A method for modularity optimization based on total variation and signless total variation

*A **thesis** presented for the degree of **Master of Science**.*

Zijun Li, Freie Universität Berlin, Germany

Matriculation number: 5377221

zijun.li@fu-berlin.de

Supervisors:

Dr. Yves van Gennip, Delft University of Technology, Netherlands

Prof. Dr. Volker John, Freie Universität Berlin, Germany

Statutory Declaration

I declare that I have developed and written the enclosed Master thesis completely by myself, and have not used sources or means without declaration in the text. Any thoughts from others or literal quotations are clearly marked.

The Master thesis was not used in the same or in a similar version to achieve an academic grading or is being published elsewhere.

Berlin (Germany),

Zijun Li

Zijun Li

Abstract

In network science, one of the significant and challenging subjects is finding out the community structure. Modularity optimization has emerged as one of the effective methods for solving this problem in a graph. In this thesis, we show that modularity optimization in unsigned graphs is equivalent to a minimization problem with a total variation and a signless total variation. The Merriman–Bence–Osher (MBO) scheme is a classic technique for solving this minimization problem. Therefore, we derive an adapted MBO algorithm for speeding up the iteration steps. We tested our method on both synthetic and real datasets and got positive results compared to other well-known approaches.

Key words. modularity, MBO scheme, data clustering, Ginzburg–Landau functional

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Motivation

A complex network is a graph structure that depicts intricate systems as nodes and edges, where nodes represent objects and edges express their relationships. There is no relationship between objects if there is no an edge between nodes. The relationships between members of complex real-world systems can be intuitively described through visualization, such as urban transportation networks, airline networks, computer communication networks, and social networks.

A typical aspect of networks is community structure, which is a segmentation of the set of network nodes in which nodes in the same community are strongly connected, while connections between nodes in other communities are sparse. Many complex networks exhibit community structures, so studying the community structure of complex networks is motivating. In addition, the community structure can be used to study the common characteristics and properties of node groups. Moreover, the functions of similar nodes and potential connections between nodes can be predicted based on prior information [19], such as the number of communities is known.

A crucial aspect of community detection is that it can be used to retrieve important information from the network, which is represented as community structures. In addition, community detection is of great theoretical and practical value for understanding the topology and predicting the behavior of real-world networks and has been widely used in many fields, such as protein function prediction [59] and criminology [35]. Many real-world networks have weighted edges. When weighted networks are used to describe complex systems, the connectivity between nodes can be better expressed. One of challenges for community detection in networks is that the community structure is not universally defined [21]. As a result, it is difficult to establish precise standards for evaluating and comparing the performance of various algorithms, as these decisions frequently rely on the individual research issue and the system under investigation. Current community detection algorithms can be mainly classified as follows:

1) Graph partitioning algorithms [26]. These algorithms recursively remove the edges from the network to obtain the communities such that there are only a number of intergroup connections.

2) Clustering algorithms [55, 54]. The idea of this type of algorithms is to iteratively merge small groups into large one. In general, communities are local, but the computation of this class of algorithms is frequently global, resulting in higher complexity.

3) Optimization algorithms [28, 71]. This group of algorithms maximizes modularity using spectral clustering, simulated annealing, or other optimization techniques. Modularity is a metric for evaluating the quality of communities that have been found.

Numerous approaches have been proposed to find the optimal community in a relatively short time, and most of these techniques are based on objective function optimization. Modularity optimization is one of the most popular methods. Newman and Girvan established the concept of modularity, $\mathcal{Q}$, to assess the partition of a community [56], transforming community detection in complicated networks into a modularity function optimization problem. A high similarity of nodes within the same community and a low similarity of points in separate communities is an optimal depiction of the partitioning outcome. Specifically, the higher the $\mathcal{Q}$ value, the more significant is the community structure of the network. However, the modularity optimization is a NP–hard problem [6], thus numerous heuristic algorithms have been developed for modularity optimization have been proposed, including extremal optimization [17], greedy or Louvain algorithms [10, 3], simulated annealing [30] and spectral methods [26, 55].

Hu et al. [32] presented a total variation (TV)-based approach for optimizing network modularity using the Merriman-Bence-Osher (MBO) scheme. The current work follows up on [32] both theoretically and computationally. In particular, we make the following contributions.

1.2. Outline of contribution

Our main contribution is the development of an improved approach to modularity optimization, namely the modularity MBO scheme using projection. We reformulated modularity optimization as the minimization of total variation and signless total variation in a graph. Moreover, we have demonstrated the competitiveness of our approach compared to Louvain method, spectral clustering, and the Clauset−Newman−Moore method through extensive numerical tests on both synthetic and real data.

1.3. Structure of the thesis

This thesis is organized as follows.

Section 1 provides a review of the relevant literature on community detection.

Section 2 reviews the background information required for modularity optimization. Then we develop an equivalent formula for modularity optimization as a minimization problem with a total variation term and a signless total variation term.

Section 3 introduces the Ginzburg–Landau framework, the MBO scheme, and convex splitting.

Section 4 implements our algorithm, i.e., the MMBO scheme. In addition, we present the Nyström extension, which uses the QR decomposition to compute the eigenvalues and eigenvectors.

Section 5 is conducted to evaluate our method on synthetic and real–world networks. We also compared our method with Hu's method [32] and other state-of-the-art techniques, such as the Louvain method and spectral clustering.

Section 6 provides a summary of our results and possible future research directions.

There are additional appendices that provide further background and proofs.

Notation. Table 1.1 contains a summary of our notation.

Notation List			
Symbol	Description	Symbol	Description
$\mathcal{A}$	a partition of V	m	the number of the eigenvalues used for the diffusion step
B	modularity matrix	N_t	number of repetitions of the diffusion step
c_i	the community where node i is located	P	null model
d_i	the strength (i.e., weighted degree) of node i	Φ	double well potential
dt	timestep	$\mathcal{Q}$	modularity
D_W	degree matrix of W	Q_W	signless Laplacian of W
$\underline{e}$	row vector containing -1 and 1	$Q_{W_{sym}}$	signless symmetric normalized Laplacian of W
E	edge set	$Q_{W_{rw}}$	signless random walk Laplacian of W
$\|E\|$	the number of edges	$\mathbb{R}_+$	positive real numbers excluding 0
ε	parameter of the GL functional	U	node-cluster association matrix
ϵ	stopping criterion	V	node set
F_ε	Ginzburg–Landau (GL) functional	$\|V\|$	the number of nodes
f_ε	graph Ginzburg–Landau (GL) functional	$vol(V)$	the volume of G
G	graph	W	adjacency matrix
I	identity matrix	X	eigenvectors matrix
K	number of clusters	γ	resolution parameter
Λ	eigenvalue matrix	$\|\|\cdot\|\|_{L_1}$	Taxicab norm operator
L_W	graph Laplacian of W	$\|\|\cdot\|\|_{L_2}$	Euclidean norm operator
$L_{W_{sym}}$	symmetric normalized Laplacian of W	$\langle\cdot,\cdot\rangle$	standard inner product
$L_{W_{rw}}$	random walk Laplacian of W	$\langle\cdot,\cdot\rangle_{rw}$	inner product for $L_{W_{rw}}$ and $Q_{W_{rw}}$

Table 1.1.: Summary of frequently used symbols throughout the thesis.

2. Method

This section introduces basic terminology and derives new formulations of modularity, showing that maximizing modularity is comparable to minimizing a graph total variation (TV) and a signless total variation.

2.1. Graphical framework

In this thesis, we consider connected, weighted, undirected graphs $G = (V, E, \omega)$ with a node set V, an edge set $E = \{e_{ij}\}_{i,j=1}^{|V|}$ and edge weight ω_{ij} between node i and node j. The adjacency matrix $W = (\omega_{ij})_{i,j=1}^{|V|}$ is a nonnegative and symmetric matrix whose entries ω_{ij} are zero if $i = j$ or $(i, j) \notin E$. Degree $d_i = \sum_{j=1}^{|V|} \omega_{ij}$ denotes the weighted degree of node i. A volume (i.e., total edge weight) of the graph G is defined as $vol(V) = \sum_{i=1}^{|V|} d_i = \sum_{i,j \in V} \omega_{ij}$.

Let $\mathcal{A} = \{A_l\}_{l=1}^{K}$ be a family of disjoint sets for partitioning the node set V, where A_l satisfies $V = \bigcup_{l=1}^{K} A_l$ and $A_{l_1} \cap A_{l_2} = \emptyset$ if $l_1 \neq l_2$. Note that A_i could be empty, so the number of partition $\mathcal{A}$ is at most K. The partition $\mathcal{A}$ is equivalent to a node assignment $\{c_j\}_{j=1}^{|V|}$, where $c_j \in \{1, ..., K\}$ denotes the community that node j belongs to. We define $A_l = \{j \in V : c_j = l\}$ for $l \in \{1, ..., K\}$. Indicator function $\delta(c_i, c_j) = 1$ if nodes i and j are in the same community and $\delta(c_i, c_j) = 0$ otherwise.

Norms $||\cdot||_{L_1}$ and $||\cdot||_{L_2}$ represent the Taxicab norm operator and the Euclidean norm operator, respectively. For a vector $u = (u_1, u_2, ..., u_n)$, the corresponding $||u||_{L_1}$ and $||u||_{L_2}$ are defined as

$$\begin{aligned} ||u||_{L_1} &:= \sum_{i=1}^{n} |u_i|, \\ ||u||_{L_2} &:= \sqrt{(u_1)^2 + (u_2)^2 + ... + (u_n)^2}. \end{aligned} \tag{2.1}$$

Given a graph $G = (V, E, \omega)$ based on adjacency matrix W, and a function $u = (u_1, u_2, ..., u_{|V|}) : V \to \mathbb{R}$, we can define the graph total variation (TV) and signless total variation (TV^+) as

$$TV_W(u) := \frac{1}{2} \sum_{i,j \in V} \omega_{ij} |u_i - u_j|, \tag{2.2}$$

$$TV_W^+(u) := \frac{1}{2} \sum_{i,j \in V} \omega_{ij} |u_i + u_j|. \tag{2.3}$$

2.1.1. Laplacians for unsigned graphs

Graphs with nonnegative edge weights are known as unsigned graphs. We assume that all entries of adjacency matrix W are nonnegative and symmetric.

We defined the graph Laplacian matrix L [9] as

$$L_{ij} := \begin{cases} d_i & \text{if } i = j, \\ -\omega_{ij} & \text{otherwise.} \end{cases}$$

A diagonal degree matrix D of W is given by

$$(D_W)_{ii} = d_i.$$

Since G is connected, i.e., there is no isolated node, it is straightforward to have $d_i > 0$. Thus, matrix D_W is invertible. Then the graph Laplacian matrix L and its normalized variants, namely, the random walk L_{rw} and the symmetric normalized L_{sym} of W can be written as

$$L_W = D_W - W, \tag{2.4}$$
$$L_{W_{rw}} = D_W^{-1} L_W = I - D_W^{-1} W, \tag{2.5}$$
$$L_{W_{sym}} = D_W^{-\frac{1}{2}} L_W D_W^{-\frac{1}{2}} = I - D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}}. \tag{2.6}$$

For all $u \in \mathbb{R}^{|V|}$, the graph Laplacian L_W satisfies the following equations

$$(L_W u)_i = \sum_{j=1}^{|V|} \omega_{ij} \left(u_i - u_j\right).$$

In general, the standard inner product $\langle \cdot, \cdot \rangle$ on $\mathbb{R}^{|V|}$ is defined as $\langle u, Lu \rangle := u^T L u$. It follows

$$\begin{aligned} \langle u, L_W u \rangle &= u^T L_W u \\ &= u^T D_W u - u^T W u \\ &= \sum_{i=1}^{|V|} d_i u_i^2 - \sum_{i,j=1}^{|V|} \omega_{ij} u_i u_j \\ &= \frac{1}{2} \left(\sum_{i=1}^{|V|} d_i u_i^2 - 2 \sum_{i,j=1}^{|V|} \omega_{ij} u_i u_j + \sum_{j=1}^{|V|} d_j u_j^2 \right) \\ &= \frac{1}{2} \left(\sum_{i,j=1}^{|V|} \omega_{ij} u_i^2 - 2 \sum_{i,j=1}^{|V|} \omega_{ij} u_i u_j + \sum_{i,j=1}^{|V|} \omega_{ij} u_j^2 \right) \qquad \text{(Since } d_i = \textstyle\sum_{j=1}^{|V|} \omega_{ij}\text{)} \\ &= \frac{1}{2} \left(\sum_{i,j=1}^{|V|} \omega_{ij} (u_i^2 - 2u_i u_j + u_j^2) \right) \\ &= \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} (u_i - u_j)^2 \geq 0. \end{aligned} \tag{2.7}$$

Note that the symmetric graph Laplacian $L_{W_{sym}}$ satisfies

$$\left\langle u, L_{W_{sym}} u \right\rangle = u^T L_{W_{sym}} u = \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} \left(\frac{u_i}{\sqrt{d_i}} - \frac{u_j}{\sqrt{d_j}} \right)^2 \geq 0. \tag{2.8}$$

Similarly, for all $u \in \mathbb{R}^{|V|}$, the signless graph Laplacian Q, and its normalized variants Q_{rw} and Q_{sym} of W are defined as

$$Q_W = D_W + W, \tag{2.9}$$
$$Q_{W_{rw}} = D_W^{-1} Q_W = I + D_W^{-1} W, \tag{2.10}$$
$$Q_{W_{sym}} = D_W^{-\frac{1}{2}} Q_W D_W^{-\frac{1}{2}} = I + D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}}. \tag{2.11}$$

We obtain $\langle u, Q_W u \rangle$ and $\left\langle u, Q_{W_{sym}} u \right\rangle$ by

$$\begin{aligned}\langle u, Q_W u \rangle &= u^T Q_W u \\ &= u^T D_W u + u^T W u \\ &= \sum_{i=1}^{|V|} d_i u_i^2 + \sum_{i,j=1}^{|V|} \omega_{ij} u_i u_j \\ &= \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} (u_i + u_j)^2 \geq 0. \end{aligned} \tag{2.12}$$

$$\left\langle u, Q_{W_{sym}} u \right\rangle = u^T Q_{W_{sym}} u = \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} \left(\frac{u_i}{\sqrt{d_i}} + \frac{u_j}{\sqrt{d_j}} \right)^2 \geq 0. \tag{2.13}$$

Note that $L_{W_{rw}}$ and $Q_{W_{rw}}$ are not symmetric in general. To ensure that the operators are self-adjoint, we define another inner product $\langle \cdot, \cdot \rangle_{rw}$ for $L_{W_{rw}}$ and $Q_{W_{rw}}$, that is, for all $i \in V$,

$$\langle u, v \rangle_{rw} := \sum_{i \in V} u_i v_i d_i.$$

For all $u \in \mathbb{R}^{|V|}$, $L_{W_{rw}}$ and $Q_{W_{rw}}$ satisfy the following relationship:

$$\left(L_{W_{rw}} u\right)_i = \frac{1}{d_i} \sum_{j \in V} \omega_{ij} (u_i - u_j),$$
$$\left(Q_{W_{rw}} u\right)_i = \frac{1}{d_i} \sum_{j \in V} \omega_{ij} (u_i + u_j).$$

Therefore, we obtain

$$\langle u, L_{W_{rw}} u \rangle_{rw} = \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} \left(u_i - u_j\right)^2 \geq 0, \tag{2.14}$$

$$\langle u, Q_{W_{rw}} u \rangle_{rw} = \frac{1}{2} \sum_{i,j=1}^{|V|} \omega_{ij} \left(u_i + u_j\right)^2 \geq 0. \tag{2.15}$$

It can be seen that L_W, $L_{W_{sym}}$, $L_{W_{rw}}$, Q_W, $Q_{W_{sym}}$ and $Q_{W_{rw}}$ are all positive semi-definite. According to spectral graph theory [9], the following theorem indicates that the multiplicity of the 0 eigenvalue reveals the number of connected components of the graph.

Theorem 1. *The number of zero eigenvalues of the Laplacian L (i.e., the multiplicity of the 0 eigenvalue) equals the number of connected components of the graph G.*[1]

2.2. Review of the modularity function

Modularity is a structural graph metric that measures the quality of partitioning the node set into clusters. Let graph G be partitioned into K clusters. We represent this partition by a $K \times K$ unweighted symmetric matrix R where each element r_{ij} represents the fraction of edges connecting community i to community j among all edges of G. The trace $Tr(R) = \sum_{i \in K} r_{ii}$ indicates the percentage of edges connecting the nodes inside the same community. For a good community partitioning, it should have a trace with a high value. However, Newman [56] found that trace $Tr(R)$ cannot measure the quality of the partition thoroughly in some cases. For example, if all nodes are put into the same community, then the value $Tr(R) = 1$ is maximal, but it is evident that this partition is not meaningful for most cases.

Newman [56] illustrated this issue by considering a random graph G' that has the same community partition and the same number of edges as G but with arbitrary random connections between nodes. If G has a community structure, the fraction of within-community edges to the total number of edges should be greater than the expectation of that fraction in the random case. A greater difference between the actual within-community edges and the expectation indicates that the network is significantly distinct from the random network, i.e., the network has a community structure. Let $R_i = \sum_{j \in K} r_{ij}$ be the sum of the i–th row of R. Then the modularity is defined as

$$\mathcal{Q} := \sum_{i \in K} (r_{ii} - R_i^2) = Tr(R) - ||R_i^2||_{L_1}, \tag{2.16}$$

where $||R_i^2||_{L_1}$ is the sum of the all components of R_i^2.

Although the preceding definition is basic and obvious, there is one flaw: it does not consider the degree of nodes in the original network $G = (V, E, \omega)$ when constructing a random graph $G' = (V, E', p)$, where p_{ij} stands for the expected weights based on an appropriate null model. A null model is a random structure that estimates the quantity of edges that can be expected to randomly connect nodes within the same community. Therefore, Newman [55] redefined the modularity based on a $|V| \times |V|$ weighted symmetric matrix W as

$$\mathcal{Q}(\mathcal{A}) = \frac{1}{vol(V)} \sum_{i,j \in V} (\omega_{ij} - p_{ij})\, \delta(c_i, c_j). \tag{2.17}$$

[1]The proof is shown in appendix A.1

In [53, 57], Newman generated a collection of random graphs. One of the common null models is the Newman–Girvan model, where the probability of connection between node i and node j is $p_{ij} = \frac{d_i d_j}{vol(V)}$. Thus, the modularity can be written as

$$\mathcal{Q}(\mathcal{A}) = \frac{1}{vol(V)} \sum_{i,j \in V} \left(\omega_{ij} - \frac{d_i d_j}{vol(V)} \right) \delta(c_i, c_j). \tag{2.18}$$

For $\mathcal{Q} > 0$, it shows that community structure might exist. When $\mathcal{Q} = 0$, it indicates that all nodes are assigned to the same community, or the connections within the communities have no difference between the raw data and the null model. If $\mathcal{Q} < 0$, there is no community structure in the network.

Newman pointed out that in reality, the value of $\mathcal{Q}$ for a strong community structure should be between 0.3 and 0.7, with greater values being rare [71]. According to Fortunato and Barthelemy [20], there is a drawback in optimizing equation (2.18) to find community partitions, i.e., it is impossible to find a community partition in networks that contains many small communities. In particular, when the number of communities K in the network is greater than $\sqrt{vol(V)}$, the best modularity of the corresponding network would be lower than that of $K \leq \sqrt{vol(V)}$. The reason is that the modularity optimization based on equation (2.18) seeks community partitions that tend to integrate tiny communities into bigger ones, making this approach ineffective of detecting small communities in the network.

To solve the above problem, Arenas [1] proposed a generalized modularity based on the Reichardt and Bornholdt method [61]:

$$\mathcal{Q}(\mathcal{A}) = \frac{1}{vol(V)} \sum_{i,j \in V} \left(\omega_{ij} - \gamma \frac{d_i d_j}{vol(V)} \right) \delta(c_i, c_j), \tag{2.19}$$

where $\gamma > 0$ is a resolution parameter [61]. The distinction between (2.19) and (2.18) is the parameter γ, which allows (2.19) to be more flexible and find more network community partitions. Nevertheless, there are still some issues with the modularity optimization even in this case.

Lancichinetti and Fortunato [41] have showed that there is a problem with equation (2.19): When γ takes smaller values, namely $\gamma < 1$, it tends to merge small communities into large ones; when γ is larger, i.e., $\gamma > 1$, it splits large communities. Moreover, it was not possible to eliminate both biases simultaneously. The best partitioning should correspond to a reasonable γ resolution according to Jeub, Sporns, and Fortunato [34] et al. However, predetermination is challenging when one does not know about the network a priori.

In practice, the choice of γ depends on a concrete experiment. The problem of how to determine γ is a matter outside the scope of this thesis. In addition, to compare the performance of our method with other algorithms, the value we choose for γ depends on which value it is used in other papers.

2.3. Reformulation of modularity optimization for binary segmentation

In this subsection, we consider only the case of two clusters and derive a new expression for the modularity $\mathcal{Q}$, transforming the maximization problem into a minimization problem.

Let u be a real-valued function on the node set V, with value u_i on node i. We define a $\{-1,+1\}$-valued function as

$$\mathcal{V}^b := \{\forall i \in V, u_i \in \{-1,+1\}\}. \tag{2.20}$$

Specially, if $u \in \mathcal{V}^b$, one defines the sets

$$V_1 = \{i \in V : u_i = 1\} \text{ and } V_{-1} = \{i \in V : u_i = -1\}.$$

It is clear that if $u_i \in \mathcal{V}^b$, i.e., $u_i \in \{-1,+1\}$, then

$$u_i^2 + u_j^2 = 2,$$

which implies that

$$\begin{aligned}(u_iu_j+1) &= -\frac{1}{2}(u_i-u_j)^2+2,\\ -(u_iu_j+1) &= -\frac{1}{2}(u_i+u_j)^2.\end{aligned}$$

If $u_i = u_j \in V_1$ or $u_i = u_j \in V_{-1}$, namely if node i and j are in the same cluster, then $\delta(c_i,c_j) = 1 = \frac{1}{2}(u_iu_j+1)$. Similarly, if $u_i \neq u_j$, then $\delta(c_i,c_j) = 0 = \frac{1}{2}(u_iu_j+1)$. Hence the modularity $\mathcal{Q}$ for $K=2$ clusters can be rewritten as

$$\begin{aligned}\mathcal{Q} &= \frac{1}{2vol(V)}\sum_{i,j\in V}(\omega_{ij}-p_{ij})(u_iu_j+1)\\ &= \frac{1}{2vol(V)}\sum_{i,j\in V}\omega_{ij}(u_iu_j+1) - \frac{1}{2vol(V)}\sum_{i,j\in V}p_{ij}(u_iu_j+1)\\ &= -\frac{1}{4vol(V)}\sum_{i,j\in V}\omega_{ij}(u_i-u_j)^2 + \frac{1}{vol(V)}\sum_{i,j\in V}\omega_{ij} - \frac{1}{4vol(V)}\sum_{i,j\in V}p_{ij}(u_i+u_j)^2\\ &= -\frac{1}{2vol(V)}\left[\frac{1}{2}\sum_{i,j\in V}\omega_{ij}(u_i-u_j)^2 + \frac{1}{2}\sum_{i,j\in V}p_{ij}(u_i+u_j)^2\right] + \frac{1}{vol(V)}\sum_{i,j\in V}\omega_{ij}.\end{aligned} \tag{2.21}$$

For all $u \in \mathcal{V}^b$, one obtains

$$\begin{aligned}&(u_i-u_j)^2 = 0 \text{ or } (u_i-u_j)^2 = 4,\\ &2|u_i-u_j| = 0 \text{ or } 2|u_i-u_j| = 4.\end{aligned}$$

Therefore, if $u \in \mathcal{V}^b$, then it follows

$$(u_i - u_j)^2 = 2|u_i - u_j|. \tag{2.22}$$

Similarly, $(u_i + u_j)^2 = 2|u_i + u_j|$ if $u \in \mathcal{V}^b$.

