Title
Bipartite Network Community Detection: Development and Survey of Algorithmic and Stochastic Block Model Based Methods

Author
Sun, Yidan

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Bipartite Network Community Detection: Development and Survey of Algorithmic and Stochastic Block Model Based Methods

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Yidan Sun

2021

ABSTRACT OF THE DISSERTATION

Bipartite Network Community Detection: Development and Survey of Algorithmic and Stochastic Block Model Based Methods

by

Yidan Sun
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2021
Professor Jingyi Li, Chair

In a bipartite network, nodes are divided into two types, and edges are only allowed to connect nodes of different types. Bipartite network clustering problems aim to identify node groups with more edges between themselves and fewer edges to the rest of the network. The approaches for community detection in the bipartite network can roughly be classified into algorithmic and model-based methods. The algorithmic methods solve the problem either by greedy searches in a heuristic way or optimizing based on some criteria over all possible partitions. The model-based methods fit a generative model to the observed data and study the model in a statistically principled way. In this dissertation, we mainly focus on bipartite clustering under two scenarios: incorporation of node covariates and detection of mixed membership communities.

In Chapter 2, we study an algorithmic bipartite clustering method in the context of gene co-clustering. Gene co-clustering is a widely-used technique that has enabled computational prediction of unknown gene functions within a species. However, it remains a challenge to refine gene function prediction by leveraging evolutionarily conserved genes in another species. This challenge calls for a new computational algorithm to identify gene co-clusters in two species, so that genes in each co-cluster exhibit similar expression levels in each species and strong conservation between the species. Here we develop the bipartite tight

spectral clustering (BiTSC) algorithm, which identifies gene co-clusters in two species based on gene orthology information and gene expression data. BiTSC novelly implements a formulation that encodes gene orthology as a bipartite network and gene expression data as node covariates. This formulation allows BiTSC to adopt and combine the advantages of multiple unsupervised learning techniques: kernel enhancement, bipartite spectral clustering, consensus clustering, tight clustering, and hierarchical clustering. As a result, BiTSC is a flexible and robust algorithm capable of identifying informative gene co-clusters without forcing all genes into co-clusters. Another advantage of BiTSC is that it does not rely on any distributional assumptions. Beyond cross-species gene co-clustering, BiTSC also has wide applications as a general algorithm for identifying tight node co-clusters in any bipartite network with node covariates. We demonstrate the accuracy and robustness of BiTSC through comprehensive simulation studies. In a real data example, we use BiTSC to identify conserved gene co-clusters of *D. melanogaster* and *C. elegans*, and we perform a series of downstream analyses to both validate BiTSC and verify the biological significance of the identified co-clusters.

The stochastic block model (SBM) serves as a fundamental and classic tool to study model-based community detection in a statistically principled way. It is a simple generative model but too restrictive to include empirical network characteristics. With the flexibility of the SBM, many SBM variants have been proposed to accommodate the real-world scenarios, such as node degree heterogeneity, mixed membership communities, and bipartite structure. In Chapter 3, we review a few extensions of the SBM, including degree-correction SBM, mixed membership SBM, and bipartite SBM. We conduct a survey about community detection methods based on the mixed membership SBM and bipartite SBM. We discuss these methods in detail and summarize the challenges in introducing degree-correction to the mixed membership SBM; the challenges in extending the degree-corrected mixed membership SBM to the bipartite network.

To address some of the research gaps discussed in Chapter 3, we construct a generative bipartite degree-corrected mixed membership SBM in Chapter 4. To fit the model, we

propose a variational expectation-maximization (EM) algorithm. We perform a preliminary simulation study in which the result suggests that the algorithm is sensitive to the initial values. We discuss the possible improvements for future directions.

The dissertation of Yidan Sun is approved.

Mark Stephen Handcock

Yingnian Wu

Sriram Sankararaman

Jingyi Li, Committee Chair

University of California, Los Angeles

2021

To my mom and dad.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank my advisor Prof. Jingyi Jessica Li for the continuous support, help, patience, encouragement, and career guidance throughout my graduate study. I owe the successful completion of my thesis to her. I would not be where I am today without her help.

I would like to thank my committee members, Prof. Mark Handcock, Prof. Yingnian Wu and Prof. Sriram Sankararaman, for their insightful suggestions and discussions on my dissertation.

Special thanks to Prof. Mark Handcock, Prof. Nicolas Christou and Prof. Xin Tong, for their help in career guidance.

I would like to thank my co-author Heather Zhou for the generous help. I would like to thank my friends, Kexin Li, Wei Li, Wenbin Guo, Ruochen Jiang, Dongyuan Song, Nan Xi, Guan'ao Yan, Yucheng Yang, Kun Zhou, Zhixin Zhou, and all the other members in the Junction of Statistics and Biology (http://jsb.ucla.edu), for their continuous support and encouragement.

VITA

2011 - 2015	B.S. in Mathematical Finance, Wuhan University
2015 - 2021	Ph.D. student in Department of Statistics, University of California, Los Angeles

PUBLICATIONS

Sun, Yidan Eden, Heather J Zhou, and Jingyi Jessica Li. Bipartite Tight Spectral Clustering (BiTSC) Algorithm for Identifying Conserved Gene Co-clusters in Two Species. *Bioinformatics*, 08 2020, btaa741.

CHAPTER 1

Introduction

1.1 Background and Motivation

A network is a set of nodes with edges connected between them. Mathematically, a network consisting of N nodes can be represented by an $N \times N$ symmetric adjacency matrix, where each entry indicates the corresponding edge weight. An edge weight is a similarity measure of the two nodes the edge connects and may take any positive continuous value; the higher the value, the more similar the two nodes. One of the major and fundamental challenges in statistical network analysis is to divide nodes into meaningful groups. Community detection, as an unsupervised technique, aims to solve this problem. This task has been actively studied in several fields, including statistics, biology, social science, computer science, and physics. In general, community detection aims to partition nodes into communities so that nodes within a community have denser edge connections with each other than with nodes outside the community. By definition, a community usually refers to a group of nodes that behave homogeneously [1]. Using the language of the adjacency matrix, a community refers to a group of nodes of high similarity. Note that community detection is related to the concept of graph partitioning (such as RatioCut [2] and Ncut [3]) from the computer science field. The latter one typically aims to find a minimum cut between partitions.

Community detection has wide real-world applications across a range of disciplines, such as gene regulation network [4–6], protein interaction network [7–9], social friendship network [10, 11], etc. The main motivation for developing new clustering models/methods is driven by the emerging new features of the real-world networks. Due to the complexity and existence of noise in the network data sets, no method is ideal enough to accurately and robustly capture

the structures presented among networks of various sizes and heterogeneous features (see [12] for reviews). Over the last two decades, plenty of methods have been developed with different emphases on computational complexity, empirical application, statistical optimality, and theoretical quality. These methods basically can be considered as two categories, algorithm-based approaches, and model/likelihood-based approaches. The algorithm-based approaches include the heuristic methods such as hierarchical clustering [13] and DBSCAN [14], as well as the methods based on optimizing a global measurement over all possible partitions such as spectral clustering [3, 15, 16] and modularity maximization [17]. The model-based approaches focus on fitting a probabilistic model for community structure within a network, including convex relaxations via semidefinite programs (SDPs) [18, 19]; profile likelihood [20]; MCMC [21] and variational Bayes [22, 23]; pseudo-likelihood maximization [24]. Among these methods, SDPs have the limitation of not being able to extend to the disassortative structure, such as the bipartite structure. Spectral clustering, on the other hand, has the flexibility of accommodating real-world scenarios such as the degree heterogeneity [25, 26], sparsity [24, 27], and bipartite structure [28]. Besides, spectral clustering itself is often used as the initialization method for EM algorithm [24, 29].

The increasing need to understand the features and properties of network structures has been propelled by the availability of extensive real-world network data sets in various areas. A natural network structure called being bipartite arises when there are two types of nodes in a network. In a bipartite network, one divides N nodes into N_1 type-1 nodes and N_2 type-2 nodes, and edges exist only between nodes of different types. Many networks present the form of bipartite structure, for instance, a network of drugs and proteins, where drugs are connected to proteins that they target; a network of papers and authors, where people are connected to papers they co-authored in; a network of movies and actors where actors are connected to movies they play roles in. Similar to the general network case, the definition of a community in a bipartite network is a group of type-1 nodes with similar connectivity patterns to other groups of type-2 nodes and vice versa. Community detection in the bipartite networks has wide applications and promising prospects in the real-world data sets, especially in biomedical

research [30] and social science fields. For example, we can cluster on a drug-protein bipartite network to study the drug repurposing [31, 32]; we can cluster on a transcription factor (TFs)-genes bipartite network to study gene regulation. In social science, we can cluster on a user-item bipartite network to study the recommendation system [33, 34]. More specifically, the recommender systems are used to predict users' rating/preference on items. The rationale is that similar users are likely to purchase similar items and rate them in similar ways. Hence we can recommend items to a user based on the observed rating behaviors of the group to which this user belongs. In this dissertation, we will use the term "type" and "side" interchangeably; we will use the term "community", "group" and "cluster" interchangeably.

In the field of computational biology, the wide applications of network community detection focus on gene co-expression networks, where a gene is represented by a node and co-expression correlation of two genes is represented by an edge. RNA sequencing (RNA-seq), a technology for quantifying the transcript levels across multiple biological conditions, has been increasingly used to produce large-scale gene expression data sets. Provided by the numerous public RNA-seq datasets, researchers have developed plenty of tools to analyze the gene expression data among multiple species. In particular, one common but important question is how to predict gene functions, as many genes have not been well understood. To answer this question, we can cluster genes into functional groups based on the gene co-expression network. The rationale is as follows. Gene expression levels in various biological samples could be different, and genes exhibiting similar expression patterns across samples are more likely to encode proteins in the same complex; to participate in a common metabolic pathway; to share similar biological functions. Within a species, one can construct a gene co-expression network and group genes with similar expression patterns. Based on the rationale that co-expressed genes are functionally related, we can determine functional enrichment within each group and assign the enriched functions to poorly annotated genes in the same group.

On the other hand, given gene expression data of two species, i.e., human and mouse, we want to utilize information in both species together to refine gene functions of unknown species by studying genes of well-known species. The rationale is based on a principle in

evolutionary biology: all living things on earth evolved from an ancient common ancestor, and the shared information is encoded in the orthologs. By definition, orthologs are genes in different species that are evolved from a common ancestral gene by speciation, and in general, retain a similar function along with the evolution history. Given this, to simultaneously identify functionally conserved gene groups in two species, a clustering method that can integrate both gene expression data and orthologs data is demanded. In Chapter 2, we develop a bipartite spectral clustering algorithm to fill in this gap. In sum, our method is applied to a bipartite network with node covariates, where a gene expression vector is encoded as a node covariate, and an edge is encoded as a pair of orthologs.

As we mentioned above, the algorithm-based methods do not assume a probabilistic distribution model for the data. In the statistics field, the vast amount of methods proposed have driven researchers to compare them in a principled way. Specifically, a generative model which can be served as a ground truth testbed for studying community detection is preferred. The SBM is the best-known generative model with a block structure for studying community detection. In the simplest version of SBM, each node belongs to one of the K groups, and undirected edges are generated independently with probabilities that only depend on the group memberships of the nodes. This model enjoys the desirable property of asymptotic consistency under certain conditions. While the simple SBM often fails to capture the complex structures in the real-world networks, many variants of the SBM have been proposed to incorporate these aspects, including broad node degree distributions [35] (hub nodes with many more connections than other nodes in their community); weighted edges [36]; outliers [37]; nodes with covariates [38, 39]; mixed membership communities (nodes that belong to more than one community) [40, 41]; disassortative structures (i.e., bipartite networks [42]); hierarchical community structures [43]; multilayer structures [44].

The degree-corrected stochastic block model (DCSBM) [35] is the most popular generative model to accommodate the node degree heterogeneity in the SBM, where the probability of an edge between nodes i and j is multiplied by the product of node-specific degree parameters $\theta_i\theta_j$. While original paper [35] fits the model by maximizing the likelihood, DCSBM has also been

studied under various model settings such as pseudo-likelihood, modularity maximization, and spectral clustering [24, 25, 27, 45–47]. Although the majority of existing approaches for community detection focus on nonoverlapping or disjoint communities, it is common for communities to overlap in empirical situations. There has been a growing interest in studying the mixed membership community detection problem in the last decade. For example, the mixed membership stochastic block model (MMSBM) [48] is a well-studied extension of the SBM, allowing for overlap. The main idea is to assign each node by a probability mass function $\boldsymbol{\pi}_i = (\pi_{i1}, \pi_{i2}, \ldots, \pi_{iK})^\mathsf{T}$ where each element is the probability node i belongs to a community k. In principle, in mixed membership community models, all entries of a membership vector can be continuous and positive, indicating its weight of belonging to different communities [49–56]. Besides, researchers have also attempted to incorporate degree-correction into the mixed membership communities, see examples in [57–59]. In Chapter 3, we summarize the existing approaches for detecting mixed membership communities and discuss the limitations of extending these methods to the degree-correction case.

Built on DCSBM, Larremore et al. [60] extend the model to bipartite case and propose the bipartite stochastic block model (biSBM). Intuitively, the existing clustering methods for the general SBM framework can be directly applied to the bipartite network, as the bipartite SBM is essentially a special case of the SBM with zero diagonal blocks. However, previous research [60] has shown that prespecifying the node type information is nontrivial and improves both the quality of clustering and computational time.

Compared to the unipartite networks, studying the mixed membership community detection under the bipartite scheme was given less attention. With statistical inference methods becoming increasingly sophisticated, it is possible to combine degree-correction, mixed membership in one bipartite community detection problem. As far as we know, no research has studied the degree-corrected mixed membership community detection based on bipartite stochastic block models. Therefore, clustering methods designed for filling in this gap are demanded.

1.2 Outline

In summary, this dissertation is organized as follows:

- In Chapter 2, we develop BiTSC, a flexible and robust network clustering algorithm capable of identifying conserved gene co-clusters in two species based on gene orthology information and gene expression data. We demonstrate the accuracy and robustness of BiTSC through comprehensive simulation studies. In a real data example, we apply BiTSC to identify conserved gene co-clusters of *D. melanogaster* and *C. elegans*, and we perform a series of downstream analyses to validate BiTSC and verify the biological significance of the identified co-clusters.

- In Chapter 3, we start with a detailed literature review about current exiting community detection methods for (degree-corrected) mixed membership SBM and bipartite SBM. We compare the properties of each method from multiple perspectives. We then analyze the current research gaps in these methods.

- In Chapter 4, we fill in the gap discussed in Chapter 3 by proposing a broadly applicable bipartite mixed membership SBM with degree correction. After introducing the model, we also present a variational EM algorithm that fits the model to data. We implement the algorithm and obtain some preliminary simulation results.

- In Chapter 5, we conclude this dissertation with discussion and future work.

CHAPTER 2

Bipartite Tight Spectral Clustering (BiTSC) Algorithm

2.1 Introduction

In computational biology, a long-standing problem is how to predict functions of the majority of genes that have not been well understood. This prediction task requires borrowing functional information from other genes with similar expression patterns in the same species or orthologous genes in other species. Within a species, how to identify genes with similar expression patterns across multiple conditions is a clustering problem, and researchers have successfully employed clustering methods to infer unknown gene functions [61, 62]. Specifically, functions of less well-understood genes are inferred from known functions of other genes in the same cluster. The rationale is that genes in one cluster are likely to encode proteins in the same complex or participate in a common metabolic pathway and thus share similar biological functions [63]. In the last two decades, gene clustering for functional prediction has been empowered by the availability of abundant microarray and RNA-seq data [64–68]. Cross-species analysis is another approach to infer gene functions by borrowing functional information of orthologous genes in other species, under the assumption that orthologous genes are likely to share similar functions [67, 69–73]. Although computational prediction of orthologous genes remains an ongoing challenge, gene orthology information with increasing accuracy is readily available in public databases such as TreeFam [74] and PANTHER [75]. Hence, it is reasonable to combine gene expression data with gene orthology information to increase the accuracy of predicting unknown gene functions.

Given two species, the computational task is to identify conserved gene co-clusters containing genes from both species. The goal is to make each co-cluster enriched with orthologous gene pairs and ensure that its genes exhibit similar expression patterns in each species. Among existing methods for this task, the earlier methods [76–78] took a two-step approach: in step 1, genes are clustered in each species based on gene expression data; in step 2, the gene clusters from the two species are paired into co-clusters based on gene orthology information. This two-step approach has a major drawback: there is no guarantee that gene clusters found in step 1 can be paired into meaningful co-clusters in step 2. The reason is that step 1 performs separate gene clustering in the two species without accounting for gene orthology, and as a result, any two gene clusters from different species may share few orthologs and should not be paired into a co-cluster. More recent methods abandoned this two-step approach. For example, SCSC [79] took a model-based approach, and MVBC [80] took a joint matrix factorization approach. Both SCSC and MVBC require that genes in two species are in one-to-one ortholog pairs. This notable limitation prevents SCSC and MVBC from considering the majority of genes that do not have known orthologs or have more than one orthologs in the other species. Furthermore, SCSC assumes that each orthologous gene pair has expression levels generated from a Gaussian mixture model and the gene expression levels are independent between the two species. This strong distributional assumption does not hold for gene expression data from RNA-seq experiments. MVBC is also limited by its required input of verified gene expression patterns, which are often unavailable for many gene expression datasets. OrthoClust [81] is a network-based gene co-clustering method that constructs a unipartite gene network with nodes as genes in two species. Edges are established based on gene co-expression relationships to connect genes of the same species, or gene orthology relationships to connect genes from different species. OrthoClust identifies gene clusters from this network using a modularity maximization approach, which cannot guarantee that each identified cluster contains genes from both species. There are also two open questions regarding the use of OrthoClust in practice: (1) how to define within-species edges based on gene co-expression and (2) how to balance the relative weights of within-species edges and between-species edges in clustering. Another class of methods is biological network

alignment [8, 9, 82, 83], whose aim is to find conserved node and edge mapping between networks of different species. These methods have been mostly applied to protein-protein interaction networks. If applied to gene co-clustering, they would have the same requirement as OrthoClust has for pre-computed within-species gene networks, whose construction from gene expression data, however, has no gold standards.

Here we propose bipartite tight spectral clustering (BiTSC), a novel cross-species gene co-clustering algorithm, to overcome the above-mentioned disadvantages of the existing methods. BiTSC for the first time implements a bipartite-network formulation to tackle the computational task: it encodes gene orthology as a bipartite network and gene expression data as node covariates. This formulation was first mentioned in [84] but not implemented. BiTSC implements this formulation to simultaneously leverage gene orthology and gene expression data to identify tight gene co-clusters, each of which contains similar gene expression patterns in each species and rich gene ortholog pairs between species. Existing bipartite network clustering methods, which were developed for general bipartite networks, are not well suited for this task. Some of them cannot account for node covariate information [28, 60, 85], while others have strong distributional assumptions that do not hold for gene orthology networks and gene expression data measured by RNA-seq [84, 86]. In contrast, BiTSC adopts and combines the advantages of multiple unsupervised learning techniques, including kernel enhancement [87], bipartite spectral clustering [28], consensus clustering [88], tight clustering [89], and hierarchical clustering [90]. As a result, BiTSC has three main advantages. First, BiTSC is the first gene co-clustering method that does not force every gene into a co-cluster; in other words, it only identifies tight gene co-clusters and allows for unclustered genes. This is advantageous because some genes have individualized functions [91–93] and thus should not be assigned into any co-cluster. BiTSC is also flexible in allowing users to adjust the tightness of its identified gene co-clusters. Second, BiTSC is able to consider all the genes in two species, including those genes that do not have orthologs in the other species. Third, BiTSC takes an algorithmic approach that does not rely on any distributional assumptions, making it a robust method. Moreover, we want to emphasize that BiTSC is not only a

bioinformatics method but also a general algorithm for network analysis. It can be used to identify tight node co-clusters in a bipartite network with node covariates.

2.2 Methods

2.2.1 Bipartite network formulation

BiTSC formulates the cross-species gene co-clustering problem as a community detection problem in a bipartite network with node covariates. A bipartite network contains two sides of nodes, and edges only exist between nodes on different sides, not between nodes on the same side. Each node is associated with a covariate vector, also known as node attributes. In bipartite network analysis, the community detection task is to divide nodes into co-clusters based on edges and node covariates, so that nodes in one co-cluster have dense edge connections and similar node covariates on each side [84]. In its formulation, BiTSC encodes genes of two species as nodes of two sides in a bipartite network, where an edge indicates that the two genes it connects are orthologous; BiTSC encodes each gene's expression levels as its node covariates, with the requirement that all genes in one species have expression measurements in the same set of biological samples. For the rest of the Methods section, the terms "nodes" and "genes" are used interchangeably, so are "sides" and "species", as well as "node covariates" and "gene expression levels."

In mathematical notations, there are m and n nodes on side 1 and 2, respectively. Edges are represented by a binary bi-adjacency matrix $\mathbf{A} = (a_{ij})_{m\times n}$, where $a_{ij} = 1$ indicates that there is an edge between node i on side 1 and node j on side 2, i.e., gene i from species 1 and gene j from species 2 are orthologous. Note that $\mathbf{A}$ is allowed to be a weighted bi-adjacency matrix with $a_{ij} \in [0, 1]$ when weighted orthologous relationships are considered. Node covariates are encoded in two matrices, $\mathbf{X}_1$ and $\mathbf{X}_2$, which have dimensions $m \times p_1$ and $n \times p_2$ respectively, i.e., species 1 and 2 have gene expression levels measured in p_1 and p_2 biological samples respectively. The i-th row of $\mathbf{X}_1$ is denoted as $\mathbf{x}_{1i}^T$, and similarly for $\mathbf{X}_2$. Note that all vectors are column vectors unless otherwise stated.

2.2.2 The BiTSC algorithm

BiTSC is a general algorithm that identifies tight node clusters from a bipartite network with node covariates. Table 1 summarizes the input data, input parameters, and output of BiTSC. Figure 2.1 illustrates the idea of BiTSC, and Figure 2.8 shows the detailed workflow. In the context of cross-species gene co-clustering, BiTSC inputs $\mathbf{A}$, which contains gene orthology information, and $\mathbf{X}_1$ and $\mathbf{X}_2$, which denote gene expression data in species 1 and 2. BiTSC outputs tight gene co-clusters such that genes within each co-cluster are rich in orthologs and share similar gene expression levels across multiple biological samples in each species. A unique advantage of BiTSC is that it does not force all genes into co-clusters and allows certain genes with few orthologs or outlying gene expression levels to stay unclustered.

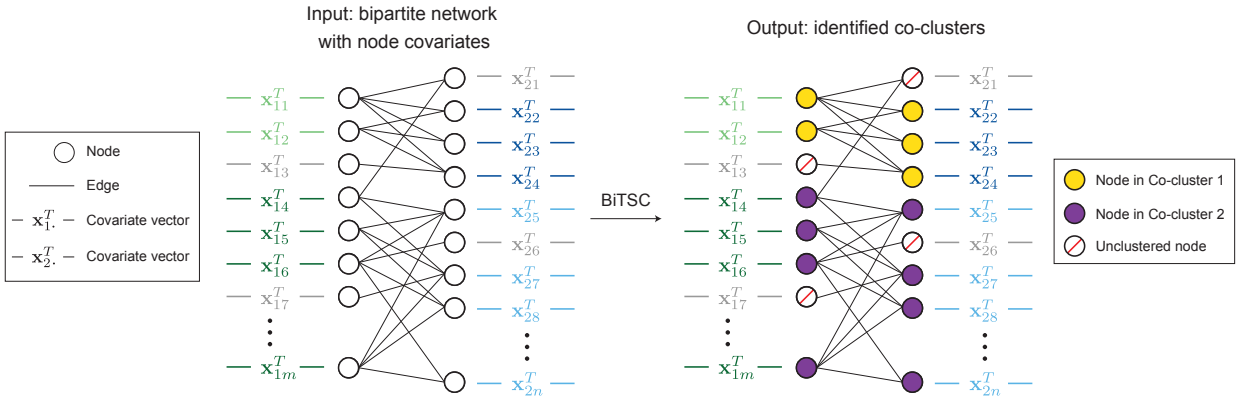

Figure 2.1: Diagram illustrating the input and output of BiTSC. The identified tight node co-clusters satisfy that, within any co-cluster, nodes on the same side share similar covariates, and nodes from different sides are densely connected. In the context of gene co-clustering, within any co-cluster, genes from the same species share similar gene expression levels across multiple conditions, and genes from different species are rich in orthologs.

As an overview, BiTSC is an ensemble algorithm that takes multiple parallel runs. In each run, BiTSC first identifies initial node co-clusters in a randomly subsampled bipartite sub-network; next, it assigns the unsampled nodes to these initial co-clusters based on node covariates. Then BiTSC aggregates the sets of node co-clusters resulted from these multiple

runs into a consensus matrix, from which it identifies tight node co-clusters by hierarchical clustering. This subsampling-and-aggregation idea was inspired by consensus clustering [88] and tight clustering [89].

Table 1 Input and output of BiTSC

Input data: bipartite network with node covariates

- $\mathbf{A}$: $m \times n$ bi-adjacency matrix
- $\mathbf{X}_1$: $m \times p_1$ covariate matrix for side 1
- $\mathbf{X}_2$: $n \times p_2$ covariate matrix for side 2

Input parameters:

- H: number of subsampling runs
- $\rho \in (0, 1)$: proportion of nodes to subsample in each run
- τ: tuning parameter for constructing the enhanced bi-adjacency matrix
- K_0: number of node co-clusters to identify in each run
- $\alpha \in (0, 1)$: tightness parameter

Output:

- Tight node co-clusters that are mutually exclusive and collectively a subset of the $(m + n)$ nodes

BiTSC has five input parameters (Table 1): H, the number of runs; $\rho \in (0, 1)$, the proportion of nodes to subsample in each run; τ, tuning parameter for constructing the enhanced bi-adjacency matrix; K_0, the number of node co-clusters in each run; $\alpha \in (0, 1)$, the tightness parameter used to find tight node co-clusters in the last step. In the h-th run, $h = 1, \ldots, H$, BiTSC has the following four steps.

1. Subsampling. BiTSC randomly samples without replacement $\tilde{m} = \lfloor \rho m \rfloor$ nodes on side 1 and $\tilde{n} = \lfloor \rho n \rfloor$ nodes on side 2, where the floor function $\lfloor x \rfloor$ gives the largest integer

less than or equal to x. We denote the subsampled bi-adjacency matrix as $\tilde{\mathbf{A}}$, whose dimensions are $\tilde{m} \times \tilde{n}$, and the two subsampled covariate matrices as $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$, whose dimensions are $\tilde{m} \times p_1$ and $\tilde{n} \times p_2$ respectively.

2. Kernel enhancement. To find initial node co-clusters from this bipartite sub-network $\tilde{\mathbf{A}}$ with node covariates $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$, a technical issue is that this sub-network may have sparse edges and disconnected nodes. To address this issue, BiTSC employs the kernel enhancement technique proposed by [87] to complement network edges by integrating node covariates. This kernel enhancement step will essentially reweight edges by incorporating pairwise node similarities on both sides. Technically, BiTSC defines two kernel matrices $\tilde{\mathbf{K}}_1$ and $\tilde{\mathbf{K}}_2$, which are symmetric and have dimensions $\tilde{m} \times \tilde{m}$ and $\tilde{n} \times \tilde{n}$, for nodes on side 1 and 2 respectively. In $\tilde{\mathbf{K}}_r$, $r = 1, 2$, the (i, j)-th entry is $k_r(\tilde{\mathbf{x}}_{ri}, \tilde{\mathbf{x}}_{rj}) = \exp(-\|\tilde{\mathbf{x}}_{ri} - \tilde{\mathbf{x}}_{rj}\|^2/p_r)$, where $\|\tilde{\mathbf{x}}_{ri} - \tilde{\mathbf{x}}_{rj}\|$ is the Euclidean distance between nodes i and j on side r in this sub-network. Then BiTSC constructs an enhanced bi-adjacency matrix $\tilde{\mathbf{B}} = (\tilde{\mathbf{K}}_1 + \tau_1 \mathbf{I}_{\tilde{m}})\tilde{\mathbf{A}}(\tilde{\mathbf{K}}_2 + \tau_2 \mathbf{I}_{\tilde{n}})$, whose dimensions are $\tilde{m} \times \tilde{n}$, where $\mathbf{I}_{\tilde{m}}$ and $\mathbf{I}_{\tilde{n}}$ are the $\tilde{m}$- and $\tilde{n}$-dimensional identity matrices, and $\tau = (\tau_1, \tau_2) \in [0, \infty)^2$ is a tuning parameter that balances the information from the subsampled bi-adjacency matrix and the two kernel matrices. Since $\tilde{\mathbf{B}}$ can be rewritten as $\tilde{\mathbf{K}}_1\tilde{\mathbf{A}}\tilde{\mathbf{K}}_2 + \tau_1\tilde{\mathbf{A}}\tilde{\mathbf{K}}_2 + \tau_2\tilde{\mathbf{K}}_1\tilde{\mathbf{A}} + \tau_1\tau_2\tilde{\mathbf{A}}$, when τ_1 and τ_2 are large, $\tau_1\tau_2\tilde{\mathbf{A}}$ dominates and covariate information has little to no impact on $\tilde{\mathbf{B}}$; when τ_1 and τ_2 are both close to 0, $\tilde{\mathbf{K}}_1\tilde{\mathbf{A}}\tilde{\mathbf{K}}_2$ dominates and covariate information contributes more to the enhanced bi-adjacency matrix.

3. Bipartite spectral clustering. BiTSC identifies initial node co-clusters from $\tilde{\mathbf{B}}$, the enhanced bi-adjacency matrix of the bipartite sub-network, by borrowing the idea from [28]. Technically, BiTSC first constructs

$$\tilde{\mathbf{W}} = (\tilde{w}_{ij})_{(\tilde{m}+\tilde{n})\times(\tilde{m}+\tilde{n})} = \begin{bmatrix} \mathbf{0}_{\tilde{m}\times\tilde{m}} & \tilde{\mathbf{B}}_{\tilde{m}\times\tilde{n}} \\ \tilde{\mathbf{B}}^T_{\tilde{n}\times\tilde{m}} & \mathbf{0}_{\tilde{n}\times\tilde{n}} \end{bmatrix},$$

which may be viewed as the adjacency matrix of a unipartite network with $(\tilde{m} + \tilde{n})$

nodes. Then BiTSC identifies K_0 mutually exclusive and collectively exhaustive clusters from $\tilde{\mathbf{W}}$ via normalized spectral clustering [15] as follows:

(a) BiTSC computes a degree matrix $\tilde{\mathbf{D}}$, an $(\tilde{m}+\tilde{n})$-dimensional diagonal matrix whose diagonal entries are the row sums of $\tilde{\mathbf{W}}$.

(b) BiTSC computes the normalized Laplacian of $\tilde{\mathbf{W}}$ as $\tilde{\mathbf{L}} = \mathbf{I}_{\tilde{m}+\tilde{n}} - \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{W}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$. Note that $\tilde{\mathbf{L}}$ is a positive semi-definite $(\tilde{m}+\tilde{n}) \times (\tilde{m}+\tilde{n})$ matrix with $(\tilde{m}+\tilde{n})$ non-negative real-valued eigenvalues: $0 = \lambda_1 \leq \cdots \leq \lambda_{\tilde{m}+\tilde{n}}$.

(c) BiTSC finds the first K_0 eigenvectors of $\tilde{\mathbf{L}}$ that correspond to $\lambda_1, \cdots, \lambda_{K_0}$. Each eigenvector has length $(\tilde{m}+\tilde{n})$. Then BiTSC collects these K_0 eigenvectors column-wise into a matrix $\tilde{\mathbf{U}}$, whose dimensions are $(\tilde{m}+\tilde{n}) \times K_0$.

(d) BiTSC normalizes each row of $\tilde{\mathbf{U}}$ to have a unit ℓ_2 norm and denotes the normalized matrix as $\tilde{\mathbf{V}}$. Specifically, $\tilde{\mathbf{V}}$ also has dimensions $(\tilde{m}+\tilde{n}) \times K_0$, and its i-th row $\tilde{\mathbf{v}}_i^T = \tilde{\mathbf{u}}_i^T / \|\tilde{\mathbf{u}}_i^T\|$, where $\tilde{\mathbf{u}}_i^T$ is the i-th row of $\tilde{\mathbf{U}}$ and $\|\cdot\|$ denotes the ℓ_2 norm.

(e) BiTSC applies K-means clustering to divide the $(\tilde{m}+\tilde{n})$ rows of $\tilde{\mathbf{V}}$ into K_0 clusters. In detail, Euclidean distance is used to measure the distance between each row and each cluster center.

The resulting K_0 clusters of $(\tilde{m}+\tilde{n})$ nodes are regarded as the initial K_0 node co-clusters.

4. Assignment of unsampled nodes. BiTSC assigns the unsampled nodes, which are not subsampled in step 1, into the initial K_0 node co-clusters. Specifically, there are $(m-\tilde{m})$ and $(n-\tilde{n})$ unsampled nodes on side 1 and 2, respectively. For each initial node co-cluster, BiTSC first calculates a mean covariate vector on each side. For example, if a co-cluster contains nodes i and j on side 1 of the bipartite sub-network, its mean covariate vector on side 1 would be computed as $(\tilde{\mathbf{x}}_{1i} + \tilde{\mathbf{x}}_{1j})/2$. BiTSC next assigns each unsampled node to the co-cluster whose mean covariate vector (on the same side as the unsampled node) has the smallest Euclidean distance to the node's covariate vector.

With the above four steps, in the h-th run, $h = 1, \ldots, H$, BiTSC obtains K_0 node co-clusters, which are mutually exclusive and collectively containing all the m nodes on side 1

and n nodes on side 2. To aggregate the H sets of K_0 node co-clusters, BiTSC first constructs a node co-membership matrix for each run. Specifically, $\mathbf{M}^{(h)} = (m_{ij}^{(h)})_{(m+n)\times(m+n)}$ denotes a node co-membership matrix resulted from the h-th run. $\mathbf{M}^{(h)}$ is a binary and symmetric matrix indicating the pairwise cluster co-membership of the $(m+n)$ nodes. That is, an entry in $\mathbf{M}^{(h)}$ is 1 if the two nodes corresponding to its row and column are assigned to the same co-cluster; otherwise, it is 0. Then BiTSC constructs a consensus matrix $\bar{\mathbf{M}} = (\bar{m}_{ij})_{(m+n)\times(m+n)}$ by averaging $\mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(H)}$, i.e., $\bar{m}_{ij} = \sum_{h=1}^{H} m_{ij}^{(h)}/H \in [0,1]$. An entry of $\bar{\mathbf{M}}$ indicates the frequency that the two nodes corresponding to its row and column are assigned to the same co-cluster, among the H runs.

Lastly, BiTSC identifies tight node co-clusters from $\bar{\mathbf{M}}$ such that within every co-cluster, all pairs of nodes have been previously clustered together at a frequency of at least α, the input tightness parameter. Specifically, BiTSC considers $(1 - \bar{\mathbf{M}})$ as a pairwise distance matrix of $(m+n)$ nodes. Then BiTSC applies hierarchical clustering with complete linkage to $(1 - \bar{\mathbf{M}})$, and it subsequently cuts the resulting dendrogram at the distance threshold $(1-\alpha)$. This guarantees that all the nodes within each resulting co-cluster have pairwise distances no greater than $(1-\alpha)$, which is equivalent to being previously clustered together at a frequency of at least α. A larger α value will lead to finer co-clusters, i.e., a greater number of smaller clusters and unclustered nodes. BiTSC provides a visualization-based approach to help users choose α: for each candidate α value, BiTSC collects the nodes in the resulting tight co-clusters and plots a heatmap of the submatrix of $\bar{\mathbf{M}}$ that corresponds to these nodes; users are encouraged to pick an α value whose resulting number of tight co-clusters is close to the number of visible diagonal blocks in the heatmap. (Please see Appendix for a demonstration in the real data example in Section 2.3.2.) Regarding the choice of K_0, i.e., the input number of co-clusters in each run, the entries of $\bar{\mathbf{M}}$ provide a good guidance. A reasonable K_0 should lead to many entries equal to 0 or 1 and few having fractional values in between [88]. Following this reasoning, BiTSC implements a computationally efficient algorithm to automatically choose K_0 (Algorithm 2 in Section 2.D) while also giving users the option to input their preferred K_0 value. In Section 2.D, we demonstrate the use of

Algorithm 2 for the real data application (Section 2.3.2).

