\cdot)$ is the (un-normalized) mutual information, which is given by the difference between the entropy of $\mathbf{z}$ and the conditional entropy of $\mathbf{z}|\mathbf{y}$:

$$\begin{aligned} I(\mathbf{z}; \mathbf{y}) &= H(\mathbf{z}) - H(\mathbf{z}|\mathbf{y}) \\ &= H(\mathbf{z}) + \sum_{k=1}^{K} \Pr(z = k) \sum_{j=1}^{K} \Pr(z = k|y = j) \log \Pr(z = k|y = j). \end{aligned} \tag{1.6}$$

In order to serve as a criterion of clustering effect, NMI is ranged from 0 to 1, where higher NMI leads to a better match between the predicted label assignment and the true label distribution. This property is shown and proved in the following proposition:

Proposition 1. $0 \leq NMI(\mathbf{z};\mathbf{y}) \leq 1$. *The left inequality holds when* $\mathbf{z}$ *and* $\mathbf{y}$ *are independent; the right inequality holds when* $\mathbf{z}$ *and* $\mathbf{y}$ *are the same label assignment.*

Proof. In this proof, we denote $p_z(k) = \Pr(z = k), p_y(j) = \Pr(y = j)$, and $p_{z,y}(k,j) = \Pr(z = k, y = j)$ for brevity. Then, changing the summation order in the definition of mutual information (1.6) gives us

$$\begin{aligned} I(\mathbf{z};\mathbf{y}) &= -\sum_{k=1}^{K} p_z(k)\log p_z(k) + \sum_{k=1}^{K}\sum_{j=1}^{K} p_{z,y}(k,j)\log\frac{p_{z,y}(k,j)}{p_y(j)} \\ &= -\sum_{k=1}^{K} p_z(k)\log p_z(k) + \sum_{k=1}^{K}\sum_{j=1}^{K} p_{z,y}(k,j)\log\frac{p_{z,y}(k,j)}{p_z(k)p_y(j)} + \sum_{k=1}^{K}\sum_{j=1}^{K} p_{z,y}(k,j)\log p_z(k) \\ &= \sum_{k=1}^{K}\sum_{j=1}^{K} p_{z,y}(k,j)\log\frac{p_{z,y}(k,j)}{p_z(k)p_y(j)}. \end{aligned} \tag{1.7}$$

(1.7) shows the symmetric property of mutual information: $I(\mathbf{z};\mathbf{y}) = I(\mathbf{y};\mathbf{z})$, because both sides are simplified to the same form above. With this alternative definition, we can prove both sides of the inequality.

First we prove the right inequality that $\mathrm{NMI}(\mathbf{z},\mathbf{y}) \leq 1$. By definition (1.6) we know $I(\mathbf{z};\mathbf{y}) - H(\mathbf{z}) = -H(\mathbf{z}|\mathbf{y}) \leq 0$ since each of the summation term in $-H(\mathbf{z}|\mathbf{y})$ is negative; as a result, $I(\mathbf{z};\mathbf{y}) \leq H(\mathbf{z})$. Following the same process we have $I(\mathbf{y};\mathbf{z}) \leq H(\mathbf{y})$. Combining this result with the symmetric property, we shall have $2I(\mathbf{z};\mathbf{y}) \leq H(\mathbf{z}) + H(\mathbf{y})$, which leads to the result that $\mathrm{NMI}(\mathbf{z},\mathbf{y}) \leq 1$.

The left inequality is given by Jensen's inequality. For a convex function $f(x) = -\log x$, Jensen's inequality states that $\mathbb{E}(f(X)) \geq f(\mathbb{E}(X))$ for any random variable X. Hence,

$$\begin{aligned} I(\mathbf{z};\mathbf{y}) &= -\sum_{k=1}^{K}\sum_{j=1}^{K} p_{z,y}(k,j)\log\frac{p_z(k)p_y(j)}{p_{z,y}(k,j)} \\ &= \mathbb{E}_{z,y}\left[-\log\frac{p_z(k)p_y(j)}{p_{z,y}(k,j)}\right] \\ &\geq -\log\mathbb{E}_{z,y}\left[\frac{p_z(k)p_y(j)}{p_{z,y}(k,j)}\right] \end{aligned}$$

$$= -\log \sum_{k=1}^{K} \sum_{j=1}^{K} \left[p_{z,y}(k,j) \times \frac{p_z(k) p_y(j)}{p_{z,y}(k,j)} \right]$$
$$= -\log \sum_{k=1}^{K} \sum_{j=1}^{K} p_z(k) p_y(j) = 0.$$

The equality holds when the labels are independent. □

We will use the NMI for measuring the clustering effect in the following chapters. For each method tested in the experimental setting, we compute the corresponding NMI between the predicted label assignment and the true label assignment, and finally plot all the NMIs and compare the clustering effectiveness based on these NMI curves.

CHAPTER 2

Methods

The basic community detection problem, also known as the community recovery task, is formalized with the following setting: for a given a simple random graph G (or equivalently, the corresponding adjacency matrix $\mathbf{A}$) and the number of community K, the task aims to find out an estimation of the community assignment vector $\hat{\mathbf{z}} \in \mathbb{N}_k^n$, where each entry $\hat{z}_i \in \mathbb{N}_k = \{1, 2, \cdots, K\}$ denotes the community that node i belongs to.

In this chapter, we discuss the extended version of this problem, where there are multiple layers in the model sharing the same community labels $\mathbf{z}$. Note that we use the vector form notation of community assignment $\mathbf{z} \in \mathbb{R}^n$ for brevity (previously we the matrix form $\mathbf{Z} \in \mathbb{R}^{n \times k}$) as mentioned in Subsection 1.1.2. This task is often studied under both the stochastic block model setting (1.2) and the degree-corrected SBM (1.3) setting.

Most of the methods adopted in this chapter, such as co-regularized clustering and aggregated spectral kernel-based clustering [PC20], bias-adjusted spectral clustering [LL21], and likelihood-based clustering [LCL20], are specifically designed to address the difficulty under the multi-layer community detection setting. We will list out all the methods and discuss their motivation and possible advantages, and will eventually compare their performances in the next chapter with NMI (1.5) as the criterion.

Besides, we also discuss the scenario *before* community detection can be applied, when the number of community K is unknown. In this case, we mainly use the edge cross-validation framework from [LLZ16] to find out a viable estimation of community number. It can be proved that when the expected node degree satisfies $\mathbb{E}(\sum d_i/n) = O(\log n)$, the method will

give a consistent estimation of the number of communities.

2.1 Basic Method: Spectral Clustering for Community Detection

Spectral clustering has been a widely used method even before the actual network community detection problem comes into sight. Initially analyzed in details by [NJW01], the general spectral clustering method first models the data with a graph where the edge weights are defined by the similarity of corresponding node pairs, then perform the clustering task and output the predicted clusters. On the other hand, the dataset is already organized as the graph format in our setting, which enables us to take advantage of the spectral clustering and view it as the most important baseline in our experiments.

For the rest part of this chapter, we first introduce the method of spectral clustering below and then describe several variants of spectral clustering as well as other optimization-based ("spectrum-free") methods in the following sections.

