SIGNAL PROCESSING ON GRAPHS: COMMUNITY DETECTION AND GRAPH LEARNING FOR MULTILAYER NETWORKS

By

Abdullah Karaaslanli

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Electrical Engineering—Doctor of Philosophy

2023

ABSTRACT

Community detection and graph learning are two important problems in graph analysis. The former problem deals with topological analysis of graphs to identify their mesoscale organization; while graph learning aims to infer the interactions between nodes of a graph from data when the graph topology is not known *a priori*. Existing community detection and graph learning methods are mostly limited to single-layer graphs, where nodes are assumed to be connected with a single static edge. However, this assumption ignores the fact that many real-world relational data have multiple dimensions, which can be better represented with *multilayer graphs*. In this thesis, we propose various community detection and graph learning methods for different types of multilayer graphs.

In Chapter 2, we tackle the community detection problem in dynamic networks. Specifically, we focus on evolutionary spectral clustering, which extends spectral clustering to dynamic networks to learn a community structure that changes smoothly over time. We show the equivalence of evolutionary spectral clustering to a variant of dynamic stochastic blockmodel. For this purpose, we first introduce a novel dynamic SBM where the evolution of communities over time is modeled with pairwise Markov random fields. We then show that the log-posterior of the proposed model is equivalent to the quality function of evolutionary spectral clustering. This equivalence is used to determine the forgetting factor in evolutionary spectral clustering and to develop two new algorithms for dynamic community detection. The proposed algorithms are applied to both simulated and real-world dynamic networks and their performances are compared to state-of-the-art dynamic community detection methods.

Chapter 3 introduces a multilayer community detection method, which is especially tailored to handle multilayer brain networks constructed from electroencephalogram(EEG) data. In particular, we first construct functional multilayer networks from EEG data, where layers correspond to different frequency bands and interlayer edges are allowed between all brain regions. Next, a new multilayer modularity metric is defined based on a multilayer null model that preserves the layer-wise node degrees while randomizing the remaining characteristics of the network. The proposed modularity is parameterized with resolution parameter to handle the resolution limit of modularity, and interlayer scale parameter to control the importance of interlayer edges in community formation. Third, a group community detection method is proposed to find the common community structure for a set of subjects. The proposed multilayer community detection method is employed to identify the group level differences between the two response types during Flanker task, *i.e.* error and correct.

In Chapter 4, we present an algorithm to learn signed graphs, which we represent as a two-layer multiplex network where one layer corresponds to positive edges while the other to negative edges. The algorithm is based on graph learning approaches developed using graph signal processing. Existing graph learning methods rely on smoothness of graph signals over the graph; however, they are only capable of learning unsigned graphs. To this end, we propose a signed graph learning approach, that learns signed graphs based on the assumption of smoothness and non-smoothness of graph signals over positive and negative edges, respectively. The proposed method is further extended using kernels to take the nonlinear relations between nodes into account. From GSP perspective, this extension corresponds to assuming smoothness/non-smoothness of graph signals in a higher dimensional space defined by the kernel. The proposed approach is applied to the problem of gene regulatory network inference from single cell gene expression data. Experiments on simulated and real single cell datasets show that the method compares favorably with other single cell gene regulatory network reconstruction algorithms.

Chapter 5 addresses the problem of how to learn multiple signed graphs simultaneously. Existing GSP based GL approaches for this problem are limited to unsigned graph topologies. Therefore, we extend the algorithm developed in Chapter 4 to learn multiple signed graphs. In particular, given multiple datasets each of which includes graph signals associated with a signed graph, we assume smoothness and non-smoothness of graph signals as in Chapter 4. Furthermore, we assume that the signed graphs are similar to each other, which is ensured by regularizing the learned signed graphs through a learned signed consensus graph. The proposed method is employed for the joint inference of multiple gene regulatory networks from single cell gene expression data. Experiments on simulated and real single cell datasets show that the method performs better than methods that can learn a single graph at a time and previous joint gene regulatory network reconstruction algorithms.

In Chapter 6, we tackle the problem of learning multiple unsigned graphs from a heterogeneous dataset, which requires clustering graph signals while learning a graph for each cluster. Namely, we present an optimization problem for joint graph signal clustering and graph topology inference. The approach extends graph cut based clustering by partitioning the graph signals not only based on their pairwise similarities but also their smoothness with respect to the graphs associated with the clusters. The proposed method also learns the representative graph for each cluster using the smoothness of the graph signals with respect to the graph topology. Results on simulated and real data indicate the effectiveness of the proposed method.

ACKNOWLEDGEMENTS

This thesis is written with the help of many great people and I would like to express my appreciation to them. First, I would like to give a special thank you to my advisor, Dr. Selin Aviyente, for giving me the opportunity to work with her and for introducing the fascinating topic of this thesis to me. Without her guidance, patience and encouragement, this thesis would not be possible. Secondly, I want to thank Dr. Tapabrata Maiti, whose co-advising helped me greatly during the preparation of parts of this thesis. I would also like to thank my co-authors, Dr. Satabdi Saha, Dr. Tamanna Munia and Meiby Ortizbouza. It was a great pleasure to work with them and to learn from their expertise. Furthermore, I would like to thank the committee members of this thesis for their time and effort. They provided comments and suggestions, which gave this thesis its final shape. Finally, I am grateful for my family and my friends, whose valuable support gave me courage while pursuing my degree.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Many real-world applications consist of networked systems, *i.e.* they include entities that are related to each other in different ways. For example, users on a social media platform connect through messaging, or genes and proteins within a cell interact through regulatory relations. Such systems can be modeled as *graphs* (or *networks*[1]), where entities and their interactions are represented by *nodes* and *edges*, respectively [166, 5, 27]. Although graphs have successfully been used in many disciplines, existing work is generally limited to *single-layer graphs*, where nodes are assumed to be connected with a single static edge. This assumption ignores the fact that many real-world relational data have multiple dimensions. For instance, a user on a social media platform can connect to another user through friendship, messaging or post-sharing. Similarly, in brain networks, the functional connectivity between brain regions occurs across multiple frequency bands [59, 248]. *Multilayer graphs* are developed to represent and study this multiplicity of interactions, simultaneously [117, 28, 6]. In a multilayer graph, different interactions are represented by layers as depicted in Figure 1.1c. Layers consist of nodes and *intralayer edges* representing entities and interactions, respectively. Beside intralayer edges, a multilayer network may include *interlayer edges* that connect nodes from different layers.

Albeit the possible oversimplification of single-layer graphs, they have been used to reveal many structural and dynamic properties of networked systems: centrality of nodes [29, 53], small-worldness [254, 172], scale-free property [15, 16] *etc.* One of the fundamental properties of graphs is community structure, where the nodes are partitioned into tightly connected groups of nodes [79, 81]. Many algorithms have been developed for detection of communities, as identification of communities has important applications in recommendation systems [197], social sciences [149] and network neuroscience [232]. However, most of these methods are developed for single-layer graphs and are not directly applicable for community detection in multilayer graphs. Considering the importance of communities in graph analysis, there is a need for developing community detection algorithms for multilayer graphs.

The topological analysis of graphs characterizes a networked system by studying the interactions between entities. However, nodes of a graph can be associated with a significant amount of data that also needs to be studied. For instance, nodes in a transportation network can have attributes related to logistic data describing how goods are traded or people in a social network are associated with various data such as age, gender

[1]Throughout this thesis, the terms graph and network are used interchangeably.

etc. [227]. *Graph Signal Processing (GSP)* is a recent research field that aims to learn from this data by incorporating the graph topology into learning algorithms. In GSP, node data is represented as a *graph signal*, which can be considered as a vector whose entries are indexed by graph nodes. Graph signals can then be studied with different tools that extend classical signal processing concepts such as Fourier transform, filtering, sampling and imputation [178].

In many applications of network science and GSP, the graph topology is assumed to be known. This assumption holds in some areas, *e.g.* friendship networks or citation networks. However, there are many cases where the graph topology is not readily available. For instance, in network neuroscience, the functional interactions between brain regions are not known and they need to be learned from data collected by functional magnetic resonance imaging (fMRI) or electroencephalogram (EEG) recordings. To this end, various *graph learning (GL)* methods are developed to infer the graph topology from graph signals. Traditional GL methods includes statistical modeling, such as probabilistic graphical models [82, 14], or physically motivated methods, where graph signals are modelled as a product of dynamic processes on the graph [196, 161]. Recently, graph learning problem is considered from a GSP perspective, where graph Fourier transform (GFT) of signals is employed [68, 137]. Due to explicit representation of graph signals with GFT, these methods provide great flexibility and are observed to perform better than traditional GL methods [69, 107, 23]. However, most of existing graph learning methods are limited to infering only a single connection between nodes and only a few works consider learning multilayer networks [164, 110].

1.1 Background and Notations

In this thesis, scalars, vectors and matrices are indicated by letters (x or N), bold lowercase letters ($\mathbf{x}$) and bold uppercase letters ($\mathbf{X}$), respectively. Entries of a vector are denoted as x_i and entries of a matrix are denoted as X_{ij}. ith row and column of $\mathbf{X}$ are indicated as $\mathbf{X}_{i\cdot}$ and $\mathbf{X}_{\cdot i}$, respectively and both are assumed to be column vectors. Superscript $^\top$ indicates transpose of vectors and matrices. $\langle \cdot, \cdot \rangle$ is used to represent the inner product. Identity matrix is shown by $\mathbf{I}$. All ones and zero vectors and matrices are shown as $\mathbf{1}$ and $\mathbf{0}$, respectively[2]. The operator diag() either takes a matrix $\mathbf{X}$ and returns a vector $\mathbf{x}$ with $x_i = X_{ii}$ or takes a vector $\mathbf{x}$ and returns a diagonal matrix $\mathbf{X}$ with $X_{ii} = x_i$. The operator upper() : $\mathbb{R}^{n\times n} \rightarrow \mathbb{R}^m$ returns upper triangular part of the input matrix where $m = n(n-1)/2$. For an $n \times n$ symmetric matrix $\mathbf{A}$, the matrix $\mathbf{S} \in \mathbb{R}^{n\times m}$ is defined such that $\mathbf{S}\text{upper}(\mathbf{A}) = \mathbf{A}\mathbf{1} - \text{diag}(\mathbf{A})$. Finally, δ_{ij} is Kronecker delta, which is 1 if $i = j$

[2]If the dimensions of these vectors/matrices are not clear from context, they will be shown with a subscript indicating the dimensions: *e.g.* $\mathbf{1}_n$ or $\mathbf{0}_{n\times n}$.

and 0, otherwise.

1.1.1 Single-layer Graphs

A single-layer network is denoted by $G = (V, E)$ where V is the node set with $|V| = n$ and $E \subseteq V \times V$ is the edge set. An edge from node u to v is represented by e_{uv} and it is associated with a weight w_{uv}. If $e_{uv} = e_{vu}$, the graph is said to be undirected and otherwise, it is a directed graph. In this thesis, the graphs are assumed to be undirected unless otherwise stated. If $w_{uv} = 1,\ \forall e_{uv} \in E$, the graph is binary; otherwise, it is weighted. G is an unsigned graph, if edge weights are constrained to only positive values. Finally, if the edge weights can take on both positive and negative values, the graph is said to be signed.

Algebraically, an unsigned graph G can be represented by a symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $A_{uv} = w_{uv}$ if $e_{uv} \in E$ and 0, otherwise. Degree of a node u is the sum of weights of the edges connected to it, *i.e.* $d_u = \mathbf{A}_{u\cdot}^{\top}\mathbf{1}$. Degree vector of G is $\mathbf{d} = \mathbf{A1}$ and $\mathbf{D} = \text{diag}(\mathbf{d})$ is its degree matrix. The combinatorial Laplacian matrix of G is $\mathbf{L} = \mathbf{D} - \mathbf{A}$. $\mathbf{L}$ is a positive semi-definite matrix and has eigendecomposition $\mathbf{L} = \mathbf{V\Lambda V}^{\top}$ where $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues and columns of $\mathbf{V}$ are eigenvectors. Eigenvalues of $\mathbf{L}$ are assumed to be sorted in ascending order, *i.e.* $0 = \Lambda_{11} \leq \Lambda_{22} \leq \cdots \leq \Lambda_{nn}$.

1.1.2 Multilayer Graphs

A multilayer network is a quadruplet $\mathcal{M} = (\mathcal{V}, \mathcal{L}, V, E)$ where $\mathcal{V}$ is the set of entities, *e.g.* people or brain regions, $\mathcal{L}$ is the set of layers with $|\mathcal{L}| = L$ [117]. $V \subseteq \mathcal{V} \times \mathcal{L}$ with $|V| = n$ is the set of nodes, which are representations of entities in layers and $E \subseteq V \times V$ is the edge set. Nodes are indicated as $u^{\hbar}$, where $u \in \mathcal{V}$ and $\hbar \in \mathcal{L}$. An edge from $u^{\hbar}$ to v^{κ} is indicated by $e_{uv}^{\hbar\kappa}$ and associated with the weight $w_{uv}^{\hbar\kappa}$. Similar to single-layer networks, $\mathcal{M}$ can be undirected/directed, binary/weighted or unsigned/signed. In this thesis, the multilayer graphs are assumed to be undirected unless otherwise stated.

V can be partitioned based on layers, that is $V = \bigcup_{\hbar=1}^{\ell} V^{\hbar}$ where $V^{\hbar}$ with $|V^{\hbar}| = n^{\hbar}$ is the set of nodes in layer $\hbar$. Similarly, E can be partitioned by $E = \bigcup_{\hbar=1}^{\ell} E^{\hbar} \cup \bigcup_{\hbar\neq\kappa=1}^{\ell} E^{\hbar\kappa}$ where $E^{\hbar}$ is the set of intralayer edges in layer $\hbar$ and $E^{\hbar\kappa}$ is the set of interlayer edges between nodes in layers $\hbar$ and κ. From the partitioning of V and E, one can define intralayer graphs $G^{\hbar} = (V^{\hbar}, E^{\hbar})$ and bipartite interlayer graphs $G^{\hbar\kappa} = (V^{\hbar}, V^{\kappa}, E^{\hbar\kappa})$[3]. Let $\mathbf{A}^{\hbar}$ be the adjacency matrix of $G^{\hbar}$ and $\mathbf{A}^{\hbar\kappa}$ be the incidence matrix of $G^{\hbar\kappa}$. $\mathcal{M}$ is represented by a

[3] A bipartite graph $G = (V_1, V_2, E)$ consists of two node sets V_1 and V_2 with $|V_1| = n_1$ and $|V_2| = n_2$ and edges are only allowed between two sets, *i.e.* $E \subseteq V_1 \times V_2$. The incidence matrix of G is $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$ where $A_{ij} = w_{ij}$ is $e_{ij} \in E$ and 0, otherwise.

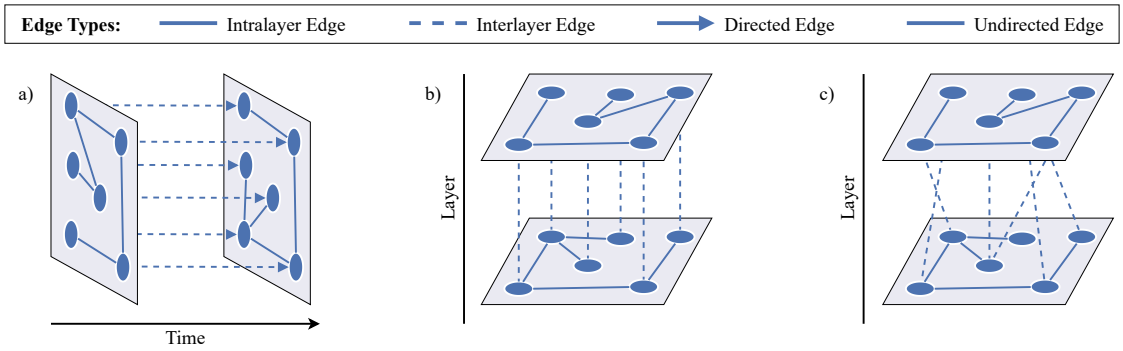

Figure 1.1: Types of graphs used in this thesis. a) shows a two layer dynamic network, where layers are ordered and correspond to time points. b) shows a two layer multiplex graph, where interlayer edges are only allowed between nodes that represent the same entity. c) shows a two layer multilayer graph, where interlayer edges can occur between any pair of nodes.

supra-adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, a symmetric block matrix defined as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}^1 & \mathbf{A}^{12} & \dots & \mathbf{A}^{1\ell} \\ \mathbf{A}^{21} & \mathbf{A}^2 & \dots & \mathbf{A}^{2\ell} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}^{\ell 1} & \mathbf{A}^{\ell 2} & \dots & \mathbf{A}^{\ell} \end{bmatrix}. \tag{1.1}$$

Using the supra-adjacency matrix of $\mathcal{M}$, its supra-Laplacian matrix can be defined analogous to the Laplacian matrix of single-layer networks.

In a multilayer graph, there are no constraints on the set E^{hk}, *i.e.* there could be an edge between any u^h and v^k as depicted in Figure 1.1c. In this thesis, we will also use two other graph types, that can be considered as constrained versions of multilayer graphs. If interlayer connections are allowed only between nodes representing the same entity, *i.e.* $E^{hk} = \{e^{hk}_{uu} | u^h \in V^h, u^k \in V^k\}$ for all $h \neq k$, the network is a *multiplex* (or *multiview*[4]) *network* (Figure 1.1b). A *dynamic network* is a type of multiplex network, whose layers are ordered and correspond to time points and interlayer edges are only allowed between consecutive time points, *i.e.* $E^{hk} = \{e^{hk}_{uu} | u^h \in V^h, u^k \in V^k\}$ if $k = h + 1$ and $E^{hk} = \varnothing$, otherwise (Figure 1.1a).

1.2 Community Detection

Edges of many real-world networks are distributed heterogeneously such that there are high number of edges within groups of nodes and low number of edges between groups. This feature is called the community structure [79]. The community structure of a single-layer graph G can be one of the following:

[4]Throughout the thesis, the terms multiplex and multiview are used interchangeably.

non-overlapping, *overlapping*, *hierarchical* or *local* [39]. In this thesis, the focus is on non-overlapping community detection, which is the partitioning of node set V as $\mathcal{P} = \{C_1, \ldots, C_K\}$ where K is the number of communities. The community structure $\mathcal{P}$ can be represented by various mathematical objects: community membership vector $\mathbf{g} \in \mathbb{R}^n$ whose entries are $g_i = r$ if $i \in C_r$, or binary indicator matrix $\mathbf{Z} \in \mathbb{R}^{n \times K}$ which is defined with entries $Z_{ir} = 1$ if $g_i = r$ and 0, otherwise. The aim of community detection is algorithmic identification of $\mathcal{P}$. This task is usually performed by optimizing a *quality function* that quantifies the goodness of a given partition to be a community structure. A plethora of quality functions are proposed for single-layer graphs [79] and an overview of the ones used in this thesis is given below.

1.2.1 Graph Cut and Association

As mentioned, a community structure is defined as the partitioning of the nodes into well-connected groups while groups are sparsely connected to each other. Therefore, one way to measure the goodness of a partition is to count the number of inter-community edges, referred to as the *cut* of a partition, which is defined as [225, 249]:

$$\text{cut}(\mathcal{P}) = \sum_{i,j=1}^{p} A_{ij}(1 - \delta_{g_i g_j}) = \text{tr}(\mathbf{Z}^\top \mathbf{L} \mathbf{Z}). \tag{1.2}$$

Instead of minimizing the cut, one can also maximize the number of intra-community edges, referred to as the *association* of a partition [64]:

$$\text{assoc}(\mathcal{P}) = \sum_{i<j}^{n} A_{ij}\delta_{g_i g_j} = \frac{1}{2}\text{tr}(\mathbf{Z}^\top \mathbf{A} \mathbf{Z}). \tag{1.3}$$

Optimizing the cut or association with respect to $\mathbf{Z}$ leads to the trivial solution, where all nodes are assigned to the same community. To prevent this, $\mathbf{Z}$ is further constrained to make sure communities have similar sizes. However, due to the discreet structure of $\mathbf{Z}$, the optimization problem is NP-hard [249]; therefore, $\mathbf{Z}$ is relaxed to take on real values, which leads to the following optimization problem:

$$\underset{\mathbf{Z}}{\text{minimize}} \quad f(\mathbf{Z}) \tag{1.4}$$

$$\text{subject to} \quad \mathbf{Z} \in \mathbb{D}, \tag{1.5}$$

where $f(\mathbf{Z})$ is either cut or association and $\mathbf{Z}$ is constrained to be in a set $\mathbb{D}$ to ensure that $\mathbf{Z}$ preserves some properties of its discrete form. These properties can include positivity ($\mathbf{Z} \geq 0$), row-sum constraint ($\mathbf{Z1} = \mathbf{1}$) or orthogonality ($\mathbf{Z}^\top \mathbf{Z} = \mathbf{I}$) [249, 223, 267]. Once a real valued $\mathbf{Z}$ is learned, clustering algorithms such as k-means can be employed to identify the community structure.

1.2.2 Modularity

Another popular quality function for community detection is the modularity function, which quantifies the quality of a community structure by comparing intra-community connections to those expected under a specified null model. It is calculated as [171]:

$$Q = \sum_{i,j} [A_{ij} - P_{ij}] \delta_{g_i g_j}, \tag{1.6}$$

where P_{ij} is the expected connection between nodes i and j under a null model. Depending on the graph under study, different expressions for P_{ij} can be assumed. The most commonly used null models are the configuration null model and Erdős–Rényi null model [21]. Despite its popularity, modularity is known to suffer from the resolution limit that limits the size of detectable communities; communities smaller than some size are mathematically undetectable. In order to detect communities of all sizes, modularity has been extended to include a resolution parameter, γ, which is tuned to uncover communities of different size [199]:

$$Q = \sum_{i,j} [A_{ij} - \gamma P_{ij}] \delta_{g_i g_j}. \tag{1.7}$$

By varying the value of γ one can detect communities of different sizes, *i.e.* when γ is large or small maximizing modularity will return correspondingly small or large communities, respectively, resulting in multi-scale community structure.

1.2.3 Stochastic Blockmodeling

Stochastic blockmodeling (SBM) is a generative network model developed to study networks with block structure, where nodes are assigned to one of K blocks. Given block assignments, edges are sampled independently from a Bernoulli distribution with a $K \times K$ edge probability matrix $\boldsymbol{\theta}$, where θ_{rs} is the probability of connectivity between blocks r and s [97, 87]. For networks with a community structure $\theta_{rr} > \theta_{rs}$, $\forall r \neq s$. In this thesis, we employ a restricted version of SBM called planted-partition model, where $\theta_{rs} = \theta_{in}$ if $r = s$ and $\theta_{rs} = \theta_{out}$, otherwise [49]. θ_{in} and θ_{out} are intra- and inter-community connectivity probabilities, respectively.

Besides generating networks, SBM is also used for statistical inference of community structure [231, 113, 2, 170, 3]. In [113], community detection with standard SBM is shown to fail in networks with a heterogeneous degree distribution. To overcome this problem, degree corrected SBM (DCSBM) is introduced, where edge probabilities are modified by the degrees of nodes. Given the community assignment $\mathbf{g}$, edges are sampled independently from a Poisson distribution with mean $\lambda_{ij} = d_i d_j \theta_{g_i g_j}$. Community

detection is then performed by maximizing the likelihood function, which can be written as:

$$\mathrm{P}(\mathbf{A}|\mathbf{g},\boldsymbol{\theta}) = \prod_{i<j}^{n} \frac{\lambda_{ij}^{A_{ij}} \exp(-\lambda_{ij})}{A_{ij}!}, \tag{1.8}$$

Different techniques, such as heuristic methods [113], variational inference [2, 3] and Markov Chain Monte Carlo methods [231, 184], are employed to maximize the log-likelihood function.

1.3 Graph Signal Processing

A graph signal over a graph G is a function $x : V \rightarrow \mathbb{R}$ and can be represented by a vector $\mathbf{x} \in \mathbb{R}^n$ where each x_i is the signal value on node i. An important concept in the processing of graph signals is their representation in graph frequency domain through graph Fourier transform (GFT). This representation allows us to characterize $\mathbf{x}$ in terms of its graph spectral content as either low- or high-frequency, where low(high)-frequency graph signals have small (large) variation with respect to the graph [210]. For an unsigned graph G, GFT is defined as the expansion of $\mathbf{x}$ in terms of the eigenbasis of the graph Laplacian [227]. Let $\mathbf{L}$ be the combinatorial Laplacian of an unsigned graph G with eigendecomposition $\mathbf{L} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$ as described in Section 1.1.1. GFT of $\mathbf{x}$ is then $\widehat{\mathbf{x}} = \mathbf{V}^\top\mathbf{x}$ and inverse GFT is [227]:

$$\mathbf{x} = \mathbf{V}\widehat{\mathbf{x}} = \sum_{i=1}^{n} \widehat{x}_i \mathbf{V}_{\cdot i}. \tag{1.9}$$

Thus, $\mathbf{x}$ is the linear combination of eigenvectors of $\mathbf{L}$ with the coefficients equal to the entries of $\widehat{\mathbf{x}}$. Eigenvectors of $\mathbf{L}$ corresponding to small eigenvalues have small variation over the graph. Thus, if most of the energy of $\widehat{\mathbf{x}}$ lies in $\widehat{x}_i$s corresponding to the small eigenvalues, then $\mathbf{x}$ varies little over G, *i.e.* it is *smooth*. On the other hand, if most of the energy of $\widehat{\mathbf{x}}$ lies in $\widehat{x}_i$s corresponding to the large eigenvalues, $\mathbf{x}$ has high variation over G, *i.e.* it is *non-smooth*. The total variation of $\mathbf{x}$ over G is then quantified as [227]:

$$\mathrm{tr}(\widehat{\mathbf{x}}^\top\boldsymbol{\Lambda}\widehat{\mathbf{x}}) = \mathrm{tr}(\mathbf{x}^\top\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top\mathbf{x}) = \mathrm{tr}(\mathbf{x}^\top\mathbf{L}\mathbf{x}), \tag{1.10}$$

which is small for low-frequency graph signals and large for high-frequency ones.

1.3.1 Unsigned Graph Learning

An unknown unsigned graph G can be learned from a set of observed graph signals based on the assumptions made about the relation between graph signals and the topology of G. In GSP based GL, two major approaches are followed: smoothness based methods [68], where the graph is learned with the assumption that graph signals vary smoothly with respect to G; and stationarity based methods, where the

graph is learned from signals that are assumed to be stationary on G [137]. In this thesis, we focus on learning graphs with the smoothness assumption because of the following reasons. First, smooth signals admit low-pass and sparse representations in the graph Fourier domain. Thus, the GL problem is equivalent to finding efficient information processing transforms for graph signals. Second, many graph-based machine learning tasks, such as spectral clustering, graph regularized learning *etc.*, are developed based on the smoothness of the graph signals. Finally, smooth graph signals are observed ubiquitously in real-world applications [137].

Smoothness based GL is first considered in [69] by modeling graph signals using factor analysis, where the transformation from factors to observed signals exploits the graph topology. By imposing a suitable prior on factors, the graph signals are modelled to have low-frequency representation in the graph Fourier domain. This analysis results in an optimization problem where G is learned by minimizing (1.10) with respect to $\mathbf{L}$ given a set of graph signals $\{\mathbf{x}_i\}_{i=1}^{p}$ as follows:

$$\begin{aligned} \underset{\mathbf{L}\in\mathbb{L}}{\text{minimize}} \quad & \text{tr}(\mathbf{X}^\top \mathbf{L}\mathbf{X}) + \alpha\|\mathbf{L}\|_F^2 \\ \text{subject to} \quad & \text{tr}(\mathbf{L}) = 2n, \end{aligned} \tag{1.11}$$

where $\mathbf{X} \in \mathbb{R}^{n\times p}$ is the data matrix whose columns are $\mathbf{x}_i$'s, $\mathbb{L} = \{\mathbf{L} : L_{ij} = L_{ji} \leq 0\ \forall i \neq j,\ \mathbf{L}\mathbf{1} = \mathbf{0}\}$ is the set of valid Laplacian matrices. The first term in (1.11) measures the total variation of the graph signals. The second term is the Frobenius norm of $\mathbf{L}$ and controls the density of the learned graph such that larger values of α result in denser graphs. Finally, the last constraint is added to prevent the trivial solution $\mathbf{L} = \mathbf{0}$.

1.4 Organization and Contributions of the Thesis

In this thesis, we develop methods for two important problems in multilayer networks: community detection and graph learning. Methods for community detection are developed by answering questions like what constitutes a community in a multilayer network and how to incorporate information from multiple layers to detect meaningful communities. In current literature, there is no consensus on the definition of communities and how to incorporate data from different layers is still an open problem. This thesis aims to answer these questions by extending quality functions defined for single-layer graphs to multilayer networks in a principled way. Graph learning approaches are developed based on recent advances in GSP, where graph frequency representation of graph signals are exploited for topology inference. Most of the existing graph learning approaches are limited to cases where the observed data is assumed to be homogeneous and low frequency with respect to a single common graph topology. Thus, we extend graph learning to multilayer network settings. These extensions lead to optimization problems which are solved by efficient algorithms.

In Chapter 2, we tackle community detection problem in dynamic networks. Specifically, we focus on evolutionary spectral clustering, which extends spectral clustering to dynamic networks by incorporating information from past time points to improve community detection at a time point. In order to answer the question of how to incorporate the past information, we show the equivalence of evolutionary spectral clustering to a variant of dynamic stochastic blockmodel. Namely, we first introduce a novel dynamic SBM (MRF-DCSBM) where the evolution of communities over time is modeled with pairwise Markov random fields. We then show that the log-posterior of the proposed model is equivalent to the quality function of evolutionary spectral clustering. This equivalence is used to determine the forgetting factor in evolutionary spectral clustering and to develop two new algorithms for dynamic community detection. The proposed algorithms are applied to both simulated and real-world dynamic networks and their performances are compared to state-of-the-art dynamic community detection methods.

Chapter 3 introduces a multilayer community detection method, which is especially tailored to handle multilayer brain networks constructed from EEG data. In particular, we first construct functional multilayer networks from EEG data, where layers correspond to different frequency bands and interlayer edges are allowed between all brain regions. Next, a new multilayer modularity metric is defined based on a multilayer null model that preserves the layer-wise node degrees while randomizing the remaining characteristics of the network. The proposed modularity is parameterized with resolution parameter to handle the resolution limit of modularity, and interlayer scale parameter to control the importance of interlayer edges in community formation. Third, a group community detection method is proposed to find the common community structure for a set of subjects. The proposed multilayer community detection method is employed to identify the group level differences between the two response types during Flanker task, *i.e.* error and correct.

In Chapter 4, we present an algorithm to learn signed graphs, which we represent as a two-layer multiplex network where one layer corresponds to positive edges while the other to negative edges. The algorithm is based on graph learning approaches developed using graph signal processing. Existing graph learning methods rely on smoothness of graph signals over the graph; however, they are only capable of learning unsigned graphs. To this end, we propose a signed graph learning approach, that learns signed graphs based on the assumption of smoothness and non-smoothness of graph signals over positive and negative edges, respectively. The proposed method is further extended with kernels to take the nonlinear relations between nodes into account. From GSP perspective, this extension corresponds to assuming smoothness/non-smoothness of graph signals in a higher dimensional space defined by the kernel. The proposed approach

is applied to the problem of gene regulatory network inference from single cell gene expression data. Experiments on simulated and real single cell datasets show that the method compares favorably with other single cell gene regulatory network reconstruction algorithms.

Chapter 5 addresses the problem of how to learn multiple signed graphs simultaneously. Existing GSP based GL approaches for this problem are limited to unsigned graph topologies. Therefore, we extend the algorithm developed in Chapter 4 to learn multiple signed graphs. In particular, given multiple datasets each of which includes graph signals associated with a signed graph, we assume smoothness and non-smoothness of graph signals as in Chapter 4. Furthermore, we assume that the signed graphs are similar to each other, which is ensured by regularizing the learned signed graphs through a learned signed consensus graph. The proposed method is employed for the joint inference of multiple gene regulatory networks from single cell gene expression data. Experiments on simulated and real single cell datasets show that the method performs better than methods that can learn a single graph at a time and previous joint gene regulatory network reconstruction algorithms.

In Chapter 6, we tackle the problem of learning multiple unsigned graphs from a heterogeneous dataset, which includes graph signals that are clustered and each cluster is associated with a different graph. Namely, we present an optimization problem for joint graph signal clustering and graph topology inference. The approach extends graph cut based clustering by partitioning the graph signals not only based on their pairwise similarities but also their smoothness with respect to the graphs associated with the clusters. The proposed method also learns the representative graph for each cluster using the smoothness of the graph signals with respect to the graph topology. Results on simulated and real data indicate the effectiveness of the proposed method.

Finally, concluding remarks are presented in Chapter 7, where we summarize the contributions of the thesis and discuss future work that will extend community detection and graph learning methods presented throughout the thesis.

CHAPTER 2

COMMUNITY DETECTION IN DYNAMIC NETWORKS

2.1 Introduction

An important problem in the study of networks is *community detection* where the nodes of a network are partitioned into tightly connected groups of nodes [79]. Identification of communities has important applications in recommendation systems [197], social sciences [149] and network neuroscience [232]. Community detection is usually performed by optimizing a *quality function* that quantifies the goodness of a given partition. Quality functions can be divided into two categories [181, 84]. The first category consists of functions that are based on heuristic definitions of what constitutes a good community, such as modularity [171], normalized cut (spectral clustering) [225, 249] and InfoMap [204]. The second category relies on statistical network modeling [87], such as stochastic blockmodels (SBM) [231], degree-corrected SBM (DCSBM) [113] or latent space models [95]. In this category, the network is assumed to be generated from a statistical network model and the communities are detected by maximizing the likelihood.

Since different quality functions define what constitutes a good community differently, community structures detected by different algorithms may vary from each other. Furthermore, as shown in [84], no single method can provide the correct community structure for all types of real-world networks. Understanding why a particular method fails in some networks is important for both finding ways to improve existing algorithms and deciding which method is more suitable for a given network. To this end, there has been an interest in quantifying the relationship between different quality functions to better understand why a particular quality function fails for certain networks. Moreover, this relationship can provide a way to select the hyperparameters, e.g. resolution parameter in the definition of modularity, in a principled way.

Recently, Newman *et al.* [167] have shown that maximizing modularity is equivalent to maximizing the likelihood function of DCSBM under the planted-partition model. A similar result is also shown for spectral clustering in [169]. These results reveal that modularity and spectral clustering assume communities to be statistically similar, i.e. the communities have similar size and edge density. The accuracy of community detection deteriorates if this assumption does not hold. This equivalence is also used for determining the resolution parameter of modularity using DCSBM parameters.

All these works consider only single-layer networks and their analysis is not applicable to multilayer networks. Considering the ubiquity of multilayer networks in real life, there is a need to extend this analysis

to multilayer networks. One of the important types of multilayer networks is dynamic networks. Community detection methods developed for single-layer networks are not directly applicable to dynamic networks, since the aim of the latter is not only to partition the nodes at each time point but also to track the changes in the partitions over time [203]. Recently, both heuristic and statistical quality functions have been extended to dynamic networks. For heuristic methods, either evolutionary clustering or multilayer network models have been used. Evolutionary clustering methods rely on the quality functions for single-layer networks, by combining *snapshot cost* at each time with a *temporal cost* to obtain community structures that change smoothly over time [46, 78, 132, 126]. The amount of smoothness is controlled by tuning the temporal cost with a *forgetting factor*, whose value is generally determined through grid search and set to be the same at all time points. In [259], an adaptive forgetting factor is proposed to eliminate grid search and to obtain a time-varying forgetting factor. Multilayer models [117], on the other hand, extends quality functions for single-layer networks to dynamic networks using the multilayer representation of dynamic networks shown in Figure 1.1. Examples of multilayer models are multislice modularity [153] and temporal normalized cut [228]. These works also require selection of *interlayer coupling* that controls the evolution of community structure. In the second category, statistical models for dynamic networks are proposed by defining a dynamic process describing the evolution of the community structure or the parameters of the statistical model [115, 268, 257, 52]. In [264, 85, 181], SBM and DCSBM are extended to dynamic networks by modeling the evolution of community structure over time as a first-order Markov process. [258] uses a state-space model to characterize the evolution of SBM parameters. In [138], both parameters and community structure are allowed to change and identifiability issues are handled by assuming stable intra-community connectivity over time. Dynamic latent space models are also developed [213].

Similar to static networks, showing the relationship between different quality functions defined for dynamic community detection can help us to refine the methods and understand their shortcomings. Using the relationship between heuristic methods and statistical models, one can also set the different hyperparameters in a more rigorous way. Recently, Pamfil *et al.* [181] have shown that multislice modularity is equivalent to dynamic planted-partition DCSBM, when the evolution of community structure over time is modeled by a first-order Markov process. This equivalence is used to determine the resolution parameter (γ) and interlayer coupling (ω) in multislice modularity through the parameters of dynamic DCSBM. Furthermore, this equivalence provides a better understanding of the assumptions and shortcomings of multislice modularity.

In this chapter, we show equivalence between evolutionary spectral clustering, i.e. preserving cluster

membership (PCM), and a novel dynamic DCSBM formulation. This equivalence provides a principled framework for the selection of the forgetting factor in PCM through dynamic DCSBM parameters. Furthermore, the equivalence between these two methods provides an efficient algorithm, i.e. spectral clustering, for likelihood maximization of dynamic DCSBM.

The contributions of this chapter can be summarized as follows:

- We introduce a new dynamic DCSBM assuming a planted-partition model. Different from previous dynamic DCSBMs, we model the evolution of community structure over time using pairwise Markov Random Fields (MRFs). In the proposed MRF model, the potential functions at the current time depend on the community structure at the previous time point. This new model is referred to as *dynamic MRF-DCSBM*.

- We show the equivalence between dynamic MRF-DCSBM and evolutionary spectral clustering by deriving a relationship between log-posterior function of the statistical model and trace maximization in PCM.

- This equivalence is exploited to propose two new dynamic community detection algorithms: online (DSC_{on}) and offline (DSC_{off}) dynamic spectral clustering.

- The equivalence between dynamic MRF-DCSBM and evolutionary spectral clustering provides a principled way to select the *forgetting factor*, which determines the amount of tradeoff between accuracy and smoothness in community structure, through the parameters of dynamic DCSBM. Unlike regular evolutionary spectral clustering, in the proposed algorithms, this factor is time dependent and adapts to the changes in community structure.

The remainder of this chapter is organized as follows. In Section 2.2, background on evolutionary spectral clustering, dynamic SBM and MRFs are presented. In Section 2.3, dynamic MRF-DCSBM is introduced and the equivalence between the log-posterior function and PCM quality function is derived. Proposed algorithms are derived in Section 2.4. Finally, experimental results and conclusions are given in Sections 2.5 and 2.6, respectively.

2.2 Background

A dynamic network $\mathcal{G}$ is a type of multilayer network, where each layer is a single-layer network observed at a time point t. $\mathcal{G}$ can be considered as a sequence of single-layer networks, i.e. $\mathcal{G} = \{G^1, \ldots, G^T\}$,

where $G^t = (V^t, E^t)$ is the network observed at time t with $|V^t| = n^t$ and $|E^t| = m^t$. Adjacency matrices in $\mathcal{G}$ are represented by a sequence $\mathcal{A} = \{\mathbf{A}^1, \ldots, \mathbf{A}^T\}$. Similarly, $\mathcal{D} = \{\mathbf{D}^1, \ldots, \mathbf{D}^T\}$ is the sequence of degree matrices where $\mathbf{D}^t$ is the degree matrix of G^t. Community structure of a dynamic network $\mathcal{G}$ is the partitioning of its nodes at each time t into K^t communities and is represented by $\mathcal{g} = \{\mathbf{g}^1, \ldots, \mathbf{g}^T\}$ and $\mathcal{Z} = \{\mathbf{Z}^1, \ldots, \mathbf{Z}^T\}$ where $\mathbf{g}^t$ and $\mathbf{Z}^t$ are defined as in Section 1.2. In the following derivations, we assume $n^t = n$ and $K^t = K$ $\forall t \in \{1, \ldots, T\}$ and the extension is discussed in Section 2.4.

2.2.1 Evolutionary Spectral Clustering

Evolutionary spectral clustering, i.e. PCM, is developed by extending association described in Section 1.2 to dynamic networks. The cost function of PCM is defined as [46]:

$$PCM := \text{tr}(\mathbf{Z}^{t\top}\mathbf{A}^t\mathbf{Z}^t) + \alpha\text{tr}(\mathbf{Z}^{t\top}\mathbf{Z}^{t-1}\mathbf{Z}^{t-1\top}\mathbf{Z}^t), \tag{2.1}$$

where α is the forgetting factor and the second term quantifies the difference between the community structures at time $t-1$ and t to ensure the smoothness of community structures across time. In PCM, α is set *a priori* empirically. Furthermore, with this formulation, α is time independent, which implies that the smoothness of community structure is the same across time. However, in real-world networks community structures may vary in a non-stationary manner.

2.2.2 Dynamic Stochastic Blockmodeling

Recently, SBM and DCSBM have been extended to dynamic networks by defining a dynamic process to model the evolution of either the community structure or the parameters of the model. In this work, the focus is on models that define a dynamic process on the evolution of community structure [264, 84, 181]. These models can be defined as follows:

Definition 2.1. Dynamic DCSBM is a generative dynamic network model with community structures $\mathcal{g} = \{\mathbf{g}^1, \ldots, \mathbf{g}^T\}$ and edge parameter matrices $\vartheta = \{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^T\}$ where

(i) The network at each time t is generated by a DCSBM with connectivity matrix $\boldsymbol{\theta}^t$ and community structure $\mathbf{g}^t$.

(ii) Community assignments follow a first order Markov Process such that:

$$\text{P}(\mathcal{g}) = \text{P}(\mathbf{g}^T|\mathbf{g}^{T-1}) \ldots \text{P}(\mathbf{g}^2|\mathbf{g}^1)\text{P}(\mathbf{g}^1), \tag{2.2}$$

where the transition probability $P(\mathbf{g}^t|\mathbf{g}^{t-1})$ describes the evolution of community structure and $P(\mathbf{g}^1)$ is prior probability for the first time point.

2.2.3 Pairwise Markov Random Fields

One way to model the joint distribution of a set of random variables is graphical models, where the nodes of the graph represent random variables and edges indicate the dependence between them. When the edges are undirected, the corresponding graphical model is Markov Random Field (MRF) [160]. Let $\mathbf{x} = \{x_1, \ldots, x_n\}$ be a set of random variables with joint distribution $P(\mathbf{x})$. MRF defines $P(\mathbf{x})$ as proportional to the product of non-negative parametric *potential functions* defined over maximal cliques of the graph. Instead of maximal cliques, one can also define the potential function over the edges of a graph [160]. Let $\psi_{ij}(x_i, x_j)$ be the potential function defined over the edges of the MRF graph. The joint distribution is then defined as:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{i,j:e_{ij} \in E_{MRF}} \psi_{ij}(x_i, x_j), \tag{2.3}$$

where E_{MRF} is the edge set of MRF graph and Z is the *partition function* to normalize the product. This type of MRF is called *pairwise MRF* and is widely used because of its simplicity [160]. Pairwise MRF has been used in community detection [198, 124, 93] by defining the potential functions following the *Potts Model*, where $\psi_{ij}(g_i, g_j) = \exp(J_{ij}\delta_{g_i g_j})$. J_{ij} indicates the belief about nodes i and j being in the same community, i.e. the larger it is the more likely nodes i and j are in the same community.