To summarize, BiTSC leverages joint information from bipartite network edges and node covariates to identify tight node co-clusters that are robust to data perturbation, i.e., subsampling. In its application to gene co-clustering, BiTSC integrates gene orthology information with gene expression data to identify tight gene co-clusters, which are enriched with orthologs and contain genes of similar expression patterns in both species. In particular, within each subsampling run, the bipartite spectral clustering step identifies co-clusters enriched with orthologs; another two steps, the kernel enhancement and the assignment of unsampled nodes, ensure that genes with similar expression patterns in each species tend to be clustered together. Moreover, the subsampling-and-aggregation approach makes the output tight gene co-clusters robust to the existence of outlier genes, which may have few orthologs or outlying gene expression patterns. The pseudocode of BiTSC is in Algorithm 1.

Algorithm 1 Pseudocode of BiTSC

- For $h = 1$ to H:
  1. Subsample $\tilde{m} = \lfloor \rho m \rfloor$ nodes from side 1 and $\tilde{n} = \lfloor \rho n \rfloor$ nodes from side 2 to obtain a subsampled bi-adjacency matrix $\tilde{\mathbf{A}}$ and two subsampled node covariate matrices $\tilde{\mathbf{X}}_1$ and $\tilde{\mathbf{X}}_2$
  2. Use kernel enhancement to construct an enhanced bi-adjacency matrix $\tilde{\mathbf{B}}$ from $\tilde{\mathbf{A}}$, $\tilde{\mathbf{X}}_1$, and $\tilde{\mathbf{X}}_2$
  3. Find K_0 initial node co-clusters from $\tilde{\mathbf{B}}$ by bipartite spectral clustering
  4. Obtain K_0 node co-clusters by assigning the unsampled nodes into the K_0 initial node co-clusters; encode the K_0 node co-clusters as a co-membership matrix $\mathbf{M}^{(h)}$
- Calculate the consensus matrix $\bar{\mathbf{M}}$ as the average of $\mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(H)}$
- Identify tight node co-clusters from $\bar{\mathbf{M}}$ with tightness parameter α

2.2.3 Six possible variants

In the development of BiTSC, we considered six possible variants of its algorithm, and we compared them with BiTSC to justify our choice of BiTSC as the proposed algorithm. The performance comparison is in Section 2.3.1.

1. Bipartite spectral clustering with kernel enhancement (Spectral-kernel). This algorithm applies kernel enhancement followed by bipartite spectral clustering to the original bipartite network with node covariates. Comparing it with BiTSC would help us evaluate the effectiveness of the subsampling-and-aggregation approach taken by BiTSC.

2. Bipartite spectral clustering (Spectral). This algorithm removes the kernel enhancement step from Spectral-kernel. Comparing it with Spectral-kernel would help us evaluate whether kernel enhancement is useful.

3. BiTSC-1. This algorithm differs from BiTSC in terms of the timing of subsampling. Unlike BiTSC, BiTSC-1 performs subsampling after applying kernel enhancement and bipartite spectral clustering to the original bipartite network. The use of "1" in the algorithm name means that bipartite spectral clustering is only applied for once. In detail, BiTSC-1 first performs kernel enhancement on the original bipartite network with node covariates to obtain an enhanced bi-adjacency matrix $\mathbf{B}$. BiTSC-1 next applies bipartite spectral clustering, same as in BiTSC but without the last K-means clustering step, to $\mathbf{B}$ to obtain $\mathbf{V}$, an $(m+n) \times K_0$ matrix. Then BiTSC-1 applies the subsampling-and-aggregation approach to the $(m+n)$ rows of $\mathbf{V}$. Specifically, in the h-th run, $h = 1, \ldots, H$, BiTSC-1 has the following three steps: (1) it randomly samples without replacement $\tilde{m}$ rows from the first m rows of $\mathbf{V}$ and $\tilde{n}$ rows from the last n rows of $\mathbf{V}$; (2) it divides the subsampled $(\tilde{m}+\tilde{n})$ rows into K_0 initial co-clusters using the K-means algorithm with Euclidean distance; (3) it assigns the unsampled $(m-\tilde{m}+n-\tilde{n})$ nodes to the K_0 initial co-clusters based on node covariates, same as in BiTSC. The remaining steps of BiTSC-1, including the aggregation of these H sets of K_0 node co-clusters into a consensus matrix and the identification of tight node

co-clusters, are the same as those of BiTSC. Comparing BiTSC-1 with BiTSC will help us evaluate the effect of the timing of subsampling, i.e., whether performing subsampling before kernel enhancement and bipartite spectral clustering aids the identification of tight node co-clusters.

4. BiTSC-1-nokernel. This algorithm removes the kernel enhancement step from BiTSC-1. Comparing it to BiTSC-1 would help us evaluate whether kernel enhancement is useful given the subsampling-and-aggregation approach.

5. BiTSC-1-NC. This algorithm modifies BiTSC-1 by changing how the unsampled nodes are assigned into the K_0 initial node co-clusters in each of the H runs. Specifically, BiTSC-1-NC only differs from BiTSC-1 in step (3) of the h-th run, $h = 1, \ldots, H$, where BiTSC-1-NC assigns the unsampled $(m - \tilde{m} + n - \tilde{n})$ nodes to the K_0 initial node co-clusters based on their corresponding rows in $\mathbf{V}$ instead of their covariates. In detail, BiTSC-1-NC calculates a mean vector for each initial node co-cluster as the average of its corresponding rows in $\mathbf{V}$; then BiTSC-1-NC assigns each unsampled node to the initial co-cluster whose mean vector has the smallest Euclidean distance to the node's corresponding row in $\mathbf{V}$. Note that "NC" in the algorithm name means that "no covariates" is used in the assignment step. Comparing BiTSC-1-NC with BiTSC-1 would help us evaluate the effect of using node covariates in the assignment step.

6. BiTSC-1-NC-nokernel. This algorithm removes the kernel enhancement step from BiTSC-1-NC. When compared to BiTSC-1-NC, it can help us evaluate whether kernel enhancement is useful in the absence of node covariates in the assignment step.

In summary, the above six possible variants of BiTSC can help us evaluate the design of BiTSC. Three variants pose contrasts to BiTSC in three aspects: Spectral-kernel does not use the subsampling-and-aggregation approach; BiTSC-1 uses a different timing for subsampling; BiTSC-1-NC does not use node covariates in the assignment of unsampled nodes to initial node co-clusters. The other three variants, Spectral, BiTSC-1-nokernel, and

BiTSC-1-NC-nokernel, remove the kernel enhancement from Spectral-kernel, BiTSC-1, and BiTSC-1-NC respectively to evaluate the effectiveness of kernel enhancement.

2.3 Results

2.3.1 Simulation

We designed multiple simulation studies to justify the algorithm design of BiTSC by comparing it with the six possible variants listed in Section 2.2.3: spectral-kernel, spectral, BiTSC-1, BiTSC-1-nokernel, BiTSC-1-NC, and BiTSC-1-NC-nokernel.

We use the weighted Rand index [94], defined in Section 2.B, as the evaluation measure of co-clustering results. The weighted Rand index compares two sets of node co-clusters: the co-clusters found by an algorithm and the true co-clusters used to generate data, and outputs a value between 0 and 1, with a value of 1 indicating perfect agreement between the two sets. The weighted Rand index is a proper measure for evaluating BiTSC and its variants because it accounts for noise nodes that do not belong to any co-clusters.

We compared BiTSC with its six possible variants in identifying node co-clusters from simulated networks with varying levels of noise nodes (i.e., θ in Section 2.3.1.1) and varying average degrees of nodes. It is expected that the identification would become more difficult as the level of noise nodes increases or as the average degree decreases. Our results in Figure 2.2 (a) are consistent with this expectation. Figure 2.2 (a) also shows that BiTSC consistently outperforms its six variant algorithms at all noise node levels and average degrees greater than five. This phenomenon is reasonable because BiTSC performs subsampling on the network, and the subsampled network, if too sparse, would make the bipartite spectral clustering algorithm fail. In fact, the three algorithms that outperform BiTSC for sparse networks, i.e., spectral-kernel, BiTSC-1, and BiTSC-1-NC, only perform bipartite spectral clustering on the entire network, so they are more robust to network sparsity. Additionally, we observe that the three variants that do not use kernel enhancement consistently have the worst performance. In summary, BiTSC has a clear advantage over its possible variants in the existence of noise

nodes and when the network is not overly sparse. These results confirm the effectiveness of the subsampling-and-aggregation approach and the kernel enhancement step, and they also show that subsampling in the first step is beneficial if the network is not too sparse, thus justifying the design of BiTSC.

In addition to validating the design of BiTSC, we also performed simulation studies to compare BiTSC with OrthoClust [81], a gene clustering method that also simultaneously uses gene expression and orthology information. We chose OrthoClust as the baseline method to compare BiTSC against because OrthoClust is the only recent method that does not (1) exclude genes not in one-to-one orthologs like SCSC [79] and MVBC [80] do or (2) have strong distributional assumptions as SCSC does. Moreover, OrthoClust has a unipartite network formulation, so its comparison with BiTSC would inform the effectiveness of our bipartite network formulation. The OrthoClust software takes three input files: two within-species gene co-expression networks constructed by users and one between-species ortholog network. Following the recommendation in OrthoClust, we constructed the within-species gene co-expression networks by connecting each gene to its closest gene(s), whose number is specified by a tuning parameter `rank`, in terms of Euclidean distance. OrthoClust allows users to specify a κ value (in the between-species ortholog network input file), which balances the weights of within-species edges and between-species edges in the cost function. Our results in Figure 2.2b show that BiTSC consistently outperforms OrthoClust under various simulation settings.

Furthermore, we performed simulation studies to show the robustness of BiTSC to its input parameters: K_0 (the number of co-clusters to identify in each subsampling run), ρ (the proportion of nodes to subsample in each run), and $\tau = (\tau_1, \tau_2)$ (the tuning parameters in the kernel enhancement step) (Section 2.C). We find that BiTSC performs well when K_0 is set to be equal to or larger than K, the number of true co-clusters (Figure 2.10a). For ρ, we recommend a default value of 0.8 (Figure 2.10b). For τ, we recommend a default value of $(1, 1)$ based on Figures 2.10c and 2.11, which show that small τ values lead to better clustering results when the network is sparse, and that BiTSC becomes more robust to τ

values when the network becomes denser. In general, small τ values put large weights on node covariates, while large τ values make BiTSC use more of the edge information to find node clusters. Users have the flexibility to set τ values based on their confidence or preference on node covariates and edges. For instance, if a user would like to find co-clusters such that between-species edges are dense but within-species gene expression might be dissimilar, he or she may opt for larger τ values provided that the kernel enhancement compensates the edge sparsity enough so that BiTSC can successfully run. Overall, BiTSC is robust to the specification of these tuning parameters.

2.3.1.1 Data generation in simulation studies

Here we describe how we generate bipartite networks with node covariates in simulation studies (Section 2.3.1). The generation process comprises three steps.

1. Network structure setup. To evaluate the capacity of BiTSC in detecting tight node co-clusters and leaving out outlier nodes, we generate a bipartite network including two types of nodes: clustered nodes in co-clusters and noise nodes that do not belong to any co-clusters. We consider K non-overlapping node co-clusters, each of which contains n_1 nodes from side 1 and n_2 nodes from side 2. Hence, there are Kn_1 and Kn_2 clustered nodes on side 1 and 2, respectively. We define θ as the ratio (# of noise nodes)/(# of clustered nodes). Then there are $\lfloor \theta K n_1 \rfloor$ and $\lfloor \theta K n_2 \rfloor$ noise nodes on side 1 and 2, respectively. Therefore, the bipartite network contains a total of $m = Kn_1 + \lfloor \theta K n_1 \rfloor$ nodes on side 1 and $n = Kn_2 + \lfloor \theta K n_2 \rfloor$ nodes on side 2.

2. Edge generation. We generate independent binary edges between nodes by following a stochastic block model [21]: if two nodes belong to the same co-cluster, an edge between them is drawn from Bernoulli(p), where $p \in (0, 1)$ is the within-cluster edge probability; otherwise, an edge is drawn from Bernoulli(q), where $q \in (0, p)$ is the not-within-cluster edge probability smaller than p. A larger q/p ratio would lead to a more obscure node co-cluster structure in the resulting bipartite network.

3. Covariate generation. We generate covariate vectors for clustered nodes and noise nodes separately. First, we assume that nodes in each co-cluster on each side have covariate vectors following a multivariate Gaussian distribution. In detail, for the n_1 nodes in the k-th co-cluster on side 1, we independently draw n_1 vectors of length p_1 from a p_1-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{1k}, (\omega_1 k)^2 \mathbf{I}_{p_1})$, $k = 1, \ldots, K$. Note that the larger the co-cluster index k, the more spread out the n_1 covariate vectors are around the mean $\boldsymbol{\mu}_{1k}$. The K co-cluster mean vectors on side 1, $\boldsymbol{\mu}_{11}, \ldots, \boldsymbol{\mu}_{1K}$, are independently drawn from a p_1-dimensional Gaussian prior $\mathcal{N}(\mathbf{0}, \sigma_1^2 \mathbf{I}_{p_1})$. Nodes in co-clusters on side 2 are simulated similarly from K p_2-dimensional Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_{2k}, (\omega_2 k)^2 \mathbf{I}_{p_2})$, with the co-cluster mean vectors on side 2, $\boldsymbol{\mu}_{21}, \ldots, \boldsymbol{\mu}_{2K}$, independently drawn from a p_2-dimensional Gaussian prior $\mathcal{N}(\mathbf{0}, \sigma_2^2 \mathbf{I}_{p_2})$. Second, for noise nodes that do not belong to any co-clusters on each side, we randomly generate their covariate vectors from uniform distributions defined by the ranges of the already-generated covariate vectors of clustered nodes. In detail, on side 1, we take the Kn_1 clustered nodes and define a range based on their values in each of the p_1 dimensions. Then we independently draw $\lfloor \theta K n_1 \rfloor$ scalars uniformly from the range of each dimension to construct $\lfloor \theta K n_1 \rfloor$ covariate vectors of length p_1. Similarly, we simulate covariate vectors of noise nodes on side 2.

In summary, the data generation process requires the following input parameters: K, the number of true co-clusters; n_r, the number of nodes in each co-cluster on side r; θ, the ratio of the number of noise nodes over the number of clustered nodes; p, the within-cluster edge probability; q, the not-within-cluster edge probability; p_r, the dimension of covariate vectors on side r; ω_r, the parameter for within-cluster variance on side r; σ^2, the variance parameter for generating co-cluster mean vectors on side r, $r = 1, 2$.

We simulated an example bipartite network using the approach described above, with $H = 50$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, $q/p = 5$, and $q = 0.03$. Figure 2.9a-b illustrate the bi-adjacency matrix and the true co-membership matrix of this simulated network. Figure 2.9c-d show the node covariates.

2.3.1.2 Average degree

Within each of the K true co-clusters, there are n_1 and n_2 nodes on side 1 and 2, respectively. The noise node ratio is denoted by θ; that is, there are $\lfloor n_1 K\theta \rfloor$ and $\lfloor n_2 K\theta \rfloor$ noise nodes on side 1 and 2. Hence, in total there are $m = n_1 K + \lfloor n_1 K\theta \rfloor$ nodes on side 1 and $n = n_2 K + \lfloor n_2 K\theta \rfloor$ nodes on side 2. Recall that $\mathbf{A} = (a_{ij})_{m\times n}$ is the bi-adjacency matrix. Let $\lambda_{1i} = \sum_{j=1}^{n} a_{ij}$ and $\lambda_{2j} = \sum_{i=1}^{m} a_{ij}$ be the degree of node i on side 1 and node j on side 2, respectively. Then, the average degree of the network is

$$\lambda = \frac{\sum_{i=1}^{m} \lambda_{1i} + \sum_{j=1}^{n} \lambda_{2j}}{m+n} = \frac{2\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}}{m+n}\,. \tag{2.1}$$

2.3.2 Real data analysis

In this section, we demonstrate how BiTSC is capable of identifying conserved gene co-clusters of *D. melanogaster* (fly) and *C. elegans* (worm). We compared BiTSC with OrthoClust [81] and performed a series of downstream bioinformatics analysis to both validate BiTSC and verify the biological significance of its identified co-clusters. We compared BiTSC with OrthoClust and not SCSC or MVBC for the same reasons as described in Section 2.3.1.

2.3.2.1 BiTSC outperforms OrthoClust in identifying gene co-clusters with enriched ortholog pairs and similar expression levels

We applied BiTSC and OrthoClust to the *D. melanogaster* and *C. elegans* developmental-stage RNA-seq data generated by the modENCODE consortium [96] and the gene orthology annotation from the TreeFam database [97]. For data processing, please see Section 2.3.2.3. We ran BiTSC with input parameters $H = 100$, $\rho = 0.8$, $\tau = (1, 1)$, $K_0 = 30$, and $\alpha = 0.9$. For the choices of K_0 and α, please see Section 2.D. We ran OrthoClust by following the instruction on its GitHub page (https://github.com/gersteinlab/OrthoClust accessed on Nov 12, 2019). Specifically, we constructed the within-species gene co-expression networks by connecting each gene with its closest five genes in terms of Pearson correlation and used $\kappa = 1$ (the default value). For details regarding the computational time of BiTSC and OrthoClust,

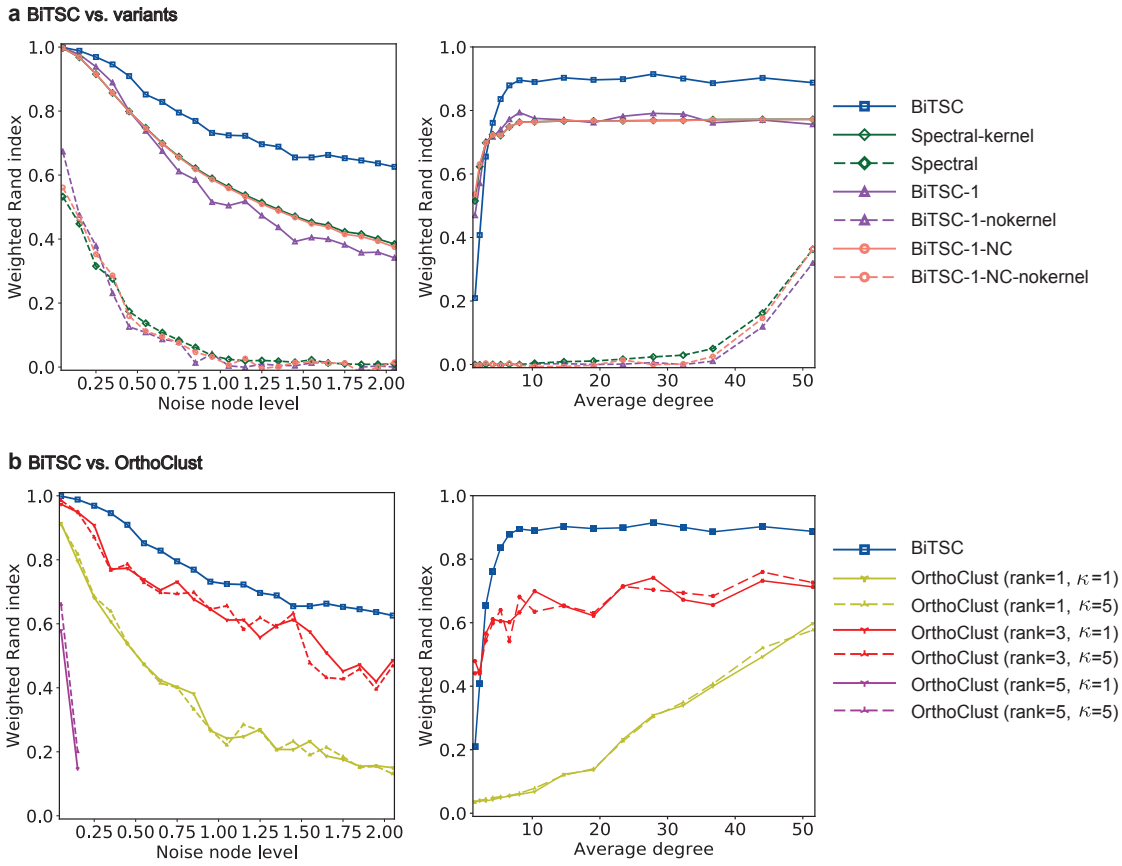

Figure 2.2: Performance of BiTSC vs. (a) its six variant algorithms (Section 2.2.3) and (b) OrthoClust. The weighted Rand index is plotted as a function of noise node level (first column) or average degree (second column). The data sets are simulated using the approach described in Section 2.3.1.1. For both columns, we set $K = 15$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, and $\omega_1 = \omega_2 = 0.1$. For the first column, we vary the noise node level θ from 0.05 to 2.05 and set $p = 0.15$ and $q = 0.03$. For the second column, we set $\theta = 0.5$, vary p from 0.005 to 0.175, set $q = p/5$, and as a result vary the average degree from 1 to 51 (see Section 2.3.1.2 for details about the average degree). For the input parameters of BiTSC and its variants, we choose $H = 50, \rho = 0.8, \tau = (1, 1), K_0 = K = 15$ and $\alpha = 0.7$ whenever applicable. We set $K_0 = K$ following the convention that when the ground truth is known in simulation studies, one can select input parameters based on the truth [25, 60, 84, 95]. For OrthoClust, rank = 3 means that in the construction of within-species gene co-expression networks, we connect each gene with its three closest genes in terms of Euclidean distance; similarly for rank = 1 and rank = 5. In this figure, genes in identified co-clusters that only contain genes from one species are treated as noise genes, just like unclustered genes, in the calculation of the weighted Rand index. This is why in the second column of (b) we cannot see the curves for OrthoClust (rank = 5) — no co-clusters with genes from both species were identified. We show that BiTSC also outperforms OrthoClust when we only consider unclustered genes as noise genes in Figure 2.5.

please see Section 2.E. To compare BiTSC and OrthoClust, we picked a similar number of large gene co-clusters identified by either method: 16 BiTSC co-clusters with at least 10 genes in each species (Table 2.1) vs. 14 OrthoClust co-clusters with at least 2 genes in each species (Table 2.2). Compared with OrthoClust, the co-clusters identified by BiTSC are more balanced in sizes between fly and worm. In contrast, OrthoClust co-clusters typically have many genes in one species but few genes in the other species; in particular, if we restricted the OrthoClust co-clusters to have at least 10 genes in each species, only two co-clusters would be left. We also ran OrthoClust again with $\kappa = 3$ (the value used in OrthoClust paper) instead of $\kappa = 1$, and the result is very similar (Table 2.3). We evaluated both methods' identified co-clusters in two aspects (using $\kappa = 1$ for OrthoClust): the enrichment of orthologous genes and the similarity of gene expression levels in each co-cluster. Figure 2.3 shows that the BiTSC co-clusters exhibit both stronger enrichment of orthologs and higher similarity of gene expression than the OrthoClust co-clusters do. Note that in general, the co-clusters identified by OrthoClust (Table 2.2), with cluster size ranging between 184 and 1002, are larger than those identified by BiTSC (Table 2.1). Although the difference in cluster size does not invalidate our analysis, for further investigation, we adjusted the input parameters of BiTSC to obtain co-clusters that are closer to those identified by OrthoClust in size. With the new input parameter choices for BiTSC, we are able to further confirm the advantages of BiTSC in Figure 2.15. Therefore, the gene co-clusters identified by BiTSC have better biological interpretations than their OrthoClust counterparts because of their more balanced gene numbers in two species, greater enrichment of orthologs, and better grouping of genes with similar expressions.

2.3.2.2 Functional analysis verifies the biological significance of BiTSC gene co-clusters

We next analyzed the 16 gene co-clusters identified by BiTSC. First, we verified that genes in each co-cluster exhibit similar functions within fly and worm. We performed the GO term enrichment test (Section 2.3.3.1) for each co-cluster in each species. The results are

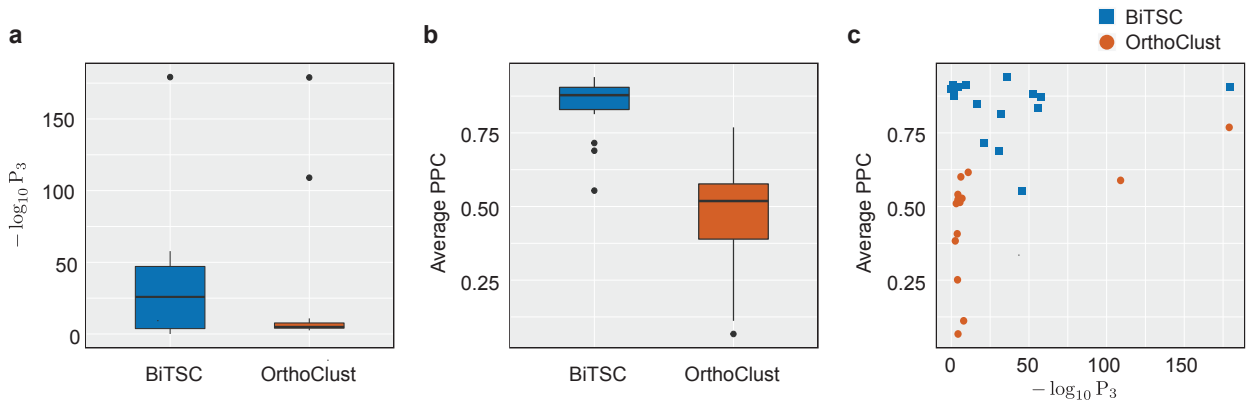

Figure 2.3: Comparison of BiTSC and OrthoClust in terms of their identified fly-worm gene co-clusters (Section 2.3.2.1). (a) Distributions of within-cluster enrichment of ortholog pairs. For BiTSC and OrthoClust, a boxplot is shown for the $-\log_{10} P_3$ values calculated by the ortholog pair enrichment test (Section 2.3.3.3) on the identified gene co-clusters. Larger $-\log_{10} P_3$ values indicate stronger enrichment. (b): Distributions of within-cluster gene expression similarity. For BiTSC and OrthoClust, a boxplot is shown for the average pairwise Pearson correlation (PPC) between genes of the same species within each identified co-cluster. (c): Within-cluster gene expression similarity vs. ortholog enrichment. Each point corresponds to one co-cluster identified by BiTSC or OrthoClust. The $-\log_{10} P_3$ and average PPC values are the same as those shown in (a) and (b).

summarized in Figure 2.6, which show that every co-cluster has strongly enriched GO terms with extremely small p-values, i.e., P_1 values. Hence, genes in every co-cluster indeed share similar biological functions within fly and worm. We also calculated the pairwise Pearson correlation coefficients between genes of the same species within each co-cluster (Figure 2.7). The overall high correlation values also confirm the within-cluster functional similarity in each species. Second, we show that within each co-cluster, genes share similar biological functions between fly and worm. We performed the GO term overlap test (Section 2.3.3.2), which output small p-values, i.e., P_2 values, suggesting that fly and worm genes in each co-cluster have a significant overlap in their GO terms. Figure 2.6 also illustrates this functional similarity between fly and worm genes in the same co-cluster. In summary, the 16 gene co-clusters exhibit clear biological functions, some of which are conserved between fly and worm.

The above analysis results are summarized in Table 2.1. Specifically, for each co-cluster, Table 2.1 lists the numbers of fly and worm genes, the numbers of genes lacking BP GO term annotations, example GO terms enriched in both species by the GO term enrichment test, and p-values from the GO term overlap test and the ortholog enrichment test (Section 2.3.3). Interestingly, we observe that when BiTSC identifies co-clusters, it simultaneously leverages gene expression similarity and gene orthology to complement each other. For example, co-clusters 10 and 11 do not have strong enrichment of orthologs but exhibit extremely high similarity of gene expression in both fly and worm; on the other hand, co-cluster 13 have relatively weak gene expression similarity but particularly strong enrichment of orthologs. This advantage of BiTSC would enable it to identify conserved gene co-clusters even based on incomplete orthology information.

We further visualized the 16 gene co-clusters using the concensus matrix $\bar{\mathbf{M}}$. Figure 2.4 plots the fly and worm genes in these co-clusters, as well as 1,000 randomly sampled unclustered genes as a background. Figure 2.13 shows that the pattern of the 16 co-clusters is robust to the random sampling of unclustered genes. We observe that many co-clusters are well separated, suggesting that genes in these different co-clusters are rarely clustered together.

We also see that some co-clusters are close to each other, including co-clusters 1 and 12, co-clusters 9 and 16, and co-clusters 10, 11 and 14. To investigate the reason behind this phenomenon, we inspected Table 2.1 and Figure 2.6 to find that overlapping co-clusters share similar biological functions. This result again confirms that the identified gene co-clusters are biologically meaningful.

Moreover, we computationally validated BiTSC's capacity of predicting unknown gene functions. Many co-clusters contain genes that do not have BP GO terms. For each of these genes, we predicted its BP GO terms as its co-cluster's top enriched BP GO terms. Then we compared the predicted BP GO terms with the gene's other functional annotation, in particular, molecular function (MF) or cellular component (CC) GO terms. Our comparison results in Figure 2.14 show that the predicted BP GO terms are highly compatible with the known MF or CC GO terms, suggesting the validity of our functional prediction based on the BiTSC co-clusters.

2.3.2.3 Data processing in the real data application

In the real data application of BiTSC (Section 2.3.2), the *D. melanogaster* (fly) and *C. elegans* (worm) data set consists of two parts: gene expression data and gene orthology information. For gene expression data, we started with 15,095 fly protein-coding genes' expression levels across 30 developmental stages and 44,969 worm protein-coding genes' expression levels across 35 developmental stages. Note that the gene expression levels are in the FPKM (Fragments Per Kilobase of transcript per Million mapped reads) unit and were processed from RNA-seq data collected by the modENCODE Consortium [96, 99]. For gene orthology information, we started with 11,403 ortholog pairs between the above mentioned fly and worm genes (obtained and processed from the TreeFam database [97, 99]). Then we removed all the fly and worm genes that have zero expression levels across all the developmental stages and have no orthologous genes, leaving us with 5,414 fly genes, 5,731 worm genes, and 10,975 ortholog pairs. After that, we performed the logarithmic transformation on the gene expression levels and standardized the transformed levels by subtracting the mean and dividing the standard

Table 2.1: Fly-worm gene co-clusters identified by BiTSC (Section 2.3.2)

Co-cluster	# of fly genes[1] (without GO[2])	# of worm genes (without GO)	Examples BP GO terms highly enriched in both species[3]	P_2[4]	P_3[5]
1	106 (19)	83 (15)	Chemical synaptic transmission; synaptic signaling	1.35e-07	2.77e-53
2	46 (8)	119 (28)	Muscle cell development	1.70e-09	1.50e-17
3	75 (3)	83 (6)	Peptide and amide biosynthetic process	1.17e-08	6.23e-180
4	73 (4)	50 (13)	ATP metabolic process	2.72e-12	1.61e-58
5	57 (4)	62 (8)	Protein catabolic process; proteolysis	3.15e-09	3.15e-56
6	83 (4)	36 (8)	Mitochondrial translation; mitochondrial gene expression	2.60e-03	2.04e-32
7	89 (13)	25 (8)	Protein localization to endoplasmic reticulum	2.69e-10	7.72e-22
8	80 (16)	26 (11)	Ribosome biogenesis; RNA metabolic processing	2.05e-16	7.50e-37
9	29 (3)	76 (19)	Cilium and cell projection organization	2.03e-09	3.81e-05
10	32 (2)	24 (6)	DNA replication and metabolic process	2.98e-06	6.53e-02
11	25 (4)	19 (4)	DNA replication	2.46e-06	1.00e-00
12	28 (4)	15 (4)	G protein-coupled glutamate receptor signaling pathway	4.17e-14	1.81e-10
13	16 (7)	25 (12)	Glycoside catabolic process; transmembrane transport	1.52e-02	5.29e-46
14	15 (1)	24 (4)	DNA metabolic process; cell cycle process	1.83e-05	3.11e-02
15	24 (2)	11 (1)	Oxidation-reduction process	1.41e-03	2.36e-31
16	14 (0)	15 (1)	Cilium organization; cell projection assembly	1.20e-07	1.83e-02

[1] Number of fly genes in the co-cluster

[2] Number of fly genes without BP GO term annotations in the co-cluster

[3] Examples of BP GO terms that are highly enriched in both species in the co-cluster (Section 2.3.3.1). These example BP GO terms are used as labels of the co-clusters in Figure 2.4.

[4] p-value of the co-cluster based on the GO term overlap test (Section 2.3.3.2)

[5] p-value of the co-cluster based on the ortholog enrichment test (Section 2.3.3.3)

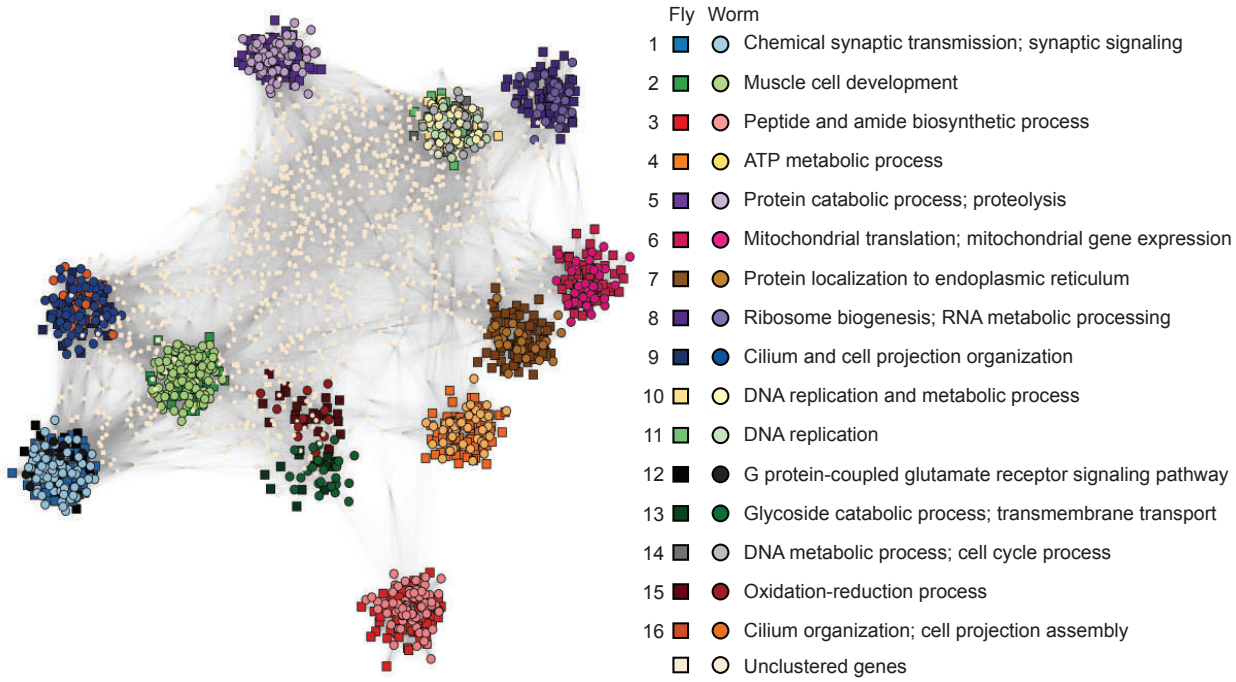

Figure 2.4: Visualization of the 16 gene co-clusters found by BiTSC from the fly-worm gene network (Section 2.3.2). The visualization is based on the consensus matrix $\bar{\mathbf{M}}$ using the R package igraph [98] (https://igraph.org, the Fruchterman-Reingold layout algorithm). Genes in the 16 co-clusters are marked by distinct colors, with squares and circles representing fly and worm genes respectively. For each co-cluster, representative BP GO terms are labeled (Table 2.1). 1,000 randomly-chosen unclustered genes are also displayed and marked in white. In this visualization, both the gene positions and the edges represent values in $\bar{\mathbf{M}}$. The higher the consensus value between two genes, the closer they are positioned, and the darker the edge is between them. If the consensus value is zero, then there is no edge.

deviation for every gene (Section 2.3.2.4). Finally, we built a bipartite network of fly and worm genes by connecting orthologous genes, and we collected every gene's standardized expression levels into its node covariate vector. In this resulting bipartite network with node covariates (Table 1), $m = 5,414$, $n = 5,731$, $p_1 = 30$, and $p_2 = 35$. The average degree of this bipartite network is 1.97 (Section 2.3.1.2).