Our notations for basic spectral clustering are similar as in (1.2), but the method is based on the most simple single-layer condition instead of the multi-layer setting. Here, the population version of the adjacency matrix is defined as $\mathcal{A} = \mathbb{E}(\mathbf{A}) = \mathbf{Z}\mathbf{B}\mathbf{Z}^\top$. $\mathcal{A}$ has exactly K nonzero eigenvalues if all elements in $\mathbf{B}$ are positive and all K different communities are presented in the assignment $\mathbf{Z}$.

When no two columns in the connectivity matrix $\mathbf{B}$ are linearly independent, the population adjacency matrix will also be of rank K, thus having K different eigenvectors. Then, the eigendecomposition of the population adjacency matrix $\mathcal{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ will give us the community assignment as

$$u_i = u_j \iff z_i = z_j$$

for any pair of nodes (i, j), which makes the community detection problem solvable.

Since the eigenvectors of any instance of the adjacency matrix $\mathbf{A}$ will converge to the

eigenvectors of the population adjacency matrix, we can apply this spectral clustering algorithm based on the observed adjacency matrix. The details of the spectral clustering algorithm are shown below in Algorithm 1.

Algorithm 1: Spectral clustering for adjacency matrix $\mathbf{A}$ on K communities

Result: Predicted community assignment $\mathbf{z}$

Step 1 Do a truncated eigendecomposition over $\mathbf{A}$ and find out the eigenvectors corresponding to the top K largest eigenvalues as $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_K) \in \mathbb{R}^{n \times K}$.

Step 2 Apply k-means algorithm to the row of the eigenvector matrix $\mathbf{U}$ to get the K clusters for the predicted community assignment.

We can also use the Laplacian instead of the adjacency matrix in the Algorithm 1 for the Laplacian-based spectral clustering. It is proved in [RCY11] for the convergence result and consistency of the normalized Laplacian-based spectral clustering.

2.2 Spectral Clustering on Mean Adjacency Matrix

In order to extend the basic spectral clustering method to the multi-layer setting, one might directly apply Algorithm 1 on the mean adjacency matrix as a baseline.

Given the assumption that all connectivity matrices $\mathbf{B}^{(l)}$ are the same and all community assignment vectors are identical, when the number of layer L increases, the difference between mean adjacency matrix $\frac{1}{L}\sum_{l=1}^{L}\mathbf{A}^{(l)}$ and the population version $\mathcal{A}$ will be very small due to the Central Limit Theorem, which means the method is expected to work well in this scenario. However, in reality, the assumption of same $\mathbf{B}$ is often not satisfied; furthermore, if $\operatorname{rank}\left(\sum \mathbf{B}^{(l)}\right) < K$, the spectral clustering method on mean adjacency matrix will degrade and will not obtain a consistent result as specified in Section 2.1. In order to address this issue and achieve better consensus on the multi-layer network model, we list different methods below and will compare their outcome in terms of NMI.

2.3 Spectral Clustering on Aggregate Spectral Kernel

Another baseline we consider in this thesis is to use a "late-fusion" fashioned improvement to the Algorithm 1 from the review paper [PC20]. Following the notations in basic spectral clustering, if we define the matrix consisting of the eigenvectors corresponding to the top-K eigenvalues from layer l as $\mathbf{U}^{(l)} \in \mathbb{R}^{n\times K}$, then the spectral kernel matrix aggregating this layer-level information is given by

$$\mathbf{K}_{agg} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{U}^{(l)}\mathbf{U}^{(l)^\top} \in \mathbb{R}^{n\times n}.$$

Applying the spectral clustering method again on aggregate kernel matrix $\mathbf{K}_{agg}$ will provide us with the desired community detection result. We will refer to this method as "SpecK" in the plot legends in the next chapter.

2.4 Clustering on Low-rank Spectral Approximation

Much like the late-fusion method in Section 2.3, we propose another baseline approach that makes use of all the individual $\mathbf{U}^{(l)}$s instead of the sum, keeping more information in this process. Denote $\mathbf{\Lambda}^{(l)}$ as the eigenvalue matrix of layer l, which is a diagonal matrix consisting of the top-K eigenvalues; then, putting the eigenvectors together gives us the kernel matrix

$$\mathbf{K}_{multi} = \left(\mathbf{U}^{(1)}\mathbf{\Lambda}^{(1)} \,|\, \mathbf{U}^{(2)}\mathbf{\Lambda}^{(2)} \,|\, \cdots \,|\, \mathbf{U}^{(L)}\mathbf{\Lambda}^{(L)}\right) \in \mathbb{R}^{n\times KL},$$

which is further processed with K-means algorithm with regard to the n rows to obtain the community assignment. We will refer to this method as "mspec" in the following chapters.

2.5 Bias-adjusted Spectral Clustering

As mentioned in the previous sections, while the spectral clustering method on mean adjacency matrix often gives good result in experiments, it will not perform well in cases when

the sum of connectivity matrices $\left\{\mathbf{B}^{(l)}\right\}$ degrades to a lower-rank matrix or has eigenvalues close to 0. To address this issue, the bias-adjustment spectral clustering algorithm [LL21] is proposed. Instead of using the mean adjacency matrix, the algorithm performs the eigendecomposition on the bias-adjusted sum of squared adjacency matrices

$$\mathbf{S}_0 = \sum_{l=1}^{L} \left[\left(\mathbf{A}^{(l)}\right)^2 - \mathbf{D}^{(l)} \right]. \tag{2.1}$$

The reason for adding the bias-adjusted term $\sum_l \mathbf{D}^{(l)}$ is shown in the following justification.

If we let $\mathbf{\Delta}^{(l)} = \mathbf{A}^{(l)} - \mathcal{A}^{(l)}$ be the difference between the observed adjacency matrix and the population version of the adjacency matrix at layer l, we shall have

$$\begin{aligned} \sum_{l=1}^{L} \left(\mathbf{A}^{(l)}\right)^2 &= \sum_{l=1}^{L} \left(\mathbf{\Delta}^{(l)} + \mathcal{A}^{(l)}\right)^2 \\ &= \mathbf{S} + 2\sum_{l=1}^{L} \mathbf{\Delta}^{(l)}\mathcal{A}^{(l)} + \sum_{l=1}^{L} \left(\mathcal{A}^{(l)}\right)^2, \end{aligned} \tag{2.2}$$

where $\mathbf{S} = \sum_{l=1}^{L} \left(\mathbf{\Delta}^{(l)}\right)^2$. In (2.2), the second term has zero expectation since each individual $\mathbf{\Delta}^{(l)}$ has mean $\mathbf{0}$ and each $\mathcal{A}^{(l)}$ is a constant. Hence, when we use the sum of squared adjacency matrices for estimation, the only error term that adds up over the layers and might cause systematic bias is $\mathbf{S}$. It is further proved in [LL21] that only the diagonal elements S_{ii} have nonzero expectations contributing to the bias, which can be expressed as

$$\begin{aligned} S_{ii} &= \sum_{l=1}^{L}\sum_{j=1}^{n} \left(A_{ij}^{(l)} - \mathcal{A}_{ij}^{(l)}\right)^2 \\ &= \sum_{l=1}^{L}\sum_{j=1}^{n} \left[\left(1 - \mathcal{A}_{ij}^{(l)}\right)^2 \mathbf{1}(A_{ij}^{(l)} = 1) + \left(\mathcal{A}_{ij}^{(l)}\right)^2 \mathbf{1}(A_{ij}^{(l)} = 0) \right]. \end{aligned} \tag{2.3}$$