2.3 Dynamic MRF-DCSBM and Log-Posterior Formulation

In this section, dynamic MRF-DCSBM that extends DCSBM to dynamic networks is introduced. Different from previous works, the evolution of community structure is defined using a fully connected pairwise MRF. We then show the equivalence between the log-posterior function of the proposed model and regularized association function given in (2.1).

2.3.1 Dynamic MRF-DCSBM

Previous works on dynamic DCSBM differ from each other based on how they define the transition probabilities $P(\mathbf{g}^t|\mathbf{g}^{t-1})$ in (2.2). The most popular approach is to define the transition probabilities independently for each node [85, 181], i.e. $P(\mathbf{g}^t|\mathbf{g}^{t-1}) = \prod_{i=1}^{n} P(g_i^t|g_i^{t-1})$, where

$$P(g_i^t|g_i^{t-1}) = p^t \delta_{g_i^t g_i^{t-1}} + \frac{1-p^t}{K}, \tag{2.4}$$

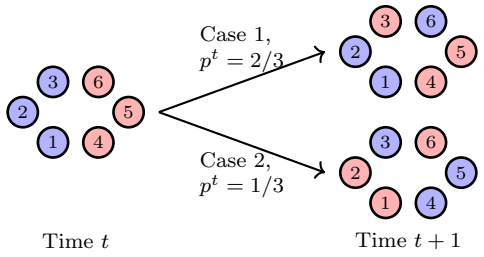

Figure 2.1: "Label-switching" issue between consecutive time points when transition probabilities $\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})$ is defined as (2.4).

which assumes that at each time a node either preserves its community membership with *copying probability* p^t or moves to one of the communities in a uniformly random manner. This model implicitly assumes that there is a one to one correspondence between communities at time $t-1$ and t. However, this is hardly the case as the rth community at time $t-1$ is not necessarily the rth community at time t. To elaborate on this problem, consider Figure 2.1, where the community structure of a dynamic network at two consecutive time points is shown. At time t, there are two communities indicated by blue and red. At time $t+1$, we consider two cases. In case 1, nodes 3 and 6 change their community memberships while the remaining nodes preserve their community memberships, which means $p^t = 2/3$. In case 2, the community structure is the same as in case 1, even though the community labels are different. In this case, one can say that only two nodes preserve their community memberships, which implies $p^t = 1/3$. Although community structures in both cases are the same, one obtains two different values of p^t due to label switching. This problem also exists when one uses transition matrices for $\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})$ which is discussed in [138] in detail.

To address this problem, in this work, $\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})$ is modeled with a fully-connected pairwise MRF where the potential functions are determined based on the community structure at the previous time point. For every node pair, a potential function $\psi_{ij}(g_i^t, g_j^t; \mathbf{g}^{t-1})$ is defined and $\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})$ is written as:

$$\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1}) = \frac{1}{Z}\prod_{i<j}^{n}\psi_{ij}(g_i^t, g_j^t; \mathbf{g}^{t-1}). \tag{2.5}$$

Following previous work that employs pairwise MRF for community detection, the potential functions are determined by Potts model, i.e., $\psi_{ij}(g_i^t, g_j^t; \mathbf{g}^{t-1}) = \exp(J_{ij}^t \delta_{g_i g_j}^t)$ where J_{ij}^t indicates the belief that two nodes i and j are in the same community at time t. We propose to determine J_{ij}^t based on $\mathbf{g}^{t-1}$ as follows:

$$J_{ij}^t = \begin{cases} J_{in}^t, & \text{if } g_i^{t-1} = g_j^{t-1} \\ J_{out}^t, & \text{if } g_i^{t-1} \neq g_j^{t-1} \end{cases}. \tag{2.6}$$

J^t_{in} refers to the belief that two nodes are in the same community at time t given that they were in the same community at time $t-1$. Similarly, J^t_{out} refers to the belief that two nodes are in the same community at time t given that they were in different communities at time $t-1$. In cases where community structure changes slowly across time, J^t_{in} is expected to be large while J^t_{out} is small. On the other hand, when there is an abrupt change in community structure, J^t_{in} decreases while J^t_{out} increases. Thus, the conditional distribution $P(\mathbf{g}^t|\mathbf{g}^{t-1})$ is able to adapt to the dynamics of community structure across time with proper selection of J^t_{in} and J^t_{out}. The selection of J^t_{in} and J^t_{out} will be discussed in Section 2.4.

2.3.2 Log-posterior Maximization of Dynamic MRF-DCSBM

One can infer the dynamic community structure by fitting dynamic MRF-DCSBM to an observed dynamic network $\mathcal{G}$. Given parameters ϑ, J^t_{in}, J^t_{out} $\forall t$ and the number of communities K, the community structure can be detected by maximizing the posterior probability $\mathrm{P}(g|\mathcal{A};\vartheta,J_{in},J_{out})$ with respect to g. Based on Definition 2.1 and given the pairwise MRF model for $\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})$, the posterior distribution is written as:

$$\begin{aligned}\mathrm{P}(g|\mathcal{A};\vartheta,J_{in},J_{out}) &= \prod_{t=1}^{T}\mathrm{P}(\mathbf{A}^t|\mathbf{g}^t,\boldsymbol{\theta}^t)\prod_{t=2}^{T}\mathrm{P}(\mathbf{g}^t|\mathbf{g}^{t-1})\mathrm{P}(\mathbf{g}^1)\\ &\propto \prod_{t=1}^{T}\prod_{i<j}^{n}\frac{(\lambda^t_{ij})^{A^t_{ij}}\exp(-\lambda^t_{ij})}{A^t_{ij}!}\prod_{t=2}^{T}\prod_{i<j}^{n}\exp(J^t_{ij}\delta^t_{g_ig_j}),\end{aligned} \tag{2.7}$$

where $\lambda^t_{ij}=d^t_i d^t_j\theta^t_{g^t_i g^t_j}$ and $\mathrm{P}(\mathbf{g}^1)$ is ignored in the second line since it is set to be a uniform distribution. Instead of maximizing the posterior probability, one can maximize its logarithm to find the community structure. Let $\mathcal{L}(g)$ be the logarithm of the posterior distribution, which can be written as:

$$\mathcal{L}(g)\propto\sum_{t=1}^{T}\sum_{i<j}^{n}\left\{A^t_{ij}\log(\lambda^t_{ij})-\lambda^t_{ij}\right\}+\sum_{t=2}^{T}\sum_{i<j}^{n}J^t_{ij}\delta^t_{g_ig_j}, \tag{2.8}$$

where terms that do not depend on g are ignored. Assuming a planted partition model, i.e $\theta^t_{kl}=\theta^t_{in}$ if $k=l$ and θ^t_{out}, otherwise, λ^t_{ij} and $\log(\lambda^t_{ij})$ can be written as:

$$\lambda^t_{ij}=d^t_i d^t_j\left\{(\theta^t_{in}-\theta^t_{out})\delta^t_{g_ig_j}+\theta^t_{out}\right\}, \tag{2.9}$$

$$\log\lambda^t_{ij}=\log(d^t_i d^t_j)+(\log\theta^t_{in}-\log\theta^t_{out})\delta^t_{g_ig_j}+\log\theta^t_{out}. \tag{2.10}$$

Substituting this into the first term of (2.8) and ignoring terms that do not depend on g, the following can be written at each time point t [167]:

$$\sum_{i<j}^{n}(A^t_{ij}\log(\lambda^t_{ij})-\lambda^t_{ij}))\propto\sum_{i<j}^{n}(\beta^t A^t_{ij}-\gamma^t d^t_i d^t_j)\delta^t_{g_ig_j}, \tag{2.11}$$

where $\beta^t = \log \theta^t_{in} - \log \theta^t_{out}$ and $\gamma^t = \theta^t_{in} - \theta^t_{out}$. It is easy to see that the right hand side of the above equation is in the form of (1.3) and it can be written in terms of a trace operator.

If J^t_{ij}s are set according to (2.6), then:

$$J^t_{ij} = J^t_{in}\delta^{t-1}_{g_i g_j} + J^t_{out}(1 - \delta^{t-1}_{g_i g_j}). \tag{2.12}$$

Substituting (2.12) and (2.11) into (2.8), the log-posterior can be written as:

$$\mathcal{L}(g) = \sum_{t=1}^{T}\sum_{i<j}^{n}(\beta^t A^t_{ij} - \gamma^t d^t_i d^t_j)\delta^t_{g_i g_j} + \sum_{t=2}^{T}\sum_{i<j}^{n}(J^t_{in}\delta^{t-1}_{g_i g_j} + J^t_{out}(1 - \delta^{t-1}_{g_i g_j}))\delta^t_{g_i g_j}. \tag{2.13}$$

Theorem 2.1. Given a $K \times K$ matrix of pairwise beliefs, $\mathbf{J}^t$, with diagonal entries, J^t_{in}, and off-diagonal entries, J^t_{out}, at each time point t and assuming $\mathbf{Z}^{t\top}\mathbf{D}^t\mathbf{Z}^t = \mathbf{I}\ \forall t \in 1, \ldots, T$, the log-posterior of dynamic MRF-DCSBM can be written as:

$$\mathcal{L}(g) \propto \sum_{t=1}^{T}\beta^t \mathrm{tr}(\mathbf{Z}^{t\top}\mathbf{A}^t\mathbf{Z}^t) + \sum_{t=2}^{T}\mathrm{tr}(\mathbf{Z}^{t\top}\mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1\top}\mathbf{Z}^t). \tag{2.14}$$

Proof. To prove the theorem, we need to show that the degree term in (2.13) at any time t is a constant when the constraint $\mathbf{Z}^{t\top}\mathbf{D}^t\mathbf{Z}^t = \mathbf{I}$ is imposed on $\mathbf{Z}^t$. In the following derivations, we will ignore the superscript t. Let κ_r represent the total degree of community r. The degree term in (2.13) can then be rewritten as:

$$\gamma\sum_{i<j}^{n} d_i d_j \delta_{g_i g_j} = \frac{\gamma}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_i d_j \delta_{g_i g_j} - \frac{\gamma}{2}\sum_{i=1} d_i^2. \tag{2.15}$$

As the second term does not depend on community structure, it can be ignored. Since $\delta_{g_i g_j} = \sum_{r=1}^{K} Z_{ir}Z_{jr}$, the first term can written as:

$$\begin{aligned}
\frac{\gamma}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_i d_j \delta_{g_i g_j} &= \frac{\gamma}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_i d_j \sum_{r=1}^{K} Z_{ir}Z_{jr}, \\
&= \frac{\gamma}{2}\sum_{r=1}^{K}\sum_{i=1}^{n} d_i Z_{ir}\sum_{j=1}^{n} d_j Z_{jr}, \\
&= \frac{\gamma}{2}\sum_{r=1}^{K}\kappa_r^2 \qquad (2.16) \\
&= \frac{\gamma}{2}\mathrm{tr}(\mathbf{Z}^\top\mathbf{D}\mathbf{Z}\mathbf{Z}^\top\mathbf{D}\mathbf{Z}), \qquad (2.17)
\end{aligned}$$

where in the last step we used the fact $\mathbf{Z}^\top\mathbf{D}\mathbf{Z}$ is a $K \times K$ diagonal matrix with entries $(\mathbf{Z}^\top\mathbf{D}\mathbf{Z})_{rr} = \kappa_r$. When the constraint $\mathbf{Z}^\top\mathbf{D}\mathbf{Z} = \mathbf{I}$ is imposed, the right hand side of (2.16) becomes a constant. □

The final form of log-posterior in (2.14) is equivalent to PCM formulation in (2.1) for each time point t. The only difference between the two expressions is that in this case the forgetting factors are not arbitrary and are determined by β^t and $\mathbf{J}^t$, which can be calculated through the parameters of dynamic DCSBM as will be shown in the next section. Moreover, the forgetting factors are time-varying allowing the algorithm to adapt to changes in the community structure. Thus, the selection of the optimal forgetting factor is circumvented through this equivalence. Finally, the effect of β^t and $\mathbf{J}^t$ in the quality function (2.14) can be explained as follows. Since β^t is equal to $\log(\theta^t_{in}/\theta^t_{out})$, as $\theta^t_{in}/\theta^t_{out}$ increases, β^t gets larger. A large $\theta^t_{in}/\theta^t_{out}$ implies that the communities are well separated from each other, thus, it is desirable to emphasize the first term in (2.14) which is achieved by a large β^t. On the other hand, when $\theta^t_{in}/\theta^t_{out}$ is small, β^t will be small as $\mathbf{A}^t$ has a less clear community structure. These observations are in line with [181], where equivalence between multislice modularity and dynamic DCSBM is shown. Similarly, J^t_{in} and J^t_{out} determine the importance of the second term. For example, if most of the nodes preserve their community memberships from time $t-1$ to t, J^t_{in} needs to be large, while J^t_{out} needs to be small. However, if there is a large variation in the community structure from time $t-1$ to t, J^t_{in} needs to be small to reduce the weight of the second term. In Section 2.4, we describe a methodology to select J^t_{in} and J^t_{out} so that the second term in (2.14) adapts to the evolution of community structure.

2.4 Dynamic Spectral Clustering

The equivalence shown in (2.14) allows for the development of two spectral clustering type algorithms, i.e. online and offline dynamic spectral clustering. In online learning, communities at each time point are determined given the community structure at the previous time point. This approach is applicable to real-time streaming networks. On the other hand, offline learning identifies the community structure at each time point given the community structure at the previous and next time points and it is applicable when network data is available for all times.

2.4.1 Algorithms

Online Learning (DSC_{on}) In real-time applications, one has network data only up to time point t. Given the community structures up to time $t-1$, $\mathbf{g}^t$ can be found by maximizing $\mathcal{L}(g)$ with respect to $\mathbf{Z}^t$ considering

Algorithm 2.1 Online Dynamic Spectral Clustering

$\mathcal{A}$: Adjacency matrices of the dynamic network, $\mathcal{K}$: Set of candidate values for number of communities, MaxIter: Number of iterations for parameter estimation ▷

function ONLINE($\mathcal{A}$, $\mathcal{K}$, MaxIter)
  $\mathbf{g}^1 \leftarrow$ SPECCLUS($(\mathbf{D}^1)^{-0.5}\mathbf{A}^1(\mathbf{D}^1)^{-0.5}$, $\mathcal{K}$)
  for $t \leftarrow 2, \ldots, T$ **do**
    Initialize θ^t_{in}, θ^t_{out}, J^t_{in} and J^t_{out} as in Section 2.4.2
    for $i \leftarrow 1, \ldots$ MaxIter **do**
      Construct $\mathbf{A}^t_{on}$ as in (2.18)
      $\mathbf{g}^t \leftarrow$ SPECCLUS($\mathbf{D}^{t\,-0.5}\mathbf{A}_{on}\mathbf{D}^{t\,-0.5}$, $\mathcal{K}$)
      Estimate θ^t_{in}, θ^t_{out}, J^t_{in} and J^t_{out} as in (2.21), (2.22), (2.27), and (2.28)
    end for
  end for
  return g, ϑ, $\{J^2_{in}, \ldots, J^T_{in}\}$, $\{J^2_{out}, \ldots, J^T_{out}\}$
end function

function SPECCLUS($\mathbf{A}, \mathcal{K}$)
  $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top \leftarrow$ Eigendecomposition of $\mathbf{A}$
  $Q^*_{mas} \leftarrow -\infty$
  for $K \in \mathcal{K}$ **do**
    $\mathbf{g} \leftarrow kmeans(\mathbf{U}(:, 1:K), K)$
    if $Q_{mas}(\mathbf{g}) > Q^*_{mas}$ **then**
      $\mathbf{g}^* \leftarrow \mathbf{g}$
      $Q^*_{mas} \leftarrow Q_{mas}(\mathbf{g})$
    end if
  end for
  return $\mathbf{g}^*$
end function

only the terms that depend on $\mathbf{Z}^t$. The corresponding optimization problem is:

$$\begin{aligned} \underset{\mathbf{Z}^t}{\text{maximize}} \quad & \text{tr}(\mathbf{Z}^{t\,\top}(\beta^t\mathbf{A}^t + \mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1\,\top})\mathbf{Z}^t), \\ \text{subject to} \quad & \mathbf{Z}^{t\,\top}\mathbf{D}^t\mathbf{Z}^t = \mathbf{I} \end{aligned} \tag{2.18}$$

where the modified adjacency matrix at each time point can be defined as $\mathbf{A}^t_{on} = \beta^t\mathbf{A}^t + \mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1\,\top}$ with the initialization $\mathbf{A}^1_{on} = \mathbf{A}^1$. $\mathbf{Z}^t$ that maximizes (2.18) is the matrix whose columns are the K eigenvectors that correspond to the largest K eigenvalues of $\mathbf{D}^{t\,-0.5}\mathbf{A}^t_{on}\mathbf{D}^{t\,-0.5}$. $\mathbf{g}^t$ is then found by applying k-means to the rows of this $\mathbf{Z}^t$. Pseudocode for this algorithm is given in Algorithm 1.

Offline Learning (DSC_{off}) In some applications, e.g. dynamic social networks, one might have access to network data for all time points. In this case, both past and future data can be used to identify the communities at time t. Given the community structures at time $t-1$ and $t+1$, $\mathbf{g}^t$ can be found by maximizing $\mathcal{L}(g)$ with

Algorithm 2.2 Offline Dynamic Spectral Clustering

$\mathcal{A}$: Adjacency matrices of dynamic network, $\mathcal{K}$: Set of candidate values for number of communities, MaxIter: Number of iterations for parameter estimation ▷

function OFFLINE($\mathcal{A}$, $\mathcal{K}$, MaxIter)
  g, ϑ, $\{J_{in}^2, \ldots, J_{in}^T\}$, $\{J_{out}^2, \ldots, J_{out}^T\} \leftarrow$ ONLINE($\mathcal{A}$, $\mathcal{K}$, 1) ▷ Initialization
  for $i \leftarrow 1, \ldots,$ MaxIter **do**
    for $t \leftarrow 1, \ldots, T$ **do**
      Construct $\mathbf{A}_{off}^t$ as in (2.20)
      $\mathbf{g}^t \leftarrow$ SPECCLUS($\mathbf{D}^{t\,-0.5}\mathbf{A}_{off}\mathbf{D}^{t\,-0.5}$, $\mathcal{K}$)
      Estimate θ_{in}^t, θ_{out}^t, J_{in}^t and J_{out}^t as in (2.21), (2.22), (2.27), and (2.28)
    end for
  end for
  return g, ϑ, $\{J_{in}^2, \ldots, J_{in}^T\}$, $\{J_{out}^2, \ldots, J_{out}^T\}$
end function

respect to $\mathbf{Z}^t$:

$$\begin{aligned} \underset{\mathbf{Z}^t}{\text{maximize}} \quad & \beta^t \operatorname{tr}(\mathbf{Z}^{t\top}\mathbf{A}^t\mathbf{Z}^t) + \operatorname{tr}(\mathbf{Z}^{t\top}\mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1\top}\mathbf{Z}^t) + \operatorname{tr}(\mathbf{Z}^{t+1\top}\mathbf{Z}^t\mathbf{J}^{t+1}\mathbf{Z}^{t\top}\mathbf{Z}^{t+1}), \\ \text{subject to} \quad & \mathbf{Z}^{t\top}\mathbf{D}^t\mathbf{Z}^t = \mathbf{I} \end{aligned} \tag{2.19}$$

where the last term can be rewritten as:

$$\begin{aligned} \operatorname{tr}(\mathbf{Z}^{t+1\top}\mathbf{Z}^t\mathbf{J}^{t+1}\mathbf{Z}^{t\top}\mathbf{Z}^{t+1}) &= \sum_{i<j}^{n} (J_{in}^{t+1}\delta_{g_ig_j}^t + J_{out}^{t+1}(1-\delta_{g_ig_j}^t))\delta_{g_ig_j}^{t+1}, \\ &= \sum_{i<j}^{n} (J_{in}^{t+1}\delta_{g_ig_j}^{t+1} - J_{out}^{t+1}\delta_{g_ig_j}^{t+1})\delta_{g_ig_j}^t + J_{out}^{t+1}\delta_{g_ig_j}^{t+1}, \\ &= (J_{in}^{t+1} - J_{out}^{t+1})\operatorname{tr}(\mathbf{Z}^{t\top}\mathbf{Z}^{t+1}\mathbf{Z}^{t+1\top}\mathbf{Z}^t) + \sum_{i<j}^{n} J_{out}^{t+1}\delta_{g_ig_j}^{t+1}. \end{aligned}$$

When this term is used in the maximization problem in (2.19) for time t, the second term in the last line can be ignored since it does not depend on $\mathbf{Z}^t$. Thus, we can change the last term in (2.19) with $(J_{in}^{t+1} - J_{out}^{t+1})\operatorname{tr}(\mathbf{Z}^{t\top}\mathbf{Z}^{t+1}\mathbf{Z}^{t+1\top}\mathbf{Z}^t)$. With this change, $\mathbf{g}^t$ can be found by applying k-means to K eigenvectors that correspond to the K largest eigenvalues of the matrix $\mathbf{D}^{t\,-0.5}\mathbf{A}_{off}^t\mathbf{D}^{t\,-0.5}$, where $\mathbf{A}_{off}^t$ is:

$$\mathbf{A}_{off}^t = \beta^t\mathbf{A}^t + \mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1\top} + (J_{in}^{t+1} - J_{out}^{t+1})\mathbf{Z}^{t+1}\mathbf{Z}^{t+1\top}. \tag{2.20}$$

For $t = 1$, the second term in $\mathbf{A}_{off}^t$ is excluded when calculating $\mathbf{A}_{off}^1$ and for $t = T$, the last term is excluded. Pseudocode for this algorithm is given in Algorithm 2.

2.4.2 Parameter Estimation

The proposed algorithms are derived based on the assumption that the parameters of dynamic DCSBM are known. However, in practice, one needs to estimate the parameters of the model while inferring the

community structure. Based on previous works in [167, 181, 192], we use an iterative scheme to estimate intra- and inter-community connectivity parameters, θ^t_{in} and θ^t_{out}, and pairwise MRF parameters, J^t_{in} and J^t_{out}, for all time points.

At each time point, first, θ^t_{in} and θ^t_{out} are initialized such that $\beta^t = 1$ and J^t_{in} and J^t_{out} are respectively set to s^t_{in} and s^t_{out} defined below. Next, community structure is found with these values. Then, intra- and inter-community connectivity parameters, θ^t_{in} and θ^t_{out}, are estimated with maximum likelihood estimation using the detected communities as follows [167]:

$$\theta^t_{in} = \frac{\sum_{i<j}^{n^t} A^t_{ij}\delta^t_{g_i g_j}}{\sum_{i<j}^{n^t} d^t_i d^t_j \delta^t_{g_i g_j}}, \tag{2.21}$$

$$\theta^t_{out} = \frac{\sum_{i<j}^{n^t} A^t_{ij}(1-\delta^t_{g_i g_j})}{\sum_{i<j}^{n^t} d^t_i d^t_j (1-\delta^t_{g_i g_j})}. \tag{2.22}$$

Similarly, J^t_{in} and J^t_{out} can be estimated using any of the approaches developed to learn the parameters of an MRF. However, estimation of J^t_{in} and J^t_{out} depends on calculating the partition function of MRF, which is not an easy task [160]. Instead of relying on these methods, we propose a procedure using the following statistics on network and community structure to estimate J^t_{in} and J^t_{out}:

$$p^t_{in} = \frac{\sum_{i<j}^{n} \delta^t_{g_i g_j}\delta^{t-1}_{g_i g_j}}{\sum_{i<j}^{n} \delta^{t-1}_{g_i g_j}}, \tag{2.23}$$

$$p^t_{out} = \frac{\sum_{i<j}^{n} \delta^t_{g_i g_j}(1-\delta^{t-1}_{g_i g_j})}{\sum_{i<j}^{n} 1-\delta^{t-1}_{g_i g_j}}, \tag{2.24}$$

$$s^t_{in} = \frac{\sum_{i<j}^{n} A^t_{ij}\delta^{t-1}_{g_i g_j}}{\sum_{i<j}^{n} \delta^{t-1}_{g_i g_j}}, \tag{2.25}$$

$$s^t_{out} = \frac{\sum_{i<j}^{n} A^t_{ij}(1-\delta^{t-1}_{g_i g_j})}{\sum_{i<j}^{n} 1-\delta^{t-1}_{g_i g_j}}, \tag{2.26}$$

where p^t_{in} is the probability of node pairs remaining in the same community from time $t-1$ to t and p^t_{out} is the probability of node pairs moving into the same community at time t when they are not in the same community at time $t-1$. s^t_{in} and s^t_{out} quantify the intra- and inter-community sparsity levels of $\mathbf{A}^t$ using the community structure at the previous time point, respectively.

For any node pair, J^t_{in} and J^t_{out} correspond to the belief that a node pair is in the same community at time t given the community structure at $t-1$. Since p^t_{in} and p^t_{out} quantify the ratios of node pairs that stay in or move into the same community from time $t-1$ to t, they can be used as indicators for the belief about a pair of nodes being in the same community. Moreover, as the pairwise MRF is fully connected and real

world networks are generally sparse, sparsity terms are used to ensure that the two terms in (2.14) are in the same range. Thus, J^t_{in} and J^t_{out} are defined as follows:

$$J^t_{in} = p^t_{in} s^t_{in}, \tag{2.27}$$

$$J^t_{out} = p^t_{out} s^t_{out}. \tag{2.28}$$

The community structure is then updated using the estimated values of θ^t_{in}, θ^t_{out}, J^t_{in} and J^t_{out}. This process is iterated until convergence, i.e. either the community structure or the parameters do not change anymore, or until the maximum number of iterations is reached.

2.4.3 Number of Communities

Determining the number of communities is an important part of community detection problem. Different methods such as Bayesian Information Criterion (BIC) [209], Integrated Completed Likelihood (ICL) [58] or Minimum Description Length (MDL) [185], have been proposed in literature. In these approaches, first a range of possible number of communities, $\mathcal{K}$, is defined. Next, the number of communities is set as the value that optimizes the given criterion.

In this work, we use a quality function based on the linear combination of asymptotic surprise [244] and modularity with configuration null model [171]. Asymptotic surprise is a heuristic quality function for community detection and is defined as follows:

$$Q_{as} = m D_{KL}\left(\frac{m_{in}}{m} \middle\| \frac{M_{in}}{M}\right), \tag{2.29}$$

where D_{KL} is Kullback–Leibler divergence, m_{in} is the number of intra-community edges, m is the total number of edges, M_{in} is the number of possible intra-community edges and M is the total number of possible edges. Modularity with configuration null model, on the other hand, compares the number of intra-community edges in an observed network to the number of intra-community edges expected under a configuration null model and is defined as:

$$Q_{cn} = \sum_{i,j}^{n} (A_{ij} - \gamma \frac{d_i d_j}{2m}) \delta_{g_i g_j}. \tag{2.30}$$

Asymptotic surprise has been previously used as a model selection approach [244, 221]. However, it is known that it can overestimate the number of communities. On the other hand, modularity can underestimate the number of communities due to its resolution limit [80]. Therefore, we propose to maximize a linear

combination of both quality functions to determine the number of communities as:

$$K^* = \underset{K \in \mathcal{K}}{\operatorname{argmax}} \quad Q_{mas} := Q_{as} + Q_{cn}. \tag{2.31}$$

2.4.4 Extensions

In previous sections, the number of nodes and communities are assumed to be the same at all time points. However, in many real-world dynamic networks, both the number of nodes and communities may change over time. The dynamic MRF-DCSBM can handle the changes in the number of nodes in the following manner. Let j be a new node added to the network at time t. Since we do not have any information about the community of this node, we set $J^t_{ij} = 0$ for all $i \neq j$ when defining the transition distribution. In the case a node is removed from the network, all information about the removed node is discarded from the transition distribution. These updates to the transition distribution change the construction of $\mathbf{A}^t_{on}$ and $\mathbf{A}^t_{off}$ as follows. When a new node joins the network at time t, we add an all-zero row and column to $\mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1^\top}$ corresponding to the new node. When a node leaves the network, the row and column corresponding to that node is removed from $\mathbf{Z}^{t-1}\mathbf{J}^t\mathbf{Z}^{t-1^\top}$. Similar changes are applied to $(J^{t+1}_{in} - J^{t+1}_{out})\mathbf{Z}^{t+1}\mathbf{Z}^{t+1^\top}$, when there are different number of nodes at times t and $t+1$. Finally, changes in the number of communities do not affect the proposed model, as our transition distribution is defined based on whether two nodes are in the same community or not, and not on the actual community label.

2.4.5 Computational Complexity

The computational complexity of the proposed algorithms is governed by the cost of eigendecomposition, which has a computational complexity of $O(n^3)$ where n is the number of nodes. If I is the maximum number of iterations for parameter estimation, the total computational complexity of both algorithms is $O(TIn^3)$ since the eigendecomposition needs to be computed at each time point I times. However, in practice T and I are small compared to n, thus computational complexity of both algorithms are approximately $O(n^3)$.

2.5 Results

The proposed algorithms[1] are compared to state-of-the-art dynamic community detection methods for both simulated and real networks. We consider dynamic community detection methods developed using heuristically defined quality functions such as evolutionary spectral clustering (PCM) [46], multislice

[1]Implementations of both algorithms can be found at https://github.com/abdkarr/DynamicSpectralClustering

modularity based generalized Louvain (GL[2]) [181], PisCES[3] [126] and DYNMOGA[4] [78]. PisCES extends spectral clustering to dynamic networks by smoothing eigenvectors of adjacency matrices over time and applying k-means to smoothed eigenvectors. DYNMOGA [78] uses genetic algorithms to maximize a multi-objective optimization problem, whose objective function includes modularity as snapshot cost and Normalized Mutual Information (NMI) [57] as temporal cost. We also compare to dynamic SBM model based methods such as (DSBM$_{Xu}$[5]) [258] and (DSBM$_{Yang}$[6]) [264]. Among these methods, GL and PisCES learn communities in an offline manner, while the rest are online.

Parameters for the different algorithms are set as follows. For the proposed algorithms, $MaxIter$ is set to 20. For PCM, the forgetting factor is selected from the set $\{0.1, 0.15, 0.2, 0.25, 0.35, 0.4\}$ as the one that maximizes the normalized association. This range is selected based on prior empirical evidence. The implementation of GL follows [181] and learns the resolution (γ) and interlayer coupling (ω) parameters using the equivalence between multislice modularity and a variant of dynamic DCSBM. Furthermore, we use a multi-iteration version of GL such that GL is run until there is no improvement in multislice modularity and each run is initialized using the community structure detected from the previous run. Forgetting factor in PisCES is determined via cross-validation over the set $\{0.05, 0.1, 0.15, 0.2\}$ as recommended in [126]. Parameters of the genetic algorithms for DYNMOGA are set as $crossoverrate = 0.8$, $mutationrate = 0.2$, $populationsize = 200$ and $numberofgenerations = 200$. The parameters of DSBM$_{Xu}$ and DSBM$_{Yang}$ are set as recommended in the corresponding papers. Finally, the number of communities for the proposed algorithms and PCM is selected as described in Section 2.4.3. GL, PisCES and DYNMOGA have their own number of communities selection procedures. DSBM$_{Xu}$ and DSBM$_{Yang}$ do not describe a procedure for selecting the number of communities, thus for simulated networks, we give the true number of communities as an input.

2.5.1 Simulated Networks

Simulated networks are generated following the benchmark model described in [20]. This model will be referred to as multilayer generative model (*MLGM*)[7]. This benchmark generates a dynamic network

[2]https://github.com/roxpamfil/IterModMax
[3]https://www.andrew.cmu.edu/user/davidch/
[4]http://staff.icar.cnr.it/pizzuti/codes.html
[5]https://github.com/IdeasLabUT/Dynamic-Stochastic-Block-Model
[6]https://homepage.cs.uiowa.edu/~tyng/publications.html
[7]https://github.com/MultilayerGM/MultilayerGM-MATLAB

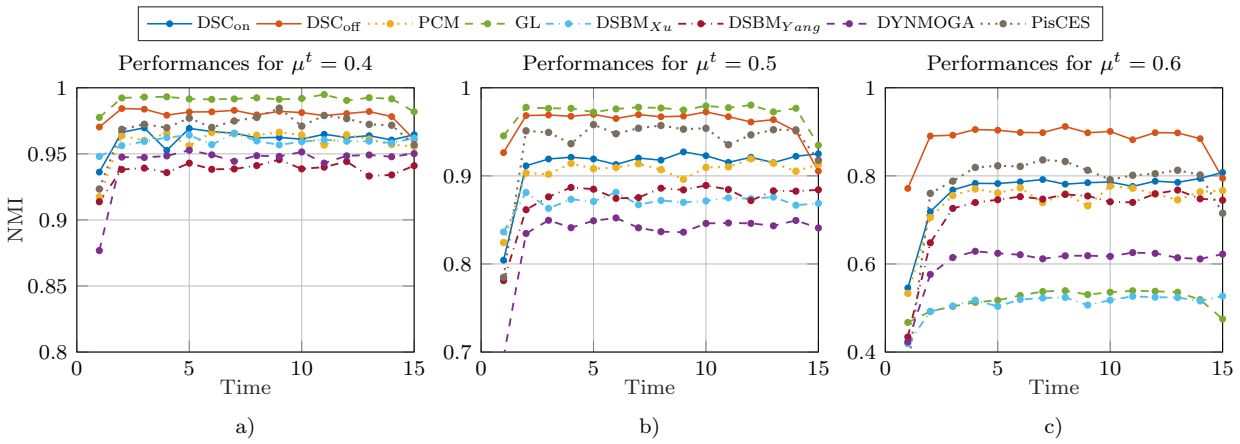

Figure 2.2: Simulation 1: Average NMI values for the different methods as a function of time. The mixing coefficient is set to (a) 0.4, (b) 0.5 and (c) 0.6.

using dynamic DCSBM described in Definition 2.1. Transition probabilities are set similar to (2.4) with the only difference being that the probability of nodes moving to other communities is determined by a categorical distribution instead of a uniform distribution. Probabilities of categorical distribution are drawn from a Dirichlet distribution with parameter ν. Node degrees are drawn from a truncated power law distribution with exponent k, minimum degree d_{min} and maximum degree d_{max} to obtain heterogeneous degree distributions. The parameters of the benchmark model are number of communities K, copying probability p^t and mixing coefficient μ^t, which indicates the ratio of edges that are set as inter-community edges. Finally, $q_1 \in [0, 1]$ and $q_2 \in [0, \infty)$ control death and birth rates of communities, respectively. At any time point, each community may disappear with probability q_1. The number of emerging communities is determined by a Poisson distribution with rate q_2. The performance of the different algorithms is quantified by Normalized Mutual Information (NMI) [57].

Simulation 1 In this simulation, we evaluate the effect of the mixing coefficient on the performance of different methods. A dynamic network with $T = 15$ and 128 nodes at each time point is generated with MLGM. Nodes are divided into $K = 4$ communities with $q_1 = 0$ and $q_2 = 0$ so that there are 4 communities at all time points. Copying probability is $p^t = 0.9\ \forall t$ with ν set to 100. Parameters of the power-law distribution are set as $k = -2.5$, $d_{min} = 8$ and $d_{max} = 16$. Aforementioned methods are applied to 100 realizations of networks generated using these settings. The number of communities is selected from $\mathcal{K} = \{2, \ldots, 10\}$ as the value that maximizes the proposed quality function, Q_{mas}. Average NMI as a function of time is reported in Figure 2.2 for three different values of the mixing coefficient, i.e., $\{0.4, 0.5, 0.6\}$. It can be seen

that offline learning methods, DSC$_{off}$, GL and PisCES, have higher accuracy than online approaches for $\mu^t = 0.4$ and $\mu^t = 0.5$, since they use both past and future networks to detect communities at any time point. Among offline methods, GL performs the best for $\mu^t = 0.4$. DSC$_{off}$ and GL have similar performances for $\mu^t = 0.5$ and perform better than PisCES. For $\mu^t = 0.6$, GL's accuracy drops significantly while the proposed offline algorithm achieves the best performance. Among online methods, DSC$_{on}$ shows similar or better performance compared to others. Methods based on dynamic SBM variants cannot detect the communities accurately except for a low mixing coefficient. The reason for this loss in accuracy is that both DSBM$_{Xu}$ and DSBM$_{Yang}$ require the selection of a set of hyperparameters and their performance depends on the correct estimation of these parameters. On the other hand, the proposed methods are hyperparemeter-free.

Simulation 2 In the previous simulation, the copying probability is set as a constant across time. However, in real-world dynamic networks, community structure can evolve at different rates across time. To generate such dynamic networks, we set $\mu^t = 0.5$ and consider three different cases with varying p^t as given in Table 2.1. Communities are detected with $\mathcal{K} = \{2, \ldots, 10\}$ and results are shown in Figure 2.3. For case 1, DSC$_{off}$ and GL have similar NMI values and perform slightly better than PisCES. The performances of all methods decrease between $t = 6$ and $t = 10$, where $p^t = 0.75$. This is expected as a small copying probability implies that the network is more non-stationary. DSC$_{off}$ performs the best among all methods for these time points. This implies that our method is more robust to changes in community membership. For case 2, the NMI values are lower for all methods compared to the first case as copying probability is further reduced. Compared to GL, the proposed offline method maintains its effectiveness, while the former's performance drops significantly. It can also be observed that DSC$_{off}$ detects communities better than PisCES. Similar results are observed for Case 3 in Figure 2.3c, where GL performs well only for the time range where $p_t = 0.9$ while DSC$_{off}$ maintains a high NMI value across all time. Among online methods, DSC$_{on}$ performs the best in all cases. In summary, the proposed methods are more stable across time and robust against changes in copying probability over time. This ability to adapt to changes in copying probability also indicates that

Table 2.1: Copying probability values for Simulation 2

	From $t = 2$ to $t = 5$	From $t = 6$ to $t = 10$	From $t = 11$ to $t = 15$
Case 1	0.90	0.75	0.90
Case 2	0.75	0.65	0.75
Case 3	0.80	0.90	0.70

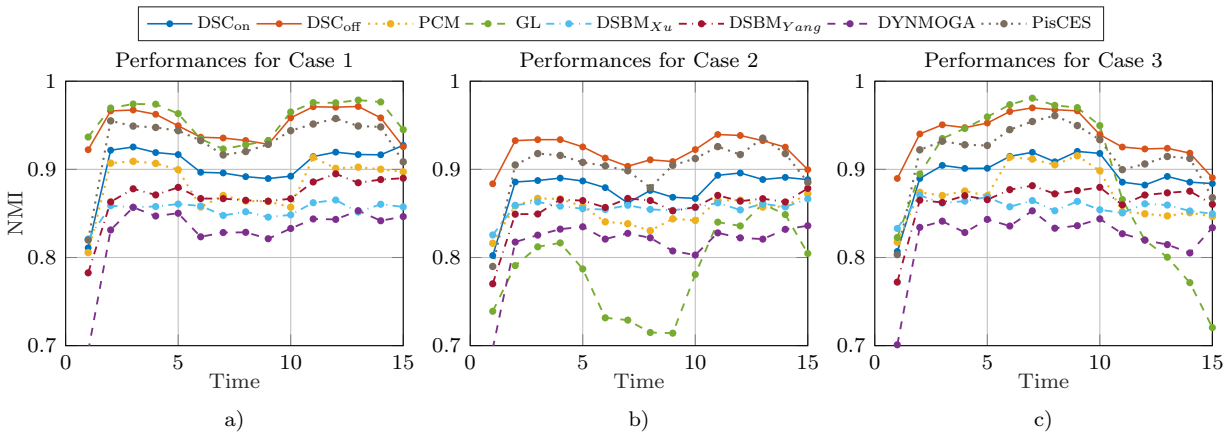

Figure 2.3: Simulation 2: Average NMI values for the different methods as a function of time. Case 1 (a), Case 2 (b) and Case 3 (c). The mixing coefficient is set to 0.5.

the method for selecting J_{in}^t and J_{out}^t proposed in Section 2.4.2 is effective.

Simulation 3 In most real world dynamic networks, the number of communities may also change over time. In this simulation, we evaluate the performance of different algorithms for changing number of communities using q_1 and q_2 to control the birth and death rates of communities. A dynamic network with $T = 15$ and 256 nodes is generated using MLGM benchmark. Number of communities for the first time point is set to $K = 8$. For subsequent time points, the number of communities is determined by $q_1 = 0.1$ and $q_2 = 1$. The remaining parameters are set as $k = -2.5$, $d_{min} = 8$, $d_{max} = 16$, $p^t = 0.9$ and $\mu^t = 0.5$. With these parameters, on average one new community emerges, one community disappears and 18% of nodes change their communities at each time.

In Figure 2.4, results are shown for the different methods with $\mathcal{K} = \{2, \ldots, 20\}$. We omitted the results for DSBM$_{Xu}$ and DSBM$_{Yang}$ as they do not perform well with changing number of communities across time. Figure 2.4a shows the average NMI as a function of time. The proposed offline algorithm performs the best compared to other offline and online methods. Among online methods, DSC$_{on}$ performs slightly better than PCM and both methods are better than DYNMOGA. Figure 2.4b illustrates the estimated number of communities for each method along with the true number of communities. It can be seen that GL and DYNMOGA overestimate the number of communities resulting in low NMI values. On the other hand, the number of communities estimated by the proposed methods along with PCM and PisCES are very close to the true number of communities. These results indicate that the quality function, Q_{mas}, proposed in Section 2.4.3 is effective at determining the number of communities.

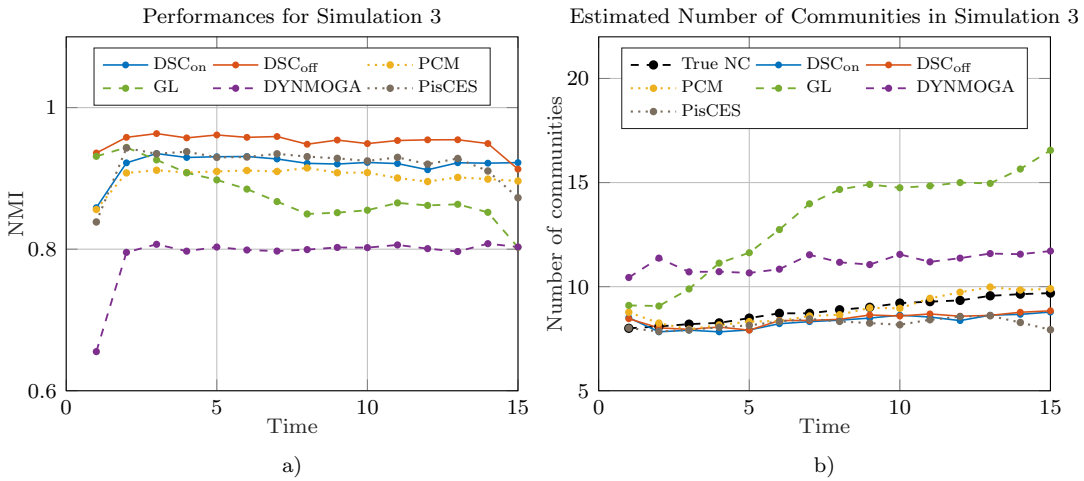

Figure 2.4: Results for 100 realizations of the network described in Simulation 3: (a) Average NMI as a function of time; (b) Estimated number of communities. Black dashed line is the true number of communities averaged over 100 realizations.