For the third term in (2.21), we know that $vol(V) = \sum_{i,j \in V} \omega_{ij}$, so $\frac{1}{vol(V)} \sum_{i,j \in V} \omega_{ij} = 1$ and independent of u_i. Let

$$\begin{aligned} \mathcal{Q}_{bin} :&= \frac{1}{2} \sum_{i,j \in V} \omega_{ij}(u_i - u_j)^2 + \frac{1}{2} \sum_{i,j \in V} p_{ij}(u_i + u_j)^2 \\ &= \sum_{i,j \in V} \omega_{ij}|u_i - u_j| + \sum_{i,j \in V} p_{ij}|u_i + u_j| \\ &= 2TV_W(u) + 2TV_P^+(u), \end{aligned} \tag{2.23}$$

then the maximization of modularity $\mathcal{Q}$ (2.17) is equivalent to the minimization of $\mathcal{Q}_{bin}(u)$ over all functions $u \in \mathcal{V}^b$, which corresponds to minimizing the TV based on G with W plus the signless TV based on G' with P.

2.4. Generalization to multiple clusters

In reality, many networks consist of more than two communities. Therefore, it is important to extend the approach of Section 2.3 to identify appropriate partitions of the node set into multiple clusters.

Given a vector-valued function $u = (u^{(1)}, ..., u^{(K)}) : V \to \mathbb{R}^K$, where $u^{(l)}$ is the l–th component of u, and a non-negatively symmetric matrix W, the generalizations of graph TV and signless TV are given as

$$TV_W(u) := \sum_{l=1}^{K} TV_W(u^{(l)}) = \sum_{l=1}^{K} \frac{1}{2} \sum_{i,j \in V} \omega_{ij}|u_i^{(l)} - u_j^{(l)}|, \tag{2.24}$$

$$TV_W^+(u) := \sum_{l=1}^{K} TV_W^+(u^{(l)}) = \sum_{l=1}^{K} \frac{1}{2} \sum_{i,j \in V} \omega_{ij}|u_i^{(l)} + u_j^{(l)}|. \tag{2.25}$$

In multiple clusters case, we can identify $u_i^{(l)}$ with an $|V| \times K$ matrix U where $u_{il} = u_i^{(l)}$.

Definition 1. Let $Pt(V)$ be the set of all partitions of the node set V. For an arbitrary partition $\{A_1, ..., A_K\}$ defined in Section 2.1, we define an $|V| \times K$ *indicator matrix* as

$$t_{il} = \begin{cases} 1, & \text{if } i \in A_l, \\ -1, & \text{if } i \in A_l^C, \end{cases} \tag{2.26}$$

where t_{il} is an element of T for $i \in \{1, ..., |V|\}$ and A_l^C is the complement of A_l. In other words, we say matrix $T \in Pt(V)$ if T is the indicator matrix of some partition.

For matrix $U \in Pt(V)$, we obtain that

$$|u_{il} - u_{jl}| = 0 \text{ if } i, j \in A_l \text{ or } i, j \in A_l^C, \tag{2.27}$$
$$|u_{il} - u_{jl}| = 2 \text{ if } i \in A_l, j \in A_l^C \text{ or } i \in A_l^C, j \in A_l, \tag{2.28}$$
$$|u_{il} + u_{jl}| = 0 \text{ if } i \in A_l, j \in A_l^C \text{ or } i \in A_l^C, j \in A_l, \tag{2.29}$$
$$|u_{il} + u_{jl}| = 2 \text{ if } i, j \in A_l \text{ or } i, j \in A_l^C. \tag{2.30}$$

Then the generalizations of graph TV and signless TV with respect to $U \in Pt(V)$ can be represented as

$$\begin{aligned} TV_W(U) &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l} + \sum_{i\in A_l, j\in A_l^C} + \sum_{i\in A_l^C, j\in A_l} + \sum_{i,j\in A_l^C}\right)\omega_{ij}|u_{il} - u_{jl}| \\ &= \sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}\omega_{ij}|u_{il} - u_{jl}| \\ &= 2\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}\omega_{ij}, \end{aligned} \tag{2.31}$$

$$\begin{aligned} TV_W^+(U) &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l} + \sum_{i\in A_l, j\in A_l^C} + \sum_{i\in A_l^C, j\in A_l} + \sum_{i,j\in A_l^C}\right)\omega_{ij}|u_{il} + u_{jl}| \\ &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l} + \sum_{i,j\in\{A_1,\ldots,A_K\}\setminus A_l}\right)\omega_{ij}|u_{il} + u_{jl}| \\ &= \sum_{l=1}^{K}\sum_{i,j\in A_l}\omega_{ij} + \left(\sum_{l=1}^{K}\sum_{i,j\in A_l}\omega_{ij} + (K-2)vol(V)\right) \\ &= 2\sum_{l=1}^{K}\sum_{i,j\in A_l}\omega_{ij} + (K-2)vol(V). \end{aligned} \tag{2.32}$$

Consider a graph G and fix K. Let $G' = (V, E', p)$ be another graph, where E' and p are an edge set and edge weights of G', respectively. Since $\delta(c_i, c_j) = 1$ for any pair of nodes that is in the same community, i.e., $i, j \in A_l$ for $l \in \{1, 2, ..., K\}$, equation

(2.17) can be written as follows

$$
\begin{aligned}
\mathcal{Q}(\mathcal{A}) &= \frac{1}{vol(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l}\omega_{ij} - \frac{1}{vol(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l}p_{ij} \\
&= \frac{1}{vol(V)}\sum_{l=1}^{K}\left(\sum_{i\in A_l, j\in V}\omega_{ij} - \sum_{i\in A_l, j\in A_l^C}\omega_{ij}\right) - \frac{1}{vol(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l}p_{ij} \\
&= \frac{1}{vol(V)}\sum_{i,j=1}^{|V|}\omega_{ij} - \frac{1}{vol(V)}\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}\omega_{ij} - \frac{1}{vol(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l}p_{ij} \\
&= -\frac{1}{vol(V)}\left(\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}\omega_{ij} + \sum_{l=1}^{K}\sum_{i,j\in A_l}p_{ij}\right) + \frac{1}{vol(V)}\sum_{i,j=1}^{|V|}\omega_{ij}. \qquad (2.33)
\end{aligned}
$$

Similar to the binary case above, the third term in (2.33) is 1. Let

$$
\begin{aligned}
\mathcal{Q}_{mul}(\mathcal{A}) :&= \sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}\omega_{ij} + \sum_{l=1}^{K}\sum_{i,j\in A_l}p_{ij} \\
&= \frac{1}{2}TV_W(U) + \frac{1}{2}TV_P^+(U) - \frac{1}{2}(K-2)vol(V). \qquad (2.34)
\end{aligned}
$$

The last term in (2.34) is a constant when K is fixed. Therefore, the maximization of the modularity $\mathcal{Q}$ of the multiple clusters over all matrices $U \in Pt(V)$ is equivalent to the minimization of $\mathcal{Q}_{mul}$ over that same set. Note that $\mathcal{Q}_{mul}$ is associated with the graph TV in a given graph $G = (V, E, \omega)$ and the signless TV in another graph $G' = (V, E', p)$.

In addition to W and P, another way to construct adjacency matrices is also considered in the thesis. Following the above definitions and notations of W and P, the modularity matrix B [55] is defined as

$$B = W - P. \qquad (2.35)$$

Note that ω_{ij} and p_{ij} may be overlapping, i.e., $\omega_{ij} \le p_{ij}$. It results in B having both positive and negative elements. We define the positive part of B as B^+ and the negative part of B as B^-, whose entries are

$$b_{ij}^+ := max\{b_{ij}, 0\} \text{ and } b_{ij}^- := -min\{b_{ij}, 0\}, \qquad (2.36)$$

so $B = B^+ - B^-$. Then (2.33) can be rewritten as

$$
\begin{aligned}
\mathcal{Q}(\mathcal{A}) &= \frac{1}{vol(V)}\sum_{i,j\in V}\left(b_{ij}^+ - b_{ij}^-\right)\delta(c_i, c_j) \\
&= -\frac{1}{vol(V)}\left(\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^C}b_{ij}^+ + \sum_{l=1}^{K}\sum_{i,j\in A_l}b_{ij}^-\right) + \frac{1}{vol(V)}\sum_{i,j=1}^{|V|}b_{ij}^+. \qquad (2.37)
\end{aligned}
$$

In this case, (2.34) is presented as

$$\mathcal{Q}_{mul}(\mathcal{A}) := \frac{1}{2}TV_{B^+}(U) + \frac{1}{2}TV_{B^-}^+(U) - \frac{1}{2}(K-2)vol(V). \qquad (2.38)$$

3. Diffuse interface methods

Graph–based classification algorithms on unsigned graphs have been proven to be useful in a number of real world applications [55, 2]. In particular, diffuse interface methods [2], an approach that uses efficient PDE techniques to handle binary segmentation problems, have been introduced with highly promising results. The Ginzburg–Landau (GL) functional associated with a graph Laplacian, whose minimization is associated with the minimization of the total variation, is widely used in diffuse interface approaches.

3.1. The classical continuum Ginzburg–Landau functional

The classical continuum Ginzburg–Landau (GL) functional is given by

$$F_\varepsilon(u) := \frac{\varepsilon}{2}\int_\Omega |\nabla u(x)|^2 dx + \frac{1}{\varepsilon}\int_\Omega \Phi\left(u\right) dx, \tag{3.1}$$

where $u \in W^{1,2}(\Omega)$ is a phase field describing the different phases in the system, ∇ denotes the spatial gradient operator and $\varepsilon > 0$ is a real constant parameter. The function $\Phi(u)$ is a double well potential with two minima. For example, a classical choice is polynomial $\Phi(u) = \frac{1}{4}(u^2-1)^2$ that has minima $u = -1$ and $u = 1$. In this case, we say node i is in one phase if $u = 1$ and it is in another phase if $u = -1$. For small $\varepsilon > 0$, minimizing F_ε will cause the function u to approach the minima of Φ. For instance, $\Phi(u) = \frac{1}{4}(u^2-1)^2$ has minima at -1 and 1, then the minimizer of (3.1) tends to have $\{u \approx -1\}$ and $\{u \approx 1\}$.

It has been established that the functional (3.1) consists of two terms: a smoothing term and a potential term. More specifically, the first term measures the differences between the components of the field and the second term quantifies how distant each component is from a target value (± 1 in the example above). Consequently, minimizing the first term results in smoother fields, while minimizing the second term penalizes variations from the double well potential minima.

Consider $u : \Omega \to \mathbb{R}$ is binary, where $\Omega \subset \mathbb{R}$, and $\Phi(u)$ has minima at $+1$ and -1. It can be known from [37] that the GL functional Γ–converges [1] to the total variation (TV) semi–norm:

$$F_\varepsilon(u) \to_\Gamma F_0(u) := \psi \int_\Omega |\nabla u| \tag{3.2}$$

as $\varepsilon \to 0$, where ψ is a constant. The proof of (3.2) goes beyond the scope of this thesis, but it was discussed at some length in [37]. In addition, the graph version of (3.2) is provided in [66].

[1]The appendix A.2 provides a summary of Γ–convergence.

In reality, multiclass classification is designed to handle situations where there are more than two clusters. For $K > 2$ clusters, given a vector–valued function $u = \left(u^{(1)}, ..., u^{(K)}\right)$, the classical GL functional of two components in (3.1) is generalized as

$$F_\varepsilon(u) = \frac{\varepsilon}{2}\int_\Omega \sum_{l=1}^{K} |\nabla u^{(l)}|^2 dx + \frac{1}{\varepsilon}\int_\Omega \Phi_{mul}\left(u\right) dx, \tag{3.3}$$

where multi-well potential function $\Phi_{mul}(u)$ has K different minimizers instead of two.

One can utilize the L_2 gradient flow to minimize (3.1), resulting in the Allen–Cahn (AC) equation as follow

$$\frac{\partial u}{\partial t} = \varepsilon \Delta u - \frac{1}{\varepsilon}\Phi'(u), \tag{3.4}$$

where Δ is the Laplacian operator. The AC equation can be evolved by using the following time-splitting approach for small ε.

(1) Step 1 is propagation using $\frac{\partial u}{\partial t} = \varepsilon \Delta u$.

(2) Step 2 is propagation using $\frac{\partial u}{\partial t} = -\frac{1}{\varepsilon}\Phi'(u)$.

For $\varepsilon \to 0$, the time-splitting scheme corresponds to iteration between step (i) and step (ii), which is the Merriman–Bence–Osher (MBO) scheme. The standard two steps of the MBO scheme are as follows.

(i) Diffusion. Let $u^{n+\frac{1}{2}} = S(\delta t)u^n$, where $S(\delta t)$ is a propagator by time δt and plays a similar role to step (1), associating with heat equation

$$\frac{\partial u}{\partial t} = \Delta u. \tag{3.5}$$

(ii) Thresholding. Let

$$u_i^{n+1} := \begin{cases} 1 & \text{if } u_i^{n+\frac{1}{2}} \geq 0 \\ -1 & \text{if } u_i^{n+\frac{1}{2}} < 0. \end{cases} \tag{3.6}$$

The iteration repeats until a convergence condition is achived.

3.2. Binary classification with the graph Ginzburg–Landau functional

In the following section the corresponding problem based on graph information is presented.

Let L be the graph Laplacian, and $u : V \to \mathbb{R}$ be a real–valued function on V, where u_i is the value of node i. The (unsigned) graph GL functional is defined as

$$\begin{aligned} f_\varepsilon(u) :&= \frac{\varepsilon}{2}\langle u, Lu\rangle + \frac{1}{\varepsilon}\sum_{i\in V}\Phi\left(u_i\right) \\ &= \frac{\varepsilon}{4}\sum_{i,j\in V}\omega_{ij}\left(u_i - u_j\right)^2 + \frac{1}{\varepsilon}\sum_{i\in V}\Phi\left(u_i\right). \qquad \text{(By (2.7))} \end{aligned} \tag{3.7}$$

The first term in $f_\varepsilon(u)$ takes the place of the gradient term in (3.1) and the second one is the double-well potential. In minimizing $f_\varepsilon(u)$, the first one tends to promotes u to assign similar values to nodes that are linked together by highly weighted edges; while the second term forces u to be close to the minima of $\Phi(u)$. Minimizers of f_ε are therefore approximate indicator vectors of clusters with few highly weighted edges connecting them [13].

Similar to the (unsigned) graph GL functional and following the idea of [36], the signless graph GL functional is given as

$$f^+_\varepsilon(u) := \frac{\varepsilon}{4}\sum_{i,j\in V}\omega_{ij}\left(u_i + u_j\right)^2 + \frac{1}{\varepsilon}\sum_{i\in V}\Phi\left(u_i\right). \qquad \text{(By (2.12))} \tag{3.8}$$

As previously, the Allen–Cahn equation for graphs is obtained by minimizing the graph GL functional from (3.7) using the gradient descent approach, given by

$$\frac{\partial u}{\partial t} = -\varepsilon\left[Lu\right]_i - \frac{1}{\varepsilon}\Phi'\left(u\right), \tag{3.9}$$

where $\left[Lu\right]_i$ denotes the i–th component of the vector Lu.

3.2.1. MBO scheme for binary classification

In [2], the authors presented a segmentation technique in a network framework in order to minimize the GL functional using the methods of gradient flow and convex splitting. Another method for approximately minimizing the GL functional is to use a graph version of the Merriman–Bence–Osher (MBO) scheme (also called threshold dynamics) [51]. The goal of the MBO approach in [51] is to produce an efficient and straightforward way to approximate flow by mean curvature.

Following the idea in [49], we started by ignoring the term $\Phi'\left(u\right)$ in (3.9) and then discretizing (3.9) as follows

$$\frac{u^{n+\frac{1}{2}} - u^n}{dt} = -\varepsilon L u^{n+\frac{1}{2}}, \tag{3.10}$$

where dt is a timestep and $(n+\frac{1}{2})$ denotes an intermediate step between the n–th step and $(n+1)$–th step. Then the diffusion step of the MBO scheme is formed as

$$u^{n+\frac{1}{2}} = \left(I_{|V|\times|V|} + \varepsilon dt L\right)^{-1} u^n, \tag{3.11}$$

where $I_{|V|\times|V|}$ is an identity matrix. Let X be a matrix whose columns are the eigenvectors of L, and Λ be a diagonal matrix with corresponding eigenvalues. When we write L as its eigendecomposition version $L = X\Lambda X^T$, then equation (3.11) becomes

$$u^{n+\frac{1}{2}} = \left[X(I_{|V|\times|V|} + \varepsilon dt\Lambda)X^T\right]^{-1} u^n. \tag{3.12}$$

The role of the double well potential can be approximated using a thresholding step since it forces the components u_i to take only two values 1 or -1, i.e.,

$$u_i^{n+1} := \begin{cases} 1, & \text{if } u_i^{n+\frac{1}{2}} \geq 0, \\ -1, & \text{if } u_i^{n+\frac{1}{2}} < 0. \end{cases} \tag{3.13}$$

One continues the iteration between the diffusion step (3.11) and the thresholding step (3.13) until a convergence criterion is achieved. Since u_i can only take two values of 1 or -1 after the thresholding step, this approach is suitable for binary segmentation.

The minimization of f_ε is affected by the parameter ε since it sets a scale for the diffuse interface. Besides, small ε also increases the relative weight of the potential term in relation to the smoothing term. The problem of discrete length scales, on the other hand, provides a restriction on the lower bound of ε. To avoid this issue, the MBO scheme uses a thresholding step instead of $\Phi'(u)$, which allows us to set $\varepsilon = 1$.

3.3. Multiple clusters extension

Recently, there have been a number of significant applications using the graph-based Ginzburg–Landau method for data clustering, community detection, and image segmentation [24, 43, 48, 49]. The majority of current studies and techniques address the binary classification problem. The multiple clusters case, where the nodes is divided into more than two clusters, is more difficult. Therefore, in this section, we show how to extend the graph GL method from binary segmentation to multi-class clustering and classification issues.

We define a set of row vectors

$$\underline{e}_j := (e_1, e_2, ..., e_K) \in \{-1, 1\}^K, \tag{3.14}$$

where components satisfy the j–th component equals 1 and all other components are -1, i.e., $e_{i=j} = 1$ and $e_{i\neq j} = -1$.

Given $|V|$ nodes, $K > 2$ clusters and a matrix $U \in Pt(V)$. Let $\Phi(u) = \frac{1}{4}(u^2-1)^2$. In the multi-class case, the graph GL functional in (3.7) generalizes to

$$f_\varepsilon(U) := \frac{\varepsilon}{2}\langle U, LU\rangle + \frac{1}{\varepsilon}\sum_{i\in V}\left(\prod_{j=1}^{K}\frac{1}{4}||U_i - \underline{e}_j||_{L_1}^2\right), \tag{3.15}$$

where $\langle U, LU\rangle = Tr\left(U^T LU\right)$ and U_i is the i–th row of U.

3.3.1. Ginzburg–Landau relaxation

Given matrix $U \in Pt(V)$. The minimization of (2.34) defined over $U \in Pt(V)$ is equivalent to the minimization of

$$S(U) := \begin{cases} \mathcal{Q}_{mul}(\mathcal{A}), & \text{if } U \in Pt(V), \\ +\infty, & \text{otherwise.} \end{cases} \tag{3.16}$$

Given two $|V| \times |V|$ non-negatively symmetric matrices F and H, the number of clusters K, and partition $\{A_1, ..., A_K\}$ defined in Section 2.1, we define the the Ginzburg–Landau relaxation of (3.16) as

$$S_\varepsilon(U) := \frac{1}{2}\langle U, L_F U\rangle + \frac{1}{2}\langle U, Q_H U\rangle - \frac{1}{2}(K-2)vol(V) + \frac{1}{\varepsilon^2}\sum_{i=1}^{|V|}\Phi_{multi}(U_i) \tag{3.17}$$

(By (2.8) and (2.12))

$$= \frac{1}{4}\sum_{l=1}^{K}\sum_{i,j\in V} f_{ij}\left(u_{il}-u_{jl}\right)^2 + \frac{1}{4}\sum_{l=1}^{K}\sum_{i,j\in V} h_{ij}\left(u_{il}+u_{jl}\right)^2$$

$$-\frac{1}{2}(K-2)vol(V) + \frac{1}{\varepsilon^2}\sum_{i=1}^{|V|}\Phi_{multi}(U_i), \tag{3.18}$$

where $\varepsilon > 0$, U_i is the i–th row of U, and Φ_{multi} is a multiwell potential [23] with minima on $\underline{e}_j$ (3.14). The goal of Φ_{multi} is to promote U_i toward one of the minima. Moreover, for the sake of this thesis, it is not important what the exact expression of Φ_{multi} is, since we would discard it while using the MBO scheme.

The following theorem shows a Γ–convergence[2] result.

Theorem 2. *The functional S_ε (3.18) defined over $\mathbb{R}^{|V|}$, Γ-converges to S (3.16) as $\varepsilon \to 0$.*

Proof. Our proof largely follow [4]. By observing (3.18), it can be seen that the third term is a constant when K is fixed, and the first and second terms are continuous and independent of ε. Thus, they would not interfere with the results of Γ–convergence [5]. Consequently, we only need to prove that $\varphi_\varepsilon(U) := \frac{1}{\varepsilon^2}\sum_{i=1}^{|V|}\Phi_{multi}(U_i)$ Γ–converges to $\varphi_0(U)$ as $\varepsilon \to 0^+$, where

$$\varphi_0(U) := \begin{cases} 0 & \text{if } U \in Pt(V), \\ +\infty & \text{otherwise.} \end{cases} \tag{3.19}$$

Let $\{\varepsilon_n\}_{n=1}^{\infty}$ be a positive sequence such that $\varepsilon_n \downarrow 0$ as $n \to \infty$. For the lower bound condition, note that $\varphi_\varepsilon(U) \geq 0$ for any U. Suppose that a sequence of matrices $\{(U)_1, (U)_2, ...\}$ satisfies $(U)_n \to U$ as $n \to \infty$. If $U \in Pt(V)$, then

$$\varphi_0(U) = 0 \leq \liminf_{n\to\infty} \varphi_{\varepsilon_n}\left((U)_n\right). \tag{3.20}$$

[2]The appendix A.2 presents an overview of Γ–convergence

If $U \notin Pt(V)$, then $\varphi_0(U) = +\infty$. Moreover, there exists a node i such that $U_i \notin \{\underline{e}_j\}_{j=1}^K$, i.e., node i is isolated or assigned to two or more clusters. Thus,

$$\varphi_0(U) = +\infty \le \liminf_{n\to+\infty} \varphi_{\varepsilon_n}((U)_n) = +\infty. \tag{3.21}$$

For the upper bound condition, if $U \in Pt(V)$, then assume $(U)_n = U$ for all n gives the required sequence and we obtain $\varphi_{\varepsilon_n}((U)_n) = 0$. Hence,

$$\varphi_0(U) = 0 \ge \limsup_{n\to\infty} \varphi_{\varepsilon_n}((U)_n) = 0. \tag{3.22}$$

If $U \notin Pt(V)$, then $(U)_n = U$ for all n still satisfies the upper bound condition. Thus, φ_ε Γ–converges to φ_0.

□

3.3.2. Multiclass MBO scheme

As discussed in Section 3.2.1, we choose $\varepsilon = 1$ in this thesis. Another parameter that we need to determine is timestep dt. A method for choosing dt in the MBO scheme is presented in [4]. Their method is an effective strategy and requires less manual adjustment of dt than other approaches[3].

Fix the number of clusters K, and consider a $|V| \times K$ indicator matrix $U \in Pt(V)$ with elements u_{il}. Then S_ε (3.17) can be minimized using the gradient-descent equation, that is,

$$\frac{d}{dt}U = -\frac{1}{2}L_F U - \frac{1}{2}Q_H U - \frac{1}{\varepsilon^2}\Phi'_{multi}(U). \tag{3.23}$$

An approximate solution U^{n+1} is obtained by alternating between the following two steps.

