Title
Graphon Estimation by Empirical Bayes Approach and Causal Discovery from Multiple Populations

Author
Peng, Zhanhao

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Graphon Estimation by Empirical Bayes Approach and Causal Discovery from Multiple Populations

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

by

Zhanhao Peng

2021

ABSTRACT OF THE DISSERTATION

Graphon Estimation by Empirical Bayes Approach
and Causal Discovery from Multiple Populations

by

Zhanhao Peng
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2021
Professor Qing Zhou, Chair

Graph is a natural representation of network data. Over the decades many researches have been conducted on graph theory, graphical models and statistical network analysis. Two main kinds of graphs: undirected graphs and directed graphs, each have their own developments and help solve different kinds of problems. Our works made contributions to these two regimes: nodes clustering and estimation in undirected graphs, and causal structure estimation using directed graphs.

In the first part of the dissertation, we focus on one type of undirected graphical model: the graphon (W-graph), including the stochastic blockmodel as a special case. It has been widely used in modeling and analyzing network data. This random graph model is well-characterized by its graphon function, and estimation of the graphon function has gained a lot of recent research interests. Most existing works focus on detecting the latent space of the model, while adopting simple maximum likelihood or Bayesian estimates for the graphon or connectivity parameters given the identified latent variables. In this project, we propose a hierarchical model and develop a novel empirical Bayes estimate of the connectivity matrix

of a stochastic blockmodel to approximate the graphon function. Based on the likelihood of our hierarchical model, we further introduce a new model selection criterion for choosing the number of communities. Numerical results on extensive simulations and two well-annotated social networks demonstrate the superiority of our approach in terms of estimation accuracy and model selection.

In the second part of the dissertation, we focus on one of the most popular directed graphical models: Bayesian networks. The intuition of our work came from the liquid association theory, which claims that gene regulatory strength differs by the cellular states. We encode this phenomenon into a statistical model and propose an algorithm to discover causal relations from observational data generated from different populations. We analyze the relationship of edges with different weights and coefficients of node wise regression in two populations. And we use this observed relationship to orient undirected edges in completed partially directed acyclic graphs (cpDAGs). Numerical results on simulations and a real data example show the effectiveness of our algorithm and its improvement on existing structural learning methods.

The dissertation of Zhanhao Peng is approved.

Xianghong Jasmine Zhou

Mark Stephen Handcock

Arash Ali Amini

Qing Zhou, Committee Chair

University of California, Los Angeles

2021

To my mother and father.

To my twenty years of campus life.

To a fruitful end and a fresh start.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, professor Qing Zhou, for his support and guidance throughout my dissertation study. His kindness, patience and immense knowledge helped me overcome many challenges.

I would also like to show my appreciation to the committee members: professor Arash Amini, professor Mark Handcock and professor Jasmine Zhou, for their comments and suggestions.

VITA

2012–2016 Bachelor of Engineering, Department of Automation, Tsinghua University.

2016–2021 Teaching Fellow and Graduate Student Researcher, Department of Statistics, University of California, Los Angeles.

CHAPTER 1

Introduction

Graphical model (*graph*) is a perfect combination of probability theory and graph theory. It is a natural tool to deal with the uncertainty and complexity problems in statistics and networks. Each individual corresponds to a vertex or node in the graph, while their relations are modeled by edges between the vertices. In statistical model, nodes represent random variables, the existence and absence of edges between nodes represent conditional independence assumptions, and together they are a compact representation of joint probability distributions. Meanwhile, graph is a natural representation of network data, consisting of relations among a set of individuals, thus it constructs a bridge between the study of variable relations and the analysis of networks.

There are two main kinds of graphs: undirected graphs and directed graphs, and there are different types of studies dealing with the problems in these two categories. Undirected graphs are popular with studies on nodes communities, they also have widely applications in physics, biology, sociology and communication (Albert and Barabási, 2002). Network data modeled by undirected graphs are often analyzed through statistical methods so that the underlying properties of the network structure can be better understood via estimation of model parameters. Examples of such properties include edge weights, degrees, clusters and diameter (Barabási and Albert, 1999; Newman et al., 2002) among others. On the other hand, directed graphs allow for causal interpretations, and studies have been focused on graph structure estimation, causal inference on variables and other fields in machine learning. Finding causal structures of directed graphs has become popular in many applied

fields as well, including genomics, epidemiology and social sciences (Bang-Jensen and Gutin, 2002; Sachs et al., 2005; Gao and Cui, 2015; Greenland et al., 1999).

The main difference of the two kinds of graphs is the orderliness of pairs of nodes in the graph. Define a graph $G = (\mathbb{V}, \mathbb{E})$ consisting of a set of vertices $\mathbb{V} = \{V_1, \cdots, V_n\}$ and a set of edges $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$. When the elements of $\mathbb{E}$ are unordered pairs, i.e. for any nodes $V_i, V_j \in \mathbb{V}$, pair (V_i, V_j) and pair (V_j, V_i) have identical representation, then the graph is a undirected graph. On the other hand, if the the elements of $\mathbb{E}$ are ordered pairs, the graph is a directed graph. It is possible that a directed graph contains undirected edges (explained in Section 1.3), which is called a partially directed graph.

In this dissertation we focus on some of the most widely applied undirected and directed graphical models. Graphon (Section 1.1) has been very popular in the studies of undirected graphical models, and Bayesian network (Section 1.3) is one of the most important directed graphical model which have undoubtedly received the most attention in this field. Even though they are all graphs represented by vertices and edges, statistical representations and methods of analysis can be very different.

1.1 SBM and graphon

To better understand the heterogeneity among vertices in a network, community detection and graph clustering methods (Girvan and Newman, 2002; Newman, 2004) have been proposed to group vertices into clusters that share similar connection profiles. A large portion of the clustering methods are developed based on the stochastic blockmodel (SBM) (Freeman, 1983), which constructs an interpretable probabilistic model for the heterogeneity among nodes and edges in an observed network.

For a simple random graph on n nodes or vertices, the relationships between the nodes are modeled by $\frac{1}{2}n(n-1)$ binary random variables representing the presence or absence of an edge. The edge variables can be equivalently represented by an $n \times n$ adjacency matrix

$\mathbf{X}$, where $X_{ij} = 1$ if node i and j are connected and $X_{ij} = 0$ otherwise. We do not consider self loops in this work, and thus $X_{ii} = 0$ for $i = 1, \ldots, n$.

Many popular graph models (Lloyd et al., 2012) make exchangeability assumption on the vertices: the distribution of degrees and densities in the random graph is invariant to permutation or relabeling of the vertices. A large class of exchangeable graphs can be defined by the so-called *graphon* function (Lovasz and Szegedy, 2006). A graphon $W(u, v)$ is a symmetric function: $[0, 1]^2 \to [0, 1]$. To generate an n-vertex random graph given a graphon $W(u, v)$, we first draw latent variables u_i from the uniform distribution $\mathcal{U}(0, 1)$ for $i = 1, \ldots, n$ independently. Then we connect each pair of vertices (i, j) with probability $W(u_i, u_j)$, i.e.

$$\mathbb{P}(X_{ij} = 1 | u_i, u_j) = W(u_i, u_j), \quad i, j = 1, \ldots, n. \tag{1.1}$$

In particular, the stochastic blockmodel mentioned above can be seen as a special case of the graphon model, where $W(u, v)$ is a piecewise constant function. Abbe (2018); Lee and Wilkinson (2019); Funke and Becker (2019) have summarized recent developments on the model, including different types of SBM extension, the graph clustering methods, and the model selection criteria. Under an SBM, the vertices are randomly labeled with independent latent variables $\mathbf{Z} = (z_1, \ldots, z_n)$, where $z_i \in \{1, \ldots, K\}$ for $i = 1, \ldots, n$ and K is the number of communities or clusters among all the nodes. The distribution of $(\mathbf{Z}, \mathbf{X})$ is specified as follows:

$$\begin{aligned} &\mathbb{P}(z_i = m) = \pi_m, \quad m \in \{1, \ldots, K\}, \quad i = 1, \ldots, n, \\ &\mathbb{P}(X_{ij} = 1 | z_i, z_j) = \theta_{z_i z_j}, \quad i, j = 1, \ldots, n, \end{aligned} \tag{1.2}$$

where $\sum_m \pi_m = 1$ and each $\theta_{km} \in [0, 1]$. Put $\pi = (\pi_1, \ldots, \pi_m)$ and $\Theta = (\theta_{ij})_{K \times K}$.

The original SBM (1.2) does not require that nodes in the same block should have higher connectivity within the block, thus the values of the diagonal of Θ are not necessarily higher than the rest. However, many community detection algorithms have a goal to assign nodes into the same cluster when they have higher probability to connect with each other. Thus

assortativity is often taken into account in the models. In assortative SBM (Gopalan et al., 2012; Li et al., 2015), a constraint is added to control the parameters so that $\theta_{ij} < \theta_{ii}$ for $i \neq j$. Lu and Szymanski (2019) proposed a regularized SBM to adjust the level of assortativeness: By tuning the parameters, a desired level of assortativeness can be achieved. It is worth mentioning that incorporating assortativeness is not a universal solution, and bipartite network is a perfect example for networks with disassortativity, where connectivity only exists between groups rather than within groups. Before applying assortative SBM to real world data, cautions need to be taken, and background knowledge is essential when making decisions on which model should be used.

Over years of research on SBM, the original model in (1.2) has been extended to achieve a better fit to real world data. For real data one major concern is that some nodes can reasonably belong to multiple groups, since even sharing the same membership labels, their interactions with different nodes may vary. One extension to incorporate soft clustering on the nodes is the mixed membership SBM (MMSBM) (Airoldi et al., 2008), in which the latent variables are no longer labels that indicate the memberships, but vectors $v \in \mathbb{R}^K$ representing the probability of belonging to each group. Consequently, in different pairs of nodes each node can belong to different groups. For a pair (i, j), the latent variable $z_{ij} \in \mathbb{R}^K$ contains exactly one 1 (0 for other elements), and is drawn from a multinomial distribution with probabilities v_i. Airoldi et al. (2008) also justified the effectiveness of variational inference method on MMSBM, which inspired some variational EM approaches to SBM and graphon estimation and facilitates the computation of model selection criteria (see details in Section 1.2).

Another limitation of SBM is that any two nodes in the same block have the same degree distribution, which makes it unlikely for the model to group nodes with very different degrees into the same block. This becomes a problem when it fits the high skewness of the real world networks. To address this problem, Karrer and Newman (2011) proposed degree-corrected SBM (DCSBM), where the binary variable X_{ij} from (1.2) is redefined as the number of edges

between the pair of nodes (i, j), and naturally follows a Poisson distribution

$$X_{ij}|\Theta \sim \text{Poisson}(\alpha_i \alpha_j \theta_{z_i z_j}), \tag{1.3}$$

where each node gets an additional parameter α. When $\alpha_i = 1$, $i = 1, \ldots, n$, this model is essentially the same as the regular SBM. Typically the parameter is forced to sum to the total degree within each block, i.e. $\sum_{i:z_i=u} \alpha_i = \sum_{i:z_i=u} d_i$, where d_i is the degree of node i. Some recent methods were developed upon the DCSBM, such as brief propagation algorithm for model selection (Yan et al., 2018), a unifying framework for modeling graphs with parameters following exponential distributions (Aicher et al., 2014).

A model that is highly related to the stochastic blockmodel is the latent space model (Hoff et al., 2002), where the latent variables $\mathbf{Z}$ are similarly associated with the nodes. The latent variable does not represent the group membership but works as a coordinate that determines the connection probability together with other nodes' coordinates. The way of calculating the probability

$$\mathbb{P}(X_{ij} = 1) = \frac{e^{-d(z_i, z_j)}}{1 + e^{-d(z_i, z_j)}}, \tag{1.4}$$

is relatively more complicated than a block matrix Θ. In (1.4), $d(\cdot, \cdot)$ is a distance measure such as Euclidean distance in the space defined. Handcock et al. (2007) has further proposed a latent space cluster model by considering the prior distribution of the latent positions, where $(z_1, \cdots, z_n)$ are assumed to be drawn from a mixture of K Gaussian distributions independently, and each Gaussian distribution has a different mean and covariance matrix that represents a different cluster, thus nodes membership and clustering are accounted for explicitly. Developments on latent space models usually have heavy connections with SBM, and these two types of models are not completely separate in the field of statistical network analysis.

Graphons, with higher complexity compared to SBM, are seen as kernel functions for random graph models (Lawrence, 2005). They seem to be a good compromise between

SBM and the latent space model. Many works on graphon studied the exchangeability of the model, where the ordering of random variables of the nodes have no information. We re-write the graphon function $W(u, v)$ in (1.1) as

$$W(u, v) = \rho_n f(u, v), \tag{1.5}$$

where $\rho_n > 0$ and $f(u, v)$ is normalized as a density function where $\iint_{(0,1)^2} f(u, v)dudv = 1$. Without loss of generality, we assume function f is in the Hölder class (Gao et al., 2015; Klopp et al., 2017). And the scaling constant ρ_n can be estimated by

$$\hat{\rho}_n = \binom{n}{2}^{-1} \sum_{i<j} X_{ij} \tag{1.6}$$

for each network size n. Our simple stochastic network is then

$$X_{ij}|u_i, u_j \sim \text{Bernoulli}(\rho_n f(u_i, u_j)),\ 1 \leq i < j \leq n, \tag{1.7}$$

with u_i, u_j as the latent variables. From the mechanism, we can see that any rearrangement of the x and y axes of the function $f(x, y)$ will result in the same probability distribution on the unlabeled graphs.

In statistical network studies, graphs are theoretically divided into dense graphs and sparse graphs based on their edge density. A dense graph with n nodes has the number of edges $|\mathbb{E}|$ close to the maximal number of edges $\binom{n}{2}$. The oppsite, a graph with only a few edges, is a sparse graph. In mathematics, dense and sparse graphs are defined as follows: Let $(G_n)_n = G_1, G_2, \ldots$ be a sequence of graphs, where each graph $G_n = (\mathbb{V}_n, \mathbb{E}_n)$ consists of a (finite) set of vertices and a (finite) multiset of edge $\mathbb{E}_n$. Assume that the sequence is growing, such that $\mathbb{V}_n \subseteq \mathbb{V}_{n+1}$ and $\mathbb{E}_n \subseteq \mathbb{E}_{n+1}$. We say that the sequence is dense if $|\mathbb{E}_n| = \Theta(|\mathbb{V}_n|^2)$ and sparse if $|\mathbb{E}_n| = o(|\mathbb{V}_n|^2)$, where $|\mathbb{E}_n|$ is the number of edges in G_n.

From the Aldous-Hoover theorem (Bickel and Chen, 2009), a graph that is represented by an exchangeable random array, such as SBM and graphon (1.1), is either dense or empty. Namely, the number of edges grows quaratically with the number of nodes (Lovasz and

Szegedy, 2006; Orbanz and Roy, 2015). Graphon was originally studied as the limit object of dense graph sequences (Lovasz and Szegedy, 2006; Borgs et al., 2016). However, some empirical studies indicate that many real networks are sparse (Newman et al., 2002), thus it would be meaningful to extend the exchangeable dense graphs into the sparse regime.

The generation of sparse graphs that obtain the property of random exchangeable graphs are introduced by the graphex model (Veitch and Roy, 2015). The model changes the support of graphon function from $(0,1)^2$ into $\mathbb{R}_+^2$ and naturally extends the model while preserving its flexibility and tractability. Following the notations above, the graph is parameterized by a symmetric measurable function similar to graphon $W : \mathbb{R}_+^2 \to (0,1)$. For each pair of nodes (i,j),

$$X_{ij}|(\theta_k, \vartheta_k)_{k=1,2,\dots} \sim \text{Bernoulli}(W(\vartheta_i, \vartheta_j)) \tag{1.8}$$

where $(\theta_k, \vartheta_k)_{k=1,2,\dots}$ is a unit-rate Poisson process on $\mathbb{R}_+^2$. To realize the process in simulation, the range of node label $(\theta_k)_{k=1,2,\dots}$ is truncated to $(0,\alpha]$. In fact, these labels have no impact on generation of the graph and are usually not used in statistical analysis. This model is a generalization of the graphon for dense exchangeable graphs. The edge density of the generated graph is determined by the function W, which can be referred to as the graphon function too. The way to adjust the sparsity of the generated graph is to change the graphon function W. Define

$$\mu(x) = \int_0^\infty W(x,y)dy, \tag{1.9}$$

$$\nu(x) = \int_0^\infty W(x,z)W(y,z)dz, \tag{1.10}$$

and assume μ is non-increasing, with generalized inverse $\mu^{-1}(x) = \inf\{y > 0 | \mu(y) \leq x\}$, such that as $x \to 0$,

$$\mu^{-1}(x) \sim l(1/x)x^{-\sigma}, \tag{1.11}$$

where $\sigma \in [0,1]$ and l is a slowly varying function at infinity, i.e.

$$\lim_{t\to\infty} \frac{l(ct)}{l(t)} = 1 \tag{1.12}$$

for all $c > 0$. The value of σ and the limits of l determine the property of the graph, which can be differentiated into dense and sparse cases:

- Dense: $\sigma = 0$ and $\lim_{t\to\infty} l(t) = 0$. In this case $\lim_{x\to 0} \mu^{-1}(x) < \infty$, thus μ and W have compact support.
- Sparse: $\sigma = 0$ and $\lim_{t\to\infty} l(t) = \infty$. In this case $\lim_{x\to 0} \mu^{-1}(x) = \infty$, thus μ and W have full support; $\sigma \in (0, 1)$, μ and W have full support and μ has polynomially decaying tails; $\sigma = 1$, μ has a very light tail, to make μ^{-1} and W integrable, l has to go to zero sufficiently fast, this case will generate a very sparse graph.

Caron and Rousseau (2017) provided some examples of graphon function to generate graphs with different density, such as the dense case: $W(x, y) = (1 - x)(1 - y)\mathbb{I}_{x\leq 1}\mathbb{I}_{y\leq 1}$, the sparse case following the power law: $W(x, y) = (x + 1)^{-1/\sigma}(y + 1)^{-1/\sigma}$, and the extremely sparse case: $W(x, y) = \frac{1}{(x+1)(1+\log(1+x))^2}\frac{1}{(y+1)(1+\log(1+y))^2}$. In order to control the density and network structure of generated data, such as incorporating the block structures in SBM, Caron and Rousseau (2017) proposed an idea to factorize W. Considering the model (1.8), change the range of ϑ_i from $\mathbb{R}_+$ to F, where F is a probability space. Then function $W : (\mathbb{R}_+ \times F)^2 \to (0, 1)$, and $(\theta_k, \vartheta_k)_{k=1,2,\dots}$ are the points generated from a Poisson process with mean measure $d\theta u d\vartheta$ on $\mathbb{R}_+ \times (\mathbb{R}_+ \times F)$. Denote $\vartheta = (\sigma, v) \in \mathbb{R}_+ \times F$, let $\xi d\vartheta = d\sigma G dv$ where G is a probability distribution on F. Re-write W as

$$W((\sigma_i, v_i), (\sigma_j, v_j)) = w(v_i, \sigma_i)\eta(\sigma_i, \sigma_j), \tag{1.13}$$

where w determines the local structure of the graph, which is similar to the graphon function in dense graph model (1.1), and η tunes the sparsity behaviour of the graph. It has been shown that this model has the same property as (1.8) and also has local structure similar to stochastic blockmodel.

The sparse graphon model (1.8) and the method in (1.13) are considered to be good choices for sparse exchangeable networks generation. Unfortunately from our experiments,

the existing variational inference methods and spectral clustering methods did not work well on sparse networks generated by this process. As a consequence, we were still using the common setting to generate data from sparse graphs: using very small connection probability so that the number of edges in the network is small.

1.2 Inference on SBM and graphon

Many efforts have been made on statistical inference of the SBM to detect block structures as well as to estimate the connectivity probabilities in the blocks. Some classical and popular methods include Markov chain Monte Carlo (MCMC), degree-based algorithms and variational inference among others.

MCMC is straightforward and simple as an algorithm, though its high computational cost brings challenges. Nowicki and Snijders (2001) developed a Gibbs sampler to estimate parameters for graphs of small sizes (up to a few hundred nodes), where the parameter Θ can be updated in individual Gibbs steps. Some MCMC methods put parameter K, i.e. the number of clusters, in the sampling process, integrating the graph clustering and model selection into one algorithm (Palla et al., 2012; Fan et al., 2015; Tang et al., 2019; Tang and Yang, 2014). These methods aimed at higher estimation accuracy and simple implementation. Meanwhile, some efforts were made to reduce the computational complexity to make MCMC feasible in real world network applications (Mørup et al., 2011; Li et al., 2016).