2.3.2.4 Processing of gene expression levels

Gene expression values in the FPKM unit are typically highly skewed with the presence of extremely large values. Logarithmic transformation has been widely used to transform FPKM values to reduce the effects of outliers and to make the transformed values more normally distributed [100–102]. Following the notations in Section 2.2.1, for gene i on side r, we constructed its j-th covariate as

$$x_{rij} = \frac{l_{rij} - \bar{l}_{ri\cdot}}{\sqrt{\sum_{j=1}^{p_r}(l_{rij} - \bar{l}_{ri\cdot})^2/(p_r - 1)}}, \tag{2.2}$$

where $l_{rij} = \log_2(\text{FPKM}_{rij} + 1)$ and $\bar{l}_{ri\cdot} = \sum_{j=1}^{p_r} l_{rij}/p_r$. In other words, the gene covariates are standardized log-transformed FPKM values. We used these covariates in our fly-worm data analysis in Section 2.3.2.

2.3.3 Bioinformatics gene function analysis

Here we introduce three hypothesis tests, which are designed to analyze the gene co-clusters identified by BiTSC in the real data application (Section 2.3.2). The three tests are from [99] and described in detail below. Note that we only include biological process (BP) gene ontology (GO) terms in the GO term enrichment test and the GO term overlap test for ease of interpretation. The GO terms are from the R package GO.db [103].

2.3.3.1 GO term enrichment test

Given a gene co-cluster, the GO term enrichment test is to check for each species, whether a GO term is enriched in this co-cluster relative to all the genes in that species. The top enriched GO terms would indicate the biological functions of this co-cluster in each species.

Suppose that there are u genes of species 1 in this co-cluster, v of which are annotated with a given GO term. Also suppose that species 1 has a total of U genes, V of which are annotated with the same GO term. The null hypothesis is that this GO term has the same enrichment level in the u genes as in the U genes, i.e., the u genes are randomly sampled from the U genes. The alternative hypothesis is that this GO term is more enriched in the u genes relative to the U genes. Under the null hypothesis, X, the number of genes that are annotated with this GO term among any u genes, follows a hypergeometric distribution with the following probability mass function

$$P(X = x) = \frac{\binom{V}{x}\binom{U-V}{u-x}}{\binom{U}{u}},\ x = 0, 1, \cdots, \min(V, u)\,.$$

Hence, this test has a p-value defined below and denoted by P_1.

$$\mathrm{P}_1 = P(X \geq v) = \sum_{x=v}^{\min(V,u)} \frac{\binom{V}{x}\binom{U-V}{u-x}}{\binom{U}{u}}\,.$$

We implemented this test, which is equivalent to the one-tail Fisher's exact test, using the R package topGO [104].

2.3.3.2 GO term overlap test

Given a gene co-cluster, the GO term overlap test is to check whether the genes from the two species share similar biological functions, i.e., whether the two sets of genes have a significant overlap in their annotated GO terms.

Specifically, for this co-cluster, we denote by A and B the sets of GO terms associated with its genes in species 1 and 2, respectively. We define a population of N GO terms as the set of terms associated with any genes in species 1 or 2. The null hypothesis is that A and B

are two independent samples from the population, i.e., the common GO terms shared by two species in this co-cluster is purely due to a random overlap. The alternative hypothesis is that A and B are positively dependent samples. Under the null hypothesis that two samples with sizes $|A|$ and $|B|$ are independently drawn from the population, Y, the number of common GO terms shared by the two samples, has the following probability mass function

$$P(Y=y)=\frac{\binom{N}{y}\binom{N-y}{|A|-y}\binom{N-|A|}{|B|-y}}{\binom{N}{|A|}\binom{N}{|B|}},\ y=0,1,\cdots,\min(|A|,|B|)\,.$$

Hence, this test has a p-value defined below and denoted by P_2.

$$\mathrm{P}_2=P(Y\geq|A\cap B|)=\sum_{y=|A\cap B|}^{\min(|A|,|B|)}\frac{\binom{N}{y}\binom{N-y}{|A|-y}\binom{N-|A|}{|B|-y}}{\binom{N}{|A|}\binom{N}{|B|}}.$$

2.3.3.3 Ortholog enrichment test

Given a gene co-cluster, the ortholog enrichment test is to check whether the genes from the two species are rich in orthologs. This test will work as a sanity check for BiTSC, which is expected to output co-clusters enriched with orthologs.

For example, given *D. melanogaster* (fly) and *C. elegans* (worm), the two species in our real data application (Section 2.3.2), we denote the population of ortholog pairs between fly and worm by $O=\{(f_1,w_1),\cdots,(f_M,w_M)\}$, where f_i and w_i are the fly gene and worm gene in the i-th ortholog pair. Note that $f_1,\ldots,f_M$ contain repetitive genes and so do $w_1,\cdots,w_M$. Given a co-cluster, F denotes its set of fly genes, and $F'=\{(f_i,w_i):f_i\in F\}$ denotes the set of ortholog pairs whose fly genes are in F. Similarly, W denotes the set of worm genes in this co-cluster, and $W'=\{(f_i,w_i):w_i\in W\}$ denotes the set of ortholog pairs whose worm genes are in W. Note that $F'\cap W'$ is the set of ortholog pairs between fly genes in F and worm genes in W. The null hypothesis is that F' and W' are two independent samples from O, i.e., their common ortholog pairs in $F'\cap W'$ are purely due to a random overlap. The alternative hypothesis is that F' and W' are positively dependent samples. Under the null hypothesis that two samples with sizes $|F'|$ and $|W'|$ are independently drawn from O, Z, the number of ortholog pairs shared by the two samples, has the following probability mass function

$$P(Z=z)=\frac{\binom{M}{z}\binom{M-z}{|F'|-z}\binom{M-|F'|}{|W'|-z}}{\binom{M}{|F'|}\binom{M}{|W'|}},\ z=0,1,\cdots,\min(|F'|,|W'|)\,.$$

Hence, this test has a p-value defined below and denoted by P_3.

$$\mathrm{P}_3=P(Z\geq|F'\cap W'|)=\sum_{z=|F'\cap W'|}^{\min(|F'|,|W'|)}\frac{\binom{M}{z}\binom{M-z}{|F'|-z}\binom{M-|F'|}{|W'|-z}}{\binom{M}{|F'|}\binom{M}{|W'|}}.$$

2.4 Discussion

BiTSC is a general bipartite network clustering algorithm. It is unique in identifying tight node co-clusters such that nodes in a co-cluster share similar covariates and are densely connected. In addition to cross-species gene co-clustering, BiTSC has a wide application potential in biomedical research. In general, BiTSC is applicable to computational tasks that can be formulated as a bipartite network clustering problem, where edges and node covariates jointly indicate a co-clustering structure. Here we list three examples. The first example is the study of transcription factor (TF) co-regulation. In a TF-gene bipartite network, TFs and genes constitute nodes of two sides, an edge indicates that a TF regulates a gene, and node covariates are expression levels of TFs and genes. BiTSC can identify TF-gene co-clusters so that every co-cluster indicates a group of TFs co-regulating a set of genes. The second example is cross-species cell clustering. One may construct a biparite cell network, in which cells of one species form nodes of one side, by drawing an edge between cells of different species if the two cells are similar in some way, e.g., co-expression of orthologous genes. Node covariates may be gene expression levels and other cell characteristics. Then BiTSC can identify cell co-clusters as conserved cell types in two species. The third example is drug repurposing. One may construct a drug-target bipartite network by connecting drugs to their known targets (usually proteins) and including biochemical properties of drugs and targets as node covariates [105]. BiTSC can then identify drug-target co-clusters to reveal new potential targets of drugs.

A natural generalization of BiTSC is to identify node co-clusters in a multipartite network,

which has more than two types of nodes. An important application of multipartite network clustering is the identification of conserved gene co-clusters across multiple species. Here we describe a possible way of generalizing BiTSC in this application context. Suppose that we want to identify conserved gene co-clusters across three species: *Homo sapiens* (human), *Mus musculus* (mouse), and *Pan troglodytes* (chimpanzee). We can encode the three-way gene orthology information in a tripartite network and include gene expression levels as node covariates. To generalize BiTSC, we may represent the tripartite network as three bi-adjacency matrices (one for human and mouse, one for human and chimpanzee, and one for mouse and chimpanzee) and three covariate matrices, one per species. A key step in this generalization is to stack three (subsampled and kernel-enhanced) bi-adjacency matrices into a unipartite adjacency matrix and apply spectral clustering. Other parts of BiTSC, such as the subsampling-and-aggregation approach, the assignment of unsampled nodes, and the hierarchical clustering in the last step to identify tight co-clusters, will stay the same. We have implemented this functionality in the BiTSC software package. Compared to the existing method OrthoClust [81] that can also perform multi-species gene co-clustering, BiTSC is more transparent in its combination of gene orthology and expression information (because guidance is provided for the selection of each tuning parameter in BiTSC) and is more focused on identifying gene co-clusters rather than within-species gene clusters.

BiTSC is also generalizable to find tight node co-clusters in a bipartite network with node covariates on only one side or completely missing. In the former case, we will perform a one-sided kernel enhancement on the bi-ajdancecy matrix by using available node covariates on one side. We also need to perform bipartite spectral clustering on the whole network to obtain an Euclidean embedding, i.e., the matrix $\mathbf{V}$ in BiTSC-1 (Section 2.2.3), for the nodes without covariates. Then we can apply the same subsampling-and-aggregation approach as in BiTSC, except that in each subsampling run we will assign the unsampled nodes without covariates into initial co-clusters based on Euclidean embedding instead of node covariates. In the latter case where all nodes have no covariates, we will skip the kernel enhancement step, and BiTSC-1-NC, a variant of BiTSC described in Section 2.2.3, will be applicable.

Another extension of BiTSC is to output soft co-clusters instead of hard co-clusters. In soft clustering, a node may belong to multiple clusters in a probabilistic way, allowing users to detect nodes whose cluster assignment is ambiguous. Here we describe two ideas of implementing soft clustering in BiTSC. The first idea is that after we obtain the consensus matrix ($\bar{\mathbf{M}}$), we replace the current hierarchical clustering by spectral clustering to find the final co-clusters; inside spectral clustering, we use fuzzy c-means clustering [106, 107] instead of the regular K-means to find soft co-clusters. The second idea is that after we obtain the distance matrix ($\mathbf{1} - \bar{\mathbf{M}}$), we use multidimensional scaling to find a two-dimensional embedding of the nodes and then perform fuzzy c-means clustering to find soft co-clusters.

Lastly, we comment that the relationship between community detection and link prediction is an open question in network research. We preliminarily explored the performance of BiTSC and OrthoClust in terms of link prediction in the fly-worm network (Section 2.3.2) and found that BiTSC achieves a reasonable and better performance than OrthoClust in this task (Figure 2.16). We leave further investigation regarding the relative advantages and disadvantages of using community detection methods vs. existing supervised learning methods for link prediction to future research.

To summarize, BiTSC is a flexible algorithm that is generalizable for multipartite networks, bipartite networks with partial node covariates, and soft node co-clustering. This flexibility will make BiTSC a widely-applicable clustering method in network analysis.

2.5 Acknowledgments

Chapter 2 is a version of [108], which is the joint work with Heather J. Zhou, and Dr. Jingyi Jessica Li. We acknowledge that the original idea of formulating the fly-worm co-clustering problem as bipartite network community detection is from Dr. Zahra Razaee's previous work at UCLA [84]. Our BiTSC algorithm also incorporates the kernel enhancement technique from Dr. Razaee's dissertation [87].

APPENDIX

2.A Code

The Python package BiTSC is open-access and available at `https://github.com/edensunyidan/BiTSC`.

2.B Evaluation metric of clustering result

In the simulation study (Section 2.3.1), we use the weighted Rand index [94], which the extension of the adjusted Rand index [109, 110], to evaluate the clustering result by comparing the identified co-clusters to the true co-clusters. The weighted Rand index was developed for the case where noise nodes exist and should stay unclustered. We use $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_K, \mathcal{V}_{K+1}\}$ to denote the true node cluster membership, where $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_K$ indicate the K true co-clusters and $\mathcal{V}_{K+1}$ indicates the set of noise nodes. We use $\tilde{\mathcal{V}} = \{\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_2, \ldots, \tilde{\mathcal{V}}_C, \tilde{\mathcal{V}}_{C+1}\}$ to denote the clustering result, where $\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_2, \ldots, \tilde{\mathcal{V}}_C$ indicate the C identified co-clusters and $\tilde{\mathcal{V}}_{C+1}$ represents the set of unclustered nodes.

Thalamuthu et al. proposed two types of adjusted Rand index. The first one, $\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}})$, considers $\mathcal{V}_{K+1}$ and $\tilde{\mathcal{V}}_{C+1}$ as two regular clusters:

$$\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}}) = \frac{\sum_{i=1}^{K+1}\sum_{j=1}^{C+1}\binom{N_{ij}}{2} - \sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2}\sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}/\binom{N}{2}}{0.5\left[\sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2} + \sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}\right] - \sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2}\sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}/\binom{N}{2}},$$

where $N_{ij} = |\mathcal{V}_i \cap \tilde{\mathcal{V}}_j|$, $N_{i\cdot} = \sum_{j=1}^{C+1} N_{ij}$, $N_{\cdot j} = \sum_{i=1}^{K+1} N_{ij}$, and $N = \sum_{i=1}^{K+1}\sum_{j=1}^{C+1} N_{ij}$. Note that N denotes the total number of nodes.

The second index $\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}})$ ignores $\mathcal{V}_{K+1}$ and $\tilde{\mathcal{V}}_{C+1}$:

$$\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}}) = \frac{\sum_{i=1}^{K}\sum_{j=1}^{C}\binom{N_{ij}}{2} - \sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2}\sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}/\binom{\tilde{N}}{2}}{0.5\left[\sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2} + \sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}\right] - \sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2}\sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}/\binom{\tilde{N}}{2}},$$

where $\tilde{N}_{i\cdot} = \sum_{j=1}^{C} N_{ij}$, $\tilde{N}_{\cdot j} = \sum_{i=1}^{K} N_{ij}$ and $\tilde{N} = \sum_{i=1}^{K}\sum_{j=1}^{C} N_{ij}$.

$\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}})$ is biased against clustering methods that do not allow unclustered nodes, especially when the number of noise nodes is large. On the other hand, $\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}})$ is biased against clustering methods that allow unclustered nodes. To balance the two, the weight Rand index was proposed as a weighted sum of the two indices [94]:

$$\text{Rand}(\mathcal{V}, \tilde{\mathcal{V}}) = \lambda \cdot \text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}}) + (1 - \lambda) \cdot \text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}}), \tag{2.3}$$

where $\lambda = |\mathcal{V}_{K+1} \cup \tilde{\mathcal{V}}_{C+1}|/N$.

We implemented the weighted Rand index by using the `adjusted_rand_score` function in the Python package `sklearn` [111].

2.C Robustness analysis of input parameters

Figure 2.10 shows the robustness analysis of BiTSC and its three variants: Spectral-kernel, BiTSC-1, and BiTSC-1-NC, against input parameters K_0 (the number of co-clusters in each subsampling run), ρ (the subsampling proportion), and τ (the kernel enhancement parameter), in the simulation setting (Section 2.3.1.1). We observe that BiTSC is robust to the choice of K_0 when K_0 is larger than K, the number of true co-clusters. BiTSC outperforms the three variants when ρ is larger than 0.7, and BiTSC is robust to the choice of τ. Hence, we set $\rho = 0.8$ and $\tau = (1, 1)$ as the default input parameters in BiTSC.

Interestingly, BiTSC-1-NC has weighted Rand indices that are extremely close to those of Spectral-kernel and invariant to ρ in Figure 2.10b. Spectral-kernel does not use subsampling, so it is expected to have a constant weighted Rand index invariant to ρ. However, BiTSC-1-NC uses subsampling, so its weighted Rand index should depend on ρ. We investigated this phenomenon and found that BiTSC-1-NC has weighted Rand indices not exactly the same but very close in values:

ρ	Weighted Rand index
0.3	0.7708656319227645
0.4	0.7706005330861144
0.5	0.7710055546541685
0.6	0.7711873136856517
0.7	0.7712402375720288
0.8	0.7713976529947345
0.9	0.7715965812884181

This result suggests that BiTSC-1-NC, where we only do subsampling on $\mathbf{V}$ and do not use node covariates but only rows in $\mathbf{V}$ to assign unsampled nodes in each subsampling run (Section 2.2.3), is very robust to ρ and highly similar to Spectral-kernel. This result is consistent with what we observed in Figure 2.2.

We further verified the robustness of BiTSC against τ in simulated networks with various sparsity levels (Figure 2.11). The performance of BiTSC and the three variants consistently peak near $\tau = 0$ and decrease as τ increases. This is within our expectation: in these simulated bipartite networks, the covariate information is consistent with the edge information, i.e., the true co-clusters feature both dense edges between species and similar covariates within species (Section 2.3.1.1); therefore, a smaller τ means better performance (Section 2.2.2). We also observe that the smaller p is, i.e., the lower the edge density, the more quickly the performance of BiTSC and the three variants drops as τ increases, demonstrating the importance of incorporating covariate information in the kernel enhancement step especially when edge density is low.

Interestingly, as is the most obvious in Figures 2.11c–h, as τ keeps increasing, eventually the performance of the three variant algorithms surpass the performance of BiTSC. This is because in the three variant algorithms, kernel enhancement is performed prior to subsampling (or in the case of spectral-kernel, no subsampling is done) (Section 2.2.3). Hence, utilizing covariate information from all of the nodes in the bipartite network can make up for a large τ to a certain extent.

2.D Choice of K_0 in the real data application

In our simulation study in Section 2.C, we observed that BiTSC is robust to K_0 values above K, the number of true co-clusters. However, in real data applications, we do not know K. Here we explain how we used the consensus distribution [88, 112] to choose $K_0 = 30$ in the real data application (Section 2.3.2).

The idea of using the consensus distribution to guide the choice of K_0 is the following: since the entries of the consensus matrix $\bar{\mathbf{M}}$ lie between 0 and 1, with each entry indicating how frequently two nodes are grouped together across multiple clustering results, a good K_0 should lead to many entries in the consensus matrix equal to 0 or 1 and few having fractional values in between. In other words, if we plot a histogram of the consensus matrix obtained for a good K_0, it would present a bimodal shape with two bins concentrated at 0 and 1. To better quantify this concentration of the consensus distribution, [88] used both the empirical cumulative distribution function (CDF) (Figure 2.12 (a)) of a consensus matrix entries and the area under CDF as the guidance to select K_0. In particular, the CDF corresponding to a good K_0 should present the shape with 2 increases (one increase at 0 and one increase at 1) and a horizontal line in between, and this bimodal shape would keep stable as K_0 past K (if known). In general, by inspecting the area under CDF curve (Figure 2.12b) to see where does the progression reach stable, users could decide an appropriate number of clusters as K_0.

Following the same sense, BiTSC provides an efficient way to select K_0 based on the idea of binary search (details in Algorithm 2). The rationale of Algorithm 2 is among a range of K_0's, we want to find the value of K_0 corresponding to which the CDF of the consensus matrix reaches stable at the earliest. In the real data analysis (Section 2.3.2), when we initialize $K_0\_\min = 10$ and $K_0\_\max = 50$ ($H = 48$, $\rho = 0.8$ and $\tau = (1, 1)$), Algorithm 2 selects 30 as the final input for K_0. To further verify the rationality in Algorithm 2, we applied BiTSC ($H = 48$, $\rho = 0.8$ and $\tau = (1, 1)$) to the fly-worm data, with K_0 ranging from 2 to 50. For each K_0 and its resulting consensus matrix, we plotted the empirical cumulative distribution function (CDF) of the matrix entries in Figure 2.12a. We also plotted the area under CDF curve as a function of K_0 in Figure 2.12b. From the result, we can observe that

distribution of entries in the consensus matrix corresponding to $K_0 = 30$ concentrates at 0 and 1, and the area under the CDF curve plateaus after this point.

2.E Computational time

Under Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0−157−generic x86_64), the computational time for the real data analysis (Section 2.3.2) is: BiTSC with a single subsampling run took 161 seconds using a single core, and OrthoClust took 258 seconds.

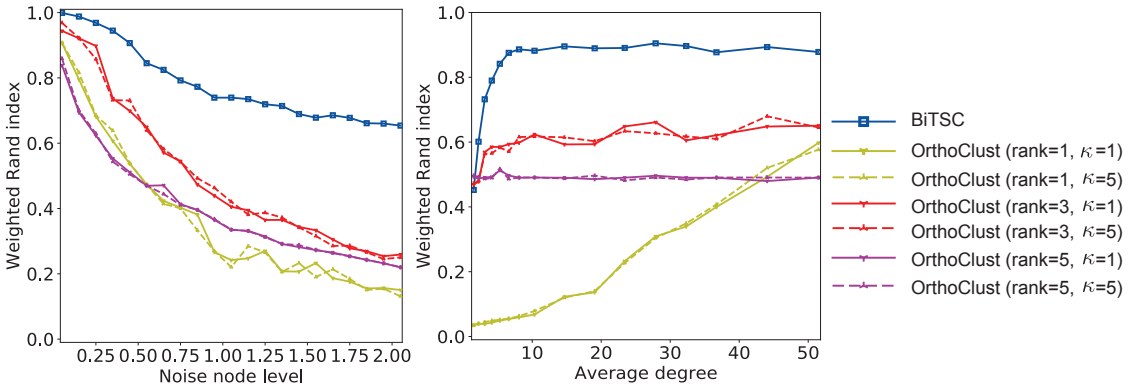

Figure 2.5: Performance of BiTSC vs. OrthoClust. This figure is an extension of Figure 2.2. The only difference is that in Figure 2.2, genes in identified co-clusters that only contain genes from one species are considered noise genes in the calculation of the weighted Rand index, just like unclustered genes. Here, we only consider unclustered genes as noise genes. BiTSC outperforms OrthoClust in a wide range of simulation settings.

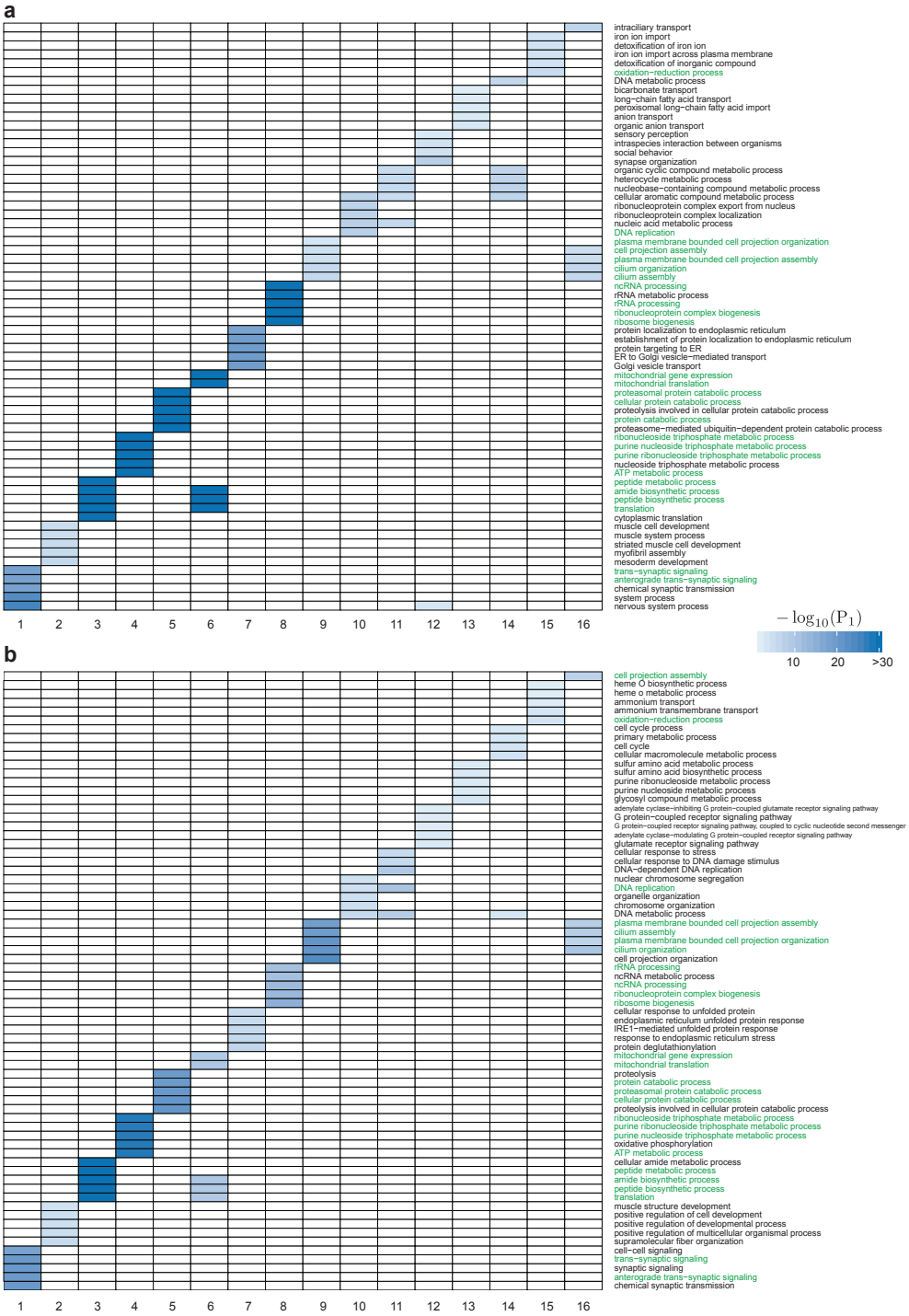

Figure 2.6: Top five enriched BP GO terms in each of the 16 fly-worm gene co-clusters identified by BiTSC (Section 2.3.2). (a) GO terms enriched in fly genes of each co-cluster. (b) GO terms enriched in worm genes of each co-cluster. The p-value (P_1 value) of each GO term is computed by the GO term enrichment test (Section 2.3.3.1). Smaller P_1 values are shown in darker colors to indicate stronger enrichment. For each co-cluster, the top enriched GO terms common to fly and worm are highlighted in green.

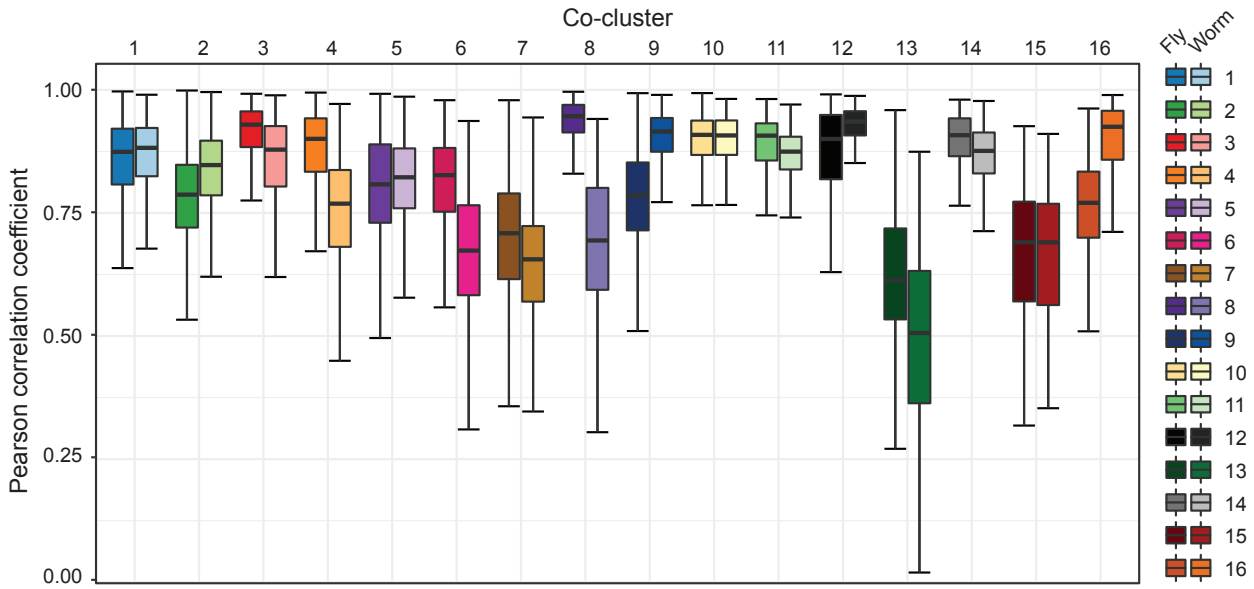

Figure 2.7: Boxplots of pairwise Pearson correlation coefficients in each species within each of the 16 fly-worm gene co-clusters identified by BiTSC (Section 2.3.2). In each co-cluster, the Pearson correlation coefficient is computed for every two genes of the same species based on their expression levels, i.e., covariate vectors.

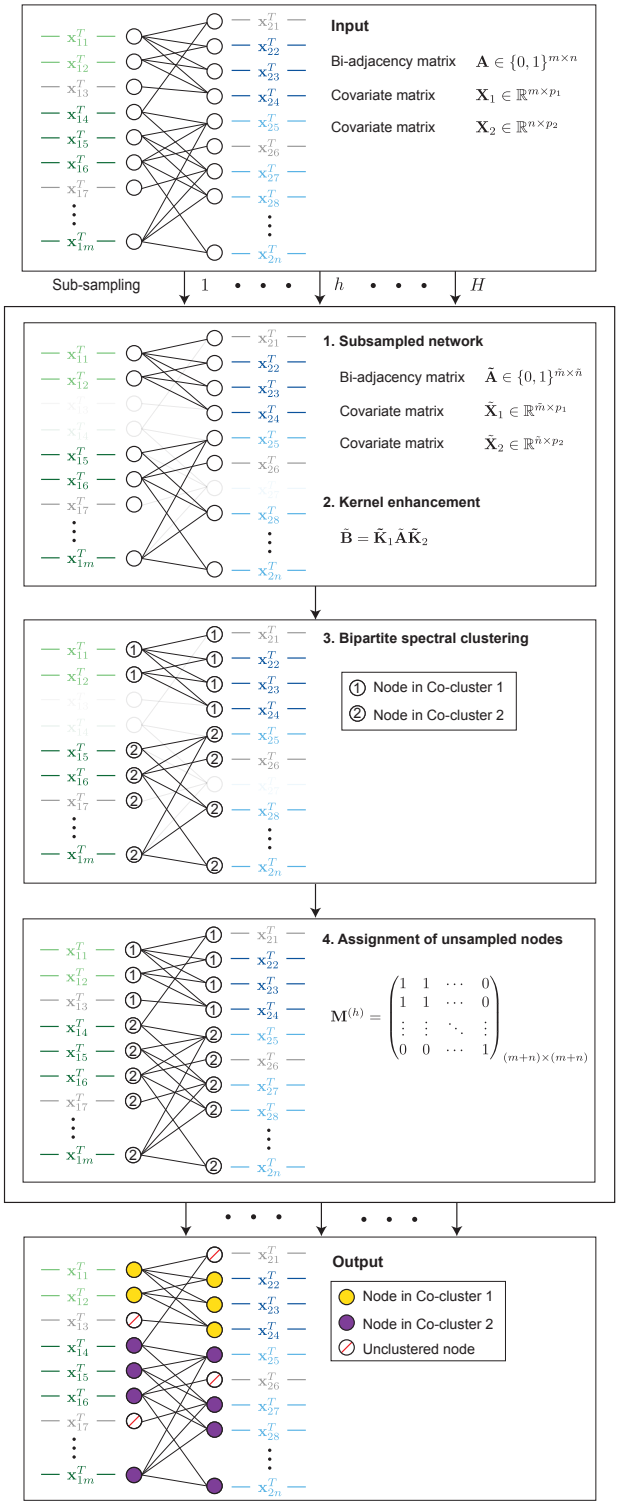

Figure 2.8: Workflow of BiTSC (Section 2.2.2).

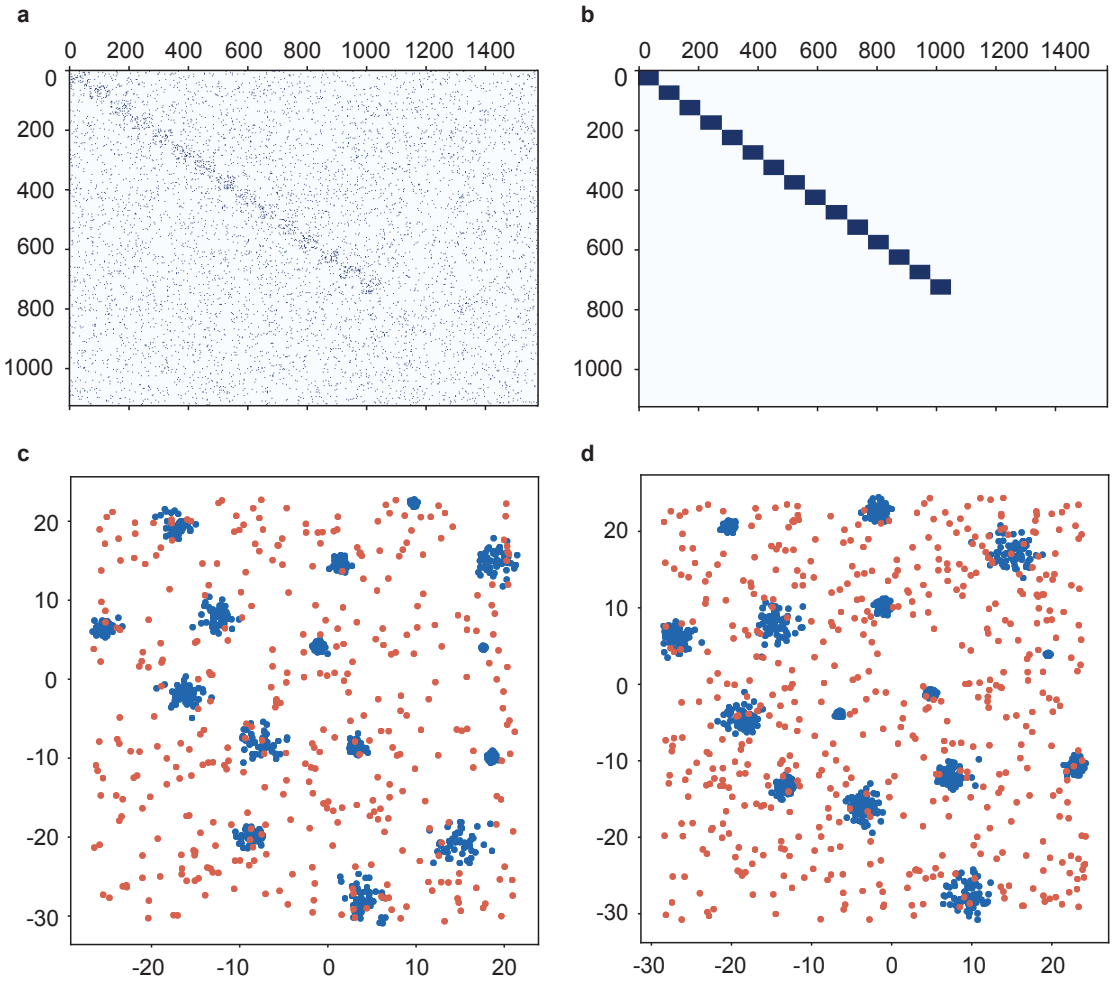

Figure 2.9: The simulated bipartite network described in Section 2.3.1.1. (a) The bi-adjacency matrix. (b) The true co-membership matrix. In both matrices, entries of ones are shown in blue colors. Nodes belonging to the same true co-cluster are ordered next to each other. (c) The node covariates on side 1. (d) The node covariates on side 2. Each point corresponds to one node, and the two axes represent the two dimensions of node covariates. Nodes in the 15 true co-clusters are marked in blue, and noise nodes not belonging to any co-clusters are marked in red.