The first term in (2.3) can be relaxed into $\sum_{l=1}^{L}\sum_{j=1}^{n} \mathbf{1}(A_{ij}^{(l)} = 1) = \sum_{l=1}^{L} d_i^{(l)}$, and the second term can be bounded by $\left(nL \max_{l;i,j} \left(\mathcal{A}_{ij}^{(l)}\right)^2\right)$; both of the approximation formulas work especially well when the network is very sparse on each layer. However, since $\mathbb{E}\left(\sum_{l=1}^{L} d_i^{(l)}\right) = \sum_{l=1}^{L}\sum_{j=1}^{n} \mathcal{A}_{ij}^{(l)} \approx nL \max_{l;i,j} \mathcal{A}_{ij}^{(l)}$, this term is much larger in scale and is

the leading part of bias terms in (2.3). As a result, we can use $\sum_{l=1}^{L} \mathbf{D}^{(l)}$ to approximate the bias term, and the bias-adjusted sum of squared adjacency matrices (2.1) becomes a good approximation of $\sum_{l=1}^{L} \left(\mathcal{A}^{(l)}\right)^2$.

Doing eigendecomposition on $\sum_{l=1}^{L} \left(\mathcal{A}^{(l)}\right)^2$ should give us the same eigenvectors but avoid the potential rank degrading issue that $\sum_l \mathbf{B}^{(l)}$ has. Hence, by using this bias-adjusted sum for spectral clustering, we can have more consistent results than the vanilla version of the spectral clustering algorithm. The process of bias-adjusted spectral clustering is shown in Algorithm 2, which we will refer to as "bias adjusted" in the plot legend afterward.

Algorithm 2: Bias-adjusted spectral clustering on multi-layer network

Result: Predicted community assignment $\mathbf{z}$

Step 1 Compute the bias-adjusted sum $\mathbf{S}_0 = \sum_{l=1}^{L} \left[\left(\mathbf{A}^{(l)}\right)^2 - \mathbf{D}^{(l)}\right]$.

Step 2 Apply Spectral Clustering (Algorithm 1) to matrix $\mathbf{S}_0$ to obtain predicted community assignment $\mathbf{z}$.

2.6 Linked Matrix Factorization

Previously, we mainly focus on the spectrum-based methods, where most methods either make use of the eigenstructure of the adjacency matrices to generate the spectral kernel or directly use the eigenvectors to perform the clustering task for network community detection. These methods work especially well for block models given the good spectral structures of the networks generated from block models but might fall short in performance when the connectivity matrix $\mathbf{B}$ has multiple many zero eigenvalues.

For the next methods, we will discuss more general approaches which formalize community detection as an optimization problem. These methods are usually more time-consuming, but the nature of using more information also tends to make these methods perform better in some difficult scenarios.

The first method we will introduce is the Linked Matrix Factorization (referred to as LMF in the following chapters and plots) from [PC20], which is a slightly modified version from [TLD09] because we force the common factor matrix $\mathbf{P}$ in (2.4) to be strictly orthonormal. LMF is called an "intermediate fusion" method because it attempts to find the community assignment using information from every individual adjacency matrix separately. This method aims to solve the following optimization problem for a multi-layer network:

$$\left[\hat{\mathbf{P}}, \left(\hat{\mathbf{\Lambda}}^{(1)}, \cdots, \hat{\mathbf{\Lambda}}^{(L)}\right)\right] = \arg\min_{\mathbf{P}^\top \mathbf{P} = \mathbf{I}} \sum_{l=1}^{L} \left\|\mathbf{A}^{(l)} - \mathbf{P}\mathbf{\Lambda}^{(l)}\mathbf{P}^\top\right\|_F^2, \tag{2.4}$$

where the $\mathbf{P} \in \mathbb{R}^{n \times K}$ is the consensus (or common factor matrix) across layers and the $\mathbf{\Lambda}^{(l)} \in \mathbb{R}^{K \times K}$ is a symmetric matrix for capturing each layer's specific features or characteristics.

After the optimization process is finished, the K-means algorithm is applied to the rows of common factor $\hat{\mathbf{P}}$ to obtain the predicted K communities.

2.7 Co-regularized clustering

The second intermediate fusion method we will talk about is the co-regularized based clustering from [PC20] and [KRD11]. This method solves the following optimization problem

$$\begin{aligned}\left[\hat{\mathbf{U}}^{(1)}, \cdots, \hat{\mathbf{U}}^{(L)}, \hat{\mathbf{U}}^*\right] &= \arg\min \sum_{l=1}^{L} \{\mathrm{tr}(\mathbf{U}^{(l)\top}\mathbf{A}^{(l)}\mathbf{U}^{(l)}) + \gamma_l \ \mathrm{tr}(\mathbf{U}^{*\top}\mathbf{U}^{(l)}\mathbf{U}^{(l)\top}\mathbf{U}^*)\}, \\ &\text{subject to } \hat{\mathbf{U}^{(l)}}^\top \hat{\mathbf{U}}^{(l)} = \mathbf{I}, \forall l; \hat{\mathbf{U}^*}^\top \hat{\mathbf{U}}^* = \mathbf{I}\end{aligned} \tag{2.5}$$

and does a K-means clustering on the matrix $\hat{\mathbf{U}}^*$ to yield the predicted community label. We will refer to the co-regularized clustering as "co-Reg" in the following plots and sections.

2.8 Likelihood-based community detection

For an observation $\mathbf{A}$ of the multi-layered network data coming from the SBM, the expectation of adjacency matrix on each layer is a block-wise constant matrix. Based on this

fact, we can solve the following least-square optimization problem with regard to community assignment vector $\mathbf{z}$ and connectivity matrix $\mathbf{B}^{(l)}$:

$$(\hat{\mathbf{z}}, \hat{\mathbf{B}}) = \arg\min_{\mathbf{z},\mathbf{B}} \sum_{l=1}^{L} \sum_{i \neq j} \left(A_{ij}^{(l)} - B_{z_i z_j}^{(l)} \right)^2, \tag{2.6}$$

which is a more straightforward method than spectrum-based methods mentioned before. The optimization problem (2.6) has no direct solution, and one way to solve it is to use an EM-like algorithm [LCL20], which we call "likelihood-based community detection". First, we can solve the community assignment $\mathbf{z}$ based on a group of fixed connectivity matrices $\mathbf{B}$, and then we find out the optimal connectivity matrix $\mathbf{B}$ based on the community assignment found in the previous step. The details of the algorithm are discussed below.

The community assignment $\mathbf{z}$ is usually initialized with the K-means algorithm, which is proved to give consistent results in [ADK08]. However, for most experiments in this thesis, the proposed K-means initialization tends to give an initialization with very low NMI; at the same time, changing the initialization in likelihood-based algorithm to random guessing still yields the same community detection power.