An alternate is the class of variational expectation maximization (VEM) algorithms. The variational EM algorithm (Daudin et al., 2008) and variational Bayes EM (Latouche et al., 2012) approximate the conditional distribution of group labels given the network data by a class of distributions with simpler forms. For any distribution $q(\cdot)$ of the latent variables $\mathbf{Z}$, we can write a closed form approximate posterior distribution of the parameters (π, Θ) and of the latent variables $\mathbf{Z}$, where the observed-data log-likelihood can be decomposed into two

terms,

$$\ln p(\mathbf{X}) = \mathcal{L}(q(\cdot)) + \mathrm{KL}(q(\cdot)\|p(\cdot|\mathbf{X})), \tag{1.14}$$

where

$$\mathcal{L}(q(\cdot)) = \sum_{\mathbf{Z}} \iint q(\mathbf{Z}, \pi, \Theta) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}, \pi, \Theta)}{q(\mathbf{Z}, \pi, \Theta)} \right\} d\pi d\Theta, \tag{1.15}$$

and

$$\mathrm{KL}(q(\cdot)\|p(\cdot|\mathbf{X})) = -\sum_{\mathbf{Z}} \iint q(\mathbf{Z}, \pi, \Theta) \ln \left\{ \frac{p(\mathbf{Z}, \pi, \Theta|\mathbf{X})}{q(\mathbf{Z}, \pi, \Theta)} \right\} d\pi d\Theta. \tag{1.16}$$

Minimizing (1.16) with respect to $q(\mathbf{Z}, \pi, \Theta)$ is equivalent to maximizing the lower bound (1.15) with respect to $q(\mathbf{Z}, \pi, \Theta)$. However, when considering SBM, $q(\mathbf{Z}, \pi, \Theta)$ is intractable, thus we can assume that it can be factorized as

$$q(\mathbf{Z}, \pi, \Theta) = q(\mathbf{Z})q(\pi)q(\Theta) = q(\pi)q(\Theta)\prod_{i=1}^{N} q(z_i), \tag{1.17}$$

where the optimal approximation $q(z_i)$ at vertex i follows a multinomial distribution. Latouche et al. (2012) used a variational Bayes EM (VBEM) algorithm described in Beal and Ghahramani (2003) to optimize over $q(z_i)$ and $q(\pi)$, $q(\Theta)$ iteratively. Airoldi et al. (2008) has shown that when being applied to MMSBM, variational EM outperforms MCMC, with time complexity $O(nK + 2K)$ versus $O(n^2)$.

Other inference methods include a degree-based algorithm proposed by Channarond et al. (2012), which achieves classification, estimation and model selection from empirical degree data. Suwan et al. (2016) recast the SBM to a random dot product graph (Young and Scheinerman, 2007) and developed a Bayesian inference method with a prior specified empirically by adjacency spectral embedding.

The inference on SBM has also gained attention in community detection field, which involves the development of clustering methods in different types of networks. As a widely studied community detection algorithm, in Section 2.2 we also use spectral clustering as

an alternative clustering algorithm to demonstrate the uniform accuracy improvement of our method. Spectral clustering was proposed by Ng et al. (2001) and became one of the most widely used techniques in graph-based clustering due to its simple implementation and promising performance. The algorithm operates on the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{X}$ where $\mathbf{D}$ is the degree matrix and $\mathbf{X}$ is the adjacency matrix. For undirected graphs, the degree matrix is a diagonal matrix where each element represents the number of neighbors of the corresponding node. With a predetermined number of clusters K, the algorithm computes the eigenvectors of $\mathbf{L}$, denoted as $\mathbf{U}$. $\mathbf{U}$ should be a matrix with n rows where each row represents a node, and each column is the eigenvector. For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^K$ be the vector corresponding to the i-th row of $\mathbf{U}$. Finally, cluster the points y_i, $i = 1, \ldots, n$ with K-means algorithm (Lloyd, 1982) into clusters. Similar to VBEM, this method also needs to specify the number of clusters thus model selection is necessary.

Due to higher model complexity, estimating a graphon is challenging. Some works (Airoldi et al., 2013; Olhede and Wolfe, 2014; Latouche and Robin, 2016) have focused on the nonparametric perspective of this model and developed methods to estimate a graphon based on SBM approximation. These methods estimate a graphon function by partitioning vertices and computing the empirical frequency of edges across different blocks. Many algorithms put emphasis on model selection (Airoldi et al., 2013) or bandwidth determination (Olhede and Wolfe, 2014). Latouche and Robin (2016) proposed a variational Bayes approach to graphon estimation and used model averaging to generate a smooth estimate. Other SBM approximation methods include the works by Boppana (1987), Chaudhuri et al. (2012), COJA-OGHLAN (2010) etc, and rigorous consistency are guaranteed in these works for the estimation. Borgs and Chayes (2017) has reviewed the development of graphon theory over the last decades, and related models to estimate graphons on massive networks, where non-parametric approaches (Kallenberg, 1999; Bickel et al., 2011; CHOI et al., 2012) seem to out-perform SBM approximation.

After the block structure of a network is identified, most of the above methods simply use

the empirical connection probability within and between blocks to estimate Θ. When the number of nodes in a block is too small, the estimate can be highly inaccurate with a large variance. Latouche and Robin (2016) developed an alternative method under a Bayesian framework, where they put conjugate priors on the parameters (π, Θ). In particular, they assume $\theta_{ab} \sim \text{Beta}(\alpha_{ab}, \beta_{ab})$ independently for $a, b \in \{1, \ldots, K\}$, where the parameters $(\alpha_{ab}, \beta_{ab})$ in the prior are chosen *in priori*. Similar to the MLE, the connection probability θ_{ab} of each block is estimated separately and thus may suffer from the same high variance issue for blocks with a smaller number of nodes. To alleviate this difficulty, in Chapter 2, we propose a hierarchical model for network data to borrow information across different blocks. Under this model, we develop an empirical Bayes estimator for $\Theta = (\theta_{ab})$ and a model selection criterion for choosing the number of blocks. Empirical Bayes method is usually seen to have better performance when estimating many similar and variable quantities (Efron, 2010). This inspires our proposal as the connection probabilities can be similar across many different communities. By combining data from many blocks, estimates will be much more stable even if the number of nodes is small in each block.

1.3 Directed acyclic graph

Define a graph $G = (\mathbb{V}, \mathbb{E})$ consisting of a set of vertices $\mathbb{V} = \{V_1, \ldots, V_p\}$ and a set of edges $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$. The vertices can be encoded with random variables $\mathbf{X}$, such that a random variable X_i in $\mathbf{X} = (X_1, \ldots, X_p)$ represents a vertex V_i in $\mathbb{V}$. If both ordered pairs (V_i, V_j) and (V_j, V_i) are in $\mathbb{E}$, there is an undirected edge between V_i and V_j, denoted as $V_i - V_j$. If the ordered pair $(V_i, V_j) \in \mathbb{E}$ and $(V_i, V_j) \notin \mathbb{E}$, there is a directed edge denoted as $V_i \to V_j$.

We say that X_i is a neighbor of X_j when there is a directed or undirected edge between V_i and V_j. And X_i is a *parent* of X_j and X_j is a *child* of X_i when $V_i \to V_j$. Let $pa(X_i)$ denote the set of all parents of X_i and $ch(X_i)$ denote the set of all children of X_i. The union set of all the parents of every node in $ch(X_i)$ except X_i itself is called the spouses of X_i,

i.e. $sp(X_i) := \bigcup_{X_k \in ch(V_i)} pa(X_k) \backslash X_i$. The sets of parents, children and spouses of X_i are all the random variables in $\mathbf{X}$ that are dependent or conditionally dependent on X_i, the union of these sets forms the Markov blanket (Pearl, 1988) of X_i, written as $mb(X_i)$, which is consistent with the definition, i.e. the Markov blanket of a random variable X in a random variable set $S = \{X_1, \cdots, X_p\}$ is any subset S' of S conditioned on which other variables $S \backslash S'$ are independent with X,

$$X \perp\!\!\!\perp S \backslash S' \mid S'. \tag{1.18}$$

A *partially directed path* from V_1 to V_k is defined on a sequence of node $(V_1, \ldots, V_k)$ where $V_i \to V_{i+1}$ or $V_i - V_{i+1}$ for $i = 1, \ldots, k-1$. If there is no undirected edge in the path, the partially directed path is a *directed path*. A node V_i is an *ancestor* of V_j and V_j is a *descendant* of V_i if there is a directed path from V_i to V_j. A *v-structure* is defined as $V_i \to V_k \leftarrow V_j$ where V_k is the *collider*. A directed cycle is defined as a directed path from one node to itself, and a partially directed cycle is a partially directed path from one node to itself.

A graph G is a *Directed Acyclic Graph* (DAG) if G contains only directed edges without any directed cycles in the graph. DAG encodes conditional independencies of nodes in the graph. For a graph with p nodes $\{V_1, \ldots, V_p\}$, denoting X_i as the value of variable V_i, the joint probability distribution can be factorized as

$$P(X_1, \ldots, X_p) = \prod_{i=1}^{p} P(X_i | pa(X_i)), \tag{1.19}$$

the set of the variables $\mathbf{X}$ is also said to form a *Bayesian network*.

One of the most important concepts in the study of Bayesian networks is *d-seperation* introduced by Pearl (1988), who later introduced *Markov Equivalence* on DAG (Verma and Pearl, 1990a).

Definition 1.1 (d-connectedness) *A path between two vertices X and Y is active (d-connected) if there is a collider free (unblocked) path between them.*

Definition 1.2 (d-seperation) *Vertices X and Y are d-connected conditioned on a set of vertices $\mathbb{Z}$ ($X, Y \notin \mathbb{Z}$) if there is a collider-free path between X and Y that traverses no member of $\mathbb{Z}$. If no such path exists, then X and Y are d-separated (blocked) by $\mathbb{Z}$. Meanwhile, if a collider is a member of $\mathbb{Z}$, or has a descendant in $\mathbb{Z}$, it does not block any path that passes this collider.*

Definition 1.3 (Markov Equivalence) *Let G_1 and G_2 be two DAGs defined on the same set of vertices $\mathbb{V}$. G_1 and G_2 are Markov equivalent, denoted as $G_1 \sim G_2$, if they encode the same set of conditional independencies. For any vertices $X, Y \in \mathbb{V}$ and set of vertices $\mathbb{Z} \subseteq \mathbb{V}$, if X and Y are d-separated by $\mathbb{Z}$ in G_1 if and only if X and Y are d-separated by $\mathbb{Z}$ in G_2, then $G_1 \sim G_2$.*

A *Markov equivalence class* of DAGs encode the same Markov properties as in Definition 1.3. The *skeleton* of G is obtained by ignoring the direction of edges in G. A Markov equivalence class of DAGs is the set of DAGs that share the same skeleton and the same v-structures (Verma and Pearl, 1990a). Shown by Andersson et al. (1997), an equivalence class can be uniquely represented by a *completed partially directed acyclic graph* (cpDAG) that has both directed and undirected edges without directed cycles.

It is natural to interpret the direction in DAG as causal relations, and edges can represent the strength of "causal influence" (Hauser and Bühlmann, 2012). Causal structure estimation under the DAG framework has been the one of most popular problems in causal learning. Because of the ability of identifying confounding variables and modeling dependency relations, DAGs have been utilized as a tool in many fields, including epidemiology (Greenland et al., 1999), genomics (Gao and Cui, 2015), health and medical studies (Williams et al., 2018; Tennant et al., 2020) and social sciences (Velikova et al., 2014).

1.4 Structure learning algorithms on DAG

To put the estimation of DAG structure and causal relations into a statistical framework, structure equation model (SEM) (Pearl, 2009) is proposed. Tarka (2018) has reviewed historical and recent developments on it. SEM typically consists of a set of equations with explanatory variables. The joint distribution in DAGs can be represented by a set of structural equations

$$\mathbf{X} = \mathbf{A}\mathbf{X} + \mathbf{E}, \tag{1.20}$$

where the vectors $\mathbf{X}$ and $\mathbf{E}$ denote the random variables in the graph and their corresponding exogenous noises. The (i, j)-th element of the matrix $\mathbf{A}$ represents the strength of direct causal effect of node V_i on V_j. All the variables can be arranged in a *topological order*, which means that the nodes can be put into a linear ordered sequence, for every directed edge in the DAG, the parents always come before the children in the order. The matrix $\mathbf{A}$ can be converted into a strictly lower triangular matrix by permuting its rows and columns based on the *topological order*.

Suppose $\mathbf{X} = (X_1, \ldots, X_p) \sim \mathcal{N}_p(0, \mathbf{\Sigma})$, then there exists a weighted adjacency matrix $\mathbf{A} = (a_{ij})_{p\times p}$ and $\mathbf{\Omega} := \mathrm{diag}(\omega_j^2)$ such that $\mathbf{\Sigma} = (\mathbf{I} - \mathbf{A})^{-T}\mathbf{\Omega}(\mathbf{I} - \mathbf{A})^{-1}$, and

$$X_j = \sum_{i:X_i \in pa(X_j)} a_{ij}X_i + \epsilon_j, \quad \epsilon_j \sim \mathcal{N}(0, \omega_j^2), \quad j = 1, \ldots, p, \tag{1.21}$$

where ϵ_j are mutually independent, and independent from $pa(X_j)$.

(1.21) defines a linear Gaussian structure equation model (SEM). The weighted adjacency matrix $\mathbf{A}$ represents edge coefficients of DAG G. Structural learning methods on Gaussian SEM aim to estimate $(\mathbf{A}, \mathbf{\Omega})$ from i.i.d. samples of $\mathbf{X} = (X_1, \ldots, X_p)$.

The problem of DAG structure learning is challenging and rewarding, and a substantial amount of research has been dedicated to this problem. Some major obstacles include the high complexity, as the number of possible DAGs is super-exponential to the number of

vertices (Robinson, 1977); the acyclic nature of DAGs that heavily increases the computation time; the unidentifiability of the true DAG from the equivalence class.

There are two main approaches to structure learning of a DAG, constraint-based method and score-based method. The constraint-based method uses substantial amount of conditional independence tests to recover the conditional independence relationships between the variables. A well-known constraint based algorithm is the conceptual Inductive Causation algorithm (Verma and Pearl, 1990b) that was implemented in the PC algorithm (Spirtes et al., 2001). The PC algorithm recursively deletes edges based on conditional independence tests from a complete undirected graph, then orients v-structures in the skeleton and the remaining edges without any new conditional independencies or directed cycles being introduced. Other well-known constraint-based methods include the max-min hill-climbing (MMHC) algorithm (Tsamardinos et al., 2006) and fast causal inference (FCI) (Spirtes et al., 2001). Different from the PC algorithm, FCI allows the presence of latent variables and can sometimes discover unknown confounding variables.

The PC algorithm is one of the oldest algorithms that provides an architecture for future constraint-based method development, it sets several assumptions on the data distribution P on DAG G (Spirtes et al., 2001).

Assumption 1.1 (Causal Markov) *A node V in G is independent of all its non-descendent nodes in G when given $pa(V)$.*

Assumption 1.2 (Faithfulness) *No hidden conditional independence holds unless entailed by the Causal Markov relationship.*

Assumption 1.3 (Causal Sufficiency) *When two variables have their values observed in the data, the common causes should also have been observed.*

Under these assumptions, the PC algorithm is guaranteed to converge to the true Markov equivalence class in the large sample limit. The steps of the algorithm are illustrated in

Figure 1.1. The true DAG is shown in red (it is the only graph in its equivalence class), the estimation procedures are shown on the right. (I) PC first starts with a fully connected graph. (II) It deletes edges through independence tests, the edge V_a — V_b is removed since $V_a \perp\!\!\!\perp V_b$. (III) It then deletes edges through conditional independence tests, the edge V_a — V_d and V_b — V_d are removed since $V_a \perp\!\!\!\perp V_d|V_c$ and $V_b \perp\!\!\!\perp V_d|V_c$. (IV) PC identifies v-structures and (V) finalizes orientation without introducing new v-structures. The v-structure $V_a \to V_c \leftarrow V_b$ in the graph is identified by the fact that $V_a \not\perp\!\!\!\perp V_c|V_b$ whereas $V_a \perp\!\!\!\perp V_c$. In some cases, orientation rules do not apply to an undirected edge, the algorithm will leave the edge as undirected, thus the output is a partial DAG.

When the data is non-discrete, PC algorithm works well in linear Gaussian DAG, i.e. in the SEM each variable can be represented as a linear combination of other variables, and the noise term follows a Gaussian distribution. However, when the form of dependence is unknown, conditional independence tests face more difficulties. Recently many approaches based on functional causal models (FCMs) have been proposed, where the effect between variables are written as functions (Hoyer et al., 2009; Zhang and Hyvärinen, 2009; Zhang et al., 2015).

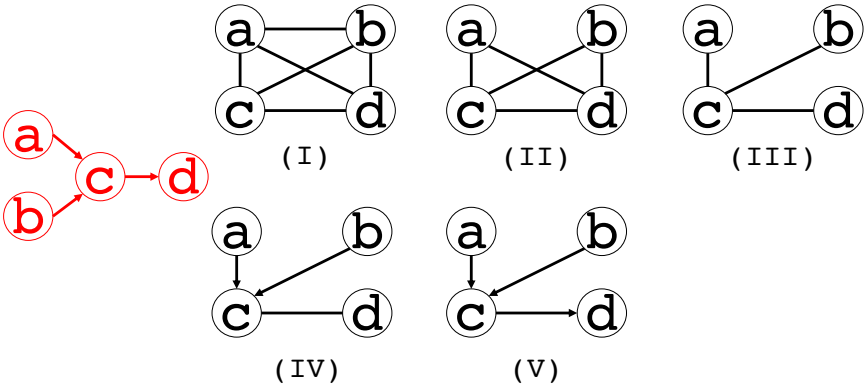


Figure 1.1: Illustration of PC algorithm mechanism.

The score-based methods typically consist of a score metric and a search algorithm. The score metric evaluates the goodness of fit of the estimated DAG to the data, and the

search algorithm maximizes the scoring function, which are designed in various ways, such as Bayesian information criteria (BIC) (D and Heckerman, 1997), l_1-penalized likelihood (Fu and Zhou, 2013) among others. Some popular score-based algorithms include K2 algorithm (Cooper and Herskovits, 2004), GES algorithm (Chickering, 2003) and CCDr algorithm (Aragam and Zhou, 2015).

Since DAGs in the same equivalence class encode the same set of conditional independencies, for DAGs that are Markov equivalent, conditional independence tests have the same outcomes, and score functions have the same values. Thus a single DAG cannot be differentiated from its equivalence class from observational data. This is known as the non-identifiability issue of linear Gaussian DAGs.

It is natural to believe that interventions on the data can solve the identifiability problem, however, interventions or sets of compared experiments are not easy to achieve in practice. Even though randomized controlled experiment is the gold standard tool for causal inference, the high expense can make the experiments impracticable. For instance, wet lab molecular biological experiments require a lot of effort and costs. Moreover, in many modern science studies, the huge amount of variables and a lack of background knowledge makes it more challenging. Facing this issue, researchers have made effort on causal inference from observational data (Verma and Pearl, 1990b; Chickering, 2003; Heckerman et al., 1995; Judea, 2010). Some studies have made identifiability assumptions and tried to solve the non-identifiability problem (Shimizu et al., 2006, 2011; Hoyer et al., 2009; Peters et al., 2014; Peters and Bhlmann, 2013; Monti et al., 2020; Wang and Zhou, 2021). Several methods exploit the non-linear and non-Gaussian structural equation models to identify the true DAG from purely observational data. Shimizu et al. (2006) developed a method to discover the true causal structure assuming non-Gaussian errors under the linear SEM and proposed a linear non-Gaussian acyclic model (LiNGAM), and further developed a direct method for learning a linear non-Gaussian structure equation model (DirectLiNGAM) (Shimizu et al., 2011) that requires no algorithmic parameters and converges fast. Hoyer et al. (2009) proved

that nonlinearity can distinguish single DAGs from their equivalence class, and proposed a non-linear noise additive (NNA) model, which was implemented by Peters et al. (2014) later. Zhang and Hyvärinen (2009) proposed a post-nonlinear (PNL) causal model: Taking two variable case as an example, the effect y is generated by a post-nonlinear transformation on the non-linear function of the cause x plus the noise e,

$$y = f_2(f_1(x) + e), \tag{1.22}$$

where f_1 is non-constant and f_2 is invertible, x and e are independent. It is a very general model, and both LiNGAM and NNA can be seen as special cases of PNL: In LiNGAM, f_1 is linear and e is non-Gaussian, f_2 is the identity mapping; in NNA, f_1 is non-linear and f_2 is the identity mapping. If the other direction $y \to x$ is true, the data generating process given by PNL is

$$x = g_2(g_1(y) + e'), \tag{1.23}$$

where g_1 is non-constant, g_2 is invertible, y and e' are independent. Zhang and Hyvärinen (2009) has established the conditions for the identifiability of causal directions in PNL causal model.