Step 1: Diffusion. Given U^n, we get $U^{n+\frac{1}{2}}$ as the solution of

$$\frac{d}{dt}U = -\frac{1}{2}(L_F + Q_H)U. \tag{3.24}$$

So, an explicit solution of $U^{n+\frac{1}{2}}$ with timestep dt would be

$$U^{n+\frac{1}{2}} = exp(-\frac{1}{2}dtL_{mix})U^n := JU^n, \tag{3.25}$$

where $J = exp(-\frac{1}{2}dtL_{mix})$ and $L_{mix} = L_F + Q_H$.

Step 2: Thresholding. We can view the threshold processing as determining the position of the largest element of U_i, where U_i is the i–th row of U. If u_{il} is the largest

[3]See the appendix A.3 for more details.

component of U_i, then we assign $u_{il} = 1$ and the rest of elements in U_i are -1. In other words, one obtains U^{n+1} by solving

$$(U_i)^{n+1} = \underline{e}_j \text{ with } j = \operatorname*{argmax}_{l=1,...,K} \left\{ (u_{il})^{n+\frac{1}{2}} \right\} \tag{3.26}$$

The iteration continues until a stopping condition is achieved. In the end, one gets $U^{n+1} = (U_1^{n+1}, ..., U_{|V|}^{n+1})^T$.

4. The modularity MBO algorithms

This section focuses on the description of the modularity MBO (MMBO) scheme. First, we will demonstrate the algorithm of MMBO using projection on the eigenvectors. Then, an alternative variant of the MMBO scheme, i.e., MMBO using finite differences, will be presented. Finally, we will explain how to compute the eigenvalues and eigenvectors of $L_{mix} = L_F + Q_H$ using the Nyström extension with QR decomposition.

4.1. The MMBO algorithm using projection on the eigenvectors.

The most expensive part of this process is the assessment of the matrix exponential in (3.25). In practice, computing the full spectrum of L_{mix} may be impossible when L_{mix} is large. As a result, it makes logical to use a pseudo-spectral technique, which involves solving the matrix exponential using the eigenvalues and eigenvectors of L_{mix}. Specifically, employ the eigendecomposition of $L_{mix} = X_{mix}\Lambda_{mix}X_{mix}^T$, where Λ_{mix} is a diagonal matrix of eigenvalues and X_{mix} is a matrix whose columns are the eigenvectors of L_{mix}, then J in (3.25) can be rewritten as

$$J = exp(-\frac{1}{2}dtL_{mix}) = X_{mix}exp(-\frac{1}{2}dt\Lambda_{mix})X_{mix}^T. \tag{4.1}$$

Therefore, (3.25) is expressed as

$$U^{n+\frac{1}{2}} = \left(X_{mix}exp(-\frac{1}{2}dt\Lambda_{mix})X_{mix}^T\right)U^n. \tag{4.2}$$

One method of approximating Λ_{mix} is to use a truncated matrix, which retains only the m smallest eigenvalues rather than the full matrix. Likewise, we approximate X_{mix} in a similar way.

The MBO scheme is repeated until there is trivial difference between the current iteration and the previous one. As in [24], the stopping criterion is denoted by

$$\frac{\max_i ||U_i^{n+1} - U_i^n||_{L_2}^2}{\max_i ||U_i^{n+1}||_{L_2}^2} < \epsilon \in \mathbb{R}_+. \tag{4.3}$$

The MMBO scheme using projection on the eigenvectors is summarized in Algorithm 1. Unless otherwise stated, the parameters of our approach are defined as follows.

- $K \in \mathbb{N}$ which is the number of clusters.
- $m \in \mathbb{N}$ which is the number of eigenvalues and eigenvectors to use for (4.2).

Algorithm 1 The MMBO scheme using projection on the eigenvectors

```
Require: $K, m, \epsilon, \gamma, F, H$ are $|V| \times |V|$ adjacency matrices
    Initialize:
    $D, L_F \leftarrow F$                                   ▷ Computation $L_F$ of matrix $F$
    $Q_H \leftarrow H$                                      ▷ Computation $Q_H$ of matrix $H$
    $L_{mix} \leftarrow L_F + Q_H$
    $\Lambda_m, X_m \leftarrow L_{mix}$                     ▷ Eigendecomposition ($L_{mix}$)
    $\Lambda_m^{min} \leftarrow \Lambda_m$                  ▷ Pick the smallest eigenvalue
    $U^0 = (-1)_{|V| \times K}$
    for $i = 1 \rightarrow |V|$ do
        $l = random.sample(K, 1)$
        $u_{il}^0 = 1$
    end for
    $d_{max} \leftarrow D$                                  ▷ Pick the largest degree of $D$
    $dt_{low} \leftarrow \frac{0.15}{(\gamma+1) d_{max}}$
    $dt_{upp} \leftarrow \frac{1}{\Lambda_m^{min}} log \left( ||U^0||_{L_2} \right)$
    $dt \leftarrow \sqrt{dt_{low} dt_{upp}}$
    $M_{exp} \leftarrow exp(-\frac{1}{2} dt \Lambda_m)$
    $n \leftarrow 0$
    while Stop criterion not satisfied and $n \leq 10000$ do
        Diffusion:
        $U^{n+\frac{1}{2}} \leftarrow X_m M_{exp} X_m^T U^n$
        Thresholding:
            $$(U_i)^{n+1} = \underline{e}_j \text{ with } j = \underset{l=1,\ldots,K}{\operatorname{argmax}} \left\{ (u_{il})^{n+\frac{1}{2}} \right\}$$
        $n = n + 1$
    end while
```

- $F_{|V| \times |V|}, H_{|V| \times |V|}$ which are adjacency matrices whose elements are non-negative.

- $\epsilon \in \mathbb{R}_+$ which is the stopping criterion in (4.3).

The command $random.sample(K, N)$ returns a list of length N, with uniformly random sampling from 1 to K of N distinct elements ($N \leq K$).

To obtain better performance, we employed the symmetric normalized L_{mix_s} and the random walk L_{mix_r} in numerical experiments. Consequently, we need to compute $L_{F_{sym}}$ and $Q_{H_{sym}}$ or $L_{F_{rw}}$ and $Q_{H_{rw}}$ at initialization.

4.2. Alternative variant of the MMBO scheme

The MMBO scheme using finite difference employs an alternative approach in the diffusion step of the MBO iteration. When the diffusion step (3.11) is repeated $N_t \in \mathbb{N}_+$ times before the threshold processing (3.13), dt should be divided by N_t. Consequently, using (3.12) and the diffusion step of MBO scheme for the matrix U is equivalent to

$$\begin{aligned} \frac{U^{n+\frac{1}{2}} - U^n}{dt} &= -\frac{1}{2}\left(L_F + Q_H\right) U^{n+\frac{1}{2}}, \\ \Longleftrightarrow U^{n+\frac{1}{2}} &= \left[I + \frac{1}{2} dt \left(L_F + Q_H\right)\right]^{-1} U^n. \end{aligned} \tag{4.4}$$

To obtain $U^{n+\frac{1}{2}}$, we repeat this diffusion step N_t times for timestep $\frac{dt}{N_t}$, and it leads to

$$U^{n+\frac{1}{2}} = \left[I + \frac{dt}{2N_t}\left(L_F + Q_H\right)\right]^{-N_t} U^n. \tag{4.5}$$

Let L_{mix}^{-1} be the pseudoinverse matrix of L_{mix}. Since L_{mix} is symmetric, its eigenvectors matrix X_{mix} is orthogonal, i.e., $X_{mix}^T = X_{mix}^{-1}$. Then we define a matrix J_{fd} as

$$\begin{aligned} J_{fd} :&= \left[I + \frac{dt}{2N_t} L_{mix}\right]^{-N_t} \\ &= \left[X_{mix}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right) X_{mix}^T\right]^{-N_t} \\ &= \left(X_{mix}^T\right)^{-N_t}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right)^{-N_t}\left(X_{mix}\right)^{-N_t} \\ &= \left(\left(X_{mix}^T\right)^{-1}\right)^{N_t}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right)^{-N_t}\left(X_{mix}^{-1}\right)^{N_t} \\ &= \left(X_{mix}\right)^{N_t}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right)^{-N_t}\left(X_{mix}^T\right)^{N_t} \\ &= \left[X_{mix}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right)^{-1} X_{mix}^T\right]^{N_t}. \end{aligned} \tag{4.6}$$

We use the same thresholding step, and stopping conditions as for the MMBO scheme using projection. The MMBO scheme using finite difference is summarized in Algorithm 2. The main difference between Algorithm 1 and Algorithm 2 is the diffusion step, as seen in (4.1) and (4.6).

Algorithm 2 The MMBO scheme using finite difference

```
Require: $K, m, \epsilon, N_t, \gamma, F, H$ are $|V| \times |V|$ adjacency matrices
  Initialize:
  $D, L_F \leftarrow F$                                    ▷ Computation $L_F$ of matrix $F$
  $Q_H \leftarrow H$                                       ▷ Computation $Q_H$ of matrix $H$
  $L_{mix} \leftarrow L_F + Q_H$
  $\Lambda_m, X_m \leftarrow L_{mix}$                      ▷ Eigendecomposition ($L_{mix}$)
  $\Lambda_m^{min} \leftarrow \Lambda_m$                   ▷ Pick the smallest eigenvalue
  $U^0 = (-1)_{|V| \times K}$
  for $i = 1 \rightarrow |V|$ do
      $l = random.sample(K, 1)$
      $u_{il}^0 = 1$
  end for
  $d_{max} \leftarrow D$                                   ▷ Pick the largest degree of $D$
  $dt_{low} \leftarrow \frac{0.15}{(\gamma+1)d_{max}}$
  $dt_{upp} \leftarrow \frac{1}{\Lambda_m^{min}} log\left(||U^0||_{L_2}\right)$
  $dt \leftarrow \sqrt{dt_{low} dt_{upp}}$
  $J_{fd} \leftarrow \left[X_{mix}\left(I + \frac{dt}{2N_t}\Lambda_{mix}\right)^{-1} X_{mix}^T\right]^{N_t}$
  $n \leftarrow 0$
  while Stop criterion not satisfied and $n \leq 10000$ do
      Diffusion:
      for $s = 1 \rightarrow N_t$ do
          $U^{n+\frac{1}{2}} \leftarrow J_{fd} U^n$
          $s = s + 1$
      end for
      Thresholding:
          $$(U_i)^{n+1} = \underline{e}_j \text{ with } j = \underset{l=1,\ldots,K}{\operatorname{argmax}} \left\{(u_{il})^{n+\frac{1}{2}}\right\}$$
      $n = n + 1$
  end while
```

4.3. Nyström extension with QR decomposition.

One recurrent difficulty in many fields of large-scale machine learning is how to meaningfully and efficiently approximate a large matrix. The number of matrix entries for these large-scale problems can range from tens of thousands to millions, making it difficult to perform operations on the matrix or even to store it. The Nyström approximation, which generates a low-rank approximation of the original matrix from a subset of its columns, is an effective way to solve this issue. The choice of sampling method is critical for the Nyström approximation performance, since various samples provide different approximations of the original adjacency matrix F. Figure 4.1 illustrates a general workflow of the Nyström extension with QR decomposition.

Consider a node set V with N points and a collection of functions $\omega(x_i, x_j)$. An N–dimensional symmetric positive semi–definite matrix $W \in \mathbb{R}^{N\times N}$ is defined as

$$W = \begin{bmatrix} \omega(x_1,x_1) & \dots & \omega(x_1,x_N) \\ \vdots & \ddots & \vdots \\ \omega(x_N,x_1) & \dots & \omega(x_N,x_N) \end{bmatrix}. \tag{4.7}$$

Suppose we sample k distinct points uniformly at random from N points, then the matrix $W \in \mathbb{R}^{N\times N}$ is partitioned as

$$W = \begin{bmatrix} W_{11} & W_{21}^T \\ W_{21} & W_{22} \end{bmatrix}, \tag{4.8}$$

where $W_{11} \in \mathbb{R}^{k\times k}$ indicates weights between sampling points and $W_{12} \in \mathbb{R}^{k\times(N-k)}$ denotes weights from sampling points to remaining points. The SVD of W_{11} is $W_{11} = U\Lambda_k U^T$, where U is the eigenvectors matrix whose columns are the eigenvectors of W_{11} and $\Lambda_k = diag(\lambda_1, ..., \lambda_k)$ is a diagonal matrix with the corresponding eigenvalues.

Let a pseudoinverse of W_{11} be $W_{11}^{-1} = U\Lambda_k^{-1}U^T$. We define U_W to the eigenvectors matrix whose columns are the eigenvectors of the approximation of W. According to [2], U_W has the form

$$U_W = \begin{bmatrix} U \\ W_{21}U\Lambda_k^{-1} \end{bmatrix}.$$

Then approximation of W, namely $\bar{W}$, can be written as

$$\begin{aligned}\bar{W} = U_W \Lambda_k U_W^T &= \begin{bmatrix} U \\ W_{21} U \Lambda_k^{-1} \end{bmatrix} \Lambda_k \begin{bmatrix} U \\ W_{21} U \Lambda_k^{-1} \end{bmatrix}^T \\ &= \begin{bmatrix} U \Lambda_k U^T & U \Lambda_k \Lambda_k^{-1} U^T W_{21}^T \\ W_{21} U \Lambda_k^{-1} \Lambda_K U^T & W_{21} U \Lambda_k^{-1} \Lambda_k \Lambda_k^{-1} U^T W_{21}^T \end{bmatrix} \\ &= \begin{bmatrix} U \Lambda_k U^T & W_{21}^T \\ W_{21} & W_{21} U \Lambda_k^{-1} U^T W_{21}^T \end{bmatrix} \\ &= \begin{bmatrix} W_{11} & W_{21}^T \\ W_{21} & W_{21} W_{11}^{-1} W_{21}^T \end{bmatrix}. \end{aligned} \tag{4.9}$$

Comparing (4.8) and (4.9), we obtain that W_{22} can be approximated by

$$W_{22} \approx W_{21} W_{11}^{-1} W_{21}^T.$$

Let $\mathbf{1}_k \in \mathbb{R}^{k \times 1}$ be a k dimensional unit column vector, then we defined the degree of $\bar{W}$ as

$$\begin{aligned}\begin{bmatrix} \bar{d}_1 \\ \bar{d}_2 \end{bmatrix} &= \begin{bmatrix} W_{11} & W_{21}^T \\ W_{21} & W_{21} W_{11}^{-1} W_{21}^T \end{bmatrix} \begin{bmatrix} \mathbf{1}_k \\ \mathbf{1}_{N-k} \end{bmatrix} \\ &= \begin{bmatrix} W_{11} \mathbf{1}_k + W_{21}^T \mathbf{1}_{N-k} \\ W_{21} \mathbf{1}_k + \left(W_{21} W_{11}^{-1} W_{21}^T \right) \mathbf{1}_{N-k} \end{bmatrix}. \end{aligned} \tag{4.10}$$

As can be seen, the degree of $\bar{W}$ can be computed from its first k columns (i.e., W_{11} and W_{21}) without acquiring the full matrix $\bar{W}$.

The following relation holds [7]:

$$W \approx \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} W_{11}^{-1} \begin{bmatrix} W_{11}^T & W_{21}^T \end{bmatrix}. \tag{4.11}$$

When the number of points N is large, matrix W requires a considerable amount of memory to store. The Nyström extension implicitly approximates W with W_{11} and the first k columns of W. This avoids the need to use the W directly, thus greatly reducing the computational and storage burden.

Let D and $\hat{D}$ be the diagonal degree matrices of W and $\begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} W_{11}^{-1} \begin{bmatrix} W_{11}^T & W_{21}^T \end{bmatrix}$, respectively. Then it satisfies $D \approx \hat{D}$. The normalized W can be represented as

$$\begin{aligned}\hat{W} := D^{-\frac{1}{2}} W D^{-\frac{1}{2}} &\approx \hat{D}^{-\frac{1}{2}} \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} W_{11}^{-1} \begin{bmatrix} W_{11}^T & W_{21}^T \end{bmatrix} \hat{D}^{-\frac{1}{2}} \\ &:= \begin{bmatrix} \hat{W}_{11} \\ \hat{W}_{21} \end{bmatrix} W_{11}^{-1} \begin{bmatrix} \hat{W}_{11}^T & \hat{W}_{21}^T \end{bmatrix}, \end{aligned} \tag{4.12}$$

where $\begin{bmatrix} \hat{W}_{11} \\ \hat{W}_{21} \end{bmatrix} = \hat{D}^{-\frac{1}{2}} \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix}$. We perform the QR decomposition of $\begin{bmatrix} \hat{W}_{11} \\ \hat{W}_{21} \end{bmatrix}$, i.e., $\begin{bmatrix} \hat{W}_{11} \\ \hat{W}_{21} \end{bmatrix} = \mathbf{QR}$, where $\mathbf{Q} \in \mathbb{R}^{|V| \times k}$ is orthonormal and $\mathbf{R} \in \mathbb{R}^{k \times k}$ is an upper triangular matrix. Then the matrix S_W is defined as

$$S_W := \mathbf{R} W_{11}^{-1} \mathbf{R}^T. \tag{4.13}$$

The eigendecomposition of S_W is $S_W = \Phi_S \Lambda_k \Phi_S^T$, where the columns of U_S are orthogonal. Thus, $L_{W_{sym}}$ and $L_{W_{rw}}$ have the following approximate eigendecomposition

$$\begin{aligned} L_{W_{sym}} &= I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \\ &\approx I - \begin{bmatrix} \hat{W}_{11} \\ \hat{W}_{21} \end{bmatrix} W_{11}^{-1} \begin{bmatrix} \hat{W}_{11}^T & \hat{W}_{21}^T \end{bmatrix} \\ &= I - \mathbf{QR} W_{11}^{-1} \mathbf{R}^T \mathbf{Q}^T \\ &= I - \mathbf{Q} S_W \mathbf{Q}^T \\ &= I - \mathbf{Q} \Phi_S \Lambda_k \Phi_S^T \mathbf{Q}^T \\ &= \mathbf{Q} \Phi_S \left(I - \Lambda_k \right) \Phi_S^T \mathbf{Q}^T \\ &:= U_{sym} \Sigma_{sym} U_{sym}^T, & (4.14) \\ L_{W_{rw}} &= D^{-\frac{1}{2}} L_{W_{sym}} D^{\frac{1}{2}} \approx U_{left} \left(I - \Lambda_k \right) U_{right}^T, & (4.15) \end{aligned}$$

where the eigenvectors in $U_{sym} = \mathbf{Q} \Phi_S$ are orthonormal, eigenvectors of $L_{W_{rw}}$ are $U_{left} := \hat{D}^{-\frac{1}{2}} U_{sym}$ and $U_{right} := \hat{D}^{\frac{1}{2}} U_{sym}$, and the eigenvalues of both $L_{W_{sym}}$ and $L_{W_{rw}}$ are $I - \Lambda_k$.

Generally, there are two choices of $L_{mix} = L_F + Q_H$ in this thesis, based on the following matrices F and H:

- (i) Let $F = W$ and $H = P = \frac{d_i d_j}{vol(V)}$.
- (ii) Let $B = W - P$, then $F = B^+$ and $H = B^-$.

Similar to (4.8), the null model $P \in \mathbb{R}^{N \times N}$ can be expressed as

$$P = \begin{bmatrix} P_{11} & P_{21}^T \\ P_{21} & P_{22} \end{bmatrix}, \tag{4.16}$$

where $P_{11} \in \mathbb{R}^{k \times k}$ and $P_{21} \in \mathbb{R}^{k \times (N-k)}$ are computed using (4.10).

In either instance, we can compute the symmetric normalization of L_{mix} by

$$\begin{aligned}
L_{mix_s} &= L_{F_{sym}} + Q_{H_{sym}} \\
&= I - D_F^{-\frac{1}{2}} F D_F^{-\frac{1}{2}} + I + D_H^{-\frac{1}{2}} H D_H^{-\frac{1}{2}} \\
&=: I - \hat{F} + I + \hat{H} \\
&= 2I - \begin{pmatrix} \hat{F}_{11} & \hat{F}_{21}^T \\ \hat{F}_{21} & \hat{F}_{22} \end{pmatrix} + \begin{pmatrix} \hat{H}_{11} & \hat{H}_{21}^T \\ \hat{H}_{21} & \hat{H}_{22} \end{pmatrix} \\
&= 2I - \begin{pmatrix} \hat{F}_{11} - \hat{H}_{11} & \hat{F}_{21}^T - \hat{H}_{21}^T \\ \hat{F}_{21} - \hat{H}_{21} & \hat{F}_{22} - \hat{H}_{22} \end{pmatrix} \\
&=: 2I - \begin{pmatrix} M_{FH_{11}} & M_{FH_{21}}^T \\ M_{FH_{21}} & M_{FH_{22}} \end{pmatrix} && \text{(Defines a new matrix } M_{FH}\text{)} \\
&\approx 2I - \begin{bmatrix} M_{FH_{11}} \\ M_{FH_{21}} \end{bmatrix} M_{FH_{11}}^{-1} \begin{bmatrix} M_{FH_{11}}^T & M_{FH_{21}}^T \end{bmatrix} && \text{(By (4.11))} \\
&= 2I - \mathbf{Q}_s \mathbf{R}_s M_{FH_{11}}^{-1} \mathbf{R}_s^T \mathbf{Q}_s^T && \text{(Computes QR decomposition of } \begin{bmatrix} M_{FH_{11}} \\ M_{FH_{21}} \end{bmatrix}\text{)} \\
&= 2I - \mathbf{Q}_s S_{M_{FH}} \mathbf{Q}_s^T && \text{(Let } S_{M_{FH}} := \mathbf{R}_s M_{FH_{11}}^{-1} \mathbf{R}_s^T\text{)} \\
&= 2I - \mathbf{Q}_s \Phi_{sym} \Lambda_{sym} \Phi_{sym}^T \mathbf{Q}_s^T && \text{(Eigendecomposition } S_{M_{FH}} = \Phi_{sym} \Lambda_{sym} \Phi_{sym}^T\text{)} \\
&= \mathbf{Q}_s \Phi_{sym} \left(2I - \Lambda_{sym}\right) \Phi_{sym}^T \mathbf{Q}_s^T . && \text{(4.17)}
\end{aligned}$$

We define the $\Sigma_{mix_s} := 2I - \Lambda_{sym}$ as the eigenvalues of L_{mix_a}. $U_{mix_s} := \mathbf{Q}_s \Phi_{sym}$ is a matrix whose columns are the eigenvectors of L_{mix_s}. Furthermore, the random walk of L_{mix} has the expression

$$\begin{aligned}
L_{mix_r} = L_{F_{rw}} + Q_{H_{rw}} &= I - D_F^{-1} F + I + D_H^{-1} H \\
&=: I - \tilde{F} + I + \tilde{H} \\
&= \mathbf{Q}_r \Phi_{rw} \left(2I - \Lambda_{rw}\right) \Phi_{rw}^T \mathbf{Q}_r^T && \text{(Similar to (4.17))} \\
&=: U_{mix_r} \Sigma_{mix_r} U_{mix_r}^T , && \text{(4.18)}
\end{aligned}$$

According to the idea of (4.17), its eigenvalues and eigenvectors are $2I - \Lambda_{rw}$ and $\mathbf{Q}_r \Phi_{rw}$.