Scalability Analysis We compare the scalability of the aforementioned methods with increasing number of nodes. A dynamic network with $T = 10$ is generated using MLGM benchmark. Number of nodes are set to 2^m where m varies from 6 to 12 in increments of 1. Number of communities is set to 2^{m-5}, such that the average community size remains the same with increasing number of nodes. The remaining parameters are set as $k = -2.5$, $d_{min} = 8$, $d_{max} = 16$, $p^t = 0.9$, $\mu^t = 0.5$, $q_1 = 0$ and $q_2 = 0$.

The average run time for community detection is reported across 10 realizations in Figure 2.5. Number of communities is assumed to be known so the run time corresponds only to the time required for community detection and forgetting factor estimation. Results for DSBM$_{Xu}$ and DSBM$_{Yang}$ are not reported, as we could not obtain their results in a reasonable time for networks with more than 1000 nodes. It can be observed that the proposed methods have lower computational complexity than other methods for the considered network sizes. Although the proposed methods learn forgetting factor through an iterative scheme, these results indicate that this learning process does not have high computational complexity compared to methods that require *a priori* selection of forgetting factor. This is due to the fact that the learning algorithm converges fast.

2.5.2 Real World Networks

In this section, the proposed algorithms are applied to real-world dynamic networks and their performances are compared to aforementioned methods. As the number of communities may change over time, results for DSBM$_{Xu}$ and DSBM$_{Yang}$ are not reported since they do not perform well in such cases. For

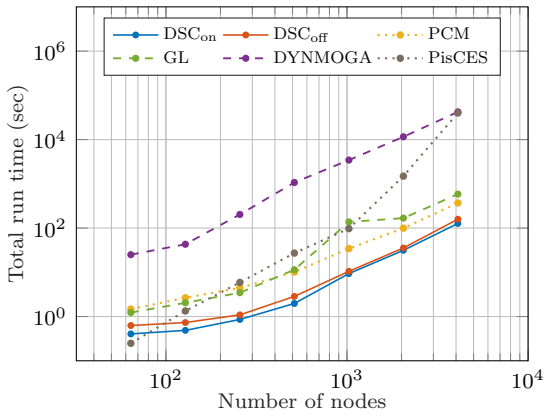

Figure 2.5: Computational complexity of methods with respect to number of nodes.

the first dataset, metadata about the nodes are used as ground truth community structure and performances are evaluated using NMI. For the remaining datasets, we do not have any information about ground truth community structure, thus we compare the detected communities using quality functions developed for community detection. We use three such metrics: modularity (see (2.30)) with resolution parameter set as [167]; asymptotic surprise (AS) and conductance [79], which quantifies the ratio of inter-community edges to the total degree. Smaller values of conductance indicate better community structure. These metrics are computed for the communities detected by each algorithm at each time point and averaged over time. Finally, parameters of the methods are set as in the simulations except that generation number and population size of DYNMOGA are reduced to 50 due to computational complexity.

Reality Mining This dynamic network is constructed by using the data from MIT Reality mining project [72]. The data is collected from cell phones of 94 students and staff at MIT over a year. Cell phones are equipped with Bluetooth sensors which record nearby Bluetooth devices every 5 minutes. These recordings are used to construct a dynamic network with 46 time points, each of which correspond to 1 week. Affiliations of students and staff are available and used as ground truth community structure as in [258]. Namely, there are 2 communities corresponding to people who work in MIT Media Lab and first-year business school students.

Table 2.2: Mean NMI values for detected communities of reality mining

	DSC$_{on}$	DSC$_{off}$	PCM	GL	DYNMOGA	PisCES
NMI	**0.677**(**0.005**)	0.652(0.014)	0.637(0.009)	**0.724**(**0.309**)	0.517(0.008)	0.466(0.016)

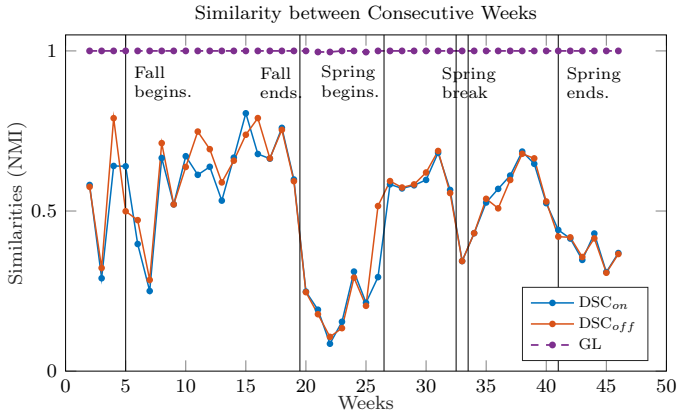

Figure 2.6: Similarity between the community structures at consecutive time points for reality mininig data.

Due to randomness in community detection algorithms, communities are found by running each algorithm 100 times. Number of communities are selected from $\mathcal{K} = \{2, \ldots, 10\}$. Average NMI over time and runs along with standard deviation across runs are reported in Table 2.2. The values that are significantly higher than the rest are given in bold, where significance is determined by t-test at $\alpha = 0.05$. The highest NMI values are obtained by GL and DSC$_{on}$ followed by DSC$_{off}$. Although the mean NMI value of GL is higher than DSC$_{on}$, standard deviation of the former is high and thus no significant difference is found between NMI values of the two methods.

The similarities between community structures at consecutive time points are calculated by NMI and plotted in Figure 2.6 to show how community structures detected by DSC$_{\text{on}}$ and DSC$_{\text{off}}$ change over time. The similarity of communities detected by GL across time is also reported. It can be seen that there are drops in similarities of DSC$_{\text{on}}$ and DSC$_{\text{off}}$ between weeks 20 and 25, around week 33 and after week 40. These drops are expected, since these weeks correspond to winter break, spring break and end of the school year, respectively [144]. These changes are not detected by GL.

Enron Email Data This dataset is a dynamic email communication network between Enron employees constructed by [258] using Enron corpus [191] that include 500, 000 emails from 1998 to 2002. A snapshot network is generated for each week by connecting two employees with an edge, if they communicated through an email. There are $T = 120$ time points and 184 nodes corresponding to the employees. More details about the dynamic network can be found in [258].

Community structure is found using aforementioned methods for 100 runs due to randomness in methods' outputs. Number of communities is estimated from the range $\mathcal{K} = \{2, 3, \ldots, 20\}$. Table 2.3 reports mean

Table 2.3: Conductance, Modularity and AS values of detected communities of Enron E-mail data

	Conductance	Modularity	AS
DSC$_{on}$	**1.060**(**0.072**)	**0.726**(**0.010**)	127.04(2.65)
DSC$_{off}$	1.311(0.094)	0.697(0.008)	132.80(2.02)
PCM	1.696(0.165)	0.658(0.015)	120.6(3.34)
GL	4.225(0.446)	0.421(0.008)	91.58(2.22)
DYNMOGA	1.160(0.024)	0.635(0.001)	**151.88**(**0.39**)
PisCES	4.648(0.060)	0.424(0.006)	63.79(1.79)

and standard deviation of conductance, modularity and AS. The values that are significantly better than the rest are given in bold, where significance is determined by t-test at $\alpha = 0.05$. In terms of conductance and modularity, the best values are obtained by DSC$_{on}$. In terms of asymptotic surprise, DYNMOGA followed by the proposed offline method outperform the rest.

As reported in [258], the structure of the network starts to change after week 89, due to the resignation of some of the CEOs and the federal investigation the company falls under. These changes include an increase in the number of edges and in the amount of communication between company's CEOs and presidents. To see how these changes affect the community structure, the similarity between community structures at consecutive time points is plotted in Figure 2.7. It is observed that the similarities increase after week 89 for both of the proposed methods. As the amount of communication between employees increases after week 89, the community structure becomes more stable across time due to increasing connectivity. This finding is in line with the results in [258].

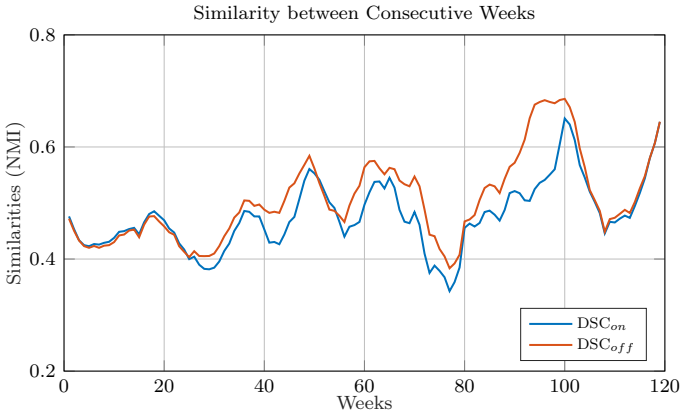

Figure 2.7: Similarity between the community structures at consecutive time points for Enron e-mail data. Moving average of the similarity is taken with a window size of 7 to reduce noise.

Table 2.4: Conductance, Modularity and AS values of detected communities of Day 1 of middle school data

	Conductance	Modularity	AS
DSC_{on}	3.799(0.212)	0.697(0.004)	4264.8(30.15)
DSC_{off}	**3.641**(**0.260**)	**0.699**(**0.004**)	4319.6(34.81)
PCM	4.105(0.300)	0.689(0.006)	4200.1(45.14)
GL	6.462(0.151)	0.652(0.001)	**4486.4**(**6.32**)
DYNMOGA	11.192(0.429)	0.607(0.004)	4061.7(25.00)
PisCES	11.031(0.183)	0.595(0.008)	2750.0(81.17)

Middle School Network The third real world network we consider is a dynamic social network between students of a middle school in Utah. The data is collected by [243] for two days between 8:25 a.m. and 3:15 p.m. The interactions between students are obtained by proximity sensors that have a time resolution of 20 seconds. 591 7th and 8th graders participated in the study. A school day consists of 7 class periods and 2 lunch times and students switch their classrooms between class periods. A dynamic network with $T = 28$ snapshots is generated for each day as in [221]. Each snapshot corresponds to a 15 minute interval and two students are connected with an edge if they interacted during this period.

Each of the community detection methods is applied to the constructed dynamic networks for each day for 100 runs. The set of candidate number of communities is set as $\mathcal{K} = \{10, 11, \ldots, 30\}$ for DSC_{on}, DSC_{off} and PCM. The mean and standard deviation of conductance, modularity and AS for all methods are reported in Tables 2.4 and 2.5 for the first and second days, respectively. The values that are significantly better than others are shown in bold, where statistical significance is determined as before. For the first day, DSC_{off} followed by DSC_{on} gives the best results in terms of conductance and modularity. GL performs the best in terms of asymptotic surprise followed by the proposed methods. For the second day, the proposed methods achieve the best performances in terms of all metrics.

Table 2.5: Conductance, Modularity and AS values of detected communities of Day 2 of middle school data

	Conductance	Modularity	AS
DSC_{on}	4.067(0.226)	0.686(0.004)	4398.8(31.14)
DSC_{off}	**3.962**(**0.210**)	**0.688**(**0.003**)	**4470.3**(**30.52**)
PCM	4.300(0.266)	0.677(0.006)	4319.3(51.32)
GL	14.555(1.558)	0.232(0.074)	1042.1(107.60)
DYNMOGA	11.255(0.443)	0.611(0.004)	4191.2(26.23)
PisCES	9.054(0.172)	0.646(0.005)	3272.4(69.66)

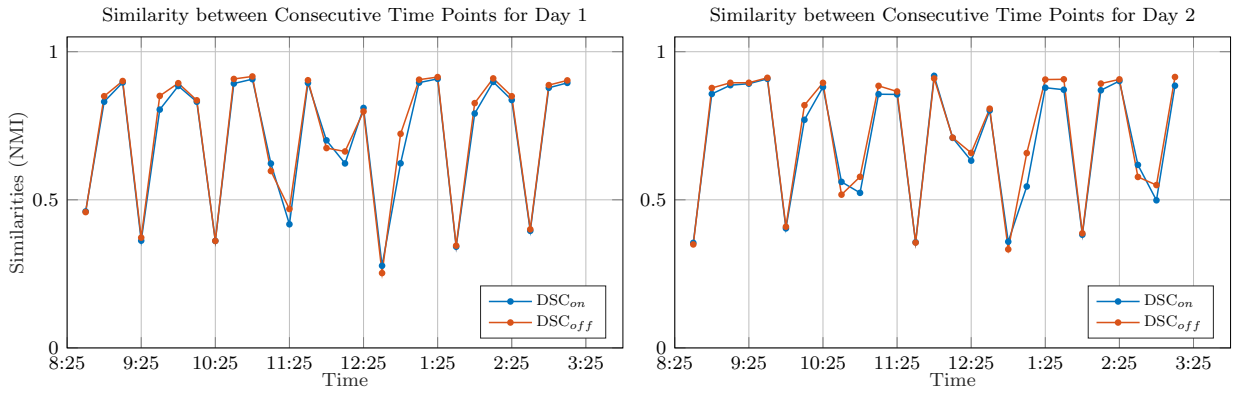

Figure 2.8: Similarity between detected communities at consecutive time points on first and second days of middle school data.

The community structure of the middle school network changes substantially during a day as the students switch their classrooms during breaks between class periods. In Figure 2.8, we plot the average NMI between detected communities at consecutive time points for both days. As can be seen from the figure, the similarity drops every hour or so corresponding to break times, which indicates the effectiveness of the proposed methods in tracking changes in the community structure.

DBLP Finally, we consider a dynamic co-authorship network generated from DBLP database by [10] and studied previously in [264, 125]. The dynamic network is generated from the papers published in 28 conferences over 10 years (1997 - 2006). A snapshot network is generated for each year by connecting two authors with an edge, if they co-authored a paper during that year. There are 958 authors in total. All of the methods are applied to the generated dynamic network for 20 runs. Candidate values for the number of communities are set as $\mathcal{K} = \{60, 61, \ldots, 120\}$ for DSC$_{on}$, DSC$_{off}$ and PCM. The means and standard deviations of conductance, modularity and AS are reported in Table 2.6. For each metric, the values that are significantly better than others are shown in bold, where significance is determined by t-test at $\alpha = 0.05$.

Table 2.6: Conductance, Modularity and AS values of detected communities of DBLP co-authorship data

	Conductance	Modularity	AS
DSC$_{on}$	**2.626**(**0.328**)	**0.722**(**0.018**)	4288.7(67.44)
DSC$_{off}$	**2.470**(**0.309**)	**0.729**(**0.014**)	4276.6(66.40)
PCM	3.872(0.884)	0.688(0.022)	4222.3(87.85)
GL	2.874(0.127)	0.675(0.003)	4052.2(27.37)
DYNMOGA	3.245(0.286)	0.716(0.003)	**4473.4**(**19.76**)
PisCES	72.573(0.551)	0.544(0.007)	3227.4(82.12)

In terms of conductance and modularity, best performances are obtained by the two proposed methods. In terms of AS, DYNMOGA followed by the proposed online method outperform the rest.

2.6 Conclusions

In this chapter, we investigated the equivalence between statistical inference and heuristic quality function based community detection methods for dynamic networks. In particular, we proposed a new dynamic MRF-DCSBM that captures the evolution of community membership. We showed that under the planted partition model, the log-posterior of MRF-DCSBM is equivalent to the objective function of evolutionary spectral clustering, where the weight of temporal smoothness is time dependent and adapts to the community structure. This equivalence resulted in the derivation of two new community detection methods. The proposed methods are shown to be more accurate at detecting the community structure when the network is noisy and more robust to non-stationarities in the network structure compared to state-of-the-art methods. The proposed methods are also shown to have superior performance on various real-world networks, where they are able to track changes in community structure over time.

Future work will consider approaches developed for parameter estimation in MRFs for estimating the parameters J_{in}^t and J_{out}^t. Moreover, the implementation of the proposed methods can be speeded up by faster eigendecomposition techniques and incremental spectral clustering [175].

CHAPTER 3

COMMUNITY DETECTION IN MULTILAYER NETWORKS

3.1 Introduction

Advances in neuroimaging technologies allow the brain to be modeled as a complex network, where the nodes correspond to the different brain units and the edges represent structural or functional connections among the units [37]. In order to characterize the topology and dynamics of brain networks, various descriptive and inferential network measures such as centrality, degree distribution and small-worldness [27, 156, 140, 18, 17] with respect to disease, task, learning, cognitive control, attention and memory [37, 33, 17, 140, 48, 22, 239, 218] are utilized. Current network models have been mostly limited to examining a single network instance either of a subject, a frequency band or a task. However, most neurophysiological recordings, such as the electroencephalogram (EEG), allows one to capture brain dynamics across multiple temporal and spatial scales. Reducing this rich information into a single network disregards the high amount of dependency that exists between networks of different subjects, frequency bands or tasks. Thus, a principled mathematical framework to accurately study this multiplicity of brain connectivity is needed.

Recently, multilayer networks have gained attention in network neuroscience [59, 61, 240, 24, 155, 247], due to their ability to represent and study multi-dimensional and multi-scale data. Initial work to model multiplicity of brain connectivity primarily employs multiplex networks, where the meaning of layer can vary depending on context, such as different modalities, subjects, and frequency bands. For example, Battiston *et al.* [19] introduce a two layer network combining structural and functional modalities using diffusion tensor imaging (DTI) and functional MRI (fMRI), respectively. Another line of work considers multiplex networks, where each layer corresponds to a different subject, to investigate intra- and inter-subject variability of brain connectivity [25]. Finally, multiplex networks where each layer corresponds to the connectivity in different frequency bands are considered to study the connectivity across multiple frequency bands, simultaneously [271, 214, 215, 56, 266]. While this line of work reveals important characteristics of multiplicity of brain connectivity, it restricts interlayer edges by using multiplex networks. Recently, this restriction on interlayer edges has been removed by modeling the brain connectivity using multilayer networks, where interlayer edges are allowed between any brain regions [65, 35, 240, 36]. For example, magnetoencephalography (MEG) [35, 36, 240] and EEG [162] recordings are used to construct functional multilayer networks, where each layer corresponds to the links within a frequency band, and the interlayer edges correspond to the

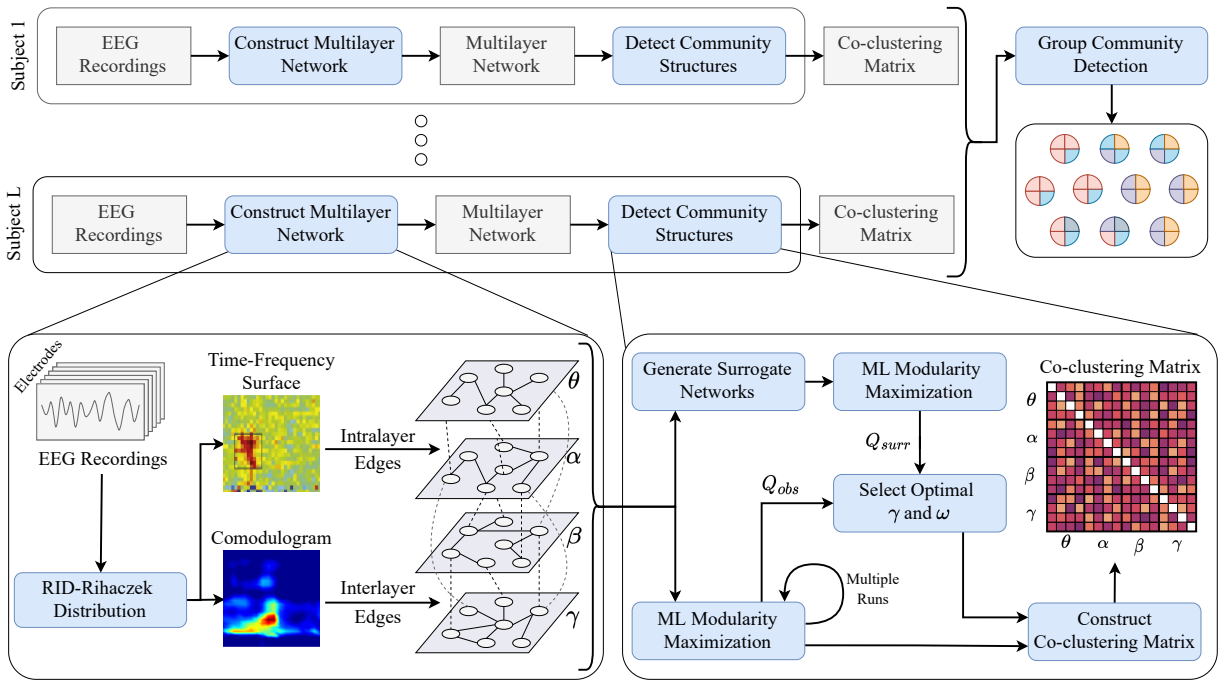

Figure 3.1: Flowchart of the proposed approach for community detection of multi-frequency EEG networks. Bottom two panels illustrate multilayer network construction (left) and community detection for each subject (right).

cross-frequency coupling across frequency bands.

Topological characteristics of multiplex and multilayer brain networks have been analyzed with various graph theoretical tools, such as hub node identification [61], motif analysis [19] and algebraic connectivity [36]. An important tool in the analysis of graphs is community detection [81]. Communities are defined as groups of nodes that are more strongly connected among themselves than they are to the rest of the network. Various community detection methods have been developed and applied to single-layer brain networks to find communities, which often correspond to specialized functional subnetworks of the brain [143, 232]. Although these methods can be applied to multiplex and multilayer graphs, they do not achieve good performance as they do not take the heterogeneity of connections across layers. Thus, recent work aims to extend community detection methods to these high-dimensional graphs [133, 194, 62, 189, 43]. However, most of these extensions are limited to multiplex networks except the following recent work. Pramanik *et al.* [189] extends the definition of modularity to multilayer networks. The proposed multilayer modularity metric is maximized using Girvan-Newman and Louvain algorithms. However, this approach does not take the resolution limit of modularity into account [189], limiting its practical use. Chen *et al.* [43], on the other hand, extends the definition of normalized cut to multilayer networks by constructing a block supra-Laplacian

matrix and proposes a spectral clustering algorithm based on this supra-Laplacian matrix. Although the method is developed for multilayer networks, it does not take the heterogeneity of interlayer edge weights into account.

In this chapter, we aim to characterize the topological organization of multilayer brain networks through multilayer community detection. In order to achieve this goal, we first construct multi-frequency networks from EEG data, where the intralayer and interlayer edges are quantified by previously published time-frequency phase synchrony [12] and phase amplitude coupling [157] measures, respectively. Thus, the constructed network is a multilayer network with interlayer edges allowed between all brain regions. Next, a new multilayer modularity metric is defined based on a multilayer null model that preserves the layer-wise node degrees while randomizing the remaining characteristics of the network. The proposed modularity is parameterized with resolution parameter to handle the resolution limit of modularity, and interlayer scale parameter to control the importance of interlayer edges in community formation. The optimal values of these parameters are determined using a surrogate data based procedure. Third, a group community detection method is proposed to find the common community structure for a set of subjects. The method uses subjects' co-clustering matrices obtained from multiple runs of modularity maximization, thus it is able to address the issue of degeneracy in modularity maximization [88]. Finally, the group level differences between the two response types during Flanker task, *i.e.* error and correct, are evaluated from a multi-frequency network perspective. The proposed approach is outlined in Figure 3.1.

3.2 Multi-frequency EEG Networks

3.2.1 EEG Data

The EEG data was acquired during a cognitive control-related error processing task where the subjects performed a letter version of the speeded reaction Flanker task [150]. The experimental protocol of this study was approved by the Institutional Review Board (IRB) of the Michigan State University (IRB: LEGACY13-144). The data collection was conducted by following the regulations approved by this protocol. Prior to data acquisition, all subjects signed an informed and written consent form. The EEG signals were recorded with a BioSemi ActiveTwo system using a cap with 64 Ag-AgCl electrodes placed at standard locations of the International 10-20 system. The sampling rate of the data was 512 Hz. After using standard artifact rejection algorithms [63], volume conduction was minimized using the Current Source Density (CSD) Toolbox [238].

During recording, each subject was presented with a string of five letters at each trial. Letters could be

congruent (*e.g.* SSSSS) or incongruent stimuli (*e.g.* SSTSS) and the subject was instructed to respond to the center letter with a standard mouse. The trials started with a flanking stimulus (*e.g.* SS SS) of 35ms followed by the target stimuli (*e.g.* SSSSS/SSTSS) displayed for about 100 ms. The total display time is 135 ms, followed by a 1200 to 1700 ms inter-trial break between the trials. These trials capture the Error-Related Negativity (ERN) after an error response and the Correct-Related Negativity (CRN) after a correct response. For each subject, 480 total trials (each of 1-second in duration) were recorded, where the number of error trials varied from 20 to 61 across the subjects. For a fair comparison between ERN/CRN, the same number of correct trials were selected randomly. As earlier studies suggested a rise in synchronization related to ERN for the 25-75 ms time window [179], all of the analysis in this paper was conducted for the 25-75 ms time period following the response. For each subject and each response type (error and correct), a multilayer network with four layers is constructed where layers correspond to the four EEG frequency bands: θ (4-7 Hz), α (8-12 Hz), β (13-30 Hz), γ (31-100 Hz). In this chapter, we consider data from 20 participants.

3.2.2 Construction of Multilayer EEG Networks

Intralayer Edges

For a multilayer brain network where each layer corresponds to a different frequency band, the intralayer edges correspond to functional connectivity and can be quantified using measures of correlation, coherence or phase synchrony. Prior work illustrates the superior performance of reduced interference Rihaczek (RID-Rihaczek) time-frequency distribution-based phase synchrony index, i.e. RID-TFPS, in terms of time and frequency resolution and robustness to noise [12, 11]. This complex time-frequency distribution can be utilized to calculate the phase difference $\phi_{u,v}(t,f)$, between two signals x_u and x_v as:

$$\phi_{u,v}(t,f) = \arg\left[\frac{C_u(t,f)C_v^*(t,f)}{|C_u(t,f)||C_v(t,f)|}\right], \tag{3.1}$$

where $C_u(t,f)$ and $C_v(t,f)$ are the complex time-frequency distributions of x_u and x_v, respectively. Phase Locking Value (PLV) quantifies the consistency of the phase differences across trials and is computed as follows [120]:

$$\text{PLV}_{u,v}(t,f) = \frac{1}{K}\left|\sum_{k=1}^{K} e^{j\phi_{u,v}^k(t,f)}\right|, \tag{3.2}$$

where K is the total number of trials and $\phi_{u,v}^k(t,f)$ is the phase difference between x_u^k and x_v^k for trial k. After the pairwise PLV values are computed, the average pairwise synchrony within a predefined time window of interest, $W = [t_1, t_2]$, and a chosen frequency band is used as intralayer edge weights, *i.e.*

$w_{uv}^{\hbar\hbar} = \frac{1}{|W|}\frac{1}{|\hbar|}\sum_{t\in W}\sum_{f\in\hbar}\mathrm{PLV}_{u,v}(t,f)$, $1 \le u, v \le N$, where N is the number of brain regions, $|W|$ is the length of the time interval and $|\hbar|$ is the bandwidth of the particular frequency band $\hbar$.

Interlayer Edges

In a multilayer network, where the different layers correspond to different frequencies, the interlayer edges can be quantified through measures of cross-frequency coupling. In particular, phase amplitude coupling (PAC) which computes the modulation of the amplitude/power of a high frequency rhythm by the phase of a slower frequency rhythm is a commonly used metric [31, 242]. Prior work introduces a RID-Rihaczek time-frequency-based PAC measure and illustrated its superior performance with respect to Hilbert transform and wavelet-based methods [157, 158]. To quantify PAC, we first extract the instantaneous amplitude envelope of the high frequency component at node u, $a_{f_a}^u(t)$, and the instantaneous low frequency phase component at node v, $\phi_{f_p}^v(t)$, using RID-Rihaczek distribution, where f_p and f_a are frequencies within the $\hbar$th and κth frequency bands, respectively. $a_{f_a}^u(t)$ is obtained from the frequency constrained time marginal of $C_u(t,f)$ as:

$$a_{f_a}^u(t) = \int_{f_{a_1}}^{f_{a_2}} C_u(t,f)df, \tag{3.3}$$

where f_{a_1} and f_{a_2} is the bandwidth around the chosen high frequency. Similarly, the low frequency phase at node v is obtained from $C_v(t,f)$, as:

$$\phi_{f_p}^v(t) = \arg\left[\frac{C_v(t,f_p)}{|C_v(t,f_p)|}\right]. \tag{3.4}$$

Once the amplitude and phase components are extracted, PAC is estimated by distributing $a_{f_a}^u(t)$ and $\phi_{f_p}^v(t)$ to a composite vector in the complex plane at each time point and measuring the amplitude normalized modulation index (MI) [180]:

$$\mathrm{MI}_{u,v}(f_p,f_a,t) = \frac{1}{\sqrt{K}}\frac{\left|\sum_{k=1}^K a_{f_a}^{u,k}(t)e^{j\phi_{f_p}^{v,k}(t)}\right|}{\sqrt{\sum_{k=1}^K a_{f_a}^{u,k}(t)^2}}. \tag{3.5}$$

The weights of the interlayer edges between node u and v are computed as:

$$w_{uv}^{\hbar\kappa} = \frac{1}{|W|}\frac{1}{|\hbar||\kappa|}\sum_{t\in W}\sum_{f_p\in\hbar}\sum_{f_a\in\kappa}\mathrm{MI}_{u,v}(f_p,f_a,t). \tag{3.6}$$

3.3 Multilayer Modularity

As mentioned in Section 1.2, modularity function quantifies the quality of a partition by comparing the intra-community edge density to that expected under a null model and is calculated as follows [171]:

$$Q = \sum_{i=1}^{N} \sum_{j=1}^{N} (A_{ij} - \gamma P_{ij}) \delta_{g_i g_j}, \tag{3.7}$$

where P_{ij} is the expected edge weight between nodes i and j under the null model, g_i is the community of node i, and $\delta_{g_i g_j} = 1$ if $g_i = g_j$ and 0, otherwise. γ is the resolution parameter [199] to overcome the resolution limit of modularity [80]. By tuning γ, one can change the resolution of the modularity function such that larger γ values can detect smaller communities. The selection of P_{ij} depends on the null model which is a random graph with some properties, *e.g.* edge density, of the observed network preserved. Different null models can be used to define P_{ij} depending on the graph under study. For example in the configuration null model, the degree of each node is the same as that of the observed network so that the identified community structure is not affected by the heterogeneity of the degree distribution. This assumption is based on the fact that nodes with a high degree tend to connect with each other merely because they have high number of connections and not necessarily because they are within the same community [113]. To prevent this tendency to bias community detection, the null model preserves the node degrees. On the other hand, Erdős–Rényinull model does not make such an assumption and allows the identified community structure to be influenced by the degree distribution.

Based on this insight on the role of null models, we extend the definition of modularity function to multilayer networks by considering which properties of the observed multilayer network we want to preserve in the null model. In neuronal networks such as the multi-frequency brain networks, the edge weights are expected to be heterogeneous across layers [36, 266]. This is due to the fact that after a given task, usually oscillations across only a subset of frequencies are activated. Thus, the edge weights across layers cannot be homogeneous. It is important to take this heterogeneity into account to prevent trivial partitions based on the layer label rather than the true community membership. Therefore, the null model used in the definition of the modularity function should preserve the heterogeneity of edge weights across layers. We define *multilayer configuration null model*, which preserves layer-wise node degrees while randomizing the remaining characteristics of the observed multilayer graph. The expected edge weight between u^h and v^k

based on multilayer configuration null model is then defined as:

$$P_{uv}^{\hbar\kappa} = \frac{s_{u^\hbar}^{\kappa} s_{v^\kappa}^{\hbar}}{(1+\delta_{\hbar\kappa})m^{\hbar\kappa}}, \tag{3.8}$$

where $m^{\hbar\hbar}$ is the total weight of the intralayer edges in layer $\hbar$, $m^{\hbar\kappa}$ is the total weight of the interlayer edges between layers $\hbar$ and κ, and $\delta_{\hbar\kappa} = 1$ if $\hbar = \kappa$ and 0, otherwise. The multilayer modularity is then defined as follows:

$$Q = \sum_{\hbar=1}^{L}\sum_{i=1}^{n^\hbar}\sum_{j=1}^{n^\hbar}(A_{ij}^{\hbar\hbar} - \gamma P_{ij}^{\hbar\hbar})\delta_{g_i^\hbar g_j^\hbar} + \omega\sum_{\hbar=1}^{L}\sum_{\kappa=1}^{L}\sum_{i=1}^{n^\hbar}\sum_{j=1}^{n^\kappa}(A_{ij}^{\hbar\kappa} - \gamma P_{ij}^{\hbar\kappa})\delta_{g_i^\hbar g_j^\kappa}, \tag{3.9}$$

where γ is the resolution parameter and ω is the scaling parameter that weighs the importance of interlayer connections. (3.9) can be optimized with greedy algorithms, such as the Louvain algorithm [26], developed for maximizing the single-layer modularity function defined in (1.6). In this chapter, we use the Leiden algorithm, which is an extension of the Louvain algorithm with better performance [246].

3.3.1 Resolution Parameter and Inter-layer Scale Selection

We propose a statistical testing approach comparing the modularity value of the observed multilayer network to that of surrogate networks to determine the resolution and interlayer scale parameters in (3.9). Since the multilayer EEG networks are fully connected and weighted, we focus on randomization techniques presented in [8] and extend it for generating multilayer surrogate networks. In particular, we select two edges $e_{uv}^{\hbar\kappa}$ and $e_{st}^{\ell m}$ and swap their edge weights. Edges are selected such that $\hbar = \ell$ and $\kappa = m$, which ensures that the heterogeneity of edge weights across layers is preserved in the surrogate network.

Assume that we are given an observed multilayer network $\mathcal{M}$ and c surrogate multilayer networks generated from $\mathcal{M}$ as described above. We perform community detection on surrogate multilayer networks for a given pair of (γ, ω) values. We then calculate the modularity values of the detected community structures and compute the average modularity, Q_{surr}. Next, we perform modularity maximization for $\mathcal{M}$ c times and compute the average of the modularity values for the c community structures, Q_{obs}. This process is repeated for different pairs of $(\gamma, \omega) \in \Gamma \times \Omega$ where Γ and Ω are given sets of resolution parameters and interlayer scales, from which the optimal parameter values are searched. The pair with the largest difference, $Q_{obs} - Q_{surr}$, is selected as the optimal parameter values.

3.3.2 Group Community Detection

Once the community structures of the multilayer networks for a group of subjects are detected, it is often desirable to find a group community structure, which summarizes the shared communities across subjects

Algorithm 3.1 Multilayer community detection for a set of subjects' multilayer networks

Input: $\mathbb{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_S\}$: Multilayer networks of S subjects, Γ and Ω: Search sets of resolution parameter and inter-layer scale, c: Number of times to run multilayer modularity maximization.

Output: P_{group}: Group community structure

$\mathbb{C} \leftarrow \{\}$
for $s \in \{1, \ldots, S\}$ **do**
  $Q_{max} = -\infty$ ▷ To store maximum $Q_{obs} - Q_{surr}$
  for $(\gamma, \omega) \in \Gamma \times \Omega$ **do**
    $Q_{obs} \leftarrow 0, Q_{surr} \leftarrow 0, \mathbb{P} \leftarrow \{\}$
    for $i \in \{1, \ldots, c\}$ **do**
      P is found community structure and Q is its modularity value ▷
      $P, Q \leftarrow$ MLModularityMaximization$(\mathcal{M}_s, \gamma, \omega)$
      $Q_{obs} \leftarrow Q_{obs} + Q/c,$
      $\mathbb{P} \leftarrow \mathbb{P} \cup \{P\}$
      $\mathcal{N} \leftarrow$ GenerateSurrogateMLNetwork$(\mathcal{M}_s)$
      $P, Q \leftarrow$ MLModularityMaximization$(\mathcal{N}, \gamma, \omega)$
      $Q_{surr} \leftarrow Q_{surr} + Q/c$
    end for
    if $Q_{obs} - Q_{surr} > Q_{max}$ **then**
      $Q_{max} \leftarrow Q_{obs} - Q_{surr}, \mathbb{P}_{opt} \leftarrow \mathbb{P}$
    end if
  end for
  $\mathbf{C} \leftarrow$ ConstructCoClusteringMatrix$(\mathbb{P}_{opt})$
  $\mathbb{C} \leftarrow \mathbb{C} \cup \{\mathbf{C}\}$
end for
$P_{group} \leftarrow$ SC-ML($\mathbb{C}$, K) ▷ K is the average of the number of communities detected for each subject

[24, 70, 121]. In this paper, we propose a group community structure detection method based on multiplex graphs. Given L subjects, for each subject we maximize the modularity function with the optimal γ and ω values c times to obtain c community structures. Since modularity maximization is an NP-hard problem [32], modularity maximization algorithms yield locally optimal results. By running the algorithm multiple times, one can obtain a collection of informative community structures for each subject. From these community structures, for each subject we construct a co-clustering matrix $\mathbf{A}^{\hbar}$, $\hbar \in \{1, 2, \ldots, L\}$ where $A^{\hbar}_{uv}$ is the number of times nodes u and v are in the same community for subject $\hbar$ across all runs. The resulting L co-clustering matrices can be modeled as the layers of a multiplex graph, where each layer is an undirected, weighted graph corresponding to a subject. The group community structure is then found using Spectral Clustering on Multi-Layer graphs (SC-ML) [67], which finds a common community structure shared by the layers of a multiplex graph. SC-ML applies spectral clustering to a modified Laplacian defined as:

$$\mathbf{L}_{mod} = \sum_{\hbar=1}^{L} \mathbf{L}^{\hbar} - \alpha \sum_{\hbar=1}^{L} \mathbf{U}^{\hbar}\mathbf{U}^{\hbar\top}, \tag{3.10}$$

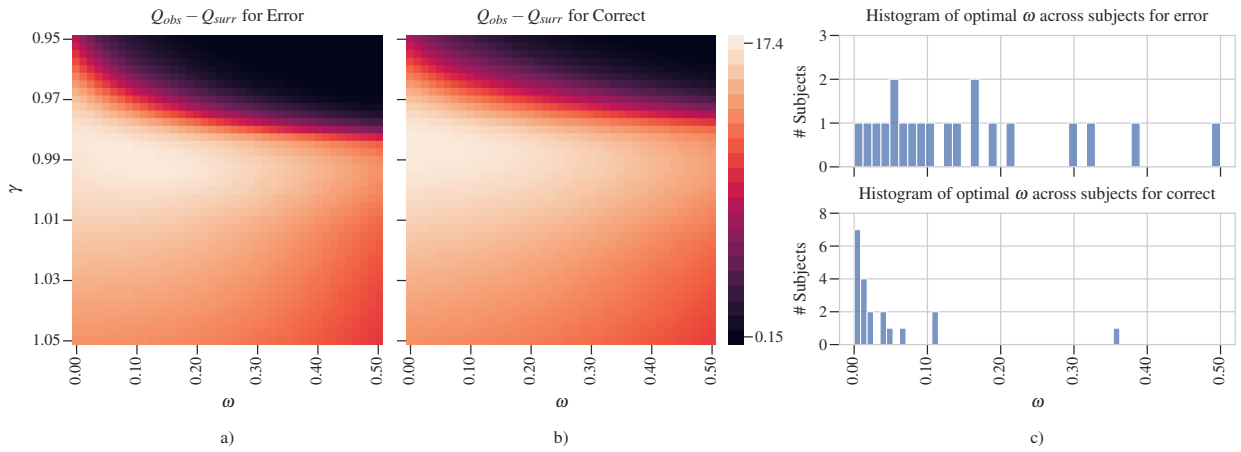

Figure 3.2: Selection of the resolution (γ) and inter-layer scale (ω) parameters: a) and b) show the average of $Q_{obs} - Q_{surr}$ across 20 subjects for error and correct responses, respectively. c) shows the histogram of optimal ω values for error (top) and correct (bottom) responses across subjects.

where $\mathbf{L}^{\hbar}$ is the normalized graph Laplacian for layer $\hbar$ defined as $\mathbf{L}^{\hbar} = (\mathbf{D}^{\hbar})^{-1/2}(\mathbf{D}^{\hbar} - \mathbf{A}^{\hbar})(\mathbf{D}^{\hbar})^{-1/2}$, $\mathbf{D}^{\hbar}$ is the diagonal matrix of node degrees and $\mathbf{U}^{\hbar}$ is the low-rank subspace embedding of layer $\hbar$. In this work, we set $\alpha = 0.5$, following the guidelines in [67]. Algorithm 1 gives the complete procedure to obtain group community structure from a given set of multilayer networks.

3.4 Results

3.4.1 Optimal resolution and scale parameters

Using the statistical testing approach described in Section 3.3.1, we first study the optimal values of γ and ω. For each subject and each response type, 100 surrogate networks are generated and their community structures are found for each $(\gamma, \omega) \in \Gamma \times \Omega$, where $\Gamma = \{\gamma : \gamma = 0.95 + 0.0025n, n \in \{0, 1, \ldots, 40\}\}$ and $\Omega = \{\omega : \omega = 0.0 + 0.0125n, n \in \{0, 1, \ldots, 40\}\}$. For each subject, 100 community structures are detected for each $(\gamma, \omega) \in \Gamma \times \Omega$. Modularity values of these community structures are evaluated and the optimal γ and ω values for each subject are then found from $Q_{obs} - Q_{surr}$.

Figures 3.2a. and 3.2b. show the average of $Q_{obs} - Q_{surr}$ across subjects for error and correct responses, respectively. For both response types, optimal γ is found to be close to 0.99, while optimal ω values are observed to be more diverse across subjects, ranging between 0.0-0.2 for error and between 0.0-0.1 for correct. In Figure 3.2c, we plotted the histograms of the optimal ω values across subjects for both response types. This figure shows that the optimal ω values are non-zero for all subjects except one for the error response. On the other hand, for correct response, the optimal ω values for 7 subjects is 0, while most of the

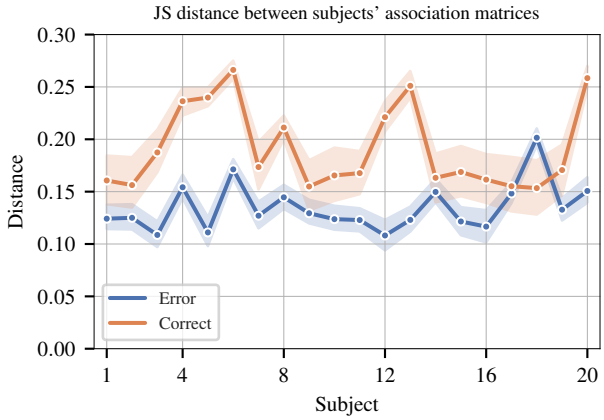

Figure 3.3: Consistency of the community structure for error and correct responses as measured by JS distance. Average JS distance of each subject with respect to other subjects is shown. Shaded area is the 95% confidence interval.

remaining subjects have optimal ω values close to 0.