Assumption 1.4 *The data (x, y) are generated by the PNL causal model* (1.22)*, with f_1 and f_2 being third-order differentiable.*

Assumption 1.5 *Densities p_e and p_x are third-order differentiable, p_e is positive on $(-\infty, +\infty)$, and $(\log p_e)''$ is zero at most at some discrete points.*

Suppose the data were generated from the PNL model given the required conditions above, and f_1 is not invertible, then the causal directions can be determined in principle. There are five special situations that PNL fails to identify causal relations (e.g. linear Gaussian case), which are listed in the paper (Zhang and Hyvärinen, 2009). Despite the ability to orient undirected edges in most cases, these FCM methods are not computationally

efficient as in the linear case, thus a typical practice is to first use conditional independence tests to estimate a cpDAG and then apply the non-linear methods for further orientation.

Recent developments on causal discovery using observational data have been reviewed by Mooij et al. (2016) and Glymour et al. (2019). In Chapter 3 we also propose a simple method for causal discovery from observational data.

1.5 Outline

In this dissertation, we propose two methods related to graph modeling. The remaining chapters of the dissertation are organized as follows:

- In Chapter 2, we propose a hierarchical model and develop a novel empirical Bayes estimate of the connectivity matrix of a stochastic blockmodel to approximate the graphon function. Based on a regularized likelihood under our hierarchical model, we further introduce a new model selection criterion for choosing the number of communities. Numerical results on extensive simulations and two well-annotated social networks demonstrate the superiority of our approach in terms of estimation accuracy and model selection.

- In Chapter 3, we propose an edge orientation algorithm for causal discovery in cpDAGs. We go through our intuition behind the model, justify the theory, and verify the effectiveness of the algorithm by experiments on different simulated datasets and a real protein-signaling network. Numerical results demonstrate the improvement of our algorithm on existing structural learning methods.

- In Chapter 4, we conclude the dissertation with discussion and future work.

CHAPTER 2

An empirical Bayes approach to stochastic blockmodels and graphons

This chapter introduces a method for parameter estimation and model selection on undirected graphical models. In Section 2.1, we will develop our empirical Bayes method for the SBM and the graphon, focusing on connection probability estimation and model selection on the number of blocks. Then we will compare the performance of our methods with other existing methods on simulated data in Section 2.2 and on two real-world networks in Section 2.3.

Our method has two major novel components: 1) shrinkage estimation for connectivity parameters, and 2) a novel likelihood-based model selection criterion, both under our proposed hierarchical model. As demonstrated by extensive simulations and experiments on real-world data, these contributions give us substantial gain in estimation accuracy and model selection performance, especially for graphons. Moreover, our method is very easy to implement and does not cost much extra computational resources compared to existing approaches.

2.1 An Empirical Bayes method

Let us first consider the SBM. After the vertices of an observed network have been partitioned into clusters by a graph clustering algorithm, we develop an empirical Bayes estimate of the connection probability matrix Θ based on a hierarchical Binomial model. Under this

framework, we further propose a model selection criterion to choose the number of blocks. Our method consists of three steps:

- **Graph clustering** For a network with n vertices, cluster the vertices into K blocks by a clustering algorithm. Let $Z : [n] \to [K]$ denote the cluster assignment, where $[m] := \{1, \ldots, m\}$ for an integer m.
- **Parameter estimation** Given Z, we find an empirical Bayes estimate $\widehat{\Theta}_{\mathrm{EB}} = (\hat{\theta}_{ij}^{\mathrm{EB}})_{K \times K}$ by estimating the hyperparameters of the hierarchical binomial model.
- **Model Selection** Among multiple choices of K, we select the $\hat{K}$ that maximizes a penalized marginal likelihood under our hierarchical model.

In Section 2.1.3, we generalize our method to the graphon model, following the idea of SBM approximation to a graphon.

Given the $\mathbf{Z}$ estimated by either variational Bayes approach or spectral clustering, we will develop our hierarchical model and empirical Bayes estimates.

2.1.1 Estimating connection probabilities

In this subsection, we consider the SBM and assume a partition $Z : [n] \to [K]$ of the nodes is given, where K is the number of blocks. Note that $Z^{-1}(a)$ for $a \in [K]$ is the subset of nodes in the a-th cluster. Let

$$B_{ab} = \{(i, j) : (i, j) \in Z^{-1}(a) \times Z^{-1}(b), i < j\}$$

be the collection of node pairs in the (i, j)th block. According to the SBM, the connection probability between any $(i, j) \in B_{ab}$ is θ_{ab}. Recall that $\mathbf{X} = (X_{ij})$ is the observed adjacency matrix. Let $X_{ab}^B = \sum_{(i,j) \in B_{ab}} X_{ij}$ be the number of edges in block (a, b). Then, we have

$$X_{ab}^B \mid \theta_{ab} \sim \text{Binomial}(n_{ab}, \theta_{ab}), \tag{2.1}$$

where $n_{ab} = |B_{ab}| = |Z^{-1}(a)| \cdot |Z^{-1}(b)|$ for $a \neq b$ and $n_{aa} = |Z^{-1}(a)| \cdot (|Z^{-1}(a)| - 1)/2$ as self loops are not allowed. Based on the empirical frequency of edges in the block (a, b), we have an MLE for the edge connection probability

$$\hat{\theta}_{ab}^{\text{MLE}} = \frac{X_{ab}^B}{n_{ab}}, \qquad a, b \in \{1, \ldots, K\}. \tag{2.2}$$

When K is large, the number of nodes, and thus n_{ab}, in some blocks will be small, which leads to a high variance of the MLE. To stabilize the estimates, we may borrow information across blocks to improve estimation accuracy. To do this, we set up a hierarchical model by putting conjugate prior distributions on θ_{ab}. To accommodate the heterogeneity in θ_{ab}, we use two sets of hyperparameters so that the within and between-block connectivities are modeled separately:

$$\theta_{ab} \mid (\alpha_d, \beta_d) \sim \text{Beta}(\alpha_d, \beta_d), \quad a, b \in \{1, \ldots, K\}, \tag{2.3}$$

where $d = 0$ for $a = b$ and $d = 1$ for $a \neq b$, i.e. the diagonal and off-diagonal elements of the connectivity matrix Θ follow $\text{Beta}(\alpha_0, \beta_0)$ and $\text{Beta}(\alpha_1, \beta_1)$, respectively. The prior distribution (2.3) together with (2.1) defines the distribution $[\mathbf{X}, \Theta \mid (\alpha_d, \beta_d)_{d=0,1}]$. Here (α_d, β_d), $d = 0, 1$, are hyperparameters to be estimated by our method. A diagram of our model is shown in Figure 2.1. The connectivity parameters θ_{ab}, $a, b \in \{1, \ldots, K\}$, follow beta distributions of two sets of hyperparameters, i.e. (α_0, β_0) for diagonal blocks (red) and (α_1, β_1) for off-diagonal blocks, and the number of edges X_{ab}^B in a block, depends on θ_{ab} as in (2.1). Note that the use of two sets of hyperparameters is in line with common assumptions of the stochastic blockmodel, such as assortativity (Danon et al., 2005) or disassortativity, i.e. within-group connectivities are different than between-group connectivities.

The conditional posterior distribution of θ_{ab} given $(X_{ab}^B, \alpha_d, \beta_d)$ is

$$\theta_{ab} | (X_{ab}^B, \alpha_d, \beta_d) \sim \text{Beta}(\alpha_d + X_{ab}^B, \beta_d + n_{ab} - X_{ab}^B),$$

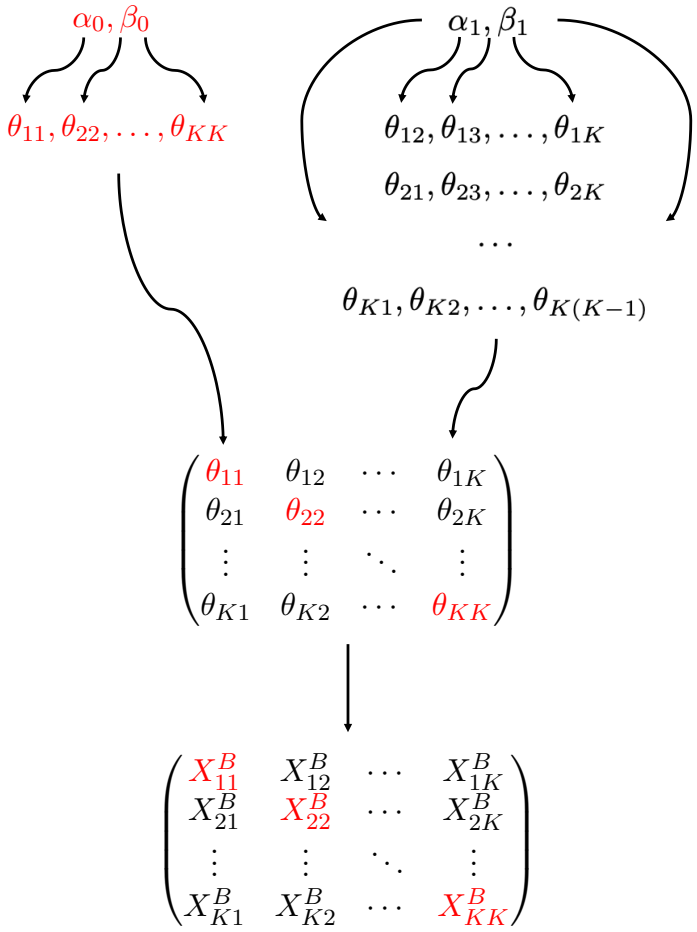


Figure 2.1: A diagram of the hierarchical model.

and the conditional posterior mean of θ_{ab} is

$$\begin{aligned}\hat{\theta}_{ab}^{\mathrm{EB}}(\alpha_d, \beta_d) &\equiv \mathbb{E}(\theta_{ab}|X_{ab}^B, \alpha_d, \beta_d) \\ &= \frac{\alpha_d + X_{ab}^B}{\alpha_d + \beta_d + n_{ab}} = \eta_{ab}\frac{\alpha_d}{\alpha_d + \beta_d} + (1 - \eta_{ab})\frac{X_{ab}^B}{n_{ab}},\end{aligned} \tag{2.4}$$

for $a, b \in \{1, \ldots, K\}$, where

$$\eta_{ab} = \frac{\alpha_d + \beta_d}{\alpha_d + \beta_d + n_{ab}} \in [0, 1] \tag{2.5}$$

is the shrinkage factor that measures the amount of information borrowed across blocks. When the variance among θ_{ab} across the blocks is high, α_d and β_d will be estimated to be small. Thus, η_{ab} will be close to 0 so that the estimate $\hat{\theta}_{ab}^{\mathrm{EB}}$ will be close to $\hat{\theta}_{ab}^{\mathrm{MLE}}$. When the variance among θ_{ab} is low, our estimates of α_d and β_d will be large, the shrinkage factor approaches 1, and eventually $\hat{\theta}_{ab}^{\mathrm{EB}}$ will become identical across all blocks. In this case, we are essentially pooling data in all blocks to estimate θ_{ab}. Generally speaking, the shrinkage factor η_{ab} is determined by the data through the estimation of the hyperparameters (α_d, β_d), and it leads to a good compromise between the above two extreme cases.

Given the partition Z from a graph clustering algorithm, we maximize the marginal likelihood of the observed adjacency matrix $\mathbf{X}$ to estimate the hyper-parameters (α_d, β_d) for $d = 0, 1$. Let $\mathbf{X}_{ab}$ denote the adjacency submatrix for nodes in the block (a, b) defined by the partition Z. Integrating over Θ, the marginal log-likelihood function for the diagonal blocks is

$$\begin{aligned}\mathcal{L}(\alpha_0, \beta_0|\mathbf{X}, Z) &= \sum_{a=1}^{K} \log \mathbb{P}(\mathbf{X}_{aa}|\alpha_0, \beta_0) \\ &= \sum_{a=1}^{K} \log \int_{\theta_{aa}} \mathbb{P}(\mathbf{X}_{aa}|\theta_{aa}) p(\theta_{aa}|\alpha_0, \beta_0) d\theta_{aa} \\ &= \sum_{a=1}^{K} \log \mathrm{Beta}(\alpha_0 + X_{aa}^B, \beta_0 + n_{aa} - X_{aa}^B) - K \log \mathrm{Beta}(\alpha_0, \beta_0),\end{aligned} \tag{2.6}$$

where $\mathrm{Beta}(x, y) = \int_0^1 t^{x-1}(1-t)^{y-1} dt$ is the beta function. Similarly, the marginal log-

likelihood function for the off-diagonal blocks is

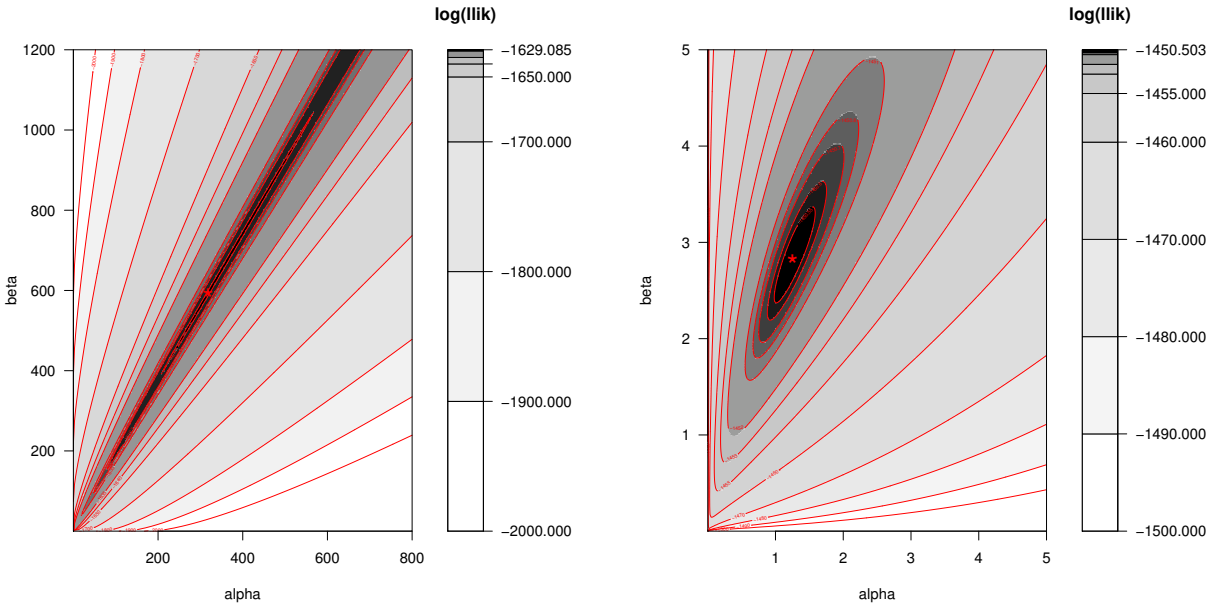
$$\begin{aligned}&\mathcal{L}(\alpha_1, \beta_1 | \mathbf{X}, Z) \\ &= \sum_{a<b} \log \text{Beta}(\alpha_1 + X_{ab}^B, \beta_1 + n_{ab} - X_{ab}^B) - \frac{1}{2} K(K-1) \log \text{Beta}(\alpha_1, \beta_1).\end{aligned} \tag{2.7}$$

We find the maximum likelihood estimates of the hyper parameters, i.e.

$$(\hat{\alpha}_d, \hat{\beta}_d) = \arg\max_{\alpha_d, \beta_d} \mathcal{L}(\alpha_d, \beta_d | \mathbf{X}, Z), \tag{2.8}$$

for $d = 0, 1$. Then we can estimate Θ by plugging the MLE of the hyper-parameters in (2.8) into (2.4), i.e.

$$\hat{\theta}_{ab}^{\text{EB}} = \begin{cases} \hat{\theta}_{aa}^{\text{EB}}(\hat{\alpha}_0, \hat{\beta}_0), & a = b \\ \hat{\theta}_{ab}^{\text{EB}}(\hat{\alpha}_1, \hat{\beta}_1), & a \neq b \end{cases}. \tag{2.9}$$



(a) Diagonal log-likelihood.

(b) Off-diagonal log-likelihood.

Figure 2.2: A typical contour plot of the likelihood functions.

Since the hyper-parameters are estimated by using all blocks, our empirical Bayes estimates of θ_{ab} also make use of information from all data to improve the accuracy. Though

(2.8) does not have a closed form solution, we can use an optimization algorithm such as bounded limited-memory BFGS (L-BFGS-B) (Byrd et al., 1995) to find the maximizer. As shown in Figure 2.2, the left figure $\mathcal{L}(\alpha_0, \beta_0)$ and the right figure $\mathcal{L}(\alpha_1, \beta_1)$ from a graph generated by a SBM with $n = 200$, $K = 5$, $\theta_{ab} = 0.7$ for $a = b$ and $\theta_{ab} = 0.3$ for $a \neq b$. The maximizers are marked as stars in the plots. For a typical dataset, the global maximizers can be easily found.

Suwan et al. (2016) developed a different empirical Bayesian method for SBMs under a random dot product graph formulation. They introduce K latent positions, $\nu_1, \ldots, \nu_K \in \mathbb{R}^d$, and define the connection probabilities by inner products between the latent positions, $\theta_{ab} = \langle \nu_a, \nu_b \rangle$ for $1 \leq a, b \leq K$. The prior distribution for ν_k is a multivariate Gaussian distribution $\nu_k \sim \mathcal{N}_d(\widehat{\mu}_k, \widehat{\Sigma}_k)$. In particular, the parameters $\widehat{\mu}_k, \widehat{\Sigma}_k$ in the prior are chosen by Gaussian mixture modeling of pre-estimated latent positions obtained via adjacency spectral embedding. Thus, these prior distributions are called *empirical* priors and they are used to model the uncertainty in the latent positions $\nu_1, \ldots, \nu_K$. In our method, the hyperparameters (α, β) in the beta prior distributions are not pre-estimated by a separate method, but instead are estimated under a coherent hierarchical model. In addition to modeling uncertainty in the connectivity probabilities θ_{ab}, the hyperparameters also lead to information sharing via shrinkage.

2.1.2 Selecting partitions

So far we have regarded the number of blocks K as given in our empirical Bayes method. The choice of K will certainly impact the performance of our method. If K is too small, for SBM many blocks will not be identified, and for graphon the approximated function will only have a small number of constant pieces, both leading to highly biased estimates. On the other hand, if K is too big, the number of vertices in each block will be very small, resulting in high variances. Thus, it is important to select a proper number of blocks to achieve the best estimation accuracy.

Our empirical Bayes approach under the hierarchical model also provides a useful criterion for this model selection problem. Note that (2.6) and (2.7) define the conditional likelihood of $\mathbf{X}$ given the hyperparameters (α_d, β_d) and the partition Z input from a graph clustering algorithm. We can compare this likelihood for different input partitions and select the best one.

Suppose we have m candidate partition schemes $Z_1, \ldots, Z_m$. Denote the corresponding number of communities by $K_1, \ldots, K_m$. Our goal is to choose the optimal partition that maximizes the joint likelihood of the observed adjacency matrix $\mathbf{X}$ and the partition Z with a penalty on the model complexity. To do this, we include Z in our model as in (1.2) and put a Jeffreys prior (Jeffreys, 1946) on π, i.e.

$$\pi \sim \text{Dirichlet}(\tau_1, \ldots, \tau_K), \quad \tau_1 = \ldots = \tau_K = 1/2.$$

For a partition Z with K communities, the joint likelihood of $\mathbf{X}$ and Z given the hyperparameters $(\alpha_0, \alpha_1, \beta_0, \beta_1)$ is

$$\begin{aligned} &\mathbb{P}(\mathbf{X}, Z|\alpha_0, \alpha_1, \beta_0, \beta_1) \\ &= \mathbb{P}(\mathbf{X}|Z, \alpha_0, \alpha_1, \beta_0, \beta_1) \int \mathbb{P}(Z|\pi)p(\pi)d\pi \\ &= \mathbb{P}(\mathbf{X}|Z, \alpha_0, \alpha_1, \beta_0, \beta_1) \frac{\Gamma(\sum_{i=1}^K \tau_i) \prod_{i=1}^K \Gamma(n_i + \tau_i)}{\Gamma(n + \sum_{i=1}^K \tau_i) \prod_{i=1}^K \Gamma(\tau_i)}, \end{aligned} \tag{2.10}$$

after marginalizing out the parameter π, where n_i is the number of nodes in cluster i defined by the partition Z. Maximizing over the hyperparameters leads to the MLE $(\hat{\alpha}_0, \hat{\alpha}_1, \hat{\beta}_0, \hat{\beta}_1)$ defined in (2.8). Evaluating the likelihood (2.10) at the estimated hyperparameters, we define the goodness-of-fit part for our model selection criterion as

$$\begin{aligned} J_Z &= \log \mathbb{P}(\mathbf{X}, Z|\hat{\alpha}_0, \hat{\alpha}_1, \hat{\beta}_0, \hat{\beta}_1) \\ &= \sum_{d \in \{0,1\}} \mathcal{L}(\hat{\alpha}_d, \hat{\beta}_d|\mathbf{X}, Z) + \log \frac{\Gamma(\sum_{i=1}^K \tau_i) \prod_{i=1}^K \Gamma(n_i + \tau_i)}{\Gamma(n + \sum_{i=1}^K \tau_i) \prod_{i=1}^K \Gamma(\tau_i)}, \end{aligned} \tag{2.11}$$

where $\mathcal{L}(\hat{\alpha}_d, \hat{\beta}_d|\mathbf{X}, Z)$ is as in (2.6) and (2.7) for $d = 0, 1$. Following the ICL-like (integrated complete likelihood) criterion in Mariadassou et al. (2010), we add two penalty terms to

control model complexity: The first term corresponds to a penalty on the number of parameters in π and the second the number of parameters in Θ. Therefore, our model selection criterion is to choose the partition

$$\hat{Z} = \underset{Z \in \{Z_1, \ldots, Z_m\}}{\arg\max} \left\{ J_Z - \frac{1}{2} \left[(K-1) \log n + \frac{K(K+1)}{2} \log \frac{n(n-1)}{2} \right] \right\}, \tag{2.12}$$

where K is the number of clusters defined by the partition Z. As we have mentioned in the introduction, there are quite a few graph clustering algorithms, and the performance of many of them is highly dependent on the input number of partitions. Our criterion for selecting the number of clusters applies to any method used for the node clustering step, and thus it protects our method from inferior input node clustering results. The ICL model selection criterion (2.12) is indeed an approximation to the marginal likelihood $\mathbb{P}(\mathbf{X}|K)$ (Mariadassou et al., 2010). The joint likelihood depends on the EB estimates of the hyperparameters, which is unique to our hierarchical model. While the VBEM criterion (Latouche et al., 2012) uses a standard SBM likelihood without a hierarchical structure nor estimation of priors. We can easily apply other penalty terms in various model selection criteria to our likelihood, and fully expect similar behavior in terms of selecting the number of clusters, since most of them approximate in some way the marginal likelihood or the Bayes factor.