With the initialized guess to the community assignment $\mathbf{z}$, we can first estimate the connectivity matrix. Denote $n_p^{(l)}(\mathbf{z}) = \sum 1(z_i = p)$ as the count of node at layer l within community p, and $N_{pq}^{(l)}(\mathbf{z}) = \sum A_{ij}^{(l)} \mathbf{1}(z_i = p, z_j = q)$ as the block sum of the community p and q. Based on these notations above, for a given community assignment vector $\mathbf{z}$, the lease square question (2.6) has a closed form solution to the connectivity matrix $\mathbf{B}$ as

$$B_{pq}^{(l),new} = \begin{cases} \dfrac{N_{pq}^{(l)}(\mathbf{z})}{n_p^{(l)}(\mathbf{z}) n_q^{(l)}(\mathbf{z})}, & p \neq q; \\ \dfrac{N_{pq}^{(l)}(\mathbf{z})}{n_p^{(l)}(\mathbf{z})(n_q^{(l)}(\mathbf{z}) - 1)}, & p = q. \end{cases} \tag{2.7}$$

Afterwards, with the estimated matrix $\mathbf{B}^{new}$, we can update the community assignment by simply going over all K possible value for each node i and find out the community

assignment that minimizes the following L_2 distance:

$$z_i^{new} = \arg\min_{k \in \mathcal{N}_k} \sum_{l=1}^{L} \sum_{i \neq j} \left(A_{ij}^{(l)} - B_{z_i z_j}^{(l),new} \right)^2, \quad i = 1, 2, \cdots, n. \tag{2.8}$$

Note that in the right-hand side of (2.8), when solving for the new community assignment, we actually use the estimation of $\mathbf{z}$ from the previous round of iteration.

Once we obtain the estimation of $\mathbf{B}^{new}$ and $\mathbf{z}^{new}$, we should compute the least square error in (2.6) again. If the loss reduces, we update the estimation: $\mathbf{B} \leftarrow \mathbf{B}^{new}, \mathbf{z} \leftarrow \mathbf{z}^{new}$. This process is then repeated until convergence. We shall also extend the method to the scenario when the underlying model is DCSBM instead of MLSBM. It can be proved that, under DCSBM, the estimation process of $\mathbf{B}$ (2.7) and $\mathbf{z}$ (2.8) remains the same; furthermore, the estimation to $\mathbf{\Theta} = \text{diag}\{\theta_1, \cdots, \theta_n\}$ is simply computed by

$$\theta_i = \frac{n_{z_i}^{(l)}(\mathbf{z}) \times d_i}{\sum_{j:z_j = z_i} d_i}. \tag{2.9}$$

The process of likelihood-based community detection is shown in Algorithm 3 below.

Algorithm 3: Likelihood-based community detection on multi-layer network

Result: Predicted community assignment $\mathbf{z}$

Initialization Compute the initial community assignment $\mathbf{z}_0$ by K-means.

while *not converged* **do**

Obtain the estimation of connectivity matrices $\mathbf{B}^{new}$ by (2.7);

Compute the estimated $\mathbf{z}^{new}$ by (2.8);

if *the model is DCSBM* **then**

Estimate $\mathbf{\Theta}^{new}$ according to (2.9);

else

$\mathbf{\Theta}^{new} = \mathbf{I}_n$.

end

Compute the loss function based on the estimated $\mathbf{B}^{new}$, $\mathbf{z}^{new}$ and $\mathbf{\Theta}^{new}$.

Update the estimations if the loss function improves.

end

2.9 Rank Estimation Methods

In previous methods, we try to solve the community detection problem with various methods. A drawback of the community detection algorithms is that the number of the communities must be given in advance. For simulated data, this is a minor problem; however, when processing the real-world datasets, we typically do not have access to the ground truth, not even the community number. In this scenario, rank estimation methods are required to determine the number of communities.

Inspired by the cross-validation idea which is widely used in model selection tasks in statistics and computer science, one can use a edge cross-validation (ECV) method [LLZ16] to perform the rank estimation task. The general ECV for rank selection is given as follows:

Algorithm 4: general ECV framework for rank estimation with loss function $\mathcal{L}$

input : the adjacency matrix $\mathbf{A}$, the training proportion p, rank range $[K_{min}, K_{max}]$ and the number of replications T.

output: The best rank selected by the ECV

Step 1 for $t = 1, \cdots, T$ **do**

Randomly choose a subset of node pairs $\Omega \in \mathcal{V} \times \mathcal{V}$, by selecting each pair independently with probability p;

for $k = K_{min}, \cdots, K_{max}$ **do**

Fit a DCSBM on the sampled subset of the graph with rank k to get parameters $\mathbf{z}$, $\mathbf{B}$ and $\mathbf{\Theta}$;

Apply the model to estimate the population adjacency matrix $\mathcal{A}$ on the held-out set $\{A_{ij} \,|\, (i,j) \in \Omega^c\}$;

Evaluate loss $\mathcal{L}_k^{(t)}$ by computing the loss function $\mathcal{L}$ between the input adjacency matrix $\mathbf{A}$ and the estimated population adjacency matrix $\mathcal{A}$.

end

end

Step 2 Let $L_k = \sum_{t=1}^{T} \mathcal{L}_k^{(t)}/T$ and return $\hat{K} = \arg\min_k L_k$ as the rank.

CHAPTER 3

Experiments and Discussion

In this chapter, we test the methods in different experimental settings and try to give an explanation about the reason why some methods perform better in specific settings.

Recall both of the block models in (1.2) and (1.3) where the general model structure is $\mathbb{E}(\mathbf{A}^{(l)}) = \mathbf{\Theta Z B}^{(l)}\mathbf{Z}^\top\mathbf{\Theta}$. There are two types of experiments exploring the community detection problem by conditioning on the connectivity matrix $\mathbf{B}^{(l)}$. In the first experiment inherited from [LL21], two pre-defined different connectivity matrices are used to address the necessity of bias-adjusted spectral clustering; while in the second experiment, the connectivity matrices for each layer are randomly generated with a fixed diagonal/off-diagonal ratio. Both of the experiments are designed such that clustering on average adjacency matrix should not give a satisfactory result, which would further address the importance of other "late-fusion" methods or optimization-based methods.

We also test the behavior of the rank-estimation method, specifically, the Edge Cross-Validation (ECV) algorithm, on another simulated dataset, and discuss the pros and cons of the algorithm based on the results.

3.1 Two connectivity matrix experiment

3.1.1 Experimental Setting and Brief Analysis

In this section we try experimental settings based on Jing Lei's paper [REF]. The number of communities $K = 2$, and the connectivity matrix $\mathbf{B}^{(l)}$ for each layer is selected from the

following options with equal probability:

$$\mathbf{B}_1 = \begin{pmatrix} 3/4 & \sqrt{3}/8 \\ \sqrt{3}/8 & 1/4 \end{pmatrix}, \mathbf{B}_2 = \begin{pmatrix} 7/8 & 3\sqrt{3}/8 \\ 3\sqrt{3}/8 & 1/8 \end{pmatrix}. \tag{3.1}$$

Both $(\mathbf{B}_1+\mathbf{B}_2)$ and $(\mathbf{B}_1^2+\mathbf{B}_2^2)$ have an eigenvalue very close to 0 which will make the spectral clustering process on mean adjacency matrix and on mean of squared adjacency matrix very difficult. However, the eigenstructures of both individual connectivity matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ are good enough for spectrum-based methods to detect the community on the layer-level, which in theory emphasizes the importance of the bias-adjustment spectral clustering.