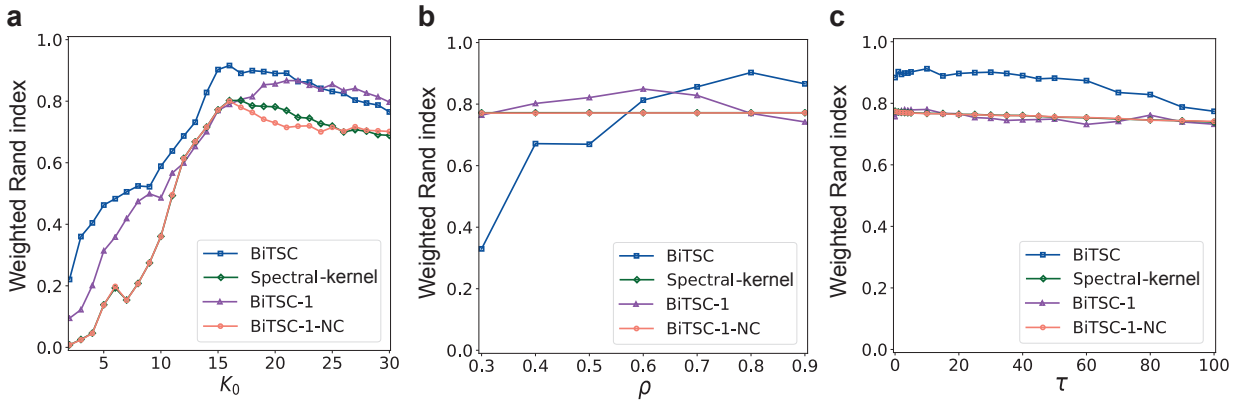

Figure 2.10: Clustering performance (weighted Rand index) of BiTSC and three variant algorithms (Section 2.2.3) as functions of (a) K_0, the number of clusters in each run, (b) the subsampling proportion ρ, and (c) the kernel enhancement parameter τ. A bipartite network was simulated as described in Section 2.3.1.1 using $K = 15$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, $p = 0.15$, and $q = p/5$. We set $H = 50$ and $\alpha = 0.7$ whenever applicable. For (a), we set $\rho = 0.8$ and $\tau = (1, 1)$. For (b), we set $K_0 = 15$ and $\tau = (1, 1)$. For (c), we set $K_0 = 15$ and $\rho = 0.8$.

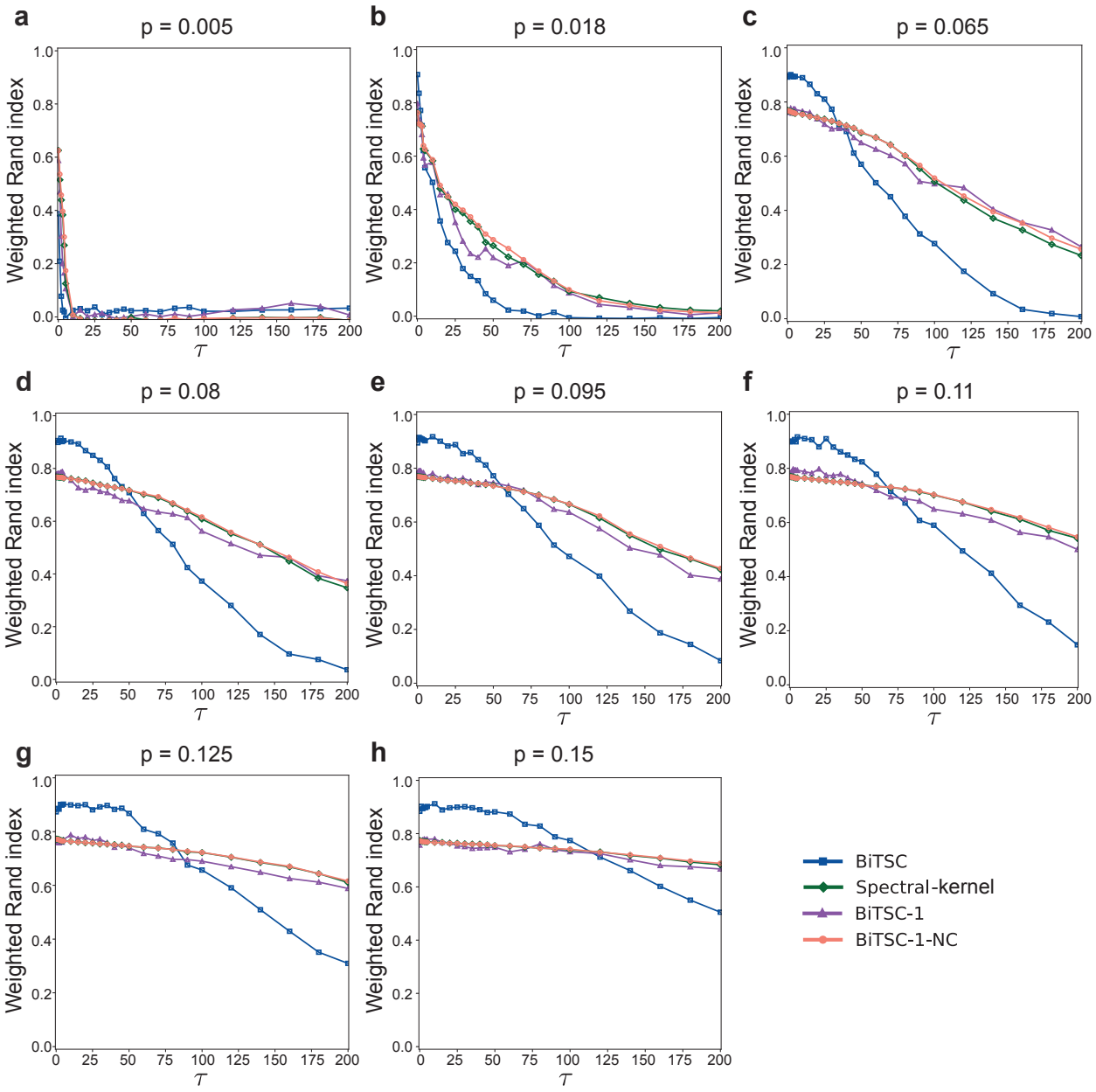

Figure 2.11: Clustering performance (weighted Rand index) of BiTSC and three variant algorithms (Section 2.2.3) as functions of τ ($\tau_1 = \tau_2$). For each of (a) through (h), a bipartite network was simulated following the procedure described in Section 2.3.1.1 using $K = 15$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, and $q = p/5$. For the input parameters of the algorithms, we set $H = 50$, $\rho = 0.8$, $K_0 = K = 15$, and $\alpha = 0.7$ whenever applicable. Note that Figure 2.10c is a truncated version of panel (h) here.

Table 2.2: Fly-worm gene co-clusters identified by OrthoClust (Section 2.3.2)

Co-cluster[1]	# of genes Fly	# of genes Worm	P_2[2]	P_3[3]	Average PPC[4] Fly	Average PPC Worm	Average PPC[5] Total
1	216	4	2.54e-03	1.25e-07	0.52758649	0.27495886	0.52752971
2	296	5	4.39e-02	1.14e-08	0.11182023	-0.09538319	0.11177321
3	999	3	9.57e-01	6.71e-04	0.51022172	-0.22759466	0.51021819
4	449	3	3.95e-02	6.07e-05	0.54076835	0.30057179	0.54076212
5	646	5	2.14e-01	4.88e-05	0.06707985	0.20971478	0.06708681
6	530	3	3.47e-01	9.99e-05	0.25106598	0.03598257	0.25106156
7	8	502	4.07e-03	1.34e-11	0.77520991	0.61604135	0.61608481
8	3	442	6.34e-02	5.79e-05	0.46570141	0.52308973	0.52308803
9	2	610	4.44e-01	2.86e-03	0.51949823	0.38322655	0.38322734
10	265	286	1.47e-08	9.72e-110	0.78480461	0.33437622	0.58870568
11	79	105	1.18e-08	1.10e-179	0.90816313	0.62631328	0.7687559
12	5	675	3.37e-03	7.17e-07	0.35234373	0.60067804	0.60066888
13	5	864	2.27e-02	1.54e-04	0.56978234	0.40720803	0.40721284
14	4	467	2.58e-04	2.78e-06	0.13064446	0.51414297	0.51412525

[1] With $\kappa = 1$, OrthoClust outputs a total of 24 clusters, among which 14 co-clusters have ≥ 2 genes in each species.

[2] p-value of the co-cluster based on the GO term overlap test (Section 2.3.3.2).

[3] p-value of the co-cluster based on the ortholog enrichment test (Section 2.3.3.3).

[4] The average pairwise Pearson correlation (PPC) is computed for all fly gene pairs in each co-cluster.

[5] The average PPC is computed for all gene pairs of the same species in each co-cluster. Values in this column are shown in Figure 2.3b-c.

Table 2.3: Fly-worm gene co-clusters identified by OrthoClust (Section 2.3.2)

Co-cluster[1]	# of genes Fly	# of genes Worm	P_2[2]	P_3[3]	Average PPC[4] Fly	Average PPC Worm	Average PPC[5] Total
1	412	4	3.14e-01	1.68e-06	0.1812038	-0.05888426	0.1811872
2	417	2	1.00e-00	1.33e-03	0.24701688	-0.08194976	0.2470133
3	216	16	1.11e-06	3.51e-16	0.52421706	0.29469462	0.5231781
4	412	3	4.16e-02	4.68e-05	0.115830253	-0.05725930	0.1158242
5	900	5	8.00e-01	3.03e-06	0.51815480	0.08545191	0.518146
6	650	2	2.93e-01	3.24e-03	0.63868879	0.41859354	0.6386879
7	4	692	3.35e-05	1.35e-05	-0.13076364	0.5729412	0.572928
8	4	734	2.15e-01	1.70e-05	0.021641929	0.473630719	0.4736222
9	250	361	3.40e-04	1.20e-151	0.78019168	0.33264337	0.5172098
10	6	1067	2.32e-01	3.94e-05	0.49751377	0.42596621	0.4259682
11	2	427	2.72e-01	1.40e-03	-0.64244048	0.10860390	0.1085944
12	78	99	1.23e-06	3.40e-176	0.92014421	0.70566047	0.8179273
13	2	517	1.00e-00	2.05e-03	-0.796179190	0.240137690	0.2401283
14	2	692	1.14e-01	2.23e-04	0.296111471	0.50715018	0.5071478

[1] With $\kappa = 3$, OrthoClust outputs a total of 22 clusters, among which 14 co-clusters have ≥ 2 genes in each species.

[2] p-value of the co-cluster based on the GO term overlap test (Section 2.3.3.2).

[3] p-value of the co-cluster based on the ortholog enrichment test (Section 2.3.3.3).

[4] The average pairwise Pearson correlation (PPC) is computed for all fly gene pairs in each co-cluster.

[5] The average PPC is computed for all gene pairs of the same species in each co-cluster.

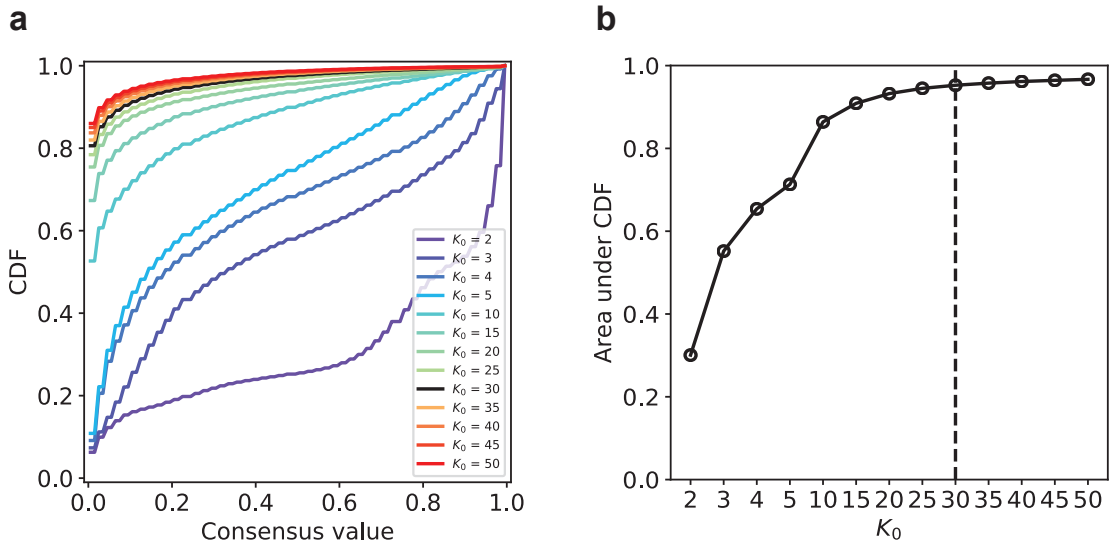

Figure 2.12: (a) The empirical CDFs of the entries of the consensus matrix for K_0 between 2 and 50 (in increments of 5 after K_0 passing 5) in the real data example (Section 2.3.2). (b) The area under CDF curve as a function of K_0. $K_0 = 30$ was chosen.

Algorithm 2 Automatic selection of K_0 in BiTSC

Result: K_0

Input: H, ρ, and τ for BiTSC; candidate K_0's in the interval $[a, b]$ (default $[5, 50]$)

Initialization:

$K_0\_\min = a$ and $K_0\_\max = b$, $K_0\_\text{mid} = \left\lceil \frac{K_0\_\min + K_0\_\max}{2} \right\rceil$

Calculate AUC_K_0_min and AUC_K_0_max as follows:

1. Run BiTSC with $H = H$, $\rho = \rho$, $\tau = \tau$, and $K_0 = K_0\_\min$ to obtain the consensus matrix;
2. Plot the empirical cumulative distribution function (CDF) of the entries of the consensus matrix;
3. Calculate the area under the CDF curve and define it as AUC_K_0_min;
4. Obtain AUC_K_0_max similarly based on Step 1-3;

while *True* **do**

  if $\frac{|\text{AUC\_}K_0\_\min - \text{AUC\_}K_0\_\max|}{\text{AUC\_}K_0\_\min} \leq 0.05$ **then**

    Return K_0_mid as the selected K_0;

  else

    K_0_mid $= \left\lceil \frac{K_0\_\min + K_0\_\max}{2} \right\rceil$, calculate AUC_$K_0$_mid based on Step 1-3

    if $|\ \text{AUC\_}K_0\_\min - \text{AUC\_}K_0\_\text{mid}\ | \geq |\ \text{AUC\_}K_0\_\text{mid} - \text{AUC\_}K_0\_\max\ |$ **then**

      $K_0\_\min \leftarrow K_0\_\text{mid}$; $K_0\_\max \leftarrow K_0\_\max$;

      AUC_K_0_min $\leftarrow$ AUC_K_0_mid; AUC_K_0_max $\leftarrow$ AUC_K_0_max;

    else

      $K_0\_\min \leftarrow K_0\_\min$; $K_0\_\max \leftarrow K_0\_\text{mid}$;

      AUC_K_0_min $\leftarrow$ AUC_K_0_min; AUC_K_0_max $\leftarrow$ AUC_K_0_mid;

    end

  end

end

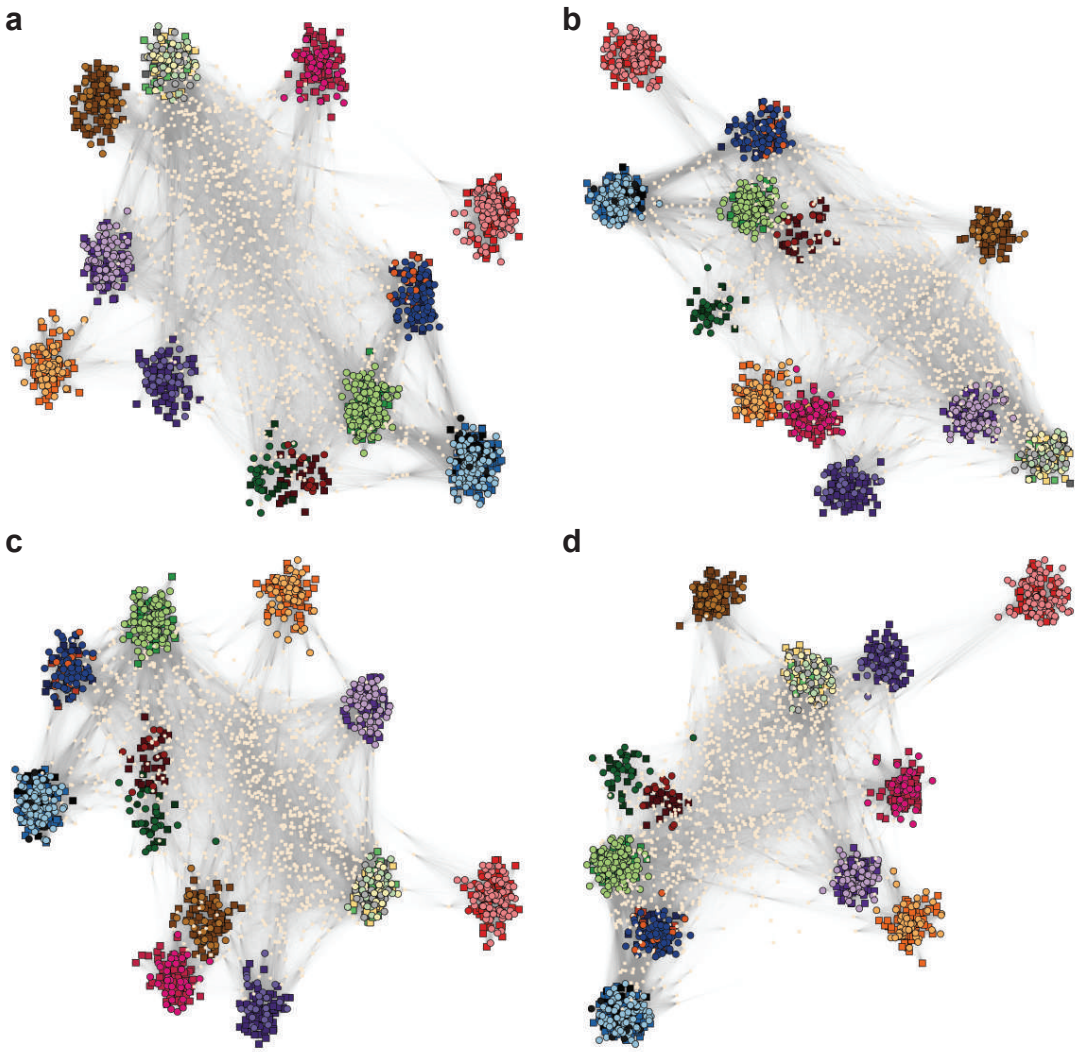

Figure 2.13: Robustness of the random selection of 1,000 unclustered genes in Figure 2.4. (a)-(d) Visualization of the 16 gene co-clusters found by BiTSC, same as in Figure 2.4 except that four different random seeds were used to sample the 1,000 unclustered genes.

Molecular Function (MF) or Celluar Component (CC) GO terms[1] closely related to featured BP GO terms[2] in each co-cluster				
Co-cluster	Fly gene[3]	Fly gene MF/CC GO terms	Worm gene[4]	Worm gene MF/CC GO terms
1			D1065.1	Cytosol; transporter activity; membrane
			D2021.2	Membrane; transferase activity
2	FBgn0052311	Cytoskeletal binding	C09B8.6	M band; striated muscle dense body
			D2092.4	Myofibril (part of: body wall muscle cell)
			F09F7.2	Cytoskeleton; myosin complex striated muscle myosin thick filament (part of: body wall muscle cell);
			F55C12.1	Cytoskeleton
			W03A5.7	Nucleus (part of: body wall muscle cell)
			W04D2.1	Striated muscle thin filament; striated muscle dense body (part of: body wall muscle cell)
3			C37A2.7	Ribosome
			C53H9.2	Cytosol
			K02B2.5	Cytosol; ribosome
			Y37E3.8	Ribosome; cytosolic large ribosomal subunit
4			C25H3.9	Mitochondrion
			F35D11.5	Mitochondrion
			F58F12.1	Mitochondrion; proton-transporting ATP synthase complex
			F59C6.5	Mitochondrion
			M176.3	Mitochondrion
			R53.4	Mitochondrion; proton-transporting ATP synthase complex
			T02H6.11	Oxidoreductase Activity; mitochondrion
			Y48E1B.5	Mitochondrion
			Y71H2AM.5	Mitochondrion
			Y94H6A.8	Oxidoreductase Activity; mitochondrion
6	FBgn0025336	Mitochondrion	F25H5.6	Mitochondrion
			M01F1.6	Mitochondrion
			T21B10.1	Mitochondrion
7	FBgn0011016	Endoplasmic reticulum	F44E7.9	Integral component of membrane
	FBgn0019925	Endomembrane system	F56A8.3	Integral component of membrane
	FBgn0021795	Endoplasmic reticulum	T04G9.5	Endoplasmic reticulum; endoplasmic reticulum membrane
	FBgn0028327	Endoplasmic reticulum	Y56A3A.21	Endoplasmic reticulum; integral component of membrane
	FBgn0030990	Endomembrane system	Y71F9AM.6	Endoplasmic reticulum; endoplasmic reticulum membrane
	FBgn0034500	Endomembrane system		
	FBgn0039303	Transport/localization; endomembrane system		
8	FBgn0010520	Nucleolus; small-subunit processome	C37H5.5	Nucleus; nucleolus; chromatin binding
	FBgn0030067	Nucleolus; preribosome, large subunit precursor	F44G4.1	Nucleolus; preribosome, large subunit precursor
	FBgn0030504	Nucleolus	JC8.2	Nucleus
	FBgn0031361	Small ribosomal subunit rRNA binding; nucleolus; small-subunit processome	T05H4.10	Nucleus
	FBgn0033169	RNA binding	T23D8.3	Nucleus; preribosome, small subunit precursor
	FBgn0034528	Nucleic acid binding	T23G7.3	Nucleic acid binding
	FBgn0034734	Nucleolus; small-subunit processome	Y54H5A.1	Nucleus
	FBgn0037489	Chromatin binding; nucleus	Y73E7A.2	Nucleus; nucleolus
	FBgn0037561	ATP binding; RNA binding; nucleolus	ZK512.2	Nucleotide binding; nucleic acid binding; RNA binding; ATP binding
	FBgn0260456	mRNA binding; RNA binding; ribonucleoprotein complex		
9	FBgn0032800	Cell projection; cilium	C27F2.1	Cell projection; cytoskeleton; cilium
			C52B9.3	Cytoplasm; cytoskeletal protein binding
			F56D12.4	Cytoplasm
			K07E8.6	Cytoplasm
			Y97E10AR.2	Plasma membrane
			ZK643.1	Cytoplasm (part of: body wall muscle cell)
10			C32E8.5	Nucleus; mRNA binding
11	FBgn0013548	Nucleus	C09H10.6	Nucleus
12			C17H11.1	G protein-coupled receptor activity
13	FBgn0034605	Transferase activity; UDP-glycosyltransferase activity; intracellular membrane-bounded organelle	AC3.7	UDP-glycosyltransferase activity; transferase activity
	FBgn0040252	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	C10H11.3	Glucuronosyltransferase activity; UDP-glycosyltransferase activity;transferase activity
	FBgn0040255	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	C32C4.7	UDP-glycosyltransferase activity; transferase activity
	FBgn0040257	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	H23N18.1	UDP-glycosyltransferase activity; transferase activity
			T19H12.10	UDP-glycosyltransferase activity; transferase activity
			T19H12.11	UDP-glycosyltransferase activity; transferase activity
			Y49C4A.8	UDP-glycosyltransferase activity; transferase activity
14	FBgn0032169	Nucleic acid binding	C25A1.4	Nucleic acid binding; RNA binding
16			D1009.5	Cell projection; cytoskeleton

1. GO Terms Information is collected from FlyBase (flybase.org) and WormBase (wormbase.org). The Terms are based on experimental evidence or predictions or assertions.
2. BP GO terms that are highly enriched in both species in a given co-cluster.
3. Fly genes (non-BP-GO-annotated) with MF/CC GO terms that are closely related to featured BP GO terms in the co-cluster. Please see Supplementary Materials for the featured BP GO terms that the MF/CC GO terms in this table are closely related to.
4. Worm genes (non-BP-GO-annotated) with MF/CC GO terms that are closely related to featured BP GO terms in the co-cluster.

Figure 2.14: MF and CC GO terms of the genes without BP GO terms in the 16 gene co-clusters identified by BiTSC (Section 2.3.2).

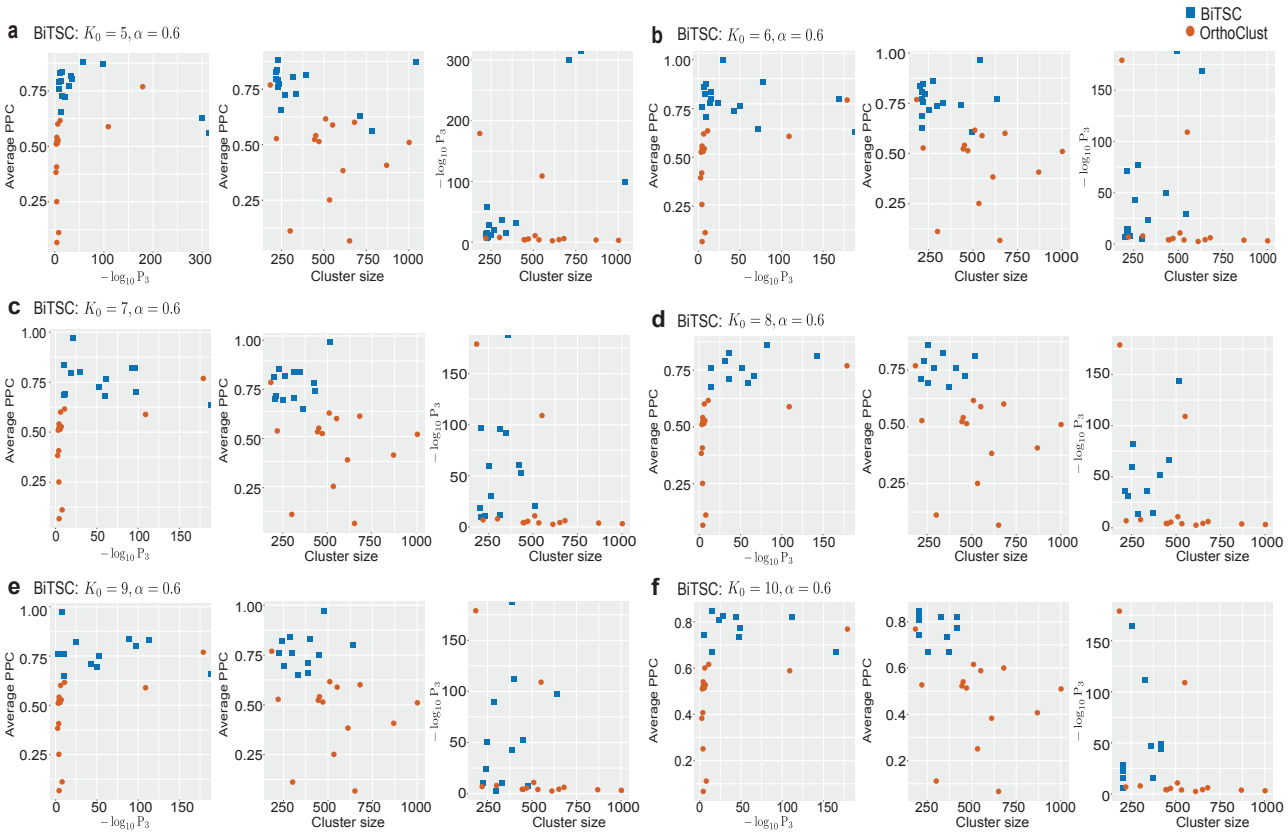

Figure 2.15: Comparison of BiTSC and OrthoClust in terms of their identified co-clusters (Section 2.3.2.1). This figure is an extension of Figure 2.3. After running the OrthoClust algorithm, we kept the 14 co-clusters with at least two genes in each species (Section 2.3.2.1; Table 2.2). The cluster sizes are between 184 and 1002. For BiTSC, we set $H = 100$, $\rho = 0.8$, and $\tau = (1, 1)$ as we did in Section 2.3.2.1. Then we varied K_0 between 5 and 10 and set $\alpha = 0.6$ so that BiTSC outputs co-clusters with sizes comparable to those identified by OrthoClust. For a given choice of K_0 and α, we kept the co-clusters that have at least 10 genes from each species and at least 200 genes total. Here, This figure shows that controlling for cluster size, the co-clusters identified by BiTSC exhibit higher gene expression similarity (second column of each panel) and greater enrichment of orthologs (third column of each panel).

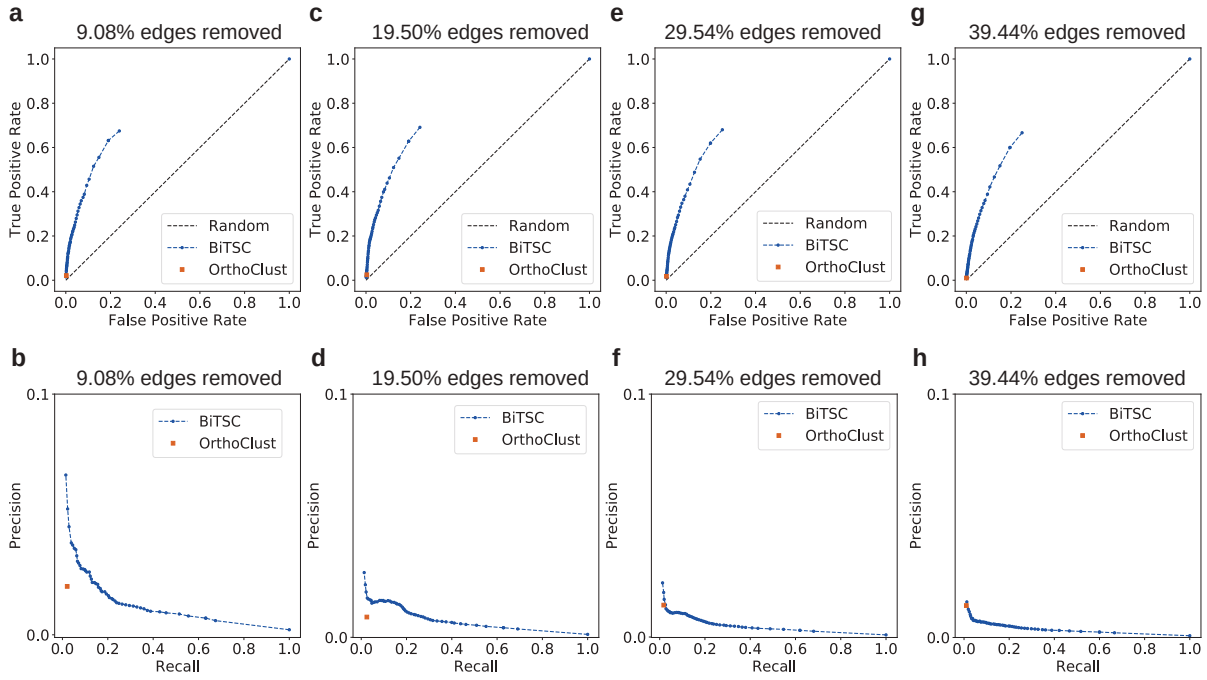

Figure 2.16: Comparison of BiTSC and OrthoClust in terms of link prediction performance on the fly-worm data set (Section 2.3.2). From the 5414 fly genes and the 5731 worm genes in the fly-worm data set, we sampled a proportion of fly genes and worm genes each, removed all edges among the sampled genes from the original network, and run BiTSC on the remaining network with $H = 100, \rho = 0.8, \tau = (1, 1)$, and $K_0 = 30$ (the same parameters as we used in Section 2.3.2) as well as OrthoClust. The number of removed edges divided by the number of edges in the original network $(10,975)$ is shown in the title of each plot. For BiTSC, we performed link prediction by thresholding the consensus matrix with cutoffs between 0 and 1. For OrthoClust, we performed link prediction by connecting every pair of genes in the same identified co-cluster. Denote the set of sampled fly genes as F and the set of sampled worm genes as W. Let $F' \subset F$ be the set of fly genes in F that are orthologous to at least one gene in W, and similarly, let $W' \subset W$ be the set of worm genes in W that are orthologous to at least one gene in F. That is, we removed the "isolated" genes in F and W to obtain F' and W'. Then, the link prediction performance of BiTSC and OrthoClust was evaluated based on the set of all gene pairs between F' and W' and the ROC curves and precision-recall curves were obtained accordingly. We chose to evaluate BiTSC and OrthoClust based on all gene pairs between F' and W' instead of F and W due to the low edge density in the original network. If we considered all gene pairs between F and W, then the negatives would overwhelm the positives in the truth.

CHAPTER 3

A Survey of Mixed Membership SBMs and Bipartite SBMs

3.1 Introduction

The SBM [1, 113] serves as a benchmark tool in studying community detection problems, where it assumes a process of edge formulation. A lot of methods have been developed based on SBM to recover the latent block structure and node communities (see [1] for a review). In a network of N nodes divided into K disjoint groups, the SBM assumes that each node exclusively belongs to one cluster, and edges are generated independently with probabilities only depending on the clusters to which the corresponding two nodes belong. The flexibility of the SBM allows researchers to further generalize the SBM to more advanced variants. In the last two decades, the development in technology has rapidly increased the availability of network data with numerous structural features. For example, some networks are accompanied by the metadata or node covariates, such as the gene expression levels in a gene regulatory network or the personal age information in a social network. [38, 84] extend the statistical inference methods for the SBM to incorporate such metadata. A lot of networks contain noisy nodes or outliers [37, 114, 115], i.e., there may exist nodes that do not belong to any clusters and connect with the other nodes in an arbitrary way. To tackle this issue, Cai et al. [114] propose a generalized SBM with outliers and formulate the community detection by a convex optimization problem; Qian et al. [37] develop a degree-corrected version of this method. Besides, in real-world data, noises could also exist among edges [115]. For example, when studying the gene regulatory networks, the known interaction (an observed

edge) between a transcription factor (a protein) and a DNA-binding site (a gene) is measured based on the physical binding between them [116]. This implies that an observed edge may not represent the actual regulation (as the detected interaction may only be caused by sequencing errors). In contrast, an unobserved edge (zero in the adjacency matrix) does not necessarily imply there is certainly no regulation. In general, the adjacency matrices of empirical network data could contain many false zeros (i.e., missing data) due to measurement errors. Given this, several researches [24, 36, 48] have accounted for the sparsity in their stochastic block modeling.

Compared to the above methods, which all consider partitioning network nodes into disjoint groups, there is a line of research concentrating on detecting the mixed membership communities (see [117, 118] for review). The overlap in node membership is a very common characteristic of many real-world networks. For example, a professor doing interdisciplinary researches could have collaborations in multiple research areas. A gene or protein could participate in multiple signaling pathways or functional complexes [7]. For this reason, extensions of the SBM to mixed membership scenarios have received a lot of attention recently. In this chapter, we give a review of the community detection methods on the mixed membership SBM.

So far, a community we discussed in either the non-overlapping case or the overlapping case refers to a group within which the edge density is relatively higher than the edge density between groups. Nevertheless, the definition of a community needs to be further generalized when the network exhibits a disassortative community structure such as the bipartite network [119]. A bipartite network, as a special case of a network, containing two types of nodes, where edges are only allowed to connect nodes of different types. In Chapter 2, we have studied a bipartite ortholog network by connecting orthologous genes in two species; clustering on such a network could lead to evolutionarily conserved gene clusters [30, 108].

In this chapter, we mainly focus on the community detection methods under the SBM and its three types of extensions: (1) degree-corrected SBM; (2) mixed membership/overlapping SBM; (3) bipartite SBM. As far as we know, is no research has integrated all three of the

real-world network features in one model. We give a literature review on the existing methods under (degree-corrected) mixed membership SBM and (degree-corrected) bipartite SBM. The main questions we wish to answer are the following: What are the challenges in introducing degree-correction in the mixed membership SBM? What are the challenges in extending the degree-corrected mixed membership SBM to the bipartite network?

Outline. The rest of the chapter is organized as follows. In Section 3.2, we review a detailed definition of the SBM. In Section 3.3, we review a detailed definition of the DCSBM. In Section 3.4, we present an overview of the mixed/overlapping SBM and relevant community detection methods. In Section 3.5, we review a detailed definition of the biSBM and relevant community detection methods. In Section 3.6, we generally summarize the challenges in benchmarking these methods. And we close this chapter by discussing future research directions.

Notation. We use bold uppercase letters to denote matrices. We use bold lowercase letters to denote vectors. We use unbolded letters to denote entries in a matrix or a vector. $\mathbf{1}$ denotes an all-ones vector. For a vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p$ denotes its l_p-norm. For a matrix $\mathbf{X}$, $\|\mathbf{X}\|_F$ denotes its Frobenius norm. Denote $[N] := \{1, \ldots, N\}$.