3.4.2 Consistency of Community Structures for Error and Correct

After obtaining the optimal community structure for each subject and both response types, the consistency of community structures across subjects within each response type is assessed. A multiplex graph is constructed where layer h corresponds to hth subject's co-clustering matrix as described in Materials and Methods. The distance between any two layers is used to quantify the consistency of the community structures for those two subjects. Jensen-Shannon (JS) distance for graphs [60], which is always in $[0, 1]$ and is shown to be effective in assessing similarity of graphs based on their community structure [60], is used as the distance measure. Figure 3.3 shows the average JS distance between each subject and the others for each response type. This plot shows that the average distance for each subject with respect to the other subjects is lower for error response compared to the correct response.

3.4.3 Group Community Structure for Error and Correct Responses

Once the optimal community structures are obtained for each subject and for each response type, the group community structure is detected using SC-ML. The number of communities is determined as the average of the number of communities detected for each subject. These values are 5 and 9 for error and correct responses, respectively. Figure 3.4 illustrates the group community structure for error and correct responses for the multi-frequency networks. For error response, the group community structure consists of communities that include nodes from multiple layers. Community structure of θ, α and β are found to be very similar to each other. On the other hand, the community structure for the γ is different and has one

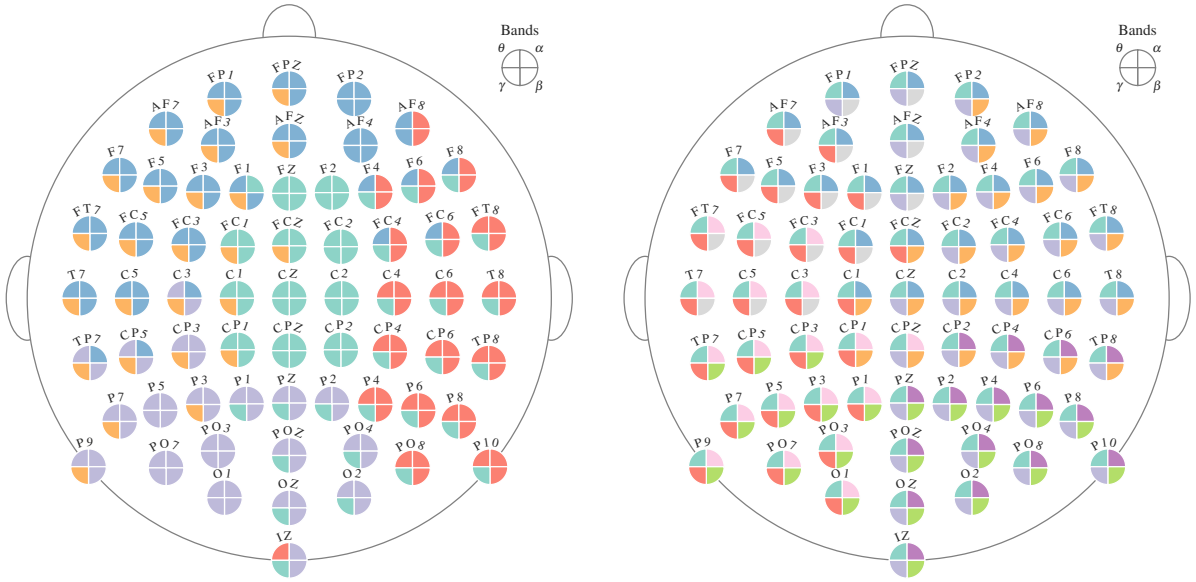

Figure 3.4: Multilayer group community structures for error (a)) and correct (b)) responses. Each electrode is shown with a circle with 4 quadrants, corresponding to the 4 frequency bands. Different colors represent different communities. Correspondence of the quadrants to the frequency bands are shown at the upper right corners of a) and b).

within layer community, while the rest are across layers. For correct response, all communities are within a single layer. Nodes in the θ band are all assigned to a single community, while the other bands have distinct community structures.

In order to better interpret the multilayer community structure, community structure for θ band is detected using single-layer modularity (see (3.7)). In particular, for each subject the community structure for the θ band is detected using single-layer modularity for each $\gamma \in \Gamma = \{\gamma : \gamma = 0.95 + 0.0025n, n \in \{0, 1, \ldots, 40\}\}$. The optimal resolution parameter is selected using the surrogate network approach. Using this optimal resolution parameter, group community structure for θ band for a given response type is found using SC-ML. The number of communities is determined as the average number of communities detected for each subject's θ band. Figure 3.5 shows the group community structure for θ band for error response. We do not consider the community structure for the correct response in the θ band, since all of its nodes were assigned to a single community with the proposed multilayer modularity as shown in Figure 3.4a. Comparing Figure 3.5 with Figure 3.4a, it can be seen that there are similarities between the community structures detected by single-layer and multilayer modularity maximization. For instance, the green community in Figure 3.5 is also detected in Figure 3.4a. Similarly, most of the nodes in purple and red communities in Figure 3.5 are in

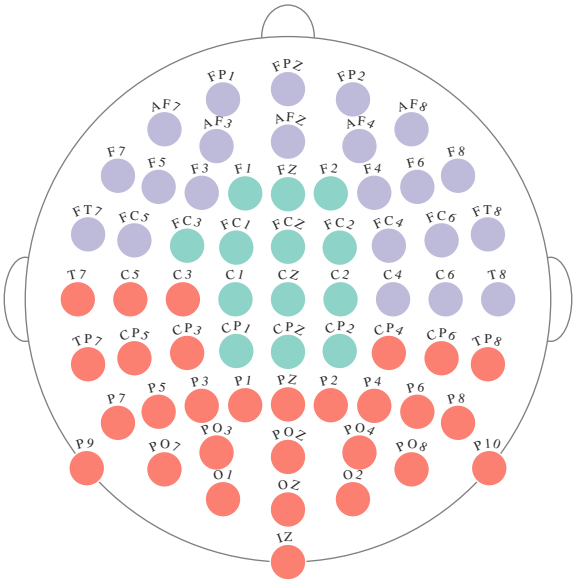

Figure 3.5: Community structure of θ band functional connectivity network found by maximizing the single-layer modularity function (see (3.7)) for error response. Each electrode is shown with a circle where the different colors correspond to different communities.

the same communities in the structure detected by the proposed multilayer modularity.

3.5 Discussion

The study of the community structure in multilayer functional connectivity networks reveal some interesting differences between error and correct responses at both the individual and group level. First, we observe the different role that inter-layer coupling plays in community formation for error vs. correct response. At the individual subject level, Figure 3.2 illustrates that while inter-layer connections are not important for the community structure of correct response as indicated by the optimal value of the scale parameter, ω, being close to 0 for the majority of subjects, they are influential in community formation following the error response. Our prior work comparing PAC between response types supports this observation as there is significantly higher cross-frequency coupling during error monitoring [157]. This increased cross-frequency coupling is between low frequency cognitive control signals which are activated after an error response and high-frequency oscillations related to motor activity and visual processing in the gamma band [94].

At the group level, the community structures in Figure 3.4, show a community comprised of the frontal-central nodes corresponding to the medial prefrontal cortex (mPFC), e.g. Fz, FCz, FCz, FC2, in the θ and α bands with parietal-occipital nodes corresponding to the visual , e.g. Pz, POz, Oz, and motor cortices, e.g. C2, C4, C6, in the γ band during ERN. mPFC is known to play an important role during ERN. In

particular, it is thought to detect conflicts and recruit additional resources from other brain areas including the lateral prefrontal cortices, visual and motor cortices to coordinate task relevant large scale networks and support adaptations of goal-directed behavior [174]. Physiologically, these interactions may occur through local and long range synchronized oscillation dynamics, particularly in the theta range (4–8 Hz). While this mPFC community structure in θ band has been observed in prior work that indicates the role of mPFC during ERN [179], the cross-frequency nature of this community is a new finding made possible by the proposed multilayer model. Our recent work shows that the phase of the θ band oscillations from the frontal-central regions modulate the amplitude of the γ band oscillations in the parietal-occipital regions following an error response supporting this finding [159]. Prior studies from others also hypothesize that error-related negativity initiates the medial frontal based top-down control mechanisms to improve the performance after an error response [98]. More recently, it has been proposed that low frequency network oscillations in prefrontal cortex, e.g. theta, guide the expression of motor-related activity in action planning and guide perception-related activity, e.g. gamma, in memory access [200]. Thus, the communities detected are consistent with previous literature reflecting higher theta-gamma coupling in the medial frontal cortex and relating this with error-related negativity. Another observation that can be made from Figure 3.4a is that the nodes corresponding to α and β bands are primarily in the same communities. This is line with recent work that indicates interlayer connectivity is dominated by one-to-one interactions for alpha-to-beta bands while for θ–γ band networks, there are additional interlayer connections between distant nodes in addition to the one-to-one connections [240]. The community structure for the correct response is mostly within-layer indicating the lack of coupling across different frequency bands.

When the group community structure for θ band in Figure 3.5 is compared to the that of Figure 3.4a, some similarities are observed. As mentioned before, the community consisting of frontal and central electrodes in Figure 3.5 is also found by the proposed multilayer community detection method. Partitioning of the remaining electrodes is also consistent across both Figures. In order to quantify the similarity of community structures of θ band shown in Figures 3.4a and 3.5, we use Normalized Mutual Information (NMI) [57]. For Figures 3.4a and 3.5, NMI is found to be 0.60, indicating an agreement between the community structures in the θ band detected by single-layer and multilayer modularity maximization methods. This consistency between the community structures across the two definitions of modularity is enabled by the way we define multilayer modularity. Our definition of multilayer modularity takes the heterogeneity of edge weights into account, thus we are able to resolve the structure within layers.

Finally, Figure 3.3 shows that there is more group level consistency in terms of topological organization for the error response compared to the correct response. This is in line with prior work [179] that shows that the organization of the functional connectivity networks for correct response is similar to pre-stimulus networks. Thus, there is more variation across subjects for the correct response compared to response-evoked networks following an error response.

3.6 Conclusions

This paper introduced a multilayer model of functional connectivity of the brain. In particular, we provided a data-driven approach to construct multi-frequency connectivity networks where layers correspond to different frequency bands. The resulting networks capture both within and cross-frequency coupling in a single framework. We then introduced a new definition of modularity for multilayer networks such that the null model preserves the heterogeneity of edge weights across layers. The community detection algorithm resulting from the maximization of this multilayer modularity function is applied to EEG data collected during error monitoring. The results indicate that following an error response, the brain organizes itself to form cross-frequency communities. This cross-frequency community formation is not observed for the correct response which indicates that the cross-frequency coupling is primarily associated with cognitive control. Moreover, we observed that the community structures detected for the error response were more consistent across subjects compared to the community structures for correct response.

Future work will consider extension of this multilayer model to higher dimensions, e.g. multi-aspect multilayer brain networks such as temporal multi-frequency connectivity networks. Compared to current work where subjects' community structure is found separately and then combined through group community detection, future work can use multi-aspect multilayer networks constructed from subjects' multilayer networks. This approach will allow simultaneous detection of communities of subjects similar to [25]. Future work will also consider different null models in the definition of modularity such as the constant Potts model, which is shown to be resolution limit free [245]. Finally, in this work we aimed to find the optimal resolution and inter-layer scale parameter; future work can focus on a multi-scale approach where the aim is to combine community structures from different resolutions and inter-layer scales [104].

CHAPTER 4

LEARNING SIGNED GRAPHS

4.1 Introduction

Gene regulatory networks (GRNs) represent fundamental molecular regulatory interactions among genes that establish and maintain all required biological functions characterizing a certain physiological state of a cell in an organism [152]. Cell type identity in an organism is determined by how active transcription factors interact with a set of cis-regulatory regions in the genome and controls the activity of genes by either activation or repression of transcription [75]. Usually, the relationship between these active transcription factors and their target genes characterize GRNs. Due to the inherent causality captured by these meaningful biological interactions in GRNs, genome-wide inference of these networks holds great promise in enhancing the understanding of normal cell physiology, and also in characterizing the molecular compositions of complex diseases [207, 147].

GRNs can be mathematically characterized as graphs where nodes represent genes and the edges quantify the regulatory relations. GRN reconstruction attempts to infer this regulatory network from high-throughput data using statistical and computational approaches. Multiple methods encompassing varying mathematical concepts have been proposed during the last decade to infer GRNs using gene expression data from bulk population sequencing technologies, which accumulate expression profile from all cells in a tissue. These methods can be broadly classified into two groups: the first group infers a static GRN, considering steady state of gene expression, while the second group uses temporal measurements to capture the expression profile of the genes in a dynamic process. A thorough evaluation of the static and dynamic models used in bulk GRN reconstruction can be found in [135, 38].

Recent advances in RNA-sequencing technologies have enabled the measurement of gene expression in single cells. This has led to the development of several computational approaches aimed at quantifying the expression of individual cells for cell-type labelling and estimation of cellular lineages. Several algorithms have been developed to arrange cells in a projected temporal order (pseudotime trajectory) based on similarities in their transcriptional states. In parallel, several dynamic models for single cell GRN reconstruction have also been developed taking into account the estimated pseudotimes. Since single cell network reconstruction algorithms try to establish functional relationships between genes taking into account the entire population of cells, it is debatable as to whether additional knowledge regarding cell state transitions may provide any

added benefits [45, 190]. In summary, direct application of bulk GRN reconstruction methods may not be adequate for single cell network inference.

The complex nature of single-cell transcriptomics data pose unique challenges in GRN inference. Changes in gene expression due to cell-cell stochastic variation, cell-cycle heterogeneity and high sparsity due to insufficient sequencing depths and capture inefficiency for genes with low expression form some of the unique characteristics of these datasets [183, 4]. Most importantly the high sparsity/high zero values feature in single cell datasets has garnered a lot of attention and several statistical methods have been designed to particularly model this phenomenon [114, 76, 201]. Recent research has indicated that these zero values referred to as "dropouts" most likely result from biological variation and may be indicative of heterogeneity in gene expression for varying cell types [236, 229].

To account for these unique challenges a variety of algorithms for network reconstruction in scRNAseq data have been recently proposed, but most of these methods fail to outperform network estimation methods developed for bulk data or microarrays [190, 45]. To that end, we propose a network reconstruction algorithm that learns the co-expression between genes using smoothness based GL algorithms. As mentioned in Section 1.3, smoothness based GL is first considered in [69] and different variations of this framework with constraints on the learned topology and for handling noisy graph signals were considered in [107, 99, 23, 106, 206]. All of the previous works learn unsigned graphs with the exception of [141], where a signed graph is learned by employing signed graph Laplacian defined by [119]. By using signed Laplacian, [141] aim to learn positive edges between nodes whose signal values are similar and negative edges between nodes whose signal values have opposite signs with similar absolute values. However, this approach is not suitable when graph signals are either all positive- or negative-valued, as in the case of gene expression data.

Considering the advantages of GL approaches in learning graph topologies that are consistent with the observed signals, in this chapter, we propose a novel GL algorithm for the reconstruction of GRNs. In particular, we assume gene expression data obtained from cells are graph signals residing on an unknown graph structure, which corresponds to the GRN. One important characteristic of GRNs is that they are signed graphs, where positive and negative edges correspond to activating and inhibitory regulations between genes. To this end, we propose a novel and computationally efficient signed GL approach, scSGL, that reconstructs the GRN under the assumption that graph signals admit low-frequency representation over activating edges, while admitting high-frequency representation over inhibitory edges. Biologically, this modelling implies that two genes that are connected with an activating edge have similar expressions, while two genes connected

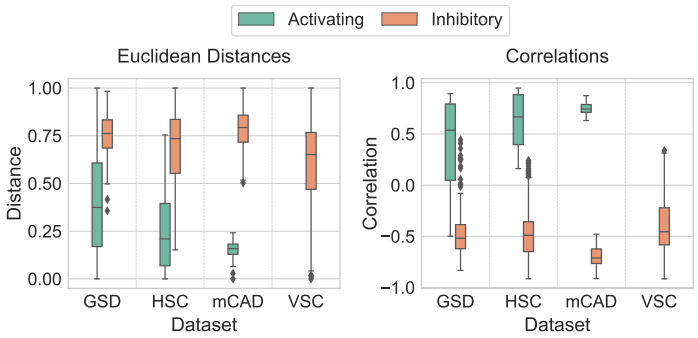

Figure 4.1: Euclidean distances (left, normalized to $[0, 1]$) and correlations (right) between expressions of gene pairs in curated datasets studied in Section 4.3. Values are calculated only for gene pairs that are connected in the ground truth GRNs and they are reported separately for activating and inhibitory edges. Only inhibitory edges are reported for VSC, since its GRN includes only inhibitory edges.

with an inhibitory edge have dissimilar expressions. In Figure 4.1, we show how these assumptions hold for curated datasets studied in Section 4.3. The figure shows that Euclidean distances between expressions are smaller for gene pairs connected by activating edges than for those connected by inhibitory edges. The figure also reports correlations between expressions, which indicates that expressions of gene pairs connected with activating and inhibitory edges are positively correlated, *i.e.* similar, and negatively correlated, *i.e.* dissimilar, respectively. We also performed a Wilcoxon Rank Sum test to determine whether the calculated associations for the positive ground truth connections were significantly lower than the associations for the negative ground truth connections for Euclidean distances. We test the null hypothesis, H_0 : the distributions of both populations are equal versus the alternative hypothesis H_a : the distribution of the negative associations are stochastically greater than the distribution of positive associations. In case of the correlation distances we want to test H_a : the distribution of the positive associations are stochastically greater than the distribution of negative associations. The calculated p-values were all less than 0.01, hence justifying our assumptions for all curated datasets except VSC, which only has negative associations. Another important characteristic of scRNAseq is high proportion of dropouts. We address this issue by employing kernel functions to map graph signals to a higher dimensional space and assuming low- and high-frequency representation for these high dimensional graph signals. This mapping allows us to use kernels that are appropriate for modelling single cell data structures.

The remainder of the chapter is organized as follows. In Section 4.2, the proposed signed graph learning approach is given. Performance of scSGL on various synthetic and real datasets are reported in Section 4.3. Finally, Section 4.4 includes discussion and concluding remarks.

4.2 Learning Signed Graphs from Graph Signals

4.2.1 Signed Graphs Revisited

In Section 1.1, a signed graph $G = (V, E)$ is defined as a network whose edges are associated with weights that can be both negative and positive edges. The edge set E can be partitioned into two sets based on the edge signs. Namely, $E = E^+ \cup E^-$, where $E^+ = \{e_{uv} | e_{uv} \in E, w_{uv} > 0\}$ and $E^- = \{e_{uv} | e_{uv} \in E, w_{uv} < 0\}$. Using this partitioning, G can be considered as a two-layer multiplex network, where layers are $G^+ = (V, E^+)$ and $G^- = (V, E^-)$. Edge weights of G^+ and G^- are determined from G: edge weights in G^+ are w_{uv}, while edges of G^- are $|w_{uv}|$. Since both layers are now unsigned graphs, we can define their adjacency matrices and combinatorial Laplacian matrices as described in Section 1.1. These matrices are indicated by $\mathbf{A}^+$, $\mathbf{A}^-$, $\mathbf{L}^+$ and $\mathbf{L}^-$. Finally, any GSP concepts developed for unsigned graphs can also be employed.

4.2.2 Signed Graph Learning

Consider a data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, whose columns are observed graph signals over an unknown signed graph G. In Section 1.3, an unsigned graph is learned with the assumption that the observed graph signals have low-frequency representation in graph spectral domain. In order to learn a signed graph G, one needs to make some additional assumptions about the graph signals $\mathbf{X}$. In this chapter, we make the following assumptions:

1. Signal values on nodes connected by positive edge values are similar to each other, *i.e.* variation over positive edges is small.

2. Signal values on nodes connected by negative edge values are dissimilar to each other, *i.e.* variation over negative edges is large.

From GSP perspective, these assumptions correspond to graph signals being low- and high-frequency over positive and negative edges, respectively. Assumption 1 implies that the graph signals have low-frequency representation in the graph Fourier domain of G^+. On the other hand, assumption 2 implies that the graph signals have high-frequency representation in graph Fourier domain of G^-. We use (1.10) to quantify how well the graph signals fit these assumptions. Thus, to learn an unknown signed graph, we minimize

$\text{tr}(\mathbf{X}^\top \mathbf{L}^+ \mathbf{X})$ with respect to $\mathbf{L}^+$ while maximizing $\text{tr}(\mathbf{X}^\top \mathbf{L}^- \mathbf{X})$ with respect to $\mathbf{L}^-$:

$$\begin{aligned} \underset{\mathbf{L}^+, \mathbf{L}^- \in \mathbb{L}}{\text{minimize}} \quad & \text{tr}(\mathbf{X}^\top \mathbf{L}^+ \mathbf{X}) - \text{tr}(\mathbf{X}^\top \mathbf{L}^- \mathbf{X}) + \alpha_1 \|\mathbf{L}^+\|_F^2 + \alpha_2 \|\mathbf{L}^-\|_F^2 \\ \text{subject to} \quad & \text{tr}(\mathbf{L}^+) = 2n, \text{tr}(\mathbf{L}^-) = 2n, (\mathbf{L}^+, \mathbf{L}^-) \in \mathbb{C}, \end{aligned} \tag{4.1}$$

where Frobenius norms and the first two constraints are similar to (1.11). $\mathbf{L}^+$ and $\mathbf{L}^-$ are constrained to be in the set $\mathbb{C} = \{(\mathbf{L}^+, \mathbf{L}^-) : L^+_{ij} = 0 \text{ if } L^-_{ij} \neq 0 \text{ and } L^-_{ij} = 0 \text{ if } L^+_{ij} \neq 0\}$ to ensure that they are not non-zero at the same indices.

4.2.3 Kernelized Signed Graph Learning

Traditional machine learning and signal processing applications are mostly developed based on linear modelling due to their simplicity. However, real world problems require nonlinear estimation that can detect more complex patterns in the data. For this purpose, kernels are introduced to capture the nonlinearity by mapping signals to a high-dimensional space [96]. Kernels correspond to dot products in a higher dimensional feature space and overcome explicit construction of the feature space; thus providing simplicity of linear methods in nonlinear estimation. Given data from input space $\mathcal{X}$, and a mapping function $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is an Hilbert space, a kernel function can be expressed as an inner product in the corresponding feature space, *i.e.* $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, where $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a finitely positive semi-definite kernel function [222]. An explicit representation of the feature map ϕ is not necessary and the dimension of mapped feature vectors could be high and even infinite.

By using different kernels, learning algorithm can be augmented to exploit various (nonlinear) associations between input data. This is especially crucial in GRN inference as shown in [230], where 17 different association measures between gene expressions are compared in terms of their performance in GRN inference and various other tasks on single-cell transcriptomic datasets. In its current form, (4.1) cannot be used directly for different kernels. Thus, the optimization problem in (4.1) is extended using kernels. The first term in (4.1) can be written as $\text{tr}(\mathbf{X}^\top \mathbf{L}^+ \mathbf{X}) = \text{tr}(\mathbf{X}\mathbf{X}^\top \mathbf{L}^+) = \sum_{i,j} \langle \mathbf{X}_{i\cdot}, \mathbf{X}_{j\cdot} \rangle L^+_{ij}$ and the second term can be written similarly. By replacing dot products with a given kernel function, *i.e.* $\kappa(\mathbf{X}_{i\cdot}, \mathbf{X}_{j\cdot})$, the problem in (4.1) can be extended to incorporate the different kernels as:

$$\begin{aligned} \underset{\mathbf{L}^+, \mathbf{L}^- \in \mathbb{L}}{\text{minimize}} \quad & \text{tr}(\mathbf{K}\mathbf{L}^+) - \text{tr}(\mathbf{K}\mathbf{L}^-) + \alpha_1 \|\mathbf{L}^+\|_F^2 + \alpha_2 \|\mathbf{L}^-\|_F^2 \\ \text{subject to} \quad & \text{tr}(\mathbf{L}^+) = 2n, \text{tr}(\mathbf{L}^-) = 2n, (\mathbf{L}^+, \mathbf{L}^-) \in \mathbb{C}, \end{aligned} \tag{4.2}$$

where $\mathbf{K} \in \mathbb{R}^{n \times n}$ is the kernel matrix with $K_{ij} = \kappa(\mathbf{X}_{i\cdot}, \mathbf{X}_{j\cdot})$. From GSP perspective, this modification implies that graph signals on each node, *i.e.* $\mathbf{X}_{i\cdot}$, are first mapped to a (higher dimensional) Hilbert space and

the signed graph is learned in this new space. Namely, let $\mathbf{\Phi} \in \mathbb{R}^{n \times \widehat{p}}$ be the matrix constructed from mapping $\mathbf{X}_{i\cdot}$'s to the Hilbert space $\mathcal{H}$ with dimension $\widehat{p}$ where rows of $\mathbf{\Phi}$ are $\phi(\mathbf{X}_{i\cdot})$. When learning unknown signed graph G with a kernel, each column of $\mathbf{\Phi}$ is a graph signal over G and they are assumed to have low- and high-frequency representation with respect to G^+ and G^-, respectively.

Extending signed graph learning problem in (4.1) using kernels brings flexibility and any association metric in [230] can be implemented in this framework if it is a positive semi-definite kernel. In this chapter, we consider three kernels: correlation coefficient, r, measure of proportionality, ρ [195] and a modification of Kendall's tau (τ_{zi}) for zero inflated non-negative continuous data [188]. These kernels are selected because ρ [195], a measure of association for compositional data and τ_{zi}, a measure of association for zero inflated non-negative continuous data [188] are shown to perform consistently better in all learning scenarios investigated in [230]. The strong performance of ρ can be explained on the basis that scRNA-seq captures only a small proportion of messenger RNA in each cell and therefore gene expression measurements can be viewed as relative measures of abundance (as seen in compositional data). On the other hand, τ_{zi}, a modification of Kendall's rank correlation coefficient, is expected to provide less biased estimates of association in the setting of zero-inflated continuous data, a characteristic of single cell transcriptomic datasets [188]. To compare and contrast these two measures, the correlation kernel r is additionally investigated since it's widely used in GRN reconstruction algorithms.

4.2.4 Optimization

The problem in (4.1) is non-convex due to the last constraint, which is called complementarity constraints [217]. In [251], it is shown that alternating direction method of multipliers (ADMM) converges for problems with complementarity constraints under some assumptions. First, we rewrite the problem in vector form. Let $\mathbf{k} = \text{upper}(\mathbf{K})$, $\mathbf{d} = \text{diag}(\mathbf{K})$, $\boldsymbol{\ell}^+ = \text{upper}(\mathbf{L}^+)$, $\boldsymbol{\ell}^- = \text{upper}(\mathbf{L}^-)$. Then, (4.2) can be rewritten as:

$$\begin{aligned} \underset{\boldsymbol{\ell}^+ \le \mathbf{0},\ \boldsymbol{\ell}^- \le \mathbf{0}}{\text{minimize}} \quad & \langle 2\mathbf{k} - \mathbf{S}^\top \mathbf{d}, \boldsymbol{\ell}^+ \rangle - \langle 2\mathbf{k} - \mathbf{S}^\top \mathbf{d}, \boldsymbol{\ell}^- \rangle + \alpha_1 \langle (2\mathbf{I} + \mathbf{S}^\top \mathbf{S})\boldsymbol{\ell}^+, \boldsymbol{\ell}^+ \rangle + \alpha_2 \langle (2\mathbf{I} + \mathbf{S}^\top \mathbf{S})\boldsymbol{\ell}^-, \boldsymbol{\ell}^- \rangle \\ \text{subject to} \quad & \mathbf{1}^\top \boldsymbol{\ell}^+ = -n, \mathbf{1}^\top \boldsymbol{\ell}^- = -n, \text{ and } \boldsymbol{\ell}^+ \perp \boldsymbol{\ell}^-, \end{aligned} \tag{4.3}$$

where $\mathbf{S}$ is defined in Section 1.1, the first two terms correspond to trace terms in (4.2), the last two terms correspond to Frobenius terms of (4.2) and first two constraints are the same as the first two constraints of (4.2). The last constraint with $\boldsymbol{\ell}^+ \le \mathbf{0}$ and $\boldsymbol{\ell}^- \le \mathbf{0}$ correspond to the complementarity constraints. By

introducing two slack variables $\mathbf{v} = \boldsymbol{\ell}^+$ and $\mathbf{w} = \boldsymbol{\ell}^-$, the problem is written in standard ADMM form:

$$\begin{aligned} \underset{\mathbf{v},\mathbf{w},\boldsymbol{\ell}^+,\boldsymbol{\ell}^-}{\text{minimize}} \quad & \iota_S(\mathbf{v},\mathbf{w}) + h(\boldsymbol{\ell}^+,\boldsymbol{\ell}^-) + \iota_H(\boldsymbol{\ell}^+) + \iota_H(\boldsymbol{\ell}^-) \\ \text{subject to} \quad & \mathbf{v} - \boldsymbol{\ell}^+ = \mathbf{0}, \mathbf{w} - \boldsymbol{\ell}^- = \mathbf{0}, \end{aligned} \tag{4.4}$$

where $\iota_S(\cdot)$ is the indicator function for the complementarity set $S = \{(\mathbf{v},\mathbf{w}) : \mathbf{v} \le 0,\ \mathbf{w} \le 0,\ \mathbf{v} \perp \mathbf{w}\}$, $h(\boldsymbol{\ell}^+,\boldsymbol{\ell}^-)$ is the objective function in (4.3), and $\iota_H()$ is the indicator function for the hyperplane $H = \{\boldsymbol{\ell} : \mathbf{1}^\top \boldsymbol{\ell} = -n\}$. The augmented Lagrangian of (4.4) is:

$$\begin{aligned} \mathcal{L}_\rho(\mathbf{v},\mathbf{w},\boldsymbol{\ell}^+,\boldsymbol{\ell}^-,\lambda_1,\lambda_2) =& \iota_S(\mathbf{v},\mathbf{w}) + h(\boldsymbol{\ell}^+,\boldsymbol{\ell}^-) + \iota_H(\boldsymbol{\ell}^+) + \iota_H(\boldsymbol{\ell}^-) \\ & + \lambda_1^\top(\mathbf{v} - \boldsymbol{\ell}^+) + \frac{\rho}{2}\|\mathbf{v} - \boldsymbol{\ell}^+\|_2^2 + \lambda_2^\top(\mathbf{w} - \boldsymbol{\ell}^-) + \frac{\rho}{2}\|\mathbf{w} - \boldsymbol{\ell}^-\|_2^2, \end{aligned} \tag{4.5}$$

where λ_1 and λ_2 are Lagrange multipliers and $\rho > 0$ is the Augmented Lagrangian parameter. Steps in kth iteration of ADMM are as follows:

$(\mathbf{v},\mathbf{w})$-step: The $(\mathbf{v},\mathbf{w})$-step of ADMM can be found as the projection onto the complementarity set S:

$$(\mathbf{c}^{k+1},\mathbf{w}^{k+1}) = \underset{\mathbf{v},\mathbf{w}}{\text{argmin}} \quad \iota_S(\mathbf{v},\mathbf{w}) + \frac{\rho}{2}\|\mathbf{v} - \boldsymbol{\ell}^{+k} + \frac{\lambda_1^k}{\rho}\|_2^2 + \frac{\rho}{2}\|\mathbf{w} - \boldsymbol{\ell}^{-k} + \frac{\lambda_2^k}{\rho}\|_2^2 = \Pi_S(\mathbf{y}), \tag{4.6}$$

where $\mathbf{y} = [(\boldsymbol{\ell}^{+k} - \lambda_1^k/\rho)^\top, (\boldsymbol{\ell}^{-k} - \lambda_2^k/\rho)^\top]^\top$ and $\Pi_S(\cdot)$ is the projection operator on the set S.

$(\boldsymbol{\ell}^+,\boldsymbol{\ell}^-)$-step : Using the fact that optimization can be performed separately for $\boldsymbol{\ell}^+$ and $\boldsymbol{\ell}^-$, $\boldsymbol{\ell}^+$-step can be written as:

$$\begin{aligned} \boldsymbol{\ell}^{+k+1} &= \underset{\boldsymbol{\ell}^+}{\text{argmin}} \quad \mathbf{z}^\top \boldsymbol{\ell}^+ + \alpha_1 \langle (2\mathbf{I} + \mathbf{S}^\top\mathbf{S})\boldsymbol{\ell}^+, \boldsymbol{\ell}^+\rangle + \iota_H(\boldsymbol{\ell}^+) + \frac{\rho}{2}\|\mathbf{v}^{k+1} - \boldsymbol{\ell}^+ + \frac{\lambda_1^k}{\rho}\|_2^2 \\ &= \Pi_H[((4\alpha_1 + \rho)\mathbf{I} + 2\alpha_1\mathbf{S}^\top\mathbf{S})^{-1}(\rho\mathbf{v}^{k+1} + \lambda_1^k - \mathbf{z})], \end{aligned} \tag{4.7}$$

where $\mathbf{z} = 2\mathbf{k} - \mathbf{S}^\top\mathbf{d}$ and $\Pi_H(\cdot)$ is the projection operator on the hyperplane H. Similarly, $\boldsymbol{\ell}^-$-step can be written as:

$$\boldsymbol{\ell}^{-k+1} = \Pi_H[((4\alpha_2 + \rho)\mathbf{I} + 2\alpha_2\mathbf{S}^\top\mathbf{S})^{-1}(\rho\mathbf{w}^{k+1} + \lambda_2^k + \mathbf{z})]. \tag{4.8}$$

Lagrange multipliers update The updates of Lagrange multipliers are:

$$\lambda_1^{k+1} = \lambda_1^k + \rho(\mathbf{v}^{k+1} - \boldsymbol{\ell}^{+k+1}), \tag{4.9}$$

$$\lambda_2^{k+1} = \lambda_2^k + \rho(\mathbf{w}^{k+1} - \boldsymbol{\ell}^{-k+1}). \tag{4.10}$$

Computational and Storage Complexity The computational complexity of the optimization procedure described above can be found by determining how many computations are required for each ADMM step. Let $M = n(n-1)/2$ where n is the number of nodes. $(\mathbf{v}, \mathbf{w})$-step can be performed in $O(M)$ time, or $O(n^2)$ time. $(\boldsymbol{\ell}^+, \boldsymbol{\ell}^-)$-step requires the inversion of the matrix $(4\alpha_1 + \rho)\mathbf{S} + 2\alpha_1\mathbf{S}^\top\mathbf{S}$, which needs to be calculated only once before the optimization iterations. The inverse matrix has a closed form solution which can be found using Woodbury matrix identity. It has a decomposition of the form $\mathbf{A}^\top\mathbf{A}$ where $\mathbf{A}$ is a sparse matrix with $O(n^2)$ non-zero entries. Thus, matrix-vector multiplication of $(\boldsymbol{\ell}^+, \boldsymbol{\ell}^-)$-step can be done in $O(n^2)$ time. Updates of Lagrangian multipliers can also be performed $O(M)$ time, or $O(n^2)$. Let I be the number of iterations required for the convergence of ADMM. Thus, overall time complexity of scSGL is $O(In^2)$. The storage complexity of scSGL is determined by the size of the inverse matrix required in $(\boldsymbol{\ell}^+, \boldsymbol{\ell}^-)$-step. Since this matrix has a decomposition of the form $\mathbf{A}^\top\mathbf{A}$, we only need to store $\mathbf{A}$. Thus, the storage complexity of scSGL is $O(n^2)$.

Based on the above analysis, computational and storage complexity of ADMM is quadratic in the number of nodes and is not affected by the number of graph signals. Note that, scSGL also requires the construction of the kernel matrix before running the optimization. Since there are already very efficient tools to construct kernel matrices [230], we did not include their complexity in the analysis above. Finally, there are recent works in GSP literature for scaling GL methods to learning graphs with millions of nodes [109]. These approaches can be employed to scale scSGL, which we left as a future pursuit.

4.2.5 Hyperparameter Selection

The optimization problem in (4.2) requires the selection of two regularization parameters α_1 and α_2, which determine the density of the learnt graph, *i.e.* large values of α_1 (α_2) result in denser $\mathbf{L}^+$ ($\mathbf{L}^-$). Their values can be set to obtain a graph with desired positive and negative edge densities. We propose a resampling approach [74] to determine desired positive and negative edge densities empirically. The approach has following steps:

1. We randomly shuffle each column of the data matrix $\mathbf{X}$ to generate a surrogate data matrix.
2. Association between rows of the surrogate data matrix are calculated by the kernel employed in (4.2).
3. Thresholds λ_1 and λ_2 are selected as the pth and $(100-p)$th percentiles of the values in the kernel matrix calculated in Step 2.

4. Steps (1-3) are repeated k times to construct the empirical distribution of the thresholds λ_1 and λ_2.

5. Finally, $\widehat{\lambda}_1$ and $\widehat{\lambda}_2$ are selected to be the medians of the empirical distributions constructed in Step 4.

6. The kernel matrix for the original data $\mathbf{X}$ is constructed.

7. The number of entries in the kernel matrix that are smaller than $\widehat{\lambda}_1$ are determined and normalized by the total number of entries in the kernel matrix to obtain the density of $\mathbf{L}^-$. Similarly, number of entries in the kernel matrix greater than $\widehat{\lambda}_2$ is used to determine the density of $\mathbf{L}^+$.

8. α_1 and α_2 are then selected to learn graphs with the estimated graph densities found in Step 7.

For all the datasets analyzed in the Section 4.3, we learned the densities of positive and negative parts by setting $p = 5$.

4.3 Results

In this section, performance of scSGL is evaluated and compared to state-of-the-art GRN inference methods on various simulated and experimental scRNAseq datasets. We selected GENIE3 [103], GRNBOOST2 [146], PIDC [41] and PPCOR [116] for comparison as they are the top performing methods in [190]. GENIE3, GRNBOOST2 and PPCOR were originally developed for bulk analysis, while PIDC is developed for single cell gene expression data. Among these methods, GENIE3 and GRNBOOST2 return fully connected directed networks, while the remaining two infer undirected networks. Finally, only PPCOR algorithm returns signed graphs.

4.3.1 Performance Metrics

AUPRC Ratio Given the inherent sparsity of gene networks, we used the area under the precision-recall curves (AUPRC) ratio as the primary evaluation metric. AUPRC are calculated by comparing inferred graphs to ground truth gene regulations. During this calculation, signs of the learned edges are ignored as AUPRC is restricted to binary classification. In particular, we first take the absolute value of edge weights and then compare them to ground truth edges. Thus, these metrics indicate how well methods detect edges without considering the signs of the inferred edges. Ground truth networks are considered as undirected and self-loops are ignored. Following [190], we defined *AUPRC ratio* as the ratio of AUPRC value of the methods to AUPRC of the random estimator.

AUPRC Ratio Activating/Inhibitory One of our goals is to learn whether the edges are activating or inhibitory. AUPRC as defined above cannot evaluate the sign information. Thus, for curated datasets, whose ground truth gene regulations include signed edge information, we calculate AUPRC for activating and inhibitory edges separately. In particular, for methods that learn signed graphs we compare the learned positive edges to activating edges in ground truth and learned negative edges to inhibitory edges in the ground truth. For methods that do not learn signed edges, we evaluate the inferred edges with respect to the ground truth activating and inhibitory edges separately to calculate two AUPRC values.

4.3.2 Synthetic Datasets

Curated Datasets From BEELINE: The first simulation datasets we consider are curated from "published Boolean models of GRNs" [190]. These datasets were generated using the recently proposed single cell GRN simulator BoolODE [190]. BoolODE converts boolean functions specifying a GRN directly to ODE equations using GeneNetWeaver [216, 134], a widely used method to simulate bulk transcriptomic data from GRNs. These datasets are generated from four literature-curated Boolean models: mammalian cortical area development (mCAD), ventral spinal cord (VSC) development, hematopoietic stem cell (HSC) differentiation and gonadal sex determination (GSD). These models represent different types of graph structures, with varying numbers of positive and negative edges; thus serving as good examples for illustrating the robustness of the proposed method in modelling signed graph topologies. BoolODE is used to create ten random simulations of the synthetic gene expression datasets with 2,000 cells for each model. For each dataset, one version with a dropout rate of 50% and another with a rate of 70% are also considered to evaluate the performance of the methods under missing values.

AUPRC ratios are calculated separately for activating and inhibitory edges and their average over realizations are reported in Figure 4.2. For most of the datasets, scSGL performs better than other benchmarking methods in inferring both activating and inhibitory edges. Although there is a difference between the performances of different kernels, scSGL generally performs better than state-of-the-art methods irrespective of the selected kernel. Comparing the performances of different kernels, it is observed that τ_{zi} results in higher AUPRC ratios in GSD, HSC and VSC while ρ performs better in mCAD datasets. It is also observed that AUPRC ratios are higher for activating edges then inhibitory edges. Increasing the dropout ratio causes a drop in the performance of all methods for inferring the activating edges but not for learning the inhibitory edges. Overall, the best performing kernel is τ_{zi}, which might be because of its robustness to increasing

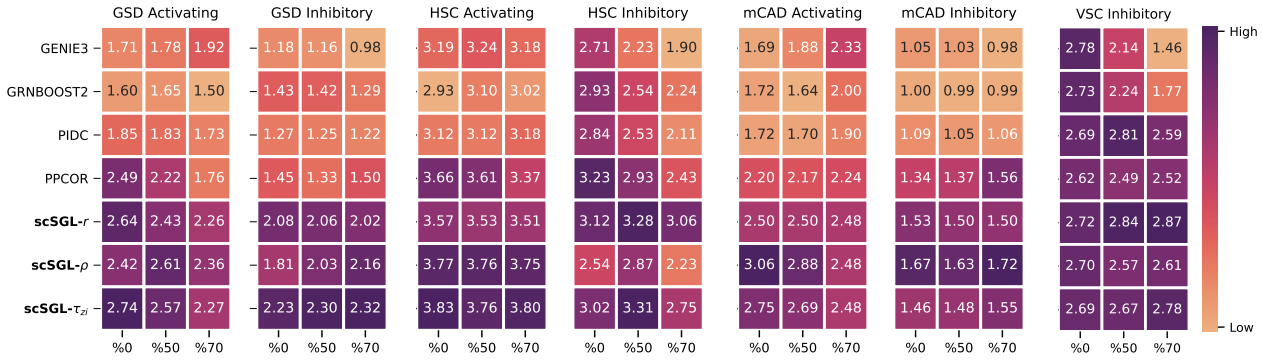

Figure 4.2: Performance of scSGL and state-of-the-art methods on curated datasets as measured by AUPRC ratio for activating and inhibitory edges. x-axis indicates dropout ratio in the dataset.

dropout ratio compared to other kernels.

Parameter Sensitivity Analysis: To mimic the zero inflated and overly dispersed nature of most scRNAseq datasets, we simulated gene expression data from a multivariate zero-inflated negative binomial (ZINB) distribution for our second simulation. These datasets were then used to conduct parameter sensitivity analysis for the proposed methods. Given a known graph structure, synthetic datasets are generated from a ZINB distribution by adapting an algorithm developed by [262]. The three parameters of the ZINB distribution; λ, k and ω, which control its mean, dispersion and degree of zero-inflation, respectively were determined from a real scRNAseq dataset to make the simulations mirror the properties of real datasets. The procedure to generate simulated gene expression data is as follows:

1. For each simulation setting, we first draw a binary graph G from a random graph model with n genes.

2. Each edge of G is assigned a weight, W_{ij} such that:

$$W_{ij} = \begin{cases} Unif(0.3, 0.7) & \text{with probability } 0.5, \\ Unif(-0.7, -0.3) & \text{otherwise.} \end{cases}$$

3. p random samples are drawn from multivariate Gaussian distribution with precision matrix $\mathbf{W}$. The random samples are used as the columns of the matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$.