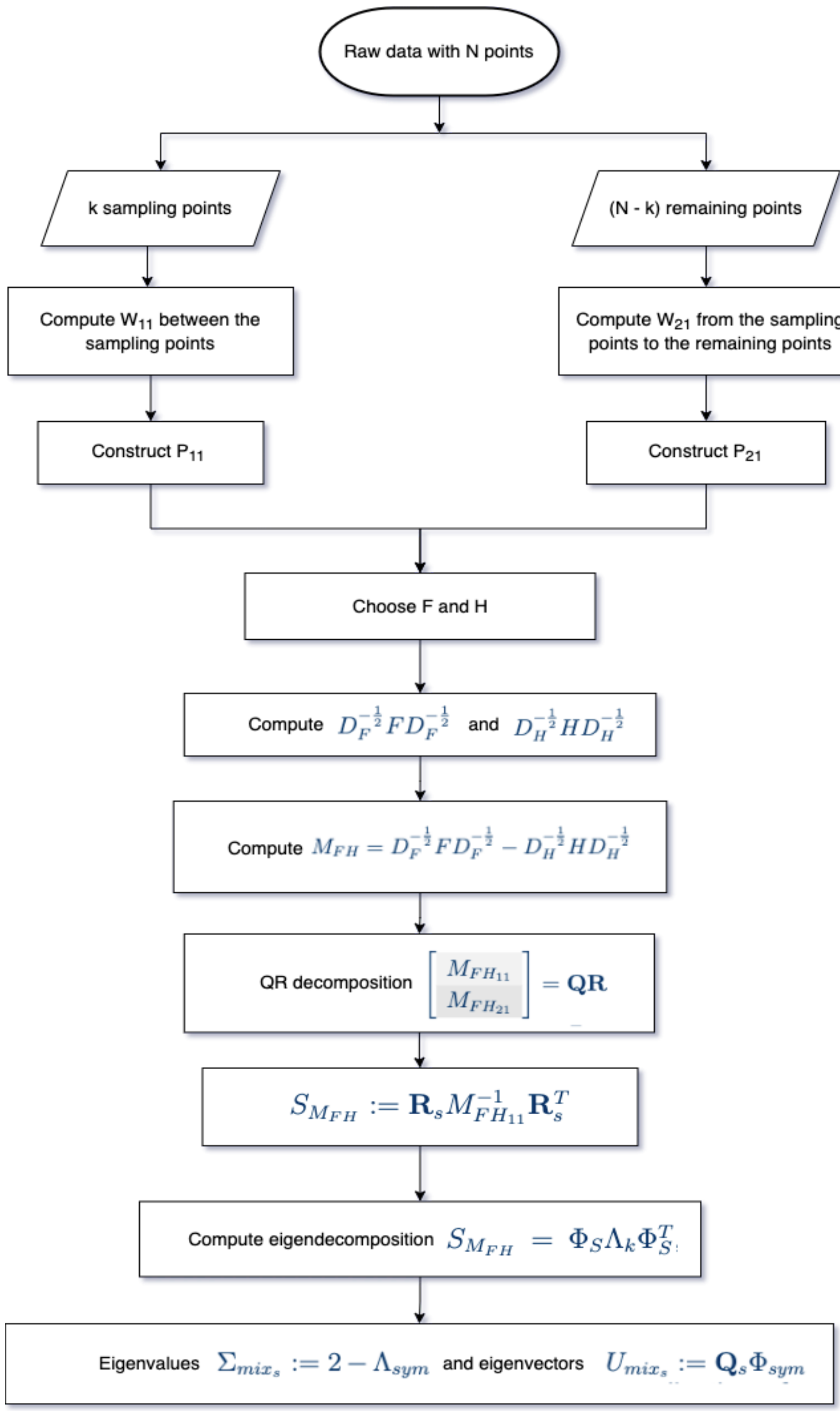

Raw data with N points
k sampling points
(N - k) remaining points
Compute W_{11} between the sampling points
Compute W_{21} from the sampling points to the remaining points
Construct P_{11}
Construct P_{21}
Choose F and H
Compute $D_F^{-\frac{1}{2}} F D_F^{-\frac{1}{2}}$ and $D_H^{-\frac{1}{2}} H D_H^{-\frac{1}{2}}$
Compute $M_{FH} = D_F^{-\frac{1}{2}} F D_F^{-\frac{1}{2}} - D_H^{-\frac{1}{2}} H D_H^{-\frac{1}{2}}$
QR decomposition $\begin{bmatrix} M_{FH_{11}} \\ M_{FH_{21}} \end{bmatrix} = \mathbf{QR}$
$S_{M_{FH}} := \mathbf{R}_s M_{FH_{11}}^{-1} \mathbf{R}_s^T$
Compute eigendecomposition $S_{M_{FH}} = \Phi_S \Lambda_k \Phi_S^T$
Eigenvalues $\Sigma_{mix_s} := 2 - \Lambda_{sym}$ and eigenvectors $U_{mix_s} := \mathbf{Q}_s \Phi_{sym}$

Figure 4.1.: The flowchart of Nyström extension with QR decomposition.

5. Numerical experiments

This section displays the results of numerical experiments for a variety of circumstances. All algorithms are implemented in Python 3.8. Since our approach uses a random starting seed in the initialization part, we run the script 20 times and provide two types of tables: (1) the best modularity and the total time for 20 runs; (2) the average modularity and the average time for each run. Furthermore, to improve performance on networks with arbitrary weights, we use L_{mix_s} in (4.17) and L_{mix_r} in (4.18).

Our method is compared to Hu's modularity MBO algorithm [32] and several well-known algorithms, including the Louvain method [3] the Clauset–Newman–Moore (CNM) approach [54], a classic Girvan–Newman (GN) method [56], and spectral clustering [63]. The functions for Louvain, CNM, GN and spectral clustering can be called directly in Python libraries as $NetworkX$ [65], $community$ [18], and $scikit-learn$ [58]. Furthermore, our experiments were tested on a MacBook Air (13-inch, 2017), where the processor is 1.8 GHz Dual-Core Intel Core i5 and the memory is 8 GB 1600 MHz DDR3. It is notable that different hardware configurations may have an impact on the running time of algorithms.

5.1. Related algorithms and evaluation metrics introduction

5.1.1. Algorithms comparison

In [32], Hu et al. interpreted modularity optimization as a minimization problem for a graph-based total variation functional. This allows the use of techniques from image processing and PDEs to be used in community detection. However, their model is non-convex, while the traditional total variation optimization is convex. This distinction restricts the available optimization tools and forces one to rely on some special initialization procedures and multiple runs of the solver to obtain reliable results. This method requires the number of clusters K as input.

The $Girvan-Newman$ (GN) approach for detecting and analyzing community structures is based on iteratively removing edges that have the highest number of shortest paths between nodes. The network is divided into smaller parts, called communities or clusters, by removing one edge at a time from the graph. The number of shortest paths across an edge is indicated by the edge betweenness score [56]. It is possible to detect which edges in a network occur more frequently by calculating the edge betweenness. Then, the edges that connect different communities are more likely to have a high edge betweenness since all shortest paths from one community to another have to transit them. Once the edges of the network with the highest betweenness are removed, the network is split into small pieces, so-called communities. At that time, community clustering will be easier. When the modularity score reaches its maximum value, this approach stops clustering, turning it into an optimization problem. This approach does not require the number of clusters K as an input, in contrast to Hu's method.

The $Clauset-Newman-Moore$ (CNM) method employs modularity as its metric and goal, which means it is constructed to maximize the modularity score $\mathcal{Q}$. It accomplishes its goal by tracking possible combinations of two communities and their impact on $\mathcal{Q}$. In each iteration, the algorithm merges the two communities that would result in the greatest improvement in $\mathcal{Q}$, and then updates the knowledge about possible combinations and impacts on $\mathcal{Q}$. The algorithm terminates when no two communities can be combined to produce a higher $\mathcal{Q}$. Similar to GN, there is no need for this method to take the number of clusters K as input.

The $Louvain$ method is an unsupervised algorithm that attempts to maximize the modularity score and performs admirably in practice. This technique has the advantage that the number of clusters and their size do not have to be entered before execution. However, Louvain has a has a big drawback in that it requires a large amount of memory to store the network. It consists mainly of two phases that are repeated iteratively.

Phase 1: Partition network greedily using modularity.

Phase 2: Agglomerate found clusters into new nodes.

In Phase 1, it has following steps:

(i) Start with every node in its own community.

(ii) Nodes are ordered randomly and we do the following for each node i. Move node i to the community of neighbor j that leads to maximum $\Delta\mathcal{Q}$. If all $\Delta\mathcal{Q} < 0$, then node i remains in its current community.

(iii) Repeat the cycle through all nodes until $\Delta\mathcal{Q} = 0$.

Phase 2 aims to create a weighted network of communities from Phase 1. It performs the following steps:

(1) Let each community c_i form a new node i.

(2) Then the edges between new nodes i and j are the sum of edges between nodes in c_i and c_j. The new node has self-loops to represent any connections between nodes of the same community. Once the new network has been established, Phase 2 is complete and Phase 1 can be reapplied to it.

The Louvain method repeats Phase 1 and Phase 2 to the resulting network, and so on until $\Delta\mathcal{Q} = 0$.

The Louvain method and the CNM algorithm are similar. In particular, they both execute Phase 1 (i). However, the first difference between them is that CNM joins the two communities that have the greatest improvement in modularity rather than doing Phase 1 (ii). The second point is that Phase 2 is not applied by CNM.

The $spectral\ clustering$ is a clustering approach based on graph theory [67, 63]. The basic idea is to cluster the data based on the eigenvectors resulting from the eigen-decomposition of the Laplacian matrix. Data points are considered as nodes in a connected graph, and clusters are found by partitioning this graph into subgraphs based on its spectral decomposition. We use spectral clustering and then process the data points using the k–means algorithm so that they can be grouped into k clusters, where k needs to be specified in advance.

Louvain, CNM and GN are greedy optimization algorithms that aim to maximize modularity. This means that they will repeat their iterations until the modularity score is maximized. This is different from the characteristic defined by the stopping condition (4.3). The stopping condition (4.3) is determined by the actual clusters and is more dependent on whether nodes are still being reassigned. Therefore, it is meaningless to apply (4.3) to the Louvain, CNM and GN method. In the following examples, Louvain, CNM, and GN are presented in a different table because they do not share a stopping condition with the MMBO schemes and Hu's method.

5.1.2. Evaluation metrics

We employ adjusted rand index (ARI) [25], purity [44], inverse purity, and normalized mutual information (NMI) [40, 42] to evaluate these methods.

Purity is a statistic that quantifies the proportion of nodes in a cluster that are members of the same class. Let $\mathcal{C} := \{C_1, ..., C_K\}$ be the set of K clusters to be evaluated and $\mathcal{C}' := \{C'_1, ..., C'_{K'}\}$ be the set of K' clusters of the ground truth. Purity can be defined as

$$Purity(\mathcal{C}, \mathcal{C}') = \sum_{i=1}^{K} \frac{1}{N} \max_{1\leq j\leq K'} |C_i \bigcap C'_j|, \tag{5.1}$$

where N is the total number of clustered nodes. Purity reaches its maximum value of 1 when each node is in a cluster containing only that single node. A high purity can be achieved by having many clusters since this metric does not penalize cluster size.

Inverse purity is simply the purity of the second partition with respect to the first one, that is, $Purity(C', C)$. Inverse purity is defined as

$$InP(\mathcal{C}, \mathcal{C}') = Purity(\mathcal{C}', \mathcal{C}) = \sum_{i=1}^{K'} \frac{1}{N} \max_{1\leq j\leq K} |C'_i \bigcap C_j|. \tag{5.2}$$

Since inverse purity has a bias opposite to the bias of purity, we can achieve a maximum inverse purity by grouping all nodes into a single cluster.

Given N nodes, and $\mathcal{X} = \{X_1, ..., X_r\}$ and $\mathcal{Y} = \{Y_1, ..., Y_s\}$ two arbitrary partitions of these nodes, let n_{ij} be the number of nodes that are in both cluster X_i and Y_j, i.e.,

	Partition Y					
Partition X		Y_1	Y_2	...	Y_s	$\sum_{j=1}^{s} n_{ij}$
	X_1	n_{11}	n_{12}	...	n_{1s}	a_1
	X_2	n_{21}	n_{22}	...	n_{2s}	a_2
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	...
	X_r	n_{r1}	n_{r2}	...	n_{rs}	a_r
	$\sum_{i=1}^{r} n_{ij}$	b_1	b_2	...	b_s	

Table 5.1.: Contingency table [12], where a_i is the sum of the i–row, that is $a_i = \sum_{j=1}^{s} n_{ij}$, and b_j is the sum of the j–column, that is $b_j = \sum_{i=1}^{r} n_{ij}$.

$n_{ij} = |X_i \bigcap Y_j|$, a_i be the sum of the i–row, that is, $a_i = \sum_{j=1}^{s} n_{ij}$, and b_j be the sum of the j–column, that is $b_j = \sum_{i=1}^{r} n_{ij}$, and let $\begin{pmatrix} N \\ 2 \end{pmatrix}$ represent the total number of possible pairs from a given set. Table 5.1 can also be shown in another way, that is, Table 5.2. In Table 5.2, TP means that a pair of nodes is placed in the same cluster

		Partition $\mathcal{Y}$	
		Pair in the same cluster	Pair in different clusters
Partition $\mathcal{X}$	Pair in the same cluster	True positive (TP)	False negative (FN)
	Pair in different clusters	False positive (FP)	True negative (TN)

Table 5.2.: Contingency Table for Comparing Partitions $\mathcal{X}$ and $\mathcal{Y}$.

in $\mathcal{X}$ and in the same cluster in $\mathcal{Y}$; FN denotes a pair of nodes that are placed in the same cluster in $\mathcal{X}$ but in different clusters in $\mathcal{Y}$; FP signifies that a pair of nodes is placed in different clusters in $\mathcal{X}$ but in the same clusters in $\mathcal{Y}$; TN means that a pair of nodes is placed in different clusters in $\mathcal{X}$ and in different clusters in $\mathcal{Y}$. Moreover, they can be calculated using the values in Table 5.1, that is,

$$TP = \sum_{i=1}^{r}\sum_{j=1}^{s} \begin{pmatrix} n_{ij} \\ 2 \end{pmatrix}, \tag{5.3}$$

$$FN = \sum_{i=1}^{r} \begin{pmatrix} a_i \\ 2 \end{pmatrix} - TP, \tag{5.4}$$

$$FP = \sum_{j=1}^{s} \begin{pmatrix} b_j \\ 2 \end{pmatrix} - TP, \tag{5.5}$$

$$TN = \begin{pmatrix} N \\ 2 \end{pmatrix} - TP - FN - FP. \tag{5.6}$$

The Rand index (RI) [60], a measure of the similarity between two data clusterings, can be calculated by

$$RI = \frac{TP+TN}{TP+FP+FN+TN} = \frac{TP+TN}{\binom{N}{2}}. \tag{5.7}$$

Intuitively, $(TP+TN)$ represents the number of agreements between $\mathcal{X}$ and $\mathcal{Y}$ while $(FP+FN)$ represents the number of disagreements between $\mathcal{X}$ and $\mathcal{Y}$. One of the drawbacks of RI is that it it does not consider the possibility of a coincidental agreement between the two partitions. The number and sizes of the clusters in each partition, as well as the total number of nodes, have a significant impact on the number of agreements of two partitions. To solve this problem, adjusted Rand (ARI) was proposed in [33], which can be calculated by

$$ARI = \frac{\binom{N}{2}(TP+TN) - [(TP+FN)(TP+FN)+(FP+TN)(FN+TN)]}{\binom{N}{2}^2 - [(TP+FN)(TP+FN)+(FP+TN)(FN+TN)]} \tag{5.8}$$

or

$$ARI = \frac{\binom{N}{2}\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]}{\frac{1}{2}\binom{N}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]}. \tag{5.9}$$

The ARI takes value in the range $[-1,1]$, where 1 represents the perfect match, and 0 indicates random labeling. Negative ARI suggests that the two partitions have less in common than what is expected from a random result.

Other indicators commonly used to evaluate the clustering quality are normalized mutual information (NMI), which is based on the concept of entropy and mutual information. The $entropy$ of a random variable is the average level of uncertainty inherent to the variable's possible outcomes, which is defined as follows based on the set of clusters $\mathcal{C}$

$$H(\mathcal{C}) = -\sum_{i=1}^{K} \frac{|C_i|}{N} log\left(\frac{|C_i|}{N}\right). \tag{5.10}$$

The joined entropy of $\mathcal{C}$ and $\mathcal{C}'$ is

$$joinH(\mathcal{C},\mathcal{C}') := -\sum_{i=1}^{K}\sum_{j=1}^{K'} \frac{|C_i \cap C'_j|}{N} log\left(\frac{|C_i \cap C'_j|}{N}\right). \tag{5.11}$$

The $mutual\ information$ (MI) evaluates the reduction in uncertainty in predicting sections of a system's outcome after observing the outcome of other parts of the system. Suppose we know the value of one of the random variables in a system. In this case, there is a corresponding reduction in uncertainty for predicting the others, and mutual information measures this reduction in uncertainty. We define the MI of $\mathcal{C}$ and $\mathcal{C}'$ as

$$\begin{aligned} MI(\mathcal{C},\mathcal{C}') &= H(\mathcal{C}) + H(\mathcal{C}') - joinH(\mathcal{C},\mathcal{C}') && (5.12)\\ &= \sum_{i=1}^{K}\sum_{j=1}^{K'} \frac{|C_i \bigcap C'_j|}{N} log\left(\frac{N|C_i \bigcap C'_j|}{|C_i||C'_j|}\right). && (5.13)\end{aligned}$$

There are many variants of the concept of mutual information. We concentrate on a well-known format, namely, $normalized\ mutual\ information$ (NMI). It is defined as

$$NMI(\mathcal{C},\mathcal{C}') = \frac{2MI(\mathcal{C},\mathcal{C}')}{H(\mathcal{C}) + H(\mathcal{C}')}. \tag{5.14}$$

5.2. Zachary's karate club

Zachary's karate club (ZKC) is a well known dataset proposed by Wayne W. Zachary in his work [73] describing the interactions in a college karate club. After Girvan and Newman [56] used the network, it became a typical example of community structure in networks.

This dataset is well known for displaying community structure when nodes in a network can be partitioned into densely linked clusters. The network illustrates the relationships among 34 members of a karate club: each node represents a person, and the links/edges reflect people communicating outside of the karate club (e.g., spending social time together, like meeting up for a coffee, separate from the karate club).

According to the clustering result in [73], the network in Zachary's karate club can be divided into two groups, centered on the 'Officer' John A (node 33), and the instructor Mr. Hi (node 0), respectively. After an argument between Mr. Hi and John A., the network accurately predicts how the karate club would split into two new clubs. It is reasonable to assume that members' interactions with other club members will influence each member's decision to join one of the two sides. The network is able to predict which faction each person would join based on the relationships between the individuals (i.e., the network).

Group leader	Nodes
Mr. Hi (node 0)	$0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 16, 17, 19, 21$
John A (node 33)	$9, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33$

Table 5.3.: Nodes clustering result according to [73].

Parameter	Value
K	$2, 3, 4, 5, 6, 7, 8$
m	Equal to K
ϵ	0
N_t	3
γ	1

Table 5.4.: Parameter setting of the MMBO schemes in ZKC for modularity optimization and clustering.

Methods	Best modularity	K for obtaining the best modularity	Total time (sec.)
GN	0.360	2	1.04
CNM	0.381	3	0.12
Louvain	0.420	4	0.14

Table 5.5.: Performance of different algorithms in ZKC in terms of modularity and runtime. Dashes indicate no calculation. The time is the total amount of time required to execute each approach 20 times.

Methods	Average modularity	Average time (sec.)
GN	0.360 (± 0.0)	0.052 (± 0.006)
CNM	0.381 (± 0.0)	0.006 (± 0.002)
Louvain	0.416 (± 0.004)	0.007 (± 0.003)

Table 5.6.: The average modularity and average running time of different algorithms in ZKC.

Methods		Best modularity	K for obtaining the best modularity	Total time (sec.)
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	0.420	4	0.16
	$L_{W_{rw}}, Q_{P_{rw}}$	0.396	4	0.18
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.401	4	0.18
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.388	4	0.19
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	0.420	4	0.20
	$L_{W_{rw}}, Q_{P_{rw}}$	0.396	4	0.20
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.401	4	0.21
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.388	4	0.22
Hu's method	$L_{W_{sym}}$	0.406	4	0.20
	$L_{W_{rw}}$	0.382	4	0.21
Spectral clustering		0.410	4	0.76
Ground truth		0.358	2	–

Table 5.7.: Compare different algorithms in modularity and running time in ZKC. Dashes indicate no calculation.

Methods		Average modularity	Average time (sec.)
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.372\ (\pm 0.048)$	$0.008\ (\pm 0.004)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.341\ (\pm 0.055)$	$0.009\ (\pm 0.003)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.367\ (\pm 0.034)$	$0.009\ (\pm 0.004)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.356\ (\pm 0.032)$	$0.010\ (\pm 0.005)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.372\ (\pm 0.048)$	$0.010\ (\pm 0.005)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.341\ (\pm 0.055)$	$0.010\ (\pm 0.006)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.363\ (\pm 0.038)$	$0.011\ (\pm 0.005)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.356\ (\pm 0.032)$	$0.011\ (\pm 0.006)$
Hu's method	$L_{W_{sym}}$	$0.360\ (\pm 0.046)$	$0.010\ (\pm 0.003)$
	$L_{W_{rw}}$	$0.310\ (\pm 0.072)$	$0.011\ (\pm 0.004)$
Spectral clustering		$0.410\ (\pm 0.0)$	$0.038\ (\pm 0.006)$

Table 5.8.: The average modularity score and average running time of algorithms for $K = 4$ in ZKC.

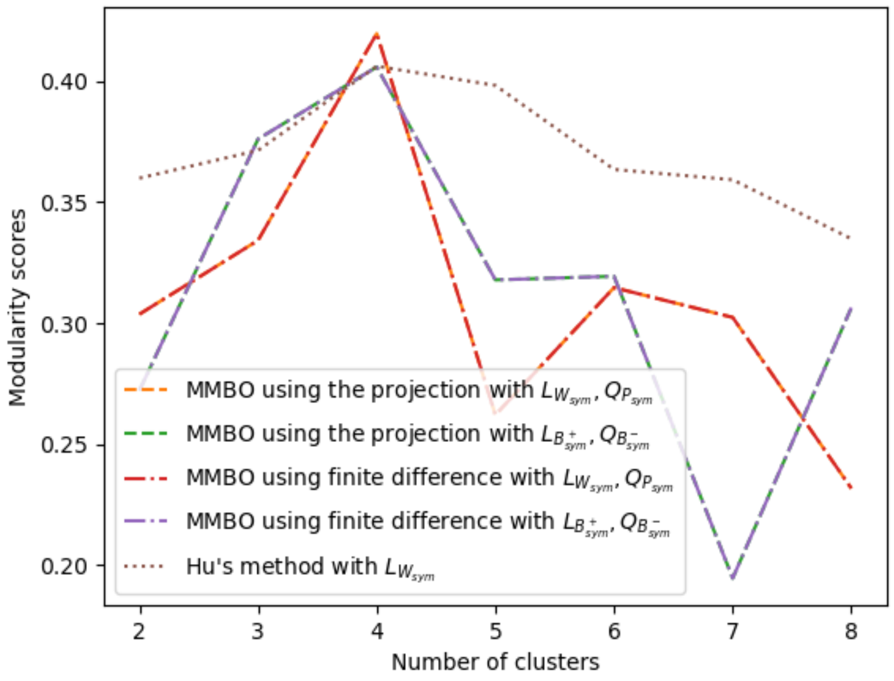
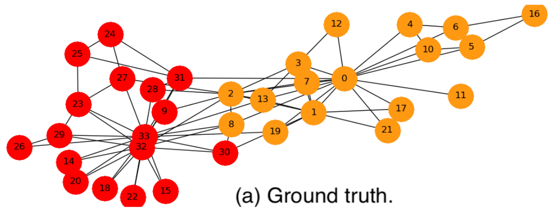
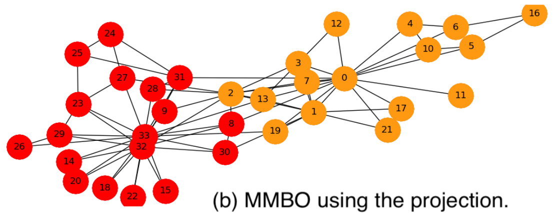


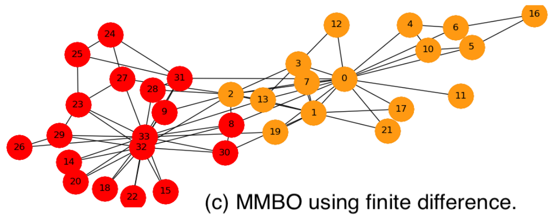
Figure 5.1.: Best modularity score versus number of clusters on ZKC. The purple and red lines cover the green dotted line and the yellow curve, respectively.