3.2 Simple SBM

In a network of N nodes, the $N \times N$ adjacency matrix $\mathbf{A}$ is defined by $A_{ij} = 1$ if there is an edge between i and j and 0 otherwise, and edges correspond to independent Bernoulli random variables. Here we focus on the undirected networks without self-edges; then $\mathbf{A}$ is symmetric and $A_{ii} = 0$. Suppose the network contains K groups, we let $\boldsymbol{\pi}_i = (\pi_{i1}, \cdots, \pi_{iK})^\mathsf{T}$ be a K-dimensional membership indicator of node i, where only one element equals one and others equal to zero; that is, $\pi_{ik} = 1$ if node i belongs to group k. The SBM assumes that the edge generation probability between two nodes is a function of the communities they belong to. Let $\boldsymbol{\Omega}$ be the $K \times K$ symmetric matrix, representing the Bernoulli edge generation

probabilities between different communities. Then $A_{ij} \sim \text{Ber}(\Omega_{gh})$ if node i belongs to group g and node j belongs to group h; or equivalently, $A_{ij} \sim \text{Ber}(\boldsymbol{\pi}_i^\mathsf{T} \boldsymbol{\Omega} \boldsymbol{\pi}_j)$. If we integrate all the node membership vectors into an $N \times K$ matrix $\mathbf{\Pi}$, then we can write $\mathbb{E}[\mathbf{A}] = \mathbf{\Pi}\mathbf{\Omega}\mathbf{\Pi}^\mathsf{T}$. Note that the non-zero elements in column k of $\mathbf{\Pi}$ correspond to the nodes in community k. And nodes within the same groups are statistically identical. As the SBM explicitly assumes a parametric probability distribution of networks, community detection under the SBM becomes estimation of parameters $\mathbf{\Pi}$ and $\mathbf{\Omega}$. Compared to many other community detection schemes, SBM-based community detection methods are convenient to study their statistical properties and asymptotic consistency under certain conditions.

In the connectivity matrix $\mathbf{\Omega}$, the diagonal elements Ω_{kk} represent the edge probabilities within blocks, while the off-diagonal elements Ω_{gh} represent the edge probabilities between blocks. $\forall\, k, g, h = 1, \cdots, K$, $g \neq h$, when $\Omega_{kk} = \Omega_{gh}$, it recovers the Erdős-Rényi random graph; when $\Omega_{kk} > \Omega_{gh}$, it recovers the assortative block structure, where edges within blocks are more dense than between blocks; when $\Omega_{kk} < \Omega_{gh}$, it recovers the disassortative structure, where edges between blocks more dense than within blocks. Note that there is a special case of the disassortative mixing structure when $\Omega_{kk} = 0$, $\forall k$. In this case, edges only connect nodes from different blocks, which is commonly used to study the bipartite structure [60] (see details in Section 3.5).

3.3 Degree-correction SBM

In network analysis, the node's degree stands for the total number of edges connected to it. In other words, the degree of node i is equal to the sum of the corresponding row or column of the adjacency matrix. The simple version of SBM (Section 3.2) assumes that within the same group, nodes are statistically equivalent, so that node degree distribution is homogeneous. However, empirical networks often have nodes with varying degrees, as there exist "popular" (high-degree, so-called hub) nodes and "unpopular" (low-degree) nodes. This common phenomenon makes the node degree distribution frequently exhibit the heavy tail or right-skewed property [37, 47, 120]. In particular, it has been claimed that the node

degrees in some networks follow approximately a power-law distribution [121] (Section 4.C). Besides, previous research [20, 24, 35] shows that in the network with heavy-tailed distribution, the classical clustering algorithm (e.g., spectral clustering) often fails to recover the block structure. That is, these methods tend to group nodes with similar degrees.

Karrer et al. [35] proposed the degree corrected stochastic block model (DCSBM) to accommodate the node degree heterogeneity while preserving the overall block structure. The DCSBM introduces the node-specific degree correction parameters θ_i, which are absorbed into the edge generation probability. In detail, the number of expected edges between node i and j follows a Poisson distribution $A_{ij} \sim \text{Poi}(\theta_i \theta_j \boldsymbol{\pi}_i^{\mathsf{T}} \boldsymbol{\Omega} \boldsymbol{\pi}_j)$. Note here θ_i, θ_j and Ω are arbitrary up to multiplicative constants, which can be fixed by imposing the following constraints on θ_i:

$$\sum_{i=1}^{N} \theta_i \boldsymbol{\pi}_i = \sum_{i=1}^{N} \boldsymbol{\pi}_i \tag{3.1}$$

That is, the sum of node degree parameters within a community equal to the number of nodes in that community. The authors [35] use a Kernighan-Lin-like procedure to iteratively maximize the likelihood function by treating true community labels $\boldsymbol{\pi}_i$ and degree parameters θ_i as fixed parameters. And they test the efficacy of DCSBM both on synthetic and empirical networks with heterogeneous node degrees. In addition, DCSBM has also been studied under other various scenarios, including but not limited to pseudo-likelihood [24], spectral clustering [26, 27, 46] and modularity based approaches [25, 45].

Note that in DCSBM, multi-edges (i.e., $A_{ij} > 1$) are allowed. Even though many real-world networks only have single edges, many researchers have discussed the advantage by assuming edges are Poisson-distributed in SBM [25, 35, 122]. The inclusion of multi-edges usually makes the computations simpler and easier without affecting the actual outcome significantly. Specifically, a Bernoulli random variable with a small mean is well approximated by a Poisson random variable having the same mean (see detailed discussion in [122]). And most real networks are sparse, so one can expect the approximation to work well. Besides, Zhao et al. [25] found no significant difference in the solutions produced by the model that uses the Poisson distribution instead of the Bernoulli distribution used in the original model.

3.4 Mixed membership SBM

In most real-world networks, a node could belong to multiple groups, or in other words, have mixed or overlapping communities [123]. For example, in a gene regulatory network, a gene could participate in more than one signaling pathway or biological process. In a friendship network, a person could be a member of several social clubs. Mixed membership community detection has been an increasing research area during the last decade, and several extensions of the SBM [41, 117] have been proposed in the literature to model the mixed membership scenario. These models characterize each node by a membership vector indicating its weight of belonging to different communities, making the clustering results more interpretable than those in the non-overlapping case. Compared to the node membership $\boldsymbol{\pi}_i$ in Section 3.2, $\boldsymbol{\pi}_i$ in the mixed membership case is a soft membership vector which is not constrained always to have only one entry equal to one and all the others equal to zero. In this section, we first introduce several famous mixed membership SBMs: MMSBM, OSBM, OCCAM, LCM and we discuss relevant methods developed for identifying mixed membership communities.

The first introduced and well-studied generative model for mixed membership communities is the mixed membership stochastic block model (MMSBM) [48]. Under this model, all node membership vectors are drawn independently from a Dirichlet prior, with π_{ik} representing the probability of node i belong to group k. In detail, the complete generative process is stated as follow:

$$\boldsymbol{\pi}_i \sim \text{Dirichlet}(\boldsymbol{\alpha}), \quad i \in [N], \quad \boldsymbol{\alpha} \in \mathbb{R}_+^K \tag{3.2}$$

$$\boldsymbol{z}_{i\to j} \sim \text{Multinomial}(\boldsymbol{\pi}_i), \quad \boldsymbol{z}_{i\leftarrow j} \sim \text{Multinomial}(\boldsymbol{\pi}_j) \tag{3.3}$$

$$A_{ij} \sim \text{Bernoulli}(\boldsymbol{z}_{i\to j}{}^{\mathsf{T}} \boldsymbol{\Omega} \boldsymbol{z}_{i\leftarrow j}) \tag{3.4}$$

where $\boldsymbol{z}_{i\to j}$ and $\boldsymbol{z}_{i\leftarrow j}$ are the K-dimensional edge-specific membership indicators, respectively denoting the group membership of node i and node j when they interact with each other. Note that a node could belong to different groups when interacting with different nodes. The parameter $\alpha_0 = \sum_k \alpha_k$ controls the level of overlap; the larger the α_0, the greater the overlap.

Alternatively, the edge probability in (3.4) can be replaced by a simplified version:

$$A_{ij} \sim \text{Bernoulli}(\boldsymbol{\pi}_i^\mathsf{T} \boldsymbol{\Omega} \boldsymbol{\pi}_j) \tag{3.5}$$

which is more popularly adopted by other literature. Note that, under the MMSBM, the edge probability between a pair of nodes does not necessarily increase as the number of communities that they share increases. Also, the MMSBM has been applied to various scenarios. For example, Huang et al. [124] applied it to the heterogeneous networks. In this case, a network could consist of different types of nodes and interactions among types. Under their model, MMSBM is a special case containing only one single type of node. Kao et al. [125] propose a hybrid MMSBM (HMMB) by capturing mixed-membership, power-law distributed node degrees, and sparsity into one hybrid model and infer the parameters based on the MCMC inference procedure. Besides, each degree correction parameter θ_i in their model is viewed as a random variable following a prior power-law distribution.

The overlapping stochastic block model (OSBM) [49] is another well-known extension of SBM to the overlapping case. Under the OSBM, each node is assigned with a $\{0,1\}$-vector of length K; that is, $\boldsymbol{\pi}_i \in \{0,1\}^K$. And elements π_{ik} are independent Bernoulli random variables. In particular, the edge probabilities are given by:

$$A_{ij} \sim \text{Bernoulli}(\sigma(\boldsymbol{\pi}_i^\mathsf{T} \mathbf{W} \boldsymbol{\pi}_j + \boldsymbol{\pi}_i^\mathsf{T} \mathbf{u} + \mathbf{v}^\mathsf{T} \boldsymbol{\pi}_j + W^*)) \tag{3.6}$$

where $\sigma(\cdot)$ is the sigmoid function. $\mathbf{W} \in \mathbb{R}_+^{K \times K}$, $\mathbf{u}, \mathbf{v} \in \mathbb{R}_+^K$, and $W^* \in \mathbb{R}$. In this sense, two nodes are more likely to be connected if they have more than one group in common. The author uses an EM-like algorithm to infer the parameters of the model.

The overlapping continuous community assignment model (OCCAM) [126] is a degree-corrected mixed membership SBM. The formulation is slightly different than MMSBM and as follows:

$$A_{ij} \sim \text{Bernoulli}(\rho_N \theta_i \theta_j \boldsymbol{\pi}_i^\mathsf{T} \boldsymbol{\Omega} \boldsymbol{\pi}_j) \tag{3.7}$$

where ρ_N is the scaling factor for theoretical analysis, and $\rho_N \to 0$ as $N \to \infty$. Also, the degree parameters are constrained by (3.1) and $\forall\, i,\ \|\boldsymbol{\pi}_i\|_2 = 1$ for identifiability.

So far, the models that we discuss focus on the mixed membership communities for nodes. On the other hand, Ball et al. [50] have proposed the link communities model (LCM) to study the mixed membership communities by grouping edges instead of nodes; it is a relaxation of the DCSBM [35]. This model has intuitive interpretation when applied to real-world networks, especially social networks. For example, the connections could represent co-workers, classmates, family members, and friendships, etc. Under the LCM, the existence of mixed membership communities is caused by the existence of multiple types of edge. Unlike the MMSBM and OSBM, under which the edge generation probabilities are determined based on the node memberships, in LCM, the mixed membership node memberships and the mixed membership edge memberships are determined simultaneously. Mathematically, in a network with N nodes and K communities, each node i is assigned by a propensity vector $\boldsymbol{\eta}_i$ with each element η_{ik} representing the propensity of node i connected by edges of a group k, and the number of edges of a group k between node i and j is Poisson-distributed with mean value at $\eta_{ik}\eta_{jk}$. And the number of edges of all groups between node i and j are generated by:

$$A_{ij} \sim \text{Poisson}(\sum_k \eta_{ik}\eta_{jk}) \tag{3.8}$$

The author uses an EM algorithm to maximize the log-likelihood, and the estimated membership vectors contain non-negative values.

Above, we have introduced several types of mixed membership SBMs. Next, we will discuss the methods developed for detecting mixed membership communities, which are mainly divided into two groups: model-based methods (based on variants of SBMs) and algorithm-based methods. The model-based methods focus on fitting a generative model to an observed network and estimating the parameters by maximizing the (approximated) log-likelihood. For example, under the MMSBM, few studies [48, 127] have used the variational EM algorithms to maximize the approximated posterior distributions of latent variables. On the other hand, several algorithmic-based methods for detecting mixed membership communities have been developed, including non-negative matrix factorization (NMF) based approaches [51, 54, 128], spectral clustering based approaches [53, 56–59, 129], and tensor-based approaches [52].

NMF is a dimension reduction technique, and it can be used for clustering. In general, given a data matrix $\mathbf{V} \in \mathbb{R}_{+}^{M \times N}$ with N data points and M features, NMF solves the optimization problem: $\arg\min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0} \|\mathbf{V} - \mathbf{W}\mathbf{H}^{\mathsf{T}}\|_F^2$, where $\mathbf{W} \geq \mathbf{0}$ and $\mathbf{H} \geq \mathbf{0}$. In terms of the network clustering problem, NMF has been proven that it is basically equivalent to Kernel K-means clustering and Laplacian spectral clustering [130, 131]. In particular, the objective function reduces to a symmetric factorization: $\arg\min_{\mathbf{H} \geq 0} \|\mathbf{A} - \mathbf{H}\mathbf{H}^{\mathsf{T}}\|_F^2$ which is called symmetric NMF (SNMF). SNMF has been widely studied in the context of mixed membership communities. For example, Psorakis et al. [51] proposed a Bayesian variant of SNMF (BNMF). Specifically, each entry H_{ik} here represents the participation strength of node i into community k and normalizing each row in $\mathbf{H}$ gives each node a soft membership vector. Also, under the MMSBM, GeoNMF [54] algorithm uses the SNMF to estimate the node memberships consistently under specific conditions. Note that this method assumes $\mathbf{\Omega}$ is a diagonal matrix to guarantee the unique solution of SNMF.

The spectral algorithm [16] is the most well-studied technique for detecting both non-overlapping and overlapping communities due to its computational scalability and theoretical properties [27, 132–135]. The general procedures of a spectral algorithm are summarized as follow:

1. Calculate the first K leading eigenvectors of the adjacency matrix $\mathbf{A}$ or the Laplacian matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is the diagonal matrix with $D_{ii} = \sum_j A_{ij}$
2. Integrate the eigenvectors by columns into a $N \times K$ matrix, denoted by $\mathbf{U}$
3. Apply a clustering algorithm on the rows of $\mathbf{U}$, and find K vectors representing the centers of K clusters in the low dimensional space
4. Project each row of $\mathbf{U}$ onto the K representative cluster centers, and determine the membership vector for the corresponding node

It may include the regularization [27] of $\mathbf{L}$ in Step 1 and normalization [15, 24] of $\mathbf{U}$ in Step 2 for adapting the network sparsity [136] and node degree heterogeneity, respectively.

While K-means clustering in Step 4 is usually applied in the non-overlapping case, it is not expected to work in the overlapping case due to the existence of mixed memberships. For this reason, researchers have tried several modifications to replace K-means clustering. For example, Fuzzy C-Means (FCM) [137] is the first spectral algorithm to find mixed membership communities, which replaces the K-means with a heuristic fuzzy clustering algorithm.

Recently, approaches have been proposed under the framework of the SBM, and the theoretical consistency of parameter estimation is studied given specific assumptions. In summary, these methods are built on the rationale that each latent node position in the low-dimensional space (each row in $\mathbf{U}$) is a mixture combination of pure nodes in K communities (we call a node i "pure" if $\pi_{ik} = 1$ for some k and all other elements are 0). The OCCAM by Zhang et al. [58] replaces K-means by K-medians, and the node membership vectors in this model have unit l_2 norm: $\|\boldsymbol{\pi}_i\|_2 = 1$. Their method is able to recover the representative cluster centers corresponding to the pure nodes, given the assumption that each community contains a proportion of pure nodes. Based on the OCCAM, Arroyo et al. [59] estimate a non-orthogonal sparse eigenbasis of the principal subspace of $\mathbf{A}$ instead of directly calculate the eigenvectors of $\mathbf{A}$, which lead to computational efficiency but in the sacrifice of identifiability. The Spectral Algorithm with Additive Clustering (SAAC) [53] models each node membership vector as a binary vector, $\boldsymbol{\pi}_i \in \{0, 1\}^K$; this algorithm is based on the property that each row of node i in $\mathbf{U}$ is the sum of rows of pure nodes that share the mixed membership communities with node i. Anandkumar et al. [52] propose a tensor-based spectral decomposition approach; this method is built on the MMSBM and has been shown theoretically consistent, but the computational complexity prohibits its implementation on large-scale network data.

Besides, three papers [55–57] focus on studying an interesting property about the eigenspace of $\mathbf{A}$, that is: the rows of $\mathbf{U}$ form a low-dimensional simplex geometrical structure [57]. In particular, the row in $\mathbf{U}$ which corresponds to a pure node will fall on one of the K corners (vertices) of the simplex, and the row corresponding to a mixed node will fall in the interior of the simplex. The corners can be found by running a corner-finding (vertex-hunting) method such as the successive projection algorithm (SPA) on $\mathbf{U}$ [138, 139].

After finding the corners, these methods will reconstruct all node memberships by projecting the rows of $\mathbf{U}$ onto those of the pure nodes. As proved by Jin et al. [57], including the node heterogeneity will complicate the situation, and they normalize each row of $\mathbf{U}$ based on the first leading eigenvector to retain the simplex structure.

MMSBM is not identifiable, and identification constraints have also been discussed among those methods [53, 54]. In general, the sufficient conditions for identifiability (up to a permutation of the group labels) of MMSBM includes: (1) $\mathbf{\Omega}$ has full rank ($\text{rank}(\mathbf{\Omega}) = K$); (2) there is at least one "pure" node in each community (i.e., for each $k \in [K]$, there exists at least one i such that $\pi_{ik} = 1$). Additionally, few methods also include the degree-correction into their model [55, 57, 58], which will introduce more free parameters (node degree parameters) and thus require more restricted constraints to be identifiable. In this case, the above constraint (1) becomes: $\mathbf{\Omega}$ is full rank and has unit diagonals ($\text{rank}(\mathbf{\Omega}) = K$ and $\Omega_{kk} = 1, \forall k$); and with addition constraint on the degree parameters (3) equation (3.1). Also, for those methods based on the property of the simplex structure of eigenspace, the extension to the node degree-correction case is not simple and intuitive as in [35, 126]. In this case, the node degree heterogeneity will blur the simplex structure and challenge the original theoretical proof [55–57].

Note that even some of these papers study the asymptotic consistency theoretically under method-specific assumptions and conditions. Their corresponding algorithms are not guaranteed to perform as perfectly as the theoretical analysis (see the summarized comparison of these methods in Table 3.1). Also, It is worth pointing out that all methods for finding overlapping communities can also be adapted to find non-overlapping communities by assigning each node solely to the community to which it most strongly belongs in the overlapping division. Also, continuous community membership can be converted to binary type by thresholding each element in $\boldsymbol{\pi}_i$.

The Latent Position Cluster Model (LPC) of Handcock et al. [10] is very similar to the idea of mixed membership SBM [41]. Under the LPC, each node i has a latent position vector $\boldsymbol{\pi}_i$ in K-dimensional Euclidean latent space [140], where $\boldsymbol{\pi}_i$ is drawn from a finite mixture of

K multivariate Gaussian distribution. The edge generation probability between node i and j is a function of their Euclidean distance in the latent space.

3.5 Bipartite SBM

In general, a bipartite network can be viewed as a special case of a unipartite network, with the constraint that nodes within the same type cannot connect with each other. Similar to unipartite networks, community detection in bipartite networks aims to find node groups with similar connectivity patterns to other groups. With the flexible and intuitive structure of SBM, the effort in studying (degree-corrected) SBM has been extended to the bipartite case, such as biSBM [60] extended from DCSBM, biLCM [141] extended from LCM and BiMPCA [142] extended from Mixed-SCORE.

We let N_1 and N_2 denote the number of nodes in two types. We denote $N := N_1 + N_2$. Without loss of generality, we let $1 \leq i \leq N_1$ to index type 1 nodes and $N_1 < i \leq N$ to index type 2 nodes. We define $t_i \in \{1, 2\}$ as a function of node index i, where $t_i = 1$ indicating type 1 node and $t_i = 2$ indicating type 2 node. Then the adjacency matrix presents a structure with zero diagonal blocks:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}^\mathsf{T} & \mathbf{0} \end{bmatrix} \tag{3.9}$$

where $\mathbf{B}$ is $N_1 \times N_2$ bipartite adjacency matrix with each entry equal to 1 if corresponding two nodes are connected and 0 otherwise. Let K_1 and K_2 be the number of groups in type 1 and type 2 (without loss of generality, we assume $K_1 \leq K_2$), and the connectivity matrix of a bipartite SBM is defined by a $K_1 \times K_2$ matrix $\mathbf{\Omega}$ where Ω_{gh} represents the expected number of edges between two nodes lying in group g and group h, $g \in [K_1]$, $h \in [K_2]$. Thus, from a block model point of view, a bipartite SBM is simply an SBM where diagonal blocks equal to 0.

In general, when studying the community detection in bipartite networks under the framework of SBMs, a natural way is to directly apply community detection methods for SBMs to bipartite networks, as the SBM is capable of producing bipartite structure by

forcing nodes from the same type not to connect each other. However, Larremore et al. [60] have shown that simply applying unipartite community detection methods to the bipartite case using the adjacency matrix in Equation (3.9) will lose the information. In particular, they compare two types of model fitting approaches, one with an explicit constraint on the bipartite structure and one without. The two types of cases have numerically identical likelihood functions. However, they have different solution spaces as the former takes the bipartite structure as given and the latter does not. The authors find that compared to the bipartite case, the non-bipartite counterpart is likely to group nodes of the same type into the same community (i.e., there exist communities which only contains nodes of the same type), and computation, in this case, is less efficient as it solves the problem in a larger space. In their analysis, the algorithm fitted based on the biSBM outperforms the one fitted based on the non-bipartite SBM.

Similar to the unipartite scenario, community detection in the bipartite network can be divided into two types: algorithm-based methods and model-based methods. Algorithm-based methods involve optimizing an objective function over all possible network partitions such as modularity based method [143, 144], spectral based methods [28, 29, 85, 145–148] and NMF based method [149–151]. In particular, several literatures focus on studying the theoretical property of bipartite spectral clustering. For example, Zhou et el. [147] propose that in the bipartite case, spectral clustering performs better by replacing the first step singular value decomposition by principal component analysis; the authors also develop a data-driven approach to choose the regularization parameter. In another paper [29], the same authors use this type of spectral clustering as an initial method for the EM algorithm. On the other hand, Model-based methods fit the probabilistic generative model to the observed network and estimate the model parameters by optimizing a log-likelihood function [42, 60, 86, 141, 152].

In general, we do not assume any correspondence between communities in type-1 and type-2. In other words, there is no requirement for a diagonal block structure in $\mathbf{\Omega}$. Like in SBM [22], in biSBM we assume that the connectivity matrix $\mathbf{\Omega}$ cannot have two columns equal and the corresponding rows also equal. That is, $\forall\, g, g' \in [K_1],\ g \neq g',\ \exists\, h \in [K_2],$ s.t. $\Omega_{gh} \neq \Omega_{g'h}$

and $\forall\, h, h' \in [K_2],\ h \neq h',\ \exists\, g \in [K_1],$ s.t. $\Omega_{gh} \neq \Omega_{gh'}$. This is a necessary constraint for defining K_1 distinct clusters on side 1 and K_2 distinct clusters on side 2 with different structural properties. Therefore, settings where this assumption is violated correspond to ill-specified models with redundant clusters. Note that it will recover an one-to-one matching structure when $\mathbf{\Omega}$ is symmetric ($K_1 = K_2 = K$) and has the assortative structure: $\mathbf{\Omega} = q \cdot \mathbf{1}_K \mathbf{1}_K^\top + (p - q) \cdot \mathbf{I}_K$ (the variables p and q are positive constants, and $p > q$), see a detailed study about this special case in [152].

3.6 Summary and discussion

This chapter mainly introduces the current methods developed for detecting mixed membership communities under the SBM framework. We summarize them in Table 3.1. Even these methods, in general, share the same goal, in practice, it still remains a challenge to compare them systematically. The first issue lies in the fact that there is no unified generative model for defining the mixed membership community structure, as different models rely on different assumptions. And we are not clear about how similar the mixed membership community structure produced by the generative models with that existed in real-world data sets. The users' domain knowledge of the data is important to decide which method to apply or not. Another issue is due to the lack of universal evaluation criteria for mixed membership communities. Early efforts [117, 153] focus on comparing clustering methods that output a nonfuzzy node membership vector (the element in $\boldsymbol{\pi}_i$ can either be 0 or 1) rather than a fuzzy vector (the element in $\boldsymbol{\pi}_i$ can be a continuous positive value). Besides, extending the quality measures for disjoint to mixed membership communities is nontrivial. Compared to disjoint community detection, there are only two widely used measures for the overlapping case, which are the extended Normalized Mutual Information (NMI) [153] and the Omega index [154]. These two measurements are the extension of NMI and the extension of the Adjusted Rand Index (ARI) [110], respectively. Note that when there is no overlap, the extended NMI does not coincide with the standard NMI, but the Omega index does reduce to the ARI. Also, note that these two criteria cannot be directly applied to the fuzzy overlapping

scenario unless the node memberships are thresholded to binary vectors.

In addition, the extension of mixed membership SBM to the bipartite network is an important research direction, which is also the focus of Chapter 4. The main motivation is that real-world bipartite networks are frequently observed to have overlapped communities, and the methods designed based on the general unipartite networks should not be directly applied to bipartite networks [60]. For example, in the paper-author network, a researcher could work as a statistician in a biology paper, and he/she could also work as a software engineer in a computer science paper. In the gene orthology network, a gene could also belong to multiple functional groups.

The relevant efforts have been made to study the bipartite mixed membership stochastic block models. In particular, Whang et al. [86] formulate a mixed membership SBM with hard (binary) node memberships. Moreover, this model takes a nonparametric Bayesian approach to avoid inputting the number of clusters K. However, the resultant number of clusters, in this case, usually tends to be much greater than necessary. Compared to the mixed membership case, some of the communities here may be less interpretable or meaningful. They validate the model based on the link prediction task of drug-protein data, and their model does not consider the node degree heterogeneity.

Besides, Godoy-Lorite et al. [155] formulate a bipartite structure based on MMSBM to study the user-item recommender system. In this model, each node (user or item) is assigned with a mixture membership vector, and the edge (rating) between a pair of user and item is generated from a categorical distribution parameterized by the corresponding node membership vectors. This idea is intuitive as a user or item could belong to multiple groups.

3.7 Acknowledgements

I would like to thank Dr. Jingyi Jessica Li, Dr. Xin Tong, Dr. Zhixin Zhou, Kexin Li, Guan'ao Yan, and Ruochen Jiang for helping with the editing and technical discussions for this chapter.

Table 3.1: List of (degree-corrected) mixed membership stochastic block models and (degree-corrected) bipartite stochastic block models.

Model	Node Membership	Parameter Estimation	Bipartite[1]	Degree Correction[2]	Consistency[3]
MMSBM [48]	$\|\boldsymbol{\pi}_i\|_1 = 1$	variational EM	✗	✗	✗
OSBM [49]	$\pi_{ik} \in \{0, 1\}$	variational EM	✗	✗	✗
LCM [50]	$\pi_{ik} \in \mathbb{R}_+$	EM	✗	✗	✗
HMMB [125]	$\|\boldsymbol{\pi}_i\|_1 = 1$	MCMC	✗	✗	✗
biSBM [60]	$\exists! \, k, \text{ s.t. } \pi_{ik} = 1$	Kernighan-Lin	✓	✓	✗
mbiSBM [152]	$\exists! \, k, \text{ s.t. } \pi_{ik} = 1$	variational EM	✓	✓	✗
SC-RRE [29]	$\exists! \, k, \text{ s.t. } \pi_{ik} = 1$	spectral algorithm + EM	✓	✗	✓
biLCM [141]	$\pi_{ik} \in \mathbb{R}_+$	EM	✓	✗	✗
GeoNMF [54]	$\|\boldsymbol{\pi}_i\|_1 = 1$	NMF based method	✗	✗	✓
SPACL [56]	$\|\boldsymbol{\pi}_i\|_1 = 1$	spectral algorithm	✗	✗	✓
SPOC [55]	$\pi_{ik} \in \{0, 1\}$	spectral algorithm	✗	✗	✓
Tensor-Spectral [52]	$\|\boldsymbol{\pi}_i\|_1 = 1$	tensor power iteration	✗	✗	✓
SAAC/SBMO [53])	$\pi_{ik} \in \{0, 1\}$	spectral algorithm	✗	✗	✓
SPCA [59]	$\|\boldsymbol{\pi}_i\|_1 = 1$	spectral algorithm	✗	✓	✗
Mixed-SCORE [57]	$\|\boldsymbol{\pi}_i\|_1 = 1$	spectral algorithm	✗	✓	✓
OCCAM [58]	$\|\boldsymbol{\pi}_i\|_2 = 1$	spectral algorithm	✗	✓	✓

[1] The method is applied to bipartite network or not

[2] The method considers degree-correction or not

[3] The method justifies the consistent estimation or statistical optimality under theoretical limits or not

CHAPTER 4

A Bipartite Mixed Membership Stochastic Block Model

In this chapter, we focus on designing an algorithm that detects overlap communities from a bipartite network and accounts for node degree heterogeneity. As discussed in Section 3.6, studying the mixed membership communities in the bipartite networks has an increasing need in real-world applications, such as the bipartite networks of user-item, drug-protein, paper-author, etc. In Chapter 3, we have shown that the SBM is flexible enough to be extended to accommodate the characteristics existing in the real-world networks. Here we would like to construct a general and interpretable model, based on the SBM, to simultaneously capture the following three key features prevalent in real-world networks: (1) heterogeneous node degrees; (2) mixed membership communities; (3) bipartite structure.

4.1 Model

Here, we propose a new model: bipartite mixed membership stochastic block model with degree correction. Four well-cited papers serve as the foundation:

- The degree-corrected stochastic block model (DCSBM) [35]. See details in Chapter 3.3.
- The mixed membership stochastic block model (MMSBM) [48]. See details in Chapter 3.4.
- The bipartite stochastic block model (biSBM) [60]. See details in Chapter 3.5.
- The matched bipartite stochastic block model (mbiSBM) [152]. See details in Chapter 3.5.

The bayesian generative model is stated as follows.

For each node i on the side 1 and node j on the side 2, let $\boldsymbol{\pi}_i$ denote the mixed membership vector of node i and let $\boldsymbol{\pi}_j$ denote the mixed membership vector of node j, then both of them are generated from the Dirichlet prior:

$$\boldsymbol{\pi}_i \overset{\text{i.i.d}}{\sim} \text{Dirichlet}(\boldsymbol{\alpha}_1) \tag{4.1}$$

$$\boldsymbol{\pi}_j \overset{\text{i.i.d}}{\sim} \text{Dirichlet}(\boldsymbol{\alpha}_2) \tag{4.2}$$

Between each pair of node i on side 1 and node j on side 2, the edge is generated from a Poisson distribution:

$$A_{ij} \sim \text{Poisson}(\theta_i \theta_j \boldsymbol{z}_{i\to j}^{\mathsf{T}} \boldsymbol{\Omega} \boldsymbol{z}_{i\leftarrow j}) \tag{4.3}$$

where $\boldsymbol{z}_{i\to j}$ denotes the latent group membership of node i when interacting with node j, and $\boldsymbol{z}_{i\leftarrow j}$ (or $\boldsymbol{z}_{j\to i}$) denotes the latent group membership of node j when interacting with node i. Both of them are generated from the multinomial distributions in (4.4) and (4.5):

$$\boldsymbol{z}_{i\to j} \overset{\text{i.i.d}}{\sim} \text{Multinomial}(1, \boldsymbol{\pi}_i) \tag{4.4}$$

$$\boldsymbol{z}_{i\leftarrow j} \overset{\text{i.i.d}}{\sim} \text{Multinomial}(1, \boldsymbol{\pi}_j) \tag{4.5}$$

Note that the group membership of each node defined in (4.4) and (4.5) are edge-specific, and each node can have different membership when interacting with nodes on the other side. With abuse of notation, we denote $\boldsymbol{\pi} := \{\boldsymbol{\pi}_i,\ i \in [N]\}$, $\boldsymbol{z} := \{\boldsymbol{z}_{i\to j},\ t_i \neq t_j,\ i, j \in [N]\}$, $\boldsymbol{\theta} := \{\theta_i,\ i \in [N]\}$ and $\boldsymbol{\alpha} := \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2\}$. The observed data are $\mathbf{A}$, the latent variables are $\{\boldsymbol{z}, \boldsymbol{\pi}\}$ and the hyperparameters are $\{\boldsymbol{\theta}, \boldsymbol{\Omega}\}$. The joint probability function is

$$p(\mathbf{A}, \boldsymbol{z}, \boldsymbol{\pi} | \boldsymbol{\theta}, \boldsymbol{\Omega}, \boldsymbol{\alpha}) = \prod_{\substack{i<j \\ t_i \neq t_j}} p(A_{ij} | \theta_i, \theta_j, \boldsymbol{z}_{i\to j}, \boldsymbol{z}_{i\leftarrow j}, \boldsymbol{\Omega}) p(\boldsymbol{z}_{i\to j} | \boldsymbol{\pi}_i) p(\boldsymbol{z}_{i\leftarrow j} | \boldsymbol{\pi}_j) \prod_{i=1}^{N} p(\boldsymbol{\pi}_i | \boldsymbol{\alpha}_{t_i}) \tag{4.6}$$

As shown in (3.5) in Chapter 3.4, the latent variable $\boldsymbol{z}$ in the bipartite case could also be integrated out analytically, which leads to estimation over a smaller set of latent parameters (see (4.7)). Here we choose to keep the $\boldsymbol{z}$ to simplify inference (see discussion in [48]).

$$A_{ij} \sim \text{Poisson}(\theta_i \theta_j \boldsymbol{\pi}_i^{\mathsf{T}} \boldsymbol{\Omega} \boldsymbol{\pi}_j) \tag{4.7}$$

Following the exact definition in Section 3.5, the adjacency matrix $\mathbf{A}$ of the bipartite network contains zero diagonal blocks

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}^{\mathsf{T}} & \mathbf{0} \end{bmatrix} \tag{4.8}$$

where $\mathbf{B}$ is the $N_1 \times N_2$ bi-adjacency matrix. Figure 4.1 presents a diagram on the generation process under our model. As illustrated in (4.3), one can multiply θ_i and θ_j by a scalar c, and divide $\mathbf{\Omega}$ by c^2 without changing the edge distribution [45] (note that the elements in $\mathbf{\Omega}$ do not represent probabilities here). Given this, we set the following normalization constraint for θ_i without loss of generality:

$$\begin{aligned} \sum_{i:\, t_i=1} \boldsymbol{\pi}_i \theta_i &= \sum_{i:\, t_i=1} \boldsymbol{\pi}_i \\ \sum_{i:\, t_i=2} \boldsymbol{\pi}_i \theta_i &= \sum_{i:\, t_i=2} \boldsymbol{\pi}_i \end{aligned} \tag{4.9}$$

Note that Equation (4.9) and Equation (3.1) are basically equivalent, except that $\boldsymbol{\pi}_i$ here is mixed rather than binary. When $\theta_i = 1, \forall i$, we recover the non degree-corrected model. In general, the constraint (4.9) is not strictly held as we assume $\boldsymbol{\pi}_i$ is random rather than deterministic. Instead, we view (4.9) in the sense of average. In the following study, we use constraint (4.9) by taking the expectation with respect to the variational distribution of $\boldsymbol{\pi}_i$ (see (4.20) for details). Note that the general form (4.7) can include the following models as special cases with additional constraints:

- When all nodes are pure, $N_1 = N_2$, and $\mathbf{\Omega} = \mathbf{\Omega}^{\mathsf{T}}$, it recovers DCSBM
- When $N_1 = N_2$, $\mathbf{\Omega} = \mathbf{\Omega}^{\mathsf{T}}$, and $\theta_i = 1, \forall i$, it recovers MMSBM
- When all nodes are pure, it recovers biSBM
- When all nodes are pure and $\mathbf{\Omega} = \mathbf{\Omega}^{\mathsf{T}}$, it recovers mbiSBM

Recall Section 3.2, a node i is pure if there is only one entry in $\boldsymbol{\pi}_i$ is 1 and all other entries are 0. Mathematically, a node i is pure if $\exists\, k,$ s.t., $\pi_{ik} = \sum_l \pi_{il} = 1$.