4. To mimic the dropout phenomenon present in real single cell datasets, we next introduced additional zeros to the gene expression matrix $\mathbf{X}$. Following [187], the dropout probability for each row of $\mathbf{X}$ was calculated as: $\pi_{ij} = \exp(-\alpha {X_{ij}}^2)$, where α represents the exponential decay parameter that controls the dependence between the dropout probability and gene expression.

5. A binary indicator was next sampled for each entry: $\xi_{ij} \sim Bernoulli(\pi_{ij})$, with $\xi_{ij} = 1$ indicating that the corresponding entry of X_{ij} would be replaced by 0. The dropout probability for each gene vector was calculated as $\omega_i = \sum_{j=1}^{p} \xi_{ij}$.

6. Using a modification of the NORTA (Normal to Anything) method [262] we generated samples from a multivariate zero inflated negative binomial distribution based on $\mathbf{X}$ generated in Step 3 using mean λ, dispersion κ and zero-inflation parameters ω_j's.

7. To mirror real scRNA-seq gene expression data behavior, the gene expression mean λ and standard deviation κ were estimated from a real scRNA-seq dataset, Peripheral Blood Mononuclear Cells (PBMC) freely available from 10X Genomics.

The ZINB simulator is then used to generate expression data from three different graph models: random networks, networks with a given community structure and networks with hubs. Random networks are generated using Erdős–Rényi model with desired edge density. Since Erdős–Rényi model is not realistic due to its binomial degree distribution, we also consider networks with hubs. These networks are generated using a Barabási–Albert model whose degree distribution follows a power-law function. Finally, networks with community structure, also known as modular networks, are generated using a disjoint union of random graphs. To investigate the robustness of scSGL, we simulated datasets from the aforementioned network topologies by varying the following parameters: *(i)* number of genes (10, 50, 100 and 250), *(ii)* number of cells (100, 300, 500 and 1000) and *(iii)* dropout probabilities (0.26-0.36). To account for the inherent randomness of the simulations, 10 independent data replicates were generated for every parameter combination and the mean AUPRC ratios obtained by averaging over the replicates are reported in Figure 4.3.

Recent investigations of scRNAseq datasets have revealed that dropout rates are primarily driven by a combination of technical and biological factors [75]. Consequently, while mean gene expression and proportion of zeros are linked, this may vary based on cell type, sex, and other biological and technical factors. While investigating the impact of dropout rates on network estimation accuracy, we found a steady decline in AUPRC ratios for all methods with an increase in the number of zeroes. scSGL irrespective of the kernel choice maintained the highest AUPRC ratios across all network topologies. Gene expression in scRNAseq datasets can be intepreted as relative measures of abundance owing to the datasets being a combination of gene expression derived from several cell-types. This could be the reason why proportionality measures perform well [230]. The strong performance of τ_{zi} can be explained on the basis that it explicitly

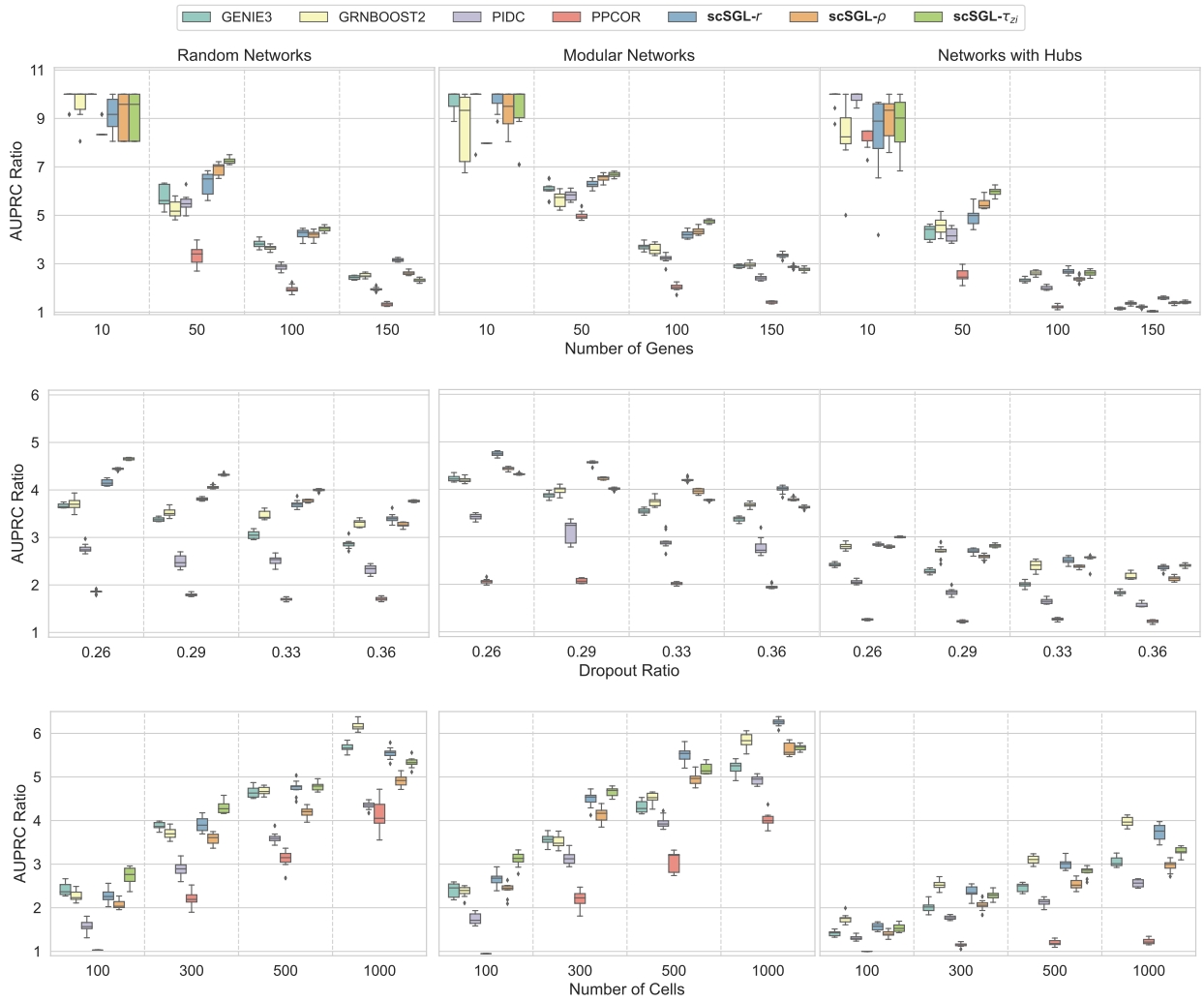

Figure 4.3: Performance of various methods for synthetic datasets with varying number of genes (top row), dropout ratio (middle row) and number of cells (bottom row).

models the dropouts present in scRNAseq datasets. Despite the poor performance of regularized correlation networks (PPCOR), we see a strong performance of scSGL when using the correlation kernel. This proves that gene-gene relationships are in fact non-linear in nature. This belief is also strengthened by the above average performance of tree-based machine learning algorithms like GENIE3 and GRNBOOST2. It is to be noted that PIDC, the only other method capable of modelling excess zeroes, while accounting for non-linear relationships fails to achieve a top-ranking AUPRC ratios.

Next, we evaluated the impact of cell sizes on network reconstruction. Figure 2 demonstrates a clear rise in AUPRC ratios when the number of cells are increased. PIDC, the only other single cell network estimation technique, achieves a below average performance at the lowest sample size of 100. This could be due to the fact that PIDC requires large sample sizes for accurate estimation of pairwise joint probability distributions

for calculating mutual information. In general, PPCOR has the worst performance among all methods. It should also be noted that the performance of GRNBOOST2 was equivalent to scSGL for all the network topologies when the sample size was 10 times the number of genes. These results indicate the importance of sample size in accurate network estimation for all of the methods and network topologies is considered.

Finally, the performance of each of the methods was evalulated by varying the number of genes. All methods had high AUPRC ratios across network topologies when the number of genes was small. While the AUPRC ratios of all the methods declined with an increase in the number of genes, scSGL performed significantly better than most of the benchmarking methods. This dip in performance could be attributed to the fact that all methods learn very dense networks. With an increase in the number of nodes, there is an increasing number of false edges detected by every algorithm. The performance of scSGL could further be improved with a more biologically informed framework for hyperparameter selection.

Computational Complexity: Methods are compared in terms of their scalability to datasets with large number of genes. For this purpose, synthetic data generation process used in parameter sensitivity analysis is employed to create three datasets with 500, 1000 and 2000 genes. Each dataset is generated from Barabási–Albert model, includes 1000 cells, and has a dropout ratio of 0.26. Average run time and AUPRC ratios over 10 replicates are reported in Figure 4.4. We reported results only for the correlation kernel, as other kernels have similar performances and run times. It is observed that scSGL runs significantly faster than GENIE3, GRNBOOST2 and PIDC while having superior performance in terms of AUPRC ratio. Although PPCOR runs faster than scSGL, it shows poor performance.

4.3.3 Real Datasets

For real datasets, we consider scRNAseq expressions of human embryonic stem cells (hESC) and mouse embryonic stem cells (hESC) which include 758 and 451 cells, respectively. We inferred GRNs between 500 highly varying genes along with highly varying TFs [190]. Inferred GRNs are compared to three different databases of gene regulations: STRING [237], cell-type specific [50] and nonspecific [130, 83, 92]. AUPRC ratios are reported in Figure 4.5. All methods have performance values close to random estimator. Except PPCOR, which has random performance in both datasets and for all databases, methods have comparable performances, with scSGL showing slightly better performance in hESC and while benchmarking methods working slightly better in mESC.

To add biological meaning to the estimated networks we compared them to the reference networks in

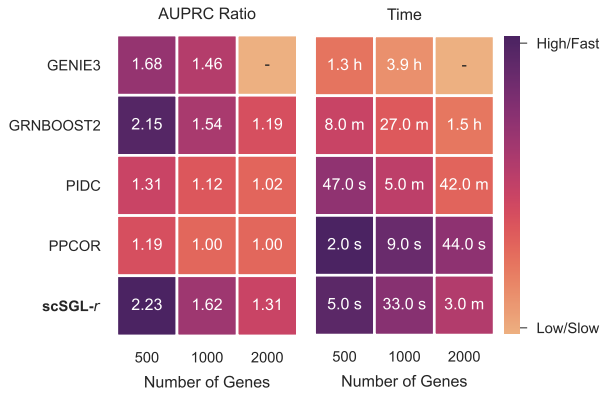

Figure 4.4: Scalability analysis of different methods. Run time of benchmarking methods are calculated using BEELINE pipeline [190]. Run time of scSGL includes kernel construction and optimization procedure. All methods are run on the same computer. Results of GENIE3 for 2000 genes are not reported due to its high run time.

the STRING database. The STRING database is a compendium of protein-protein interactions created by gathering information from varying sources like experimental studies, text mining *etc*. The edges in the STRING network are classified as high confidence (minimum score of 0.700), medium confidence (minimum score of 0.400) and and low confidence (minimum score of 0.150). In hESC dataset, scSGL-ρ identified the maximum number of high confidence associations present in the STRING reference network. scSGL-ρ, scSGL-r and scSGL-τ_{zi} each identified 60, 56 and 24 high confidence STRING interactions, respectively, with an edge confidence greater than 0.5. The interactions identified by scSGL-r form a network of 56 unique genes including genes Nanog, Sox2, Sox4, Pou5f1, Ctnnb1, Gata2, Gata3 and many others. Lineage-specific marker genes, Cdk6, Col5a1, Vim, and Itg5, which are known to have regulatory roles in cell differentiation

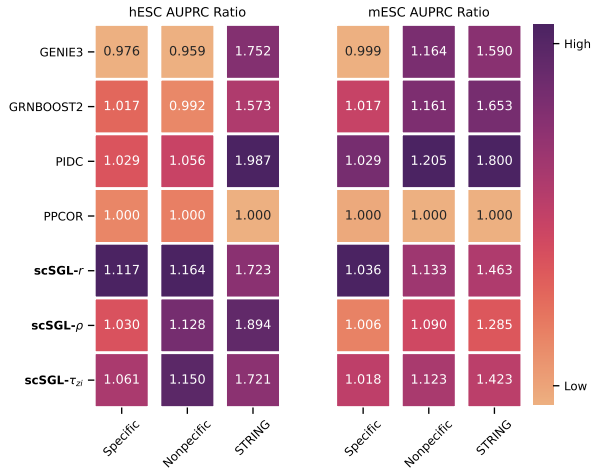

Figure 4.5: Performance of methods for two real-world scRNAseq datasets. Inferred graphs are compared to three different gene regulatory databases.

were also detected by scSGL-r but with edge confidence less than 0.5 (0.1-0.3) [30, 47]. scSGL-ρ and scSGL-r identified 20 common genes including Sox2, Sox4, Gata6, Ctnnb1 and Bmp4. scSGL-τ_{zi} identified the least number of genes but successfully retrieved lineage markers Nanog, Sox2, Sox4, Pou5f1,Ctnnb1, Gata2, Gata3. All three kernel methods identified genes Sox4, Ctnnb1, Bmp4 and Gata6. According to the STRING database, the 56 genes identified by scSGL-r are associated with 839 significantly enriched biological process gene ontology (GO) terms that include cell differentiation, chromosome separation, specification of animal organ position, mitotic nuclear division and organ formation. Genes identified by scSGL-ρ and scSGL-τ_{zi} had similar functional enrichments for biological processes. To demonstrate some of the learned associations in hESC, we plotted the subnetwork of 24 lineage specific marker genes using scSGL [47]. Figure 4.6a shows the presence of activating relationships between key definitive endoderm (DE) markers like Gata6, Gata4, and Eomes and joint inhibition of pluripotency markers Pou5f1, Nanog, and Sox2. Gata4 and Gata6 have been reported as necessary for the development and function of a number of endoderm-derived tissues and cells [253, 250] and onset of Gata4 and Gata6 expression has been reported to be coincident with the beginning of endoderm gene expression [77]. In addition, inhibition of pluripotency markers by the key DE markers indicates progression of the cells towards a DE state.

In mESC dataset, scSGL-ρ, scSGL-r and scSGL-τ_{zi} each identified 67, 103 and 55 high confidence STRING interactions, respectively, with an edge confidence greater than 0.5. The three estimated networks

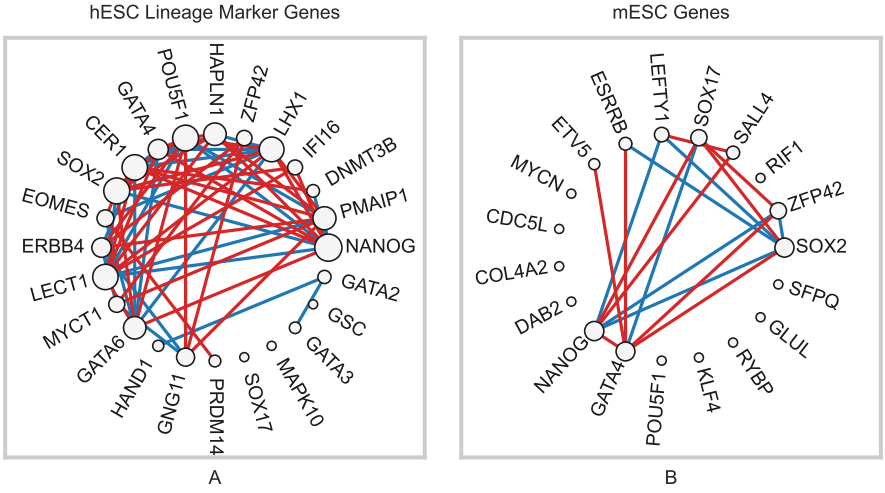

Figure 4.6: The subnetworks of 24 lineage specific genes in hESC (A) and 19 well known marker genes in mESC (B). We report results of scSGL-r as it has the highest AUPRC ratio in Figure 4.5. For clarity, only those edges whose absolute edge weight fall into the top 1 percentile are shown. Node sizes are proportional to their degrees.

capture interactions regulated by known transcription factors Sox2, Nanog, Klf4, Myc and Sall4 [270]. scSGL-r identified known relationships between Sox2 and Nanog; Esrrb with Sox2 and Rybp among many others. scSGL-ρ identified known relationships between Esrrb and Etv5 and indirect interactions between Sall4 and Rybp regulated by TF Oct4. scSGL-τ_{zi} identified most of the important relationships identified by scSGL-r along with additional relationships between Sox2, Nanog, and Rif1. According to the STRING database, the 103 genes identified by scSGL-r are associated with 908 significantly enriched biological process GO terms that include cell fate determination, specification and commitment, mitotic DNA replication and regulation of nodal signalling pathway. Similar to hESC analysis, scSGL, irrespective of the chosen kernel, identified genes with similar functional enrichments for biological processes. To demonstrate some of the learned associations in mESC, we plotted the subnetwork of 19 well known marker genes+TF in mESC differentiation, estimated using scSGL. As can be seen in Figure 4.6b, Nanog, Gata4, Sox2, Sox17, Zfp42 and Lefty1 emerge as some of the hub nodes with high degrees of associations. The learned network also captures vital signed associations between Sox2, Nanog, Sox17, Zfp42 and Gata4. It is well known that Sox2 and Nanog form the core of a transcription factor network that promotes embryonic stem cell pluripotency and self- renewal. Zfp42 is also known to be a direct target of Nanog, which is augmented by Sox2 [226]. In addition, Sox17 together with Gata4 expression reinforce a transcriptional network that antagonizes Nanog expression to initiate differentiation [173].

Finally, to analyze the relation between edges identified by scSGL and benchmarking methods, the intersection between the top 1000 edges is reported as an UpSet plot [123] in Figure 4.7. In both datasets, PPCOR does not have any intersection with other methods probably because of its poor performance reported in Figure 4.5. The remaining 6 methods have an intersection set with cardinality around 40 edges. The same number of common edges is found in the intersection of PIDC, GENIE3, GRNBOOST2, scSGL-τ_{zi}, scSGL-r and in the intersection of PIDC, GENIE3, scSGL-τ_{zi}, scSGL-r, scSGL-ρ. These observations hold for both datasets, indicating the reproducibility of the proposed approach across different datasets. Edges identified by τ_{zi} and r have more intersecting edges with benchmarking methods and with each other than those identified by ρ, which indicates that the benchmarking methods have more common edges with correlation based association metrics than with proportionality measures. scSGL methods have more common edges with PIDC than with GENIE3 and GRNBOOST2, which may be due to the fact that PIDC learns co-expression GRN similar to scSGL, while GENIE3 and GRNBOOST2 learn directed interactions between genes.

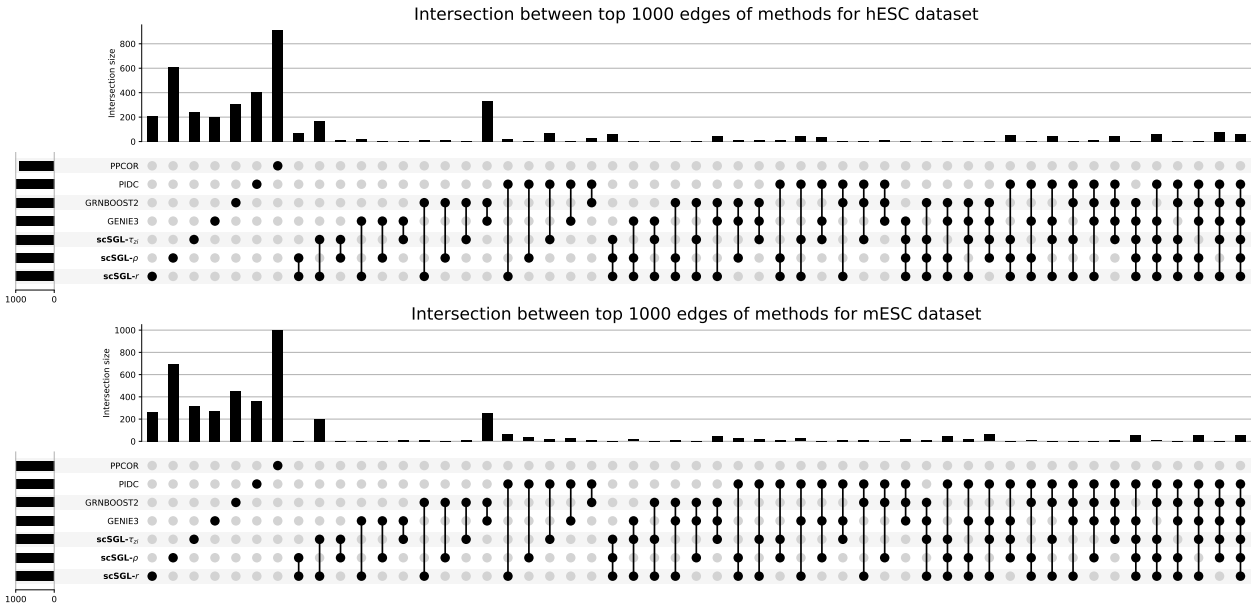

Figure 4.7: UpSet plot that shows intersection between the top 1000 edges by scSGL with 3 kernels and benchmarking methods in hESC and mESC datasets.

4.4 Conclusions

In this chapter, we have introduced a novel network inference algorithm based on GSP. Our proposed algorithm scSGL identifies functional relationships between genes by learning the signed adjacency matrix from the gene expression data under the assumption that graph signals are similar over positive edges and dissimilar over negative edges. This novel technique also takes into account the nonlinearity of the gene interactions by employing kernel mappings. We applied scSGL to four curated datasets derived from "published Boolean models of GRN" and two real experimental scRNAseq datasets during differentiation. To conduct an in-depth analysis of gene co-expression network reconstruction from scRNAseq datasets, we generated simulations from zero inflated negative binomial distributions. These simulations, generated using different parameter combinations, were used to investigate the robustness of our proposed method to changing cell sizes, gene numbers and dropout rates.

For the curated datasets, scSGL consistently obtained higher AUPRC ratios in comparison to the benchmarking methods, despite each dataset having a different number of stable cell states. Parameter sensitivity analysis reflected the superior performance of scSGL in estimating networks under varying network topologies. The performance remained consistent even when the gene numbers increased, the dropout rates were high and the sample sizes were low. This indicated the robustness of scSGL in modelling networks under varying characteristics of scRNA-seq datasets.

The networks estimated from real data using scSGL identified important functional relationships between target genes and transcription factors and exhibited enrichment for appropriate functional processes. We also demonstrated that scSGL attained performance comparable to state-of-the-art-methods in real data experiments, with the performance of all the GRN reconstruction methods methods being close to random. Accuracy evaluation of the predicted networks for the real datasets were done using cell-type specific, non-specific and functional networks described in [190]. However, most of the information in these ground truth datasets have been accumulated based on tissue level data and hence it's not completely appropriate to calculate precision and recall rates from these databases.

Although scRNAseq techniques provide significant advantages over bulk data such as increased sample size with higher depth coverage and and presence of highly distinct cell clusters, it also comes laced with multiple sources of technical and biological noise. Moreover, the inability to differentiate between technical and biological noise, and the absence of adequate noise modelling techniques further exacerbate the problem [89, 233]. scSGL aims to capture the node similarities and dissimilarities based on distances between graph signals. These graph signals exhibit smoothness, which implies that within a given node cluster, genes tend to be homogeneous, while varying across clusters. This leads to densely connected graphs where the heterogeneity induced by distinct cell sub-populations can be simultaneously curbed. Using single cell data with cell cluster labels, easily obtained from single cell clustering algorithms [186], in conjunction with scSGL can aid in identifying functional modules that are associated with a cell type [252]. Integrating pseudotemporal ordering with scSGL can further help in identifying the functional modules associated with differential pathways [261].

Despite the availability of a large number of computational methods, accurate GRN reconstruction still remains an open problem. Most reconstruction methods are based on the assumption that presence of an edge implies regulatory relationships. They also have the tendency to establish links between genes regulated by the same regulator. These issues can generate a lot of false positives and therefore additional sources of data such as ChIP-seq measurements that help in identifying direct interactions between TFs and target genes, can provide a way to filter out the spurious interactions [1]. Finally, gene regulation has multiple layers beyond direct TF-target interaction, but functional relationships can only be established if these relationships induce persistent changes in transcriptional state. As single cell data sources over multiple modalities continue to become available, it will be interesting to see how integration of these data types aids GRN reconstruction using scSGL [235].

CHAPTER 5

LEARNING MULTIVIEW SIGNED GRAPHS

5.1 Introduction

As mentioned in previous chapter, gene expression arises from a network of regulatory interactions between transcription factors, co-factors and signaling molecules [211, 265]. Elucidating the topology of this underlying regulatory network is essential for understanding the mechanisms that govern complex biological processes in human physiology and pathology. A major focus area in clinical research lies in studying the changes in gene coexpression networks across different tissues, cell types/states, and conditions. For example, in the extensively studied breast cancer datasets from the cancer genome atlas, there are four main subtypes of breast cancer [165]. The variation between these subtypes holds the key to inferring how genes transcriptionally regulate each other and how their expressions and interactions change across subgroups. In addition one would expect the gene relationships corresponding to different subtypes to be similar to each other since they originate in the same tissue, but also posses crucial differences since they are in different stages of disease progression [55, 122, 91]. Thus, instead of estimating a single network for all the subtypes, constructing class-specific graphical models for different conditions will provide a more robust and deeper understanding of group-specific characteristics.

Recent advances in RNA sequencing have made it possible to profile the gene expression of individual cells. Dozens of algorithms have been proposed for the reconstruction of gene regulatory networks from scRNA-seq datasets [45, 190]. Most of these algorithms, however estimate a single gene regulatory network, assuming the data samples to be identically and independently distributed; hence ignoring the presence of natural subgroups that may be present within the data. Given the assumption of a grouped dataset, one should be able to apply these algorithms to estimate networks from each subgroup separately; but this procedure of independent group-wise network estimation will fail to model the shared structures between the subgroups, eventually leading to information loss. Therefore, there is a pressing need to develop joint graph estimation models that would allow information borrowing across subgroups while retaining subgroup specific heterogenity.

Multiple algorithms have been proposed for joint estimation of networks from high dimensional data. Most of these methods assume that the data has a Gaussian distribution. Seminal paper [90] paved the way for penalized estimation of multiple Gaussian graphical models, and demonstrated the use of lasso based

penalty functions for better estimation across multiple groups. Later, [55] proposed the fused graphical lasso and the group graphical lasso penalties for better estimation. These methods however are not directly applicable to single cell datasets. Despite many advantages, scRNA-seq datasets are undermined by a series of technical limitations, such as dropouts and a high level of noise, which renders void the assumption of Gaussianity [75, 44, 4]. Few methods have been proposed for joint estimation of multiple networks from scRNA-seq datasets. [154] developed PIPER, a penalized local Poisson graphical model [7] for joint estimation of multiple networks in scRNA-seq datasets. One of the main limitations of PIPER is that the Poisson distribution has one single parameter characterizing both the mean and the standard deviation. Single cell datasets would be better characterized by a negative binomial distribution which has a separate dispersion parameter or a zero inflated negative binomial distribution which could account for the excessive zeroes. To account for the non-Gaussian nature of the scRNA-seq datasets, [255] proposed a modification of the joint Gaussian copula graphical model based on the Gaussian copula transformation proposed in [128]. To facilitate estimation of Kendall's τ correlation matrix in the presence of dropouts they propose a modified Kendall's τ metric that only utilizes the completely observed values, and excludes the missing values. [66] proposed a three step hybrid joint estimation strategy that relies on (a) integrated application of a Bayesian zero inflated Poisson based model imputation strategy and single cell imputation technique McImpute [105, 148], (b) data Gaussianization [127] and eventually (c) joint estimation of a Gaussian graphical model [55]. Contrary to [154], the last two proposed approaches estimate graphical models for continuous data and rely on a data transformation step for making the data continuous.

In this chapter, we focus on GSP based GL for the joint inference of multiple GRNs, where gene expressions from cells are considered as graph signals on the unknown GRNs. Since GSP based GL methods employ explicit representation of graph signals in the graph frequency domain, they have more flexibility in modeling signals compared to previous network inference methods, such as statistical models reviewed above for GRN inference. However, existing GSP based GL algorithms [69, 107, 219, 163] have two important shortcomings for multiple GRN learning. First, they cannot learn signed graphs, which is a more suitable model for GRNs as they include activating and inhibitory edges. Second, with the exception of [163], they can only learn a single graph. Thus, they are not applicable to the joint inference of multiple GRNs problem.

This chapter presents a multiple signed graph learning algorithm (scMSGL) for joint inference of GRNs from multiple classes (conditions/disease states). Based on the method developed in Chapter 4, scMSGL

learns multiple GRNs by deriving an optimization problem using three assumptions: (i) expressions of genes connected with activating edges are similar to each other, (ii) expressions of genes connected with inhibitory edges are dissimilar to each other, and (iii) GRNs corresponding to the different datasets are related to each other. Thus, scMSGL optimizes the total variation of graph signals to learn signed graphs while ensuring that the learned signed graphs are similar to each other through regularization with respect to a learned signed consensus graph. The proposed method has several advantages over existing approaches. First, it performs joint GRN inference taking advantage of the shared information across datasets while not making any specific parametric assumptions about the data. Second, during application to single cell data, scMSGL is kernelized as in Chapter 4 to take the structure of scRNA-seq data into account. For instance, it can employ proportionality measures to reflect relative rather than absolute abundance or zero-inflated Kendall's tau to handle drop-outs [230]. Finally, the proposed method learns an additional consensus graph, which captures the common structure across all graphs.

5.2 Methods

Let $\{\mathbf{X}^i\}_{i=1}^N$ be a given set consisting of N matrices. $\mathbf{X}_i \in \mathbb{R}^{n\times p_i}$ is a data matrix constructed from p_i graph signals defined on an unknown signed graph $G^i = (V, E^i, \mathbf{W}^i)$ with $|V| = n$. It is assumed that E^i's and associated edge weights are different but similar to each other. Based on this assumption, when learning G^i's, one can have better performance by borrowing information across graphs. For example, when analyzing scRNA-seq expressions from different disease states/conditions, the datasets generated from the varying groups are generally assumed to share a common gene co-expression structure. Thus, jointly learning cell-type specific graphs can improve inference by allowing information sharing across cell-types. To this end, we propose an optimization problem (scMSGL) that learns G^i's simultaneously. In the proposed approach, the learned G^i's are regularized to be close to a consensus graph G, which is also learned by combining information from G^i's. Thus, the proposed formulation ensures that information is shared across graphs when learning G^i's. Furthermore, the structure of G reflects the common connections shared across G^i's, whose inference may be beneficial if one is interested in learning the common gene co-expression structure over the different cell-types/disease-stage subgroups.

Let $\mathbf{L}^{i,+}$ and $\mathbf{L}^{i,-}$ be the Laplacian matrices of the positive and negative parts of G^i, respectively. Similarly, define $\mathbf{L}^+$ and $\mathbf{L}^-$ for the consensus graph G. Let $\mathcal{L}^+ = \{\mathbf{L}^{1,+}, \ldots, \mathbf{L}^{N,+}, \mathbf{L}^+\}$ and $\mathcal{L}^- = \{\mathbf{L}^{1,-}, \ldots, \mathbf{L}^{N,-}, \mathbf{L}^-\}$.

The optimization problem for jointly learning G^i's and G is then:

$$\begin{aligned}
\underset{\mathcal{L}^+,\mathcal{L}^-}{\text{minimize}} \quad & \sum_{s\in\{+,-\}}\sum_{i=1}^{N}\left\{\text{tr}(\mathbf{K}^{i,s}\mathbf{L}^{i,s}) + \alpha_s\|\mathbf{L}^{i,s}\|_F^2 + \beta_s\|\mathbf{L}^{i,s}-\mathbf{L}^s\|_{F,off}^2\right\} \\
& + \gamma_+\|\mathbf{L}^+\|_{1,off} + \gamma_-\|\mathbf{L}^-\|_{1,off} \\
\text{subject to} \quad & \mathbf{L}^{i,s}\in\mathbb{L},\ \text{tr}(\mathbf{L}^{i,s}) = 2n,\ \forall i,\ \forall s\in\{+,-\} \\
& (\mathbf{L}^{i,+},\mathbf{L}^{i,-})\in\mathbb{C}\ \forall i, \mathbf{L}^+,\ \mathbf{L}^-\in\mathbb{L},\ (\mathbf{L}^+,\mathbf{L}^-)\in\mathbb{C},
\end{aligned} \tag{5.1}$$

where $\mathbf{K}^{i,+} = \mathbf{K}^i$, $\mathbf{K}^{i,-} = -\mathbf{K}^i$, and $\mathbf{K}^i$ is a kernel matrix constructed from $\mathbf{X}^i$ as described in Section 4.2. $\|\cdot\|_{F,off}$ and $\|\cdot\|_{1,off}$ are the Frobenius norm and the $\boldsymbol{\ell}_1$-norm of the off-diagonal entries, respectively. The first term in the summation measures the smoothness and non-smoothness of $\mathbf{X}$ over $G^{i,+}$ and $G^{i,-}$, respectively. The second term controls the density of the learned $G^{i,+}$ ($G^{i,-}$) such that for larger values of α_+ (α_-), we learn denser graphs. The third term ensures that $G^{i,+}$ ($G^{i,-}$) are close to the positive (negative) part of consensus graph G with β_+ (β_-) controlling how close they should be. The last term is a regularizer that controls the sparsity of positive and negative parts of G with larger values of γ_+ and γ_- resulting in a sparser consensus graph. Finally, the constraints are the same as in (4.1).

5.2.1 Optimization

The problem in (5.1) can be written in a vectorized form, where one learns the upper triangular parts of the Laplacian matrices. Let $\mathbf{k}^{i,s} = \text{upper}(\mathbf{K}^{i,s})$, $\mathbf{d}^i = \text{diag}(\mathbf{K}^{i,s})$, $\boldsymbol{\ell}^{i,s} = \text{upper}(\mathbf{L}^{i,s})$ and $\boldsymbol{\ell}^s = \text{upper}(\mathbf{L}^s)$ for $s\in\{+,-\}$. Also, let $\mathcal{L}_v^+ = \{\boldsymbol{\ell}^{1,+},\ldots,\boldsymbol{\ell}^{N,+},\boldsymbol{\ell}^+\}$ and $\mathcal{L}_v^- = \{\boldsymbol{\ell}^{1,-},\ldots,\boldsymbol{\ell}^{N,-},\boldsymbol{\ell}^-\}$. The vectorized form of (5.1) is:

$$\begin{aligned}
\underset{\mathcal{L}_v^+,\mathcal{L}_v^-}{\text{minimize}} \quad & \sum_{s\in\{+,-\}}\sum_{i=1}^{N}\left\{\langle\mathbf{k}^{i,s}-\mathbf{S}^\top\mathbf{d}^{i,s},\boldsymbol{\ell}^{i,s}\rangle + \alpha\|-\mathbf{S}\boldsymbol{\ell}^{i,s}\|_2^2 + 2\alpha\|\boldsymbol{\ell}^{i,s}\|_2^2 + \beta\|\boldsymbol{\ell}^{i,s}-\boldsymbol{\ell}^s\|_2^2\right\} + \gamma_+\|\boldsymbol{\ell}^+\|_1 + \gamma_-\|\boldsymbol{\ell}^-\|_1 \\
\text{subject to} \quad & \mathbf{1}^\top\boldsymbol{\ell}^{i,+} = -n, \mathbf{1}^\top\boldsymbol{\ell}^{i,-} = -n, \boldsymbol{\ell}^{i,+}\le 0,\ \boldsymbol{\ell}^{i,-}\le 0,\ \boldsymbol{\ell}^{i,+}\perp\boldsymbol{\ell}^{i,-}\ \forall i \text{ and } \boldsymbol{\ell}^+\le 0,\ \boldsymbol{\ell}^-\le 0,\ \boldsymbol{\ell}^+\perp\boldsymbol{\ell}^-,
\end{aligned} \tag{5.2}$$

where $\mathbf{S}$ is defined in Section 1.1. The first term in the summation corresponds to the first term in (5.1), and the correspondence between the remaining terms to the term in (5.1) can be deduced using the hyperparameters. First two constraints correspond to the trace constraints in (5.1). The constraint $\boldsymbol{\ell}^{i,+}\perp\boldsymbol{\ell}^{i,-}$ together with $\boldsymbol{\ell}^{i,+}\le 0$ and $\boldsymbol{\ell}^{i,-}\le 0$ are complementarity constraints [217] and corresponds to $(\mathbf{L}^{i,+},\mathbf{L}^{i,-})\in C$ in (5.1).

The problem in (5.2) is non-convex due to complementarity constraints. However, ADMM is shown to be convergent for problems with complementarity constraints under some assumptions [251]. To write the

problem in standard ADMM form, introduce auxiliary variables $\mathbf{v}^i = \boldsymbol{\ell}^{i,+}$ and $\mathbf{w}^i = \boldsymbol{\ell}^{i,-}$ for all i. Similarly, introduce $\mathbf{v} = \boldsymbol{\ell}^+$ and $\mathbf{w} = \boldsymbol{\ell}^-$. Also, let $\mathcal{V} = \{\mathbf{v}^1, \ldots, \mathbf{v}^N, \mathbf{v}\}$ and $\mathcal{W} = \{\mathbf{w}^1, \ldots, \mathbf{w}^N, \mathbf{w}\}$. Then, the problem in its standard ADMM form is:

$$\begin{aligned} \underset{\mathcal{L}_v^+, \mathcal{L}_v^-, \mathcal{V}, \mathcal{W}}{\text{minimize}} \quad & \sum_{i=1}^{N} \iota_S(\mathbf{v}^i, \mathbf{w}^i) + \sum_{s\in\{+,-\}} \sum_{i=1}^{N} \left\{ f(\boldsymbol{\ell}^{i,s}, \boldsymbol{\ell}^s) + \iota_H(\boldsymbol{\ell}^{i,s}) \right\} + \iota_S(\mathbf{v}, \mathbf{w}) + \gamma_+ \|\boldsymbol{\ell}^+\|_1 + \gamma_- \|\boldsymbol{\ell}^-\|_1 \\ \text{subject to} \quad & \mathbf{v}^i = \boldsymbol{\ell}^{i,+}, \mathbf{w}^i = \boldsymbol{\ell}^{i,-}, \mathbf{v} = \boldsymbol{\ell}^+, \text{ and } \mathbf{w} = \boldsymbol{\ell}^-, \end{aligned} \tag{5.3}$$

where $f(\boldsymbol{\ell}^{i,s}, \boldsymbol{\ell}^s) = \langle \mathbf{k}^{i,s} - \mathbf{S}^\top \mathbf{d}^{i,s}, \boldsymbol{\ell}^{i,s} \rangle + \alpha \|-\mathbf{S}\boldsymbol{\ell}^{i,s}\|_2^2 + 2\alpha \|\boldsymbol{\ell}^{i,s}\|_2^2 + \beta \|\boldsymbol{\ell}^{i,s} - \boldsymbol{\ell}^s\|_2^2$, $\iota_S(\cdot,\cdot)$ is the indicator function for the complementarity set $S = \{(\mathbf{v}, \mathbf{w}) : \mathbf{v} \le 0,\ \mathbf{w} \le 0,\ \mathbf{v} \perp \mathbf{w}\}$, and $\iota_H(\cdot)$ is the indicator function for the hyperplane $H = \{\boldsymbol{\ell} : \mathbf{1}^\top \boldsymbol{\ell} = -n\}$. Augmented Lagrangian can then be written as:

$$\begin{aligned} L_p(\mathcal{L}_v^+, \mathcal{L}_v^-, \mathcal{V}, \mathcal{W}) = & \sum_{i=1}^{N} \iota_S(\mathbf{v}^i, \mathbf{w}^i) + \sum_{s\in\{+,-\}} \sum_{i=1}^{N} \left\{ f(\boldsymbol{\ell}^{i,s}, \boldsymbol{\ell}^s) + \iota_H(\boldsymbol{\ell}^{i,s}) \right\} \\ & + \sum_{i=1}^{N} \left\{ \lambda_{i,+}^\top (\mathbf{v}^i - \boldsymbol{\ell}^{i,+}) + \frac{\rho}{2} \|\mathbf{v}^i - \boldsymbol{\ell}^{i,+}\|_2^2 + \lambda_{i,-}^\top (\mathbf{w}^i - \boldsymbol{\ell}^{i,-}) + \frac{\rho}{2} \|\mathbf{w}^i - \boldsymbol{\ell}^{i,-}\|_2^2 \right\} \\ & + \iota_S(\mathbf{v}, \mathbf{w}) + \gamma_+ \|\boldsymbol{\ell}^+\|_1 + \gamma_- \|\boldsymbol{\ell}^-\|_1 \\ & + \lambda_+^\top (\mathbf{v} - \boldsymbol{\ell}^+) + \frac{\rho}{2} \|\mathbf{v} - \boldsymbol{\ell}^+\|_2^2 + \lambda_+^\top (\mathbf{w} - \boldsymbol{\ell}^-) + \frac{\rho}{2} \|\mathbf{w} - \boldsymbol{\ell}^-\|_2^2, \end{aligned} \tag{5.4}$$

where ρ is the parameter of augmented Lagrangian, $\lambda_{i,+}$, $\lambda_{i,-}$, λ_+ and λ_- are the Lagrange multipliers. Using augmented Lagrangian, ADMM steps at kth iteration are then found as follows:

$$(\widehat{\mathcal{V}}, \widehat{\mathcal{W}}) = \underset{\mathcal{V}, \mathcal{W}}{\text{argmin}} \quad L_p(\widehat{\widehat{\mathcal{L}}}_v^+, \widehat{\widehat{\mathcal{L}}}_v^-, \mathcal{V}, \mathcal{W}), \tag{5.5}$$

$$(\widehat{\mathcal{L}}_v^+, \widehat{\mathcal{L}}_v^+) = \underset{\mathcal{L}_v^+, \mathcal{L}_v^-}{\text{argmin}} \quad L_p(\mathcal{L}_v^+, \mathcal{L}_v^-, \widehat{\mathcal{V}}, \widehat{\mathcal{W}}), \tag{5.6}$$

$$\widehat{\lambda}_{i,+} = \widehat{\widehat{\lambda}}_{i,+} + \rho(\widehat{\mathbf{v}}^i - \widehat{\boldsymbol{\ell}}^{i,+}),\ \forall i, \tag{5.7}$$

$$\widehat{\lambda}_{i,-} = \widehat{\widehat{\lambda}}_{i,-} + \rho(\widehat{\mathbf{w}}^i - \widehat{\boldsymbol{\ell}}^{i,-}),\ \forall i, \tag{5.8}$$

$$\widehat{\lambda}_{+} = \widehat{\widehat{\lambda}}_{+} + \rho(\widehat{\mathbf{v}} - \widehat{\boldsymbol{\ell}}^{+}), \tag{5.9}$$

$$\widehat{\lambda}_{-} = \widehat{\widehat{\lambda}}_{-} + \rho(\widehat{\mathbf{w}} - \widehat{\boldsymbol{\ell}}^{+}), \tag{5.10}$$

where $\widehat{\ }$ and $\widehat{\widehat{\ }}$ represent the values of variables at kth and $(k-1)$th iteration, respectively. To solve (5.5), we use the fact that it can be solved for each $(\mathbf{v}^i, \mathbf{w}^i)$ pair (and $(\mathbf{v}, \mathbf{w})$), separately. This separation leads to a set of optimization problems all of which can be solved by projection onto the complementarity set S. The problem in (5.6) is separable across $\mathcal{L}_v^+$ and $\mathcal{L}_v^-$, leading to two optimization problems both of which can be solved with Block Coordinate Descent (BCD) [224].