2.1.3 Graphon estimate

Now we assume that the true model is a graphon as in (1.1). We use an SBM with K blocks as an approximation to the graphon, i.e., we approximate $W(u, v)$ by a piecewise constant function: We divide the unit interval $[0, 1]$ into K pieces based on π so that the length of the k-th piece is π_k. Let the endpoints of these pieces be $c_k = \sum_{i=1}^{k} \pi_i$ for $k = 1, \cdots, K$ and put $c_0 \equiv 0$. Then the graphon function defined on $[0, 1] \times [0, 1]$ is approximated by a $K \times K$ blockwise constant function,

$$\widetilde{W}(u, v) = \theta_{ab} \qquad \text{if } (u, v) \in [c_{a-1}, c_a) \times [c_{b-1}, c_b).$$

To estimate a graphon W, we first run a clustering algorithm to estimate a partition

Z and then apply the empirical Bayes method to obtain $\hat{\theta}^{\mathrm{EB}}_{ab}$. Let n_k denote the size of the the k-th cluster of vertices. We calculate its proportion to estimate π_k by $\hat{\pi}_k = n_k/n$ and compute the cumulative proportion $\hat{c}_k = \sum_{i=1}^{k} \hat{\pi}_i$ for $k = 1, \cdots, K$. Define a binning function,

$$\mathrm{bin}(x) = 1 + \sum_{k=1}^{K} \mathbb{I}(c_k \leq x), \tag{2.13}$$

and the graphon W is then estimated by

$$\widehat{W}(x, y) = \hat{\theta}^{\mathrm{EB}}_{\mathrm{bin}(x),\mathrm{bin}(y)}, \qquad x, y \in [0, 1). \tag{2.14}$$

As shown by Bickel and Chen (2009), the graphon is not identifiable in the sense that any measure-preserving transformation on $[0, 1]$ will define an equivalent random graph. Following their method, imposing the constraint that

$$g(x) = \int_0^1 W(x, y)dy$$

is nondecreasing leads to identifiability. For SBM approximation, the corresponding constraint is that

$$g(l) = \sum_{k=1}^{K} \pi_k \theta_{lk} \tag{2.15}$$

is nondecreasing in l. This constraint can be satisfied by relabeling the K clusters of nodes.

As for the SBM, selecting a proper number of clusters K is important for the estimation of a graphon. We will apply the same model selection criterion (2.12) to choose the optimal partition Z and the associated K among a collection of partitions.

2.2 Results on simulated graphs

In this section we present numerical results on graphs simulated from stochastic blockmodels and graphon functions. We compare our method with other existing methods in terms

of estimating connection probabilities (Section 2.1.1) and model selection for choosing the number of clusters (Section 2.1.2).

For stochastic blockmodels, we compare our estimated connectivity matrix $\widehat{\Theta}_{\text{EB}}$ (2.9) to the maximum likelihood estimate $\widehat{\Theta}_{\text{MLE}}$ as in (2.2) and the variational Bayes inference $\widehat{\Theta}_{\text{VBEM}}$ from Latouche et al. (2012). Variational Bayes inference provides a closed-form approximate posterior distribution for (π, Θ) by minimizing the KL divergence between an approximated and the underlying distributions of $[Z \mid \mathbf{X}]$. It constructs point estimates for the parameters based on EM iterations. We compute the mean squared error (MSE)

$$\text{MSE} = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i} (\widehat{\Theta}'_{ij} - \Theta'_{ij})^2 \tag{2.16}$$

of an estimated $n \times n$ connection probability matrix $\widehat{\Theta}'$. Here, $\Theta' = (\Theta_{ij})_{n \times n}$ is the true connection probability matrix among the n nodes, i.e. $\Theta'_{ij} = \theta_{ab}$ if $Z^*(i) = a$ and $Z^*(j) = b$ for $i, j = 1, \ldots, n$, where Z^* is the true partition, and $\widehat{\Theta}'_{ij} = \hat{\theta}_{ab}$ if $Z(i) = a$ and $Z(j) = b$. For graphons, $\widehat{W}(x, y)$ is estimated by SBM approximation as in Section 2.1.3, and correspondingly the MSE is calculated as

$$\text{MSE} = \iint_0^1 (W(x, y) - \widehat{W}(x, y))^2 dx dy. \tag{2.17}$$

Due to the nonidentifiability of graphons, the MSE is calculated after relabeling node clusters based on the constraint (2.15) to make $\widehat{W}$ comparable to W.

We compare our model selection criterion (2.12) to the variational Bayes method developed by Latouche et al. (2012) (VBEM) and the cross validation risk of precision parameter (CVRP) in Airoldi et al. (2013). The CVRP is defined as

$$\mathcal{J}_{\text{CVRP}}(K) = \frac{2K}{n-1} - \frac{(n+1)K}{n-1} \sum_{i=1}^{K} (\frac{n_i}{n})^2, \tag{2.18}$$

where n_i is the number of vertices in group i. Then, the number of clusters K is selected by minimizing the risk $\mathcal{J}_{\text{CVRP}}$, i.e.

$$\hat{K}_{\text{CVRP}} = \arg\min_K \mathcal{J}_{\text{CVRP}}(K). \tag{2.19}$$

We use $\mathcal{J}_{\text{EB}}$, $\mathcal{J}_{\text{VBEM}}$ and $\mathcal{J}_{\text{CVRP}}$ to denote the three criteria above respectively.

2.2.1 Results on SBMs

We designed a constrained SBM that generates affiliation networks, i.e. two vertices within the same community connect with probability λ, and from different communities with probability $\epsilon < \lambda$. We also added a parameter $\rho \in (0, 1]$ to control the sparsity of the graph. The corresponding true connectivity matrix is

$$\Theta^* = \rho \begin{pmatrix} \lambda & \epsilon & \cdots & \epsilon \\ \epsilon & \lambda & \cdots & \vdots \\ \vdots & & \ddots & \epsilon \\ \epsilon & \cdots & \epsilon & \lambda \end{pmatrix}_{K^* \times K^*},$$

where K^* is the number of communities.

To generate dense graphs (model 1), we set $\lambda = 0.9$, $\epsilon = 0.1$, and $\rho = 1$. We generated graphs with $n = 200$ vertices and the number of communities $K^* \in \{10, 11, \ldots, 18\}$. For each choice of K^*, we generated 100 networks independently. For each network, all the nodes were randomly divided into K^* clusters with equal probability $1/K^*$, and then connected according to the connectivity matrix Θ^* and their cluster labels. Note that the simulated node clusters had very different sizes, ranging between 7 and 35, due to the high variance in block size.

We also used $\lambda = 0.9$, $\epsilon = 0.1$ and $\rho = 0.2$ to generate sparse graphs (model 2), while keeping $K^* = 10$ but changing the network size $n \in \{200, 250, 300, 350, 400, 450\}$. For each network size n, we followed the same procedure as in model 1 and generated 100 networks independently.

For a simulated graph, we applied the variational Bayes algorithm (Latouche et al., 2012) with an input number of clusters $K = 1, \ldots, 20$, from which we obtained K communities and a Bayesian estimate $\widehat{\Theta}_{\text{VBEM}}(K)$ of the connecting probabilities among the $K \times K$ blocks. Given the estimated communities by the variational Bayes algorithm, we found $\widehat{\Theta}_{\text{MLE}}(K)$ as in (2.2) and our empirical Bayes estimate $\widehat{\Theta}_{\text{EB}}(K)$ as in (2.9) and compared them to the

VBEM estimate. As the estimates were functions of K, so were their MSEs as defined in (2.16). Let $\text{MSE}_{\text{MLE}}(K)$ be the mean squared error of the MLE by plugging $\widehat{\Theta}_{\text{MLE}}(K)$ into (2.16), where each element in $\widehat{\Theta}'_{ij}$ is given by $\widehat{\Theta}_{\text{MLE}}(K)$ and the partition Z. Then we define $\tilde{K}$ as the number of clusters that minimizes the MSE of the MLE, i.e.

$$\tilde{K} = \arg\min_{K} \text{MSE}_{\text{MLE}}(K) \tag{2.20}$$

over the input range of K. As our focus is on the estimation of the connectivity matrix, here the MSE of parameter estimates is the gold standard criterion. Although degree distributions and sub-graph patterns are very useful in evaluating a network model, they are less relevant to our contribution since we are not proposing a new model but instead improving the estimation accuracy of existing models.

For the 100 graphs generated under the same matrix Θ^*, they share the same K^* while each one of them defines a corresponding $\tilde{K}$. Both K^* and $\tilde{K}$ were used in our comparisons on model selection criteria for the number of blocks. In particular, for a general graphon, K^* may not be clearly defined and in such a case, $\tilde{K}$ serves as the reference for comparison.

For dense graphs (model 1), as shown in Figure 2.3, we compared the MSEs (2.16) of the three estimates of Θ to the true connectivity matrix and presented the ratio of the MSE of our EB estimate to the MSEs of the MLE and VBEM estimate. In the figure the true number of blocks K^* (marked in red) ranges from 10 to 18 and the results for graphs with each K^* are shown in a panel. For the 100 graphs generated under each K^*, the MSE ratios of the estimates $\widehat{\Theta}_{\text{MLE}}$ and $\widehat{\Theta}_{\text{VBEM}}$ over $\widehat{\Theta}_{\text{EB}}$ are plotted against the input number of blocks K chosen in the clustering step. For dense stochastic blockmodels, the accuracy of MLE and that of VBEM were close, whereas EB gave better estimates for almost all K values, i.e. MSE ratios were smaller than 100%. We see a significantly smaller MSE ratio when K is close to K^*, especially when K^* is relatively small. For example, the MSE ratios EB/MLE and EB/VBEM were lower than 10% at $K = K^*$ when $K^* = 10, \ldots, 15$. When K^* went bigger, such as $K^* = 17, 18$ in the simulation, the $\tilde{K}$ for most of the graphs was less than

K^*, and the MSE ratios reached a minimum level at some $K < K^*$, which was slightly above 50%.

Table 2.1 presents the model selection results on the simulated dense graphs from model 1, among the $\hat{K}$ chosen by (a) CVRP, (b) VEBM, and (c) EB. Each row in a table reports the frequency of $\hat{K}$ across 100 graphs. The last two columns report two mean absolute deviations, the minimum of which among the three methods is in boldface for each K^*. We define E_{K^*} and $E_{\tilde{K}}$ as the average deviation of the selected number of blocks $\hat{K}$ from K^* and from $\tilde{K}$ respectively, i.e.

$$E_{K^*} = \frac{1}{M}\sum_{t=1}^{M}|\hat{K}_t - K^*|, \qquad E_{\tilde{K}} = \frac{1}{M}\sum_{t=1}^{M}|\hat{K}_t - \tilde{K}_t|, \tag{2.21}$$

where $t \in \{1, \ldots, M\}$ is the index of the graphs generated under the same Θ^*, $\hat{K}_t$ is the estimated number of clusters by a model selection criterion, and $\tilde{K}_t$ is the $\tilde{K}$ defined by (2.20) for the t-th graph. When K^* was small, such as $10 \leq K^* \leq 13$, $\mathcal{J}_{\text{VBEM}}$ and $\mathcal{J}_{\text{EB}}$ gave the same results, where both accurately selected $\hat{K} = K^*$ as the optimal number of blocks. As K^* increased, $\mathcal{J}_{\text{EB}}$ outperformed $\mathcal{J}_{\text{VBEM}}$, and was comparable to $\mathcal{J}_{\text{CVRP}}$ in terms of E_{K^*}. In fact, for a limited graph size $n = 200$ here, the average number of vertices in each block will be smaller as K^* increases, making it hard for small communities to be detected. Therefore, $\tilde{K}$ may better reflect the number of clusters that fit well the observed network. Considering this, we see $\mathcal{J}_{\text{EB}}$ had both smaller E_{K^*} and $E_{\tilde{K}}$ than $\mathcal{J}_{\text{VBEM}}$ in general, which indicates the superiority of our model selection method. $\mathcal{J}_{\text{CVRP}}$ showed relatively stable performance in terms of E_{K^*} and $E_{\tilde{K}}$, but the results were not satisfactory for small K^*. Figure 2.4 and Table 2.2 provides the values of the model selection criteria. In the figure, the true number of blocks K^* (marked as red) ranges from 10 to 18 and the results for graphs with each K^* are shown in a panel. $\mathcal{J}_{\text{CVRP}}$, $\mathcal{J}_{\text{VBEM}}$, $\mathcal{J}_{\text{EB}}$ are all standardized to $[0, 1]$, and $\mathcal{J}_{\text{CVRP}}$ is taken negative, thus the model is selected by the maximizer of each criterion. For the 100 graphs generated under each set of parameters, the values of three criteria are plotted against the input number of blocks $K = 1, \ldots, 20$ used in clustering. The number of clusters selected by

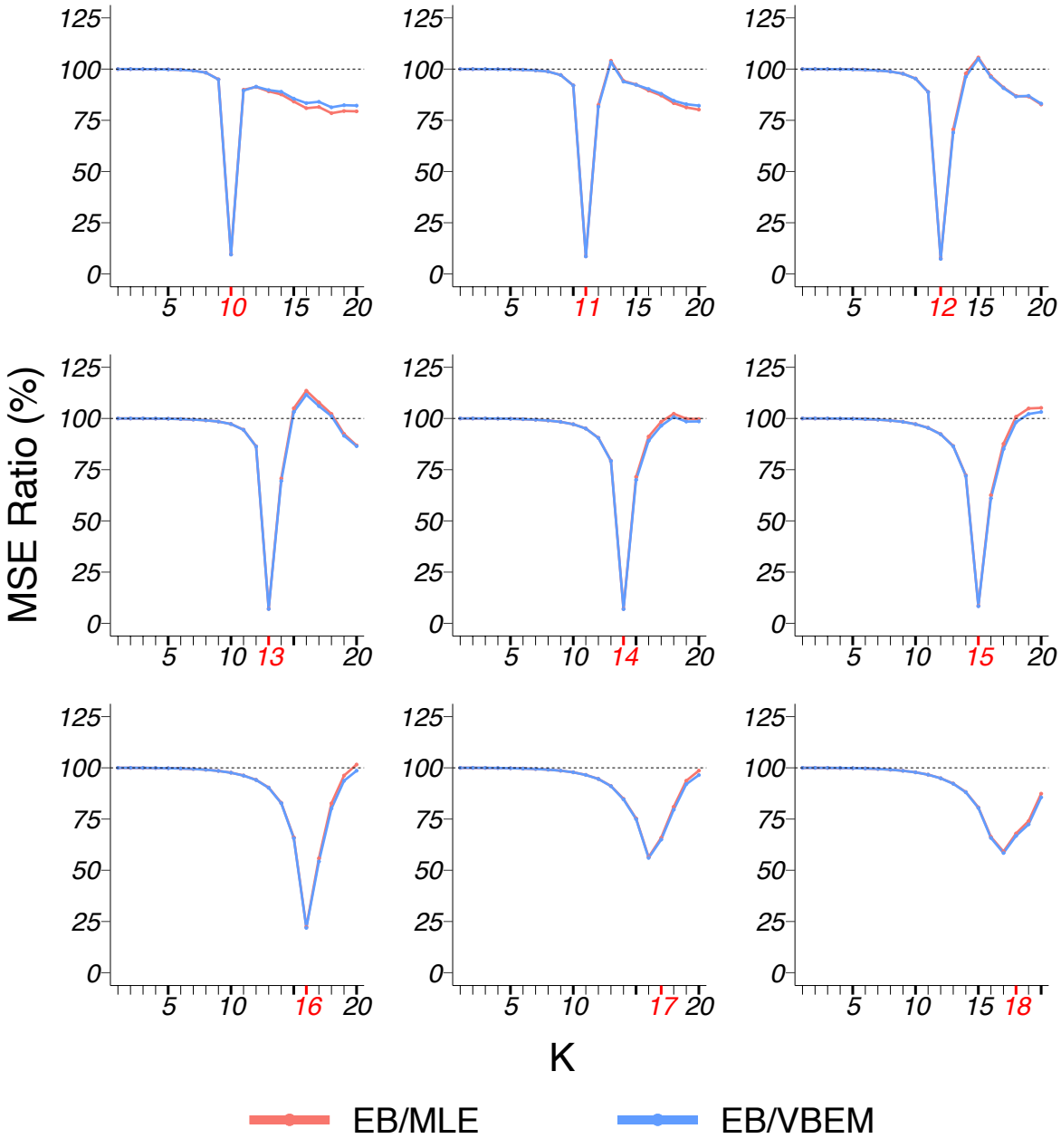


Figure 2.3: MSE ratios in model 1 simulation.

Table 2.1: Model selection comparison for model 1.

(a) CVRP

$K^*\backslash\hat{K}$	8	9	10	11	12	13	14	15	16	17	18	E_{K^*}	$E_{\hat{K}}$
10		99	1									0.99	0.99
11			100									1.00	1.00
12			3	96	1							1.02	1.02
13					67	33						0.67	0.67
14					6	93		1				1.06	1.06
15						23	77					1.23	1.26
16						2	13	85				1.17	1.31
17							1	29	70			**1.31**	**1.33**
18								3	87	10		**1.93**	**1.27**

(b) VBEM

$K^*\backslash\hat{K}$	8	9	10	11	12	13	14	15	16	17	18	E_{K^*}	$E_{\hat{K}}$
10			100									**0.00**	**0.00**
11				100								**0.00**	**0.00**
12					100							**0.00**	**0.00**
13						100						**0.00**	**0.00**
14						4	96					0.04	0.45
15					1	2	35	62				0.39	0.85
16						1	28	53	18			1.12	1.26
17						6	53	35	6			2.59	2.61
18					1	7	32	44	16			3.33	2.67

(c) EB

$K^*\backslash\hat{K}$	8	9	10	11	12	13	14	15	16	17	18	E_{K^*}	$E_{\hat{K}}$
10			100									**0.00**	**0.00**
11				100								**0.00**	**0.00**
12					100							**0.00**	**0.00**
13						100						**0.00**	**0.00**
14							100					**0.00**	**0.00**
15							1	99				**0.01**	**0.04**
16								30	70			**0.30**	**0.44**
17								33	67			1.33	1.35
18								1	95	4		1.97	1.31

EB is highlighted by the dashed lines. In the table, the average MSE of the estimates $\widehat{\Theta}$ to Θ by the three methods of the 100 graphs generated under different K^* are shown in each row. The input number of clusters K ranges from 10 to 18, for each K^* and K the minimal MSE among three methods is boldfaced.

In summary, from the simulation results on dense graphs (model 1), EB has demonstrated the highest estimation accuracy, especially when the clustering algorithm finds the true number of communities, and the EB model selection criterion generally selects the best model.