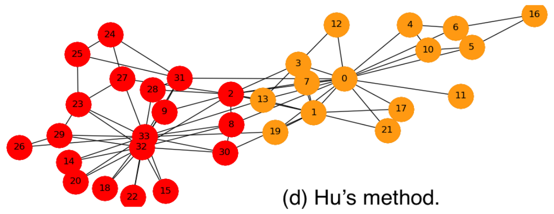
From a clustering perspective, Figure 5.2 depicts the ground truth and the performance of the various approaches in Zachary's karate club. It is important to note that $K = 2$ was chosen for the MMBO scheme, Hu's method, and spectral clustering, meaning that they find no more than two clusters. While the Louvain method, CNM algorithm, and GN method automatically select K to produce the best modularity score. In this case, K is not necessarily 2. The case presented here shows that the MMBO schemes perform better than Hu's method and spectral clustering. In particular, when compared to the ground truth, it is evident that the MMBO schemes misclassify only one node, namely node 8. On the other hand, we can see that two nodes, namely node 2 and node 8, are both misclassified by both Hu's method and spectral clustering. It is noteworthy that the Louvain method clusters all nodes into 4 clusters and provides as high modularity as the MMBO scheme using projection. Similarly, the CNM algorithm finds 3 clusters. However, this contradicts the fundamental truth that there are only 2 clusters.

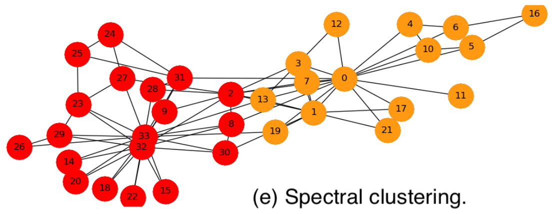
According to [24], $N_t = 3$ is the optimal value for the number of diffusion step iterations. We ran each method 20 times and then reported the approach with the highest modularity, as shown in Table 5.7. We present the results of the methods in terms of modularity score and running time. As can be seen, both Louvain and the MMBO scheme using projection with $L_{W_{sym}}, Q_{P_{sym}}$ achieve the highest modularity score. In addition, the running time of the MMBO scheme is lower than that of GN, CNM and spectral clustering. Moreover, we found that $m = 2K = 4$ is the best value for clus-

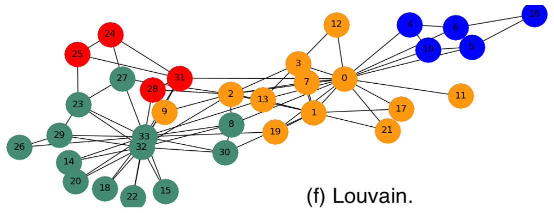

(a) Ground truth.


(b) MMBO using the projection.


(c) MMBO using finite difference.


(d) Hu's method.


(e) Spectral clustering.


(f) Louvain.

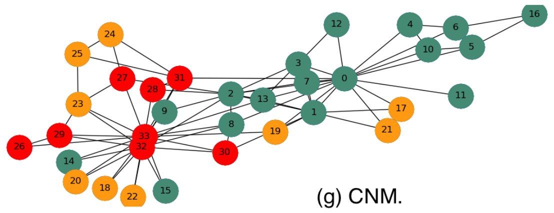

(g) CNM.

Figure 5.2.: Community detection on Zachary's karate club network with different methods.

tering, which is the same as that suggested by Louvain's algorithm. This means that this could be the natural clustering of the network.

Although Zachary's karate club graph is a simple case of the real network datasets, it provides reasonable evidence that the MMBO scheme using projection might perform better than other methods in analyzing certain real social networks.

5.3. MNIST

The MNIST handwritten digits database is a widely used dataset in computer vision and deep learning [72]. It comprises of $70,000$ 28×28 pixel images of handwritten digits ranging from 0 to 9. The dataset consists of pairs of handwritten digit images and ground truth. We aim to group different digits into distinct communities. An image is represented by a node of the graph, and to create feature vectors, we project the images onto 50 principal components as determined by PCA.

Parameter	Value
N	$70,000$
k	500
τ	0.02

Table 5.9.: Parameter settings for Nyström extension on MNIST.

An adjacency matrix W is created as follows. Firstly, for each image, the graph is built by projecting it onto 50 principal components. Secondly, we choose an appropriate weighted function. Let x_i and x_j be the coordinates of the input data node i and node j, then $||x_i - x_j||_{L_2}$ represents the L_2 distance between node i and node j. The Gaussian function $\omega(x_i, x_j)$,

$$\omega(x_i, x_j) = exp\left(-\tau||x_i - x_j||_{L_2}^2\right), \tag{5.15}$$

is a common weight function, where $\tau = \sigma^{-2}$ is known as the Gaussian kernel of variance σ^2. The weights $\omega(x_i, x_j) \neq 0$ if and only if either node i or j is one of the other's 10 nearest neighbors. In this section, the weighted matrix W is constructed by using Gaussian function $\omega(x_i, x_j)$ (5.15) with $\tau = 0.02$. Additionally, we choose the

Parameter	Value
N_t	5
ϵ	10^{-5}
γ	0.5

Table 5.10.: Parameter setting of the MMBO scheme on MNIST.

same N_t as Hu et al. in [32] to compare the results of MMBO with those of Hu et al., that is, $N_t = 5$.

The best partition obtained by the Louvain method is around $K = 125$. It is possible that the input K does not result in a partition containing K clusters since the results could contain empty clusters. As a result, the search should be extended to $K > 10$, since a greater K allows the MMBO scheme to explore partitions that maximize modularity [32]. To conduct a fair comparison of modularity, we employ the same number of clusters for the MMBO and Louvain. In other words, the Louvain approach is first used to determine the optimal partition K. Then, we utilize the MMBO schemes with K to compute the modularity score.

Methods	Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
Louvain	0.939	0.147	0.959	0.124	0.596	2188.5
CNM	0.717	0	0.115	0.052	0.01	9828.7
Spectral clustering	0.914	0.563	0.758	0.816	0.745	42368.1

Table 5.11.: Performance of different algorithms in MNIST in terms of modularity and runtime. The time is the total amount of time required to execute each approach 20 times.

Figure 5.3 displays the spectra of the first 150 eigenvalues, excluding the first (zero) eigenvalues, generated by the different choices of L_{mix} and $L_{W_{sym}}$. Specifically, in panel (a), the yellow dotted line and the green one overlap since the eigenvalues of $L_{W_{sym}}$ and $L_{W_{rw}}$ are identical. Additionally, the red dashed line is nearly hidden by the purple one in panel (b). Different selections of L_{mix} might result in differences in the first 50 eigenvalues, but the eigenvalues of all L_{mix} are almost identical when the index is greater than 130.

There are two main parts to the execution time of the MMBO schemes and the Hu's method: the computation of the eigenvalues and eigenvectors of L_{mix} or $L_{W_{sym}}$ and the MBO iteration steps. In practice, the choice of programming language can affect the speed, but the MBO iteration is often the most time-consuming part of the computation, as shown in Table 5.13.

We obtain the eigenvalues and eigenvectors using the Nyström extension with QR decomposition. Let $m = K$, where K is determined by the Louvain method. When $m = 125$, the size of the eigenvector matrix X_{mix} is relatively large. Table 5.13 demonstrates the amount of time necessary to calculate these two aspects using various methods. The MMBO scheme using projection requires the least amount of time to execute MBO iterations.

Tables 5.14 and 5.15 summarize the results of the MMBO scheme and other algorithms on the MNIST using $m = K = 125$ and the standard stopping criterion (4.3). We run these methods 20 times and then report the best modularity scores. Based on the highest modularity, Table 5.14 shows the results for that one run. Besides,

Methods	Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
Louvain	$0.938\ (\pm 0.001)$	$0.143\ (\pm 0.003)$	$0.958\ (\pm 0.001)$	$0.124\ (\pm 0.008)$	$0.596\ (\pm 0.003)$	$101.4\ (\pm 11.2)$
CNM	$0.717\ (\pm 0.0)$	$0\ (\pm 0.0)$	$0.115\ (\pm 0.0)$	$0.052\ (\pm 0.0)$	$0.002\ (\pm 0.0)$	$536.6\ (\pm 20.3)$
Spectral clustering	$0.895\ (\pm 0.019)$	$0.440\ (\pm 0.123)$	$0.673\ (\pm 0.085)$	$0.776\ (\pm 0.040)$	$0.685\ (\pm 0.060)$	$3127.5\ (\pm 227.1)$

Table 5.12.: The average performance of algorithms on MNIST when $K = 125$ and strict stopping criterion.

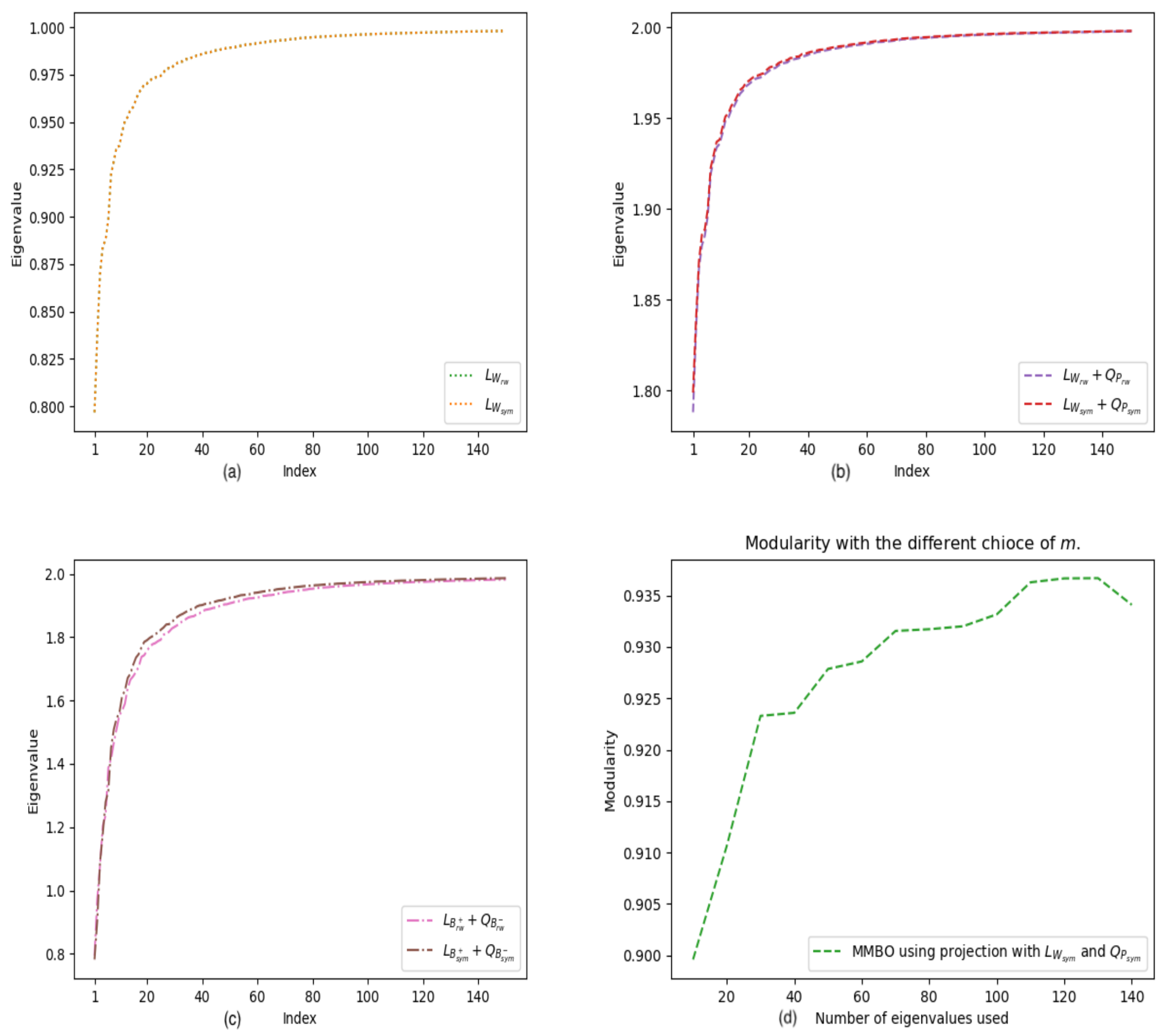


Figure 5.3.: Comparison of the spectra of different models on the MNIST. In (a), the green dotted line is covered by the yellow one. In (b), the purple curve is almost hidden by the red one.

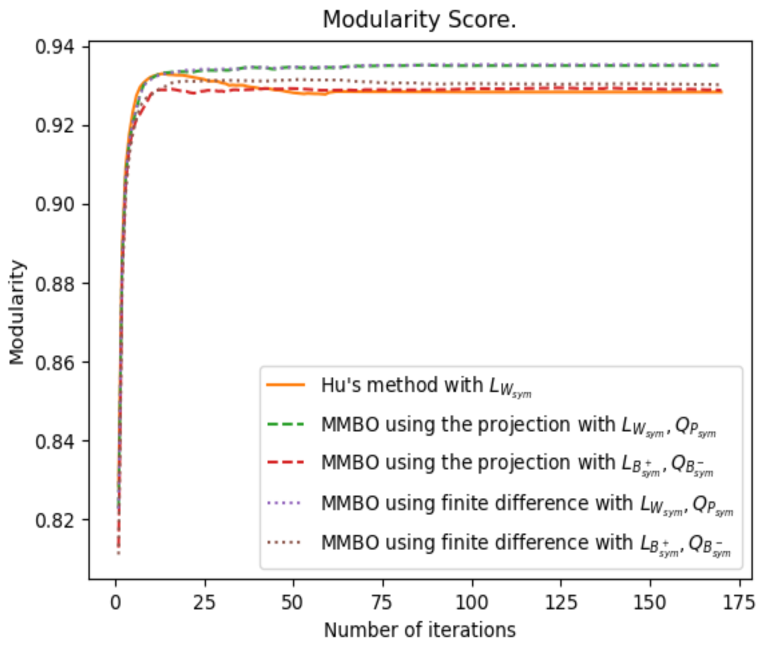


Figure 5.4.: Modularity score versus the number of iterations on the MNIST.

Methods		Average time for computing eigenvalues and eigenvectors (sec.)	Average timing of MBO iteration steps (sec.)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$10.8\ (\pm 1.5)$	$31.9\ (\pm 4.2)$	$131\ (\pm 18)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$8.3\ (\pm 1.1)$	$33.2\ (\pm 8.8)$	$126\ (\pm 36)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$10.7\ (\pm 2.0)$	$34.8\ (\pm 6.5)$	$137\ (\pm 27)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$10.3\ (\pm 2.3)$	$27.8\ (\pm 5.9)$	$115\ (\pm 25)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$10.8\ (\pm 1.5)$	$125.6\ (\pm 13.8)$	$142\ (\pm 16)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$8.3\ (\pm 1.1)$	$121.2\ (\pm 12.3)$	$139\ (\pm 14)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$10.7\ (\pm 2.0)$	$125.0\ (\pm 10.2)$	$145\ (\pm 12)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$10.3\ (\pm 2.3)$	$122.3\ (\pm 11.8)$	$143\ (\pm 13)$
Hu's method	$L_{W_{sym}}$	$8.1\ (\pm 1.2)$	$247.8\ (\pm 11.9)$	$139\ (\pm 7)$
	$L_{W_{rw}}$	$8.5\ (\pm 1.3)$	$238.4\ (\pm 12.5)$	$134\ (\pm 7)$

Table 5.13.: Average computation time on MNIST for the MMBO scheme and the Hu's method when using $m = K = 125$ and the standard stopping criterion (4.3). The number of iterations is rounded to the nearest integer.

Methods		Best modularity	ARI	Purity	Inverse purity	NMI	Total time (sec)
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	0.938	0.463	0.782	0.512	0.626	860.5
	$L_{W_{rw}}, Q_{P_{rw}}$	0.935	0.459	0.762	0.523	0.631	836.1
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.936	0.457	0.740	0.533	0.629	914.2
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.929	0.448	0.640	0.686	0.618	768.7
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	0.938	0.454	0.789	0.495	0.629	2732.9
	$L_{W_{rw}}, Q_{P_{rw}}$	0.935	0.446	0.816	0.482	0.642	2594.4
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.939	0.462	0.751	0.511	0.632	2718.1
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.934	0.453	0.777	0.532	0.639	2656.5
Hu's method	$L_{W_{sym}}$	0.926	0.433	0.689	0.530	0.601	5122.6
	$L_{W_{rw}}$	0.898	0.414	0.596	0.758	0.638	5004.9

Table 5.14.: The best modularity of algorithms on MNIST when using $m = K = 125$ and the standard stopping criterion (4.3). The time is the total amount of time required to execute each approach 20 times.

the times in Table 5.14 represent a total of 20 runs for each method, while Table 5.15 displays the average running time for each approach.

Observing the Figure 5.4, the modularity score for the MMBO schemes and Hu's approach increases dramatically in the first 25 iterations. After that, the modularity then remains stable at a high level. This trend is similar to the results shown in [32]. It is noteworthy that the MMBO scheme generally provides better modularity than Hu's approach.

However, as can be seen in Figure 5.4, the modularity improves slowly after the first 25 iterations, but continues to iterate as the stopping condition (4.3) is somewhat too strict. It is possible to choose a $modularity - related$ stopping condition: if the absolute change in modularity is less than 10^{-5}, then the stopping condition has been satisfied. The best modularity and average performance of the MMBO schemes and Hu's method are shown in Tables 5.17 and 5.18 using $m = K = 125$ and the modularity-related stopping condition. Figure 5.3(d) displays the modularity score of the MMBO scheme using projection with $L_{W_{sym}}, Q_{P_{sym}}$ versus the number of eigenvalues used under $m = K = 125$ and the modularity-related stopping condition.

As can be seen in Tables 5.13 and 5.16, the application of the modularity-related stopping condition results in fewer MBO iterations than the standard stop condition (4.3), drastically reducing the total running time. However, the modularity-related stopping condition does not necessarily lead to higher average modularity. As shown in Tables 5.15 and 5.18, the performance of the MMBO scheme using projection appears to improve when the modularity-related stopping condition is used. However, for the MMBO scheme using finite difference, it might be better to use the standard stopping condition.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
	$L_{W_{sym}}, Q_{P_{sym}}$	$0.935\ (\pm 0.003)$	$0.411\ (\pm 0.052)$	$0.772\ (\pm 0.010)$	$0.469\ (\pm 0.043)$	$0.616\ (\pm 0.010)$	$43.0\ (\pm 5.7)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$0.934\ (\pm 0.001)$	$0.418\ (\pm 0.041)$	$0.757\ (\pm 0.005)$	$0.489\ (\pm 0.034)$	$0.618\ (\pm 0.013)$	$41.7\ (\pm 9.9)$
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.921\ (\pm 0.015)$	$0.482\ (\pm 0.025)$	$0.697\ (\pm 0.043)$	$0.647\ (\pm 0.114)$	$0.621\ (\pm 0.008)$	$45.7\ (\pm 8.5)$
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.898\ (\pm 0.031)$	$0.310\ (\pm 0.138)$	$0.337\ (\pm 0.303)$	$0.594\ (\pm 0.092)$	$0.553\ (\pm 0.065)$	$38.4\ (\pm 8.2)$
	$L_{W_{sym}}, Q_{P_{sym}}$	$0.936\ (\pm 0.002)$	$0.417\ (\pm 0.037)$	$0.771\ (\pm 0.018)$	$0.472\ (\pm 0.023)$	$0.623\ (\pm 0.006)$	$136.6\ (\pm 15.3)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$0.935\ (\pm 0.002)$	$0.411\ (\pm 0.028)$	$0.788\ (\pm 0.043)$	$0.460\ (\pm 0.047)$	$0.617\ (\pm 0.018)$	$129.7\ (\pm 13.4)$
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.932\ (\pm 0.007)$	$0.389\ (\pm 0.050)$	$0.739\ (\pm 0.070)$	$0.450\ (\pm 0.033)$	$0.595\ (\pm 0.038)$	$135.9\ (\pm 12.2)$
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.903\ (\pm 0.031)$	$0.358\ (\pm 0.056)$	$0.632\ (\pm 0.165)$	$0.543\ (\pm 0.117)$	$0.523\ (\pm 0.101)$	$132.8\ (\pm 14.1)$
Hu's	$L_{W_{sym}}$	$0.929\ (\pm 0.005)$	$0.469\ (\pm 0.054)$	$0.722\ (\pm 0.014)$	$0.582\ (\pm 0.082)$	$0.623\ (\pm 0.027)$	$256.1\ (\pm 13.1)$
method	$L_{W_{rw}}$	$0.926\ (\pm 0.008)$	$0.386\ (\pm 0.068)$	$0.710\ (\pm 0.077)$	$0.518\ (\pm 0.133)$	$0.598\ (\pm 0.043)$	$250.2\ (\pm 23.8)$
Spectral clustering		$0.895\ (\pm 0.019)$	$0.440\ (\pm 0.123)$	$0.673\ (\pm 0.085)$	$0.776\ (\pm 0.040)$	$0.685\ (\pm 0.060)$	$3127.5\ (\pm 227.1)$

Table 5.15.: The average performance of the comparison algorithm on MNIST when $m = K = 125$ and the standard stopping criterion.

Methods		Average time for computing eigenvalues and eigenvectors (sec.)	Average timing of MBO iteration steps (sec.)	Number of iterations
	$L_{W_{sym}}, Q_{P_{sym}}$	$10.8\ (\pm 1.5)$	$9.6\ (\pm 3.1)$	$30\ (\pm 10)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$8.3\ (\pm 1.1)$	$9.8\ (\pm 3.7)$	$30\ (\pm 12)$
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$10.7\ (\pm 2.0)$	$8.7\ (\pm 5.5)$	$33\ (\pm 21)$
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$10.3\ (\pm 2.3)$	$10.7\ (\pm 4.9)$	$36\ (\pm 16)$
	$L_{W_{sym}}, Q_{P_{sym}}$	$10.8\ (\pm 1.5)$	$35.1\ (\pm 9.3)$	$35\ (\pm 9)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$8.3\ (\pm 1.1)$	$30.3\ (\pm 8.2)$	$31\ (\pm 8)$
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$10.7\ (\pm 2.0)$	$26.3\ (\pm 10.2)$	$28\ (\pm 12)$
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$10.3\ (\pm 2.3)$	$29.3\ (\pm 11.8)$	$42\ (\pm 13)$
Hu's	$L_{W_{sym}}$	$8.1\ (\pm 1.2)$	$66.1\ (\pm 12.8)$	$30\ (\pm 6)$
method	$L_{W_{rw}}$	$8.5\ (\pm 1.3)$	$79.8\ (\pm 12.5)$	$35\ (\pm 5)$

Table 5.16.: Average computation time on MNIST for the MMBO scheme and the Hu's method when using $m = K = 125$ and the modularity-related stopping condition. The number of iterations is rounded to the nearest integer.