5.2.2 Hyperparameter Selection Procedure

scMSGL requires the selection of six hyperparameters, three of which control the properties of the positive parts of the learned graphs while the remaining control the negative parts. As mentioned above, α_+ (α_-) and γ_+ (γ_-) control the edge density of positive (negative) parts of the learned G^i's and G, respectively. β_+ (β_-) controls how similar the learned $G^{i,+}$'s ($G^{i,-}$'s) are to the consensus graph. We select these hyperparameters similar to that suggested in [55], where hyperparameter selection is guided to learn graphs with desired properties. Alternative to other model selection approaches, such as cross-validation or Bayesian information criterion, this approach can achieve a model that is interpretable and plausible in practice. Thus, we tune the hyperparameters such that the obtained graphs have a desired edge density and view similarity. In particular, assume that one wants the densities of positive and negative edges in the learned G^i's and G to be d_+ and d_-, respectively. Furthermore, assume that the pairwise similarity between $G^{i,+}$ and $G^{j,+}$, $\forall i \neq j$ is desired to be c_+, where the similarity is quantified by the correlation coefficient. Similarly, let c_- be the desired similarity amount for the negative edges of the graphs. Once d_+, d_-, c_+, c_- are fixed, we select the six hyperparameters accordingly. The values of d_+, d_-, c_+, and c_- are selected based on prior knowledge on the datasets under study as detailed in Results section.

5.3 Results

The performance of scMSGL is evaluated on both simulated and two real scRNA-seq datasets. For simulated data, learned graphs are compared to ground truth networks to quantify the performance of scMSGL. Simulated data are used to benchmark the performance of scMSGL against scSGL and three GRN inference algorithms, GENIE3, GRNBOOST2 and PIDC, whose details are given in Chapter 4. These methods and scSGL can only learn a single graph from each dataset at a time. Therefore, they are applied to each $\mathbf{X}^i$ separately and the learned graphs are compared to ground truth G^i's. In addition, we benchmark against Joint Graphical Lasso with fused lasso penalty (JGL-Fused) method [55], which learns multiple related Gaussian graphical models, and Joint Gene Networks with scRNA-seq data (JGNsc) [66] algorithm, which jointly learns the graphs for multiple classes of single cell data.

As a performance metric we employed signed version of area under precision recall curve (AUPRC) ratio, which can measure how well a method can infer activating, inhibitory and non-existing edges. Given the ground truth GRN G and the output of a GRN inference algorithm $\widehat{G}$, let G^+ and G^- be the activating and inhibitory edges in the ground truth GRN and $\widehat{G}^+$ and $\widehat{G}^-$ be the activating and inhibitory edges in the

inferred network. We compare $\widehat{G}^+$ to G^+ with AUPRC to measure how well the algorithm finds the activating edges. Similarly, we compare $\widehat{G}^-$ to G^- to measure the performance on inhibitory edges. Let $AUPRC^+$ and $AUPRC^-$ represent these values. We calculate signed AUPRC ratio as follows:

$$SignedAUPRCRatio = \frac{1}{2}\left(\frac{AUPRC^+}{AUPRC^+_{random}} + \frac{AUPRC^-}{AUPRC^-_{random}}\right), \tag{5.11}$$

where $AUPRC^+_{random}$ and $AUPRC^-_{random}$ are the performance measures of a random estimator. Finally, note that if an algorithm infers an unsigned GRN, we use the inferred GRN for both $\widehat{G}^+$ and $\widehat{G}^-$.

5.3.1 Selected Hyperparameter Values

Hyperparameters of scMSGL are set as described in Section 5.2.2 section with $d_+ = d_- = d$ and $c_+ = c_- = c$. We used the BEELINE [190] pipeline to run GENIE3, GRNBOOST2 and PIDC. GENIE3 and GRNBOOST2 employs random forest and gradient boosting regressors, respectively and hyperparameters of these regressors are set to the default values used in GENIE3 and GRNBOOST2 toolboxes. PIDC uses mutual information to learn gene regulations and it requires a discretizer and an estimator for probability distribution estimation. We used the discretizer and estimator recommended by PIDC toolbox. scSGL requires α_+ and α_-, which are determined the same way as α_s's of scMSGL, i.e., they are set to values such that learned graphs have desired edge densities of $d_+ = d_- = d$. JGL-Fused requires two parameters λ_1 and λ_2, which are analogous to the parameter of scMSGL, α_s and β_s, respectively. Therefore, they are set the same way, i.e. we choose λ_1 and λ_2 such that the learned graphs' desired edge densities satisfy $d_+ + d_- = 2d$[1] and view similarity of $c_+ = c_- = c$. Finally, JGNsc consists of three steps: imputation, Gaussian transformation and GRN inference with JGL-Fused method. The hyperparameters of the first two steps are set to the default values provided in JGNsc toolbox and λ_1 and λ_2 of JGL-Fused step are set as described above[2].

For all datasets, we use $c = 0.5$. For simulated data, since benchmarking GRN inference methods (GENIE3, GRNBOOST2 and PIDC) learn fully connected graphs, we set $d = 0.4$ for fair comparison. For real data, we set $d = 0.1$ for ease of analysis.

[1] JGL-Fused does not allow edge densities of the negative and positive parts of the learned graph to be controlled separately, therefore we learned a graph with edge density equal to $2d$, which is the same edge density for scMSGL if the edge signs are not considered.

[2] JGNsc [66] recommends to use Akaike information criterion for selection of λ_1 and λ_2. In our analysis, we found this selection technique does not perform well and its time complexity was high.

5.3.2 Simulated Data

Data Generation: To validate the performance of scMSGL, we simulate gene expression data from a multivariate zero-inflated negative binomial (ZINB) distribution. Namely, given a known graph structure, we generate synthetic datasets using the algorithm described in Section 4.3.2 Two graph structures are considered for creating the baseline graph G with $n = 100$ genes: random graphs following an Erdős–Rényi (ER) model with an edge density of 0.1 and hub graphs following a Barabási–Albert (BA) model with a degree distribution that follows the power-law. We then convert G to a signed graph by randomly selecting half of the edges and assigning a negative sign to them while assigning a positive sign to the other half. Next, we generate $N = 5$ individual networks $\{G_i\}_{i=1}^N$ by adding $0.9 \times \binom{n}{2} \times \eta$ new edges to the baseline graph G. Half of the added edges are set as negative edges, while the other half are set as positive. The ZINB simulator is then used to generate datasets $\{X_i\}_{i=1}^N$ from the underlying graphs $\{G_i\}_{i=1}^N$. The three parameters of the ZINB distribution; λ, k and ω, which control its mean, dispersion and degree of zero-inflation, respectively were determined using a real scRNA-seq dataset [101]. Each simulation is repeated 10 times and the average performance over 10 realizations is reported.

Sensitivity to the Number of Cells: We first study the performance of the methods with varying number of cells when the dropout ratio is set to 0.26, $\eta = 0.1$, i.e. 90% of the edges are common across views and the correlation kernel is used for both scSGL and scMSGL. From left panel of Figure 5.1, it can be seen that for the different cell numbers, scMSGL has higher AUPRC ratios than methods that learn from a single dataset. This indicates that the proposed method incorporates valuable information across views, which improves the performance. As expected, the performance of all methods improves with increasing number of cells. These observations hold for both random graph models.

Sensitivity to Dropout Ratio: In the second analysis, we evaluate the performance of the different methods with increasing dropout ratio while fixing the number of cells to 400 and $\eta = 0.1$. Results are shown in the middle panel of Figure 5.1 for both random graph models, with correlation kernel used for scSGL and scMSGL. Similar to cell sensitivity analysis, scMSGL performs better compared to all other methods irrespective of which graph model is used to generate the datasets. Except for PIDC, AUPRC ratios of all methods drop with increasing dropout ratio as expected. Performance of PIDC mostly remains the same. Since PIDC performs poorly at all drop-out levels, this result does not imply robustness against dropouts.

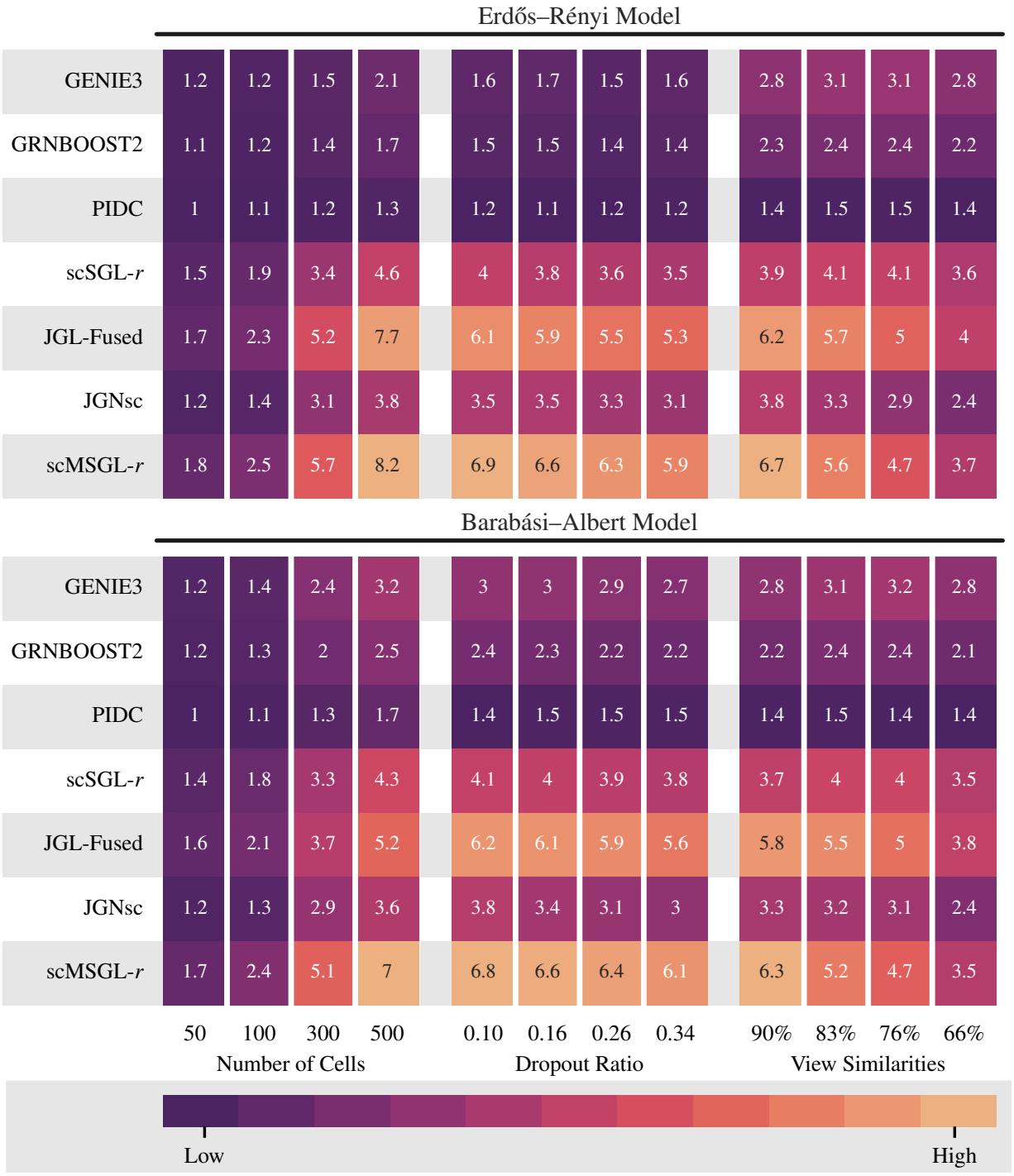

Figure 5.1: Performance of different methods on various datasets quantified by AUPRC ratio. All datasets have 100 genes. Left panel reports the results for varying number of cells. Middle one reports the results for varying dropout ratios. Right panel report results for varying degrees of view similarities, which is measured by the percentage of common edges across views in the ground truth graphs. Top plot shows the results for Erdős–Rényi model and the bottom plot shows the results for Barabási–Albert model.

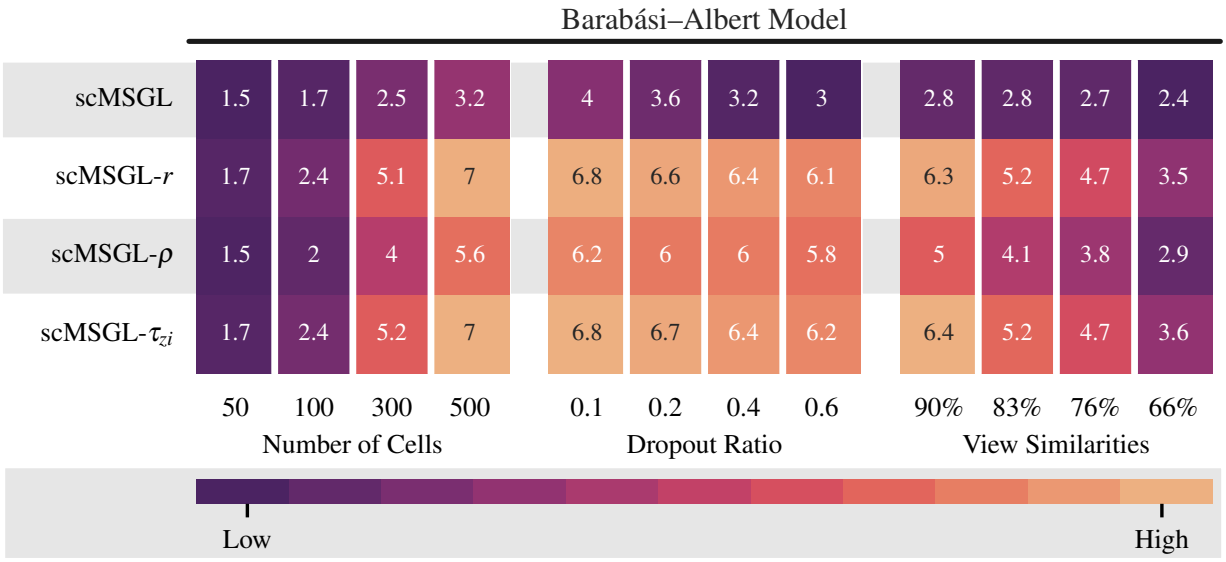

Figure 5.2: Performance of scMSGL without any kernel (first row) and with different kernels on datasets generated from BA model and studied in Figure 5.1.

Sensitivity to View Similarity: Next, we study the effect of view similarity on the performance of algorithms. Datasets are generated with varying η values while fixing the number of cells to 400 and the dropout ratio to 0.26. Results are reported in right panel of Figure 5.1, where the correlation kernel is employed for scSGL and scMSGL. When view similarity is 90%, the best performing algorithm is scMSGL, while for lower view similarity values JGL-Fused performs slightly better than scMSGL. The reason that JGL-Fused performs better than scMSGL for smaller view similarity values could be due to the difference in the regularization terms used to impose similarity across views. JGL-Fused uses a ℓ_1 norm penalty, while we employ a squared Frobenius norm. Compared to fused lasso, squared Frobenius norm is susceptible to outliers, which can degrade the performance. The performance of single-view algorithms does not get affected by the changes in view similarity, as they learn each view independently. On the other hand, there is a drop in the performances of all joint graph learning methods with decreasing view similarity. This is an expected behaviour, since both methods assume the dependence of views.

Kernel Comparison: Formulation of scMSGL allows us to use various kernels. Therefore, we study how the performance changes with respect to the kernel type. Datasets are created using the BA model and results are shown in Figure 5.2 for varying number of cells, dropout ratios and view similarities. The best performing kernel is τ_{zi}, followed by the correlation kernel. When Figures 5.1 and 5.2 are compared, scMSGL has higher AUPRC ratios than single-view approaches and JGNsc irrespective of the kernel choice.

The change in the performance of τ_{zi} and ρ with varying cell numbers, dropout ratios and view similarity are very similar to that of the correlation. Finally, to better understand the effect of kernels, the performance of scMSGL without any kernels, i.e. $\mathbf{K}^i = \mathbf{X}^i\mathbf{X}^{i\top}$, is also reported. Figure 5.2 shows all kernels have significantly higher performance compared to when no kernel is used, which indicates the importance of kernel usage in GRN inference.

Time Complexity Comparison: We compare the different methods based on their run time complexity. We generated datasets using BA model with varying number of cells and number of genes. Table 5.1 reports the run time of scSGL, scMSGL, JGL-Fused and JGNsc in seconds. Run times of GENIE3, GRNBOOST2 and PIDC are not reported as they are shown to have higher time complexity than scSGL in [112]. Reported run times correspond to one run without hyperparameter search. Run time of scSGL is the total run time to infer all views.

In the first dataset, number of genes, dropout ratio and η are fixed to 100, 0.26, and 0.1, respectively and number of cells varies. Results for this dataset indicate that scMSGL is faster than joint graph learning methods JGL-Fused and JGNsc. JGL-Fused also uses an ADMM based optimization, however it needs singular value decomposition at each ADMM iteration. scMSGL does not need this expensive operation; thus, it runs much faster than JGL-Fused. scSGL is faster than scMSGL, which is expected as scMSGL optimization takes longer time to converge due to added regularization terms and consensus graph learning. Finally, all methods except JGNsc are observed to run faster with increasing number of cells, since the inference problem becomes easier with higher number of cells, which makes iterative optimization procedure used by all methods converge faster. JGNsc runs slower with increasing number of cells, as its imputation step needs to handle a larger data matrix.

In the second dataset with increasing number of genes, the number of cells, dropout ration and η are fixed to 500, 0.26, and $\eta = 0.1$, respectively. As before, scMSGL is faster than joint graph learning methods and is slower than scSGL. Increasing the number of genes is observed to increase run time complexity of all methods, as it makes the problem harder.

5.3.3 Analysis of scRNA-seq data from mouse embryonic stem cell differentiation

Central to the differentiation process and many other cellular processes is the expression of right combination of genes or modules of genes. Accurate characterization of the co-expression networks for progenitor and multiple cell types can help in understanding the cascade of cellular state transitions [139]. In this

Table 5.1: Run time of scMSGL and benchmarking methods in seconds with respect to number of cells and genes. All methods run on the same computing cluster with compute nodes that have similar compute power. Run times of JGL-Fused and JGNsc for 500 genes are not reported, we were not able to run them in a reasonable time limit (4 hours).

	Number of Cells				Number of Genes			
Method	50	100	300	500	50	100	300	500
scSGL-*r*	1.10	0.54	0.35	0.38	0.15	0.37	5.88	26.68
JGL-Fused	175.64	117.65	95.95	98.03	10.02	95.98	1703.66	–
JGNsc	196.76	160.37	233.06	373.15	130.13	373.06	2541.45	–
scMSGL-*r*	14.00	12.39	10.00	8.51	0.35	3.89	110.49	304.71

section, we study the differentiation process of mouse embryonic stem cells (mESC) using single cell RNA sequencing datasets [118]. This data was generated using high-throughput droplet-microfluidic approach and was primarily used to study differentiation in mESC before and after leukemia inhibitory factor (LIF) withdrawal. Since LIF maintains pluripotency of mESC, LIF withdrawal is considered to initiate the differentiation process. The dataset contains cells sampled from 4 states (or natural subgroups): before LIF withdrawal, day 0 and after the withdrawal for days 2, 4 and 7. The subgroups contain 933, 303, 683 and 798 cells, respectively. This dataset has been previously analyzed using joint graphical estimation in [154, 255] and similar to them we only consider the 72 stem cell markers in our application [193] [3].

We first estimated the subgroup specific and the consensus graphs. Based on the results of simulated data, we employ the zero-inflated Kendall's tau kernel. Next, we calculate the signed node degrees of each gene, i.e., $D^+_{ii} = \sum_{j=1}^n W^+_{ij}$ and $D^-_{ii} = \sum_{j=1}^n W^-_{ij}$ from learned graphs G^+ and G^-. We then consider the genes with top signed degrees as hub genes whose signed degrees are reported in Figure 5.3. The result confirms the importance of regulator genes NANOG, SOX2, POU5F1, ZFP42, UTF1 in early stages of differentiation. NANOG has been reported to maintain pluripotency by inhibiting genes that activate differentiation to lineages associated with extraembryonic endoderm [40, 145]. Figure 5.3 clearly shows that the number of inhibitory relationships associated with NANOG decreases as the ES cells proceed to a matured state. POU5F1 and SOX2 also exhibit higher number of inhibitory relationships in the the first few days. SOX2, NANOG and POU5F1 are known to play a fundamental role in the self-renewal and pluripotency of mouse embryonic stem cells [270]. Reduction in expression of NANOG has been shown to be correlated with the induction of gene GATA4 which initiates differentiation of pluripotent cells [100] and therefore GATA4 has

[3]The dataset was downloaded from GEO database [73] (with ID GSE65525). For the preprocessing steps, please refer to "Data Analysis" subsection of the "Experimental Procedures" section in [118]. The only preprocessing we performed was log-transformation to make count data continuous.

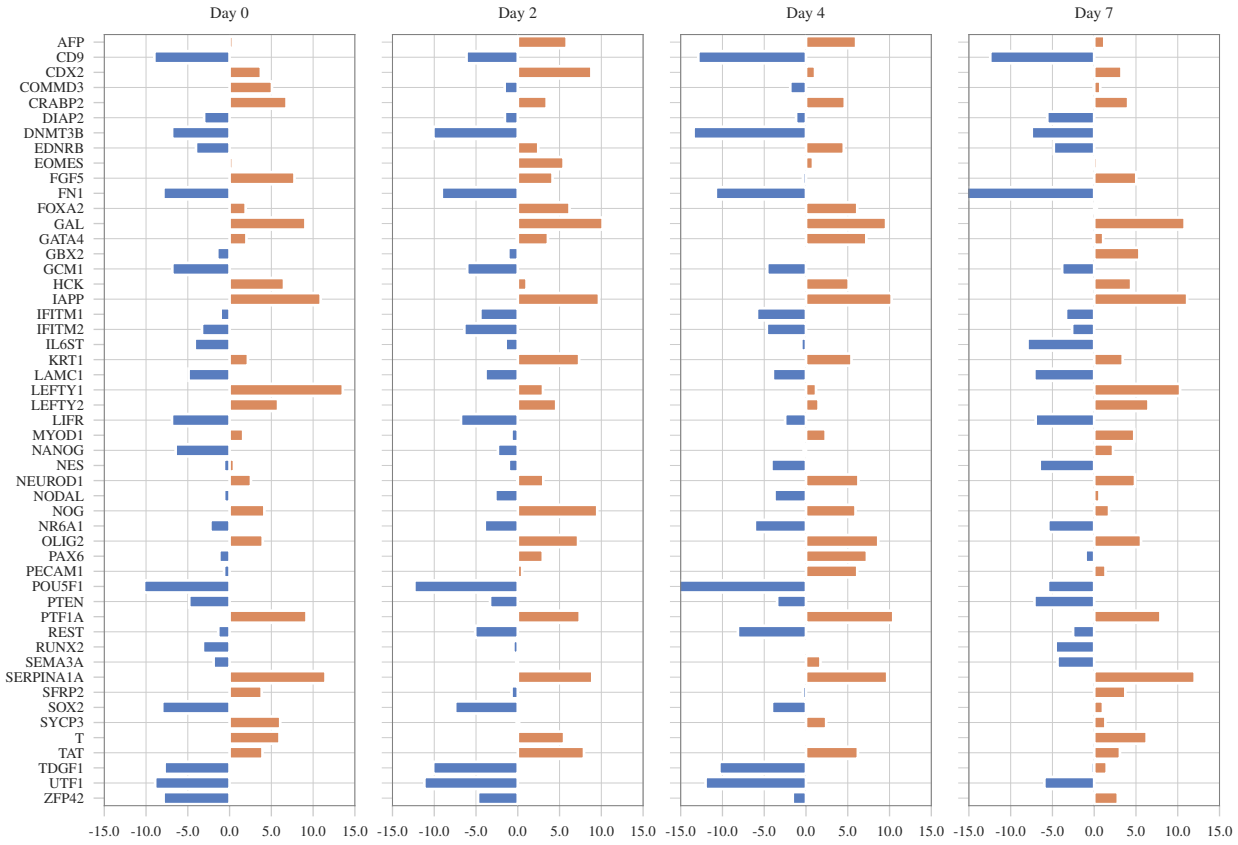

Figure 5.3: Genes with the highest node degrees. Orange and blue bars indicate that the degree is calculated using activating and inhibitory edges, respectively. Only genes whose activating or inhibitory degrees is among the top 15 genes in any view are shown.

Table 5.2: Node Degree of MYC in Learned Graphs

	Group 3	Intermediate	Group 4
Total Degree	5.436	3.334	4.180
Avg. Edge Weight	0.077	0.037	0.039

been correctly identified as a hub gene in Days 2 and 4. Collectively, these results confirm the fundamental roles of SOX2, NANOG and POU5F1 in the pluripotency stage and how an eventual reduction in their expression initiates differentiation.

Analysis of scRNA-seq data from medulloblastoma

Medulloblastoma (MB) is a highly malignant cerebellar tumor mostly affecting young children [176]. Several studies have been done to pinpoint the genetic drivers in each of the four distinct tumor subgroups: WNT-pathway-activated, SHH-pathway-activated, and the less-well-characterized Group 3 and Group 4 [176]. Among these subgroups, Group 3 and Group 4 tumors account for the majority of the MB diagnoses,

with Group 3 MB having a metastatic diagnosis of approximately 50%. Transcription factors (TFs) MYC2 and OTX2 have commonly been identified as key oncogenic TFs in Group 3 and 4 tumorogenesis. [66] used the joint single cell network algorithm to study the roles of MYC and OTX2 utilizing the MB scRNA-seq data set (GSE119926) by [101] [4]. Using the same selected samples from a subset of 17 individuals that were grouped into three subsets Group 3, Group 4 and an intermediate cell type, we estimate the joint gene regulatory network for the three groups for $\sim$ 750 genes among which most are enzyme-related genes from mammalian metabolic enzyme database [51]. Bulk profiling studies for MB cells have consistently observed overlapping transcriptional and epigenetic signatures in Group 3 and Group 4 tumors suggesting shared developmental origins [177, 101]. Based on this, we hypothesize that a joint analysis of the different MB cell-types would better capture the local functional interactions of MYC and OTX2 across different tumor subtypes and would eventually help in delineating their global role in regulating metabolic processes in MB cells.

Subgroup specific networks along with the consensus graph were estimated with zero-inflated Kendall's tau kernel. Table 5.2 shows that the average edge weight for the MYC network is considerably higher for Group 3 compared to Group 4 and the intermediate subgroup. Figure 5.4 further shows that Group 3 MYC network has stronger edge connections and higher density in compared to the intermediate group. In Group 4, almost all the connections become activating except for Aldh3a2 and Eno2; which were found to be strongly downregulated in all the tumor subgroups confirming their role in cancer resistance [151, 42]. This varying network structure over the subgroups confirms the major role MYC plays in initiation, maintenance, and progression of Group 3 tumors [205]. In Figure 5.4, it is shown that OTX2 has a denser network for Group 4 MB cells in comparison to the other groups. In Group 3 MB cells, OTX2's connections to the metabolic genes are very distinct from the MYC's. In addition, scMSGL detected relationships between OTX2 and metabolic genes PAICS and PPAT in Group 3 tumors. These genes related to the human purine biosynthesis pathways have been previously reported to be induced by MYC [129]. This confirms that OTX2 is functionally cooperating with MYC to regulate gene expression in medulloblastoma [205, 131]. Broadly these results suggest that MYC and OTX2 play significant roles in in the transcriptional regulation of the metabolic genes and the mechanisms underlying MYC and OTX2 mediated MB maintenance and

[4] A detailed overview of the MB scRNA-seq data generation and processing can be found in the Methods section of [101] under subsection "Human scRNA-seq data generation and processing". We log-transformed the obtained datasets to make count data continuous.

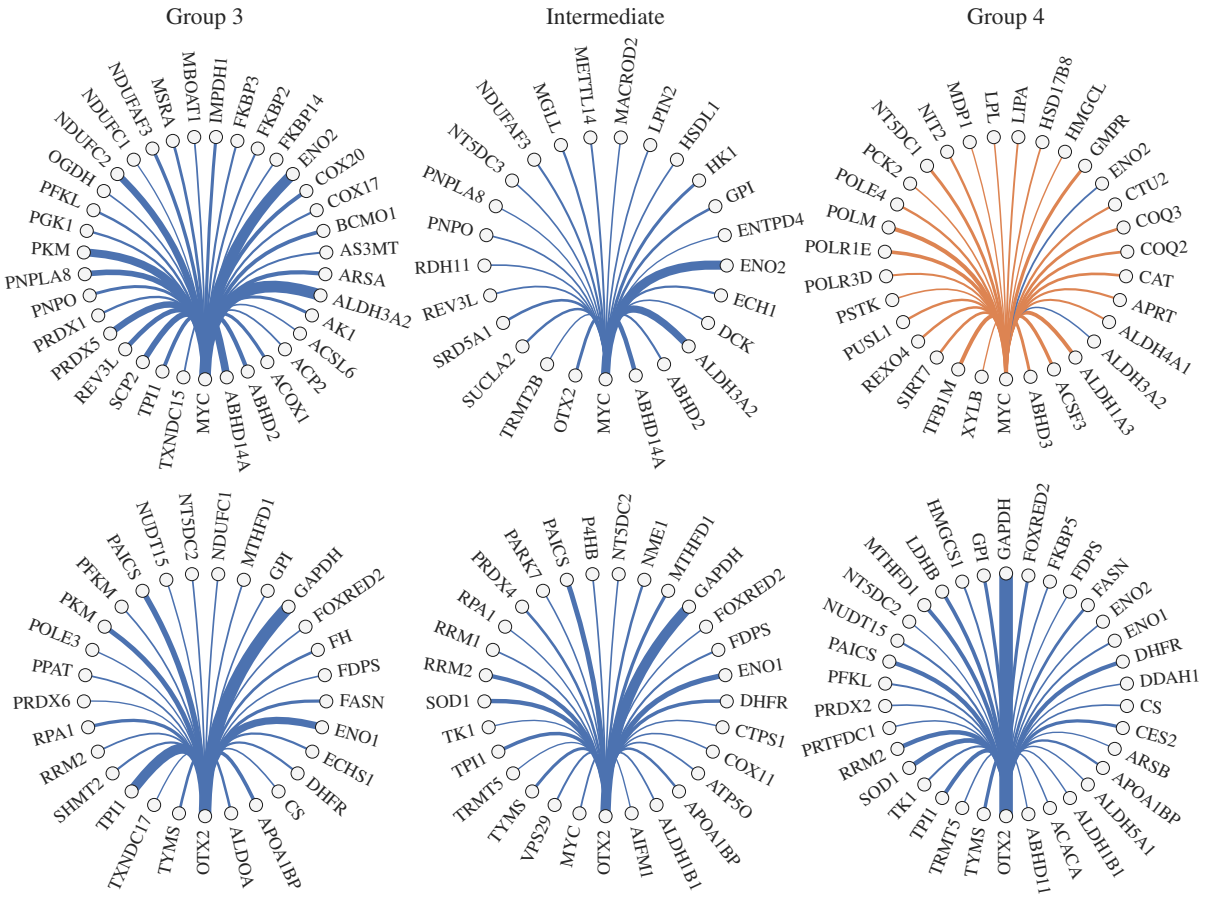

Figure 5.4: Connections of MYC (top) and OTX2 (bottom) genes. Edge widths are proportional to connection weights. Orange and blue edge colors indicate that the connection is activating and inhibitory, respectively. Only the top one third of the connections in all views of the multiview graph are shown.

progression likely vary in different subgroups of MB cells.

5.4 Conclusion

In this paper, we presented scMSGL for joint inference of multiple GRNs from scRNA-seq datasets having multiple classes. scMSGL learns functional relationships between genes across multiple related classes of single cell gene expression datasets under the assumption that there exists a shared structure across classes. The main novelty of our paper lies in the formulation of a highly efficient optimization framework that extends the signed graph learning [112] approach to high dimensional datasets with multiple classes. The kernelization trick embedded within the algorithm renders it capable of handling sparse and noisy features; expected to demonstrate highly non-linear relationships. Furthermore, the estimation of the consensus graph may help in understanding the joint structure existing within the multiple classes. Using simulation studies, we demonstrated the superior performance of scMSGL over single view learning and

existing joint learning methods for ER and BA graph models. In addition, performance was ascertained by varying a number of simulation parameters such as dropout levels, cell numbers and view similarity and scMSGL demonstrated superior performance in all scenarios. Applying scMSGL to the mESC dataset, we robustly identified previously reported regulatory markers as the hub genes for the different days and captured the progression of the differentiation process by analyzing these changes in hubs over the days. For the medulloblastoma data, scMSGL efficiently captured the significant roles that key oncology markers MYC and OTX2 play in the transcriptional regulation of metabolic genes.

There are various aspects of the proposed method that can be considered for improvement as future work. One challenge in implementing scMSGL is how to select the kernel function. This challenge can be addressed by combining information from multiple kernels during learning. An open problem in graph learning literature is hyperparameter selection, which is also a limitation of the proposed method. Current work selects the hyperparameters by searching the values that would result in graphs with desired properties. Future work can improve the accuracy of the learned graphs through better hyperparameter selection and multi-kernel strategies. Computational complexity of scMSGL is quadratic with respect to the number of genes (similar to scSGL) and linear in number of views. Therefore, its application to datasets with very large number of genes is not feasible. However, recent developments in GSP to scale GL to large-scale problems [108] can be exploited to scale scMSGL. Finally, additional sources of data that help in identifying direct interactions between TFs and target genes, can provide a way to filter out false positives. The current availability of single-cell epigenomic datasets has made it easier to further explore the regulatory relationship between TF and genes. Single-cell assay for transposase-accessible chromatin with sequencing (scATAC-seq), for example, allows the identification of DNA regulatory elements within accessible genomic DNA regions in single cells, hence enabling the identification of direct regulations in GRNs. Integration of multiomics profiles within the framework of scMSGL could be an interesting avenue for future research.

CHAPTER 6

SIMULTANEOUS GRAPH SIGNAL CLUSTERING AND GRAPH LEARNING

6.1 Introduction

In many modern data science applications, relationships between entities, such as features or data samples, are well described with a graph structure. While many real-world data are intrinsically graph-structured, *e.g.* social and traffic networks, there is still a large number of applications, where the graph topology is not readily available. For instance, gene regulations in biological applications or neuronal connections in the brain are not known. In these applications, the graphs need to be learned since they reveal the relational structure and may assist in a variety of learning tasks. Graph learning (GL) deals with the inference of a topological structure among entities from a set of observations on these entities, *i.e.* graph signals.

Methodologies to learn a graph from data include naive methods such as k-nearest neighborhood (k-NN), probabilistic graphical models [14, 54, 142, 102] and more recently GSP [137, 68] and graph neural networks (GNN) [269, 256, 34]. While the probabilistic graphical models assume the normality of the data, which is not true for most real-world data, GSP based GL methods define observations on a collection of nodes as graph signals and fall into two categories. The first category assumes graph signals are outcomes of diffusion processes on graphs and reconstructs a graph from signals according to the diffusion model [241, 182, 220, 219]. The second category of methods promotes the smoothness of graph signals quantified by the Laplacian quadratic form or more generally via total variation [69, 107, 23]. GNN-based methods, on the other hand, typically require a large volume of training data and the learned connectivity is often less explainable compared to probabilistic graph models and GSP methods.

Most of the work on GSP based GL has focused on the case where all data points follow the same relational model described by a single graph. However, in practice, the data may be coming from multiple graphs, i.e., multiview graphs. Examples of this setup include gene regulation networks where regulations vary across different cell types, and in social networks, where a set of users have varying interactions across different social media platforms. In this chapter, we address the problem of multiple graph learning from a heterogeneous set of graph signals, where each cluster is associated with a different graph structure. To this end, we propose GRASCale algorithm for simultaneous GRAph Signal Clustering And graph LEarning. Previous works that perform the same task employ only relations of the graph signals to the graphs associated with the clusters for clustering assignment. However, clustering algorithm can also benefit from side information in the form

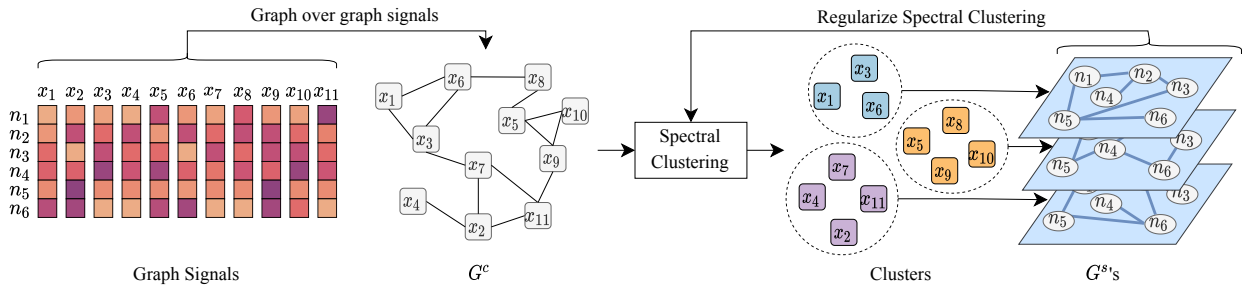

Figure 6.1: Overview of the proposed approach: Pairwise similarity between graph signals (G^c) and smoothness of the graph signals with respect to graphs associated with each cluster (G^s's) are used jointly in spectral clustering while simultaneously learning G^s's.

of pairwise relations between graph signals. For instance, in a recommendation system, when clustering graph signals, e.g. ratings for items, generated by a set of users, connections among the users can be used to inform the clustering algorithm. Therefore, we formulate GRASCale[1] with the following key contributions:

- We propose a new framework which is an extension of conventional spectral clustering where both the signals' pairwise similarity and smoothness with respect to the underlying graph structure are taken into account.
- The proposed methodology can learn the graph structures for mixed (heterogeneous) graph data.
- An efficient prox-linear block-coordinate descent (BCD) with improved consensus clustering based initialization is introduced for optimization.

The overall framework is depicted in Figure 6.1.

6.1.1 Related Work

Most of the existing work on GL considers simple data, where all data points follow the same model defined with only one graph. In recent work, GSP community has addressed the problem of learning multiple graphs from heterogenous data in two different settings: i) multiple views of the same data and ii) heterogenous data with possibly unknown cluster information.

The first class of methods, also known as joint inference of multiple graphs [164], considers the setting where multiple related networks each with a subset of observations is available. In this setting, the membership of the signals to the graphs is known and the graphs are closely related to each other. This problem setting has been most widely studied for inferring the topology of dynamic networks [110, 263, 13,

[1]Codes are available at the following github repository: https://github.com/SPLab-aviyente/GRASCale

212]. Assuming that the variation is smooth across time, the problem is reduced to learning multiple closely related graphs regularized with a term that promotes changes between consecutive graphs to be small in some pre-specified norm. More recently, the problem of joint inference of multiple graphs from the observed graph signals has been formulated with the assumption of graph stationarity [164]. In this formulation, the signals are assumed to be stationary, and pairwise similarity between all graphs is used to regularize the optimization.

The second class of methods focuses on the case where the data is heterogeneous and each subgroup has its own graph structure. This problem has been addressed for both the supervised and unsupervised settings. The supervised setting, also known as multi-category GL problem, assumes that the number of classes and the signals that belong to each class are known *a priori* [208, 111]. In this case, the goal is to learn multiple graphs each associated to a class of signals such that the representation of signals within a class and discrimination of signals in different classes are both taken into consideration. In the unsupervised setting, the number of clusters is known but the membership of the different graph signals is not known. In this case, the goal is to simultaneously cluster the data and learn the representative graph for each cluster [9, 136]. In [136], graph signals are modelled by a graph Laplacian mixture model (GLMM), which extends the factor analysis model of [69] to jointly model the smooth graph signals and identify the clusters through Gaussian mixture model (GMM). This model assumes that the number of clusters is known *a priori* and the distribution of the data is Gaussian. The model is fitted to data through the expectation-maximization algorithm for simultaneous graph inference and clustering. On the other hand, [9] proposes K-graphs, which is an extension of k-means clustering where the graph signals are assigned to the clusters based on their smoothness over each cluster's representative graph. Once the signals are clustered, the representative graphs are updated with graph learning algorithms. Both of GLMM and K-Graphs algorithms assign a graph signal to a cluster based on only the smoothness of the signal with respect to the graph associated with that cluster and do not explicitly take the pairwise relationships between the graph signals into account.

6.2 Method

6.2.1 Graph Signal Clustering with Regularized Graph Cut

Assume we are given a dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{p}$ where $\mathbf{x}_i \in \mathbb{R}^n$ is a graph signal over a graph $G^s \in \mathcal{G} = \{G^1, \ldots, G^k\}$. All graphs in $\mathcal{G}$ are defined over the same vertex set V with $|V| = n$ and have their own edge set E^s, i.e., $G^s = (V, E^s, \mathbf{W}^s)$, $\forall G^s \in \mathcal{G}$. Let the partitioning of graph signals in $\mathcal{X}$ be defined as

$C = \{C_1, \ldots, C_k\}$ where C_s includes all of the graph signals defined over G^s. In this paper, it is assumed that the partitioning of the graph signals, C, is not known *a priori*. The problem of learning C can be considered as a clustering problem. Let $G^c = (V^c, E^c, \mathbf{W}^c)$ be the graph that represents the similarity between the elements of $\mathcal{X}$ where V^c is the node set with $|V^c| = p$. Node $v_i^c \in V^c$ corresponds to $\mathbf{x}_i$ and w_{ij}^c is the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. C can then be learned by applying spectral clustering to G^c. However, spectral clustering as formulated in (1.4) does not use the fact that $\mathbf{x}_i$'s are graph signals. One can improve the clustering by incorporating information from the graphs in $\mathcal{G}$. Therefore, we propose a regularized graph cut (regcut) by assuming that the graph signals are smooth over the graphs they are defined on:

$$\text{regcut}(C) = \sum_{i,j=1}^{p} W_{ij}^c (1 - \delta_{g_i g_j}) + \alpha \sum_{s=1}^{K} \sum_{i=1}^{p} \delta_{g_i s} \mathbf{x}_i^\top \mathbf{L}^s \mathbf{x}_i \tag{6.1}$$

where $\mathbf{x}_i^\top \mathbf{L}^s \mathbf{x}_i$ is the smoothness of $\mathbf{x}_i$ over G^s as defined in Section 1.3. By regularizing the graph cut with smoothness, we ensure that if $\mathbf{x}_i$ is assigned to the sth cluster it is smooth with respect to G^s. As in Section 1.2.1, we relax $\mathbf{Z}$ to take on real values and obtain the following optimization problem:

$$\underset{\mathbf{Z} \in \mathbb{D}}{\text{minimize}} \quad \text{tr}(\mathbf{Z}^\top \mathbf{L}^c \mathbf{Z}) + \alpha \sum_{s=1}^{k} \text{tr}(\text{diag}(\mathbf{Z}_{\cdot s}) \mathbf{X}^\top \mathbf{L}^s \mathbf{X}), \tag{6.2}$$

where $\mathbf{X}$ is the data matrix with $\mathbf{X}_{\cdot i} = \mathbf{x}_i$ and $\mathbf{Z}$ is constrained as in Section 1.2.1.