Detecting the true number of blocks for a sparse graph (model 2) is harder because of fewer edge connections in a block. Thus, we fixed $K^* = 10$ and varied the network size n from 200 to 450. In terms of estimation accuracy, Figure 2.6 illustrates the results for graphs with each network size n are shown in a panel, plotted in the same format as Figure 2.3. The figure shows that our EB estimate had better performance than MLE in almost all the cases (except when $K = 1$ under which the two estimates were identical), and the MSE ratio kept decreasing as K increased. In particular, for $K = K^* = 10$, the MSE ratio of EB over MLE was about 95%. If the number of blocks is overestimated (say $K > 15$), the MSE ratio can drop to $< 90\%$. When compared to VBEM, for a small network size n and a small number of blocks K, EB estimates can be slightly less accurate ($< 5\%$ increase in MSE), but as K increases and becomes close to K^*, the MSE ratio goes down to the same level as that of EB over MLE. As reported in Table 2.3, for all the cases $\mathcal{J}_{\text{EB}}$ achieved the best model selection performance with the smallest E_{K^*} and $E_{\tilde{K}}$ among the three methods. This highlights the usefulness of our model selection criterion for the more challenging sparse graph settings. More details are shown in Table 2.4 and Figure 2.5, which are in the same formats as Table 2.2 and Figure 2.4.

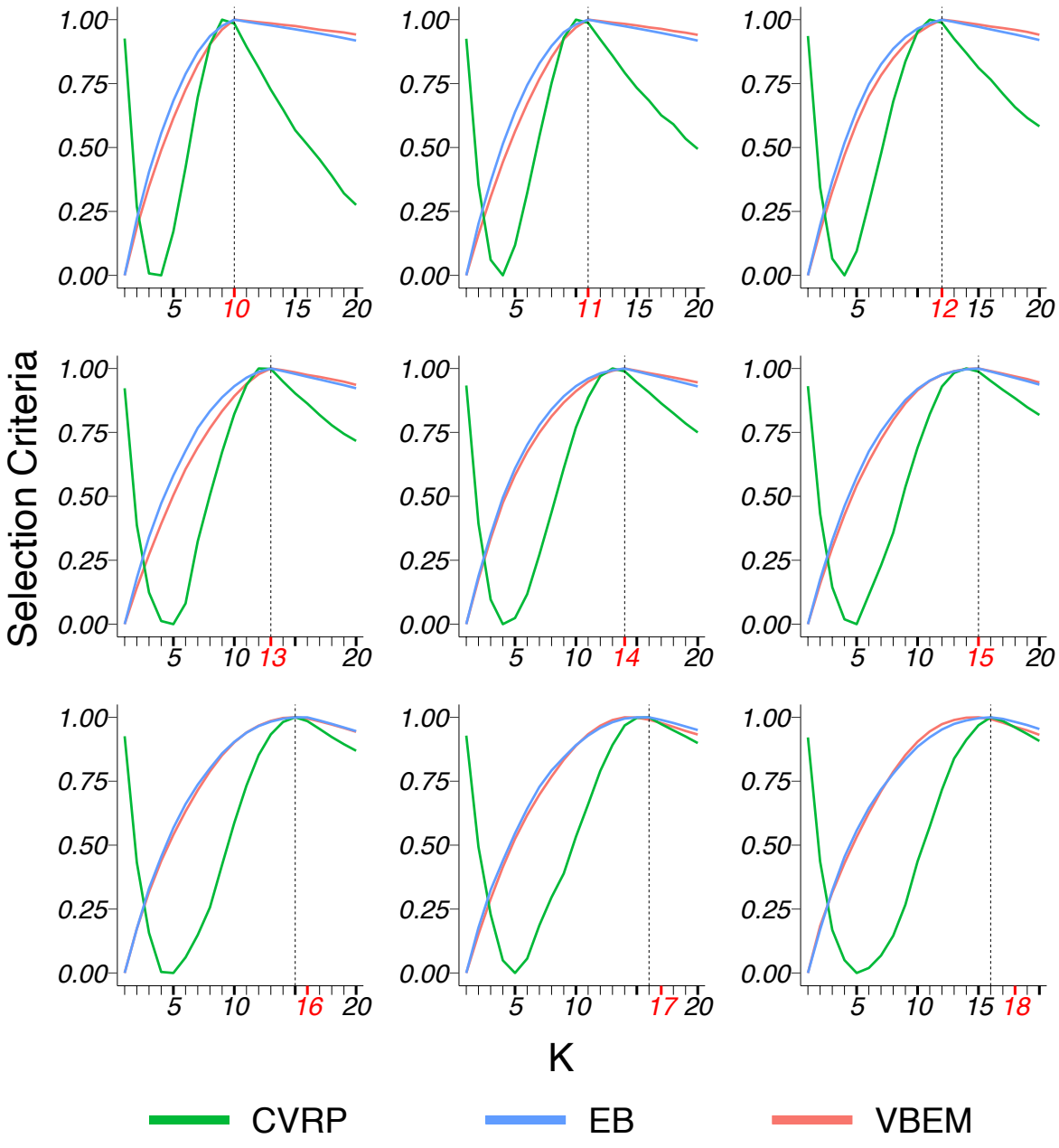


Figure 2.4: Values of model selection criteria in model 1 simulation.

Table 2.2: MSE values for model 1.

$K^* \setminus K$		9	10	11	12	13	14	15	16	17	18	19	20	
	MLE	304	25	35	46	56	67	77	93	109	120	131	148	$\times 10^{-5}$
10	VBEM	304	25	36	46	55	66	76	90	106	116	127	143	$\times 10^{-5}$
	EB	**289**	**2**	**32**	**42**	**50**	**59**	**65**	**75**	**89**	**94**	**105**	**117**	$\times 10^{-5}$
	MLE	538	249	30	41	**52**	60	72	85	96	112	123	139	$\times 10^{-5}$
11	VBEM	538	249	31	41	**52**	60	72	85	95	111	120	136	$\times 10^{-5}$
	EB	**522**	**229**	**3**	**34**	54	**56**	**66**	**76**	**83**	**94**	**100**	**111**	$\times 10^{-5}$
	MLE	683	417	215	35	44	55	**67**	79	89	99	112	130	$\times 10^{-5}$
12	VBEM	683	417	216	36	45	56	**67**	79	89	100	112	129	$\times 10^{-5}$
	EB	**668**	**398**	**192**	**3**	**31**	**54**	70	**76**	**81**	**86**	**97**	**107**	$\times 10^{-5}$
	MLE	1012	708	434	212	41	51	**62**	**75**	**85**	**98**	111	128	$\times 10^{-5}$
13	VBEM	1012	708	434	212	42	52	63	76	86	99	112	128	$\times 10^{-5}$
	EB	**996**	**689**	**410**	**183**	**3**	**36**	65	85	91	100	**103**	**111**	$\times 10^{-5}$
	MLE	921	692	487	305	167	48	60	71	81	**91**	**102**	**114**	$\times 10^{-5}$
14	VBEM	921	692	488	305	168	49	61	73	82	93	104	116	$\times 10^{-5}$
	EB	**906**	**673**	**464**	**276**	**133**	**3**	**43**	**65**	**79**	94	**102**	**114**	$\times 10^{-5}$
	MLE	969	733	543	389	262	149	57	68	78	**88**	**99**	**114**	$\times 10^{-5}$
15	VBEM	969	733	543	390	263	150	58	70	81	91	102	116	$\times 10^{-5}$
	EB	**953**	**712**	**518**	**359**	**227**	**108**	**5**	**43**	**69**	89	104	120	$\times 10^{-5}$
	MLE	1044	842	653	495	361	237	137	70	77	89	101	**114**	$\times 10^{-5}$
16	VBEM	1044	842	653	495	362	238	138	72	80	92	104	117	$\times 10^{-5}$
	EB	**1028**	**822**	**629**	**466**	**326**	**197**	**91**	**16**	**43**	**74**	**97**	115	$\times 10^{-5}$
	MLE	1132	907	705	541	388	264	190	124	124	125	137	146	$\times 10^{-5}$
17	VBEM	1132	907	705	541	389	265	191	125	126	128	140	149	$\times 10^{-5}$
	EB	**1116**	**887**	**681**	**512**	**354**	**224**	**143**	**70**	**82**	**102**	**129**	**144**	$\times 10^{-5}$
	MLE	1097	905	733	583	458	348	247	161	137	142	141	164	$\times 10^{-5}$
18	VBEM	1097	905	733	583	458	348	248	162	139	144	144	167	$\times 10^{-5}$
	EB	**1082**	**886**	**709**	**553**	**423**	**307**	**199**	**107**	**81**	**96**	**104**	**143**	$\times 10^{-5}$

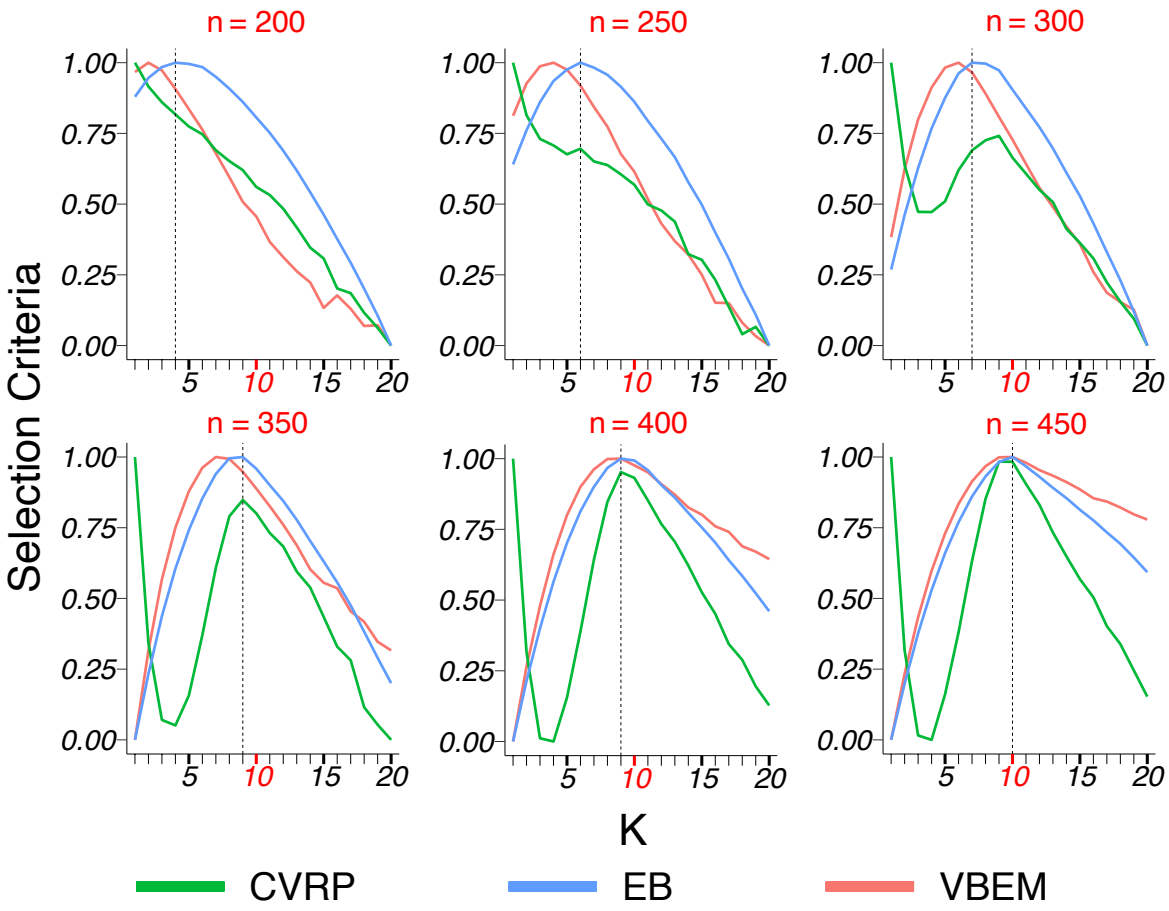


Figure 2.5: Values of model selection criteria in model 2 simulation.

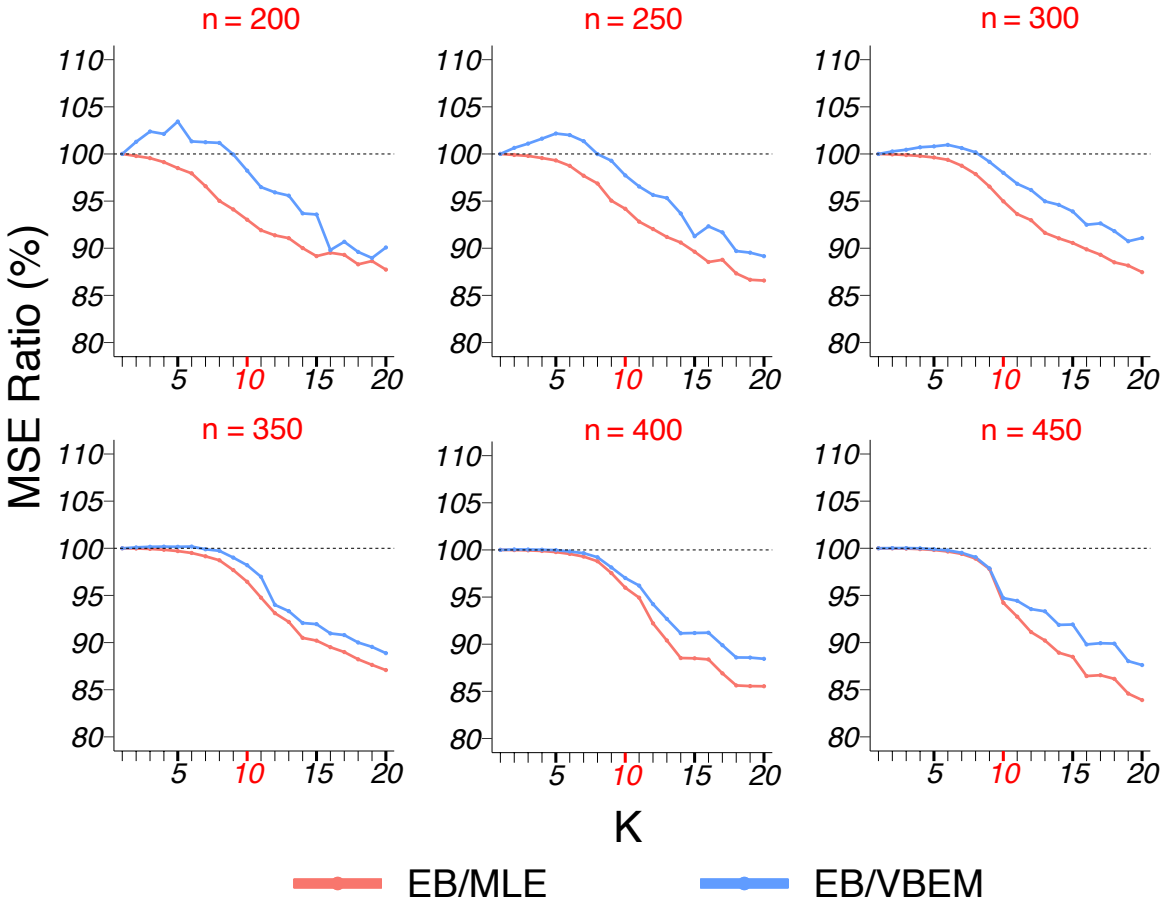


Figure 2.6: MSE ratios in model 2 simulation.

Table 2.3: Model selection comparison for model 2.

(a) CVRP

$n\backslash\hat{K}$	1	2	3	4	5	6	7	8	9	10	11	12	E_{K^*}	$E_{\tilde{K}}$
200	100												9	2.84
250	100												9	6.86
300	95								1	4			8.56	8.84
350	71							1	14	14			6.55	8.17
400	37								28	35			3.61	5.21
450	17								11	71	1		1.65	2.50

(b) VBEM

$n\backslash\hat{K}$	1	2	3	4	5	6	7	8	9	10	11	12	E_{K^*}	$E_{\tilde{K}}$
200	28	51	19	2									8.05	2.18
250		8	30	42	13	6			1				6.16	4.04
300			1	11	31	37	20						4.36	4.59
350						14	43	36	7				2.64	4.22
400							3	34	47	14	1	1	1.27	2.83
450							1	3	37	52	6	1	0.54	1.25

(c) EB

$n\backslash\hat{K}$	1	2	3	4	5	6	7	8	9	10	11	12	E_{K^*}	$E_{\tilde{K}}$
200		6	12	24	29	24	4	1					**5.31**	**2.09**
250				6	21	38	21	12	2				**3.82**	**2.20**
300					1	13	32	35	18	1			**2.41**	**2.74**
350							2	31	47	20			**1.15**	**2.81**
400								10	38	48	3	1	**0.63**	**2.13**
450								2	13	78	7		**0.24**	**0.97**

Table 2.4: MSE values for model 2.

$n \setminus K$		1	2	3	4	5	6	7	8	9	10	11	12	
	MLE	**229**	225	225	229	236	238	247	256	266	269	273	283	$\times 10^{-5}$
200	VBEM	**229**	**221**	**219**	**222**	**225**	**230**	**236**	**240**	251	255	260	269	$\times 10^{-5}$
	EB	**229**	224	224	227	233	233	239	243	**250**	**250**	**251**	**258**	$\times 10^{-5}$
	MLE	**230**	217	204	192	186	182	184	187	192	193	202	206	$\times 10^{-5}$
250	VBEM	**230**	**215**	**201**	**188**	**181**	**176**	**177**	**181**	184	186	194	198	$\times 10^{-5}$
	EB	**230**	217	203	192	185	180	180	**181**	**182**	**182**	**187**	**190**	$\times 10^{-5}$
	MLE	**231**	**208**	186	165	147	130	121	120	118	124	128	133	$\times 10^{-5}$
300	VBEM	**231**	**208**	**185**	**164**	**145**	**128**	**119**	**117**	114	120	123	128	$\times 10^{-5}$
	EB	**231**	**208**	186	165	147	130	120	**117**	**113**	**117**	**119**	**123**	$\times 10^{-5}$
	MLE	**231**	**202**	175	151	129	110	94	81	74	75	77	77	$\times 10^{-5}$
350	VBEM	**231**	**202**	**174**	**150**	**128**	**109**	**93**	**80**	**73**	73	76	76	$\times 10^{-5}$
	EB	**231**	**202**	175	**150**	**128**	110	**93**	**8**0	**73**	**72**	**73**	**72**	$\times 10^{-5}$
	MLE	**232**	**201**	171	**141**	**117**	95	**77**	61	49	44	43	45	$\times 10^{-5}$
400	VBEM	**232**	**201**	**170**	**141**	**117**	95	**77**	**60**	48	44	42	44	$\times 10^{-5}$
	EB	**232**	**201**	**170**	**141**	**117**	**94**	**77**	**60**	**47**	**43**	**41**	**42**	$\times 10^{-5}$
	MLE	**232**	**199**	**167**	**139**	**114**	**91**	**70**	52	36	25	26	28	$\times 10^{-5}$
450	VBEM	**232**	**199**	**167**	**139**	**114**	**91**	**70**	52	36	25	26	27	$\times 10^{-5}$
	EB	**232**	**199**	**167**	**139**	**114**	**91**	**70**	**51**	**35**	**24**	**24**	**26**	$\times 10^{-5}$

2.2.2 Results on graphon models

Following the same design as in Latouche and Robin (2016), we choose a graphon function

$$W(x, y) = \rho\lambda^2(xy)^{\lambda-1}$$

with two parameters $\lambda \leq 1/\sqrt{\rho}$. Here, ρ controls the sparsity of the graph, as the expected number of edges is proportional to ρ, and λ controls the concentration of the degrees, so that more edges will concentrate on fewer nodes if λ is large. We chose $\rho \in \{10^{-1}, 10^{-1.5}, 10^{-2}\}$ and $\lambda \in \{2, 3, 5\}$, and simulated graphs of size $n = 100$ (model 3) and of size $n = 316$ ($\approx 10^{2.5}$) (model 4). For each network, we used SBM approximation (Section 2.1.3) with the number of clusters $K = 1, 2, \ldots, 10$. Using (2.20), we also defined $\tilde{K}$ as the number of blocks that minimizes the MSE (2.17) of the MLE.

The MSE ratios between our EB estimate and the other two competing methods, MLE and VBEM, are shown in Figure 2.7 for graphs of size $n = 100$ and Figure 2.8 for graphs of size $n = 316$. In general, our EB method achieved higher accuracy with smaller MSEs than the other two methods. For most cases, our EB estimate was more accurate than the MLE, with the MSE ratios between 60% and 100%. Compared to VBEM, our EB estimate achieved substantially smaller MSEs with ratios below 20%. For both graph sizes, the improvement of the EB method over the other two competitors was especially significant when the graph was sparse (ρ small).