As for the experimental settings, we set number of nodes $n = 600$, number of layers $L = 10$, and MLSBM (1.2) for simulation. It is worth noticing that we adopt the modified model with the parameter ρ for sparsity control, i.e. $\mathbb{E}(\mathbf{A}^{(l)}) = \rho\mathbf{Z}\mathbf{B}^{(l)}\mathbf{Z}^\top$ as specified in (1.4). The range of ρ is $[0.02, 0.4]$. For each ρ, the community detection is performed with different methods; then we compare the NMI and summarize them in the following plot.

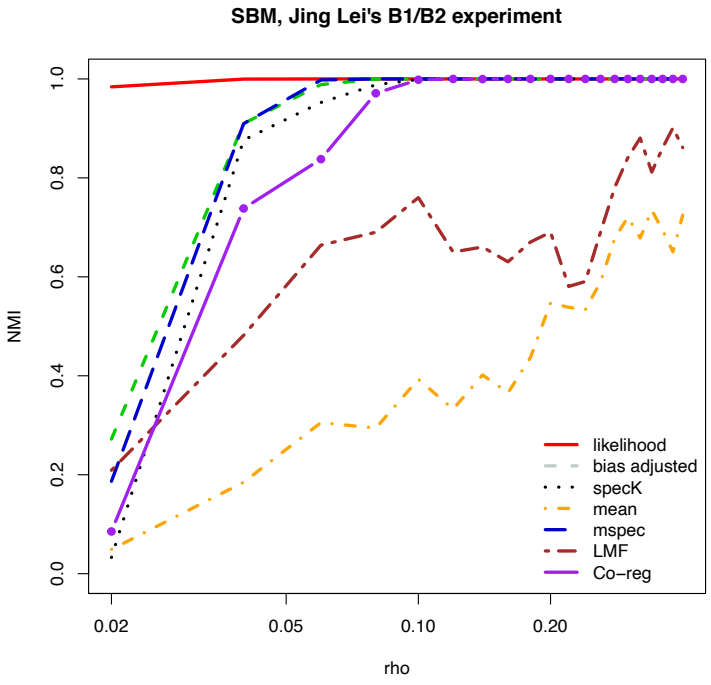


Figure 3.1: Comparison of different methods under Jing Lei's experiment

Results. From Figure 3.1 we can see most of the methods compared outperform the spectral clustering method on the mean adjacency matrix. Besides, the likelihood-based method consistently performed perfectly in this case.

3.1.2 Changing the model to DCSBM

In Figure 3.2, we only change the model to DCSBM and keep the other parameters unchanged. We need to modify the methods slightly, using the Laplacian as the input, in order to make them compatible with the DCSBM setting. Here, to generate the degree-correction parameter $\mathbf{\Theta}$, we use a Pareto distribution with parameters $x_m = 2/3$ and $\alpha = 3$. This distribution has an expectation of 1 and keeps all the corrected degrees positive.

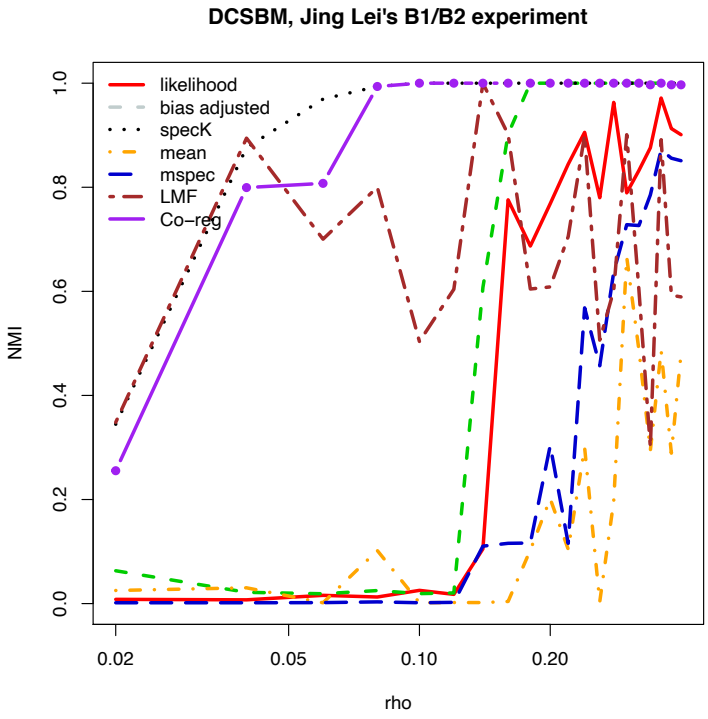


Figure 3.2: Comparison of different methods under Jing Lei's setting and DCSBM

Results. This time, the likelihood-based method falls behind in terms of NMI. It is worth noticing that, under the DCSBM setting and in the very sparse regime, the estimation of the degree-correction parameter $\mathbf{\Theta}$ could have a much larger variance. Figure 3.3 shows the results from numerical experiments computing the distance $\sum_{i=1}^{n} |\hat{\theta}_i - \theta_i|/\theta_i$, where the $\hat{\theta}_i$ is the estimated degree correction parameters from likelihood-based model. The error could be as high as 35% as $\rho = 0.02$. The variance in the estimation of $\mathbf{\Theta}$ contributes to the underwhelming performance of the likelihood-based method in Figure 3.2 especially in the very sparse regime, while the other two methods compared (bias adjustment clustering and aggregate spectral kernel) are less affected by the degree correction parameter $\mathbf{\Theta}$.

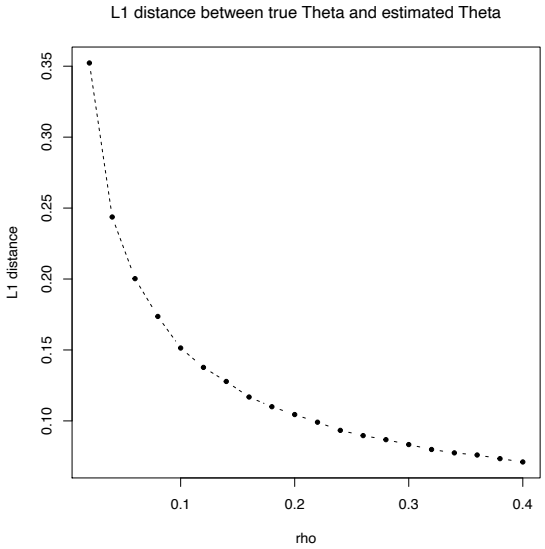


Figure 3.3: The L_1 distance between true Θ and the Θ estimated by likelihood-based method