6.2.2 Joint Graph Signal Clustering and Graph Learning

For the optimization problem in (6.2), one needs to know G^c and the graphs in $\mathcal{G}$. Since these graphs are generally not available, they need to be learned. G^c can be learned from $\mathbf{X}$ using the GL methods or more classical approaches such as k-nearest neighbor graphs. However, for graphs in $\mathcal{G}$, we cannot use these approaches as we do not know the partitioning of the graph signals. Thus, the graphs in $\mathcal{G}$ must be learned simultaneously with clustering. Therefore, we extend (6.2) with GL:

$$\underset{\mathbf{Z}, \mathbf{L}^1, \ldots, \mathbf{L}^k}{\text{minimize}} \quad \text{tr}(\mathbf{Z}^\top \mathbf{L}^c \mathbf{Z}) + \alpha_1 \sum_{s=1}^{k} \left[\text{tr}(\text{diag}(\mathbf{Z}_{\cdot s}) \mathbf{X}^\top \mathbf{L}^s \mathbf{X}) + (\mathbf{Z}_{\cdot s}^\top \mathbf{1}) \alpha_2 \|\mathbf{L}^s\|_F^2 \right] \tag{6.3}$$

$$\text{subject to} \quad \mathbf{Z} \in \mathbb{D}, \ \mathbf{L}^s \in \mathbb{L}, \text{tr}(\mathbf{L}^s) = 2n \ \forall s \in \{1, \ldots, k\}, \tag{6.4}$$

where each $\mathbf{L}^s$ is learned by assuming that graph signals in the sth cluster are smooth over G^s. As in (1.11), the Frobenius norm controls the sparsity of the learned graphs such that large values of α_2 result in denser graphs. However, in this setting we weigh this sparsity term with $\mathbf{Z}_{\cdot s}^\top \mathbf{1}$ which corresponds to the number of signals in cluster s to ensure that the sparsity levels of the learned graphs are similar for a given α_2. As the

value of the smoothness term increases with the number of signals in the cluster, multiplying the sparsity term with $\mathbf{Z}_{.s}^\top \mathbf{1}$ ensures that the relative importance of the sparsity term with respect to smoothness term remains similar across s. Finally, we set $\mathbb{D} = \{\mathbf{Z} \in \mathbb{R}^{p\times k} \mid \mathbf{Z} \geq 0, \mathbf{Z1} = \mathbf{1}\}$.

6.2.3 Optimization

The problem in (6.3) is a multi-convex problem, i.e., it is convex in each variable separately but non-convex when all variables are considered together. Therefore, we employ block coordinate descent (BCD) to solve (6.3) [224]. At each iteration of BCD, the problem is solved cyclically over each variable while fixing the remaining variables. When solving with respect to a variable, we perform inexact minimization with prox-linear update as it results in easy-to-solve problems with fast convergence when extrapolation is used [260]. Before applying BCD, we first vectorize (6.3) where we learn the upper triangular part of $\mathbf{L}^s$. Let $\boldsymbol{\ell}^s \in \mathbb{R}^m$ be the upper triangular part of $\mathbf{L}^s$ where $m = n(n-1)/2$. Define the operator mt with $\text{mt}(\boldsymbol{\ell}^s) = \mathbf{L}^s$. Then, (6.3) can be rewritten as:

$$\underset{\mathbf{Z},\mathbf{L}^1,\ldots,\mathbf{L}^k}{\text{minimize}} \quad \text{tr}(\mathbf{Z}^\top \mathbf{L}^c \mathbf{Z}) + \alpha_1 \sum_{s=1}^{k} \Big[\text{tr}(\text{diag}(\mathbf{Z}_{.s})\mathbf{X}^\top \text{mt}(\boldsymbol{\ell}^s)\mathbf{X}) + (\mathbf{Z}_{.s}^\top \mathbf{1})\alpha_2(2\langle \boldsymbol{\ell}^s, \boldsymbol{\ell}^s\rangle + \langle \mathbf{S}\boldsymbol{\ell}^s, \mathbf{S}\boldsymbol{\ell}^s\rangle)\Big] \tag{6.5}$$

$$\text{subject to} \quad \mathbf{Z} \geq 0, \mathbf{Z1} = \mathbf{1}, \boldsymbol{\ell}^s \leq 0, \mathbf{1}^\top \boldsymbol{\ell}^s = -n \;\forall s, \tag{6.6}$$

where $\mathbf{S}$ is defined in Section 1.1. Prox-linear updates at the tth iteration of BCD can then be found as follows:

$$\mathbf{Z}^{(t+1)} = \underset{\substack{\mathbf{Z}\geq 0,\\ \mathbf{Z1}=\mathbf{1}}}{\text{argmin}} \quad \langle \widehat{\mathbf{G}}_Z^{(t)}, \mathbf{Z} - \widehat{\mathbf{Z}}^{(t)}\rangle + \frac{\lambda_Z}{2}\|\mathbf{Z} - \widehat{\mathbf{Z}}^{(t)}\|_F^2, \tag{6.7}$$

$$\boldsymbol{\ell}^{s\,(t+1)} = \underset{\substack{\boldsymbol{\ell}^s \leq 0,\\ \mathbf{1}^\top \boldsymbol{\ell}^s = -n}}{\text{argmin}} \quad \langle \widehat{\mathbf{g}}_s^{(t)}, \boldsymbol{\ell}^s - \widehat{\boldsymbol{\ell}^s}^{(t)}\rangle + \frac{\lambda_s}{2}\|\boldsymbol{\ell}^s - \widehat{\boldsymbol{\ell}^s}^{(t)}\|_F^2, \tag{6.8}$$

where $\widehat{\mathbf{G}}_Z^{(t)}$ is the gradient of the objective function in (6.5) with respect to $\mathbf{Z}$ evaluated at $\widehat{\mathbf{Z}}^{(t)}$, $\widehat{\mathbf{g}}_s^{(t)}$ is the gradient with respect to $\boldsymbol{\ell}^s$ evaluated at $\widehat{\boldsymbol{\ell}^s}^{(t)}$, and:

$$\widehat{\mathbf{Z}}^{(t)} = \mathbf{Z}^{(t-1)} + w(\mathbf{Z}^{(t-1)} - \mathbf{Z}^{(t-2)}), \tag{6.9}$$

$$\widehat{\boldsymbol{\ell}^s}^{(t)} = \boldsymbol{\ell}^{s\,(t-1)} + w(\boldsymbol{\ell}^{s\,(t-1)} - \boldsymbol{\ell}^{s\,(t-2)}), \tag{6.10}$$

where $0 \leq w \leq 1$ is the extrapolation parameter. Finally, λ_Z and λ_s are step sizes and can be set to the Lipschitz constants of the gradient of the objective function in (6.5) with respect to $\mathbf{Z}$ and $\boldsymbol{\ell}^s$. Solutions of

Algorithm 6.1 GS Clustering with Simultaneous GL

Input: $\mathbf{X}$, $\mathbf{L}^s$, α_1, α_2, k and max_iter
Set $t \leftarrow 1$
Initialize $\mathbf{Z}^{(t)}$, $\mathbf{Z}^{(t-1)}$, $\boldsymbol{\ell}^{s\,(t)}$ and $\boldsymbol{\ell}^{s\,(t-1)}$
repeat
  Update $\mathbf{L}^{s\,(t+1)}$ with (6.8) **for** $s \in \{1, \ldots, k\}$
  Update $\mathbf{Z}^{(t+1)}$ with (6.7)
  Set $t \leftarrow t + 1$
until convergence or $t \geq$ max_iter
Output: $\mathbf{Z}^{(t)}$, $\mathbf{L}^{1\,(t)}, \ldots, \mathbf{L}^{k\,(t)}$

both (6.7) and (6.8) are projections onto simplex. In particular, for (6.8), we can rewrite it as follows:

$$\boldsymbol{\ell}^{s\,(t+1)} = \underset{\boldsymbol{\ell}^s}{\operatorname{argmin}} \quad \|\boldsymbol{\ell}^s - \widehat{\boldsymbol{\ell}}^{s\,(t)} + \frac{1}{\lambda_s}\widehat{\mathbf{g}}_s^{(t)}\|_F^2 \quad \text{subject to} \quad \boldsymbol{\ell}^s \leq 0, \mathbf{1}^\top \boldsymbol{\ell}^s = -n, \tag{6.11}$$

whose solution of is the projection of $\widehat{\boldsymbol{\ell}}^{s\,(t)} - \frac{1}{\lambda_s}\widehat{\mathbf{g}}_s^{(t)}$ onto the negative simplex, which can be performed efficiently using the algorithm described in [71]. To solve (6.7), we rewrite it as follows:

$$\mathbf{Z}^{(t+1)} = \underset{\mathbf{Z}}{\operatorname{argmin}} \quad \|\mathbf{Z} - \widehat{\mathbf{Z}}^{(t)} + \frac{1}{\lambda_Z}\widehat{\mathbf{G}}_Z^{(t)}\|_F^2 \quad \text{subject to} \quad \mathbf{Z} \geq 0, \mathbf{Z}\mathbf{1} = \mathbf{1} \tag{6.12}$$

which can be solved separately with respect to rows of $\mathbf{Z}$. Let $\mathbf{A} = \widehat{\mathbf{Z}}^{(t)} - \frac{1}{\lambda_Z}\widehat{\mathbf{G}}_Z^{(t)}$, then the subproblem of (6.12) with respect to ith row of $\mathbf{Z}$ is:

$$\mathbf{Z}_{i\cdot}^{(t+1)} = \underset{\mathbf{Z}_i}{\operatorname{argmin}} \quad \|\mathbf{Z}_{i\cdot} - \mathbf{A}_{i\cdot}\|_2^2 \quad \text{subject to} \quad \mathbf{Z}_{i\cdot} \geq 0, \mathbf{Z}_{i\cdot}^\top \mathbf{1} = 1, \tag{6.13}$$

whose solution is the projection of $\mathbf{A}_{i\cdot}$ onto the positive simplex, which can be performed efficiently using the algorithm described in [71].

Overall optimization procedure is given in Algorithm 6.1. [260] show that BCD with prox-linear update converges for multi-convex problems, when the objective function consists of smooth and separable non-smooth terms. The problem in (6.5) satisfies these assumptions; thus, Algorithm 1 is guaranteed to converge.

6.2.4 Initialization

BCD type algorithms may converge to poor local minima [224]. To overcome this problem, one can run the algorithm multiple times and consider the solution with the smallest objective value. One can also initiate the algorithm at a better point such that it converges to a solution with lower objective value. In this section, we describe a procedure to select better initializations for the proposed BCD algorithm.

Consider the set $\mathcal{Z} = \{\mathbf{Z}^1, \ldots, \mathbf{Z}^b\}$ which is obtained by running Algorithm 6.1 b times. Each $\mathbf{Z}^i$ indicates a possible partitioning of the graph signals. One can obtain a better clustering by combining

Algorithm 6.2 Initialization Procedure

Input: b
Initialize $\mathcal{Z}$ as an empty set
for $i \leq b$ **do**
    Run Algorithm 6.1 and add learned $\mathbf{Z}$ to $\mathcal{Z}$
end for
Find $\mathbf{Z}^0$ by applying consensus clustering to $\mathcal{Z}$
Run Algorithm 6.1 with initial point set to $\mathbf{Z}^0$
Output: Solutions of the last run

information from all Z^i's using consensus clustering [234], an ensemble learning method to combine multiple clusterings. We follow the consensus clustering procedure described in [121], where the consensus clustering $\mathbf{Z}^0$ is found from an association matrix $\mathbf{A}$ whose entries A_{ij} are equal to the number of times graph signals $\mathbf{x}_i$ and $\mathbf{x}_j$ are assigned to the same cluster in $\mathcal{Z}$. This association matrix can be used as the input to spectral clustering to find $\mathbf{Z}^0$. Once $\mathbf{Z}^0$ is found, we rerun the Algorithm 1 one more time, where $\mathbf{Z}$ is initialized at $\mathbf{Z}^0$ (the rest of the variables are initialized randomly). The clustering and learned graphs obtained from this run are used as the final result. This initialization procedure is given in Algorithm 6.2.

In our experiments, we set $b = 9$ and we set the maximum number of iterations for each run to a small number, e.g., 100, since even sub-optimal solutions can result in a good consensus clustering.

6.2.5 Hyperparameter Selection

The proposed method requires the selection of three hyperparameters: number of clusters k, α_1 and α_2. In literature, various methods have been proposed to determine the number of clusters in spectral clustering. These methods generally define a quality function Q and find the number of clusters as the value that optimizes Q. Possible choices of Q are eigengap [249], modularity [168], Bayesian information criterion (BIC) [209], integrated completed likelihood (ICL) [58]. α_2 controls the sparsity level of the learned graphs such that larger values of α_2 result in denser graphs. We set it to a value that results in graphs with a pre-determined sparsity level. This approach is similar to previous graph construction schemes, such as in k-NN graphs, where one wants to construct a graph with each node having at least k neighbors. The selection of α_1 is explained in detail through parameter sensitivity analysis in Section 6.3.1.

6.3 Results

In this section, the performance of GRASCale is evaluated on synthetic and real datasets and is compared to various state-of-the-art clustering and graph learning algorithms. We compare methods based on the quality of the resulting clustering as well as the accuracy of the learned graphs associated with each cluster. For

the first comparison, we consider normalized spectral clustering (SC), GLMM and K-Graphs. For the latter comparison, GL (see Section 1.3.1), GLMM and K-Graphs are considered. As mentioned in Section 1.2.1, SC clusters signals only based on their pairwise similarities. Thus, by comparing GRASCale to SC, we can illustrate the benefits of considering graph signal smoothness. GLMM and K-Graphs perform simultaneous graph signal clustering and graph learning similar to the proposed method. However, they only rely on the smoothness of the signals with respect to graphs associated with each cluster. By comparing GRASCale against them, we can illustrate the benefits of incorporating pairwise similarities. Finally, when applying GL, we assume the partitioning of the signals is known; thus the performance of GL provides an upper bound for the performance of GRASCale in the graph learning task. We used the formulation of [107] for implementing GL.

Parameter Selection: SC, GLMM, K-Graphs and GRASCale require the number of clusters k as an input. We provided the ground truth k as an input to all methods. GL, GLMM and K-graphs require a hyperparameter that controls the sparsity of the learned graphs similar to α_2 in (6.3). For all methods, we set this hyperparameter to a value that results in graphs with sparsity levels between 0.1 and 0.15[2]. GLMM and K-Graphs algorithms are based on alternating minimization, which causes their results to vary across runs. Therefore, we run each algorithm 10 times and report the average performance. For GRASCale, we set $b = 9$ as mentioned in Section 6.2.5. Thus, each algorithm is run 10 times. Finally, SC is applied to a binary k-nearest neighbor graph with the number of neighbors set to 5. The same graph is used as $\mathbf{L}^c$ for the proposed method.

Performance Metrics: Normalized mutual information (NMI) [57] is used to quantify the performance of clustering. For the graph learning task, F1-score is used to quantify how close the learned graphs are to the ground truth graphs. We measure F1-score for all s and report the average.

6.3.1 Synthetic Data

Data Generation: Given a graph G with Laplacian $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, we can generate a graph signal $\mathbf{x}$ that is smooth with respect to G by filtering a given signal $\mathbf{x}_0$ with a low-pass graph filter [69, 107]. Mathematically, this is equivalent to $\mathbf{x} = h(\mathbf{L})\mathbf{x}_0$ where $h(\mathbf{L}) = \mathbf{V}h(\mathbf{\Lambda})\mathbf{V}^\top$ is a low-pass graph filter. Based on this, we generate the synthetic data as follows. We first generate k graphs $\mathcal{G} = \{G^1, \ldots, G^k\}$ based on a random graph model,

[2]Real-world graphs are generally sparse, so it is desirable to learn sparse graphs. Therefore, we learn graphs at this range of sparsity level. In our experiments, we observed smaller values sparsity level can result in disconnected graphs. To prevent this, we did not consider smaller values.

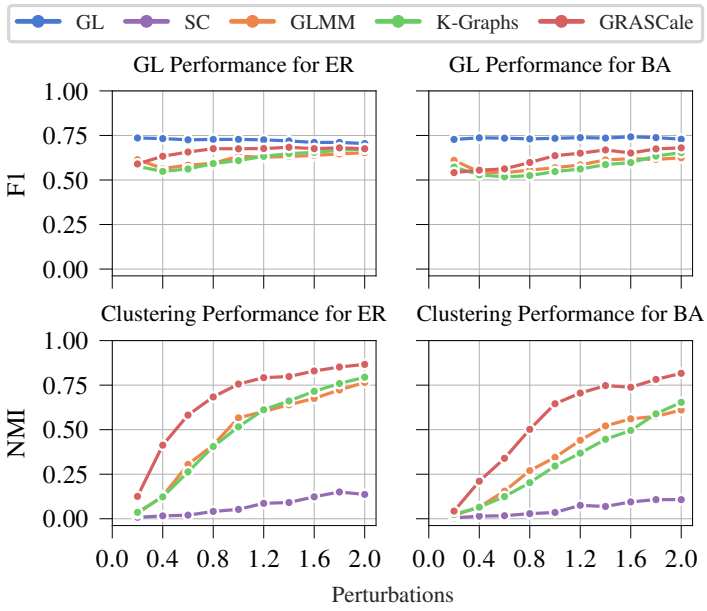

Figure 6.2: Results for Experiment 1 when cluster sizes are equal. Upper row illustrates the graph learning performance and the bottom row shows the clustering performance. Left and right columns are performances for ER and BA graph models, respectively.

such as Erdős–Rényi (ER) [86] or Barabási–Albert (BA) models [5], where each G^s has n nodes. For each G^s, we generate p_s smooth graph signals as described above with $h(\mathbf{\Lambda}) = \sqrt{\mathbf{\Lambda}}^{\dagger}$ and $\mathbf{x}_0 \sim \mathcal{P}$, where † is the pseudo-inverse operator and $\mathcal{P}$ is a probability distribution to be determined. The graph signals are then used to construct data matrices $\mathbf{X}^s \in \mathbb{R}^{n \times p_s}$, from which we build $\mathbf{X} = [\mathbf{X}^1, \ldots, \mathbf{X}^s] \in \mathbb{R}^{n \times p}$ where $p = p_1 + \cdots + p_s$. White Gaussian noise with variance equal to 10% of the signal power is added to the data matrix. Finally, we generate 20 different realizations of each dataset in all experiments and report the average performance across realizations.

Experiment 1: In this experiment, we generate signals from $\mathcal{G} = \{G^1, G^2, G^3\}$ where each G^s is generated by swapping the edges of a given graph G $\lceil m_G \times pert \rceil$ times. m_G is the number of edges in G and $pert > 0$ refers to the amount of perturbation. Smaller values of $pert$ causes graphs in $\mathcal{G}$ to be highly correlated; thus, clustering the graph signals generated from these graphs becomes a harder task. We generated G with 50 nodes from two random graph models: ER with edge probability $p_{ER} = 0.1$ and BA model with $m_{BA} = 3$. We generated $\mathbf{X}$ as described above with $\mathcal{P} = \mathcal{N}(0, \mathbf{I})$. In Figure 6.2, we report the results when the cluster sizes are equal, i.e., $p_s = 200$ for all s. It can be observed that the clustering performance for all methods increases with the amount of perturbation. This is due to the fact that as the perturbation level increases, the

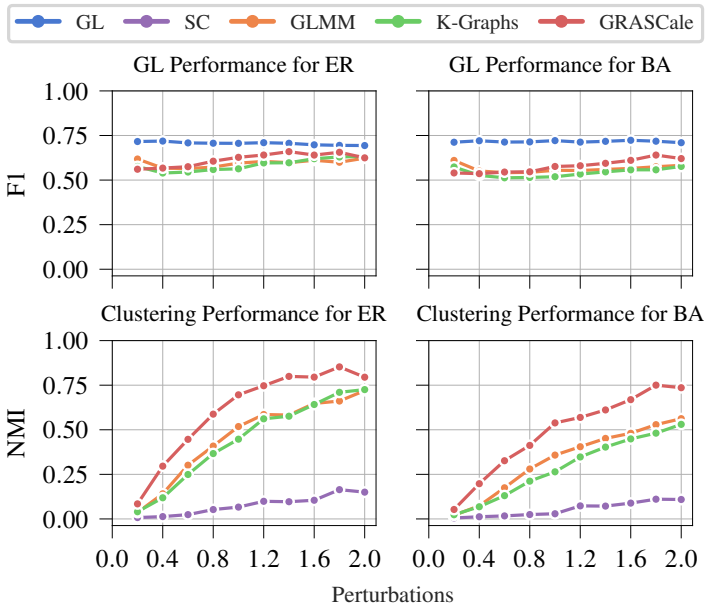

Figure 6.3: Results for Experiment 1 when cluster sizes are different. Upper row shows graph learning performance and bottom row shows clustering performance. Left and right columns are performances for ER and BA graph models, respectively.

different clusters become more distinct. GRASCale performs better than GLMM and K-Graphs for both ER and BA models. SC is observed to perform very poorly as the signals are generated independently from each other. Thus, pairwise similarities between signals that are in the same cluster are not strong, resulting in low NMI values for SC. In terms of graph learning, GL performs the best as expected since it assumes that the cluster membership of the signals is known *a priori*. There is a slight improvement in the graph learning performances of GLMM, K-Graphs and GRASCale as perturbation level increases and their performances converge to that of GL. Graph learning performances of GLMM, K-Graphs and the proposed method for small perturbation levels may seem counter-intuitive considering their low NMI values. However, graphs in $\mathcal{G}$ are very correlated for small values of perturbation, thus graph signals in a given cluster carry information about other graphs too. Therefore, methods can still perform well for graph inference even though the graph clusters may not be accurately identified.

Figure 6.3 illustrates the results for the same simulation setting when there is heterogeneity in cluster sizes, i.e., $p_1 = 300$, $p_2 = 200$, and $p_3 = 100$. Results are very similar to that of Figure 6.2. There is a slight drop in the performance of all algorithms compared to Figure 6.2 across all perturbation levels and graph models.

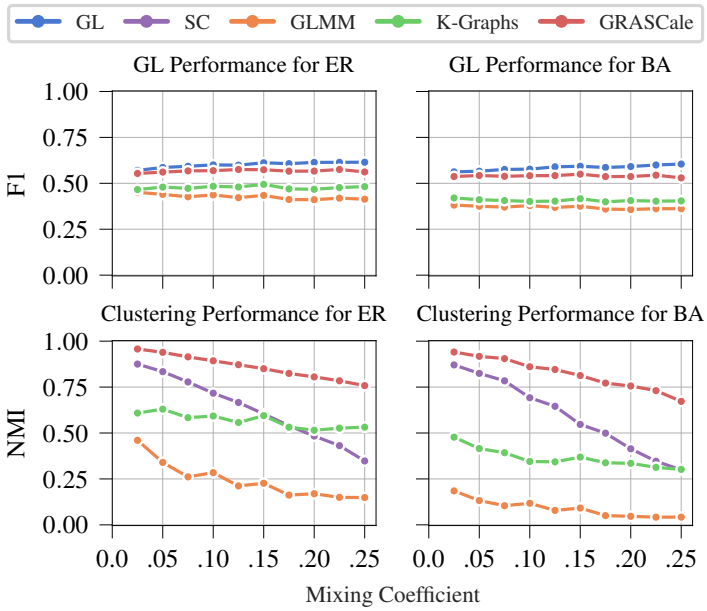

Figure 6.4: Results for Experiment 2. Upper row shows graph learning performance and bottom row shows clustering performance. Left and right columns are performances for ER and BA graph models, respectively.

Experiment 2: In the previous experiment, signals were generated independently; thus they do not have any explicitly imposed pairwise relations. In this experiment, we generate graph signals that have pairwise relations and are also smooth with respect to graphs associated with clusters. In order to achieve this goal, we first generate a data matrix $\mathbf{Y} \in \mathbb{R}^{n \times p}$ with $n = 50$ and $p = 600$. Rows of $\mathbf{Y}$ are generated by filtering a signal $\mathbf{y} \in \mathbb{R}^p$ through a low-pass graph filter defined on G^c. The signal similarity graph G^c has p nodes and $\mathbf{y} \sim \mathcal{N}(0, \mathbf{I})$. If there is an edge between nodes v_i and v_j in G^c, columns $\mathbf{Y}_{\cdot i}$ and $\mathbf{Y}_{\cdot j}$ will be similar to each other. We construct G^c from a planted partition model [49] whose nodes are partitioned into three equal sized clusters: $C_1 = \{v_1, \ldots, v_{200}\}$, $C_2 = \{v_{201}, \ldots, v_{400}\}$, and $C_3 = \{v_{401}, \ldots, v_{600}\}$. Planted partition model has two parameters p_{in} and p_{out}, which determine the intra- and inter-cluster connectivity, respectively. We set $p_{in} = 0.05(1 - \mu)$ and $p_{out} = 0.05\mu$, where $\mu > 0$ is the mixing coefficient. Larger values of μ causes the clusters to be less distinguishable. For the low-pass filter, we used a heat kernel $h(\mathbf{\Lambda}^c) = \exp(-5\mathbf{\Lambda}^c)$ where $\mathbf{\Lambda}^c$ is the eigenvalue matrix corresponding to the Laplacian matrix of G^c [107]. We generated graphs in $\mathcal{G}$ as in the first experiment with $pert$ set to 2. Once $\mathbf{Y}$ and $\mathcal{G}$ are generated, columns of $\mathbf{Y}$ in C_s are filtered by the graph filter corresponding to $G^s \in \mathcal{G}$ for all s to construct $\mathbf{X}$.

Figure 6.4 shows the performance of the different algorithms. With the introduction of pairwise similarity within clusters, the performance of SC is observed to improve significantly. However, its NMI value is still

lower than GRASCale, since the latter benefits from both pairwise relations and smoothness of the graph signals. GLMM and KGraphs have lower performance than the proposed method, as these methods employ only smoothness of the graph signals. Increasing the mixing coefficient causes a decrease in the performance of all methods, as larger values of μ result in less distinguishable clusters. The decrease in NMI values for SC and GRASCale with increasing μ values follows a similar trend. This indicates that the proposed method indeed uses the pairwise relations between signals. For the graph learning task, F1 score of the proposed method is higher than those of GLMM and K-Graphs and is very close to that of GL due to its high clustering performance.

Parameter Sensitivity: We study the sensitivity of the performance of GRASCale to the selection of α_1 and α_2 on a dataset from Experiment 2. We consider a dataset generated from the BA graph model with μ set to 0.25. The ground truth graph has a density around 0.12 in this dataset. We apply our algorithm to this dataset with varying α_1 and α_2 values and the performances are reported in Figure 6.5. For the x-axis, densities of the learned graph are used rather than the values of α_2. Figure 6.5 shows that the density of learned graphs is important for the performance. In particular, low density graphs have poor performance in terms of F1 and NMI, as these graphs are very sparse and do not contain enough information. Similarly, high density values also result in low performance, since learned graphs include many false positive edges. Finally, this figure also shows that the proposed method is not sensitive to the value of α_1 as long as the learned graphs have a reasonable density. In particular, there is a large range of α_1 values, where F1-score and NMI are stable. Based on this observation, we set $\alpha_1 = 10$ in all of our data analysis without any fine-tuning.

6.3.2 Real Data

In this section, the proposed method is applied to a real world data clustering problem, where the aim is to cluster the digits of MNIST dataset while learning a graph for each digit. More specifically, we selected 400 images corresponding to digits 0, 1, 2 and 3. After vectorizing each image, we obtain a data matrix of size 400×1600, where the rows and columns correspond to pixels and images, respectively. SC, GLMM, K-Graphs and GRASCale are applied to the constructed data matrix and the clustering performance is reported in Figure 6.6. The best performing method is GRASCale, and it is followed by SC; while GLMM and K-Graphs have significantly lower performance. These results indicate that using pairwise similarities of the signals and their smoothness together improve the clustering performance.

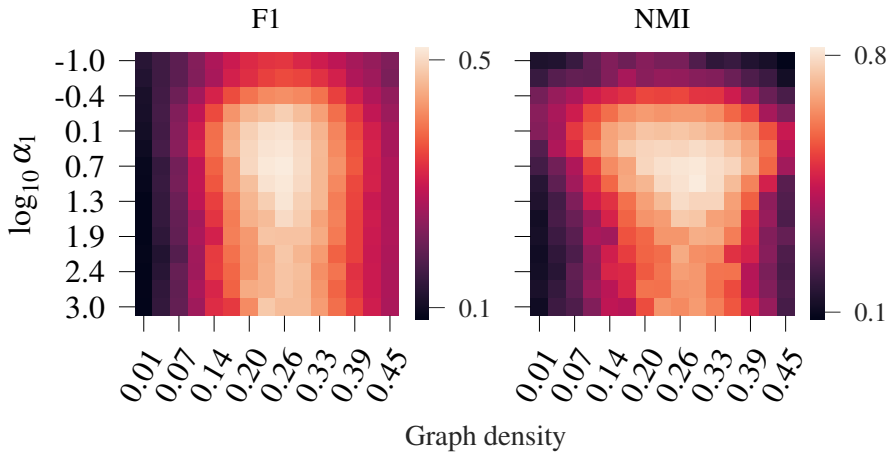

Figure 6.5: Sensitivity of F1 and NMI values to varying values of α_1 and density of learned graphs. Left panel shows the graph learning performance and the right panel shows the clustering performance.

As mentioned in [136], learning a graph for each cluster can be helpful for the interpretablity of clustering. By analyzing the graph structure learned for each cluster, one can deduce why a set of graph signals are assigned to the same cluster; which leads to explainable data science [202]. In Figure 6.7, we plot the graphs learned for each digit by GRASCale. It can be seen that the method learns very interpretable graph structures. The learned graphs for digits 0, 2, 3 have high resemblance to the digits themselves. Although the graph found for digit 1 has a meaningful structure, it is noisier than the other graphs. This is due to the fact that there is a lot of variation across samples for writing digit 1. This means that while we tend to cluster digits based on their numerical values, it might be the case that there is also a clustering within each digit based on the writing style.

6.4 Conclusions

In this paper, we presented GRASCale for simultaneous graph signal clustering and graph learning. Compared to previous methods developed for the same task, GRASCale uses two types of information:

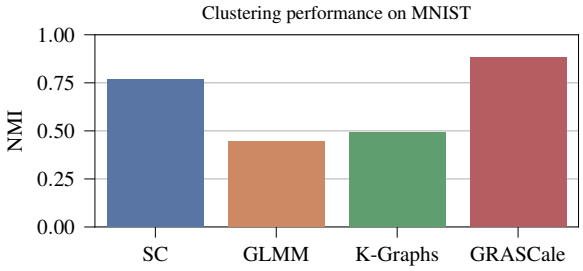

Figure 6.6: Clustering performance for MNIST dataset.

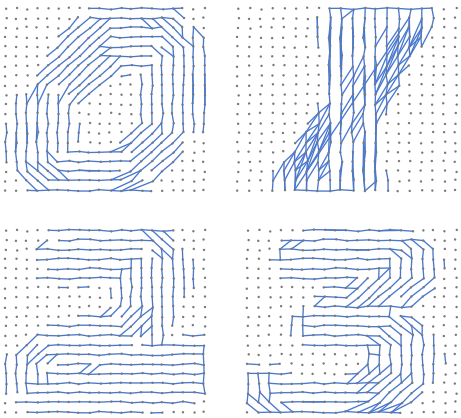

Figure 6.7: Graph structures learned for each digit by the proposed method. Points correspond to pixels, while lines indicate the inferred edges between pixels. Only top 300 edges are shown.

pairwise relations between graph signals and their smoothness with respect to graphs associated with clusters. Our results on synthetic and real datasets indicate that incorporating these complementary pieces of information within the same framework improves clustering and graph learning performance significantly.

In the presented formulation of GRASCale, we assumed $\mathbf{L}^c$ is constructed *a priori*; however, this graph can also be learned along with clusters and graphs associated with clusters. In future work, we will consider this extension of jointly learning $\mathbf{L}^c$ along with the individual graphs, $\mathbf{L}^s$.

CHAPTER 7

CONCLUSIONS

Community detection and graph learning are two important problems in network science and graph signal processing. The former problem deals with topological analysis of graphs to identify their mesoscale organization; while, graph learning aims to infer the interactions between nodes of a graph from data when the graph topology is not known *a priori*. Existing community detection and graph learning methods are mostly limited to single-layer graphs and they cannot handle multilayer graphs efficiently. In this thesis, we aimed to fill this gap by proposing multiple community detection and graph learning methods for various types of multilayer graphs.

Dynamic networks are a type of multilayer networks, where layers correspond to different time points and interlayer edges are only allowed between consecutive time points. Existing community detection methods for dynamic networks identify the community structure of each time point while regularizing the identified community structures to change smoothly across time. However, it is not known how to set the regularization parameter that determines how smoothly community structure changes across time. In Chapter 2, we answered this question based on recent theoretical developments that explain community detection algorithms using statistical models. In particular, we proposed a new dynamic stochastic blockmodel which models the community changes across time with a Markov random field. Fitting the proposed model to an observed dynamic network was then shown to be equivalent to evolutionary spectral clustering under some assumptions. This equivalence was employed to determine the regularization parameter of evolutionary spectral clustering and to propose two novel spectral clustering based algorithms for dynamic networks. Performance of the proposed algorithms was investigated using simulated and real data; and it is observed that they outperform existing dynamic community detection methods.

Human brain operates at different frequency bands and the functional connectivity between brain regions is different at each band. Recent developments aim to study these functional networks simultaneously through multilayer network modeling, where each layer corresponds to a frequency band. However, existing work is mostly limited to multiplex networks, where interlayer edges are only allowed between nodes that represent the same brain region. In Chapter 3, we addressed this shortcoming by proposing a multilayer community detection algorithm, which is especially tailored for multi-frequency brain networks. First, phase synchrony and phase amplitude coupling measures were used to construct a multilayer EEG network, where interlayer

edges are allowed between any two brain regions. Next, we proposed a multilayer modularity metric to detect communities in the constructed networks. An important characteristic of multi-frequency brain networks is the heterogeneity in edge weights across frequency bands, which can bias community detection methods to partition nodes based on layers rather than the true community structure. Therefore, the proposed modularity metric was developed based on a new null model which preserves this heterogeneity. We parameterized the proposed metric to handle resolution limit of the modularity and to be able to control importance of interlayer edges. Finally, a new method that can address the degeneracy of modularity maximization was proposed to identify group community structure of a set of subjects. The proposed approach was applied to EEG data collected during a study of error monitoring in the human brain. The results revealed important differences in the brain organization following error and correct responses.

Regulatory interactions between genes can be studied with networks, where nodes and edges correspond to genes and their regulatory relations, respectively. An important characteristic of gene regulatory networks (GRN) is that they are signed graphs, where edge signs represent activating and inhibitory regulations. Existing GSP based graph learning methods cannot be used to infer GRNs, since they are restricted to learn only unsigned graph topologies. Therefore, in Chapter 4, we proposed a GSP based signed graph learning approach, which models a signed graph as a two layer multiplex network where one layer corresponds to positive edges while the other corresponds to the negative edges. We then devised an optimization problem to learn each layer based on the assumption that graph signals are smooth and non-smooth over positive and negative layers, respectively. The optimization problem was further kernelized to be able to handle various characteristics of observed graph signals such as missing values or non-linearity. The proposed problem was solved with an efficient ADMM based optimization procedure. We employed the proposed signed graph learning method to identify GRN from single cell gene expression data. The method was benchmarked against state of the art GRN inference methods on simulated and real data and it was shown that it outperforms them in terms of accuracy and computational time complexity.

Given multiple datasets, each of which includes graph signals defined on a different signed graph, we can apply the method presented in Chapter 4 to each dataset separately to learn multiple signed graphs. When it is assumed that the multiple signed graphs are related, this approach will be suboptimal since it does not impose any shared structure on the learned signed graphs. Therefore, in Chapter 5, we extended the signed graph learning approach proposed in Chapter 4 to learn multiple related signed graphs. Namely, multiple signed graphs were learned simultaneously by solving an optimization problem that assumes smoothness

and non-smoothness of the datasets as in Chapter 4. Furthermore, we imposed a shared structure to learned signed graphs through a regularization term, that ensures the learned graphs are similar to a consensus graph. Our optimization procedure also learned the consensus graph, which represents the shared structure of the learned signed graphs. We employed the method for the inference of multiple related GRNs from single cell datasets that were generated from multiple treatment conditions or disease states. Results on simulated data showed that the proposed approach has better performance than methods that can learn a single graph at a time and previous joint multiple GRN reconstruction algorithms. Real data analysis revealed that the method learns signed graphs that are inline with previous biological findings.

Existing work on multiple unsigned graph learning assumes that we are given multiple datasets, each of which includes graph signals defined on a different graph. However, there are applications where we are given a single heterogeneous dataset, which consists of graph signals from multiple clusters and each cluster includes graph signals defined on a graph. In such cases, the aim is jointly cluster graph signals and infer the graphs associated with clusters. In Chapter 6, we proposed an algorithm for this task. Compared to existing work, the novelty of the method is that it partitions the graph signals not only based on their smoothness with respect to the graphs associated with the clusters but also their pairwise similarities. The method is developed by extending graph cut based clustering. It can also learn the representative graph for each cluster using the smoothness of the graph signals. The results on simulated and real data indicate the effectiveness of the proposed method compared to existing algorithms.

7.1 Future Work

In this section, we present some future research directions that can be considered to address the shortcomings of the algorithms presented in this thesis.

Community Detection in Multiplex Networks: Dynamic community detection methods proposed in Chapter 2 were developed based on showing the equivalence between evolutionary spectral clustering and statistical modeling. This approach can be followed to propose a multiplex community detection algorithm. Existing work on community detection in multiplex networks identify community structures of layers while regularizing them based on some assumptions, such as there is a set of nodes which are in the same community across all layers. Such assumptions can be used to propose new statistical models for multiplex networks. We can then answer the question of under which conditions fitting these models to an observed multiplex network is equivalent to existing multiplex community detection algorithms. As in Chapter 2, proving this

equivalence can pave the way for developing novel multiplex community detection algorithms and addressing the shortcomings of the existing ones.

Multi-aspect Multilayer Community Detection: An important task in brain network analysis is to study networks from multiple subjects simultaneously, which helps one to understand characteristics that are shared and different across subjects. In Chapter 3, we performed this by finding multilayer community structures of each subject independently and the shared community structure across subjects was found by group community detection. However, this approach is suboptimal as subjects' multilayer networks are processed independently. This shortcoming can be handled by using multi-aspect multilayer approach, which is an extension of multilayer networks to multiple dimensions. In multi-aspect multilayer network, the layer set is the product of sets of elementary layers, *i.e.* $\mathcal{L} = \mathcal{L}_1 \times \mathcal{L}_2 \ldots \mathcal{L}_d$, where $\mathcal{L}_i$ is the set of elementary layers [117]. In our case, $\mathcal{L}_1$ is the set of frequency bands and $\mathcal{L}_2$ is that set of subjects. Thus, in our multi-aspect multilayer network, each layer includes the interactions of a subject's frequency band. Future work can develop new community detection algorithms for multi-aspect multilayer networks.

Multiple Signed Graph Learning from Heterogeneous Datasets: Multiple signed graph learning method of Chapter 5 is designed for cases where multiple datasets are available. However, some problems include only a single heterogeneous dataset, which needs to be clustered while learning a signed graph for each cluster. For example, in cell type-specific GRN inference, a single scRNA-seq dataset is often used and the goal is to identify cell types and learn a GRN for each type. In Chapter 6, we proposed an approach which simultaneously performs clustering and learns unsigned graphs. Future work can extend this work to signed graphs, where graph signals will be clustered while a signed graph will be learned for each cluster.

BIBLIOGRAPHY

[1] Sara Aibar et al. "SCENIC: single-cell regulatory network inference and clustering". In: *Nature methods* 14.11 (2017), pp. 1083–1086.

[2] Christopher Aicher, Abigail Z Jacobs, and Aaron Clauset. "Learning latent block structure in weighted networks". In: *Journal of Complex Networks* 3.2 (2015), pp. 221–248.

[3] Edo M Airoldi et al. "Mixed membership stochastic blockmodels". In: *Advances in neural information processing systems* 21 (2008).

[4] Kyle Akers and TM Murali. "Gene regulatory network inference in single-cell biology". In: *Current Opinion in Systems Biology* 26 (2021), pp. 87–97.

[5] Réka Albert and Albert-László Barabási. "Statistical mechanics of complex networks". In: *Reviews of modern physics* 74.1 (2002), p. 47.

[6] Alberto Aleta and Yamir Moreno. "Multilayer networks in a nutshell". In: *Annual Review of Condensed Matter Physics* 10 (2019), pp. 45–62.

[7] Genevera I Allen and Zhandong Liu. "A local poisson graphical model for inferring networks from sequencing data". In: *IEEE transactions on nanobioscience* 12.3 (2013), pp. 189–198.

[8] Gerrit Ansmann and Klaus Lehnertz. "Constrained randomization of weighted networks". In: *Physical Review E* 84.2 (2011), p. 026103.

[9] Hesam Araghi, Mohammad Sabbaqi, and Massoud Babaie–Zadeh. "K-Graphs: An Algorithm for Graph Signal Clustering and Multiple Graph Learning". In: *IEEE Signal Processing Letters* 26.10 (2019), pp. 1486–1490.

[10] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. "An event-based framework for characterizing the evolutionary behavior of interaction graphs". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 3.4 (2009), pp. 1–36.

[11] Selin Aviyente and Ali Yener Mutlu. "A time-frequency-based approach to phase and phase synchrony estimation". In: *IEEE Transactions on Signal Processing* 59.7 (2011), pp. 3086–3098.

[12] Selin Aviyente et al. "A phase synchrony measure for quantifying dynamic functional integration in the brain". In: *Human brain mapping* 32.1 (2011), pp. 80–93.

[13] Brian Baingana and Georgios B Giannakis. "Tracking switched dynamic network topologies from information cascades". In: *IEEE Transactions on Signal Processing* 65.4 (2016), pp. 985–997.

[14] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data". In: *The Journal of Machine Learning Research* 9 (2008), pp. 485–516.

[15] Albert-László Barabási and Réka Albert. "Emergence of scaling in random networks". In: *science* 286.5439 (1999), pp. 509–512.