The model selection results are reported in Table 2.5. In the table the reported is the mean absolute deviation $E_{\tilde{K}}$ for graphs generated under each combination of (ρ, λ). The minimal $E_{\tilde{K}}$ among the three methods is highlighted in boldface. Since the true number of communities under the graphon model is not clearly defined, we used $\tilde{K}$ as the ground-truth to evaluate model selection performance. For both $n = 100$ and $n = 316$, the mean absolute deviation $E_{\tilde{K}}$ (2.21) of the $\hat{K}$ selected by our criterion $\mathcal{J}_{EB}$ was either the smallest or was very close to the smallest value among the three methods. While EB and VBEM were generally comparable, CVRP showed unstable performance as its $E_{\tilde{K}}$ could be much

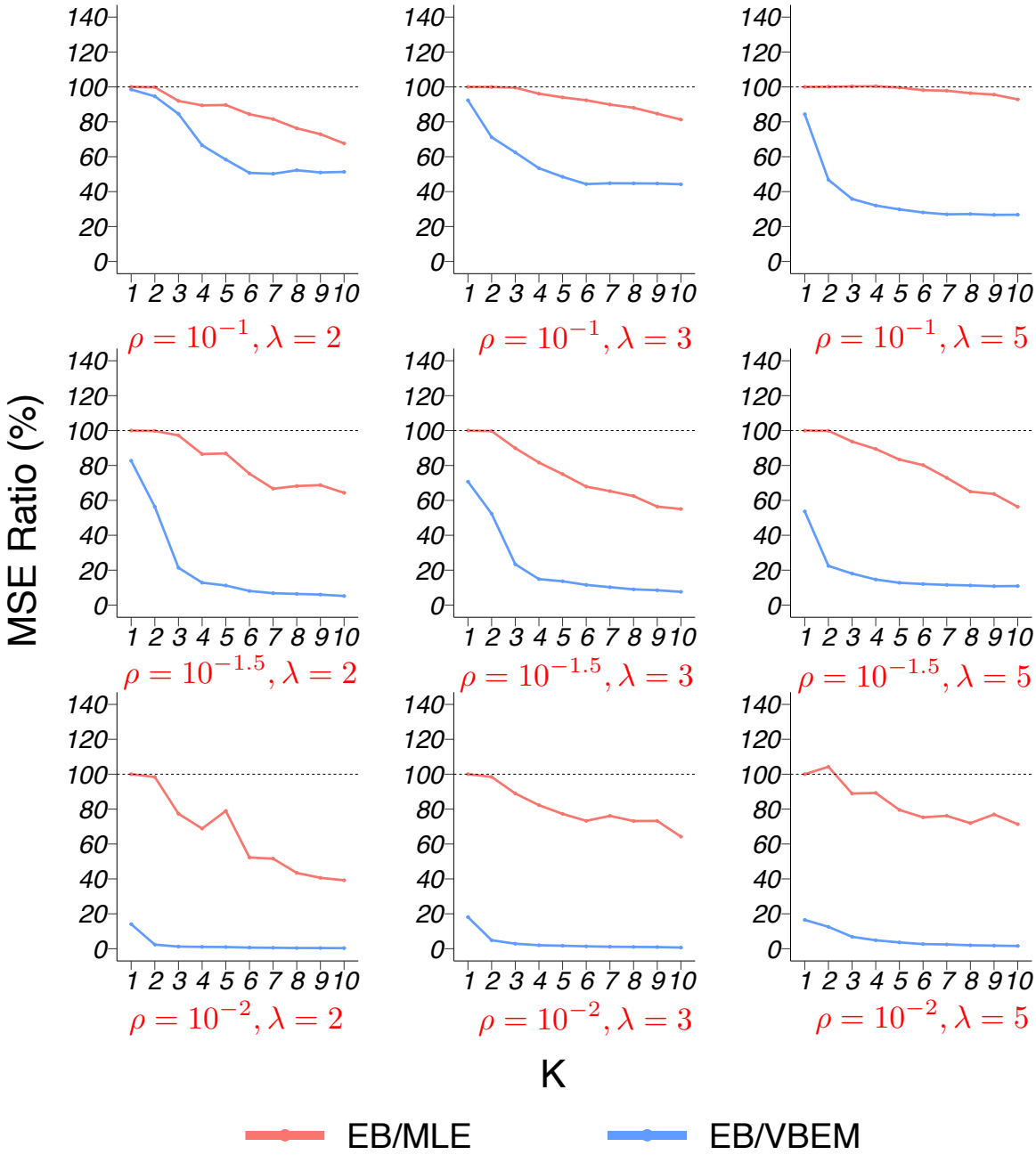


Figure 2.7: MSE ratios in model 3 simulation with graph size $n = 100$.

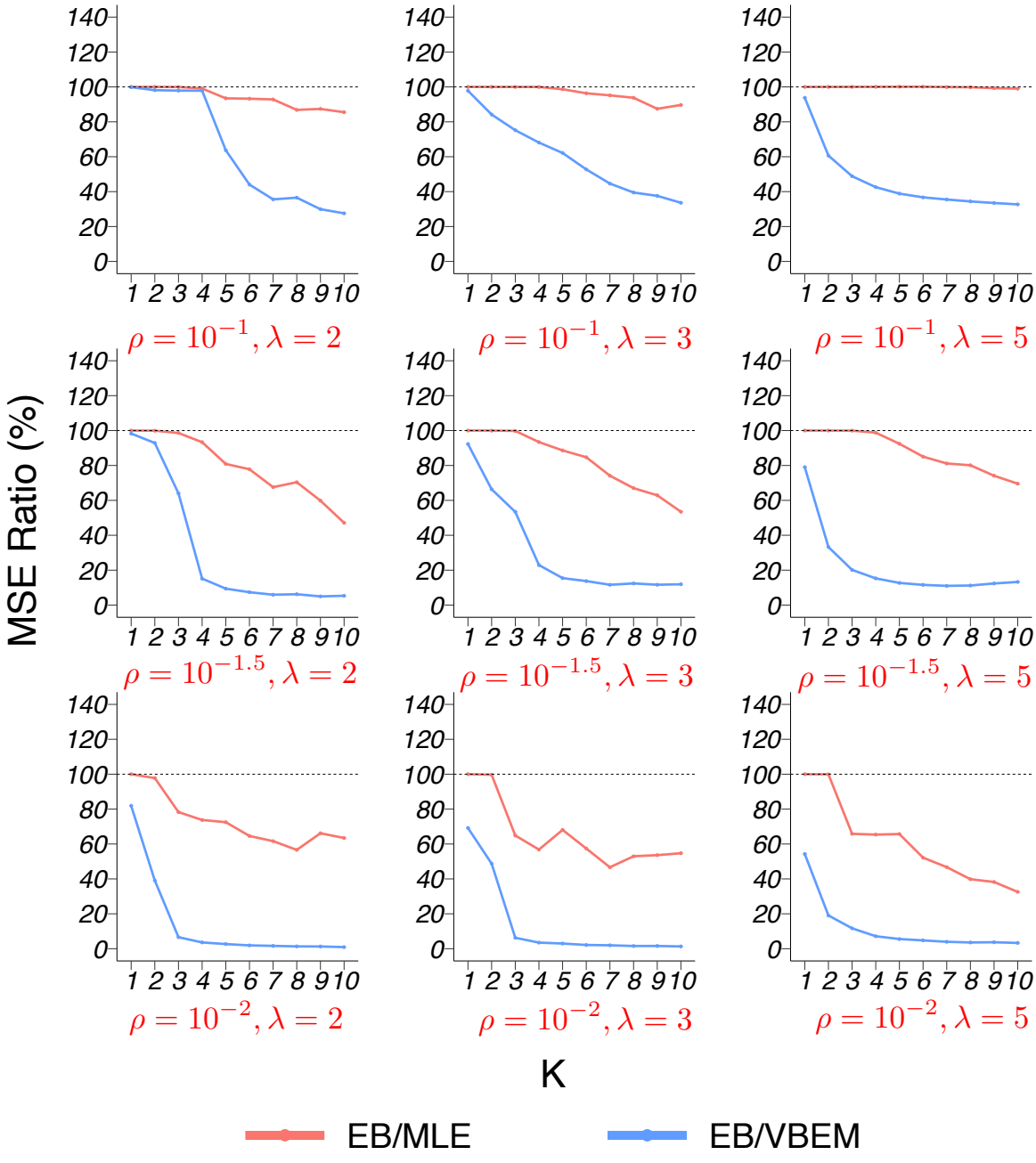


Figure 2.8: MSE ratios in model 4 simulation with graph size $n = 316$.

Table 2.5: Model selection comparison for graphons.

		$n = 100$			$n = 316$		
		CVRP	VBEM	EB	CVRP	VBEM	EB
$\rho = 10^{-1}$	$\lambda = 2$	1.16	**0.96**	1.11	4.92	2.55	**2.38**
	$\lambda = 3$	5.42	**1.54**	2.03	5.8	1.92	**1.91**
	$\lambda = 5$	3.88	**1.28**	1.63	7.43	1.66	**1.50**
$\rho = 10^{-1.5}$	$\lambda = 2$	2.01	1.86	**1.83**	4.76	3.72	**3.70**
	$\lambda = 3$	1.81	1.02	**0.95**	3.93	2.02	**1.96**
	$\lambda = 5$	2.05	1.03	**0.98**	4.58	**1.60**	1.79
$\rho = 10^{-2}$	$\lambda = 2$	0.86	**0.85**	0.86	2.56	**2.24**	2.25
	$\lambda = 3$	**1.41**	1.45	1.48	1.48	1.35	**1.31**
	$\lambda = 5$	**1.52**	1.61	1.7	2.77	1.72	**1.67**

larger than the other two methods in some cases (such as $\rho = 10^{-1}$ and $\rho = 10^{-1.5}$).

Similar to the illustration of the SBM results, under the case where the graph size (number of nodes) n equals 100, Table 2.6 demonstrates the average MSE of the estimates $\widehat{\Theta}$ to Θ^* by the three methods. The average values of the 100 graphs generated under different set of parameters ρ and λ are shown in each row. The input number of clusters K ranges from 1 to 10, for each ρ, λ and K the minimal MSE among three methods is boldfaced. Figure 2.9 shows the model selection curves. With the graphon $W(x, y) = \rho\lambda^2(xy)^{\lambda-1}$, $\rho \in \{10^{-1}, 10^{-1.5}, 10^{-2}\}$ and $\lambda \in \{2, 3, 5\}$, the results for graphs with each set of parameters ρ and λ are shown in a panel. $\mathcal{J}_{\text{CVRP}}$, $\mathcal{J}_{\text{VBEM}}$, $\mathcal{J}_{\text{EB}}$ are all standardized to $[0, 1]$, and $\mathcal{J}_{\text{CVRP}}$ is taken negative, thus the model is selected by the maximizer of each criterion. For the 100 graphs generated under each set of parameters, the values of three criteria are plotted against the input number of blocks $K = 1, \ldots, 10$ used in clustering. The number of clusters selected by EB is highlighted by the dashed lines. Table 2.7 and Figure 2.10 provide the details of the results when $n = 316$ in the same way.

Table 2.6: MSE values for model 3.

ρ	λ		1	2	3	4	5	6	7	8	9	10	
10^{-1}	2	MLE	**80**	**26**	28	34	43	47	57	70	80	91	$\times 10^{-4}$
		VBEM	81	28	31	46	65	78	92	103	114	120	$\times 10^{-4}$
		EB	**80**	**26**	**26**	**31**	**38**	**40**	**46**	**54**	**58**	**62**	$\times 10^{-4}$
	3	MLE	**229**	78	**47**	50	57	62	73	83	94	104	$\times 10^{-4}$
		VBEM	248	109	75	90	110	130	147	163	177	191	$\times 10^{-4}$
		EB	**229**	**77**	**47**	**48**	**53**	**58**	**66**	**73**	**79**	**84**	$\times 10^{-4}$
	5	MLE	**680**	**247**	**161**	**144**	**144**	148	153	164	171	184	$\times 10^{-4}$
		VBEM	806	529	451	452	483	517	555	581	612	638	$\times 10^{-4}$
		EB	**680**	248	**161**	145	**144**	**145**	**150**	**158**	**163**	**171**	$\times 10^{-4}$
$10^{-1.5}$	2	MLE	**8**	**11**	**13**	14	17	18	21	22	24	25	$\times 10^{-4}$
		VBEM	10	20	59	98	133	169	204	234	269	305	$\times 10^{-4}$
		EB	**8**	**11**	**13**	**13**	**15**	**14**	**14**	**15**	**16**	**16**	$\times 10^{-4}$
	3	MLE	**23**	**13**	16	19	24	28	32	34	40	42	$\times 10^{-4}$
		VBEM	32	25	63	102	132	166	202	236	265	304	$\times 10^{-4}$
		EB	**23**	**13**	**15**	**15**	**18**	**19**	**21**	**21**	**23**	**23**	$\times 10^{-4}$
	5	MLE	**68**	**30**	32	35	40	45	54	64	70	88	$\times 10^{-4}$
		VBEM	127	132	166	213	258	301	339	369	415	452	$\times 10^{-4}$
		EB	**68**	**30**	**30**	**31**	**33**	**36**	**39**	**42**	**45**	**49**	$\times 10^{-4}$
10^{-2}	2	MLE	**82**	190	239	336	346	429	425	442	498	490	$\times 10^{-6}$
		VBEM	58	795	1525	2183	2821	3461	4098	4705	5248	5799	$\times 10^{-5}$
		EB	**82**	**187**	**185**	**232**	**273**	**224**	**219**	**192**	**202**	**192**	$\times 10^{-6}$
	3	MLE	**23**	44	53	58	68	71	68	73	74	67	$\times 10^{-5}$
		VBEM	127	889	1644	2397	3103	3814	4509	5164	5774	6413	$\times 10^{-5}$
		EB	**23**	**43**	**47**	**48**	**53**	**52**	**52**	**53**	**54**	**43**	$\times 10^{-5}$
	5	MLE	**68**	**136**	153	160	173	165	174	171	160	166	$\times 10^{-5}$
		VBEM	413	1129	1998	2941	3804	4667	5413	6215	6943	7586	$\times 10^{-5}$
		EB	**68**	142	**136**	**143**	**137**	**124**	**132**	**123**	**123**	**118**	$\times 10^{-5}$

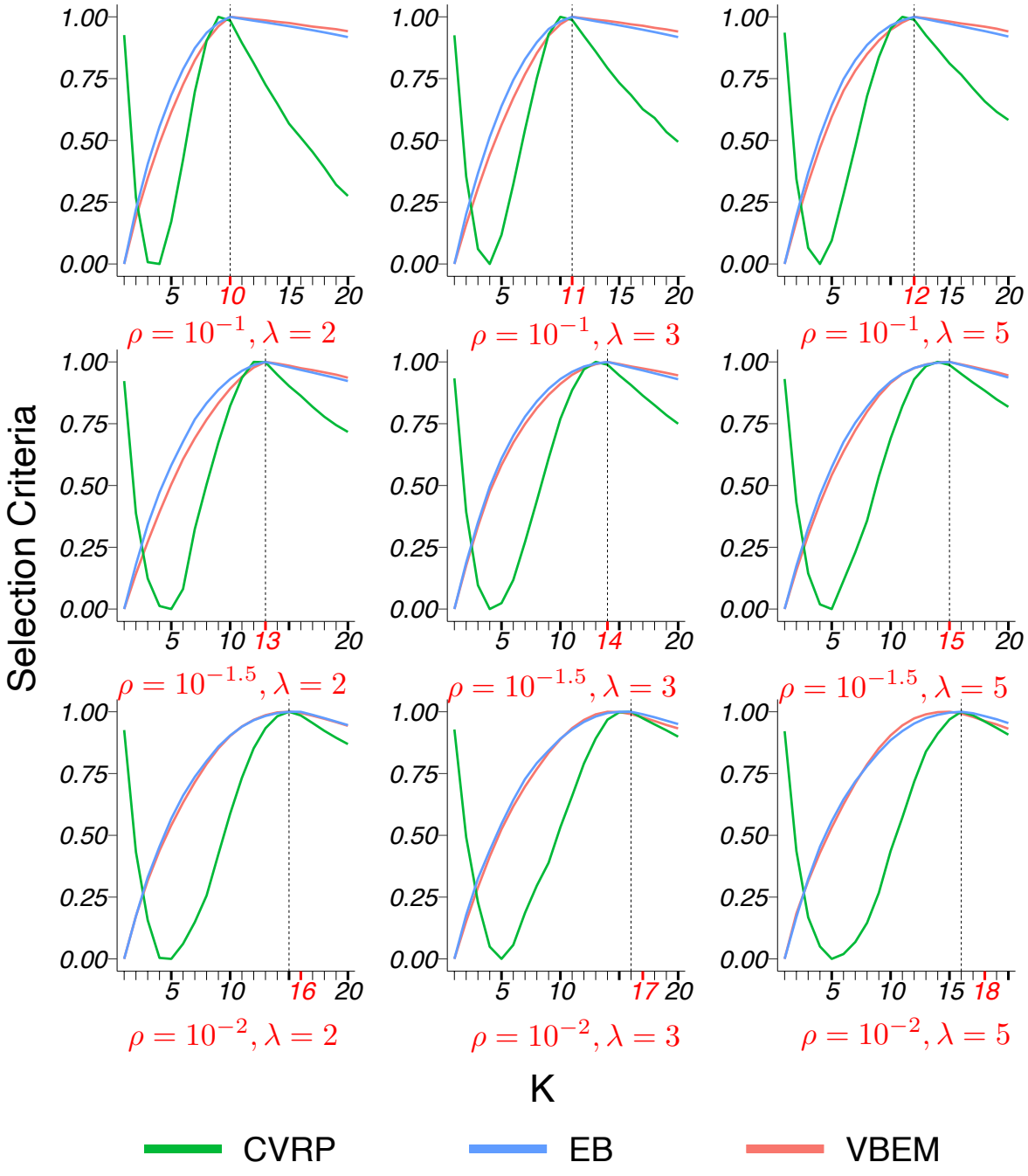


Figure 2.9: Values of model selection criteria in model 3 simulation.

Table 2.7: MSE values for model 4.