Methods		Best modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
	$L_{W_{sym}}, Q_{P_{sym}}$	0.939	0.459	0.836	0.475	0.649	556.1
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.937	0.437	0.831	0.405	0.634	469.8
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.937	0.519	0.798	0.604	0.632	475.2
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.936	0.433	0.783	0.643	0.612	422.5
	$L_{W_{sym}}, Q_{P_{sym}}$	0.939	0.459	0.837	0.486	0.650	923.0
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.935	0.446	0.816	0.482	0.635	776.2
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.939	0.462	0.751	0.511	0.633	743.7
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.934	0.453	0.777	0.532	0.624	795.3
Hu's	$L_{W_{sym}}$	0.935	0.449	0.763	0.544	0.629	1488.7
method	$L_{W_{rw}}$	0.934	0.454	0.787	0.615	0.641	1769.5

Table 5.17.: The best modularity of algorithms on MNIST when $m = K = 125$ and the modularity-related stopping condition. The time is the total amount of time required to execute each approach 20 times.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
	$L_{W_{sym}}, Q_{P_{sym}}$	$0.936\ (\pm 0.003)$	$0.392\ (\pm 0.067)$	$0.778\ (\pm 0.058)$	$0.442\ (\pm 0.046)$	$0.611\ (\pm 0.038)$	$20.6\ (\pm 4.6)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$0.934\ (\pm 0.003)$	$0.390\ (\pm 0.047)$	$0.769\ (\pm 0.062)$	$0.444\ (\pm 0.039)$	$0.608\ (\pm 0.026)$	$18.3\ (\pm 4.8)$
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.925\ (\pm 0.012)$	$0.427\ (\pm 0.092)$	$0.702\ (\pm 0.096)$	$0.556\ (\pm 0.048)$	$0.602\ (\pm 0.030)$	$19.7\ (\pm 7.5)$
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.884\ (\pm 0.052)$	$0.310\ (\pm 0.123)$	$0.575\ (\pm 0.208)$	$0.551\ (\pm 0.092)$	$0.474\ (\pm 0.156)$	$21.2\ (\pm 7.2)$
	$L_{W_{sym}}, Q_{P_{sym}}$	$0.936\ (\pm 0.003)$	$0.393\ (\pm 0.066)$	$0.781\ (\pm 0.056)$	$0.441\ (\pm 0.045)$	$0.614\ (\pm 0.036)$	$46.1\ (\pm 10.8)$
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	$0.934\ (\pm 0.001)$	$0.393\ (\pm 0.053)$	$0.749\ (\pm 0.067)$	$0.455\ (\pm 0.027)$	$0.608\ (\pm 0.034)$	$38.8\ (\pm 9.3)$
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.923\ (\pm 0.016)$	$0.413\ (\pm 0.049)$	$0.704\ (\pm 0.047)$	$0.533\ (\pm 0.022)$	$0.596\ (\pm 0.036)$	$37.2\ (\pm 12.2)$
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.893\ (\pm 0.041)$	$0.328\ (\pm 0.125)$	$0.574\ (\pm 0.203)$	$0.556\ (\pm 0.024)$	$0.486\ (\pm 0.153)$	$39.8\ (\pm 14.1)$
Hu's	$L_{W_{sym}}$	$0.930\ (\pm 0.005)$	$0.395\ (\pm 0.054)$	$0.749\ (\pm 0.014)$	$0.462\ (\pm 0.082)$	$0.602\ (\pm 0.027)$	$74.4\ (\pm 14)$
method	$L_{W_{rw}}$	$0.926\ (\pm 0.008)$	$0.386\ (\pm 0.068)$	$0.710\ (\pm 0.077)$	$0.518\ (\pm 0.133)$	$0.598\ (\pm 0.043)$	$88.5\ (\pm 13.8)$

Table 5.18.: The average performance of the comparison algorithm on MNIST when using $m = K = 125$ and the modularity-related stopping condition.

We found that the MMBO scheme using projection on the eigenvectors always performs better than Hu's method, CNM and spectral clustering. Moreover, the MMBO scheme using projection has as high modularity score as Louvain, but the running time is much shorter and the ARI is much larger.

5.4. Stochastic block model

The stochastic block model (SBM) [31] evolved from the study of social networks. The goal of SBM is to split nodes into separate groups, also known as blocks, so all nodes in the same block have the same connection pattern as nodes in other blocks. The stochastic block model is constructed by generating an undirected edge between each pair of nodes independently. The probability of an edge linking two nodes is solely determined by the block in which they are located. In network science, statistics, machine learning, and other fields, the stochastic block model serves as a useful benchmark for the problem of reconstructing community structure in graph data [15, 45].

Parameter	Value
Nodes	$N = 3000$
Block	$5, 10, 15$
Block Size	N / Number of blocks
Probability	Strong community structure: $p_{same} = 0.95$, $p_{diff} = 0.01$
	Weak community structure: $p_{same} = 0.3$, $p_{diff} = 0.1$

Table 5.19.: Parameter setting of SBM.

Assume that N nodes are separated into several equally-sized blocks. The connections between nodes within the same community are stronger than those between nodes belonging to different communities. The probability of connection of nodes located in the same community is p_{same}, whereas the probability of an edge between two nodes in different communities is p_{diff}.

We construct two types of SBM: strong and weak community structures. In a strong community structure, we set $p_{same} = 0.95$, and $p_{diff} = 0.01$. In contrast, in the weak community structure, the probabilities are $p_{same} = 0.3$ and $p_{diff} = 0.1$. In summary, the SBM are generated using the set of parameters in Table 5.19. An example of adjacency matrices of the strong and weak community structure at $K = 5$ is shown in Figure 5.5, where the dark indicates 1 and the white represents 0.

Figures 5.6 and 5.7 depict the first 25 eigenvalues of L_{mix_s}, L_{mix_r} and L_{sym}, including the first (zero) eigenvalues, in SBM when $K = 10, 15$. It is worth to note that almost all the eigenvalues are distinct, except for $Q_{P_{sym}}$, despite the fact that they sometimes

Parameter	Value
Clusters	$K = 5, 10, 15$
Size	Equal to block size
m	Equal to K
N_t	3
ϵ	10^{-4}
γ	1

Table 5.20.: Parameter setting of the MMBO schemes in SBM.

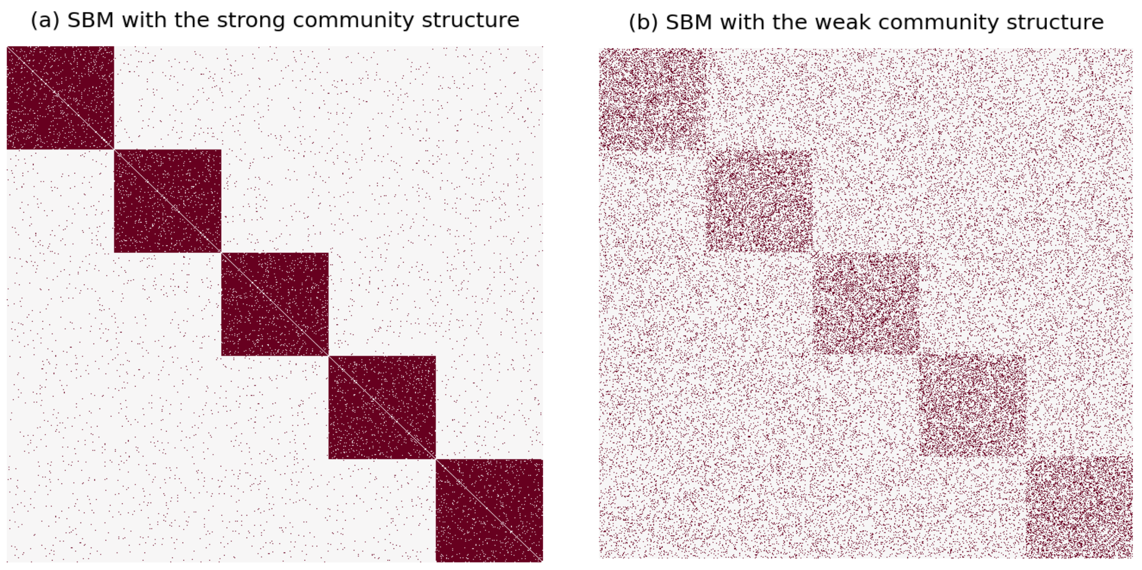


Figure 5.5.: Adjacency matrices of the strong and weak community structure at $K = 5$.

appear identical on graphs. In SBM with the strong community structure, the eigenvalues in Figures 5.6 and 5.7 seem to remain constant because the difference between them is too small. However, the change in the eigenvalues is more apparent in the weak community structure, as shown in Figure 5.7(e) and (f).

Let $\lambda_i(W)$ be the i–th eigenvalue of W. Suppose a matrix $W \in \mathbb{R}^{N\times N}$ have real eigenvalues ordered as follows:

$$\lambda_1(W) \leq \lambda_2(W) \leq ... \leq \lambda_N(W).$$

We say there exists a jump in the eigenvalues if two consecutive eigenvalues change by more than 0.01, i.e.,

$$\frac{\lambda_{i+1}(W) - \lambda_i(W)}{\lambda_i(W)} > 0.01.$$

In the case of $K = 10$, L_{mix_s} reveals a sudden jump between 9 and 10, while $L_{W_{sym}}$

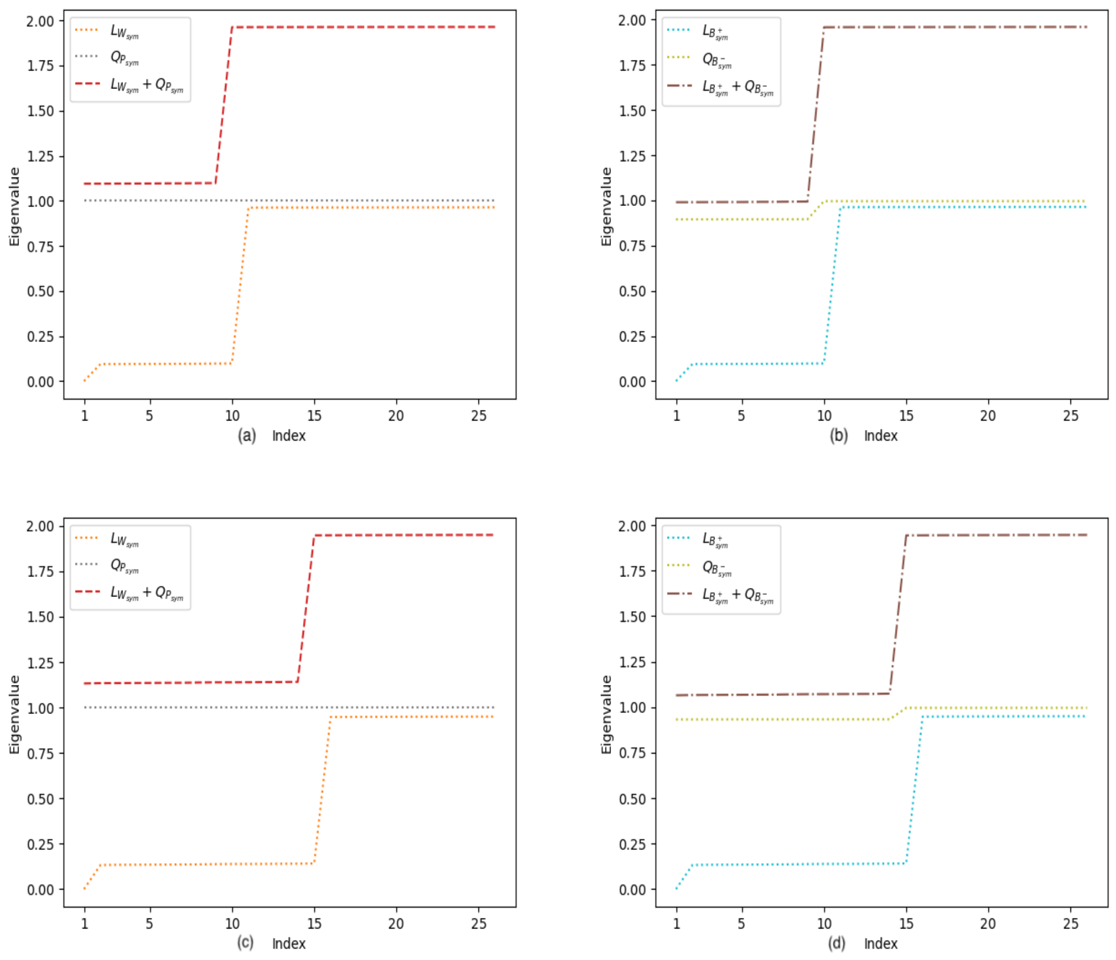


Figure 5.6.: The spectra of L_{mix_s} and L_{sym} in SBM with strong community structure, where (a) and (b) are for the case $K = 10$ and (c) and (d) are for $K = 15$.

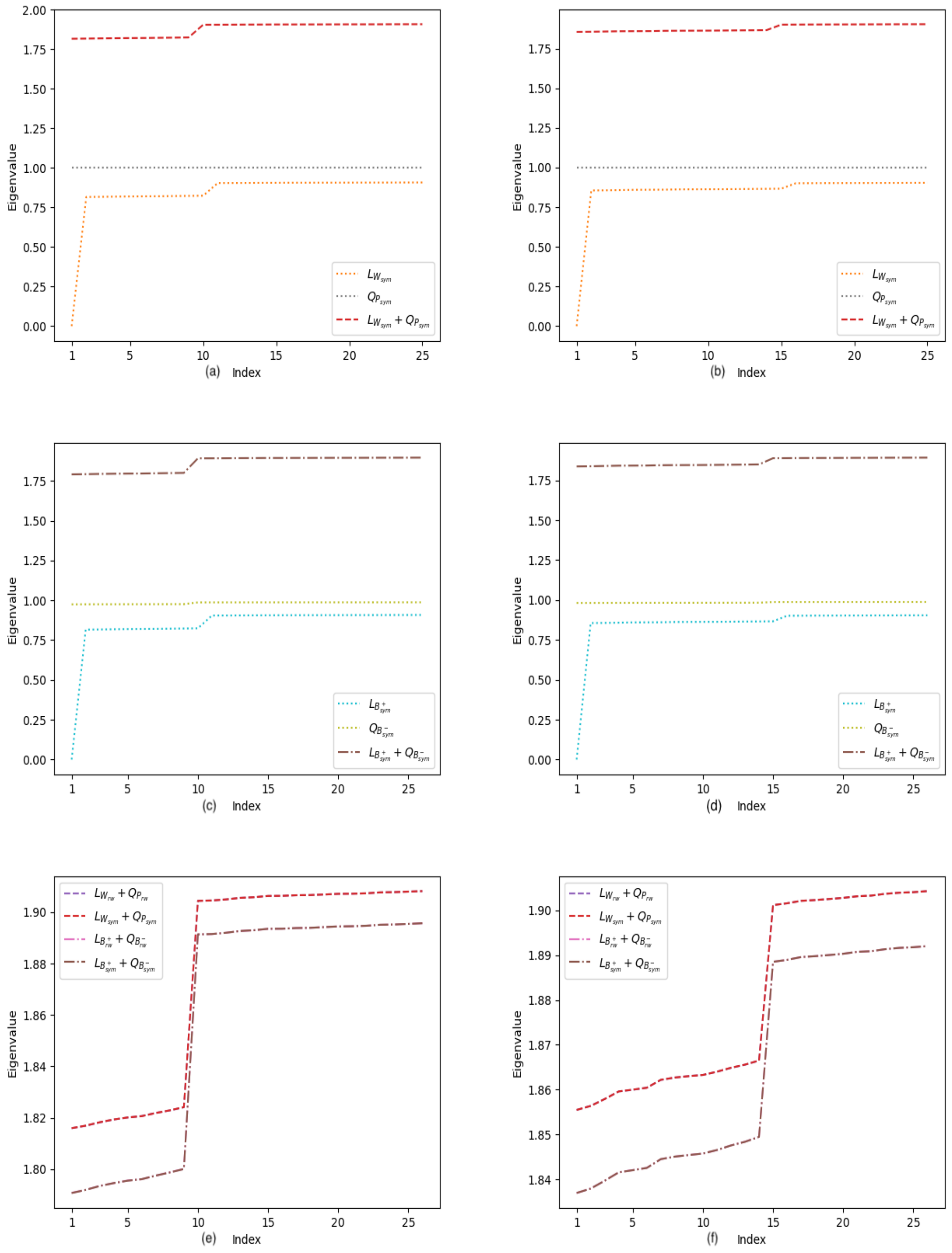


Figure 5.7.: The spectra of L_{mix} and L_{sym} in SBM with weak community structure, where the left column is the case of $K = 10$ and the right column is the case of $K = 15$. In (e) and (f), the purple and pink curves are covered by red and brown ones, respectively.

demonstrates a similar jump between 10 and 11. By observing the jump of $L_{W_{sym}}$, it shifts not only to the left but also upwards.

The theory of rank–one matrix updates[1] [8, 29] illustrates the behavior that the jumps in the eigenvalues of L_{mix_s} or L_{mix_r} shifted one to the right. Take $L_{mix_s} = L_{W_{sym}} + Q_{P_{sym}}$ as an example, Weyl's inequality[2] [22, 70] explains well the relationship of the upper and lower bounds of the eigenvalues of L_{mix_s} and the eigenvalues of $L_{W_{sym}}$ and $Q_{P_{sym}}$, i.e.,

$$\begin{aligned}
&\lambda_1(L_{W_{sym}}) + \lambda_1(Q_{P_{sym}}) \leq \lambda_1(L_{mix_s}) \leq \lambda_N(L_{mix_s}) \leq \lambda_N(L_{W_{sym}}) + \lambda_N(Q_{P_{sym}}),\\
&\lambda_i(L_{W_{sym}}) + \lambda_1(Q_{P_{sym}}) \leq \lambda_i(L_{mix_s}) \leq \lambda_i(L_{W_{sym}}) + \lambda_N(Q_{P_{sym}}).
\end{aligned}$$

The inequalities above demonstrate that the upper and lower bounds of the eigenvalues of L_{mix_s} are restricted by the eigenvalues of $L_{W_{sym}}$ and $Q_{P_{sym}}$. In addition, only one eigenvalue of $Q_{P_{sym}}$ is equal to 2, and the rest of the eigenvalues equals 1, i.e., $\lambda_i(Q_{P_{sym}}) = 1$ for $i \in \{1, ..., N-1\}$ and $\lambda_N(Q_{P_{sym}}) = 2$. Hence, it is straightforward to get

$$\lambda_i(L_{W_{sym}}) + 1 \leq \lambda_i(L_{mix_s}) \leq \lambda_i(L_{W_{sym}}) + 2.$$

This explains why the eigenvalues of $L_{W_{sym}}$ are shifted upward to L_{mix_s}.

In order to investigate the effect of the number of eigenvectors used on the final result, in Figures 5.8 and 5.9, we plot the modularity score produced by varying m. Different L_{mix} options are utilized by the MMBO scheme. When $m = 9$, all L_{mix_s} and L_{mix_r} reach the highest modularity, and Hu's method reaches a peak at $m = 10$. Combining Figures 5.6, 5.8 and 5.9, we determine that the number of eigenvectors chosen is at least the same as the number of clusters, i.e., $m \geq K$.

Tables 5.22 to 5.27 show the best modularity scores and total running times obtained for 20 runs of various approaches. Tables 5.28 to 5.33 show the average modularity of the methods and the average time for each run.

The Louvain method and the MMBO scheme using projection provide comparable modularity scores. When K is large, however, the running time of the MMBO scheme using projection is significantly lower than that of Louvain and CNM. Moreover, the MMBO schemes perform better than CNM and Hu's method in the weak community structure since it provides high modularity and is closer to ground truth.

It is worth noting that L_{mix_r} generally has a slightly shorter running time than L_{mix_s}. Moreover, L_{mix_s} and L_{mix_r} using B^+ and B^- both have greater modularity score than that of W and P.

As can be seen from these tables, spectral clustering has the highest modularity score for both strong and weak community structures, and it always assigns the proper community. With respect to the strong community structure, both Louvain and CNM provide a better modularity than MMBO, but the running time of the MMBO scheme

[1]See the appendix A.4 for more details.
[2]Also see the appendix A.4 for more details.

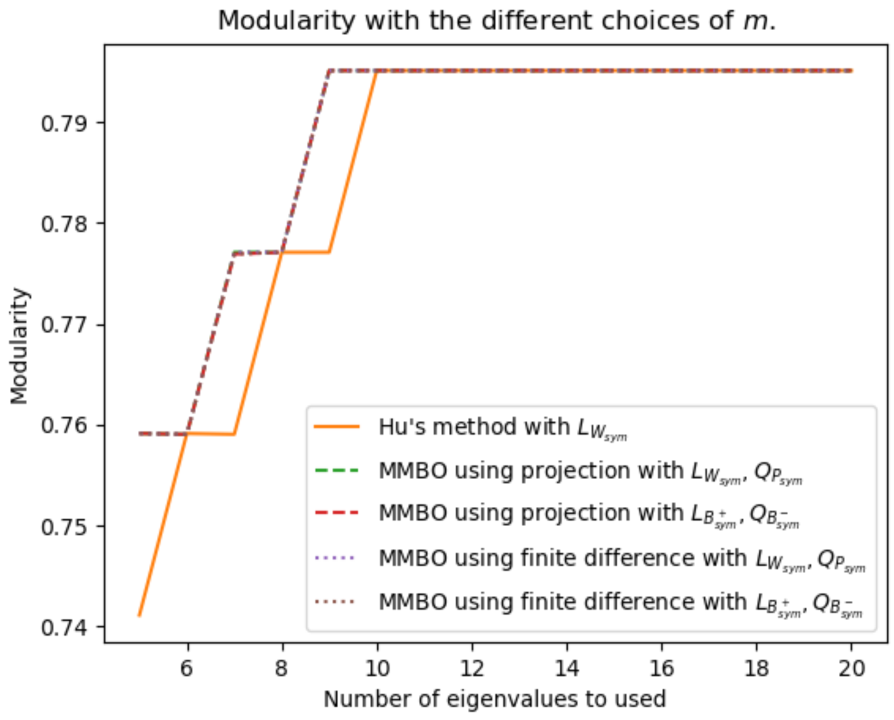


Figure 5.8.: Modularity in the number of eigenvectors used in the strong community structure of the SBM when $K = 10$. All MMBO schemes are overlapping.

is significantly faster than either. However, for the weak community structure, both MMBO schemes always provide a higher modularity than the Louvain method, especially when $K = 10, 15$. In addition, CNM has the worst performance in the weak community structure.

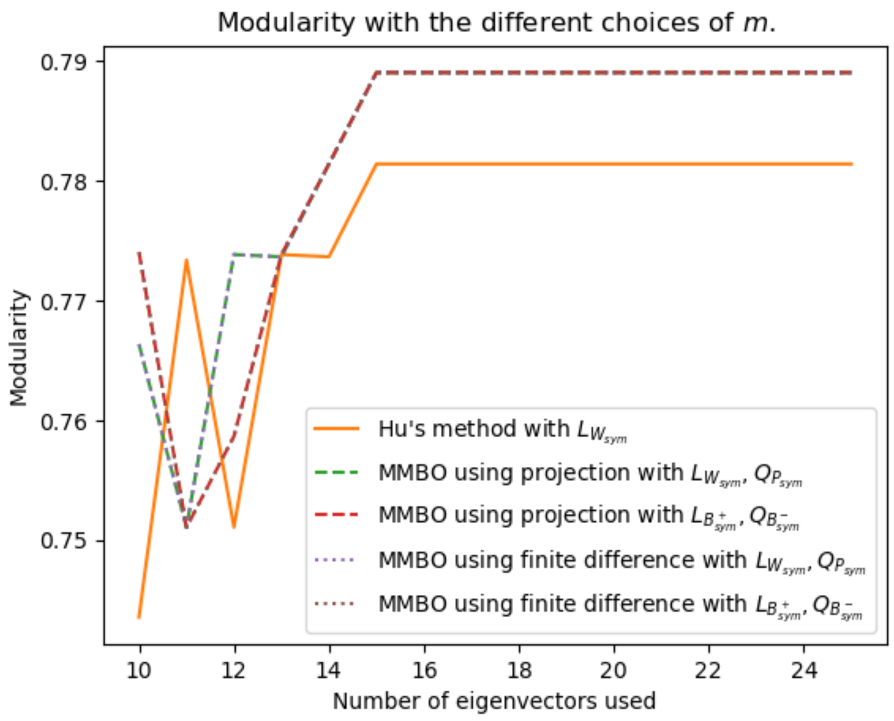


Figure 5.9.: Modularity in the number of eigenvectors used in the strong community structure of the SBM when $K = 15$. The green and brown lines cover the purple dotted line and the red curve, respectively.