3.2 Random-generated connectivity matrix experiment

3.2.1 Experimental Setting and Brief Analysis

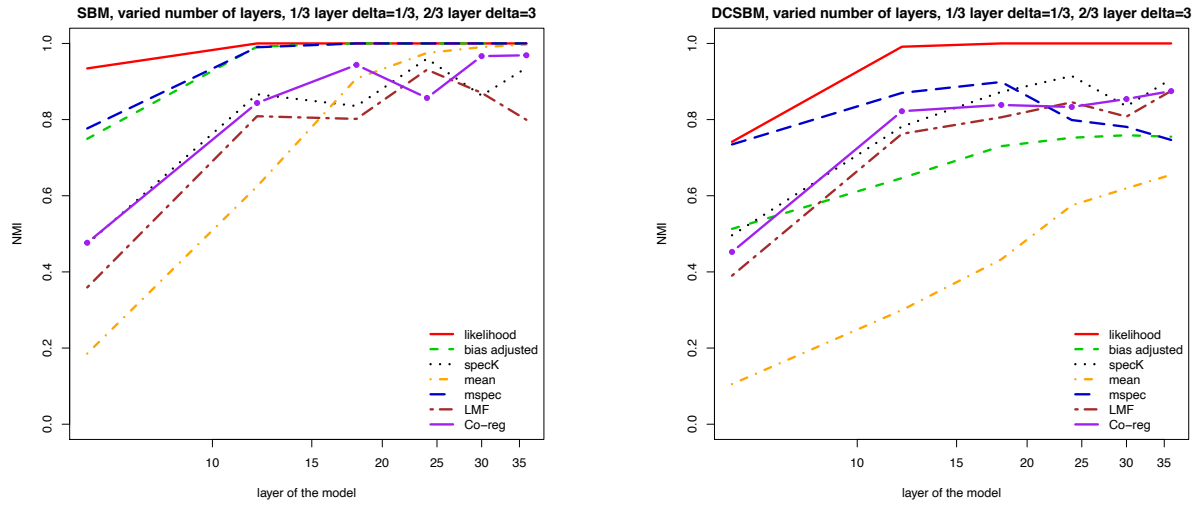
In this section, we try experimental settings based on S. Paul's paper [REF]. We set the number of nodes $n = 600$, the number of communities $K = 4$, and the average expected

degree $d = 12$. We change the number of layers to see how the addition of more layers affects different methods' performance in the community detection task.

Here, the connectivity matrix $\mathbf{B}^{(l)}$ for each layer is generated randomly. The off-diagonal elements $B_{ij}^{(l)}$ (they are the probability of forming edges between two nodes in the different communities) are sampled from a uniform distribution $U[a, b]$; while the diagonal elements (the probability of forming edges between two nodes in the same community) are sampled from another uniform distribution $U[\delta a, \delta b]$. Specifically, in this section, the factor δ is set to be 3 for 2/3 of the layers, which means the in-community connection propensity is much stronger than the cross-community ones. For the rest of the layers, δ is set to be 1/3 to balance out the influence of the previous layers with $\delta = 3$. As for the choice of a and b, we generally let $a = 0.1$ and $b = 0.2$. The $\mathbf{B}$s are normalized after the sampling process.

The independent variable in this experimental setting is the number of layers in the whole graph. Both the models, i.e., MLSBM and the DCSBM, are tested in the experiment. For DCSBM, the degree-correction parameter $\mathbf{\Theta}$ is generated in the same way as Section 3.1, the Pareto Distribution with parameters $x_m = 2/3$ and $\alpha = 3$.

Results. The results are shown in Figure 3.4. It is expected that most of the models can perform the community detection task well since the signal is very strong in each individual layer. However, averaging over the adjacency matrices will make the model lose information about the original multi-layer data, thus the average adjacency matrix-based clustering is also expected to perform the worst out of all the methods. Among all methods, the likelihood-based method and the clustering method based on low-rank spectral approximation perform the best on both DCSBM and MLSBM. Bias-adjusted clustering performs excellently on MLSBM which is expected but falls short when the degree correction parameters are introduced.



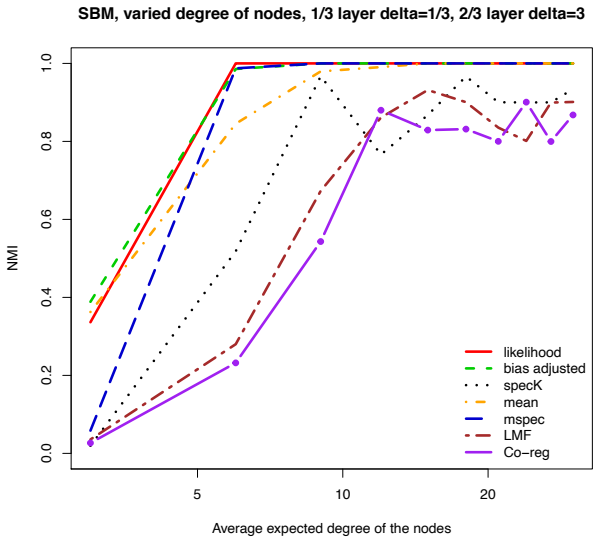
(a) Multi-layer Stochastic Block Model (b) Multi-layer Degree-corrected Stochastic Block Model

Figure 3.4: Comparison of different methods under S Paul's setting

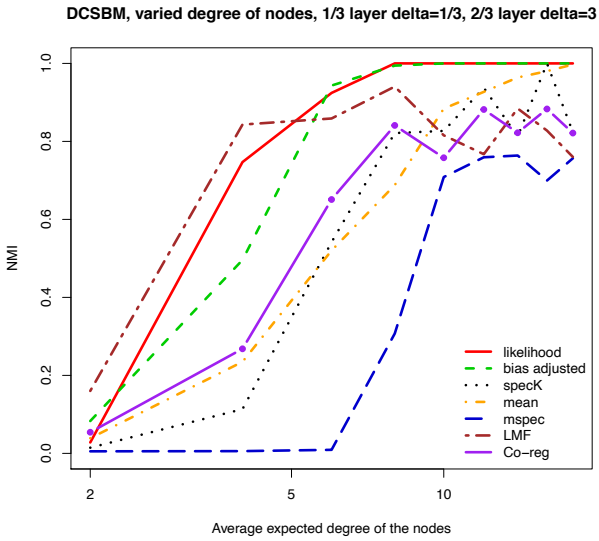
3.2.2 Modification 1: varying the expected degree

In this section, we propose a modification to the original experiment Subection 3.2.1, only changing the independent variables to the average expected degree, instead of the number of layers, and keeping all other settings identical. The average expected degree is given by $d^{(l)} = \mathbb{E}\left(\sum_{i=1}^{n} d_i^{(l)}\right)/n$, which has similar effect as the parameter ρ in Section 3.1. By conducting this experiment, we hope to see how the sparsity within each layer affects different methods' community detection results. The simulated results are summarized in Figure 3.5.

Results. As we can see from the plots, in the more sparse regime, the spectrum-based methods tend to perform better, while most optimization-based methods are more sensitive to the possible disturbance from the sparse regime. It remains as the case that in DCSBM settings; spectral clustering on average adjacency matrix will yield the overall worst performance, while the likelihood-based detection remains the most consistent method. Besides, the low-rank estimation methods, including the clustering on low-rank spectral approximation and clustering on the aggregate spectral kernel, generally give more consistent results