ρ	λ		1	2	3	4	5	6	7	8	9	10	
		MLE	**782**	**240**	**120**	77	82	84	87	114	112	121	$\times 10^{-5}$
	2	VBEM	783	244	122	78	121	177	228	270	327	374	$\times 10^{-5}$
		EB	**782**	**240**	**120**	**76**	**77**	**78**	**81**	**99**	**98**	**103**	$\times 10^{-5}$
		MLE	**225**	**69**	**33**	**20**	**15**	**14**	15	15	18	18	$\times 10^{-4}$
10^{-1}	3	VBEM	230	82	44	29	23	26	31	36	42	48	$\times 10^{-4}$
		EB	**225**	**69**	**33**	**20**	**15**	**14**	**14**	**14**	**16**	**16**	$\times 10^{-4}$
		MLE	**674**	**230**	**131**	**103**	**91**	**85**	**83**	**83**	84	85	$\times 10^{-4}$
	5	VBEM	719	379	268	242	234	232	234	241	248	257	$\times 10^{-4}$
		EB	**674**	**230**	**131**	**103**	**91**	**85**	**83**	**83**	**83**	**84**	$\times 10^{-4}$
		MLE	**78**	**24**	21	22	28	32	39	45	52	78	$\times 10^{-5}$
	2	VBEM	80	26	32	138	242	342	450	512	623	686	$\times 10^{-5}$
		EB	**78**	**24**	**20**	**21**	**23**	**25**	**27**	**32**	**31**	**37**	$\times 10^{-5}$
		MLE	**225**	**68**	**33**	37	43	52	65	84	96	125	$\times 10^{-5}$
$10^{-1.5}$	3	VBEM	244	102	62	151	248	323	416	455	517	559	$\times 10^{-5}$
		EB	**225**	**68**	**33**	**35**	**38**	**44**	**48**	**56**	**60**	**67**	$\times 10^{-5}$
		MLE	**674**	**213**	**102**	73	79	93	106	119	151	180	$\times 10^{-5}$
	5	VBEM	853	641	506	472	577	680	785	854	903	947	$\times 10^{-5}$
		EB	**674**	**213**	**102**	**72**	**73**	**79**	**86**	**96**	**112**	**125**	$\times 10^{-5}$
		MLE	**8**	**8**	15	15	17	18	20	22	20	18	$\times 10^{-5}$
	2	VBEM	10	20	176	305	476	626	778	915	1065	1250	$\times 10^{-5}$
		EB	**8**	**8**	**12**	**11**	**13**	**12**	**13**	**12**	**14**	**12**	$\times 10^{-5}$
		MLE	**23**	**9**	16	20	21	25	33	29	33	31	$\times 10^{-5}$
10^{-2}	3	VBEM	33	19	168	325	471	656	792	994	1121	1301	$\times 10^{-5}$
		EB	**23**	**9**	**11**	**11**	**14**	**14**	**15**	**15**	**18**	**17**	$\times 10^{-5}$
		MLE	**67**	**23**	35	39	44	60	70	88	105	130	$\times 10^{-5}$
	5	VBEM	124	120	196	358	523	655	829	984	1084	1277	$\times 10^{-5}$
		EB	**67**	**23**	**23**	**26**	**29**	**31**	**33**	**35**	**40**	**42**	$\times 10^{-5}$

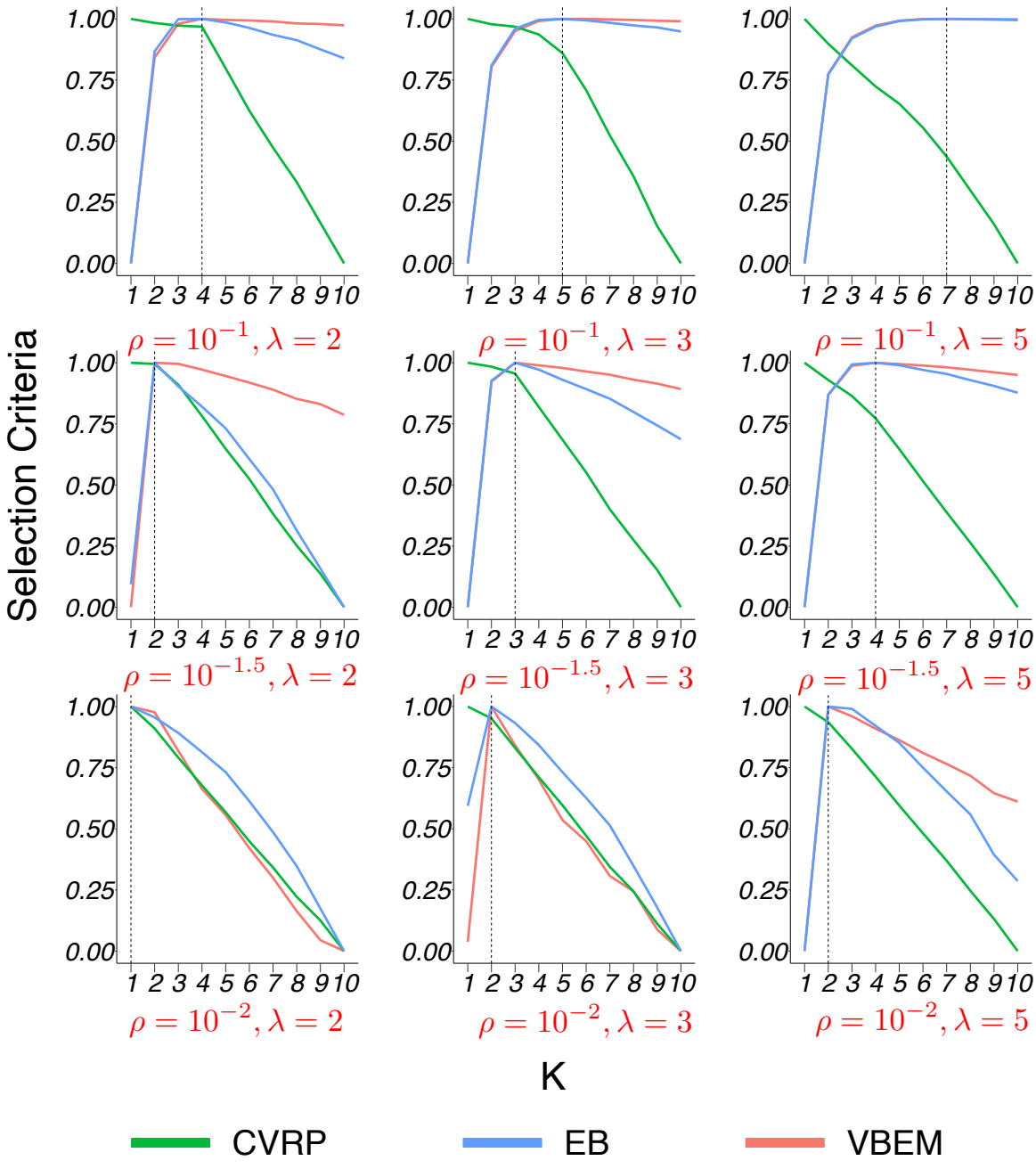


Figure 2.10: Values of model selection criteria in model 4 simulation.

We briefly summarize a few key observations from the simulation studies. It is seen that EB estimates had smaller MSEs than the other two methods in most of the cases above. For the dense SBM (model 1), the accuracy of EB estimate was much higher. The relative low variance in connectivity across different blocks led to higher degree of shrinkage and information sharing among the EB estimates. For the sparse SBM (model 2) and graphon models (model 3 and 4), EB showed moderate improvements over the two competing methods in general. When the graph is sparse, EB can be much more accurate than VBEM, as shown in Figures 2.7 and 2.8. As for model selection, EB generally selected the number of clusters $\hat{K}$ that was closer to K^* and $\tilde{K}$ in all the models above, which demonstrates the usefulness of our hierarchical model for deriving likelihood-based model selection criterion.

2.2.3 Alternative clustering and complexity

Our results and numerical comparisons in section 2.2.1 and section 2.2.2 were conducted to demonstrate the uniform accuracy improvement: By varying the input number of clusters so some cluster results could be very inaccurate, our EB estimates reached smaller MSEs for almost all the clustering results. To further demonstrate this point, we also applied our EB estimates after spectral clustering. As shown in Figure 2.11, our method improved the parameter estimation accuracy as well: The EB/MLE MSE ratio shows a similar pattern as in Figure 2.3 for both (a) dense and (b) sparse SBMs (the same settings as in Figure 2.3, $K^* = 10$).

The computation of our EB method is only the maximization of the likelihood (2.6, 2.7). The objective is the sum of two separate functions. Thus, we just need to maximize two bi-variate functions, regardless of the problem size (n, K). In general, the computation time is negligible compared to the graph clustering step. Table 2.8 reports the average running times (in seconds) of spectral clustering (T_C) and our EB estimation (T_E) by BFGS for various network size n and number of communities K, on a single 2.6 GHz Intel i7 core.

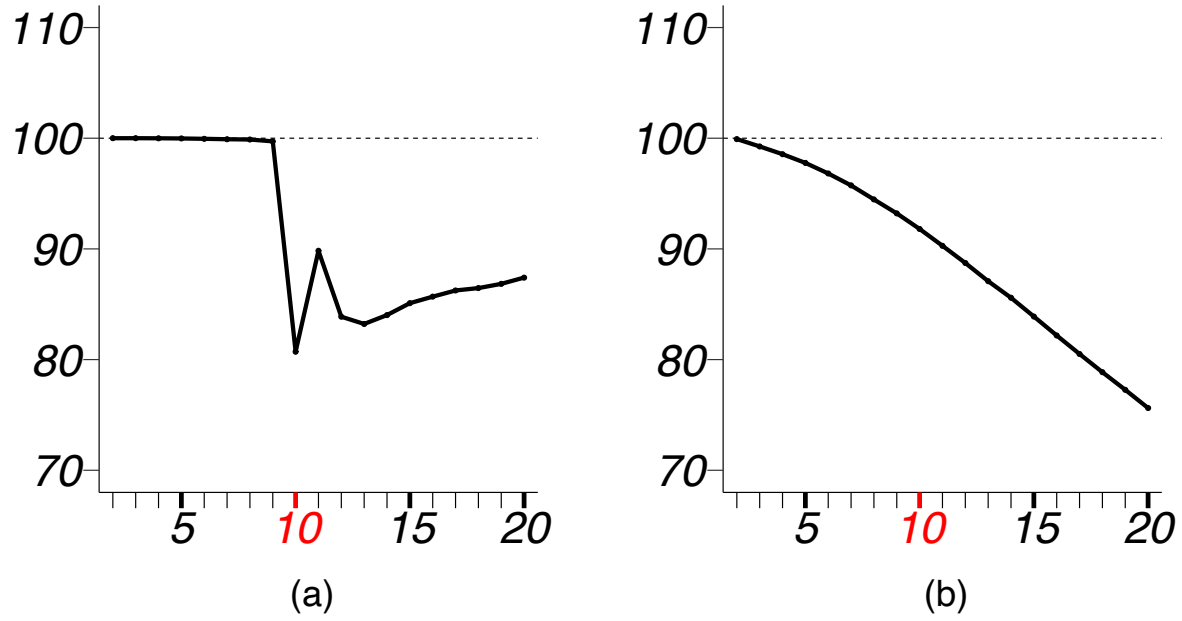


Figure 2.11: MSE ratios in spectral clustering simulation.

Table 2.8: Simulation running time.

(n, K)	(100, 10)	(1000, 10)	(1000, 100)	(5000, 10)	(5000, 100)	(10000, 500)
T_C	0.06	0.7	4.4	6.7	149	2696
T_E	0.08	0.1	0.2	0.6	1.9	11.6

2.3 Real data examples

In this section, we apply our empirical Bayes method on two real-world networks, Emails network data and French blogs data. The main reason we chose these two datasets is their well-annotated memberships. For real-world networks, we do not have the underlying connectivity matrix as the ground truth, which makes it difficult to evaluate estimation accuracy. However, for a network with known node labels that indicate their community memberships (the "ground truth"), the true partition Z_{true} of the vertices is given. Thus, we

will develop accuracy metrics based on Z_{true} to compare different methods. On the real data, besides MSE, we also used cross-validation as an objective evaluation criterion (Figure 2.12 and Figure 2.13).

2.3.1 Email-Eu-core network

The Email-Eu-core network (Eucore) is a directed network generated using email data from a large European institute, consisting of incoming and outgoing communications between members of the institute from 42 departments. Leskovec and Krevl (2014) organized the data and labeled which department each individual node belongs to, i.e. the "ground-truth" community memberships. The network has $n = 1005$ nodes and 25,571 directed edges, which we converted to undirected ones by removing their orientations. We applied VBEM to detect communities with an input number of clusters $K = 30, 31, \ldots, 50$.

Given the known community memberships, we constructed a connectivity matrix $\Theta^* = (\theta^*_{ab})_{K^* \times K^*}$ with entries

$$\theta^*_{ab} = X^B_{ab}/n_{ab}, \qquad a, b \in \{1, \ldots, K^*\}, \tag{2.22}$$

where X^B_{ab} is the number of edges observed in block (a, b), $n_{ab} = |Z^{-1}_{\text{true}}(a)| \cdot |Z^{-1}_{\text{true}}(b)|$ for $a \neq b$ and $n_{aa} = |Z^{-1}_{\text{true}}(a)| \cdot (|Z^{-1}_{\text{true}}(a)| - 1)/2$, and K^* is the true number of communities. Then the MSE (2.16) between an estimate $\widehat{\Theta}(K)$ and Θ^* (2.22) were used as an accuracy metric to compare estimated connectivity matrices, where K is the input number of clusters.

We also used test data likelihood as another comparison metric. We randomly sampled 70% of the nodes, denoted by V_{o}, as observed training data, and estimated a connectivity matrix $\widehat{\Theta} = (\hat{\theta}_{ij})_{K^* \times K^*}$ from their edge connections and true memberships. Denote by V_{t} the test data nodes not used in the estimation. Recall that X_{ij} is the (i, j)th element in the adjacency matrix of the network. Then test data likelihood $\mathcal{L}_{\text{test}}$ was calculated according

to (1.2) given the $\widehat{\Theta}$ estimated by a method,

$$\mathcal{L}_{\text{test}} = \prod_{i \in V_{\text{o}}, j \in V_{\text{t}}} \hat{\theta}_{z_i z_j}^{X_{ij}} (1 - \hat{\theta}_{z_i z_j})^{1 - X_{ij}} \times \prod_{k < j \in V_{\text{t}}} \hat{\theta}_{z_j z_k}^{X_{jk}} (1 - \hat{\theta}_{z_j z_k})^{1 - X_{jk}}, \tag{2.23}$$

where z_i, z_j, z_k are the known labels of the nodes. Note that $X_{ij} \in \{0, 1\}$ is the edge connection between a vertex i in the training data and a vertex j in the test data, while X_{jk} is the edge connection between two vertices j and k in the test data. We repeated this procedure 100 times independently to find the distribution of test data likelihood $\mathcal{L}_{\text{test}}$ across random sample splitting of the n nodes into V_{o} and V_{t}.

The MSE ratios of EB over the other two competing methods were calculated and plotted against K in Figure 2.12(a), which shows the ratio of MSE of EB estimate over that of MLE and VBEM for different values of K. It is clear that EB achieved smaller MSE than the other two methods for all values of K. The MSE ratios ranged from 60% to 90%. When the input number of communities K was close to or greater than $K^* = 42$, the improvement of EB over the competing methods became more substantial. Figure 2.12(b) shows the box-plot of test data log-likelihood values across 100 random sample splitting. From the box-plots, we see that the test data likelihood of EB was significantly higher than the other two estimates. These comparisons confirm that EB estimates were more accurate than the other two competing methods in terms of both metrics.

We further applied the three model selection methods, CVRP, VBEM and EB, on the whole network, and they gave estimates $\hat{K} = 31$, 43 and 37, respectively. The $\hat{K}$ by VBEM and EB were both reasonably close to the ground-truth of $K^* = 42$.

2.3.2 Political blogs

Next we consider the French political blogosphere network from Latouche et al. (2011). The network is made of 196 vertices connected by 2864 edges. It was built from a single day snapshot of political blogs automatically extracted on October 14th, 2006 and manually classified by the "Observatoire Presidentiel" project (Zanghi et al., 2008). In this network,

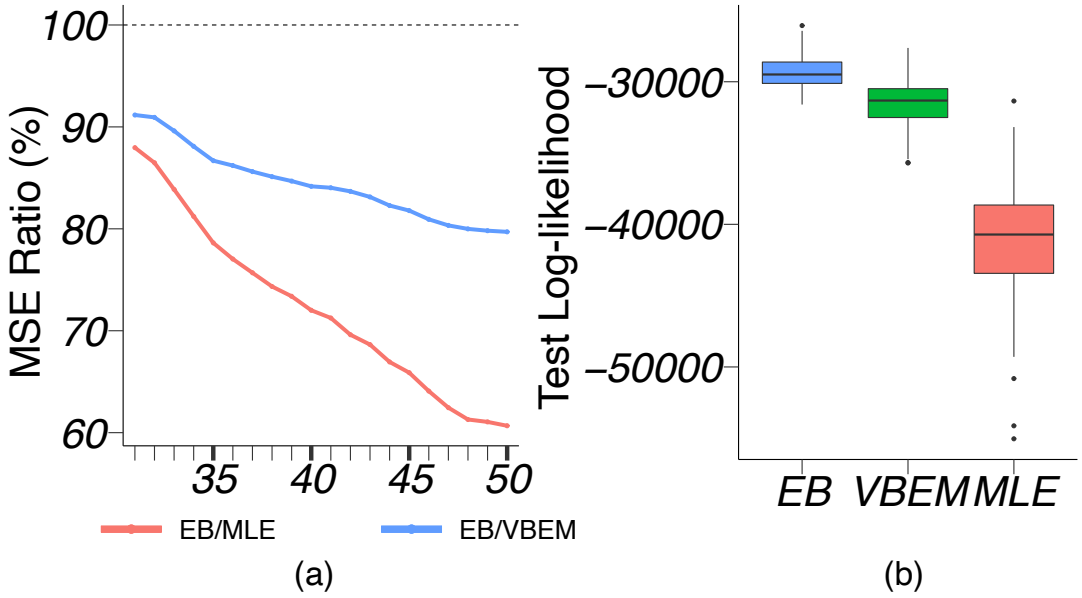


Figure 2.12: Results for Email-Eu-core network analysis.

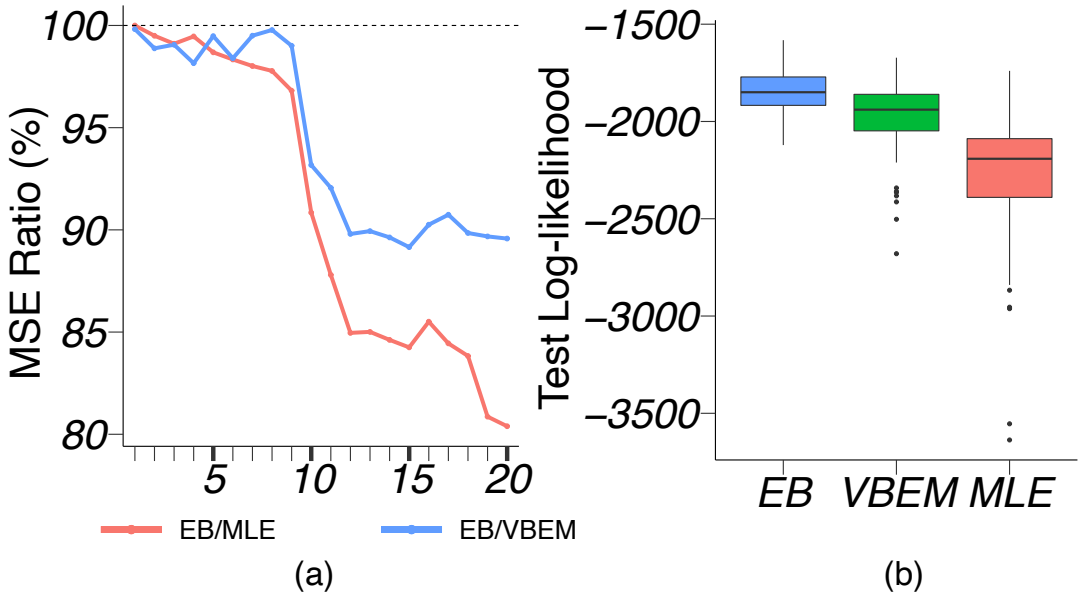


Figure 2.13: Results for French blogsphere network analysis.

nodes correspond to hostnames and there is an edge between two nodes if there is a known hyperlink from one hostname to the other. The four main political parties that are present in the data set are the UMP (french republican), liberal party (supporters of economic-liberalism), UDF (moderate party), and PS (french democrat). However, in the dataset annotated by Latouche et al. (2011) there are $K^* = 11$ different node labels in total, since they considered analysts as well as subgroups of the parties.

We applied the same analyses as in Section 2.3.1 with input $K = 1, \ldots, 20$. The MSE and test data likelihood results are shown in Figure 2.13. When K was close to or greater than $K^* = 11$, EB provided more accurate estimates than both MLE and VBEM with smaller MSEs. Similarly, the box-plots in Figure 2.13(b) demonstrate that the test data log-likelihood calculated with EB estimates was significantly higher than the two competing methods. In terms of model selection, CVRP, VBEM and EB estimated $\hat{K} = 1$, 12 and 10 respectively, while the true $K^* = 11$. Again, the latter two criteria worked quite well on this network.

CHAPTER 3

Causal discovery from multiple populations

This chapter introduces a method for causal discovery from observational data generated from two populations. In Section 3.1, we introduce our intuition behind the project, which is based on a phenomenon that gene regulatory strength differs by their cellular states. In Section 3.2, we build a model to encode this phenomenon and deal with the situation where data were generated under multiple populations that share the same graph structure. We demonstrate and prove the effect of edges with different weights across populations on coefficients of regression among nodes in networks. In Section 3.3, based on the observed pattern of our defined *difference indicator* matrix obtained from node wise regression, we propose an algorithm to discover causal relations and identify differential edges in cpDAGs. Finally we analyze the performance of our algorithm on simulated and real world datasets in Section 3.4.

3.1 Gene regulatory — intuition behind the work

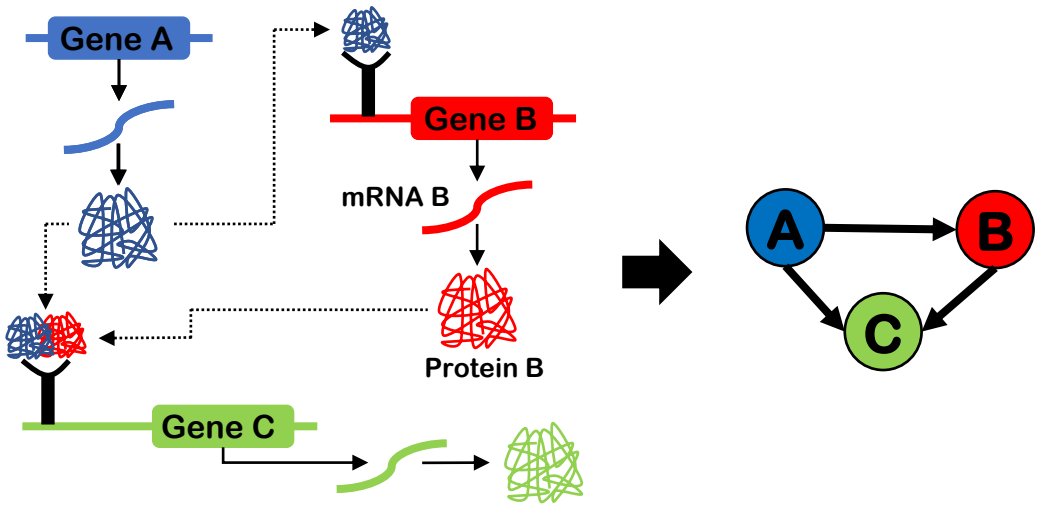
High-throughput gene expression profiling has enabled the study of gene interactions and activities. Different types of techniques, including microarray (Kuwabara, 2003; Passador-Gurgel et al., 2007), RNA-sequencing (Mortazavi et al., 2008; Nagalakshmi et al., 2010; Garber et al., 2011), single cell sequencing (Pennisi, 2012; Wen and Tang, 2018; Tang et al., 2019) have generated an enormous amount of data on genome and transcriptome profiling.