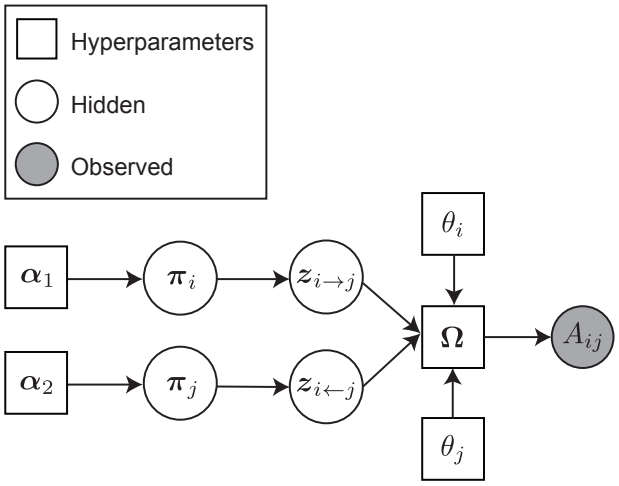

Figure 4.1: A diagram of the generative bipartite degree-corrected mixed membership stochastic block model.

4.2 Model fitting and variational EM algorithm

In our question, we are interested in finding the posterior distribution of the per-node mixed membership vectors $\boldsymbol{\pi}$ and the per-pair membership indicators $\boldsymbol{z}$. Moreover, the posterior inference allows us to form a predictive distribution of unobserved edges and evaluate the latent structures of the observed network. Also, we are interested in estimating the connectivity matrix $\boldsymbol{\Omega}$ and the Dirichlet parameters $\boldsymbol{\alpha}$. Let $\Sigma := \{\boldsymbol{z}, \boldsymbol{\pi}\}$ be the collection of the latent variables. Let $\Delta := \{\boldsymbol{\theta}, \boldsymbol{\Omega}, \boldsymbol{\alpha}\}$ be the collection of model parameters. Then the posterior distribution of Σ given the observed data $\mathbf{A}$ can be written as:

$$p(\Sigma|\mathbf{A}, \Delta) = \frac{p(\mathbf{A}, \Sigma|\Delta)}{p(\mathbf{A}|\Delta)} \tag{4.10}$$

However, the marginal distribution of $\mathbf{A}$ in the denominator requires the integration of the numerator over all the latent variables Σ, which will cost exponential time to compute [156, 157]. One way to avoid calculating the marginal distribution while estimating the posterior distribution is to use Markov chain Monte Carlo (MCMC), which is not scalable to large networks. Following the idea of how the MMSBM is originally fitted by [48], we derive a mean-field variational Bayes method for implementing the posterior inference and parameter estimation. See detailed reviews about the variational inference in [157–159].

The main idea lies in the variational method is to approximate a posterior distribution by a tractable family of distributions that has a simpler format than the true posterior. More specific, the variational distribution $q(\Sigma)$ is parameterized by the variational parameters Δ' (see details later). The variational parameters are fitted so that the variational distribution is as close as possible in Kullback-Leibler (KL) divergence to the true posterior. In detail, KL divergence is given by:

$$\begin{aligned}
\mathrm{KL}(q(\Sigma) \parallel p(\Sigma|\mathbf{A}, \Delta)) &= \mathbb{E}_q \left[\log \frac{q(\Sigma)}{p(\Sigma|\mathbf{A}, \Delta)} \right] \\
&= \mathbb{E}_q \left[\log q(\Sigma) \right] - \mathbb{E}_q \left[\log p(\Sigma|\mathbf{A}, \Delta) \right] \\
&= \mathbb{E}_q \left[\log q(\Sigma) \right] - \mathbb{E}_q \left[\log p(\Sigma, \mathbf{A}|\Delta) \right] + \log p(\mathbf{A}|\Delta) \qquad (4.11)
\end{aligned}$$

Note that KL divergence itself is not solvable as it involves calculating the marginal distribution of the data. However, minimizing KL divergence is equivalent to maximizing the negative of the first two items in Equation (4.11), defined by:

$$\begin{aligned}
\mathcal{L}(q, \Delta) &:= \mathbb{E}_q \left[\log p(\Sigma, \mathbf{A}|\Delta) \right] - \mathbb{E}_q \left[\log q(\Sigma) \right] \\
&= \mathbb{E}_q \left[\ell(\Delta) \right] - \mathbb{E}_q \left[\log q(\Sigma) \right] \qquad (4.12)
\end{aligned}$$

Here $\ell(\cdot)$ denotes the complete log likelihood function. As $\mathrm{KL} \geq 0$, then $\log p(\mathbf{A}|\Delta) \geq \mathcal{L}(q, \Delta)$, so $\mathcal{L}(q, \Delta)$ can be considered as a lower bound on the log likelihood $\log p(\mathbf{A}|\Delta)$. One maximizes the objective function in (4.12) over likelihood parameters and the variational parameters iteratively until convergence. Under the scheme of mean-field variational inference, each latent variable is mutually independent, the fully factored variational density is defined as:

$$q(\boldsymbol{z}, \boldsymbol{\pi}|\boldsymbol{\phi}, \boldsymbol{\gamma}) = \prod_{\substack{i<j \\ t_i \neq t_j}} q_1(\boldsymbol{z}_{i \to j}|\boldsymbol{\phi}_{i \to j}) q_1(\boldsymbol{z}_{i \leftarrow j}|\boldsymbol{\phi}_{i \leftarrow j}) \prod_{i=1}^{N} q_2(\boldsymbol{\pi}_i|\boldsymbol{\gamma}_i) \qquad (4.13)$$

where $q_1(\cdot)$ is the Multinomial distribution and $q_2(\cdot)$ is the Dirichlet distribution (See Figure 4.2 for an illustration). With abuse of notation, we denote $\boldsymbol{\gamma} := \{\boldsymbol{\gamma}_i,\ i \in [N]\}$ and $\boldsymbol{\phi} := \{\boldsymbol{\phi}_{i \to j},\ t_i \neq t_j,\ i, j \in [N]\}$. We use $\Delta' := \{\boldsymbol{\gamma}, \boldsymbol{\phi}\}$ to collect the variational parameters which need to be estimated. In the following step, we apply an iterative coordinate ascent variational expectation-maximization (EM) algorithm by alternatively optimizing the

likelihood parameters Δ and the variational parameters Δ', while holding the others fixed. The E-step and M-step are repeated until the lower bound (4.13) converges. Plugging in the log likelihood function (4.6) and the variational distribution (4.13) into the objective function (4.12), we get:

$$
\begin{aligned}
\mathcal{L}(q,\Delta) &= \mathbb{E}_q\left[\ell(\Delta)\right] - \mathbb{E}_q\left[\log q(\Sigma)\right] \\
&= \mathbb{E}_q\left[\log p(\mathbf{A}, \boldsymbol{z}, \boldsymbol{\pi}|\boldsymbol{\theta}, \boldsymbol{\Omega}, \boldsymbol{\alpha})\right] - \mathbb{E}_q\left[\log q(\boldsymbol{z}, \boldsymbol{\pi}|\boldsymbol{\phi}, \boldsymbol{\gamma})\right] \\
&= \mathbb{E}_q \log \left[\prod_{\substack{i<j \\ t_i \neq t_j}} p(A_{ij}|\theta_i, \theta_j, \boldsymbol{z}_{i\to j}, \boldsymbol{z}_{i\leftarrow j}, \boldsymbol{\Omega}) p(\boldsymbol{z}_{i\to j}|\boldsymbol{\pi}_i) p(\boldsymbol{z}_{i\leftarrow j}|\boldsymbol{\pi}_j) \prod_{i=1}^{N} p(\boldsymbol{\pi}_i|\boldsymbol{\alpha}_{t_i}) \right] \\
&- \mathbb{E}_q \log \left[\prod_{\substack{i<j \\ t_i \neq t_j}} q_1(\boldsymbol{z}_{i\to j}|\boldsymbol{\phi}_{i\to j}) q_1(\boldsymbol{z}_{i\leftarrow j}|\boldsymbol{\phi}_{i\leftarrow j}) \prod_{i=1}^{N} q_2(\boldsymbol{\pi}_i|\boldsymbol{\gamma}_i) \right] \\
&= \mathbb{E}_q \left[\sum_{\substack{i<j \\ t_i \neq t_j}} \left[\log p(A_{ij}|\theta_i, \theta_j, \boldsymbol{z}_{i\to j}, \boldsymbol{z}_{i\leftarrow j}, \boldsymbol{\Omega}) + \log p(\boldsymbol{z}_{i\to j}|\boldsymbol{\pi}_i) + \log p(\boldsymbol{z}_{i\leftarrow j}|\boldsymbol{\pi}_j)\right] + \sum_{i=1}^{N} \log p(\boldsymbol{\pi}_i|\boldsymbol{\alpha}_{t_i}) \right] \\
&- \mathbb{E}_q \left[\sum_{\substack{i<j \\ t_i \neq t_j}} \left[\log q_1(\boldsymbol{z}_{i\to j}|\boldsymbol{\phi}_{i\to j}) + \log q_1(\boldsymbol{z}_{i\leftarrow j}|\boldsymbol{\phi}_{i\leftarrow j})\right] + \sum_{i=1}^{N} \log q_2(\boldsymbol{\pi}_i|\boldsymbol{\gamma}_i) \right] \\
&= \sum_{\substack{i<j \\ t_i \neq t_j}} \mathbb{E}_q \left[A_{ij} \log(\theta_i \theta_j \boldsymbol{z}_{i\to j}^T \boldsymbol{\Omega} \boldsymbol{z}_{i\leftarrow j}) - \theta_i \theta_j \boldsymbol{z}_{i\to j}^T \boldsymbol{\Omega} \boldsymbol{z}_{i\leftarrow j} - \log(A_{ij})! \right] \\
&+ \sum_{\substack{i<j \\ t_i \neq t_j}} \mathbb{E}_q \left[\sum_{k=1}^{K_1} z_{i\to jk} \log \pi_{ik} + \sum_{k=1}^{K_2} z_{i\leftarrow jk} \log \pi_{jk} \right] \\
&+ \sum_{i=1}^{N} \mathbb{E}_q \left[\log \frac{\Gamma(\sum_k \alpha_{t_i k})}{\prod_k \Gamma(\alpha_{t_i k})} + \sum_k (\alpha_{t_i k} - 1) \log \pi_{ik} \right] \\
&- \sum_{\substack{i<j \\ t_i \neq t_j}} \mathbb{E}_q \left[\sum_{k=1}^{K_1} z_{i\to jk} \log \phi_{i\to jk} + \sum_{k=1}^{K_2} z_{i\leftarrow jk} \log \phi_{i\leftarrow jk} \right] \\
&- \sum_{i=1}^{N} \mathbb{E}_q \left[\log \frac{\Gamma(\sum_k \gamma_{ik})}{\prod_k \Gamma(\gamma_{ik})} + \sum_k (\gamma_{ik} - 1) \log \pi_{ik} \right]
\end{aligned}
$$

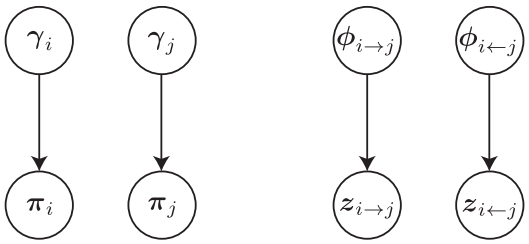

Figure 4.2: The diagram of the variational distribution. The latent variables are assumed to be mutually independent. Each variable is governed by its own variational parameter. That is, $\forall i, j \in [N],\ \boldsymbol{\pi}_i \sim \text{Dirichlet}_q(\boldsymbol{\gamma}_i),\ \boldsymbol{z}_{i\to j} \sim \text{Multinomial}_q(\boldsymbol{\phi}_{i\to j})$.

Take each expectation, and we have:

$$\begin{aligned}
\mathcal{L}(q,\Delta) = & \sum_{\substack{i<j \\ t_i\neq t_j}} \sum_{g=1}^{K_1}\sum_{h=1}^{K_2} \phi_{i\to jg}\phi_{i\leftarrow jh} f(A_{ij}, \theta_i\theta_j\Omega_{gh}) \\
& + \sum_{\substack{i<j \\ t_i\neq t_j}} \sum_{k=1}^{K_1} \phi_{i\to jk}\left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right] + \sum_{\substack{i<j \\ t_i\neq t_j}} \sum_{k=1}^{K_2} \phi_{i\leftarrow jk}\left[\psi(\gamma_{jk}) - \psi(\sum_k \gamma_{jk})\right] \\
& + \sum_{i=1}^{N}\left[\log\Gamma(\sum_k \alpha_{t_ik}) - \sum_k \log\Gamma(\alpha_{t_ik})\right] + \sum_{i=1}^{N}\sum_k (\alpha_{t_ik}-1)\left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right] \\
& - \sum_{\substack{i<j \\ t_i\neq t_j}} \left[\sum_{k=1}^{K_1} \phi_{i\to jk}\log\phi_{i\to jk} + \sum_{k=1}^{K_2} \phi_{i\leftarrow jk}\log\phi_{i\leftarrow jk}\right] \\
& - \sum_{i=1}^{N}\left[\log\Gamma(\sum_k \gamma_{ik}) - \sum_k \log\Gamma(\gamma_{ik})\right] - \sum_{i=1}^{N}\sum_k (\gamma_{ik}-1)\left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right] \quad (4.14)
\end{aligned}$$

where $f(x,y) = x\log(y) - y - \log(x!)$ and $\psi(\cdot) = \frac{d\log\Gamma(x)}{dx}$ is the derivative of the log-gamma function, also called digamma function.

4.2.1 Variational E step

In this section, we derive the updating steps for the variational parameters Δ' by using the coordinate ascent algorithm. Basically, for each variational parameter, we will first isolate the terms in the objective function (4.14) containing the parameter and then set the first derivative equal to zero and solve the equation.

Update $\boldsymbol{\phi}$ We update the variational parameters $\boldsymbol{\phi}$. We collect the terms in $\mathcal{L}(q, \Delta)$ that are related to $\{\boldsymbol{\phi}_{i\to j}, \boldsymbol{\phi}_{i\leftarrow j}\}$, $\forall(i,j),\ i<j,\ t_i \neq t_j$. Then we maximize the following objective functions:

$$\mathcal{L}_{\boldsymbol{\phi}_{i\to j}} = \sum_{g=1}^{K_1}\sum_{h=1}^{K_2}\phi_{i\to jg}\phi_{i\leftarrow jh}f(A_{ij},\theta_i\theta_j\Omega_{gh}) + \sum_{k=1}^{K_1}\phi_{i\to jk}\left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right] - \sum_{k=1}^{K_1}\phi_{i\to jk}\log\phi_{i\to jk}$$

$$\mathcal{L}_{\boldsymbol{\phi}_{i\leftarrow j}} = \sum_{g=1}^{K_1}\sum_{h=1}^{K_2}\phi_{i\to jg}\phi_{i\leftarrow jh}f(A_{ij},\theta_i\theta_j\Omega_{gh}) + \sum_{k=1}^{K_2}\phi_{i\leftarrow jk}\left[\psi(\gamma_{jk}) - \psi(\sum_k \gamma_{jk})\right] - \sum_{k=1}^{K_2}\phi_{i\leftarrow jk}\log\phi_{i\leftarrow jk}$$

subject to the normalization constraint:

$$\sum_{k=1}^{K_1}\phi_{i\to jk} = \sum_{k=1}^{K_2}\phi_{i\leftarrow jk} = 1 \tag{4.15}$$

Apply the Lemma 1 in [152] (see details about this lemma in Section 4.A) and we have the updating steps for $\{\boldsymbol{\phi}_{i\to j}, \boldsymbol{\phi}_{i\leftarrow j}\}$ with constraint (4.15):

$$\phi_{i\to jk} \propto \exp\left\{\sum_{h=1}^{K_2}\phi_{i\leftarrow jh}f(A_{ij},\theta_i\theta_j\Omega_{kh}) + \left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right]\right\}, \quad k\in[K_1] \tag{4.16}$$

$$\phi_{i\leftarrow jk} \propto \exp\left\{\sum_{g=1}^{K_1}\phi_{i\to jg}f(A_{ij},\theta_i\theta_j\Omega_{gk}) + \left[\psi(\gamma_{jk}) - \psi(\sum_k \gamma_{jk})\right]\right\}, \quad k\in[K_2] \tag{4.17}$$

Update $\boldsymbol{\gamma}$ We update the variational parameters $\boldsymbol{\gamma}$. We collect the terms in $\mathcal{L}(q, \Delta)$ that are related to $\boldsymbol{\gamma}_i$, $\forall i \in [N]$. And we have:

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{\gamma}_i} =& \sum_{j:t_j\neq t_i}\sum_k \phi_{i\to jk}\left[\psi(\gamma_{ik}) - \psi(\sum_k\gamma_{ik})\right] + \sum_k(\alpha_{t_ik}-1)\left[\psi(\gamma_{ik}) - \psi(\sum_k\gamma_{ik})\right] \\
&- \left[\log\Gamma(\sum_k\gamma_{ik}) - \sum_k\log\Gamma(\gamma_{ik})\right] - \sum_k(\gamma_{ik}-1)\left[\psi(\gamma_{ik}) - \psi(\sum_k\gamma_{ik})\right] \\
=& \sum_k\left[\psi(\gamma_{ik}) - \psi(\sum_k\gamma_{ik})\right]\left\{\sum_{j:t_j\neq t_i}\phi_{i\to jk} + (\alpha_{t_ik}-1) - (\gamma_{ik}-1)\right\} - \left[\log\Gamma(\sum_k\gamma_{ik}) - \sum_k\log\Gamma(\gamma_{ik})\right] \\
=& \sum_k\psi(\gamma_{ik})\left\{\sum_{j:t_j\neq t_i}\phi_{i\to jk} + \alpha_{t_ik} - \gamma_{ik}\right\} - \psi(\sum_k\gamma_{ik})\sum_k\left\{\sum_{j:t_j\neq t_i}\phi_{i\to jk} + \alpha_{t_ik} - \gamma_{ik}\right\} \\
&- \left[\log\Gamma(\sum_k\gamma_{ik}) - \sum_k\log\Gamma(\gamma_{ik})\right]
\end{aligned}$$

Take the derivative of $\mathcal{L}_{\boldsymbol{\gamma}_i}$ with respect to $\gamma_{ik},\ \forall k \in [K_{t_i}]$, we have:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \gamma_{ik}} &= \psi'(\gamma_{ik}) \left\{ \sum_{j:t_j \neq t_i} \phi_{i\to jk} + \alpha_{t_i k} - \gamma_{ik} \right\} + \psi(\gamma_{ik})(-1) \\
&\quad - \psi'(\sum_k \gamma_{ik}) \sum_k \left\{ \sum_{j:t_j \neq t_i} \phi_{i\to jk} + \alpha_{t_i k} - \gamma_{ik} \right\} - \psi(\sum_k \gamma_{ik})(-1) \\
&\quad - \psi(\sum_k \gamma_{ik}) + \psi(\gamma_{ik}) \\
&= \psi'(\gamma_{ik}) \left\{ \sum_{j:t_j \neq t_i} \phi_{i\to jk} + \alpha_{t_i k} - \gamma_{ik} \right\} - \psi'(\sum_k \gamma_{ik}) \sum_k \left\{ \sum_{j:t_j \neq t_i} \phi_{i\to jk} + \alpha_{t_i k} - \gamma_{ik} \right\}
\end{aligned}$$

$$\begin{aligned}
&\text{Set} \quad \frac{\partial \mathcal{L}}{\partial \gamma_{ik}} = 0 \\
&\implies \gamma_{ik} = \sum_{j:t_j \neq t_i} \phi_{i\to jk} + \alpha_{t_i k},\ k \in [K_{t_i}]
\end{aligned} \tag{4.18}$$

where $\psi'(x) = \frac{d\psi(x)}{dx}$ is the first derivative of digamma function, also called trigamma function, and it is a positive function.

4.2.2 Variational M step

In this section, we update the likelihood parameters based on the idea of empirical Bayes estimation. We maximize the objective function with respect to the model parameters Δ, given the variational parameters Δ' fixed.

Update $\boldsymbol{\theta}$ We update the degree correction parameters $\boldsymbol{\theta}$. With an abuse of notation, we denote $\boldsymbol{\theta}_1 := (\theta_1, \theta_2, \ldots, \theta_{N_1})^\mathsf{T}$ and $\boldsymbol{\theta}_2 := (\theta_{N_1+1}, \theta_{N_1+2}, \ldots, \theta_N)^\mathsf{T}$. We first isolate the terms in $\mathcal{L}(q, \Delta)$ related to $\boldsymbol{\theta}_1$, and maximize the objective function below:

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{\theta}_1} &= \sum_{\substack{i<j \\ t_i \neq t_j}} \sum_{g=1}^{K_1} \sum_{h=1}^{K_2} \phi_{i\to jg} \phi_{i\leftarrow jh} \left[A_{ij} \log(\theta_i \theta_j \Omega_{gh}) - \theta_i \theta_j \Omega_{gh} \right] \\
&= \sum_{i=1}^{N_1} \log \theta_i \underbrace{\left[\sum_{j=N_1+1}^{N} \sum_{g=1}^{K_1} \sum_{h=1}^{K_2} \phi_{i\to jg} \phi_{i\leftarrow jh} A_{ij} \right]}_{a_i} - \sum_{i=1}^{N_1} \theta_i \underbrace{\left[\sum_{j=N_1+1}^{N} \sum_{g=1}^{K_1} \sum_{h=1}^{K_2} \phi_{i\to jg} \phi_{i\leftarrow jh} \theta_j \Omega_{gh} \right]}_{b_i}
\end{aligned} \tag{4.19}$$

subject to the constraint:

$$\sum_{i:\, t_i=1} \bar{\boldsymbol{\gamma}}_i \theta_i = \sum_{i:\, t_i=1} \bar{\boldsymbol{\gamma}}_i \tag{4.20}$$

where $\bar{\boldsymbol{\gamma}}_i := \frac{\boldsymbol{\gamma}_i}{\sum_k \gamma_{ik}}$ and $\bar{\boldsymbol{\gamma}}_i = \mathbb{E}_q[\boldsymbol{\pi}_i]$. Note that above normalization constraint holds by taking expectation of (4.9) under $\boldsymbol{\pi}_i \sim \mathrm{Dir}_q(\boldsymbol{\gamma}_i)$, $\forall i$. To integrate the notations into matrix or vector format, we denote $\boldsymbol{\Gamma}_1^{\mathsf{T}} := \left[\bar{\boldsymbol{\gamma}}_1, \bar{\boldsymbol{\gamma}}_2, \ldots, \bar{\boldsymbol{\gamma}}_{N_1}\right]$, $\boldsymbol{a}_1 := (a_1, a_2, \ldots, a_{N_1})^{\mathsf{T}}$, and $\boldsymbol{b}_1 := (b_1, b_2, \ldots, b_{N_1})^{\mathsf{T}}$. Then maximizing the objective function (4.19) with respect to $\boldsymbol{\theta}_1$ becomes solving the optimization problem:

$$\begin{aligned} \underset{\boldsymbol{\theta}_1}{\text{minimize}} \quad & -\boldsymbol{a}_1^{\mathsf{T}} \log \boldsymbol{\theta}_1 + \boldsymbol{b}_1^{\mathsf{T}} \boldsymbol{\theta}_1 \\ \text{subject to} \quad & \boldsymbol{\Gamma}_1^{\mathsf{T}} \boldsymbol{\theta}_1 = \boldsymbol{\Gamma}_1^{\mathsf{T}} \mathbf{1} \end{aligned} \tag{4.21}$$

which is suitable for an application of the Douglas-Rachford (DR) splitting algorithm [160]. This implementation is motivated by Razaee et al. [152], which for the first time propose to apply the DR splitting algorithm for updating degree heterogeneity parameters in their research context. We refer the reader to the detailed derivation of this algorithm in Appendix C.2 in [152]. In particular, it iterates the following updates from $(\boldsymbol{\theta}_1, \boldsymbol{\xi}_1, \boldsymbol{u}_1)$ to $(\boldsymbol{\theta}_1^+, \boldsymbol{\xi}_1^+, \boldsymbol{u}_1^+)$ until convergence (the same updating steps apply to $\boldsymbol{\theta}_2$, with subscript 1 replaced by 2):

$$\begin{aligned} \boldsymbol{\theta}_1^+ &= f_t(\boldsymbol{\xi}_1 - \boldsymbol{u}_1;\ \boldsymbol{a}_1, \boldsymbol{b}_1) \\ \boldsymbol{\xi}_1^+ &= \boldsymbol{\theta}_1^+ + \boldsymbol{u}_1 - \bar{\boldsymbol{\gamma}}_1 (\bar{\boldsymbol{\gamma}}_1^T \bar{\boldsymbol{\gamma}}_1)^{-1} \left[\bar{\boldsymbol{\gamma}}_1^T (\boldsymbol{\theta}_1^+ + \boldsymbol{u}_1) - \bar{\boldsymbol{\gamma}}_1^T \mathbf{1}\right] \\ \boldsymbol{u}_1^+ &= \boldsymbol{u}_1 + \boldsymbol{\theta}_1^+ - \boldsymbol{\xi}_1^+ \end{aligned} \tag{4.22}$$

where $[f_t(x;\ a, b)]_i := \frac{1}{2}\left[-(b_i t - x_i) + \sqrt{(b_i t - x_i)^2 + 4 a_i t}\right]$, $t > 0$. We set $t = 1$ in our implementation.

Update $\boldsymbol{\alpha}$ We update the Dirichlet parameters $\boldsymbol{\alpha}$. We first isolate the terms in the objective function related to $\boldsymbol{\alpha}_1$ (the same updating steps apply to $\boldsymbol{\alpha}_2$, with subscript 1 replaced by 2) and obtain:

$$\mathcal{L}_{\boldsymbol{\alpha}_1} = \sum_{i=1}^{N_1} \left[\log \Gamma(\sum_k \alpha_{1k}) - \sum_k \log \Gamma(\alpha_{1k})\right] + \sum_{i=1}^{N_1} \sum_k (\alpha_{1k} - 1) \left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik})\right] \tag{4.23}$$

For each $k \in [K_1]$, taking the first derivative of (4.23) with respect to α_{1k} gives us:

$$\frac{\partial \mathcal{L}}{\partial \alpha_{1k}} = N_1 \left[\psi(\sum_k \alpha_{1k}) - \psi(\alpha_{1k}) \right] + \sum_{i=1}^{N_1} \left[\psi(\gamma_{ik}) - \psi(\sum_k \gamma_{ik}) \right] \tag{4.24}$$

This derivative depends on $\alpha_{1k'}$, where $k' \neq k, k' \in [K_1]$, and we therefore must use an iterative method to find the maximal $\boldsymbol{\alpha}_1$. Here we apply the linear-time Newton-Raphson method for estimating the Dirichlet distribution, which has been implemented in LDA [156] and MMSBM [48]). We refer the reader to the detailed derivation in [161, 162]. Take the second derivatives with respect to each entry in $\boldsymbol{\alpha}_1$ and get:

$$\begin{aligned} \frac{\partial^2 \mathcal{L}}{\partial (\alpha_{1k})^2} &= N_1 \left[\psi'(\sum_k \alpha_{1k}) - \psi'(\alpha_{1k}) \right] \\ \frac{\partial^2 \mathcal{L}}{(\partial \alpha_{1k}) \partial (\alpha_{1j})} &= N_1 \psi'(\sum_k \alpha_{1k}), \quad j \neq k \end{aligned}$$

Here the Hessian matrix $\mathbf{H}$ is given by:

$$\begin{aligned} \mathbf{H} &= \mathbf{Q} + c\mathbf{1}\mathbf{1}^T \\ \mathbf{Q} &= \operatorname{diag}\left(-N_1 \psi'\left(\boldsymbol{\alpha}_1\right)\right) \\ c &= N_1 \psi'(\sum_k \alpha_{1k}) \end{aligned}$$

Then the updating for $\boldsymbol{\alpha}_1$ is written as:

$$\boldsymbol{\alpha}_1^{\text{new}} = \boldsymbol{\alpha}_1^{\text{old}} - \mathbf{H}^{-1} \cdot \nabla \mathcal{L} \tag{4.25}$$

where we take the inverse of Hessian matrix by:

$$\begin{aligned} \mathbf{H}^{-1} &= \mathbf{Q}^{-1} - \frac{\mathbf{Q}^{-1}\mathbf{1}\mathbf{1}^T\mathbf{Q}^{-1}}{1/c + \mathbf{1}^T\mathbf{Q}^{-1}\mathbf{1}} \\ (\mathbf{H}^{-1}\nabla\mathcal{L})_k &= \frac{(\nabla\mathcal{L})_k - b}{Q_{kk}} \\ \text{where} \quad b &= \frac{\mathbf{1}^T\mathbf{Q}^{-1}\nabla\mathcal{L}}{1/c + \mathbf{1}^T\mathbf{Q}^{-1}\mathbf{1}} = \frac{\sum_j (\nabla\mathcal{L})_j / Q_{jj}}{1/c + \sum_j 1/Q_{jj}} \end{aligned}$$

Update $\boldsymbol{\Omega}$ We update the connectivity matrix $\boldsymbol{\Omega}$. For each entry $\Omega_{gh}, g \in [K_1], h \in [K_2]$, we collect the related terms in the objective function and get:

$$\begin{aligned}\mathcal{L}_{\Omega_{gh}} &= \sum_{\substack{i<j \\ t_i \neq t_j}} \phi_{i\to jg}\phi_{i\leftarrow jh}\left[A_{ij}\log(\theta_i\theta_j\Omega_{gh}) - \theta_i\theta_j\Omega_{gh}\right] \\ &= \sum_{\substack{i<j \\ t_i \neq t_j}} \phi_{i\to jg}\phi_{i\leftarrow jh}\left[A_{ij}\log(\Omega_{gh}) - \theta_i\theta_j\Omega_{gh}\right]\end{aligned}$$

Take the first derivative with respect to Ω_{gh} and set it to zero, and we get:

$$\hat{\Omega}_{gh} = \frac{\sum_{\substack{i<j \\ t_i \neq t_j}} \phi_{i\to jg}\phi_{i\leftarrow jh}A_{ij}}{\sum_{\substack{i<j \\ t_i \neq t_j}} \phi_{i\to jg}\phi_{i\leftarrow jh}\theta_i\theta_j} \tag{4.26}$$

4.2.3 Parameter estimation

Algorithm 3 summarizes the above updates for fitting the model. As we know, the variational inference will converge to a local optimum and is sensitive to the choice of initial values [124, 157]. Therefore, we need to pay attention to the initialization of the variational parameters $\boldsymbol{\phi}$. In general, trying multiple starting values and choosing the one with the optimal value of $\mathcal{L}(q, \Delta)$ as the final output is feasible but is time-consuming. Previous researches [124, 152] suggest that taking the results from some pre-analysis results such as spectral clustering [24, 28] will improve the algorithm performance. In our study, we apply the spectral clustering algorithm developed in [15] to the adjacency matrix $\mathbf{A}$. Besides, in the simulation study (Section 4.3), as we know the true node memberships $\boldsymbol{z}$, so we can try a perturbed version of the truth as an initialization (note that this is motivated by the idea in the simulation study of [152]). Specifically, $\forall i, j \in [N]$, we initialize $\boldsymbol{\phi}_{i\to j}$ with a mixture of the true label and the Dirichlet noise, i.e., $\boldsymbol{\phi}_{i\to j} = \omega\boldsymbol{z}_{i\to j} + (1-\omega)\boldsymbol{\epsilon}_i$, where $\boldsymbol{\epsilon}_i \sim \text{Dir}(0.5 \cdot \mathbf{1})$ and $\omega \in [0, 1]$ is the proportion of the correct labels. In summary, to study the sensitivity of the algorithm, we try the following initialization for $\boldsymbol{\phi}$:

- The result of bipartite spectral clustering [15, 28]
- The perturbed truth: $\boldsymbol{\phi}_{i\to j} = \omega\boldsymbol{\pi}_i + (1-\omega)\boldsymbol{\epsilon}_i$; in our study, we set $\omega = 10\%$

Algorithm 3 The variational EM algorithm

Input: $\mathbf{B}$, K_1, K_2, convergence criteria ε, maximum number of iterations T

Initialization: $\forall\, i < j,\ i, j \in [N],\ t_i \neq t_j$

1. $\theta_i = 1,\ \theta_j = 1$
2. $\boldsymbol{\alpha}_1 = \frac{50}{K_1} \cdot \mathbf{1},\ \boldsymbol{\alpha}_2 = \frac{50}{K_2} \cdot \mathbf{1}$
3. $\boldsymbol{\phi}_{i \to j} = \frac{1}{K_1} \cdot \mathbf{1},\ \boldsymbol{\phi}_{i \leftarrow j} = \frac{1}{K_2} \cdot \mathbf{1}$
4. $\boldsymbol{\gamma}_i = \boldsymbol{\alpha}_1 + \frac{N_1}{K_1},\ \boldsymbol{\gamma}_j = \boldsymbol{\alpha}_2 + \frac{N_2}{K_2}$

while not `Converge`, nor Max Iterations T Reached **do**

1. Update $\boldsymbol{\phi}$ by (4.16)
2. Update $\boldsymbol{\gamma}$ by (4.18)
3. Update $\boldsymbol{\theta}$ by repeating (4.22) till convergence
4. Update $\boldsymbol{\alpha}$ by repeating (4.25) till convergence
5. Update $\boldsymbol{\Omega}$ using (4.26)
6. `Converge` $\leftarrow \left[\frac{(\mathcal{L}_{\text{old}} - \mathcal{L})}{\mathcal{L}_{\text{old}}} < \varepsilon\right]$

end

- The uninformative Dirichlet distribution: $\boldsymbol{\phi}_{i\to j} \sim \text{Dirichlet}(\frac{N_{t_i}}{K_{t_i}} \cdot \mathbf{1})$

4.3 Preliminary simulation study

Recall that our main interest is to estimate the per-node membership vectors $\boldsymbol{\pi}$, per-pair membership indicators $\boldsymbol{z}$, and the latent block structure $\boldsymbol{\Omega}$. In order to test the effectiveness of our method in recovering the truth, we design the simulation as follows. We generate the network data from the model introduced in Section (4.1). We set the network size at $(N_1, N_2) = (100, 200)$, and set the number of communities at $(K_1, K_2) = (4, 4)$. We use a easily identifiable planted community structure with $\boldsymbol{\Omega} = q \cdot \mathbf{1}_4\mathbf{1}_4^T + (p-q) \cdot \mathbf{I}_4$, and we set $(p, q) = (0.7, 0.1)$. Following the idea of the simulation study in [48], we set $\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_2 = \alpha \cdot \mathbf{1}$ and we try three values of $\alpha = (0.05, 0.10, 0.25)$ to simulate different settings of the node memberships. As α increases, each node membership vector is more likely to be diffused. We design two scenarios for data generation, including node degree heterogeneity (Figure 4.3) and not including node degree heterogeneity (Figure 4.4). In the first scenario, we generate the node degree parameters $\boldsymbol{\theta}$ from a Pareto distribution (see details in Section 4.C).

The final output of the algorithm includes the estimated variational parameters $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\gamma}}$, and the estimated likelihood parameter $\hat{\boldsymbol{\Omega}}$ and $\theta_i, \forall i \in [N]$ (the degree-corrected case). The posterior expectation of per-node membership vector $\boldsymbol{\pi}_i$ is approximated by $\mathbb{E}[\boldsymbol{\pi}_i|\mathbf{B}] \approx \frac{\hat{\gamma}_i}{\sum \hat{\gamma}_i}$, also denoted by $\hat{\boldsymbol{\pi}}_i$. And the posterior expectation of per-pair node membership indicator $\boldsymbol{z}_{i\to j}$ is approximated by $\mathbb{E}[\boldsymbol{z}_{i\to j}|\mathbf{B}] \approx \hat{\boldsymbol{\phi}}_{i\to j}$, also denoted by $\hat{\boldsymbol{z}}_{i\to j}$.