(a) Multi-layer Stochastic Block Model



(b) Multi-layer Degree-corrected Stochastic Block Model

Figure 3.5: Comparison of different methods with varying sparsity

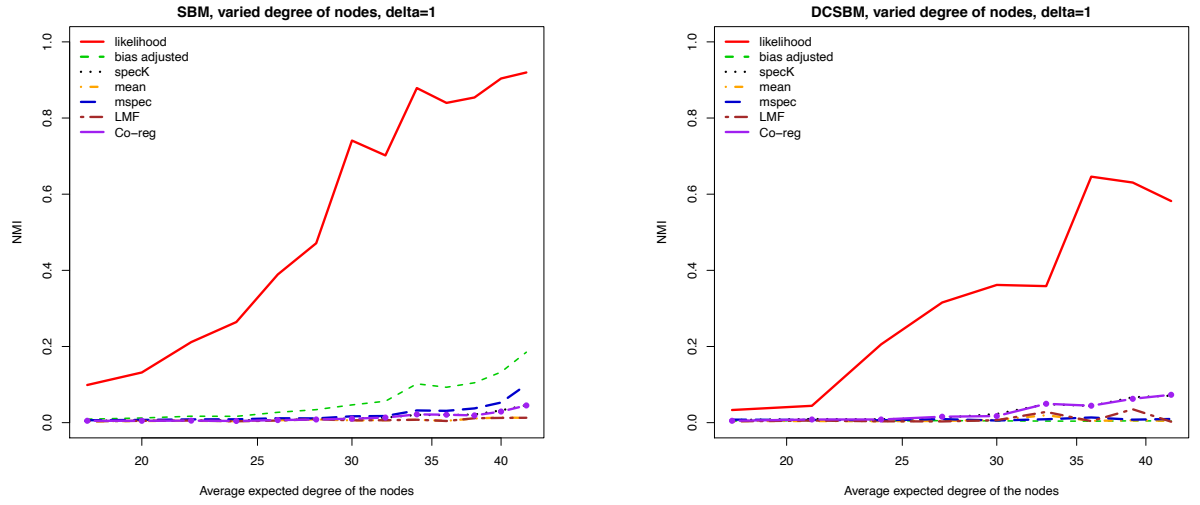
than the other methods.

3.2.3 Modification 2: setting $\delta = 1$ for all layers

A more difficult yet interesting modification to the previous problem setting in Subsection 3.2.2 is to further force $\delta = 1$ for all layers. In this case, the connectivity matrices $\mathbf{B}$ are highly unstable, resulting in most of the methods being less effective since $\mathbb{E}(\mathbf{B}^{(l)})$ is a constant matrix with $K \times K$ identical entries, or alternatively, degrades to lower rank. This problem requires a high capability of keeping all information from the original multi-layer network, which gives the optimization-based methods better potential.

The other hyper-parameters remain the same for this experiment as in Subsection 3.2.2, with $n = 600$, $d = 12$ and tuning the number of layers. The results for this much harder problem are shown in Figure 3.6.

Results. One could see from the plot that, only the likelihood-based method can produce satisfactory results for both MLSBM and DCSBM settings.



(a) Multi-layer Stochastic Block Model (b) Multi-layer Degree-corrected Stochastic Block Model

Figure 3.6: Comparison of different methods with $\delta = 1$

3.3 Rank Estimation Experiment

In this section, we provide another perspective of analyzing the network data by performing rank estimation on the FAO (Food and the Agricultural Organization of the United Nations) worldwide food import/export network dataset from [DNA15]. This multilayer network has $n_0 = 214$ nodes and $L_0 = 364$ layers. The nodes in this network represent countries and layers represent the import/export relationships of a specific food product.

We preprocess this multi-layer network dataset by selecting the top 30 layers with the most edges, computing the cumulative degree for each node across these 20 layers, and filtering out the nodes with a cumulative degree less than 20. After preprocessing, the final dataset contains $n = 177$ nodes and $L = 30$ layers.

Our experiments based on this multi-layer network is explained in the following section

and can be split into two parts, the first one of which is based on simulated data and uses the edge cross-validation framework adopted from [LLZ16], while the second one directly uses the pre-processed dataset without simulation, and the BIC (Bayesian information criterion) as the standard.

3.3.1 FAO Data Simulation and Edge CV

The main drawbacks to the FAO network are that we still have no access to the number of communities K (ground truth) and that different layers are likely to have completely different community structures due to the variability nature of the real-world dataset. Consequently, directly testing the ECV framework (Algorithm 4) with this dataset is not plausible.

Here, we design an alternative method to address this issue. We first set a pre-defined number of communities K, and then perform the spectral clustering (algorithm 1) on the processed FAO dataset to obtain the estimated parameters of the DCSBM, namely, $\mathbf{z}$, $\mathbf{B}$ and $\mathbf{\Theta}$. After that, we sample a network based on the estimated DCSBM parameters as well as the pre-defined K, and then check if the ECV framework works well and results in the correct pre-defined K on the sampled network. The simulation process is shown in the following Algorithm 5:

Specifically, we do $T = 20$ repetitions, and the range of the number of communities is selected to be $[3, 4]$. After obtaining the simulated data, we put the graphs into the ECV framework and select the best estimated rank (K) by L_2 distance as the loss function $\mathcal{L}$ specified in Algorithm 4. The holdout values for edges are set to be $[0.01, 0.09, 0.25, 0.49, 0.81]$, and the maximum rank used in ECV is set as 30. Finally, for both $K = 3$ and $K = 4$ scenarios, we compute the proportion of repetitions where ECV estimates the rank K correctly, and plot the results in Figure 3.7.

Algorithm 5: Data simulation for FAO dataset

input : the FAO network $\{\mathbf{A}^{(l)} \,|\, l = 1, 2, \cdots, 30\}$, the number of repetitions T, the range of the number of communities to test $[K_{min}, K_{max}]$.

output: The simulated data $\hat{\mathbf{A}}^{(l)}_{t,k}$.

for $t = 1, \cdots, T$ **do**

  for $k = K_{min}, \cdots, K_{max}$ **do**

    for $l = 1, \cdots, L = 30$ **do**

      Perform the spectral clustering on $\mathbf{A}^{(l)}$ with k communities to obtain the community assignment $\hat{\mathbf{z}}_l$;

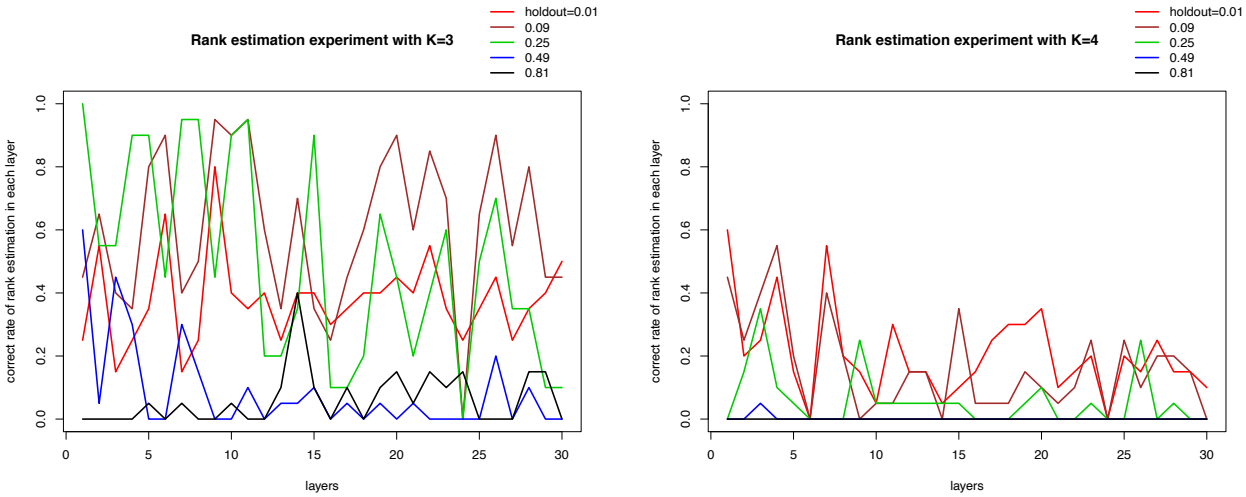
      Estimate the corresponding degree-correction parameter $\hat{\mathbf{\Theta}}_l$ and the connectivity matrix $\hat{\mathbf{B}}_l$ based on (2.9) and (2.7), respectively;

      Sample a new graph from the DCSBM(1.3) as $\hat{\mathbf{A}}^{(l)}_{t,k}$.

    end

  end

end



(a) $K = 3$ (b) $K = 4$

Figure 3.7: Comparison of rank estimation result with $K = 3$ and $K = 4$.