[16] Albert-László Barabási and Eric Bonabeau. "Scale-free networks". In: *Scientific american* 288.5 (2003), pp. 60–69.

[17] Danielle S Bassett and Olaf Sporns. "Network neuroscience". In: *Nature neuroscience* 20.3 (2017), pp. 353–364.

[18] Danielle S Bassett et al. "Learning-induced autonomy of sensorimotor systems". In: *Nature neuroscience* 18.5 (2015), pp. 744–751.

[19] Federico Battiston et al. "Multilayer motif analysis of brain networks". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.4 (2017), p. 047404.

[20] Marya Bazzi et al. "A framework for the construction of generative models for mesoscale structure in multilayer networks". In: *Physical Review Research* 2.2 (2020), p. 023100.

[21] Marya Bazzi et al. "Community detection in temporal multilayer networks, with an application to correlation networks". In: *Multiscale Modeling & Simulation* 14.1 (2016), pp. 1–41.

[22] Andrea Berger and MI Posner. "Pathologies of brain attentional networks". In: *Neuroscience & Biobehavioral Reviews* 24.1 (2000), pp. 3–5.

[23] Peter Berger, Gabor Hannak, and Gerald Matz. "Efficient graph learning from noisy and incomplete data". In: *IEEE Transactions on Signal and Information Processing over Networks* 6 (2020), pp. 105–119.

[24] Richard F Betzel and Danielle S Bassett. "Multi-scale brain networks". In: *Neuroimage* 160 (2017), pp. 73–83.

[25] Richard F Betzel et al. "The community structure of functional brain networks exhibits scale-specific patterns of inter-and intra-subject variability". In: *Neuroimage* 202 (2019), p. 115990.

[26] Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[27] Stefano Boccaletti et al. "Complex networks: Structure and dynamics". In: *Physics reports* 424.4-5 (2006), pp. 175–308.

[28] Stefano Boccaletti et al. "The structure and dynamics of multilayer networks". In: *Physics reports* 544.1 (2014), pp. 1–122.

[29] Stephen P Borgatti. "Centrality and network flow". In: *Social networks* 27.1 (2005), pp. 55–71.

[30] DA Brafman et al. "Regulation of endodermal differentiation of human embryonic stem cells through integrin-ECM interactions". In: *Cell Death & Differentiation* 20.3 (2013), pp. 369–381.

[31] Anatol Bragin et al. “Gamma (40-100 Hz) oscillation in the hippocampus of the behaving rat”. In: *Journal of neuroscience* 15.1 (1995), pp. 47–60.

[32] Ulrik Brandes et al. “On modularity clustering”. In: *IEEE transactions on knowledge and data engineering* 20.2 (2007), pp. 172–188.

[33] Urs Braun et al. “Dynamic reconfiguration of frontal brain networks during executive cognition in humans”. In: *Proceedings of the National Academy of Sciences* 112.37 (2015), pp. 11678–11683.

[34] Michael M Bronstein et al. “Geometric deep learning: going beyond euclidean data”. In: *IEEE Signal Processing Magazine* 34.4 (2017), pp. 18–42.

[35] Matthew J Brookes et al. “A multi-layer network approach to MEG connectivity analysis”. In: *Neuroimage* 132 (2016), pp. 425–438.

[36] Javier M Buldú and Mason A Porter. “Frequency-based brain networks: From a multiplex framework to a full multilayer description”. In: *Network Neuroscience* 2.4 (2018), pp. 418–441.

[37] Ed Bullmore and Olaf Sporns. “Complex brain networks: graph theoretical analysis of structural and functional systems”. In: *Nature reviews neuroscience* 10.3 (2009), pp. 186–198.

[38] Lian En Chai et al. “A review on the computational approaches for gene regulatory network construction”. In: *Computers in biology and medicine* 48 (2014), pp. 55–65.

[39] Tanmoy Chakraborty et al. “Metrics for community analysis: A survey”. In: *ACM Computing Surveys (CSUR)* 50.4 (2017), pp. 1–37.

[40] Ian Chambers et al. “Functional expression cloning of Nanog, a pluripotency sustaining factor in embryonic stem cells”. In: *Cell* 113.5 (2003), pp. 643–655.

[41] Thalia E Chan, Michael PH Stumpf, and Ann C Babtie. “Gene regulatory network inference from single-cell data using multivariate information measures”. In: *Cell systems* 5.3 (2017), pp. 251–267.

[42] Peter Mu-Hsin Chang et al. “Transcriptome analysis and prognosis of ALDH isoforms in human cancer”. In: *Scientific reports* 8.1 (2018), pp. 1–10.

[43] Chuan Chen, Michael Ng, and Shuqin Zhang. “Block spectral clustering for multiple graphs with inter-relation”. In: *Network Modeling Analysis in Health Informatics and Bioinformatics* 6.1 (2017), pp. 1–22.

[44] Geng Chen, Baitang Ning, and Tieliu Shi. “Single-cell RNA-seq technologies and related computational data analysis”. In: *Frontiers in genetics* 10 (2019), p. 317.

[45] Shuonan Chen and Jessica C Mar. “Evaluating methods of inferring gene regulatory networks highlights their lack of performance for single cell gene expression data”. In: *BMC bioinformatics* 19.1 (2018), pp. 1–21.

[46] Yun Chi et al. “Evolutionary spectral clustering by incorporating temporal smoothness”. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007, pp. 153–162.

[47] Li-Fang Chu et al. “Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm”. In: *Genome biology* 17.1 (2016), pp. 1–20.

[48] Michael W Cole and Walter Schneider. “The cognitive control network: integrated cortical regions with dissociable functions”. In: *Neuroimage* 37.1 (2007), pp. 343–360.

[49] Anne Condon and Richard M Karp. “Algorithms for graph partitioning on the planted partition model”. In: *Random Structures & Algorithms* 18.2 (2001), pp. 116–140.

[50] ENCODE Project Consortium et al. “An integrated encyclopedia of DNA elements in the human genome”. In: *Nature* 489.7414 (2012), p. 57.

[51] Callan C Corcoran et al. “From 20th century metabolic wall charts to 21st century systems biology: database of mammalian metabolic enzymes”. In: *American Journal of Physiology-Renal Physiology* 312.3 (2017), F533–F542.

[52] Marco Corneli, Pierre Latouche, and Fabrice Rossi. “Exact ICL maximization in a non-stationary temporal extension of the stochastic block model for dynamic networks”. In: *Neurocomputing* 192 (2016), pp. 81–91.

[53] L da F Costa et al. “Characterization of complex networks: A survey of measurements”. In: *Advances in physics* 56.1 (2007), pp. 167–242.

[54] Alexandre d’Aspremont, Onureena Banerjee, and Laurent El Ghaoui. “First-order methods for sparse covariance selection”. In: *SIAM Journal on Matrix Analysis and Applications* 30.1 (2008), pp. 56–66.

[55] Patrick Danaher, Pei Wang, and Daniela M Witten. “The joint graphical lasso for inverse covariance estimation across multiple classes”. In: *Journal of the Royal Statistical Society. Series B, Statistical methodology* 76.2 (2014), p. 373.

[56] Weidong Dang et al. “Rhythm-dependent multilayer brain network for the detection of driving fatigue”. In: *IEEE Journal of Biomedical and Health Informatics* (2020).

[57] Leon Danon et al. “Comparing community structure identification”. In: *Journal of statistical mechanics: Theory and experiment* 2005.09 (2005), P09008.

[58] J-J Daudin, Franck Picard, and Stéphane Robin. “A mixture model for random graphs”. In: *Statistics and computing* 18.2 (2008), pp. 173–183.

[59] Manlio De Domenico. “Multilayer modeling and analysis of human brain networks”. In: *GigaScience* 6.5 (2017), pp. 1–8.

[60] Manlio De Domenico and Jacob Biamonte. "Spectral entropies as information-theoretic tools for complex network comparison". In: *Physical Review X* 6.4 (2016), p. 041062.

[61] Manlio De Domenico, Shuntaro Sasai, and Alex Arenas. "Mapping multiplex hubs in human functional brain networks". In: *Frontiers in neuroscience* 10 (2016), p. 326.

[62] Manlio De Domenico et al. "Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems". In: *Physical Review X* 5.1 (2015), p. 011027.

[63] Arnaud Delorme and Scott Makeig. "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis". In: *Journal of neuroscience methods* 134.1 (2004), pp. 9–21.

[64] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. "Weighted graph cuts without eigenvectors a multilevel approach". In: *IEEE transactions on pattern analysis and machine intelligence* 29.11 (2007), pp. 1944–1957.

[65] Stavros I Dimitriadis. "Assessing The Repeatability of Multi-Frequency Multi-Layer Brain Network Topologies Across Alternative Researchers Choice Paths". In: *bioRxiv* (2021).

[66] Meichen Dong et al. "Joint gene network construction by single-cell RNA sequencing data". In: *Biometrics* (2022).

[67] Xiaowen Dong et al. "Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds". In: *IEEE Transactions on signal processing* 62.4 (2013), pp. 905–918.

[68] Xiaowen Dong et al. "Learning graphs from data: A signal representation perspective". In: *IEEE Signal Processing Magazine* 36.3 (2019), pp. 44–63.

[69] Xiaowen Dong et al. "Learning Laplacian matrix in smooth graph signal representations". In: *IEEE Transactions on Signal Processing* 64.23 (2016), pp. 6160–6173.

[70] Karl W Doron, Danielle S Bassett, and Michael S Gazzaniga. "Dynamic network structure of interhemispheric coordination". In: *Proceedings of the National Academy of Sciences* 109.46 (2012), pp. 18661–18668.

[71] John Duchi et al. "Efficient projections onto the l 1-ball for learning in high dimensions". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 272–279.

[72] Nathan Eagle, Alex Pentland, and David Lazer. "Inferring friendship network structure by using mobile phone data". In: *Proceedings of the national academy of sciences* 106.36 (2009), pp. 15274–15278.

[73] Ron Edgar, Michael Domrachev, and Alex E Lash. "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository". In: *Nucleic acids research* 30.1 (2002), pp. 207–210.

[74] Bradley Efron and Robert Tibshirani. "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy". In: *Statistical science* (1986), pp. 54–75.

[75] Mark WEJ Fiers et al. "Mapping gene regulatory networks from single-cell omics data". In: *Briefings in functional genomics* 17.4 (2018), pp. 246–254.

[76] Greg Finak et al. "MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data". In: *Genome biology* 16.1 (2015), pp. 1–13.

[77] JB Fisher et al. "GATA6 is essential for endoderm formation from human pluripotent stem cells". In: *Biology open* 6.7 (2017), pp. 1084–1095.

[78] Francesco Folino and Clara Pizzuti. "An evolutionary multiobjective approach for community discovery in dynamic networks". In: *IEEE Transactions on Knowledge and Data Engineering* 26.8 (2013), pp. 1838–1852.

[79] Santo Fortunato. "Community detection in graphs". In: *Physics reports* 486.3-5 (2010), pp. 75–174.

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

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

[82] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. "Sparse inverse covariance estimation with the graphical lasso". In: *Biostatistics* 9.3 (2008), pp. 432–441.

[83] Luz Garcia-Alonso et al. "Benchmark and integration of resources for the estimation of human transcription factor activities". In: *Genome research* 29.8 (2019), pp. 1363–1375.

[84] Amir Ghasemian, Homa Hosseinmardi, and Aaron Clauset. "Evaluating overfit and underfit in models of network community structure". In: *IEEE Transactions on Knowledge and Data Engineering* 32.9 (2019), pp. 1722–1735.

[85] Amir Ghasemian et al. "Detectability thresholds and optimal algorithms for community structure in dynamic networks". In: *Physical Review X* 6.3 (2016), p. 031005.

[86] Edgar N Gilbert. "Random graphs". In: *The Annals of Mathematical Statistics* 30.4 (1959), pp. 1141–1144.

[87] Anna Goldenberg et al. "A survey of statistical network models". In: *Foundations and Trends® in Machine Learning* 2.2 (2010), pp. 129–233.

[88] 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.

[89] Dominic Grün, Lennart Kester, and Alexander Van Oudenaarden. "Validation of noise models for single-cell transcriptomics". In: *Nature methods* 11.6 (2014), pp. 637–640.

[90] Jian Guo et al. "Joint estimation of multiple graphical models". In: *Biometrika* 98.1 (2011), pp. 1–15.

[91] Min Jin Ha, Veerabhadran Baladandayuthapani, and Kim-Anh Do. "DINGO: differential network analysis in genomics". In: *Bioinformatics* 31.21 (2015), pp. 3413–3420.

[92] Heonjong Han et al. "TRRUST v2: an expanded reference database of human and mouse transcriptional regulatory interactions". In: *Nucleic acids research* 46.D1 (2018), pp. D380–D386.

[93] Dongxiao He et al. "A network-specific Markov random field approach to community detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[94] Randolph F Helfrich and Robert T Knight. "Oscillatory dynamics of prefrontal cognitive control". In: *Trends in cognitive sciences* 20.12 (2016), pp. 916–930.

[95] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. "Latent space approaches to social network analysis". In: *Journal of the american Statistical association* 97.460 (2002), pp. 1090–1098.

[96] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. "Kernel methods in machine learning". In: *The annals of statistics* (2008), pp. 1171–1220.

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

[98] Clay B Holroyd and Michael GH Coles. "The neural basis of human error processing: reinforcement learning, dopamine, and the error-related negativity." In: *Psychological review* 109.4 (2002), p. 679.

[99] Junhui Hou et al. "Robust Laplacian matrix learning for smooth graph signals". In: *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2016, pp. 1878–1882.

[100] Shelley R Hough et al. "Differentiation of mouse embryonic stem cells after RNA interference-mediated silencing of OCT4 and Nanog". In: *Stem cells* 24.6 (2006), pp. 1467–1475.

[101] Volker Hovestadt et al. "Resolving medulloblastoma cellular architecture by single-cell genomics". In: *Nature* 572.7767 (2019), pp. 74–79.

[102] Cho-Jui Hsieh et al. "Sparse inverse covariance matrix estimation using quadratic approximation". In: *Advances in neural information processing systems* 24 (2011).

[103] Vân Anh Huynh-Thu et al. "Inferring regulatory networks from expression data using tree-based methods". In: *PloS one* 5.9 (2010), pp. 1–10.

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

[105] Bochao Jia et al. “Learning gene regulatory networks from next generation sequencing data”. In: *Biometrics* 73.4 (2017), pp. 1221–1230.

[106] Sai Kiran Kadambari and Sundeep Prabhakar Chepuri. “Learning product graphs from multidomain signals”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 5665–5669.

[107] Vassilis Kalofolias. “How to learn a graph from smooth signals”. In: *Artificial Intelligence and Statistics*. PMLR. 2016, pp. 920–929.

[108] Vassilis Kalofolias and Nathanaël Perraudin. “Large Scale Graph Learning From Smooth Signals”. In: *International Conference on Learning Representations*. 2018.

[109] Vassilis Kalofolias and Nathanaël Perraudin. “Large scale graph learning from smooth signals”. In: *arXiv preprint arXiv:1710.05654* (2017).

[110] Vassilis Kalofolias et al. “Learning time varying graphs”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Ieee. 2017, pp. 2826–2830.

[111] Jiun-Yu Kao et al. “Disc-glasso: Discriminative graph learning with sparsity regularization”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 2956–2960.

[112] Abdullah Karaaslanli et al. “scSGL: kernelized signed graph learning for single-cell gene regulatory network inference”. In: *Bioinformatics* 38.11 (2022), pp. 3011–3019.

[113] Brian Karrer and Mark EJ Newman. “Stochastic blockmodels and community structure in networks”. In: *Physical review E* 83.1 (2011), p. 016107.

[114] Peter V Kharchenko, Lev Silberstein, and David T Scadden. “Bayesian approach to single-cell differential expression analysis”. In: *Nature methods* 11.7 (2014), pp. 740–742.

[115] Bomin Kim et al. “A review of dynamic network models with latent variables”. In: *Statistics surveys* 12 (2018), p. 105.

[116] Seongho Kim. “ppcor: an R package for a fast calculation to semi-partial correlation coefficients”. In: *Communications for statistical applications and methods* 22.6 (2015), p. 665.

[117] Mikko Kivelä et al. “Multilayer networks”. In: *Journal of complex networks* 2.3 (2014), pp. 203–271.

[118] Allon M Klein et al. “Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells”. In: *Cell* 161.5 (2015), pp. 1187–1201.

[119] Jérôme Kunegis et al. “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.

[120] Jean-Philippe Lachaux et al. “Measuring phase synchrony in brain signals”. In: *Human brain mapping* 8.4 (1999), pp. 194–208.

[121] Andrea Lancichinetti and Santo Fortunato. “Consensus Clustering in Complex Networks”. In: *Scientific Reports* 2.1 (Mar. 2012), p. 336. ISSN: 2045-2322. DOI: 10.1038/srep00336.

[122] Wonyul Lee and Yufeng Liu. “Joint estimation of multiple precision matrices with common structures”. In: *The Journal of Machine Learning Research* 16.1 (2015), pp. 1035–1062.

[123] Alexander Lex et al. “UpSet: visualization of intersecting sets”. In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 1983–1992.

[124] Hui-Jia Li et al. “Community structure detection based on Potts model and network’s spectral characterization”. In: *Europhysics Letters* 97.4 (2012), p. 48005.

[125] Yu-Ru Lin et al. “Facetnet: a framework for analyzing communities and their evolutions in dynamic networks”. In: *Proceedings of the 17th international conference on World Wide Web*. 2008, pp. 685–694.

[126] Fuchen Liu et al. “Global spectral clustering in dynamic networks”. In: *Proceedings of the National Academy of Sciences* 115.5 (2018), pp. 927–932.

[127] Han Liu, John Lafferty, and Larry Wasserman. “The nonparanormal: Semiparametric estimation of high dimensional undirected graphs.” In: *Journal of Machine Learning Research* 10.10 (2009).

[128] Han Liu et al. “High-dimensional semiparametric Gaussian copula graphical models”. In: *The Annals of Statistics* 40.4 (2012), pp. 2293–2326.

[129] Yen-Chun Liu et al. “Global regulation of nucleotide biosynthetic genes by c-Myc”. In: *PloS one* 3.7 (2008), e2722.

[130] Zhi-Ping Liu et al. “RegNetwork: an integrated database of transcriptional and post-transcriptional regulatory networks in human and mouse”. In: *Database* 2015 (2015).

[131] Yining Lu et al. “OTX2 expression contributes to proliferation and progression in Myc-amplified medulloblastoma”. In: *American journal of cancer research* 7.3 (2017), p. 647.

[132] Xiaoke Ma and Di Dong. “Evolutionary nonnegative matrix factorization algorithms for community detection in dynamic networks”. In: *IEEE transactions on knowledge and data engineering* 29.5 (2017), pp. 1045–1058.

[133] Matteo Magnani et al. “Community detection in multiplex networks”. In: *ACM Computing Surveys (CSUR)* 54.3 (2021), pp. 1–35.

[134] Daniel Marbach et al. “Generating realistic in silico gene networks for performance assessment of reverse engineering methods”. In: *Journal of computational biology* 16.2 (2009), pp. 229–239.

[135] Daniel Marbach et al. “Wisdom of crowds for robust gene network inference”. In: *Nature methods* 9.8 (2012), pp. 796–804.

[136] Hermina Petric Maretic and Pascal Frossard. “Graph Laplacian mixture model”. In: *IEEE Transactions on Signal and Information Processing over Networks* 6 (2020), pp. 261–270.

[137] Gonzalo Mateos et al. “Connecting the dots: Identifying network structure via graph signal processing”. In: *IEEE Signal Processing Magazine* 36.3 (2019), pp. 16–43.

[138] Catherine Matias and Vincent Miele. “Statistical clustering of temporal networks through a dynamic stochastic block model”. In: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 79.4 (2017), pp. 1119–1141.

[139] Hirotaka Matsumoto et al. “SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation”. In: *Bioinformatics* 33.15 (2017), pp. 2314–2321.

[140] Marcelo G Mattar, Richard F Betzel, and Danielle S Bassett. “The flexible brain”. In: *Brain* 139.8 (2016), pp. 2110–2112.

[141] Gerald Matz and Thomas Dittrich. “Learning signed graphs from data”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 5570–5574.

[142] Rahul Mazumder and Trevor Hastie. “The graphical lasso: New insights and alternatives”. In: *Electronic journal of statistics* 6 (2012), p. 2125.

[143] David Meunier et al. “Hierarchical modularity in human brain functional networks”. In: *Frontiers in neuroinformatics* 3 (2009), p. 37.

[144] *MIT Academic Calendar 2004-2005*. https://web.archive.org/web/20051104091633/http://web.mit.edu/registrar/www/calendar0405.html. Accessed: 2020-12-21.

[145] Kaoru Mitsui et al. “The homeoprotein Nanog is required for maintenance of pluripotency in mouse epiblast and ES cells”. In: *Cell* 113.5 (2003), pp. 631–642.

[146] Thomas Moerman et al. “GRNBoost2 and Arboreto: efficient and scalable inference of gene regulatory networks”. In: *Bioinformatics* 35.12 (2019), pp. 2159–2161.

[147] Victoria Moignard et al. “Decoding the regulatory network of early blood development from single-cell gene expression measurements”. In: *Nature biotechnology* 33.3 (2015), pp. 269–276.

[148] Aanchal Mongia, Debarka Sengupta, and Angshul Majumdar. “McImpute: matrix completion based imputation for single cell RNA-seq data”. In: *Frontiers in genetics* 10 (2019), p. 9.

[149] James Moody and Douglas R White. “Structural cohesion and embeddedness: A hierarchical concept of social groups”. In: *American sociological review* (2003), pp. 103–127.

[150] Tim P Moran, Danielle Taylor, and Jason S Moser. "Sex moderates the relationship between worry and performance monitoring brain activity in undergraduates". In: *International Journal of Psychophysiology* 85.2 (2012), pp. 188–194.

[151] Jan S Moreb et al. "RNAi-mediated knockdown of aldehyde dehydrogenase class-1A1 and class-3A1 is specific and reveals that each contributes equally to the resistance against 4-hydroperoxycyclophosphamide". In: *Cancer chemotherapy and pharmacology* 59.1 (2007), pp. 127–136.

[152] Naomi Moris, Cristina Pina, and Alfonso Martinez Arias. "Transition states and cell fate decisions in epigenetic landscapes". In: *Nature Reviews Genetics* 17.11 (2016), pp. 693–703.

[153] Peter J Mucha et al. "Community structure in time-dependent, multiscale, and multiplex networks". In: *science* 328.5980 (2010), pp. 876–878.

[154] Sumit Mukherjee et al. "Identifying progressive gene network perturbation from single-cell RNA-seq data". In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2018, pp. 5034–5040.

[155] Sarah Feldt Muldoon and Danielle S Bassett. "Network and multilayer network approaches to understanding human brain dynamics". In: *Philosophy of Science* 83.5 (2016), pp. 710–720.

[156] Sarah Feldt Muldoon, Eric W Bridgeford, and Danielle S Bassett. "Small-world propensity and weighted brain networks". In: *Scientific reports* 6 (2016), p. 22057.

[157] T. T. K. Munia and S. Aviyente. "Time-frequency Based phase-Amplitude coupling Measure for neuronal oscillations". In: *Scientific reports* 9.1 (2019), pp. 1–15.

[158] Tamanna Tabassum Khan Munia and Selin Aviyente. "Comparison of Wavelet and RID-Rihaczek Based Methods for Phase-Amplitude Coupling". In: *IEEE Signal Processing Letters* 26.12 (2019), pp. 1897–1901.

[159] Tamanna TK Munia and Selin Aviyente. "Multivariate analysis of bivariate phase-amplitude coupling in EEG data using tensor robust PCA". in: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 29 (2021), pp. 1268–1279.

[160] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[161] Seth Myers and Jure Leskovec. "On the convexity of latent social network inference". In: *Advances in neural information processing systems* 23 (2010).

[162] Antonino Naro et al. "Multiplex and multilayer network EEG analyses: a novel strategy in the differential diagnosis of patients with chronic disorders of consciousness". In: *International Journal of Neural Systems* 31.02 (2021), p. 2050052.

[163] Madeline Navarro et al. "Joint Inference of Multiple Graphs from Matrix Polynomials". In: *Journal of Machine Learning Research* 23.76 (2022), pp. 1–35.

[164] Madeline Navarro et al. “Joint inference of multiple graphs from matrix polynomials”. In: *arXiv preprint arXiv:2010.08120* (2020).

[165] The Cancer Genome Atlas Network. “Comprehensive molecular portraits of human breast tumours”. In: *Nature* 490.7418 (2012), pp. 61–70.

[166] Mark Newman. *Networks*. Oxford university press, 2018.

[167] Mark EJ Newman. “Equivalence between modularity optimization and maximum likelihood methods for community detection”. In: *Physical Review E* 94.5 (2016), p. 052315.

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

[169] Mark EJ Newman. “Spectral methods for community detection and graph partitioning”. In: *Physical Review E* 88.4 (2013), p. 042822.

[170] Mark EJ Newman and Aaron Clauset. “Structure and inference in annotated networks”. In: *Nature communications* 7.1 (2016), p. 11863.

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

[172] Mark EJ Newman and Duncan J Watts. “Renormalization group analysis of the small-world network model”. In: *Physics Letters A* 263.4-6 (1999), pp. 341–346.

[173] Kathy K Niakan et al. “Sox17 promotes differentiation in mouse embryonic stem cells by directly regulating extraembryonic gene expression and indirectly antagonizing self-renewal”. In: *Genes & development* 24.3 (2010), pp. 312–326.

[174] Roland Nigbur et al. “Theta dynamics reveal domain-specific control over stimulus and response conflict”. In: *Journal of Cognitive Neuroscience* 24.5 (2012), pp. 1264–1274.

[175] Huazhong Ning et al. “Incremental spectral clustering by efficiently updating the eigen-system”. In: *Pattern Recognition* 43.1 (2010), pp. 113–127.

[176] Paul A Northcott et al. “Medulloblastoma”. In: *Nature reviews Disease primers* 5.1 (2019), pp. 1–20.

[177] Paul A Northcott et al. “Medulloblastoma comprises four distinct molecular variants”. In: *Journal of clinical oncology* 29.11 (2011), p. 1408.

[178] Antonio Ortega et al. “Graph signal processing: Overview, challenges, and applications”. In: *Proceedings of the IEEE* 106.5 (2018), pp. 808–828.

[179] Alp Ozdemir et al. “Hierarchical spectral consensus clustering for group analysis of functional brain networks”. In: *IEEE Transactions on Biomedical Engineering* 62.9 (2015), pp. 2158–2169.

[180] Tolga Esat Özkurt and Alfons Schnitzler. "A critical note on the definition of phase–amplitude cross-frequency coupling". In: *Journal of Neuroscience methods* 201.2 (2011), pp. 438–443.

[181] A Roxana Pamfil et al. "Relating modularity maximization and stochastic block models in multilayer networks". In: *SIAM Journal on Mathematics of Data Science* 1.4 (2019), pp. 667–698.

[182] Bastien Pasdeloup et al. "Characterization and inference of graph diffusion processes from observations of stationary signals". In: *IEEE transactions on Signal and Information Processing over Networks* 4.3 (2017), pp. 481–496.

[183] Lucrezia Patruno et al. "A review of computational strategies for denoising and imputation of single-cell transcriptomic data". In: *Briefings in Bioinformatics* 22.4 (2021), bbaa222.

[184] Tiago P Peixoto. "Efficient Monte Carlo and greedy heuristic for the inference of stochastic block models". In: *Physical Review E* 89.1 (2014), p. 012804.

[185] Tiago P Peixoto. "Parsimonious module inference in large networks". In: *Physical review letters* 110.14 (2013), p. 148701.

[186] Raphael Petegrosso, Zhuliu Li, and Rui Kuang. "Machine learning and statistical methods for clustering single-cell RNA-sequencing data". In: *Briefings in bioinformatics* 21.4 (2020), pp. 1209–1223.

[187] Emma Pierson and Christopher Yau. "ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis". In: *Genome biology* 16.1 (2015), pp. 1–10.

[188] Ronald S Pimentel, Magdalena Niewiadomska-Bugaj, and Jung-Chao Wang. "Association of zero-inflated continuous variables". In: *Statistics & Probability Letters* 96 (2015), pp. 61–67.

[189] Soumajit Pramanik et al. "Discovering community structure in multilayer networks". In: *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE. 2017, pp. 611–620.

[190] Aditya Pratapa et al. "Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data". In: *Nature methods* 17.2 (2020), pp. 147–154.

[191] Carey E Priebe et al. "Scan statistics on enron graphs". In: *Computational & Mathematical Organization Theory* 11 (2005), pp. 229–247.

[192] Liudmila Prokhorenkova and Alexey Tikhonov. "Community detection through likelihood optimization: in search of a sound model". In: *The World Wide Web Conference*. 2019, pp. 1498–1508.

[193] Laralynne M Przybyla and Joel Voldman. "Attenuation of extrinsic signaling reveals the importance of matrix remodeling on maintenance of embryonic stem cell self-renewal". In: *Proceedings of the National Academy of Sciences* 109.3 (2012), pp. 835–840.

[194] Maria Grazia Puxeddu, Manuela Petti, and Laura Astolfi. "A comprehensive analysis of multilayer community detection algorithms for application to eeg-based brain networks". In: *Frontiers in systems neuroscience* 15 (2021), p. 624183.

[195] Thomas P Quinn et al. "propr: an R-package for identifying proportionally abundant features using compositional data analysis". In: *Scientific reports* 7.1 (2017), pp. 1–9.

[196] Michael G Rabbat, Mark J Coates, and Robert D Nowak. "Multiple-source Internet tomography". In: *IEEE Journal on Selected Areas in Communications* 24.12 (2006), pp. 2221–2234.

[197] P Krishna Reddy et al. "A graph based approach to extract a neighborhood customer community for collaborative filtering". In: *International Workshop on Databases in Networked Information Systems*. Springer. 2002, pp. 188–200.

[198] Jörg Reichardt and Stefan Bornholdt. "Detecting fuzzy community structures in complex networks with a Potts model". In: *Physical review letters* 93.21 (2004), p. 218701.

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

[200] Justin Riddle, Amber McFerren, and Flavio Frohlich. "Causal role of cross-frequency coupling in distinct components of cognitive control". In: *Progress in Neurobiology* 202 (2021), p. 102033.

[201] Davide Risso et al. "A general and flexible method for signal extraction from single-cell RNA-seq data". In: *Nature communications* 9.1 (2018), pp. 1–17.

[202] Ribana Roscher et al. "Explainable machine learning for scientific insights and discoveries". In: *Ieee Access* 8 (2020), pp. 42200–42216.

[203] Giulio Rossetti and Rémy Cazabet. "Community discovery in dynamic networks: a survey". In: *ACM computing surveys (CSUR)* 51.2 (2018), pp. 1–37.

[204] Martin Rosvall and Carl T Bergstrom. "Maps of random walks on complex networks reveal community structure". In: *Proceedings of the national academy of sciences* 105.4 (2008), pp. 1118–1123.

[205] Martine F Roussel and Giles W Robinson. "Role of MYC in Medulloblastoma". In: *Cold Spring Harbor perspectives in medicine* 3.11 (2013), a014308.

[206] Liu Rui et al. "Simultaneous low-rank component and graph estimation for high-dimensional graph signals: Application to brain imaging". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 4134–4138.

[207] Assieh Saadatpour et al. "Characterizing heterogeneity in leukemic cells using single-cell gene expression analysis". In: *Genome biology* 15.12 (2014), pp. 1–13.

[208] Seyed Saman Saboksayr, Gonzalo Mateos, and Mujdat Cetin. "Online discriminative graph learning from multi-class smooth signals". In: *Signal Processing* 186 (2021), p. 108101.

[209] D Franco Saldana, Yi Yu, and Yang Feng. "How many communities are there?" In: *Journal of Computational and Graphical Statistics* 26.1 (2017), pp. 171–181.

[210] Aliaksei Sandryhaila and Jose MF Moura. "Discrete signal processing on graphs: Frequency analysis". In: *IEEE Transactions on Signal Processing* 62.12 (2014), pp. 3042–3054.

[211] Guido Sanguinetti and Vân Anh Huynh-Thu. *Gene regulatory networks*. Springer, 2019.

[212] Stefania Sardellitti, Sergio Barbarossa, and Paolo Di Lorenzo. "Enabling prediction via multi-layer graph inference and sampling". In: *2019 13th International conference on Sampling Theory and Applications (SampTA)*. IEEE. 2019, pp. 1–4.

[213] Purnamrita Sarkar and Andrew W Moore. "Dynamic social network analysis using latent space models". In: *Advances in neural information processing systems* 18 (2006), p. 1145.

[214] Shuntaro Sasai et al. "Frequency-specific network topologies in the resting human brain". In: *Frontiers in human neuroscience* 8 (2014), p. 1022.

[215] Shuntaro Sasai et al. "Frequency-specific task modulation of human brain functional networks: A fast fMRI study". In: *NeuroImage* 224 (2021), p. 117375.

[216] Thomas Schaffter, Daniel Marbach, and Dario Floreano. "GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods". In: *Bioinformatics* 27.16 (2011), pp. 2263–2270.

[217] Holger Scheel and Stefan Scholtes. "Mathematical programs with complementarity constraints: Stationarity, optimality, and sensitivity". In: *Mathematics of Operations Research* 25.1 (2000), pp. 1–22.

[218] William W Seeley et al. "Neurodegenerative diseases target large-scale human brain networks". In: *Neuron* 62.1 (2009), pp. 42–52.

[219] Santiago Segarra et al. "Network topology inference from spectral templates". In: *IEEE Transactions on Signal and Information Processing over Networks* 3.3 (2017), pp. 467–483.

[220] Rasoul Shafipour et al. "Identifying the topology of undirected networks from diffused non-stationary graph signals". In: *IEEE Open Journal of Signal Processing* 2 (2021), pp. 171–189.

[221] Esraa Al-sharoa, Mahmood A Al-khassaweneh, and Selin Aviyente. "Detecting and tracking community structure in temporal networks: A low-rank+ sparse estimation based evolutionary clustering approach". In: *IEEE Transactions on Signal and Information Processing over Networks* 5.4 (2019), pp. 723–738.

[222] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[223] Jonathan Richard Shewchuk. "Allow Me to Introduce Spectral and Isoperimetric Graph Partitioning". In: (Apr. 2016), p. 69.

[224] Hao-Jun Michael Shi et al. “A primer on coordinate descent algorithms”. In: *arXiv preprint arXiv:1610.00040* (2016).

[225] 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.

[226] Wenjing Shi et al. “Regulation of the pluripotency marker Rex-1 by Nanog and Sox2”. In: *Journal of biological chemistry* 281.33 (2006), pp. 23319–23325.

[227] David I Shuman et al. “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains”. In: *IEEE signal processing magazine* 30.3 (2013), pp. 83–98.

[228] Arlei Silva, Ambuj Singh, and Ananthram Swami. “Spectral algorithms for temporal graph cuts”. In: *Proceedings of the 2018 World Wide Web Conference*. 2018, pp. 519–528.

[229] Justin D Silverman et al. “Naught all zeros in sequence count data are the same”. In: *Computational and structural biotechnology journal* 18 (2020), p. 2789.

[230] Michael A Skinnider, Jordan W Squair, and Leonard J Foster. “Evaluating measures of association for single-cell transcriptomics”. In: *Nature methods* 16.5 (2019), pp. 381–386.

[231] Tom AB Snijders and Krzysztof Nowicki. “Estimation and prediction for stochastic blockmodels for graphs with latent block structure”. In: *Journal of classification* 14.1 (1997), pp. 75–100.

[232] Olaf Sporns and Richard F Betzel. “Modular brain networks”. In: *Annual review of psychology* 67 (2016), pp. 613–640.

[233] Oliver Stegle, Sarah A Teichmann, and John C Marioni. “Computational and analytical challenges in single-cell transcriptomics”. In: *Nature Reviews Genetics* 16.3 (2015), pp. 133–145.

[234] Alexander Strehl and Joydeep Ghosh. “Cluster ensembles—a knowledge reuse framework for combining multiple partitions”. In: *Journal of machine learning research* 3.Dec (2002), pp. 583–617.

[235] Tim Stuart et al. “Comprehensive integration of single-cell data”. In: *Cell* 177.7 (2019), pp. 1888–1902.

[236] Valentine Svensson. “Droplet scRNA-seq is not zero-inflated”. In: *Nature Biotechnology* 38.2 (2020), pp. 147–150.

[237] Damian Szklarczyk et al. “The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets”. In: *Nucleic acids research* 49.D1 (2021), pp. D605–D612.

[238] Craig E Tenke and Jürgen Kayser. “Generator localization by current source density (CSD): implications of volume conduction and field closure at intracranial and scalp resolutions”. In: *Clinical neurophysiology* 123.12 (2012), pp. 2328–2345.

[239] Alessandro Tessitore et al. “Default-mode network connectivity in cognitively unimpaired patients with Parkinson disease”. In: *Neurology* 79.23 (2012), pp. 2226–2232.

[240] Prejaas Tewarie et al. “Integrating cross-frequency and within band functional networks in resting-state MEG: a multi-layer network approach”. In: *Neuroimage* 142 (2016), pp. 324–336.

[241] Dorina Thanou et al. “Learning heat diffusion graphs”. In: *IEEE Transactions on Signal and Information Processing over Networks* 3.3 (2017), pp. 484–499.

[242] Adriano BL Tort et al. “Measuring phase-amplitude coupling between neuronal oscillations of different frequencies”. In: *Journal of neurophysiology* 104.2 (2010), pp. 1195–1210.

[243] Damon JA Toth et al. “The role of heterogeneity in contact timing and duration in network models of influenza spread in schools”. In: *Journal of The Royal Society Interface* 12.108 (2015), p. 20150279.

[244] Vincent A Traag, Rodrigo Aldecoa, and J-C Delvenne. “Detecting communities using asymptotical surprise”. In: *Physical review e* 92.2 (2015), p. 022816.

[245] Vincent A Traag, Paul Van Dooren, and Yurii Nesterov. “Narrow scope for resolution-limit-free community detection”. In: *Physical Review E* 84.1 (2011), p. 016114.

[246] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. “From Louvain to Leiden: guaranteeing well-connected communities”. In: *Scientific reports* 9.1 (2019), pp. 1–12.

[247] Michael Vaiana and Sarah Feldt Muldoon. “Multilayer brain networks”. In: *Journal of Nonlinear Science* (2018), pp. 1–23.

[248] Michael Vaiana and Sarah Feldt Muldoon. “Multilayer brain networks”. In: *Journal of Nonlinear Science* 30.5 (2020), pp. 2147–2169.

[249] Ulrike Von Luxburg. “A tutorial on spectral clustering”. In: *Statistics and computing* 17 (2007), pp. 395–416.

[250] Emily M Walker, Cayla A Thompson, and Michele A Battle. “GATA4 and GATA6 regulate intestinal epithelial cytodifferentiation during development”. In: *Developmental biology* 392.2 (2014), pp. 283–294.

[251] Yu Wang, Wotao Yin, and Jinshan Zeng. “Global convergence of ADMM in nonconvex nonsmooth optimization”. In: *Journal of Scientific Computing* 78.1 (2019), pp. 29–63.

[252] Zhaoning Wang et al. “Cell-Type-Specific Gene Regulatory Networks Underlying Murine Neonatal Heart Regeneration at Single-Cell Resolution”. In: *Cell reports* 33.10 (2020), p. 108472.

[253] Alistair J Watt et al. “Development of the mammalian liver and ventral pancreas is dependent on GATA4”. In: *BMC developmental biology* 7.1 (2007), pp. 1–11.

[254] Duncan J Watts and Steven H Strogatz. “Collective dynamics of ‘small-world’networks”. In: *nature* 393.6684 (1998), pp. 440–442.

[255] Nuosi Wu et al. “Joint learning of multiple gene networks from single-cell gene expression data”. In: *Computational and structural biotechnology journal* 18 (2020), pp. 2583–2595.

[256] Zonghan Wu et al. “A comprehensive survey on graph neural networks”. In: *IEEE transactions on neural networks and learning systems* 32.1 (2020), pp. 4–24.

[257] Kevin Xu. “Stochastic block transition models for dynamic networks”. In: *Artificial Intelligence and Statistics*. PMLR. 2015, pp. 1079–1087.

[258] Kevin S Xu and Alfred O Hero. “Dynamic stochastic blockmodels for time-evolving social networks”. In: *IEEE Journal of Selected Topics in Signal Processing* 8.4 (2014), pp. 552–562.

[259] Kevin S Xu, Mark Kliger, and Alfred O Hero III. “Adaptive evolutionary clustering”. In: *Data Mining and Knowledge Discovery* 28 (2014), pp. 304–336.

[260] Yangyang Xu and Wotao Yin. “A Block Coordinate Descent Method for Regularized Multiconvex Optimization with Applications to Nonnegative Tensor Factorization and Completion”. In: *SIAM Journal on Imaging Sciences* 6.3 (Sept. 2013), pp. 1758–1789. ISSN: 1936-4954. DOI: 10.1137/120887795.

[261] Zhigang Xue et al. “Genetic programs in human and mouse early embryos revealed by single-cell RNA sequencing”. In: *Nature* 500.7464 (2013), pp. 593–597.

[262] Inbal Yahav and Galit Shmueli. “On generating multivariate Poisson data in management science applications”. In: *Applied Stochastic Models in Business and Industry* 28.1 (2012), pp. 91–102.

[263] Koki Yamada, Yuichi Tanaka, and Antonio Ortega. “Time-varying graph learning based on sparseness of temporal variation”. In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 5411–5415.

[264] Tianbao Yang et al. “Detecting communities and their evolutions in dynamic social networks—a Bayesian approach”. In: *Machine learning* 82 (2011), pp. 157–189.

[265] Wencheng Yin et al. “Emergence of co-expression in gene regulatory networks”. In: *PloS one* 16.4 (2021), e0247671.

[266] Meichen Yu et al. “Selective impairment of hippocampus and posterior hub areas in Alzheimer’s disease: an MEG-based multiplex network study”. In: *Brain* 140.5 (2017), pp. 1466–1485.

[267] R. Zass and A. Shashua. “A Unifying Approach to Hard and Probabilistic Clustering”. In: *Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1*. Beijing, China: IEEE, Oct. 2005, 294–301 Vol. 1. ISBN: 978-0-7695-2334-7. DOI: 10.1109/ICCV.2005.27.

[268] Xiao Zhang, Cristopher Moore, and Mark EJ Newman. “Random graph models for dynamic networks”. In: *The European Physical Journal B* 90 (2017), pp. 1–14.

[269] Ziwei Zhang, Peng Cui, and Wenwu Zhu. “Deep learning on graphs: A survey”. In: *IEEE Transactions on Knowledge and Data Engineering* (2020).

[270] Qing Zhou et al. “A gene regulatory network in mouse embryonic stem cells”. In: *Proceedings of the National Academy of Sciences* 104.42 (2007), pp. 16438–16443.

[271] Hongliang Zou and Jian Yang. “Multi-frequency dynamic weighted functional connectivity networks for schizophrenia diagnosis”. In: *Applied Magnetic Resonance* 50.7 (2019), pp. 847–859.