We evaluate the model fitting results by illustrating five types of heatmaps; see Figure 4.3 and Figure 4.4 for non-degree-correction and degree-correction cases, respectively. In sum, we compare the observed bi-adjacency matrix $\mathbf{B}$ with two types of predicted bi-adjacency matrices $\hat{\mathbf{B}}$. The detailed steps are stated as follows. We first assign each node i to the group it is most associated with, according to its true mixed membership vector $\boldsymbol{\pi}_i$; that is, we convert each node i to a pure node. And then, we use the converted node memberships to reorder the matrices to make the block structure more explicit in the heatmaps. For example,

Figure 4.3 contains five columns, corresponding to the following five types of heatmaps: (a) the observed bi-adjacency matrix $\mathbf{B}$; (b) the predicted bi-adjacency matrix, with each entry (i, j) equal to $\hat{\theta}_i\hat{\theta}_j\hat{\boldsymbol{z}}_{i\to j}^{\mathsf{T}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{z}}_{i\leftarrow j}$; (c) the predicted bi-adjacency matrix, with each entry (i, j) equal to $\hat{\theta}_i\hat{\theta}_j\hat{\boldsymbol{\pi}}_i^{\mathsf{T}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{\pi}}_j$; (d) the posterior-mean of mixed membership vectors of type-1 nodes, with each row equal to $\hat{\boldsymbol{\pi}}_i$; (e) the posterior-mean of mixed membership vectors of type-2 nodes. Rows and columns of matrices in (a)-(c) are reordered by the most likely memberships of the nodes. Rows of matrices in (d)-(e) are reordered by the most likely memberships of the nodes. To have a distinctive view, we threshold entries in each bi-adjacency matrix at the value of 5. In Figure 4.3 and Figure 4.4, three parts (numbered by $1, 2, 3$, from top to bottom) are corresponding to the results (heatmaps in columns (b)-(e)) conducted by three types of initialization: the results in part 1 are initialized by the bipartite spectral clustering; the results in part 2 are initialized by the perturbed truth; the results in part 3 are initialized by the random Dirichlet noise. Also, within each part, there are three rows of heatmaps, corresponding to α equal to $0.05, 0.10, 0.25$ (from top to bottom).

Based on the heatmaps in Figure 4.3, we summarize the results as follows. First, we observe that the algorithm is sensitive to the initialization. In particular, when the starting values are initialized by the spectral clustering results (heatmaps in part 1), the block structures in the predicted bi-adjacency matrices (columns (b) and (c)) are more clearly revealed compared to the ones in parts 2 and 3. And when the starting values are almost noises (heatmaps in parts 2 and 3), the block structures are less obvious. This is because the algorithm initialized with a bad starting point gets stuck at a local optimum rather than a global optimum. Second, we notice that, under each scenario (each row), the predicted bi-adjacency matrix based on the edge-specific membership indicators $\boldsymbol{z}$ (column (b)) exhibits a more clear block structure than the one based on the node-specific mixed membership vectors $\boldsymbol{\pi}$ (column (c)). In particular, when the initial values are not of good quality (part 2 and 3), compared to column b, the block structures of the predicted bi-adjacency matrix in column c are not recovered, with all entries concentrated around 0.25. Given this, we further visualize the heatmaps of node-specific membership vectors of type-1 nodes (column d) and

type-2 nodes (column e). And we find that in each case (row) where the prediction of the bi-adjacency matrix (column c) is bad, the corresponding estimated node membership vectors are almost uniform (i.e., each $\hat{\boldsymbol{\pi}}_i$ is approximately equal to $0.25 \cdot \mathbf{1}_4$). Intuitively, $\hat{\theta}_i\hat{\theta}_j\hat{\boldsymbol{\pi}}_i^{\mathsf{T}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{\pi}}_j$ and $\hat{\theta}_i\hat{\theta}_j\hat{\boldsymbol{z}}_{i\to j}^{\mathsf{T}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{z}}_{i\leftarrow j}$ are basically the posterior expectation of the bi-adjacency matrix at different levels of resolution [48], and this phenomenon makes sense as the edge generation probabilities are conditional on the $\boldsymbol{z}$ rather than $\boldsymbol{\pi}$. We also observed a similar phenomenon in the degree-corrected case in Figure 4.4. For future work, we intend to investigate this phenomenon at a more detailed level to get a better understanding.

4.4 Conclusion and future work

In this chapter, we study the bipartite community detection problem in the context of Bayesian inference, where we use the mean-field variational inference technique to approximate the posterior distribution. The first step postulates a family of variational distributions over the latent variables, which are independently distributed. In the second step, we use CAVI to iteratively optimize each type of variational parameter marginally. This step aims to find the variational distribution, which is the closest in KL divergence to the exact posterior distribution. However, in general, CAVI only guarantees to converge to a local optimum of the objective and is sensitive to the initialization in practice. Moreover, the independence assumption made by the mean-field variational inference will make the issue of local optima even more worse [157]. Based on our current simulation results, the performance of the variational EM algorithm indeed highly depends on the quality of the initial values. In our next step, we need to consider proposing a good initialization strategy (i.e., close to the truth) for users to implement the algorithm. Besides, as discussed in Section 3.4 in Chapter 3, the sufficient conditions of identifiability in MMSBM include a full rank $\boldsymbol{\Omega}$ with unit diagonals and the existence of at least one pure node in each community. In future work, we would like to include a rigorous proof of the identifiability of our model.

Also, we will explore whether there exists a better posterior approximation than the mean-field approximation. It worth nothing that how to develop such approximation while

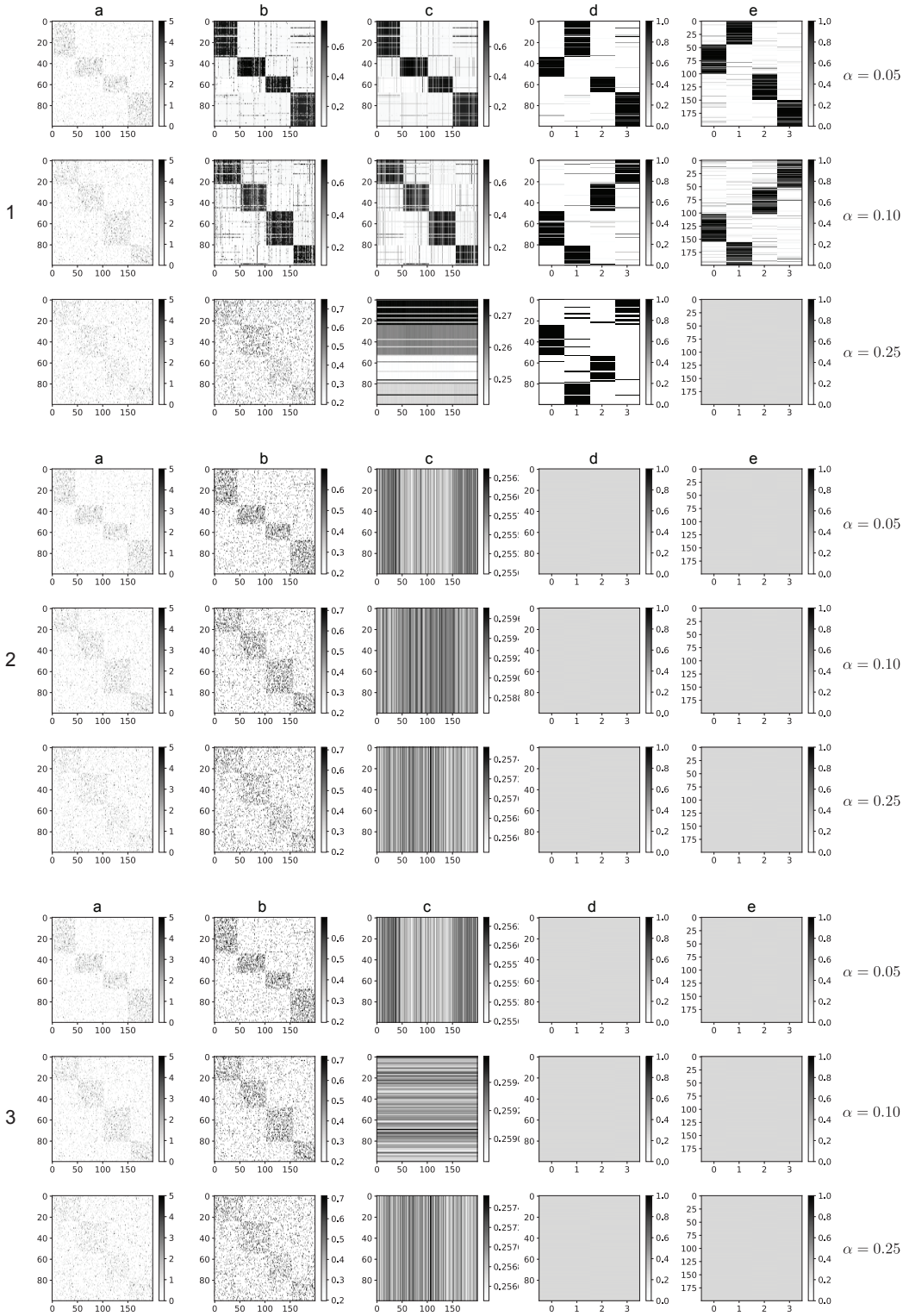

Figure 4.3: The simulated bi-adjacency matrices in column (a) are generated without node degree heterogeneity, and the algorithm is performed without degree correction. Part (1) results are based on the initialization of spectral clustering. Part (2) results are based on the initialization of the perturbed truth. Part (3) results are based on the initialization of the random Dirichlet noise. Within each part, from top to bottom, the matrices in column (a) are generated using $\alpha = 0.05, 0.10, 0.25$, respectively. See main text (Page 86) for the detailed legend of columns (a)-(e).

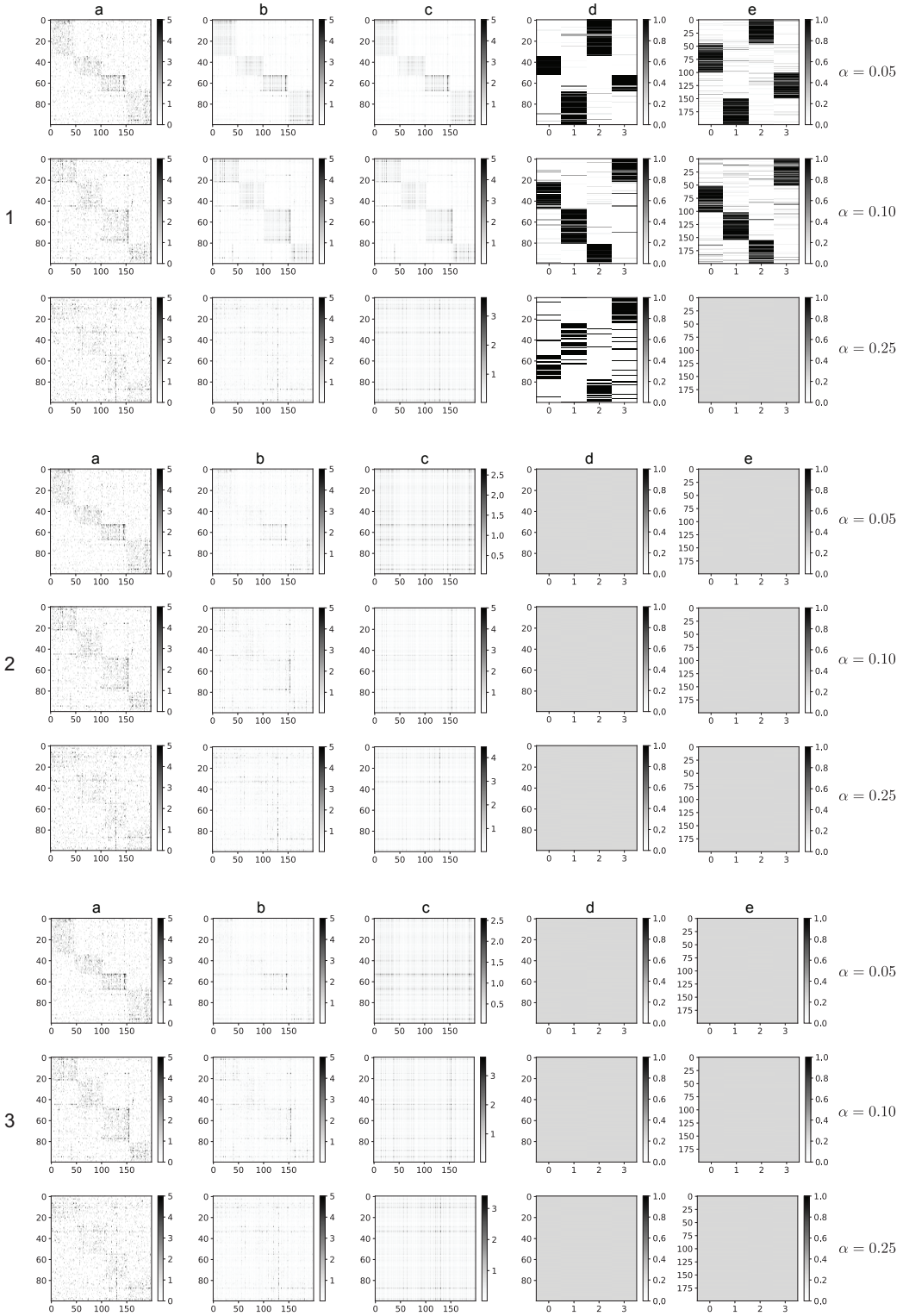

Figure 4.4: The simulated bi-adjacency matrices in column (a) are generated with node degree heterogeneity, and the algorithm is performed with degree correction. See Figure 4.3 for the detailed legend.

maintaining the optimization scalability remains an open research direction in the variational inference field [157].

On the other hand, it has been shown that CAVI is closely related to Gibbs sampler [163, 164], which enjoys rigorous statistical properties and can asymptotically sample from the exact posterior distribution; but the Gibbs sampler is computationally intensive. In future work, we may consider comparing these two techniques on a small data set. When it comes to massive data sets, we may consider upgrading variational inference to include statistical optimization [165] or parallel computation. Besides, in our current Bayesian generative model, the hyperparameters like the connectivity matrix $\boldsymbol{\Omega}$ are assumed to be fixed. In a more general scenario, we may consider a conjugate prior for $\boldsymbol{\Omega}$ to smooth the connectivity probabilities between communities [7].

Also, following the current simulation study, we can test the robustness of the algorithm performance by varying the level of edge sparsity, etc. Two types of evaluation criteria can be used: extended NMI and Omega index (see Section 3.6) by first thresholding each mixed membership vector $\hat{\boldsymbol{\pi}}_i$; or calculating a relative error rate such as $\|\hat{\boldsymbol{\Pi}} - \boldsymbol{\Pi}\|_F / \|\boldsymbol{\Pi}\|_F$, where we denote $\boldsymbol{\Pi} := [\dots \boldsymbol{\pi}_i \dots]^\mathsf{T}$ and $\hat{\boldsymbol{\Pi}} := [\dots \hat{\boldsymbol{\pi}}_i \dots]^\mathsf{T}$. Note that for the bipartite network case, we will evaluate the performance for nodes in each type separately and then take an average between the two.

4.5 Code

The code for implementing the algorithm is available on GitHub: `https://github.com/edensunyidan/Bipartite-mixed-membership-SBM`

4.6 Acknowledgements

I would like to acknowledge the original idea of updating the node degree parameters $\boldsymbol{\theta}$ by the Douglas-Rachford splitting method is proposed in Razaee et al. [84]. I would like to thank

Dr. Jingyi Jessica Li, Dr. Mark S. Handcock, Dr. Yingnian Wu, Dr. Sriram Sankararaman, Kexin Li, Ruocheng Jiang, and Guan'ao Yan for the valuable suggestions and discussions for this chapter.

APPENDIX

4.A Details of updating ϕ

Here we closely follow the statement and proof of Lemma 1 from [152]. We use $\mathcal{P}_K := \left\{\mathbf{p} \mid \mathbf{p} \in \mathbb{R}_+^K,\ \|\mathbf{p}\|_1 = 1\right\}$ to denote the set of K-dimensional probability vectors.

Lemma 4.A.1 (Razaee et al. [152], Lemma 1)**.** For any vector $\mathbf{a} = (a_1, \ldots, a_K)^\mathsf{T} \in \mathbb{R}^K$, define a function $f_\mathbf{a} : \mathcal{P}_K \to \mathbb{R}$ by $f_\mathbf{a}(\mathbf{p}) := \sum_{k=1}^K p_k(a_k - \log p_k)$. Then the solution of maximizing $f_\mathbf{a}(\mathbf{p})$ over $\mathcal{P}_K$ is given by:

$$\arg\max_{\mathbf{p}\in\mathcal{P}_K} f_\mathbf{a}(\mathbf{p}) = C\left(e^{a_1}, \ldots, e^{a_K}\right) \tag{4.27}$$

where $C = 1/\sum_k e^{a_k}$

Proof.

$$\begin{aligned} f_\mathbf{a}(\mathbf{p}) &= \sum_{k=1}^K p_k(a_k - \log p_k) \\ &= -\sum_{k=1}^K p_k \log \frac{p_k}{e^{a_k}} \\ &= -\sum_{k=1}^K p_k \log \frac{p_k}{Ce^{a_k}} - \log C \end{aligned}$$

where the first term in the last equation is the KL divergence between two discrete probability distributions, and the function achieves the maximization when the KL divergence is equal to 0, as indicated by (4.27). □

4.B Details of updating $\boldsymbol{\theta}$

In this section, we briefly review the concepts in convex optimization used in this chapter. We mainly focus on the Douglas-Rachford splitting method used in Section 4.2.2. There are many good reference books, including those by [160, 166, 167].

4.B.1 Definitions

A set $\mathcal{C}$ is convex if the line segment between any two points in $\mathcal{C}$ lies in $\mathcal{C}$, i.e., $\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{C}$ and $\forall \rho \in [0,1]$, $\rho\mathbf{x}_1 + (1-\rho)\mathbf{x}_2 \in \mathcal{C}$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if the domain of f (denoted by **dom** f) is a convex set if $\forall \mathbf{x}_1, \mathbf{x}_2 \in$ **dom** f and $\forall \rho \in [0,1]$, $f(\rho\mathbf{x}_1 + (1-\rho)\mathbf{x}_2) \leq \rho f(\mathbf{x}_1) + (1-\rho)f(\mathbf{x}_2)$. A convex function f is **proper convex** if $f(\mathbf{x}) < +\infty$ for at least one $\mathbf{x}$ and $f(\mathbf{x}) > -\infty$ for every $\mathbf{x}$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is **closed** if for each $\alpha \in \mathbb{R}$, the sublevel set $\{\mathbf{x} \in \textbf{dom } f \mid f(\mathbf{x}) \leq \alpha\}$ is a closed set.

Given a function f, the **proximal mapping** (or prox-operator) of f is defined by:

$$\textbf{prox}_f(\mathbf{v}) = \arg\min_{\mathbf{x}} \left(f(\mathbf{x}) + \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2 \right) \tag{4.28}$$

The proximal operator of the scaled function λf, where $\lambda > 0$ is defined by

$$\textbf{prox}_{\lambda f}(\mathbf{v}) = \arg\min_{\mathbf{x}} \left(f(\mathbf{x}) + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{v}\|_2^2 \right) \tag{4.29}$$

Remark. The proximal operator is summative, meaning that for a separable function $f(\mathbf{x}_1, \mathbf{x}_2) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2)$, then $\textbf{prox}_f(\mathbf{x}_1,\, \mathbf{x}_2) = \Big(\textbf{prox}_{f_1}(\mathbf{x}_1),\, \textbf{prox}_{f_2}(\mathbf{x}_2)\Big)$.

An example:

$$f(x) = \sum_{i=1}^{n} -a_i \log x_i + b_i x_i, \quad (\textbf{prox}_{tf}(v))_i = \frac{-(b_i t - v_i) + \sqrt{(b_i t - v_i)^2 + 4a_i t}}{2},\ i = 1, \ldots, n$$

An example: when f is the indicator function of a set $\mathcal{C}$:

$$f(\mathbf{x}) = I_{\mathcal{C}}(\mathbf{x}) := \begin{cases} 0 & \text{if } \mathbf{x} \in \mathcal{C} \\ +\infty & \text{if } \mathbf{x} \notin \mathcal{C} \end{cases} \tag{4.30}$$

Then the proximal operator $\textbf{prox}_f(v)$ reduces to Euclidean projection onto $\mathcal{C}$:

$$\textbf{prox}_f(\mathbf{v}) = \arg\min_{\mathbf{x} \in \mathcal{C}} \|\mathbf{x} - \mathbf{v}\|_2^2 := \Pi_{\mathcal{C}}(\mathbf{v}) \tag{4.31}$$

In particular, if $\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{A} \in \mathbb{R}^{m \times n}\}$. Suppose $m < n$ and $\mathbf{A}$ has full rank, then:

$$\begin{aligned} \Pi_{\mathcal{C}}(\mathbf{v}) &= \left(\mathbf{I} - \mathbf{A}^\mathsf{T}(\mathbf{A}\mathbf{A}^\mathsf{T})^{-1}\mathbf{A}\right)\mathbf{v} + \mathbf{A}^\mathsf{T}(\mathbf{A}\mathbf{A}^\mathsf{T})^{-1}\mathbf{b} \\ &= \mathbf{v} - \mathbf{A}^\mathsf{T}(\mathbf{A}\mathbf{A}^\mathsf{T})^{-1}(\mathbf{A}\mathbf{v} - \mathbf{b}) \end{aligned}$$

4.B.2 Douglas-Rachford splitting method

Consider the problem

$$\text{minimize } f(\mathbf{x}) + g(\mathbf{x})$$

where the extended-valued functions $f, g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ are closed convex. f and g can be used to encode constraints on the variable $\mathbf{x}$. Then Douglas-Rachford splitting is (see Section 3.2 in [167] for details):

$$\begin{aligned}
\mathbf{x}^{t+1} &:= \mathbf{prox}_{\lambda f}(\mathbf{z}^t - \mathbf{u}^t) \\
\mathbf{z}^{t+1} &:= \mathbf{prox}_{\lambda g}(\mathbf{x}^{t+1} + \mathbf{u}^t) \\
\mathbf{u}^{t+1} &:= \mathbf{u}^t + \mathbf{x}^{t+1} - \mathbf{z}^{t+1}
\end{aligned}$$

where t is an iteration index. This algorithm is useful when the proximal operators of f and g can be efficiently evaluated but the proximal operator for $f + g$ is not easy to evaluate.

Remark. Here we present a special case. When g is the indicator function of a closed convex set $\mathcal{C}$, its proximal operator $\mathbf{prox}_{\lambda g}$ reduces to projection onto $\mathcal{C}$ (see (4.31)). In this case, Douglas-Rachford splitting is a method for solving the generic convex constrained problem of minimizing f over $\mathcal{C}$ that only uses the proximal operator of the objective and projection onto the constraint set. And it directly matches to the situation of (4.21) in Section 4.2.2. Given this, the Douglas-Rachford splitting becomes:

$$\begin{aligned}
\mathbf{x}^{t+1} &:= \mathbf{prox}_{\lambda f}(\mathbf{z}^t - \mathbf{u}^t) \\
\mathbf{z}^{t+1} &:= \Pi_{\mathcal{C}}(\mathbf{x}^{t+1} + \mathbf{u}^t) \\
\mathbf{u}^{t+1} &:= \mathbf{u}^t + \mathbf{x}^{t+1} - \mathbf{z}^{t+1}
\end{aligned}$$

4.C The power law and the Pareto distribution

In statistics, the power law usually used to describe the distribution changing inversely as a power of the variable. A distribution whose cumulative format exhibits power law is known as

a Pareto distribution (see [121, 168] for the detailed review). In network analysis, the Pareto distribution is widely used to generate the node degree distribution[45, 152]. Specifically, the probability density function of a Pareto distribution (Pareto$(a, x_{\min})$) is:

$$p(x) = \begin{cases} \frac{a x_{\min}^a}{x^{a+1}} & \text{if } x \geq x_{\min} \\ 0 & \text{if } x < x_{\min} \end{cases}$$

where $x_{\min}$ is the positive minimum bound of the support, and a is a positive parameter. a and $x_{\min}$ are called as shape and scale parameters. The expected value of a random variable following a Pareto distribution Pareto$(a, x_{\min})$ is:

$$\mathbb{E}(x) = \begin{cases} \infty & \text{if } a \leq 1 \\ \frac{a x_{\min}}{a-1} & \text{if } a > 1 \end{cases}$$

The variance of a random variable following a Pareto distribution is:

$$\text{Var}(x) = \begin{cases} \infty & \text{if } a \in (1, 2] \\ \left(\frac{x_{\min}}{a-1}\right)^2 \frac{a}{a-2} & \text{if } a > 2 \end{cases}$$

If $a \leq 1$, the variance does not exist.

In our simulation study (Section 4.3), we generate the θ_i from a Pareto distribution with average value equal to 1, i.e., $\theta_i \overset{\text{iid}}{\sim} \text{Pareto}(a, \frac{a-1}{a})$. This is because if we assume θ_i is a random variable, then the constraint (4.9) can be viewed as $\mathbb{E}[\theta_i] = 1$. In this case, the variance is $[a(a-2)]^{-1}$ and we set $a = 2$ to maximize the degree heterogeneity.

CHAPTER 5

Discussion

In this thesis, we focus on developing bipartite network community detection methods, covering both algorithm-based and SBM-based approaches.

In Chapter 2, we develop an algorithm-based bipartite spectral clustering method BiTSC to identify informative and tight clusters without forcing all nodes to be grouped into one of the clusters. It combines the node covariate information with edge information together to enhance the algorithm performance, given the underlying assumption that node covariate information is consistent with the community structure information; that is, node covariates of each type within the same cluster are more similar. In our current work, we interpret the efficacy of BiTSC in the context of gene orthology network in comparative genomics. In general, BiTSC applies to any bipartite network formulation with node covariates.

In Chapter 3, we focus on the SBM-based approaches for community detection. In particular, we introduce the existing extensions of the SBM, including degree-correction SBM, mixed membership SBMs, and bipartite SBMs. We discuss and summarize the relevant existing methods. In Chapter 4, we propose a new model by combining the mixed membership SBM with the bipartite SBM and use the idea of CAVI and empirical Bayes to fit the new model. In this model, each node is assigned with a soft membership indicating its participation weight in the corresponding community. Introducing the soft membership in the generative model makes the clustering more interpretable and informative, as many real-world bipartite networks contain overlapping communities. In general, we can extend this model to include more complex characteristics such as edge sparsity, noisy nodes, and node covariates. However, this may result in overfitting and increase computational complexity in implementation. In

this case, identifiability would also be a concern as more parameters were introduced into the formulation. Besides, recent works [23, 126, 169] have studied the theoretical justification of the mean-field coordinate ascent variational inference in community detection under the SBM, which would greatly benefit future studies on the asymptotic consistency of our algorithm. For the practical implementations, the user's domain knowledge and prior information on the data are essential for deciding which type of models or methods works best for their situation.

BIBLIOGRAPHY

[1] Emmanuel Abbe. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018.

[2] L. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 11(9):1074–1085, 1992.

[3] Shi Jianbo and Malik Jitendra. Normalized cuts and image segmentation. *IEEE*, 22(8):888–905, 2000.

[4] Eric Davidson and Michael Levin. Gene regulatory networks. *Proceedings of the National Academy of Sciences*, 102(14):4935–4935, 2005.

[5] Mukesh Bansal, Giusy Della Gatta, and Diego di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, 2006.

[6] Michael Hecker, Sandro Lambeck, Susanne Toepfer, Eugene Van Someren, and Reinhard Guthke. Gene regulatory network inference: data integration in dynamic models-a review. *Biosystems*, 96(1):86–103, 2009.

[7] Edoardo Airoldi, David Blei, Eric Xing, and Stephen Fienberg. A latent mixed membership model for relational data. In *Proceedings of the 3rd International Workshop on Link Discovery*, LinkKDD '05, pages 82–89, New York, NY, USA, 2005. ACM.

[8] Rohit Singh, Jinbo Xu, and Bonnie Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, 105(35):12763–12768, 2008.

[9] Behnam Neyshabur, Ahmadreza Khadem, Somaye Hashemifar, and Seyed Shahriar Arab. NETAL: a new graph-based method for global alignment of protein–protein interaction networks. *Bioinformatics*, 29(13):1654–1662, 05 2013.

[10] Mark Handcock, Adrian Raftery, and Jeremy Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170:301 – 354, 03 2007.

[11] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., USA, 2010.

[12] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1–44, 2016. Community detection in networks: A user guide.

[13] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 01 1977.

[14] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[15] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. *On Spectral Clustering: Analysis and an algorithm*, pages 849–856. MIT Press, Cambridge, MA, USA, 2001.

[16] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

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

[18] Arash A. Amini and Elizaveta Levina. On semidefinite relaxations for the block model. *Ann. Statist.*, 46(1):149–179, 02 2018.

[19] Yingjie Fei and Yudong Chen. Achieving the bayes error rate in stochastic block model by sdp, robustly. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 1235–1269, Phoenix, USA, 25–28 Jun 2019. PMLR.

[20] Peter J. Bickel and Aiyou Chen. A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009.

[21] Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

[22] Alain Celisse, Jean-Jacques Daudin, and Laurent Pierre. Consistency of maximum-likelihood and variational estimators in the stochastic block model. *Electron. J. Statist.*, 6:1847–1899, 2012.

[23] Peter Bickel, David Choi, Xiangyu Chang, and Hai Zhang. Asymptotic normality of maximum likelihood and its variational approximation for stochastic blockmodels. *Ann. Statist.*, 41(4):1922–1943, 08 2013.

[24] Arash A. Amini, Aiyou Chen, Peter J. Bickel, and Elizaveta Levina. Pseudo-likelihood methods for community detection in large sparse networks. *Ann. Statist.*, 41(4):2097–2122, 08 2013.

[25] Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *Ann. Statist.*, 40(4):2266–2292, 08 2012.

[26] Jiashun Jin. Fast community detection by score. *Ann. Statist.*, 43(1):57–89, 02 2015.

[27] Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, pages 3120–3128. Curran Associates, Inc., 2013.

[28] Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference*

on Knowledge Discovery and Data Mining, KDD '01, pages 269–274, New York, NY, USA, 2001. ACM.

[29] Zhixin Zhou and Arash A Amini. Optimal bipartite network clustering. *Journal of Machine Learning Research*, 21(40):1–68, 2020.

[30] Georgios A Pavlopoulos, Panagiota I Kontou, Athanasia Pavlopoulou, Costas Bouyioukos, Evripides Markou, and Pantelis G Bagos. Bipartite graphs in systems biology and medicine: a survey of methods and applications. *GigaScience*, 7(4), 02 2018. giy014.

[31] Paola Pesántez-Cabrera and Ananth Kalyanaraman. Efficient detection of communities in biological bipartite networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(1):258–271, Jan 2019.

[32] Zichen Wang, Mu Zhou, and Corey Arnold. Toward heterogeneous information fusion: bipartite graph convolutional networks for in silico drug repurposing. *Bioinformatics*, 36:i525–i533, 07 2020.

[33] Tao Zhou, Jie Ren, Matú š Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Phys. Rev. E*, 76:046115, Oct 2007.

[34] Xin Li and Hsinchun Chen. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. *Decision Support Systems*, 54(2):880–890, 2013.

[35] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83:016107, Jan 2011.

[36] Christopher Aicher, Abigail Z. Jacobs, and Aaron Clauset. Learning latent block structure in weighted networks. *Journal of Complex Networks*, 3(2):221–248, 06 2014.

[37] Xin Qian, Yudong Chen, and Andreea Minca. Clustering degree-corrected stochastic block model with outliers. *CoRR*, abs/1906.03305, 2019.

[38] M. E. J. Newman and Aaron Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(1):11863, 2016.

[39] N. Binkiewicz, J. T. Vogelstein, and K. Rohe. Covariate-assisted spectral clustering. *Biometrika*, 104(2):361–377, 03 2017.

[40] Edoardo M. Airoldi, David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg. Introduction to mixed membership models and methods. In *Handbook of Mixed Membership Models and Their Applications*, pages 3–13. 2014.

[41] Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping clustering methods for networks. In *Handbook of Mixed Membership Models and Their Applications*, pages 547–567. 2014.

[42] Tzu-Chi Yen and Daniel B. Larremore. Community detection in bipartite networks with stochastic block models. *Phys. Rev. E*, 102:032309, Sep 2020.

[43] Tianxi Li, Lihua Lei, Sharmodeep Bhattacharyya, Koen Van den Berge, Purnamrita Sarkar, Peter J. Bickel, and Elizaveta Levina. Hierarchical community detection by recursive partitioning. *Journal of the American Statistical Association*, 0(0):1–18, 2020.

[44] Marc Tarrés-Deulofeu, Antonia Godoy-Lorite, Roger Guimerà, and Marta Sales-Pardo. Tensorial and bipartite block models for link prediction in layered networks and temporal networks. *Phys. Rev. E*, 99:032307, Mar 2019.

[45] Yudong Chen, Xiaodong Li, and Jiaming Xu. Convexified modularity maximization for degree-corrected stochastic block models. *Ann. Statist.*, 46(4):1573–1602, 08 2018.

[46] Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. Community detection in degree-corrected block models. *Ann. Statist.*, 46(5):2153–2185, 10 2018.

[47] Maoying Qiao, Jun Yu, Wei Bian, Qiang Li, and Dacheng Tao. Adapting stochastic block models to power-law degree distributions. *IEEE Transactions on Cybernetics*, 49:626–637, 2019.

[48] Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *J. Mach. Learn. Res.*, 9:1981–2014, June 2008.

[49] Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the french political blogosphere. *Ann. Appl. Stat.*, 5(1):309–336, 03 2011.

[50] Brian Ball, Brian Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Phys. Rev. E*, 84:036103, Sep 2011.

[51] Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. Overlapping community detection using bayesian non-negative matrix factorization. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 83:066114, 06 2011.

[52] Animashree Anandkumar, Rong Ge, Daniel Hsu, and Sham M. Kakade. A tensor approach to learning mixed membership community models. *J. Mach. Learn. Res.*, 15(1):2239–2312, January 2014.

[53] Emilie Kaufmann, Thomas Bonald, and Marc Lelarge. A spectral algorithm with additive clustering for the recovery of overlapping communities in networks. In Ronald Ortner, Hans Ulrich Simon, and Sandra Zilles, editors, *Algorithmic Learning Theory*, pages 355–370, Cham, 2016. Springer International Publishing.

[54] Xueyu Mao, Purnamrita Sarkar, and Deepayan Chakrabarti. On mixed memberships and symmetric nonnegative matrix factorizations. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2324–2333, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[55] Maxim Panov, Konstantin Slavnov, and Roman Ushakov. Consistent estimation of mixed memberships with successive projections. pages 53–64, 11 2018.

[56] Xueyu Mao, Purnamrita Sarkar, and Deepayan Chakrabarti. Estimating mixed

memberships with sharp eigenvector deviations. *Journal of the American Statistical Association*, 0(0):1–13, 2020.

[57] Jiashun Jin, Zheng Tracy Ke, and Shengming Luo. Estimating network memberships by simplex vertex hunting, 2017.

[58] Yuan Zhang, Elizaveta Levina, and Ji Zhu. Detecting overlapping communities in networks using spectral methods. *SIAM Journal on Mathematics of Data Science*, 2(2):265–283, 2020.

[59] Jesús Arroyo and Elizaveta Levina. Overlapping community detection in networks via sparse spectral decomposition, 2020.

[60] Daniel B. Larremore, Aaron Clauset, and Abigail Z. Jacobs. Efficiently inferring community structure in bipartite networks. *Phys. Rev. E*, 90:012805, Jul 2014.

[61] H. K. Lee, A. K. Hsu, J. Sajdak, J. Qin, and P. Pavlidis. Coexpression analysis of human genes across many microarray data sets. *Genome Research*, 2004.

[62] Jinahua Ruan, Dean K Angela, and Weixiong Zhang. A general co-expression network-based approach to gene expression analysis: comparison and applications. *BMC Systems Biology*, 2010.

[63] Joshua M. Stuart, Eran Segal, Daphne Koller, and Stuart K. Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–255, 2003.

[64] Sven Bergmann, Jan Ihmels, and Naama Barkai. Similarities and differences in genome-wide expression data of six organisms. *PLOS Biology*, 2(1), 12 2003.

[65] Ali Mortazavi, Brian A Williams, Kenneth McCue, Lorian Schaeffer, and Barbara Wold. Mapping and quantifying mammalian transcriptomes by rna-seq. *Nature Methods*, 5, 2008.

[66] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10:57–63, 2009.

[67] Hai-Son Le, Zoltán N. Oltvai, and Ziv Bar-Joseph. Cross-species queries of large gene expression databases. *Bioinformatics*, 26(19):2416–2423, 08 2010.