Results. One can see the results are not very good especially when $K = 4$. For $K = 3$, the results are not consistent for most of the layers.

3.3.2 BIC experiment directly on FAO dataset

Another way to do the rank selection and model selection is to use Bayesian information criterion (BIC) as a criterion. BIC is a widely-used criterion for model selection, balancing the value of likelihood and the model complexity. For the single-layer DCSBM scenario, BIC is first defined by [WB17], and is implemented in [ZA20]. The formula of BIC is given as log-likelihood minus $K(K+1)\log(n)/2$:

$$\text{BIC} = \sum_{i=1}^{n} \log \Pr(z = z_i) + \sum_{i<j} \phi(A_{ij}; \theta_i \theta_j B_{z_i z_j}) - K(K+1)\log(n)/2.$$

In this section, we take the processed FAO dataset as the input directly. For each layer l and each $K \in [K_{min}, K_{max}]$ tested, we perform spectral clustering based on the K we used and the sampled sub-graph to obtain the predicted community assignment $\mathbf{z}$, and then,

compute the BIC score based on K and $\mathbf{z}$. The resulting BIC values are averaged across layers to show which K gives the best fit of DCSBM. The resulting BIC plot is shown below.

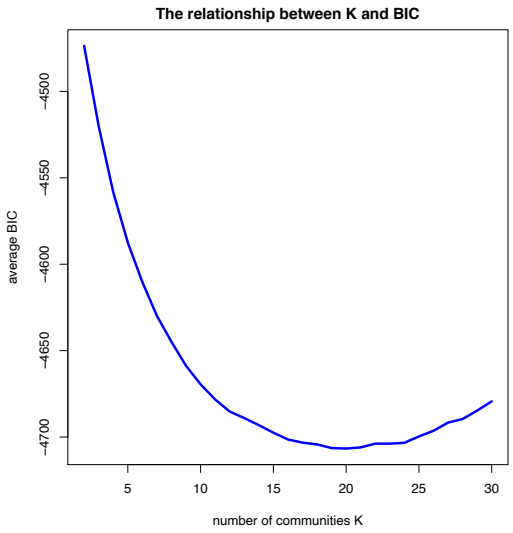


Figure 3.8: The BIC curve

Results. The model with the lowest BIC is presented with $K \approx 20$.

3.4 Discussion and Conclusion

Among the community detection methods, the likelihood-based methods give consistent results in most of the scenarios, and it performs especially well for the most difficult experiment where $\delta = 1$ (Section 3.2.3). However, this method falls short when the disturbing effect of degree correction is too large and the network, in general, is sparse, and this method is slow in computation speed. Other than that, the bias-adjusted spectral clustering outperforms the other methods in most MLSBM settings, but its performance is underwhelming in DCSBM settings. The other spectrum-based methods, including spectral clustering in aggregate kernel and clustering on low-rank spectral approximation, are faster and more consistent, while those optimization-based methods are slower and sometimes perform low

NMI results even when the network is dense enough.

For the rank estimation methods, ECV performs inconsistently at computing the correct community number especially when the number of communities is larger. The BIC is more consistent but gives us a much larger number of communities ($K \approx 20$) which is unexpected in this scenario. Both of the results occur probably due to the nature of real-world data not following the DCSBM setting.

REFERENCES

[ADK08] Aris Anagnostopoulos, Anirban Dasgupta, and Ravi Kumar. "Approximation algorithms for co-clustering." In *Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 201–210, 2008.

[CG97] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[Cov99] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[DNA15] Manlio De Domenico, Vincenzo Nicosia, Alexandre Arenas, and Vito Latora. "Structural reducibility of multilayer networks." *Nature communications*, **6**(1):1–9, 2015.

[ER11] Paul Erdös and Alfréd Rényi. "On the evolution of random graphs." In *The structure and dynamics of networks*, pp. 38–82. Princeton University Press, 2011.

[GN02] M. Girvan and M. E. J. Newman. "Community structure in social and biological networks." *Proceedings of the National Academy of Sciences*, **99**(12):7821–7826, Jun 2002.

[HLL83] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. "Stochastic blockmodels: First steps." *Social networks*, **5**(2):109–137, 1983.

[HRH02] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. "Latent space approaches to social network analysis." *Journal of the american Statistical association*, **97**(460):1090–1098, 2002.

[KN11] Brian Karrer and M. E. J. Newman. "Stochastic blockmodels and community structure in networks." *Physical Review E*, **83**(1), Jan 2011.

[KRD11] Abhishek Kumar, Piyush Rai, and Hal Daume. "Co-regularized multi-view spectral clustering." *Advances in neural information processing systems*, **24**:1413–1421, 2011.

[LCL20] Jing Lei, Kehui Chen, and Brian Lynch. "Consistent community detection in multi-layer network data." *Biometrika*, **107**(1):61–73, 2020.

[LL21] Jing Lei and Kevin Z. Lin. "Bias-adjusted spectral clustering in multi-layer stochastic block models.", 2021.

[LLZ16] Tianxi Li, Elizaveta Levina, and Ji Zhu. "Network cross-validation by edge sampling." *arXiv preprint arXiv:1612.04717*, 2016.

[MNS12] Elchanan Mossel, Joe Neeman, and Allan Sly. "Stochastic Block Models and Reconstruction.", 2012.

[NJW01] Andrew Ng, Michael Jordan, and Yair Weiss. "On spectral clustering: Analysis and an algorithm." *Advances in neural information processing systems*, **14**:849–856, 2001.

[PC20] Subhadeep Paul, Yuguo Chen, et al. "Spectral and matrix factorization methods for consistent community detection in multi-layer networks." *The Annals of Statistics*, **48**(1):230–250, 2020.

[RCY11] Karl Rohe, Sourav Chatterjee, and Bin Yu. "Spectral clustering and the high-dimensional stochastic blockmodel." *The Annals of Statistics*, **39**(4), Aug 2011.

[TLD09] Wei Tang, Zhengdong Lu, and Inderjit S Dhillon. "Clustering with multiple graphs." In *2009 Ninth IEEE International Conference on Data Mining*, pp. 1016–1021. IEEE, 2009.

[WB17] Y. X. Rachel Wang and Peter J. Bickel. "Likelihood-based model selection for stochastic block models." *The Annals of Statistics*, **45**(2):500 – 528, 2017.

[ZA20] Linfan Zhang and Arash A. Amini. "Adjusted chi-square test for degree-corrected block models.", 2020.