Community structure	Methods	Blocks	Best modularity	ARI	Purity	Inverse purity	NMI	Time (sec)
		5	0.759	1.0	1.0	1.0	1.0	1268.5
	CNM	10	0.813	1.0	1.0	1.0	1.0	821.4
		15	0.804	1.0	1.0	1.0	1.0	713.7
		5	0.759	1.0	1.0	1.0	1.0	187.3
Strong	Louvain	10	0.813	1.0	1.0	1.0	1.0	104.0
		15	0.804	1.0	1.0	1.0	1.0	71.0
	Spectral	5	0.759	1.0	1.0	1.0	1.0	39.6
	clustering	10	0.813	1.0	1.0	1.0	1.0	28.8
		15	0.804	1.0	1.0	1.0	1.0	30.6
		5	0.227	0.995	0.998	0.998	0.991	1268.2
	CNM	10	0.107	0.363	0.385	0.920	0.627	1237.0
		15	0.068	0.227	0.2	0.981	0.657	1260.7
		5	0.228	1.0	1.0	1.0	1.0	142.9
Weak	Louvain	10	0.143	0.810	0.8	1.0	0.934	156.3
		15	0.104	0.726	0.733	1.0	0.919	232.7
	Spectral	5	0.228	1.0	1.0	1.0	1.0	36.0
	clustering	10	0.149	1.0	1.0	1.0	1.0	32.9
		15	0.109	1.0	1.0	1.0	1.0	38.4

Table 5.21.: The best modularity of algorithms on SBM, where the best modularity is obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
	$L_{W_{sym}}, Q_{P_{sym}}$	0.759	1.0	1.0	1.0	1.0	34.2	5
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.759	1.0	1.0	1.0	1.0	30.7	6
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.759	1.0	1.0	1.0	1.0	32.2	4
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.759	1.0	1.0	1.0	1.0	32.2	4
	$L_{W_{sym}}, Q_{P_{sym}}$	0.759	1.0	1.0	1.0	1.0	35.7	5
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.759	1.0	1.0	1.0	1.0	31.9	6
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.759	1.0	1.0	1.0	1.0	34.5	4
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.759	1.0	1.0	1.0	1.0	34.2	4
Hu's	$L_{W_{sym}}$	0.759	1.0	1.0	1.0	1.0	12.2	2
method	$L_{W_{rw}}$	0.759	1.0	1.0	1.0	1.0	10.6	3
Ground truth		0.759	1.0	1.0	1.0	1.0	–	–

Table 5.22.: SBM with a strong community structure at $K = 5$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
	$L_{W_{sym}}, Q_{P_{sym}}$	0.228	1.0	1.0	1.0	1.0	22.7	6
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.228	1.0	1.0	1.0	1.0	22.0	6
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.228	1.0	1.0	1.0	1.0	20.4	5
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.228	1.0	1.0	1.0	1.0	20.7	5
	$L_{W_{sym}}, Q_{P_{sym}}$	0.228	1.0	1.0	1.0	1.0	27.7	6
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.228	1.0	1.0	1.0	1.0	26.3	6
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.228	1.0	1.0	1.0	1.0	26.5	5
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.228	1.0	1.0	1.0	1.0	22.9	5
Hu's	$L_{W_{sym}}$	0.205	0.782	0.8	1.0	0.906	12.7	3
method	$L_{W_{rw}}$	0.205	0.782	0.8	1.0	0.906	14.0	3
Ground truth		0.228	1.0	1.0	1.0	1.0	–	–

Table 5.23.: SBM with a weak community structure at $K = 5$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
	$L_{W_{sym}}, Q_{P_{sym}}$	0.777	0.811	0.8	1.0	0.936	28.2	5
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.777	0.811	0.8	1.0	0.936	25.6	5
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.795	0.898	0.9	1.0	0.969	27.9	4
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.795	0.898	0.9	1.0	0.969	26.1	5
	$L_{W_{sym}}, Q_{P_{sym}}$	0.795	0.898	0.9	1.0	0.969	32.0	5
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.777	0.811	0.8	1.0	0.936	30.4	5
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.795	0.898	0.9	1.0	0.969	30.9	4
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.777	0.811	0.8	1.0	0.936	30.2	5
Hu's	$L_{W_{sym}}$	0.759	0.736	0.7	1.0	0.901	16.4	3
method	$L_{W_{rw}}$	0.759	0.736	0.7	1.0	0.901	15.9	3
Ground truth		0.813	1.0	1.0	1.0	1.0	–	–

Table 5.24.: SBM with a strong community structure at $K = 10$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
	$L_{W_{sym}}, Q_{P_{sym}}$	0.146	0.898	0.9	1.0	0.969	33.8	6
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.143	0.810	0.8	1.0	0.936	32.3	4
using the	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.146	0.898	0.9	1.0	0.969	31.9	5
projection	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.146	0.898	0.9	1.0	0.969	30.0	6
	$L_{W_{sym}}, Q_{P_{sym}}$	0.146	0.898	0.9	1.0	0.969	37.8	6
MMBO	$L_{W_{rw}}, Q_{P_{rw}}$	0.143	0.810	0.8	1.0	0.936	34.1	4
using finite	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.146	0.898	0.9	1.0	0.969	36.5	5
difference	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.146	0.897	0.9	1.0	0.969	35.8	6
Hu's	$L_{W_{sym}}$	0.143	0.810	0.8	1.0	0.936	22.9	5
method	$L_{W_{rw}}$	0.143	0.810	0.8	1.0	0.936	22.5	3
Ground truth		0.149	1.0	1.0	1.0	1.0	–	–

Table 5.25.: SBM with a weak community structure at $K = 10$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	0.781	0.819	0.8	1.0	0.946	20.3	4
	$L_{W_{rw}}, Q_{P_{rw}}$	0.774	0.770	0.8	1.0	0.939	19.7	4
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.789	0.872	0.87	1.0	0.965	21.8	4
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.781	0.819	0.8	1.0	0.946	20.5	4
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	0.781	0.819	0.8	1.0	0.946	24.7	4
	$L_{W_{rw}}, Q_{P_{rw}}$	0.774	0.770	0.8	1.0	0.939	24.2	4
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.789	0.872	0.87	1.0	0.965	26.0	4
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.781	0.819	0.8	1.0	0.946	25.4	5
Hu's method	$L_{W_{sym}}$	0.777	0.811	0.8	1.0	0.936	16.7	3
	$L_{W_{rw}}$	0.766	0.726	0.73	1.0	0.919	15.9	3
Ground truth		0.804	1.0	1.0	1.0	1.0	–	–

Table 5.26.: SBM with a strong community structure at $K = 15$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	0.106	0.818	0.8	1.0	0.946	28.0	6
	$L_{W_{rw}}, Q_{P_{rw}}$	0.105	0.770	0.73	1.0	0.927	26.6	6
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.106	0.819	0.8	1.0	0.946	27.5	5
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.106	0.818	0.8	1.0	0.946	26.1	5
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	0.106	0.818	0.8	1.0	0.946	31.6	6
	$L_{W_{rw}}, Q_{P_{rw}}$	0.105	0.770	0.73	1.0	0.927	30.9	6
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	0.106	0.819	0.8	1.0	0.946	30.4	5
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	0.106	0.818	0.8	1.0	0.946	28.5	5
Hu's method	$L_{W_{sym}}$	0.104	0.770	0.8	1.0	0.937	22.7	5
	$L_{W_{rw}}$	0.104	0.770	0.73	1.0	0.927	23.6	6
Ground truth		0.109	1.0	1.0	1.0	1.0	–	–

Table 5.27.: SBM with a weak community structure at $K = 15$. The best modularity are obtained for 20 runs.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.7\ (\pm 0.4)$	$4.6\ (\pm 0.4)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.5\ (\pm 0.4)$	$4.7\ (\pm 0.3)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.6\ (\pm 0.5)$	$4.6\ (\pm 0.6)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.6\ (\pm 0.6)$	$4.7\ (\pm 0.7)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.8\ (\pm 0.6)$	$4.6\ (\pm 0.4)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.6\ (\pm 0.6)$	$4.7\ (\pm 0.3)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.7\ (\pm 0.8)$	$4.6\ (\pm 0.6)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.721\ (\pm 0.038)$	$0.891\ (\pm 0.109)$	$0.9\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.952\ (\pm 0.048)$	$1.7\ (\pm 0.7)$	$4.7\ (\pm 0.7)$
Hu's method	$L_{W_{sym}}$	$0.691\ (\pm 0.060)$	$0.814\ (\pm 0.186)$	$0.82\ (\pm 0.18)$	$1.0\ (\pm 0.0)$	$0.911\ (\pm 0.089)$	$0.6\ (\pm 0.4)$	$2.4\ (\pm 0.4)$
	$L_{W_{rw}}$	$0.676\ (\pm 0.038)$	$0.777\ (\pm 0.263)$	$0.780\ (\pm 0.26)$	$1.0\ (\pm 0.0)$	$0.890\ (\pm 0.110)$	$0.5\ (\pm 0.4)$	$2.5\ (\pm 0.5)$
Ground truth		0.759	1.0	1.0	1.0	1.0	–	–

Table 5.28.: Average performance of approaches in SBM with strong community structure for $K = 5$.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.1\ (\pm 0.3)$	$5.5\ (\pm 0.5)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.208\ (\pm 0.020)$	$0.818\ (\pm 0.182)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.918\ (\pm 0.082)$	$1.1\ (\pm 0.3)$	$5.5\ (\pm 0.5)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.0\ (\pm 0.4)$	$5.4\ (\pm 0.4)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.0\ (\pm 0.3)$	$5.6\ (\pm 0.6)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.4\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.208\ (\pm 0.020)$	$0.818\ (\pm 0.182)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.918\ (\pm 0.082)$	$1.3\ (\pm 0.3)$	$5.5\ (\pm 0.5)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.3\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.210\ (\pm 0.018)$	$0.830\ (\pm 0.17)$	$0.84\ (\pm 0.16)$	$1.0\ (\pm 0.0)$	$0.923\ (\pm 0.077)$	$1.1\ (\pm 0.3)$	$5.6\ (\pm 0.6)$
Hu's method	$L_{W_{sym}}$	$0.198\ (\pm 0.007)$	$0.732\ (\pm 0.50)$	$0.74\ (\pm 0.06)$	$1.0\ (\pm 0.0)$	$0.872\ (\pm 0.124)$	$0.6\ (\pm 0.3)$	$3.2\ (\pm 0.2)$
	$L_{W_{rw}}$	$0.198\ (\pm 0.007)$	$0.732\ (\pm 0.50)$	$0.74\ (\pm 0.06)$	$1.0\ (\pm 0.0)$	$0.872\ (\pm 0.124)$	$0.7\ (\pm 0.2)$	$3.2\ (\pm 0.2)$
Ground truth		0.228	1.0	1.0	1.0	1.0	–	–

Table 5.29.: Average performance of approaches in SBM with weak community structure for $K = 5$.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.769\ (\pm 0.008)$	$0.784\ (\pm 0.030)$	$0.77\ (\pm 0.03)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.012)$	$1.4\ (\pm 0.5)$	$4.2\ (\pm 0.8)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.769\ (\pm 0.008)$	$0.784\ (\pm 0.030)$	$0.77\ (\pm 0.03)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.012)$	$1.3\ (\pm 0.4)$	$4.1\ (\pm 0.9)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.772\ (\pm 0.023)$	$0.792\ (\pm 0.106)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.045)$	$1.4\ (\pm 0.5)$	$4.1\ (\pm 0.1)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.772\ (\pm 0.023)$	$0.792\ (\pm 0.106)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.045)$	$1.3\ (\pm 0.4)$	$4.1\ (\pm 0.9)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.772\ (\pm 0.023)$	$0.792\ (\pm 0.106)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.045)$	$1.6\ (\pm 0.6)$	$4.0\ (\pm 1.0)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.769\ (\pm 0.008)$	$0.784\ (\pm 0.030)$	$0.77\ (\pm 0.03)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.012)$	$1.5\ (\pm 0.6)$	$4.2\ (\pm 0.8)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.772\ (\pm 0.023)$	$0.792\ (\pm 0.106)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.045)$	$1.5\ (\pm 0.6)$	$4.0\ (\pm 1.0)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.769\ (\pm 0.008)$	$0.784\ (\pm 0.030)$	$0.77\ (\pm 0.03)$	$1.0\ (\pm 0.0)$	$0.924\ (\pm 0.012)$	$1.5\ (\pm 0.6)$	$4.2\ (\pm 0.8)$
Hu's method	$L_{W_{sym}}$	$0.739\ (\pm 0.020)$	$0.673\ (\pm 0.063)$	$0.69\ (\pm 0.01)$	$1.0\ (\pm 0.0)$	$0.882\ (\pm 0.019)$	$0.8\ (\pm 0.4)$	$2.8\ (\pm 0.2)$
	$L_{W_{rw}}$	$0.739\ (\pm 0.020)$	$0.673\ (\pm 0.063)$	$0.69\ (\pm 0.01)$	$1.0\ (\pm 0.0)$	$0.882\ (\pm 0.019)$	$0.8\ (\pm 0.3)$	$2.9\ (\pm 0.1)$
Ground truth		0.813	1.0	1.0	1.0	1.0	–	–

Table 5.30.: Average performance of approaches in SBM with strong community structure for $K = 10$.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.7\ (\pm 0.5)$	$5.3\ (\pm 0.7)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.6\ (\pm 0.5)$	$5.0\ (\pm 1.0)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.6\ (\pm 0.4)$	$4.8\ (\pm 0.2)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.5\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.9\ (\pm 0.5)$	$5.5\ (\pm 0.5)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.7\ (\pm 0.6)$	$5.0\ (\pm 1.0)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.8\ (\pm 0.7)$	$4.8\ (\pm 0.2)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.142\ (\pm 0.004)$	$0.796\ (\pm 0.102)$	$0.78\ (\pm 0.12)$	$1.0\ (\pm 0.0)$	$0.929\ (\pm 0.040)$	$1.8\ (\pm 0.8)$	$4.8\ (\pm 0.2)$
Hu's method	$L_{W_{sym}}$	$0.138\ (\pm 0.005)$	$0.736\ (\pm 0.074)$	$0.72\ (\pm 0.08)$	$1.0\ (\pm 0.0)$	$0.900\ (\pm 0.036)$	$1.1\ (\pm 0.3)$	$4.5\ (\pm 0.5)$
	$L_{W_{rw}}$	$0.138\ (\pm 0.005)$	$0.736\ (\pm 0.074)$	$0.72\ (\pm 0.08)$	$1.0\ (\pm 0.0)$	$0.900\ (\pm 0.036)$	$1.0\ (\pm 0.3)$	$4.5\ (\pm 0.5)$
Ground truth		0.149	1.0	1.0	1.0	1.0	–	–

Table 5.31.: Average performance of approaches in SBM with weak community structure for $K = 10$.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.770\ (\pm 0.011)$	$0.750\ (\pm 0.069)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.921\ (\pm 0.025)$	$1.0\ (\pm 0.4)$	$4.5\ (\pm 0.5)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.768\ (\pm 0.006)$	$0.743\ (\pm 0.027)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.920\ (\pm 0.016)$	$1.0\ (\pm 0.3)$	$4.5\ (\pm 0.5)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.770\ (\pm 0.019)$	$0.750\ (\pm 0.122)$	$0.73\ (\pm 0.14)$	$1.0\ (\pm 0.0)$	$0.921\ (\pm 0.044)$	$1.1\ (\pm 0.5)$	$4.4\ (\pm 0.4)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.768\ (\pm 0.013)$	$0.743\ (\pm 0.076)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.920\ (\pm 0.026)$	$1.0\ (\pm 0.4)$	$4.4\ (\pm 0.4)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.770\ (\pm 0.011)$	$0.750\ (\pm 0.069)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.921\ (\pm 0.025)$	$1.2\ (\pm 0.5)$	$4.4\ (\pm 0.4)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.768\ (\pm 0.006)$	$0.743\ (\pm 0.027)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.920\ (\pm 0.016)$	$1.2\ (\pm 0.5)$	$4.5\ (\pm 0.5)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.770\ (\pm 0.019)$	$0.750\ (\pm 0.122)$	$0.73\ (\pm 0.14)$	$1.0\ (\pm 0.0)$	$0.921\ (\pm 0.044)$	$1.3\ (\pm 0.6)$	$4.4\ (\pm 0.4)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.768\ (\pm 0.013)$	$0.743\ (\pm 0.076)$	$0.73\ (\pm 0.07)$	$1.0\ (\pm 0.0)$	$0.920\ (\pm 0.026)$	$1.3\ (\pm 0.5)$	$4.4\ (\pm 0.4)$
Hu's method	$L_{W_{sym}}$	$0.762\ (\pm 0.015)$	$0.716\ (\pm 0.095)$	$0.69\ (\pm 0.11)$	$1.0\ (\pm 0.0)$	$0.907\ (\pm 0.029)$	$0.8\ (\pm 0.3)$	$2.9\ (\pm 0.1)$
	$L_{W_{rw}}$	$0.762\ (\pm 0.004)$	$0.716\ (\pm 0.010)$	$0.69\ (\pm 0.04)$	$1.0\ (\pm 0.0)$	$0.907\ (\pm 0.012)$	$0.8\ (\pm 0.2)$	$2.8\ (\pm 0.1)$
Ground truth		0.804	1.0	1.0	1.0	1.0	–	–

Table 5.32.: Average performance of approaches in SBM with strong community structure for $K = 15$.

Methods		Modularity	ARI	Purity	Inverse purity	NMI	Time (sec)	Number of iterations
MMBO using the projection	$L_{W_{sym}}, Q_{P_{sym}}$	$0.104\ (\pm 0.002)$	$0.744\ (\pm 0.074)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.916\ (\pm 0.030)$	$1.4\ (\pm 0.5)$	$6.2\ (\pm 0.2)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.104\ (\pm 0.002)$	$0.742\ (\pm 0.028)$	$0.71\ (\pm 0.02)$	$1.0\ (\pm 0.0)$	$0.915\ (\pm 0.012)$	$1.3\ (\pm 0.4)$	$5.7\ (\pm 0.3)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.104\ (\pm 0.002)$	$0.744\ (\pm 0.075)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.916\ (\pm 0.030)$	$1.4\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.104\ (\pm 0.002)$	$0.742\ (\pm 0.076)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.915\ (\pm 0.034)$	$1.3\ (\pm 0.4)$	$5.1\ (\pm 0.1)$
MMBO using finite difference	$L_{W_{sym}}, Q_{P_{sym}}$	$0.104\ (\pm 0.002)$	$0.744\ (\pm 0.074)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.916\ (\pm 0.030)$	$1.6\ (\pm 0.5)$	$6.3\ (\pm 0.3)$
	$L_{W_{rw}}, Q_{P_{rw}}$	$0.104\ (\pm 0.002)$	$0.742\ (\pm 0.028)$	$0.71\ (\pm 0.02)$	$1.0\ (\pm 0.0)$	$0.915\ (\pm 0.012)$	$1.5\ (\pm 0.5)$	$5.7\ (\pm 0.3)$
	$L_{B^+_{sym}}, Q_{B^-_{sym}}$	$0.104\ (\pm 0.002)$	$0.744\ (\pm 0.075)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.916\ (\pm 0.030)$	$1.5\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
	$L_{B^+_{rw}}, Q_{B^-_{rw}}$	$0.104\ (\pm 0.002)$	$0.742\ (\pm 0.076)$	$0.71\ (\pm 0.09)$	$1.0\ (\pm 0.0)$	$0.915\ (\pm 0.034)$	$1.4\ (\pm 0.5)$	$5.1\ (\pm 0.1)$
Hu's method	$L_{W_{sym}}$	$0.103\ (\pm 0.001)$	$0.723\ (\pm 0.047)$	$0.69\ (\pm 0.011)$	$1.0\ (\pm 0.0)$	$0.909\ (\pm 0.028)$	$1.1\ (\pm 0.3)$	$5.6\ (\pm 0.6)$
	$L_{W_{rw}}$	$0.103\ (\pm 0.001)$	$0.719\ (\pm 0.051)$	$0.69\ (\pm 0.04)$	$1.0\ (\pm 0.0)$	$0.907\ (\pm 0.020)$	$1.2\ (\pm 0.4)$	$5.5\ (\pm 0.5)$
Ground truth		0.109	1.0	1.0	1.0	1.0	–	–

Table 5.33.: Average performance of approaches in SBM with weak community structure for $K = 15$.

6. Conclusion and Discussion

In this thesis, we have developed an improved approach for modularity optimization, namely the MMBO scheme using projection. We reformulate the modularity optimization as minimizing the total variation and the signless total variation on a graph. When working with large networks, one of the most important aspects of our techniques is computing the leading eigenvalues and eigenvectors, which allows us to use the Nyström extension described in [7] with QR decomposition. In addition, we have proposed MBO techniques that can handle large datasets (such as the MNIST dataset) while requiring low computational costs. Numerical experiments show that our method is not only competitive in terms of modularity scores but also significantly faster than both the method of Hu et al. and of Louvain.

Our research highlights several potential direction for future investigation. A signed graph is a graph where every edge has a positive or negative sign. Signed graphs can be employed to describe positive and negative interactions between groups of objects. For instance, positive edges represent connections such as friendship, while negative edges express enmity. Considering that currently many node classification methods focus on unsigned graphs, the first interesting potential direction is to investigate how to adapt methods from unsigned graphs to signed graphs. Second, it would be beneficial to investigate how well the MBO method performs in directed weighted graphs. A community in directed graphs, is defined as a collection of nodes having more edges within a cluster than between different clusters. Many algorithms were developed for undirected weighted graphs for applications such as clustering and community detection. Some complications arise when dealing with directed weighted graphs. For example, adjacency matrices of directed weighted graphs are generally asymmetric, making the spectral analysis of directed weighted graphs more complex than that of undirected weighted graphs. Therefore, extending the proposed MBO scheme to directed weighted graphs is a promising research topic for the future.

Acknowledgment. I would like to express my deepest appreciation to my supervisors, Dr. Yves van Gennip and Prof. Dr. Volker John, for their invaluable patience and guidance. I know how much time and effort they put into coaching me, not just to help me complete the work on time, but also ensured the quality of each step. They always considered this work as my own, but also steered me in the right direction whenever they felt I needed it.

A. Appendix

In this section, we give the definition of Γ–convergence and proofs of theorems stated earlier in the thesis.

A.1. The zero eigenvalue

Theorem 1 The number of zero eigenvalues of the Laplacian L (i.e., the multiplicity of the 0 eigenvalue) equals the number of connected components of the graph G.

Proof. Given a graph $G = (V, E)$ with a node set V and an edges set E. Let k be the number of connected components in G, which corresponds to the division of V into disjoint sets $A_1, A_2, ..., A_k$.

First, we show that the number of zero eigenvalues is at least the number of connected components k. We defined k vectors $\left(u^{(1)}, ..., u^{(k)}\right)$ by

$$u_i^{(l)} = \begin{cases} 1 & \text{if } i \in A_l, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1}$$

where $u_i^{(l)}$ is the i–th component of $u^{(l)}$.

Since A_i and A_j are disjoint, $\left\langle u^{(i)}, u^{(j)} \right\rangle = 0$ for $i \neq j$. Besides, $Lu^{(i)} = 0$ for all $i \in \{1, 2, ..., k\}$. Thus, there is a set of k orthonormal vectors that are all eigenvectors of L corresponding to the eigenvalue 0, i.e., the number of zero eigenvalues of L is at least k.

Next, we prove that the number of zero eigenvalues is at most k. Let $u = \left(u^{(1)}, ..., u^{(k)}\right)$ be a vector–valued function. By (2.7), we know that $u^T Lu = 0$ if and only if $u_i = u_j$ for $(i, j) \in E$, which means u is constant on every connected component. So if $Lu = 0$, it is straightforward to have $u^T Lu = 0$, which means there exist scalars $\varrho_1, ..., \varrho_k$ such that

$$u = \sum_{i=1}^{k} \varrho_i u^{(i)}, \tag{A.2}$$

where (A.2) indicates that each eigenvector of L that corresponds to the eigenvalue 0 is contained in the subspace spanned by $\{u^{(1)}, ..., u^{(k)}\}$. Thus, there are at most k linearly independent eigenvectors corresponding to the eigenvalue 0. $\square$

A.2. Γ–convergence

Definition. (Γ–convergence) Let X be a metric space and let $F_n : X \to \mathbb{R} \bigcup \{\pm\infty\}$ be a sequence of functionals on X. Then, F_n are said to Γ-converge to the functional $F : X \to \mathbb{R} \bigcup \{\pm\infty\}$ if the following two conditions hold:

- Lower bound condition: For every convergent sequence $u_n \to u$, it satisfies that

$$F(u) \leq \liminf_{n\to\infty} F_n(u_n). \tag{A.3}$$

- Upper bound condition: There exists a sequence $\{u_n\}_{n=1}^{\infty}$ such that

$$F(u) \geq \limsup_{n\to\infty} F_n(u_n). \tag{A.4}$$

A.3. The choice of timestep

In [4], Boyd et al. provided a method to automatically determine timestep dt in the MBO scheme. In particular, recursive implementations will greatly benefit from this approach. When the size of a graph gets smaller, the suitable timestep empirically decreases, and it would be impractical for a person to manually check at each recursion step.