[68] Julia F. Söllner, German Leparc, Tobias Hildebrandt, Holger Klein, Leo Thomas, Elia Stupka, and Eric Simon. An rna-seq atlas of gene expression in mouse and rat normal tissues. *Scientific Data*, 4:170185, 12 2017.

[69] Wataru Fujibuchi, Hiroyuki Ogata, Hideo Matsuda, and Minoru Kanehisa. Automatic detection of conserved gene clusters in multiple genomes by graph comparison and P-quasi grouping. *Nucleic Acids Research*, 28(20):4029–4036, 10 2000.

[70] D. Dede and H. Oğul. A three-way clustering approach to cross-species gene regulation analysis. In *2013 IEEE INISTA*, pages 1–5, Albena, Bulgaria, June 2013. IEEE.

[71] Erik Kristiansson, Tobias Österlund, Lina Gunnarsson, Gabriella Arne, D. G. Joakim Larsson, and Olle Nerman. A novel method for cross-species gene expression analysis. *BMC Bioinformatics*, 14(1):70, Feb 2013.

[72] Peter H. Sudmant, Maria S. Alexis, and Christopher B. Burge. Meta-analysis of rna-seq expression data across species, tissues and studies. *Genome Biology*, 16(1):287, 2015.

[73] Wei Chen, Xiayu Xia, Nan Song, Ying Wang, Hua Zhu, Wei Deng, Qi Kong, Xianmin Pan, and Chuan Qin. Cross-species analysis of gene expression and function in prefrontal cortex, hippocampus and striatum. *PLOS ONE*, 11(10):1–18, 10 2016.

[74] Fabian Schreiber, Mateus Patricio, Matthieu Muffato, Miguel Pignatelli, and Alex Bateman. Treefam v9: a new website, more species and orthology-on-the-fly. *Nucleic acids research*, 42(D1):D922–D925, 2013.

[75] Huaiyu Mi, Anushya Muruganujan, Dustin Ebert, Xiaosong Huang, and Paul D

Thomas. Panther version 14: more genomes, a new panther go-slim and improvements in enrichment analysis tools. *Nucleic acids research*, 47(D1):D419–D426, 2018.

[76] Sarah A Teichmann and M.Madan Babu. Conservation of gene co-regulation in prokaryotes and eukaryotes. *Trends in Biotechnology*, 20(10):407 – 410, 2002.

[77] Vera van Noort, Berend Snel, and Martijn A. Huynen. Predicting gene function by conserved co-expression. *Trends in Genetics*, 19(5):238 – 242, 2003.

[78] Berend Snel, Martijn A. Huynen, and Vera van Noort. Gene co-regulation is highly conserved in the evolution of eukaryotes and prokaryotes. *Nucleic Acids Research*, 32(16):4725–4731, 01 2004.

[79] Jun Cai, Dan Xie, Zhewen Fan, Hiram Chipperfield, John Marden, Wing H. Wong, and Sheng Zhong. Modeling co-expression across species for complex traits: Insights to the difference of human and mouse embryonic stem cells. *PLOS Computational Biology*, 6(3):1–10, 03 2010.

[80] Jiangwen Sun, Jinbo Bi, Xiuchun Tian, and Zongliang Jiang. A cross-species bi-clustering approach to identifying conserved co-regulated genes. *Bioinformatics*, 32(12):i137–i146, 06 2016.

[81] Koon-Kiu Yan, Daifeng Wang, Joel Rozowsky, Henry Zheng, Chao Cheng, and Mark Gerstein. Orthoclust: an orthology-based network framework for clustering data across multiple species. *Genome Biology*, 2014.

[82] Vikram Saraph and Tijana Milenković. MAGNA: Maximizing Accuracy in Global Network Alignment. *Bioinformatics*, 30(20):2931–2940, 07 2014.

[83] Yihan Sun, Joseph Crawford, Jie Tang, and Tijana Milenković. Simultaneous optimization of both node and edge conservation in network alignment via wave. In Mihai Pop and Hélène Touzet, editors, *Algorithms in Bioinformatics*, pages 16–39, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[84] Zahra S. Razaee, Arash A. Amini, and Jingyi Jessica Li. Matched bipartite block model with covariates. *Journal of Machine Learning Research*, 20:1–44, 2019.

[85] Feiping Nie, Xiaoqian Wang, Cheng Deng, and Heng Huang. Learning a structured optimal bipartite graph for co-clustering. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[86] J. J. Whang, P. Rai, and I. S. Dhillon. Stochastic blockmodel with cluster overlap, relevance selection, and similarity-based smoothing. In *2013 IEEE 13th International Conference on Data Mining*, pages 817–826, Dec 2013.

[87] Zahra Razaee. *Community Detection in Networks with Node Covariates*. PhD thesis, University of California, Los Angeles, 7 2017.

[88] Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52:91–118, 2003.

[89] George C Tseng and Wing H Wong. Tight clustering: A resampling-based approach for identifying stable and tight patterns in data. *Biometrics*, 61(1):10–16, 2005.

[90] Stephen C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, Sep 1967.

[91] Susumu Ohno. *Evolution by Gene Duplication*. Springer-Verlag Berlin Heidelberg, 1970.

[92] Roman L. Tatusov, Eugene V. Koonin, and David J. Lipman. A genomic perspective on protein families. *Science*, 278(5338):631–637, 1997.

[93] Eugene V. Koonin. Orthologs, paralogs, and evolutionary genomics. *Annual Review of Genetics*, 39(1):309–338, 2005. PMID: 16285863.

[94] Anbupalam Thalamuthu, Indranil Mukhopadhyay, Xiaojing Zheng, and George C. Tseng. Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics*, 22(19):2405–2412, 07 2006.

[95] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83:016107, Jan 2011.

[96] M.B. Gerstein et al. Comparative analysis of the transcriptome across distant species. *Nature*, 512(7515):445–448, 2014.

[97] Heng Li, Avril Coghlan, Jue Ruan, Lachlan James Coin, Jean-Karim Hériché, Lara Osmotherly, Ruiqiang Li, Tao Liu, Zhang Zhang, Lars Bolund, Gane Ka-Shu Wong, Weimou Zheng, Paramvir Dehal, Jun Wang, and Richard Durbin. Treefam: a curated database of phylogenetic trees of animal gene families. *Nucleic Acids Research*, 34:572–580, 01 2006.

[98] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[99] Jingyi Jessica Li, Haiyan Huang, Peter J. Bickel, and Steven E. Brenner. Comparison of d. melanogaster and c. elegans developmental stages, tissues, and cells by modencode rna-seq data. *Genome Research*, 2014.

[100] Isabella Zwiener, Barbara Frisch, and Harald Binder. Transforming rna-seq data to improve the performance of prognostic gene signatures. *PLOS ONE*, 9(1):1–13, 01 2014.

[101] Frida Danielsson, Tojo James, David Gomez-Cabrero, and Mikael Huss. Assessing the consistency of public human tissue rna-seq data sets. *Briefings in Bioinformatics*, 16:941–949, 2015.

[102] Mihaela Pertea, Daehwan Kim, Geo M Pertea, Jeffrey T Leek, and Steven L Salzberg. Transcript-level expression analysis of rna-seq experiments with hisat, stringtie and ballgown. *Nature Protocols*, 11(9):1605–1667, 2016.

[103] Marc Carlson. *GO.db: A set of annotation maps describing the entire Gene Ontology*, 2019. R package version 3.8.2.

[104] Adrian Alexa and Jorg Rahnenfuhrer. *topGO: Enrichment Analysis for Gene Ontology*, 2019. R package version 2.36.0.

[105] Jian-Ping Mei, Chee-Keong Kwoh, Peng Yang, Xiao-Li Li, and Jie Zheng. Drug-target interaction prediction by learning from local information and neighbors. *Bioinformatics*, 29(2):238–245, 2013.

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

[107] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[108] Yidan Eden Sun, Heather J Zhou, and Jingyi Jessica Li. Bipartite Tight Spectral Clustering (BiTSC) Algorithm for Identifying Conserved Gene Co-clusters in Two Species. *Bioinformatics*, 08 2020. btaa741.

[109] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[110] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[111] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[112] Frank Cowell. *Measuring Inequality*. Oxford University Press, 3 edition, 2011.

[113] Tom A. B. Snijders and Krzysztof Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, 14:75–100, 1997.

[114] T. Tony Cai and Xiaodong Li. Robust and computationally feasible community detection in the presence of arbitrary outlier nodes. *Ann. Statist.*, 43(3):1027–1059, 06 2015.

[115] M. E. J. Newman. Network structure from rich but noisy data. *Nature Physics*, 14(6):542–545, 2018.

[116] Anton Valouev, David S Johnson, Andreas Sundquist, Catherine Medina, Elizabeth Anton, Serafim Batzoglou, Richard M Myers, and Arend Sidow. Genome-wide analysis of transcription factor binding sites based on chip-seq data. *Nature Methods*, 5:829—834, 2005.

[117] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Comput. Surv.*, 45(4), August 2013.

[118] Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Model selection in overlapping stochastic block models. *Electron. J. Statist.*, 8(1):762–794, 2014.

[119] Ernesto Estrada and Juan A. Rodríguez-Velázquez. Spectral measures of bipartivity in complex networks. *Phys. Rev. E*, 72:046105, Oct 2005.

[120] Amin Coja-Oghlan and André Lanka. Finding planted partitions in random graphs with general degree distributions. *SIAM J. Discret. Math.*, 23(4):1682–1714, November 2009.

[121] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.

[122] Patrick O. Perry and Patrick J. Wolfe. Null models for network data, 2012.

[123] J. Yang and J. Leskovec. Community-affiliation graph model for overlapping network community detection. In *2012 IEEE 12th International Conference on Data Mining*, pages 1170–1175, 2012.

[124] Weihong Huang, Yan Liu, and Yuguo Chen. Mixed membership stochastic blockmodels for heterogeneous networks. *Bayesian Anal.*, 2018. Advance publication.

[125] E. K. Kao, S. T. Smith, and E. M. Airoldi. Hybrid mixed-membership blockmodel for inference on realistic network interactions. *IEEE Transactions on Network Science and Engineering*, 6(3):336–350, 2019.

[126] Anderson Y. Zhang and Harrison H. Zhou. Theoretical and computational guarantees of mean field variational inference for community detection. *Ann. Statist.*, 48(5):2575–2598, 10 2020.

[127] Prem K. Gopalan and David M. Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

[128] Fei Wang, Tao Li, Xin Wang, Shenghuo Zhu, and Chris Ding. Community discovery using nonnegative matrix factorization. *Data Min. Knowl. Discov.*, 22:493–521, 05 2011.

[129] Xueyu Mao, Purnamrita Sarkar, and Deepayan Chakrabarti. Overlapping clustering models, and one (class) svm to bind them all. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2126–2136. Curran Associates, Inc., 2018.

[130] Chris Ding, Xiaofeng He, and Horst D. Simon. *On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering*, pages 606–610.

[131] Da Kuang, Chris Ding, and Haesun Park. *Symmetric Nonnegative Matrix Factorization for Graph Clustering*, pages 106–117.

[132] Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878 – 1915, 2011.

[133] Purnamrita Sarkar and Peter J. Bickel. Role of normalization in spectral clustering for stochastic blockmodels. *The Annals of Statistics*, 43(3):962 – 990, 2015.

[134] Antony Joseph and Bin Yu. Impact of regularization on spectral clustering. *The Annals of Statistics*, 44(4):1765 – 1791, 2016.

[135] Patrick Rubin-Delanchy, Carey E. Priebe, and Minh Tang. Consistency of adjacency spectral embedding for the mixed membership stochastic blockmodel, 2017.

[136] Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Structures & Algorithms*, 27(2):251–275, 2005.

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

[138] Mário César Ugulino Araújo, Teresa Cristina Bezerra Saldanha, Roberto Kawakami Harrop Galvão, Takashi Yoneyama, Henrique Caldas Chame, and Valeria Visani. The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. *Chemometrics and Intelligent Laboratory Systems*, 57(2):65–73, 2001.

[139] N. Gillis and S. A. Vavasis. Fast and robust recursive algorithmsfor separable nonnegative matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):698–714, 2014.

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

[141] In Song Kim and Dmitriy Kunisky. Mapping political communities: A statistical analysis of lobbying networks in legislative politics. *Political Analysis*, page 1–20, 2020.

[142] Huan Qing and Jingli Wang. Bipartite mixed membership stochastic blockmodel, 2021.

[143] Roger Guimerà, Marta Sales-Pardo, and Luís A. Nunes Amaral. Module identification in bipartite and directed networks. *Phys. Rev. E*, 76:036102, Sep 2007.

[144] Michael J. Barber. Modularity and community detection in bipartite networks. *Phys. Rev. E*, 76:066102, Dec 2007.

[145] Karl Rohe, Tai Qin, and Bin Yu. Co-clustering directed graphs to discover asymmetries and directional communities. *Proceedings of the National Academy of Sciences*, 113(45):12679–12684, 2016.

[146] Tao Wu, Austin R Benson, and David F Gleich. General tensor spectral co-clustering for higher-order data. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29, pages 2559–2567. Curran Associates, Inc., 2016.

[147] Zhixin Zhou and Arash A Amini. Analysis of spectral clustering algorithms for community detection: the general bipartite setting. *J. Mach. Learn. Res.*, 20:47–1, 2019.

[148] Zhixin Zhou and Ping Li. Rate optimal chernoff bound and application to community detection in the stochastic block models. *Electronic Journal of Statistics*, 14(1):1302–1347, 2020.

[149] Y. Chen, L. Wang, and M. Dong. Non-negative matrix factorization for semisupervised heterogeneous data coclustering. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1459–1474, 2010.

[150] Hua Wang, Feiping Nie, Heng Huang, and Fillia Makedon. Fast nonnegative matrix tri-factorization for large-scale data co-clustering. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, page 1553–1558. AAAI Press, 2011.

[151] Woosang Lim, Rundong Du, and Haesun Park. Codinmf: Co-clustering of directed graphs via nmf, 2018.

[152] Zahra S. Razaee, Arash A. Amini, and Jingyi Jessica Li. Matched bipartite block model with covariates. *Journal of Machine Learning Research*, 20(34):1–44, 2019.

[153] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, mar 2009.

[154] LM Collins and CW Dent. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. *Multivariate behavioral research*, 23(2):231—242, April 1988.

[155] Antonia Godoy-Lorite, Roger Guimerà, Cristopher Moore, and Marta Sales-Pardo. Accurate and scalable social recommendation using mixed-membership stochastic block models. *Proceedings of the National Academy of Sciences*, 113(50):14207–14212, 2016.

[156] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[157] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[158] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):1183–233, 1999.

[159] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.

[160] S. Boyd, N. Parikh, and E. Chu. *Distributed Optimization and Statistical Learning Via the Alternating Direction Method of Multipliers*. Foundations and Trends in Machine Learning Series. Now Publishers, 2011.

[161] G. Ronning. Maximum likelihood estimation of dirichlet distributions. *Journal of Statistical Computation and Simulation*, 32(4):215–221, 1989.

[162] Tom Minka. Estimating a dirichlet distribution. September 2000.

[163] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984.

[164] Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

[165] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013.

[166] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004.

[167] N. Parikh and S. Boyd. *Proximal Algorithms*. Foundations and Trends in Optimization. Now Publishers, 2013.

[168] MEJ Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46(5):323–351, 2005.

[169] Yixin Wang and David M. Blei. Frequentist consistency of variational bayes. *Journal of the American Statistical Association*, 114(527):1147–1161, 2019.

[170] Shihua Zhang, Rui-Sheng Wang, and Xiang-Sun Zhang. Uncovering fuzzy community structure in complex networks. *Phys. Rev. E*, 76:046103, Oct 2007.

[171] Huawei Shen, Xueqi Cheng, Kai Cai, and Mao-Bin Hu. Detect overlapping and hierarchical community structure in networks. *Physica A: Statistical Mechanics and its Applications*, 388(8):1706–1712, 2009.

[172] Prem K Gopalan, Sean Gerrish, Michael Freedman, David Blei, and David Mimno. Scalable inference of overlapping communities. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[173] Fazle E. Faisal, Lei Meng, Joseph Crawford, and Tijana Milenković. The post-genomic era of biological network alignment. *EURASIP Journal on Bioinformatics and Systems Biology*, (1):3, 2015.

[174] Pietro Hiram Guzzi and Tijana Milenković. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in Bioinformatics*, 19(3):472–481, 01 2017.

[175] D. E. Gustafson and W. C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, pages 761–766, Jan 1978.

[176] A. Geva and I. Gath. Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(07):773–780, 1989.

[177] Frank Klawonn and Frank Höppner. What is fuzzy about fuzzy clustering? understanding and improving the concept of the fuzzifier. In *Advances in Intelligent Data Analysis V*, pages 254–264, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[178] Edo M Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 33–40. Curran Associates, Inc., 2009.

[179] Edoardo M. Airoldi, David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg, editors. *Handbook of Mixed Membership Models and Their Applications*. Chapman and Hall/CRC, 2014.

[180] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[181] Walter M. Fitch. Homology a personal view on some of the problems. *Trends in Genetics*, 16:227–231, 2000.

[182] Roy A Jensen. Orthologs and paralogs - we need to get it right. *Genome Biology*, 2(8):227–231, 2001.

[183] Gang Fang, Nitin Bhardwaj, Rebecca Robilotto, and Mark B Gerstein. Getting started in gene orthology and functional analysis. *PLOS Computational Biology*, 6(3):1–8, 03 2010.

[184] Maido Remm, Christian E.V. Storm, and Erik L.L. Sonnhammer. Automatic clustering of orthologs and in-paralogs from pairwise species comparisons11edited by f. cohen. *Journal of Molecular Biology*, 314(5):1041 – 1052, 2001.

[185] Li Li, Christian J. Stoeckert, and David S. Roos. Orthomcl: Identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13(9):2178–2189, 2003.

[186] T. J. P. Hubbard, B. L. Aken, K. Beal, B. Ballester, M. Caccamo, Y. Chen, L. Clarke, G. Coates, F. Cunningham, T. Cutts, T. Down, S. C. Dyer, S. Fitzgerald, J. Fernandez-Banet, S. Graf, S. Haider, M. Hammond, J. Herrero, R. Holland, K. Howe, K. Howe, N. Johnson, A. Kahari, D. Keefe, F. Kokocinski, E. Kulesha, D. Lawson, I. Longden, C. Melsopp, K. Megy, P. Meidl, B. Ouverdin, A. Parker, A. Prlic, S. Rice, D. Rios, M. Schuster, I. Sealy, J. Severin, G. Slater, D. Smedley, G. Spudich, S. Trevanion, A. Vilella, J. Vogel, S. White, M. Wood, T. Cox, V. Curwen, R. Durbin, X. M. Fernandez-Suarez, P. Flicek, A. Kasprzyk, G. Proctor, S. Searle, J. Smith, A. Ureta-Vidal, and E. Birney. Ensembl 2007. *Nucleic Acids Research*, 35:D610–D617, 2007.

[187] Christophe Dessimoz, Gina Cannarozzi, Manuel Gil, Daniel Margadant, Alexander Roth, Adrian Schneider, and Gaston H." Gonnet. Oma, a comprehensive, automated project for the identification of orthologs from complete genome data: Introduction

and first achievements. In *Comparative Genomics*, pages 61–72, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[188] Lars Juhl Jensen, Philippe Julien, Michael Kuhn, Christian von Mering, Jean Muller, Tobias Doerks, and Peer Bork. eggnog: automated construction and annotation of orthologous groups of genes. *Nucleic Acids Research*, 36:D250–D254, 2008.

[189] NCBI Resource Coordinators. Database resources of the national center for biotechnology information. *Nucleic Acids Research*, 44(D1):D7–D19, 2016.

[190] Todd F DeLuca, I-Hsien Wu, Jian Pu, Thomas Monaghan, Leonid Peshkin, Saurav Singh, and Dennis P Wall. Roundup: a multi-genome repository of orthologs and evolutionary distances. *Bioinformatics*, 22(16):2044–2046, 2006.

[191] Toni Gabaldon and Eugene V. Koonin. Functional and evolutionary implications of gene orthology. *Nature Reviews Genetics*, pages 360–366, 2013.

[192] Romain A. Studer and Marc Robinson-Rechavi. How confident can we be that orthologs are similar, but paralogs differ. *Trends in Genetics*, 25:210–216, 2009.

[193] Adrian M. Altenhoff and Christophe Dessimoz. Phylogenetic and functional assessment of orthologs inference projects and methods. *PLoS Comput Biol*, 2009.

[194] Marina V. Omelchenko, Michael Y. Galperin, Yuri I. Wolf, and Eugene V. Koonin. Non-homologous isofunctional enzymes: A systematic analysis of alternative solutions in enzyme evolution. *Biology Direct*, 5(1):31, Apr 2010.

[195] Paul D. Thomas, Valerie Wood, Christopher J. Mungall, Suzanna E. Lewis, Judith A. Blake, and on behalf of the Gene Ontology Consortium. On the use of gene ontology annotations to assess functional similarity among orthologs and paralogs: A short report. *PLOS Computational Biology*, 8(2):1–7, 02 2012.

[196] Adrian M. Altenhoff, Romain A. Studer, Marc Robinson-Rechavi, and Christophe Dessimoz. Resolving the ortholog conjecture: Orthologs tend to be weakly, but

significantly, more similar in function than paralogs. *PLOS Computational Biology*, 8(5):1–10, 05 2012.

[197] A R Mushegian and E V Koonin. A minimal gene set for cellular life derived by comparison of complete bacterial genomes. *Proceedings of the National Academy of Sciences*, 93(19):10268–10273, 1996.

[198] Eugene V. Koonin. Comparative genomics, minimal gene-sets and the last universal common ancestor. *Nature Reviews Microbiology*, 1, 2003.

[199] Michael Y. Galperin, D. Roland Walker, and Eugene V. Koonin. Analogous enzymes: Independent inventions in enzyme evolution. *Genome Research*, 8(8):779–790, 1998.

[200] Toni Gabaldón, Christophe Dessimoz, Julie Huxley-Jones, Albert J. Vilella, Erik LL Sonnhammer, and Suzanna Lewis. Joining forces in the quest for orthologs. *Genome Biology*, 10(9):403, Sep 2009.

[201] J. A. Hartingan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Society*, 67(337):123–129, 1972.

[202] Yizong Cheng and George M Church. Biclustering of expression data. *Proceedings of the International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, 8:93–103, 2000.

[203] Sara C Madeira, Miguel C Teixeira, Isabel Sá-Correia, and Arlindo L Oliveira. Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 7(1):153–165, 2010.

[204] Gilles Bisson and Fawad Hussain. Chi-Sim: A new similarity measure for the co-clustering task. *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008*, pages 211–217, 2008.

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

[206] Therese Sørlie, Charles M. Perou, Robert Tibshirani, Turid Aas, Stephanie Geisler, Hilde Johnsen, Trevor Hastie, Michael B. Eisen, Matt van de Rijn, Stefanie S. Jeffrey, Thor Thorsen, Hanne Quist, John C. Matese, Patrick O. Brown, David Botstein, Per Eystein Lønning, and Anne-Lise Børresen-Dale. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, 98(19):10869–10874, 2001.

[207] Therese Sørlie, Robert Tibshirani, Joel Parker, Trevor Hastie, J. S. Marron, Andrew Nobel, Shibing Deng, Hilde Johnsen, Robert Pesich, Stephanie Geisler, Janos Demeter, Charles M. Perou, Per E. Lønning, Patrick O. Brown, Anne-Lise Børresen-Dale, and David Botstein. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the National Academy of Sciences*, 100(14):8418–8423, 2003.

[208] Kai Wang, Junsuo Kan, Siu Tsan Yuen, Stephanie T Shi, Kent Man Chu, Simon Law, Tsun Leung Chan, Zhengyan Kan, Annie S Y Chan, Wai Yin Tsui, Siu Po Lee, Siu Lun Ho, Anthony K W Chan, Grace H W Cheng, Peter C Roberts, Paul A Rejto, Neil W Gibson, David J Pocalyko, Mao Mao, Jiangchun Xu, and Suet Yi Leung. Exome sequencing identifies frequent mutation of arid1a in molecular subtypes of gastric cancer. *Nature Genetics*, 43(7515):1219–1223, 2011.

[209] Shantanu Banerji et al. Sequence analysis of mutations and translocations across breast cancer subtypes. *Nature*, 486:405–409, 2012.

[210] Krzakala Florent, Moore Cristopher, Mossel Elchanan, Neeman Joe, Allan Sly, Zdeborová Lenka, and Zhang Pan. Spectral redemption: clustering sparse networks. *PNAS*, 110(52):20935–20940, 2013.

[211] M. E. J. Newman. Spectral community detection in sparse networks, 2013.

[212] Abhinav Singh and Mark D. Humphries. Finding communities in sparse networks. *Scientific Reports*, 2015.

[213] Susan E. Celniker, Laura A. L. Dillon, Mark B. Gerstein, Kristin C. Gunsalus, Steven Henikoff, Gary H. Karpen, Manolis Kellis, Eric C. Lai, Jason D. Lieb, David M. MacAlpine, Gos Micklem, Fabio Piano, Michael Snyder, Lincoln Stein, Kevin P. White, Robert H. Waterston, and modENCODE Consortium. Unlocking the secrets of the genome. *Nature*, 459, 2009.

[214] Xianghong Zhou, Ming-Chih J. Kao, and Wing Hung Wong. Transitive functional annotation by shortest-path analysis of gene expression data. *Proceedings of the National Academy of Sciences*, 99(20):12783–12788, 2002.

[215] Jeremiah J Faith, Boris Hayete, Joshua T Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J Collins, and Timothy S Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLOS Biology*, 5(1):1–13, 01 2007.

[216] Wei Vivian Li, Yiling Chen, and Jingyi Jessica Li. Trom: A testing-based method for finding transcriptomic similarity of biological samples. *Statistics in Biosciences*, 2016.

[217] opossum is a web-based system for the detection of over-represented conserved transcription factor binding sites and binding site combinations in sets of genes or sequences. `http://opossum.cisreg.ca/oPOSSUM3/`.

[218] Renqiang Min Chao Cheng and Mark Gerstein. Tip: A probabilistic method for identifying transcription factor target genes from chip-seq binding profiles. *Bioinformatics*, 27:3221–3227, 2011.

[219] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22(8):888–905, 2000.

[220] Yair Weiss. Segmentation using eigenvectors: a unifying view. *IEEE*, 1999.

[221] Fan R.K.Chung. *Spectral Graph Theory*, volume 92. CBMS Regional Conference Series in Mathematics, 1997.

[222] Jianxi Li, Ji-Ming Guo, and Wai Chee Shiu. Bounds on normalized laplacian eigenvalues of graphs. *Journal of Inequalities and Applications*, 2014(1):1, 2014.

[223] Carl Meyer, Shaina Race, and Kevin Valakuzhy. Determining the number of clusters via iterative consensus clustering. In *SDM*, 2013.

[224] Robert Tibshirani and Guenther Walther. Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics*, 14(3):511–528, 2005.

[225] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2000.

[226] Yang Yang, Yu-Cheng T. Yang, Jiapei Yuan, Zhi John Lu, and Jingyi Jessica Li. Large-scale mapping of mammalian transcriptomes identifies conserved genes associated with different cell states. *Nucleic Acids Research*, 45(4):1657–1672, 2017.

[227] Serafim Batzoglou, Lior Pachter, Jill P. Mesirov, Bonnie Berger, and Eric S. Lander. Human and mouse gene structure: Comparative analysis and application to exon prediction. *Genome Research*, 10(7):950–958, 2000.

[228] Abel Ureta-Vidal, Laurence Ettwiller, and Ewan Birney. Comparative genomics: genome-wide analysis in metazoan eukaryotes. *Nature Reviews Genetics*, 4:251–262, 2003.

[229] Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101, 2004.

[230] Piyush B Madhamshettiwar, Stefan R Maetschke, Melissa J Davis, Antonio Reverter, and Mark A Ragan. Gene regulatory network inference: evaluation and application

to ovarian cancer allows the prioritization of drug targets. *Genome medicine*, 4(5):41, 2012.

[231] Neelroop N Parikshak, Michael J Gandal, and Daniel H Geschwind. Systems biology and gene networks in neurodevelopmental and neurodegenerative disorders. *Nature Reviews Genetics*, 16(8):441, 2015.

[232] Paul H Harvey and Mark D Pagel. *The comparative method in evolutionary biology*, volume 239. Oxford university press Oxford, 1991.

[233] Ben-Yang Liao and Jianzhi Zhang. Evolutionary conservation of expression profiles between human and mouse orthologous genes. *Molecular biology and evolution*, 23(3):530–540, 2005.

[234] Ingrid Hedenfalk, David Duggan, Yidong Chen, Michael Radmacher, Michael Bittner, Richard Simon, Paul Meltzer, Barry Gusterson, Manel Esteller, Mark Raffeld, et al. Gene-expression profiles in hereditary breast cancer. *New England Journal of Medicine*, 344(8):539–548, 2001.

[235] Reika Iwakawa, Takashi Kohno, Yasushi Totoki, Tatsuhiro Shibata, Katsuya Tsuchihara, Sachiyo Mimaki, Koji Tsuta, Yoshitaka Narita, Ryo Nishikawa, Masayuki Noguchi, et al. Expression and clinical significance of genes frequently mutated in small cell lung cancers defined by whole exome/rna sequencing. *Carcinogenesis*, 36(6):616–621, 2015.

[236] Dan Robinson, Eliezer M Van Allen, Yi-Mi Wu, Nikolaus Schultz, Robert J Lonigro, Juan-Miguel Mosquera, Bruce Montgomery, Mary-Ellen Taplin, Colin C Pritchard, Gerhardt Attard, et al. Integrative clinical genomics of advanced prostate cancer. *Cell*, 161(5):1215–1228, 2015.

[237] Bert Vogelstein and Kenneth W Kinzler. Cancer genes and the pathways they control. *Nature medicine*, 10(8):789, 2004.

[238] Maayan Baron, Adrian Veres, Samuel L. Wolock, Aubrey L. Faust, Renaud Gaujoux, Amedeo Vetere, Jennifer Hyoje Ryu, Bridget K. Wagner, Shai S. Shen-Orr, Allon M.

Klein, Douglas A. Melton, and Itai Yanai. A single-cell transcriptomic map of the human and mouse pancreas reveals inter- and intra-cell population structure. *Cell Systems*, 3(4):346–360.e4, 2016.

[239] Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature Biotechnology*, 36:411–420, 2018.

[240] Jianming Zhang, Antony M. Dean, Frédéric Brunet, and Manyuan Long. Evolving protein functional diversity in new genes of drosophila. *Proceedings of the National Academy of Sciences*, 101(46):16246–16250, 2004.

[241] Sidi Chen, Benjamin H. Krinsky, and Manyuan Long. New genes as drivers of phenotypic evolution. *Nature Reviews Genetics*, 14, 2013.

[242] C. Berge. *Graphs and Hypergraphs*. Elsevier Science Ltd., Oxford, UK, UK, 1985.

[243] Y. Gao, M. Wang, Z. Zha, J. Shen, X. Li, and X. Wu. Visual-textual joint relevance learning for tag-based social image search. *IEEE Transactions on Image Processing*, 22(1):363–376, Jan 2013.

[244] Carl Kingsford and Rob Patro. Predicting protein interactions via parsimonious network history inference. *Bioinformatics*, 29(13):i237–i246, 06 2013.

[245] Stephen P. Ficklin and F. Alex Feltus. Gene coexpression network alignment and conservation of gene modules between two grass species: Maize and rice. *Plant Physiology*, 156(3):1244–1256, 2011.

[246] A. Ray, J. Ghaderi, S. Sanghavi, and S. Shakkottai. Overlap graph clustering via successive removal. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 278–285, 2014.

[247] S. B. Hopkins and D. Steurer. Efficient bayesian estimation from few samples:

Community detection and related problems. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 379–390, 2017.

[248] Austin R. Benson, Paul Liu, and Hao Yin. A simple bipartite graph projection model for clustering in networks, 2020.

[249] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[250] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 89–98, New York, NY, USA, 2003. Association for Computing Machinery.

[251] Marina Meilă. Comparing clusterings—an information based distance. *J. Multivar. Anal.*, 98(5):873–895, May 2007.

[252] Aaron F. McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms, 2011.

[253] Simone Romano, James Bailey, Vinh Nguyen, and Karin Verspoor. Standardized mutual information for clustering comparisons: One step further in adjustment for chance. volume 32 of *Proceedings of Machine Learning Research*, pages 1143–1151, Bejing, China, 22–24 Jun 2014. PMLR.

[254] Menachem Dishon and George H. Weiss. Small sample comparison of estimation methods for the beta distribution. *Journal of Statistical Computation and Simulation*, 11(1):1–11, 1980.

[255] Nicolas Wicker, Jean Muller, Ravi Kiran Reddy Kalathur, and Olivier Poch. A maximum likelihood approximation method for dirichlet's parameter estimation. *Computational Statistics and Data Analysis*, 52(3):1315 – 1322, 2008.

[256] Hari S. Hariharan and Raja P. Velu. On estimating dirichlet parameters - a comparison of initial values. *Journal of Statistical Computation and Simulation*, 48(1-2):47–58, 1993.

[257] Kai Wang Ng, Guo-Liang Tian, and Man-Lai Tang. Dirichlet and related distributions: Theory, methods and applications. 888, 01 2011.

[258] Jonathan Huang. Maximum likelihood estimation of dirichlet distribution parameters. *CMU Technique Report*, 2005.

[259] Thomas Minka. Github repository: Fastfit, 2017.

[260] Tom Minka. Beyond newton's method. April 2000.

[261] A. Narayanan. Algorithm as 266: Maximum likelihood estimation of the parameters of the dirichlet distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 40(2):365–374, 1991.

[262] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[263] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22, 2011.

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

[265] M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences*, 104(23):9564–9569, 2007.

[266] Maria A. Riolo, George T. Cantwell, Gesine Reinert, and M. E. J. Newman. Efficient method for estimating the number of communities in a network. *Phys. Rev. E*, 96:032310, Sep 2017.

[267] M. E. J. Newman and Gesine Reinert. Estimating the number of communities in a network. *Phys. Rev. Lett.*, 117:078301, Aug 2016.

[268] Junxian Geng, Anirban Bhattacharya, and Debdeep Pati. Probabilistic community detection with unknown number of communities. *Journal of the American Statistical Association*, 114(526):893–905, 2019.

[269] Cyrille Joutard, Edoardo Airoldi, Stephen Fienberg, and Tanzy Love. *Discovery of Latent Patterns with Hierarchical Bayesian Mixed-Membership Models and the Issue of Model Choice*, pages 240–275. 08 2007.

[270] Tao Zhou, Jie Ren, Matú š Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Phys. Rev. E*, 76:046115, Oct 2007.

[271] Tiago P. Peixoto. Model selection and hypothesis testing for large-scale network models with overlapping groups. *Phys. Rev. X*, 5:011033, Mar 2015.

[272] Yuchung J. Wang and George Y. Wong. Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82(397):8–19, 1987.

[273] Radford M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical report, 1993.

[274] Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[275] Wray Buntine and Aleks Jakulin. Discrete component analysis. In Craig Saunders, Marko Grobelnik, Steve Gunn, and John Shawe-Taylor, editors, *Subspace, Latent Structure and Feature Selection*, pages 1–33, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[276] Michael Braun and Jon McAuliffe. Variational inference for large-scale models of discrete choice. *Journal of the American Statistical Association*, 105(489):324–335, 2010.

[277] Kurt Miller, Michael I. Jordan, and Thomas L. Griffiths. Nonparametric latent feature models for link prediction. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1276–1284. Curran Associates, Inc., 2009.

[278] Mark J. Schervish. *Theory of Statistics*. Springer-Verlag New York, 1995.

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

[280] Morten Mørup and Lars Kai Hansen. Learning latent structure in complex networks. 2009.

[281] S. Pal and M. Coates. Scalable mcmc in degree corrected stochastic block model. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5461–5465, May 2019.

[282] Xiaoyan Lu and Boleslaw K. Szymanski. A regularized stochastic block model for the robust community detection in complex networks. *Scientific Reports*, 9(1):13247, 2019.

[283] Wenzhe Li, Sungjin Ahn, and Max Welling. Scalable mcmc for mixed membership stochastic blockmodels. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 723–731, Cadiz, Spain, 09–11 May 2016. PMLR.

[284] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables,*. Dover Publications, Inc., USA, 1974.

[285] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.