Proposition. (Lower bounds on dt) Let U^0 be the partition matrix of some partition. If U satisfies $\frac{d}{dt}U = -\frac{1}{2}(L_F + Q_H)U$ with initial U^0, then the following inequalities holds:

$$||U(dt) - U^0||_\infty \leq exp\left(2(\gamma+1)d_{max}dt\right),$$

where γ is the resolution parameter in modularity Q and d_{max} is the largest degree of D. In particular, for the number of clusters $K = 2$, this bound implies that if the timestep dt in the MBO scheme satisfies

$$dt < \frac{log2}{2(\gamma+1)d_{max}} \approx \frac{0.15}{(\gamma+1)d_{max}}, \tag{A.5}$$

then the MBO iteration is stationary.

Although (A.5) is restricted to $K = 2$, the authors consider $K = 2$ to be the worst case. Therefore, we utilized the time step restriction regardless of K.

Proposition. (Upper bounds on dt) Given U^0 be the partition matrix of some partition. Let U satisfies $\frac{d}{dt}U = -\frac{1}{2}(L_F+Q_H)U$ with initial U^0, then we have the following bounds:

$$||U||_{L_2} \leq exp(-dt\lambda_1)||U^0||_{L_2},$$

where λ_1 is the smallest eigenvalue of $\frac{1}{2}(L_F+Q_H)$. Furthermore, let $\frac{1}{2}(L_F+Q_H)$ be nonsingular. Then for any $\varepsilon > 0$, we have $||U(dt)||_\infty < \varepsilon$ if

$$dt > \frac{1}{\lambda_1} log \left(\frac{||U^0||_{L_2}}{\varepsilon} \right). \tag{A.6}$$

In practice, dt is determined by the geometric mean of (A.5) and (A.6). We always chose the parameter $\varepsilon = 1$ in the numerical test.

A.4. Weyl's inequality and rank–one matrix updates

Since an integer or real matrix is Hermitian if and only if it is symmetric, we obtain that L_{mix} is Hermitian. Thus, L_{mix} has real eigenvalues. We write an eigenvalue $\lambda_k(A)$ to represent the the k–th largest eigenvalue of A.

Weyl's inequality states a bound on the eigenvalues of the sum of matrices. The theory of rank–one matrix updates illustrates why the eigenvalues of the original matrix and the rank–one update matrix are interleaved.

Theorem. (Weyl's inequality [70]) Let A and B be $n \times n$ Hermitian matrices, and matrix $C := A + B$. Suppose A, B and C have real eigenvalues ordered as follows:

$$\begin{gathered} A : \lambda_1(A) \leq \lambda_2(A) \leq ... \leq \lambda_n(A), \\ B : \lambda_1(B) \leq \lambda_2(B) \leq ... \leq \lambda_n(B), \\ C : \lambda_1(C) \leq \lambda_2(C) \leq ... \leq \lambda_n(C). \end{gathered}$$

Then the following inequalities hold:

$$\lambda_i(A) + \lambda_1(B) \leq \lambda_i(C) \leq \lambda_i(A) + \lambda_n(B), \qquad i = 1, ..., n. \tag{A.7}$$

and in particular

$$\lambda_1(A) + \lambda_1(B) \leq \lambda_1(C) \leq \lambda_n(C) \leq \lambda_n(A) + \lambda_n(B). \tag{A.8}$$

Particularly, if B is positive definite, i.e., all eigenvalues of B are positive, then above inequalities result in

$$\lambda_i(A) \leq \lambda_i(C). \qquad i = 1, ..., n. \tag{A.9}$$

Given a real symmetric $n \times n$ matrix M with $M = QDQ^T$, where $Q \in \mathbb{R}^{n\times n}$ is an orthonormal matrix (i.e., $QQ^T = Q^TQ = I_{n\times n}$) whose columns are the eigenvectors of

M, and $D \in \mathbb{R}^{n\times n}$ is diagonal matrix containing corresponding eigenvalues, ordered as $\lambda_1(D) \leq ... \leq \lambda_n(D)$. Let $u \in \mathbb{R}^{n\times 1}$ be a vector and $z := Q^{-1}u$, then the rank–one update matrix $\tilde{M}$

$$\tilde{M} = M + uu^T = Q(D + zz^T)Q^T := QD_{update}Q^T.$$

We are interested in the eigenvalues of $\tilde{M}$. So, the issue is the eigenvalue decomposition of matrix D_{update}.

Theorem. (Rank–one matrix updates in [8]) With of the above notation and consider D_{update}. Let $\lambda_1(D) \leq ... \leq \lambda_n(D)$ be the eigenvalues of D, $\lambda_1(D_{update}) \leq ... \leq \lambda_n(D_{update})$ be the eigenvalues of D_{update}. Then

$$\lambda_i(D_{update}) = \lambda_i(D) + \mu_i, \qquad i = 1, ..., n \tag{A.10}$$

where $\mu_i \in [0, 1]$ and $\sum_i^n \mu_i = 1$. Furthermore, it holds that

$$\lambda_1(D) \leq \lambda_1(D_{update}) \leq \lambda_2(D) \leq ... \leq \lambda_n(D) \leq \lambda_n(D_{update}). \tag{A.11}$$

The Weyl inequalities provide a wealth of information regarding the low rank perturbations, and (A.11) restrict each eigenvalue of $\tilde{M}$ to the interval bounded by two consecutive eigenvalues of M. Take $L_{mix_s} = L_{W_{sym}} + Q_{P_{sym}}$ as an example, we know

$$\begin{aligned} L_{mix_s} &= L_{W_{sym}} + Q_{P_{sym}} \\ &= I - D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}} + I + D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}} \\ &= 2I - D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}} + D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}, \end{aligned}$$

and $H := 2 - D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}}$ is symmetric and so Hermitian. The eigenvalues of H are $\lambda_1(H) \leq ... \leq \lambda_n(H)$. For simplicity, we write $K := D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}$. Since we use the Newman–Girvan null model (i.e., $P_{ij} = \frac{d_i d_j}{vol(V)}$), it is straightforward to get $K_{ij} = \frac{d_i^{\frac{1}{2}} d_j^{\frac{1}{2}}}{vol(V)}$. With the theory of the rank–one update matrix, we can rewrite K as

$$K = D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}} = zz^T,$$

where $z_i = \left(\frac{d_i}{vol(V)}\right)^{\frac{1}{2}}$, $i = 1, ..., n$. Then L_{mix} is represented as

$$L_{mix} = H + K = H + zz^T.$$

By (A.11), it is straightforward to have

$$\lambda_i(H) \leq \lambda_i(L_{mix}) \leq \lambda_{i+1}(H). \qquad i \in \{1, ..., n-1\} \tag{A.12}$$

We say there exists a jump in the eigenvalues if two consecutive eigenvalues change by more than 0.01, i.e.,

$$\frac{\lambda_{i+1}(W) - \lambda_i(W)}{\lambda_i(W)} > 0.01. \tag{A.13}$$

Consider L_{mix_s} in the experiments of SBM. (A.12) explains the jumps in the eigenvalues of L_{mix_s} shifting from one to the right, i.e., if the sudden jump in L_{mix_s} occurs between the 9–th and 10–th eigenvalues, it follows that

$$\lambda_9(H) \leq \lambda_9(L_{mix_s}) \leq \lambda_{10}(H) \leq \lambda_{10}(L_{mix_s}) \leq \lambda_{11}(H). \tag{A.14}$$

It indicates that there may be a jump between the 9–th and 10–th or between the 10–th and 11–th eigenvalues of H, or even possibly no jump at all. More specifically, if there exists a jump in L_{mix_s}, then according to (A.14) and (A.13) we have

$$\begin{aligned} \lambda_{10}(L_{mix_s}) - \lambda_9(L_{mix_s}) &\leq \lambda_{11}(H) - \lambda_9(H) \\ 0.01 < \frac{\lambda_{10}(L_{mix_s}) - \lambda_9(L_{mix_s})}{\lambda_9(L_{mix_s})} &\leq \frac{\lambda_{11}(H) - \lambda_{10}(H) + \lambda_{10}(H) - \lambda_9(H)}{\lambda_9(L_{mix_s})} \\ &= \frac{\lambda_{11}(H) - \lambda_{10}(H)}{\lambda_9(L_{mix_s})} + \frac{\lambda_{10}(H) - \lambda_9(H)}{\lambda_9(L_{mix_s})} \\ &\leq \frac{\lambda_{11}(H) - \lambda_{10}(H)}{\lambda_9(L_{mix_s})} + \frac{\lambda_{10}(H) - \lambda_9(H)}{\lambda_9(H)}, \end{aligned}$$

which is not guaranteed that $\frac{\lambda_{10}(H)-\lambda_9(H)}{\lambda_9(H)} > 0.01$ or $\frac{\lambda_{11}(H)-\lambda_{10}(H)}{\lambda_9(L_{mix_s})} \geq \frac{\lambda_{11}(H)-\lambda_{10}(H)}{\lambda_{10}(H)} > 0.01$. Therefore, it is likely that there is no jump in $\lambda(H)$, even if $\lambda(L_{mix_s})$ satisfies (A.13).

In addition, (A.8) explains well the relationship between the upper and lower bounds of the eigenvalues of L_{mix} and the eigenvalues of $L_{W_{sym}}$ and $Q_{P_{sym}}$. Specifically, we have

$$\lambda_1(L_{W_{sym}}) + \lambda_1(Q_{P_{sym}}) \leq \lambda_1(L_{mix_s}) \leq \lambda_n(L_{mix_s}) \leq \lambda_n(L_{W_{sym}}) + \lambda_n(Q_{P_{sym}}). \tag{A.15}$$

The above inequality shows that the upper and lower bounds of the eigenvalues of L_{mix_s} are related to the sum of the eigenvalues of $L_{W_{sym}}$ and $Q_{P_{sym}}$. In addition, only one eigenvalue of $Q_{P_{sym}}$ is equal to 2, and the rest of the eigenvalues equals 1, i.e., $\lambda_i(Q_{P_{sym}}) = 1$ for $i \in \{1, ..., n-1\}$ and $\lambda_n(Q_{P_{sym}}) = 2$. Consequently, (A.15) can be rewritten as

$$\lambda_i(L_{W_{sym}}) + 1 \leq \lambda_i(L_{mix_s}) \leq \lambda_i(L_{W_{sym}}) + 2.$$

This explains the eigenvalues of $L_{W_{sym}}$ are shifted upward to L_{mix_s}.

B. References

[1] **Alex Arenas, Alberto Fernandez, and Sergio Gomez**. “Analysis of the structure of complex networks at different resolution levels”. In: *New journal of physics* 10.5 (2008), p. 053039.

[2] **Andrea L Bertozzi and Arjuna Flenner**. “Diffuse interface models on graphs for classification of high dimensional data”. In: *Multiscale Modeling & Simulation* 10.3 (2012), pp. 1090–1118.

[3] **Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre**. “Fast unfolding of communities in large networks”. In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[4] **Zachary M Boyd, Egil Bae, Xue-Cheng Tai, and Andrea L Bertozzi**. “Simplified energy landscape for modularity using total variation”. In: *SIAM Journal on Applied Mathematics* 78.5 (2018), pp. 2439–2464.

[5] **Andrea Braides et al.** *Gamma-convergence for Beginners*. Vol. 22. Clarendon Press, 2002.

[6] **Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner**. “On modularity clustering”. In: *IEEE transactions on knowledge and data engineering* 20.2 (2007), pp. 172–188.

[7] **Jeremy Budd, Yves van Gennip, and Jonas Latz**. “Classification and image processing with a semi-discrete scheme for fidelity forced Allen–Cahn on graphs”. In: *GAMM-Mitteilungen* 44.1 (2021), e202100004.

[8] **James R Bunch, Christopher P Nielsen, and Danny C Sorensen**. “Rank-one modification of the symmetric eigenproblem”. In: *Numerische Mathematik* 31.1 (1978), pp. 31–48.

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

[10] **Aaron Clauset, Mark EJ Newman, and Cristopher Moore**. “Finding community structure in very large networks”. In: *Physical review E* 70.6 (2004), p. 066111.

[11] *Code for implementing the spectral clustering with k–means on Zachary’s karate club*. https://www.alanshawn.com/jupyter-nb-show/2019/10/31/spectral-clustering.html. Accessed: 2022-01-27.

[12] *Contingency table*. https://en.wikipedia.org/wiki/Contingency_table.

[13] **Mihai Cucuringu, Andrea Pizzoferrato, and Yves Van Gennip**. “An MBO scheme for clustering and semi-supervised clustering of signed networks”. In: *arXiv preprint arXiv:1901.03091* (2019).

[14] **Gianni Dal Maso**. *An introduction to Γ-convergence*. Vol. 8. Springer Science & Business Media, 2012.

[15] **Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová**. "Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications". In: *Physical Review E* 84.6 (2011), p. 066106.

[16] **Tyler Derr, Yao Ma, and Jiliang Tang**. "Signed graph convolutional networks". In: *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2018, pp. 929–934.

[17] **Jordi Duch and Alex Arenas**. "Community detection in complex networks using extremal optimization". In: *Physical review E* 72.2 (2005), p. 027104.

[18] **Ewdurbin Ee Durbin and Lenards Andrew Lenards**. *The Community Python package*. 2014. URL: https://github.com/ewdurbin/community (visited on 02/06/2022).

[19] **Lidan Fan, Weili Wu, Zaixin Lu, Wen Xu, and Ding-Zhu Du**. "Influence diffusion, community detection, and link prediction in social network analysis". In: *Dynamics of information systems: algorithmic approaches*. Springer, 2013, pp. 305–325.

[20] **Santo Fortunato and Marc Barthelemy**. "Resolution limit in community detection". In: *Proceedings of the national academy of sciences* 104.1 (2007), pp. 36–41.

[21] **Santo Fortunato and Darko Hric**. "Community detection in networks: A user guide". In: *Physics reports* 659 (2016), pp. 1–44.

[22] **Joel N Franklin**. *Matrix theory*. Courier Corporation, 2012.

[23] **Cristina Garcia-Cardona, Ekaterina Merkurjev, A. Bertozzi, Arjuna Flenner, and Allon G. Percus**. "Fast Multiclass Segmentation using Diffuse Interface Methods on Graphs". In: 2013.

[24] **Cristina Garcia-Cardona, Ekaterina Merkurjev, Andrea L Bertozzi, Arjuna Flenner, and Allon G Percus**. "Multiclass data segmentation using diffuse interface methods on graphs". In: *IEEE transactions on pattern analysis and machine intelligence* 36.8 (2014), pp. 1600–1613.

[25] **Alexander J Gates and Yong-Yeol Ahn**. "The impact of random models on clustering similarity". In: *arXiv preprint arXiv:1701.06508* (2017).

[26] **Michelle Girvan and Mark EJ Newman**. "Community structure in social and biological networks". In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[27] **Tom Goldstein and Stanley Osher**. "The split Bregman method for L1-regularized problems". In: *SIAM journal on imaging sciences* 2.2 (2009), pp. 323–343.

[28] **Benjamin H Good, Yves-Alexandre De Montjoye, and Aaron Clauset**. "Performance of modularity maximization in practical contexts". In: *Physical Review E* 81.4 (2010), p. 046106.

[29] **Ming Gu and Stanley C Eisenstat**. "A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem". In: *SIAM journal on Matrix Analysis and Applications* 15.4 (1994), pp. 1266–1276.

[30] **Roger Guimera, Marta Sales-Pardo, and Luís A Nunes Amaral**. "Modularity from fluctuations in random graphs and complex networks". In: *Physical Review E* 70.2 (2004), p. 025101.

[31] **Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt**. "Stochastic blockmodels: First steps". In: *Social networks* 5.2 (1983), pp. 109–137.

[32] **Huiyi Hu, Thomas Laurent, Mason A Porter, and Andrea L Bertozzi**. "A method based on total variation for network modularity optimization using the MBO scheme". In: *SIAM Journal on Applied Mathematics* 73.6 (2013), pp. 2224–2246.

[33] **Lawrence Hubert and Phipps Arabie**. "Comparing partitions". In: *Journal of classification* 2.1 (1985), pp. 193–218.

[34] **Lucas GS Jeub, Olaf Sporns, and Santo Fortunato**. "Multiresolution consensus clustering in networks". In: *Scientific reports* 8.1 (2018), pp. 1–16.

[35] **Arzum Karataş and Serap Şahin**. "Application areas of community detection: A review". In: *2018 International congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT)*. IEEE. 2018, pp. 65–70.

[36] **Blaine Keetch and Yves van Gennip**. "A Max-Cut approximation using a graph based MBO scheme". In: *arXiv preprint arXiv:1711.02419* (2017).

[37] **Robert V Kohn and Peter Sternberg**. "Local minimisers and singular perturbations". In: *Proceedings of the Royal Society of Edinburgh Section A: Mathematics* 111.1-2 (1989), pp. 69–84.

[38] **Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos**. "Edge weight prediction in weighted signed networks". In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE. 2016, pp. 221–230.

[39] **Jérôme Kunegis, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto W De Luca, and Sahin Albayrak**. "Spectral analysis of signed graphs for clustering, prediction and visualization". In: *Proceedings of the 2010 SIAM international conference on data mining*. SIAM. 2010, pp. 559–570.

[40] **Andrea Lancichinetti and Santo Fortunato**. "Community detection algorithms: a comparative analysis". In: *Physical review E* 80.5 (2009), p. 056117.

[41] **Andrea Lancichinetti and Santo Fortunato**. "Limits of modularity maximization in community detection". In: *Physical review E* 84.6 (2011), p. 066122.

[42] **Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi**. "Benchmark graphs for testing community detection algorithms". In: *Physical review E* 78.4 (2008), p. 046110.

[43] **Xiyang Luo and Andrea L Bertozzi**. *Convergence analysis of the graph Allen-Cahn scheme*. Tech. rep. University of California Los Angeles Los Angeles United States, 2016.

[44] **Christopher D Manning and Prabhakar Raghavan**. *utze, Introduction to Information Retrieval*. 2008.

[45] **Laurent Massoulié**. “Community detection thresholds and the weak Ramanujan property”. In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 2014, pp. 694–703.

[46] **Pedro Mercado, Jessica Bosch, and Martin Stoll**. “Node classification for signed social networks using diffuse interface methods”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2019, pp. 524–540.

[47] **Pedro Mercado, Francesco Tudisco, and Matthias Hein**. “Clustering signed networks with the geometric mean of Laplacians”. In: *Advances in neural information processing systems*. 2016, pp. 4421–4429.

[48] **Ekaterina Merkurjev, Cristina Garcia-Cardona, Andrea L Bertozzi, Arjuna Flenner, and Allon G Percus**. “Diffuse interface methods for multiclass segmentation of high-dimensional data”. In: *Applied Mathematics Letters* 33 (2014), pp. 29–34.

[49] **Ekaterina Merkurjev, Tijana Kostic, and Andrea L Bertozzi**. “An MBO scheme on graphs for classification and image processing”. In: *SIAM Journal on Imaging Sciences* 6.4 (2013), pp. 1903–1930.

[50] **Barry Merriman, James K Bence, and Stanley J Osher**. “Motion of multiple junctions: A level set approach”. In: *Journal of computational physics* 112.2 (1994), pp. 334–363.

[51] **Barry Merriman, James Kenyard Bence, and Stanley Osher**. *Diffusion generated motion by mean curvature*. Vol. 27. Department of Mathematics, University of California, Los Angeles, 1992.

[52] **Luciano Modica and Stefano Mortola**. “Un esempio di Γ-convergenza”. In: *Bollettino Della Unione Matematica Italiana B* 5 (1977).

[53] **Michael Molloy, Bruce Reed, Mark Newman, Albert-László Barabási, and Duncan J Watts**. “A critical point for random graphs with a given degree sequence”. In: *The Structure and Dynamics of Networks*. Princeton University Press, 2011, pp. 240–258.

[54] **Mark EJ Newman**. “Finding community structure in networks using the eigenvectors of matrices”. In: *Physical review E* 74.3 (2006), p. 036104.

[55] **Mark EJ Newman**. “Modularity and community structure in networks”. In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[56] **Mark EJ Newman and Michelle Girvan**. “Finding and evaluating community structure in networks”. In: *Physical review E* 69.2 (2004), p. 026113.

[57] **Mark EJ Newman, Steven H Strogatz, and Duncan J Watts**. “Random graphs with arbitrary degree distributions and their applications”. In: *Physical review E* 64.2 (2001), p. 026118.

[58] **F. Pedregosa et al.** “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[59] **Paolo Perlasca, Marco Frasca, Cheick Tidiane Ba, Jessica Gliozzo, Marco Notaro, Mario Pennacchioni, Giorgio Valentini, and Marco Mesiti**. “Multi-resolution visualization and analysis of biomolecular networks through hierarchical community detection and web-based graphical tools”. In: *Plos one* 15.12 (2020), e0244241.

[60] **William M Rand**. “Objective criteria for the evaluation of clustering methods”. In: *Journal of the American Statistical association* 66.336 (1971), pp. 846–850.

[61] **Jörg Reichardt and Stefan Bornholdt**. “Statistical mechanics of community detection”. In: *Physical review E* 74.1 (2006), p. 016110.

[62] **Moshen Shahriari and Mahdi Jalili**. “Ranking nodes in signed social networks”. In: *Social network analysis and mining* 4.1 (2014), pp. 1–12.

[63] **Jianbo Shi and Jitendra Malik**. “Normalized cuts and image segmentation”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[64] **Jiliang Tang, Charu Aggarwal, and Huan Liu**. “Node classification in signed social networks”. In: *Proceedings of the 2016 SIAM international conference on data mining*. SIAM. 2016, pp. 54–62.

[65] *The NetworkX Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.* https://networkx.org/. Accessed: 2022-01-27.

[66] **Yves Van Gennip, Andrea L Bertozzi, et al.** “Γ-convergence of graph Ginzburg-Landau functionals”. In: *Advances in Differential Equations* 17.11/12 (2012), pp. 1115–1180.

[67] **Ulrike Von Luxburg**. “A tutorial on spectral clustering”. In: *Statistics and computing* 17.4 (2007), pp. 395–416.

[68] **Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou**. “Link prediction in social networks: the state-of-the-art”. In: *Science China Information Sciences* 58.1 (2015), pp. 1–38.

[69] **Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu**. “Signed network embedding in social media”. In: *Proceedings of the 2017 SIAM international conference on data mining*. SIAM. 2017, pp. 327–335.

[70] **Hermann Weyl**. “Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung)”. In: *Mathematische Annalen* 71.4 (1912), pp. 441–479.

[71] **Fang Wu and Bernardo A Huberman**. “Finding communities in linear time: a physics approach”. In: *The European Physical Journal B* 38.2 (2004), pp. 331–338.

[72] **C. Cortes Y. LeCun and C. J. Burges**. *The MNIST database of handwritten digits*. 1998. URL: http://yann.lecun.com/exdb/mnist/ (visited on 02/06/2022).

[73] **Wayne W Zachary**. “An information flow model for conflict and fission in small groups”. In: *Journal of anthropological research* 33.4 (1977), pp. 452–473.

[74] **Quan Zheng and David B Skillicorn**. “Spectral embedding of signed networks”. In: *Proceedings of the 2015 SIAM international conference on data mining*. SIAM. 2015, pp. 55–63.