These data have also made the inference of large-scale gene regulatory networks (GRNs)

possible. Many different methods were developed to enhance the understanding of cell growth, division, differentiation and development (EH, 2012; Chu et al., 2016), as well as pathological mechanism (Madhamshettiwar et al., 2012; Emmert-Streib et al., 2014; Hu et al., 2016). A GRN is a network where each vertex represents a type of gene, and regulators of gene expression are connected to target genes by interaction edges. A GRN can be either undirected or directed, and in our case we focus on directed networks, where the regulators are the parents of the edges and target genes are the children. The edges can also bear weights to specify the level of effect of a gene on another. Van den Broeck et al. (2020) has reviewed the recent developments on GRN inference.

Gene functions and regulates other genes in the network via *gene expression*. The gene expression process, summarized by Crick (1958, 1970), is very complicated and has been studied by many researchers for decades. The whole process includes several different phases, starting from transcription and ending with producing gene product (often proteins). Transcription is the process of copying a segment of DNA (contains gene sequence) into RNA. The RNA that can encode protein is called messenger RNA (mRNA). Later in the translation process, mRNA is decoded to produce a specific amino acid chain, which later folds into an active protein. Meanwhile, one of the major components of the transcription process is transcription factors, which are proteins and hence are products of different genes. Consequently, there is an indirect connection between the expression level of genes. Figure 3.1(a) (Huynh-Thu and Sanguinetti, 2019) gives a schematic of a GRN, and shows how the GRN is used to model the mechanism. In the figure, gene A directly regulates gene B, and both genes express proteins that have combinatorial regulation on gene C via complex formation. The abstracted structure of this regulatory system is shown by the directed network in the right part of the figure.

Li (2002) has proposed a liquid association theory about genome-wide coexpression dynamics, i.e. the correlation of two genes may depend on the constantly varying cellular state and other gene expression level. The author analyzed the yeast microarray data and



(a) GRN network schematic and abstract representation.

(b) An edge with different regulatory strength in two cell types.

Figure 3.1: An illustration of gene regulatory network (GRN) and differential edge.

revealed how the correlation between genes are found to change when the expression levels are different in various cell types. Chu et al. (2016) analyzed the transcriptomes of human embryonic stem cell-derived lineage-specific progenitors by single-cell RNA-sequencing (scRNA-seq), and showed that genes may have different regulatory strength during cell development and differentiation. To summarize the phenomenon in the network, as shown in Figure 3.1(b), when two types of cells share the regulatory scheme among some genes, the regulatory strength, represented as edge weight in the graph, varies in different cells or cell types.

This phenomenon of GRN can be generalized to a situation where the same network has various edge weights or functions between the vertices under different circumstances. When the data is known to be generated from different populations, the observed data from different populations can exhibit different patterns, which can be seen as an intervention on the network that leads to alternated conditional distribution of the nodes. Even though the manipulated vertices in the network are unknown from the observational data, it is still possible to discover the causal structure from statistical analysis on the difference of their relationship in different populations. In the following sections, we construct a statistical model to deal with this situation and propose a method to discover causal relationships from observational data.

3.2 Differential edges and difference indicator matrix

We assume that data $\mathbf{Z} \in \mathbb{R}_{n\times p}$ are independently generated from different known populations that sharing the same network structure G^*. In the dissertation paper we use two populations to demonstrate the method, thus the data can be separated as $\mathbf{Z}' \in \mathbb{R}_{n'\times p}$ and $\mathbf{Z}'' \in \mathbb{R}_{n''\times p}$, where $n' + n'' = n$. These assumptions are summarized in Assumption 3.1, in which $G' = G''$ means that two DAGs are defined on the same set of nodes and have the same graph structure (same unweighted adjacency matrix). The weighted adjacency matrix

$\mathbf{A}' = (a'_{ij})_{i,j=\{1,\dots,p\}}$ and $\mathbf{A}'' = (a''_{ij})_{i,j=\{1,\dots,p\}}$ have the same zero elements ($\forall\, i, j \in \{1, \dots, p\}$, $a'_{ij} = 0 \Leftrightarrow a''_{ij} = 0$), and different values of non-zero elements ($\exists\ i, j \in \{1, \dots, p\}$ s.t. $a'_{ij} \neq a''_{ij}$). The error variance matrices $\mathbf{\Omega}' = \text{diag}(\omega'^2_j)_{j=1,\dots,p}$ and $\mathbf{\Omega}'' = \text{diag}(\omega''^2_j)_{j=1,\dots,p}$ have $\mathbf{\Omega}' = \alpha\mathbf{\Omega}''$, where α is a constant scaling factor, e.g. $\alpha = 1$, then each variable X_j in the graph has the same variance of error in two populations $\omega'^2_j = \omega''^2_j$. Together, since $\mathbf{\Sigma} = (\mathbf{I} - \mathbf{A})^{-T}\mathbf{\Omega}(\mathbf{I} - \mathbf{A})^{-1}$, we have covariance matrices of two populations $\mathbf{\Sigma}' \neq \mathbf{\Sigma}''$ that leads to non-linearity in the model.

Assumption 3.1 *The samples in the observed data* $\mathbf{Z}$ *are independently generated from two populations* $\mathbf{Z} = \begin{pmatrix} \mathbf{Z}' \\ \mathbf{Z}'' \end{pmatrix}$. *And data* $\mathbf{Z}'$ *and* $\mathbf{Z}''$ *are generated from a Gaussian DAG* G' (1.21) *with parameters* $(\mathbf{A}', \mathbf{\Omega}')$ *and* G'' *with* $(\mathbf{A}'', \mathbf{\Omega}'')$, *respectively, where* $G' = G''$, $\mathbf{A}' \neq \mathbf{A}''$ *and* $\mathbf{\Omega}' = \alpha\mathbf{\Omega}''$.

Assume the existence of *differential* edges (Definition 3.1), i.e. some edges have different weights in different populations, which is a common data generation process in real world networks such as gene regulatory network. Given the observed data $\mathbf{Z}$, many structural learning algorithms in Section 1.4 can estimate the Markov equivalence class of the graph. In the following part of the section, we show that *differential* edges can be simply captured by regression among the nodes and thus distinguish DAGs in their Markov equivalence classes.

Definition 3.1 (Differential edge) *In a DAG* G, *denote the weights of edge* $V_i \to V_j$ *as* a'_{ij} *and* a''_{ij} ($a'_{ij} \neq 0$, $a''_{ij} \neq 0$) *in two populations respectively. If* $a'_{ij} \neq a''_{ij}$, *then the edge* $V_i \to V_j$ *is a differential edge, otherwise it is a non-differential edge.*

We first analyze the case in one population, denote $\mathbf{X} = (X_1, \dots, X_p)$ as the generated data, according to Assumption 3.1, $\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$. If we have

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_{ii} & \mathbf{\Sigma}^T_{-i,i} \\ \mathbf{\Sigma}_{-i,i} & \mathbf{\Sigma}_{-i,-i} \end{pmatrix}, \tag{3.1}$$

then the matrix inverse $\mathbf{\Theta} = (\theta_{ij})_{i,j=\{1,\dots,p\}}$ is

$$\mathbf{\Theta} = \begin{pmatrix} \theta_{ii} & -\theta_{ii}\mathbf{\Sigma}^T_{-i,i}\mathbf{\Sigma}^{-1}_{-i,-i} \\ -\theta_{ii}\mathbf{\Sigma}^{-1}_{-i,-i}\mathbf{\Sigma}^T_{-i,i} & \mathbf{\Sigma}^{-1}_{-i,-i}(\mathbf{I}+\theta_{ii})\mathbf{\Sigma}_{-i,i}\mathbf{\Sigma}^T_{-i,i}\mathbf{\Sigma}^{-1}_{-i,-i} \end{pmatrix} = \begin{pmatrix} \theta_{ii} & \mathbf{\Theta}^T_{-i,i} \\ \mathbf{\Theta}_{-i,i} & \mathbf{\Theta}_{-i,-i} \end{pmatrix}, \tag{3.2}$$

where $\mathbf{\Sigma}_{-i,j}$ is a vector obtained by selecting the j-th column of the sub matrix that removes the i-th row from matrix $\mathbf{\Sigma}$, and $\mathbf{\Sigma}_{-i,-j}$ is the sub matrix of $\mathbf{\Sigma}$ by removing the i-th row and j-th column.

Given a Gaussian distribution, the conditional distribution of a single node X_i given the rest of the nodes can be written as

$$X_i|\mathbf{X}_{-i} \sim \mathcal{N}(\mu_i + \mathbf{\Sigma}_{i,-i}\mathbf{\Sigma}^{-1}_{-i,-i}(\mathbf{X}_{-i} - \mu_{\mathbf{X}_{-i}}), \mathbf{\Sigma}_{i,i} - \mathbf{\Sigma}_{i,-i}\mathbf{\Sigma}^{-1}_{-i,-i}\mathbf{\Sigma}_{-i,i}). \tag{3.3}$$

Without loss of generality, let $\mu = 0$ and $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$ denote the precision matrix, then

$$\begin{aligned} X_i|\mathbf{X}_{-i} &\sim \mathcal{N}(\mathbf{\Sigma}_{i,-i}\mathbf{\Sigma}^{-1}_{-i,-i}\mathbf{X}_{-i}, \mathbf{\Sigma}_{i,i} - \mathbf{\Sigma}_{i,-i}\mathbf{\Sigma}^{-1}_{-i,-i}\mathbf{\Sigma}_{-i,i}), \\ &\sim \mathcal{N}(\mathbf{\Sigma}^T_{-i,i}\mathbf{\Sigma}^{-1}_{-i,-i}\mathbf{X}_{-i}, \mathbf{\Theta}^{-1}_{i,i}), \\ &\sim \mathcal{N}(-\frac{1}{\theta_{ii}}\mathbf{\Theta}^T_{-i,i}\mathbf{X}_{-i}, \frac{1}{\theta_{ii}}). \end{aligned} \tag{3.4}$$

From the conditional distribution (3.4), denote $\mathbf{b}_{X_i \sim \mathbf{X}_{-i}}$ as the coefficients of the regression of X_i on $\mathbf{X}_{-i}$, we have

$$\mathbf{b}_{X_i \sim \mathbf{X}_{-i}} = -\frac{1}{\theta_{ii}}\mathbf{\Theta}^T_{-i,i}. \tag{3.5}$$

When the sample size goes to infinity, the estimated coefficients of regressing X_i on other variables converge to values determined by the precision matrix $\mathbf{\Theta}$ as in (3.5). The value can be also written by the weights of the graph. Denote the weighted adjacency matrix as $\mathbf{A} = (a_{ij})_{p\times p}$, as no edge exists from one node to itself, $a_{ii} = 0$ for $i = 1, \cdots, p$. Let $\tilde{\mathbf{A}} = \mathbf{I} - \mathbf{A} = (\tilde{a}_{ij})_{p\times p}$, then

$$\tilde{a}_{ij} = \begin{cases} -a_{ij} & \text{if } i \neq j, \\ 1 & \text{otherwise.} \end{cases} \tag{3.6}$$

Then the precision matrix can be written as

$$\begin{aligned}\boldsymbol{\Theta} &= (\mathbf{I}-\mathbf{A})\boldsymbol{\Omega}^{-1}(\mathbf{I}-\mathbf{A})^T = \tilde{\mathbf{A}}\boldsymbol{\Omega}^{-1}\tilde{\mathbf{A}}^T \\ &= \begin{pmatrix} \frac{1}{\omega_1^2}\tilde{a}_{11} & \frac{1}{\omega_2^2}\tilde{a}_{12} & \cdots & \frac{1}{\omega_j^2}\tilde{a}_{1j} & \cdots & \frac{1}{\omega_p^2}\tilde{a}_{1p} \\ \frac{1}{\omega_1^2}\tilde{a}_{21} & \frac{1}{\omega_2^2}\tilde{a}_{22} & \cdots & \frac{1}{\omega_j^2}\tilde{a}_{2j} & \cdots & \frac{1}{\omega_p^2}\tilde{a}_{2p} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{1}{\omega_1^2}\tilde{a}_{i1} & \frac{1}{\omega_2^2}\tilde{a}_{i2} & \cdots & \frac{1}{\omega_j^2}\tilde{a}_{ij} & \cdots & \frac{1}{\omega_p^2}\tilde{a}_{ip} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{1}{\omega_1^2}\tilde{a}_{p1} & \frac{1}{\omega_2^2}\tilde{a}_{p2} & \cdots & \frac{1}{\omega_j^2}\tilde{a}_{pj} & \cdots & \frac{1}{\omega_p^2}\tilde{a}_{pp} \end{pmatrix} \begin{pmatrix} \tilde{a}_{11} & \tilde{a}_{21} & \cdots & \tilde{a}_{j1} & \cdots & \tilde{a}_{p1} \\ \tilde{a}_{12} & \tilde{a}_{22} & \cdots & \tilde{a}_{j2} & \cdots & \tilde{a}_{p2} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{a}_{1i} & \tilde{a}_{2i} & \cdots & \tilde{a}_{ji} & \cdots & \tilde{a}_{pi} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{a}_{1p} & \tilde{a}_{2p} & \cdots & \tilde{a}_{jp} & \cdots & \tilde{a}_{pp} \end{pmatrix}, \end{aligned} \tag{3.7}$$

where $\mathbf{I}$ is a $p \times p$ identity matrix, and $\boldsymbol{\Omega} = \text{diag}(\omega_1^2, \cdots, \omega_p^2)$ is the variance of the error. Then the element of the i-th and j-th column of $\boldsymbol{\Theta}$ can be presented by the elements of the adjacency matrix,

$$\theta_{ij} = \sum_{k=1}^{p} \frac{1}{\omega_k^2}\tilde{a}_{ik}\tilde{a}_{jk} = -\frac{1}{\omega_j^2}a_{ij} - \frac{1}{\omega_i^2}a_{ji} + \sum_{k\neq i,j}^{p} \frac{1}{\omega_k^2}a_{ik}a_{jk}. \tag{3.8}$$

As $\mathbf{b}_{X_i \sim \mathbf{X}_{-i}} \in \mathbb{R}^{p-1}$, suppose $\mathbf{b}_{X_i \sim \mathbf{X}_{-i}} = (b_{X_k|X_i \sim \mathbf{X}_{-i}})_{k=\{1,\cdots,p\}\setminus i}$, where $b_{Y_0|X\sim\mathbf{Y}}$ represents the coefficient of variable Y_0 in the regression of X on $\mathbf{Y}$, then we have

$$\mathbf{b}_{X_i \sim \mathbf{X}_{-i}} = (b_{X_k|X_i \sim \mathbf{X}_{-i}})_{k=\{1,\cdots,p\}\setminus i} = -\frac{1}{\theta_{ii}}\boldsymbol{\Theta}_{-i,i}^T = -\frac{1}{\theta_{ii}}(\theta_{ik})_{k=\{1,\cdots,p\}\setminus i}. \tag{3.9}$$

In a DAG G, suppose we regress X_m on all other nodes except X_m, denoted as $\mathbf{X}_{-m}$, then for any node $X_t \in \mathbf{X}_{-m}$, the coefficient of X_t is

$$b_{X_t|X_m\sim\mathbf{X}_{-m}} = -\frac{\theta_{mt}}{\theta_{mm}} = -\frac{-\frac{1}{\omega_m^2}a_{tm} - \frac{1}{\omega_t^2}a_{mt} + \sum_{k\neq m,t}^{p}\frac{1}{\omega_k^2}a_{mk}a_{tk}}{\frac{1}{\omega_m^2} + \sum_{k\neq m}^{p}\frac{1}{\omega_k^2}a_{mk}^2}, \tag{3.10}$$

thus the coefficient $b_{X_t|X_m\sim\mathbf{X}_{-m}}$ can be interpreted as a function of the weighted adjacency matrix $\mathbf{A}$.

We analyze the behavior of linear regression between X_m and $\mathbf{X}_{-m}$, assuming two populations have *differential* edges between V_m and some of its neighbors. Write the weighted adjacency matrix in two populations as $\mathbf{A}' = (a'_{ij})$ and $\mathbf{A}'' = (a''_{ij})$ respectively. Denote the

set of variables $X_k \in pa(X_m)$ that have $a'_{km} \neq a''_{km}$ as $pa^*(X_m)$. Similarly denote the set of variables $X_k \in ch(X_m)$ that have $a'_{mk} \neq a''_{mk}$ as $ch^*(X_m)$. Finally denote all parents of the node variables in $ch^*(X_m)$ except X_m as $sp^*(X_m)$, i.e. $sp^*(X_m) := \bigcup_{X_k \in ch^*(X_m)} pa(X_k) \backslash X_m$. Removing all the zero terms in the weighted adjacency matrix in (3.10), we have

$$b_{X_t|X_m \sim \mathbf{X}_{-m}} = -\frac{-\frac{1}{\omega_m^2} a_{tm} - \frac{1}{\omega_t^2} a_{mt} + \sum_{k:X_k \in ch(X_m) \cap ch(X_t)} \frac{1}{\omega_k^2} a_{mk} a_{tk}}{\frac{1}{\omega_m^2} + \sum_{k:X_k \in ch(X_m)} \frac{1}{\omega_k^2} a_{mk}^2}. \tag{3.11}$$

If a random variable X_t is not in the Markov blanket (Section 1.3) of X_m, i.e. $X_t \notin mb(X_m)$, then V_t has no direct edge connected to node V_m, thus $a_{tm} = a_{mt} = 0$. Secondly it has no common children with node V_m, which means for any variable $X_k \in \mathbf{X}_{-m}$, $a_{mk} \cdot a_{tk} = 0$. By plugging them into (3.10), we have $b_{X_t|X_m \sim \mathbf{X}_{-m}} = 0$ for $\forall X_t \notin mb(X_m)$, which shows that the regression of X_m on $\mathbf{X}_{-m}$ is essentially the same as regressing X_m on $mb(X_m)$. Thus for $X_t \in mb(X_m)$, (3.10) can be re-written as

$$b_{X_t|X_m \sim mb(X_m)} = -\frac{-\frac{1}{\omega_m^2} a_{tm} - \frac{1}{\omega_t^2} a_{mt} + \sum_{k:X_k \in ch(X_m) \cap ch(X_t)} \frac{1}{\omega_k^2} a_{mk} a_{tk}}{\frac{1}{\omega_m^2} + \sum_{k:X_k \in ch(X_m)} \frac{1}{\omega_k^2} a_{mk}^2}. \tag{3.12}$$

Definition 3.2 (Difference indicator matrix (DIM)) *Write the node wise regression coefficients in two populations as* $\mathbf{b}'$ *and* $\mathbf{b}''$*, in which* $b_{X_t|X_m \sim \mathbf{X}_{-m}}$ *represents the coefficient of* X_t *when regressing* X_m *on the rest of the variables in* $\mathbf{X}$*. Difference indicator matrix* $\mathbf{M} = (m_{ij})_{i,j=\{1,\cdots,p\}}$ *is defined by*

$$m_{ij} = \begin{cases} 1, & \text{if } b'_{X_j|X_i \sim \mathbf{X}_{-i}} \neq b''_{X_j|X_i \sim \mathbf{X}_{-i}}, \\ 0, & \text{if } b'_{X_j|X_i \sim \mathbf{X}_{-i}} = b''_{X_j|X_i \sim \mathbf{X}_{-i}}. \end{cases}$$

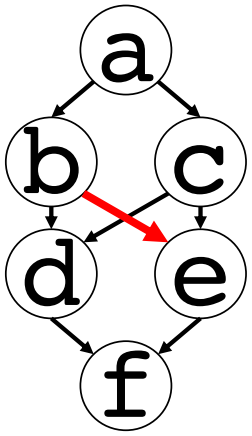
Under Assumption 3.1, Section 3.5.1 proves the following effects of *differential* edges on the difference indicator matrix $\mathbf{M} = (m_{ij})_{i,j=\{1,\ldots,p\}}$. Suppose edge $V_i \to V_j$ is the only *differential* edge connected to node V_i, given population level data, i.e. the parameters of the SEM $(\mathbf{A}', \mathbf{\Omega}')$ and $(\mathbf{A}'', \mathbf{\Omega}'')$ are known, we have

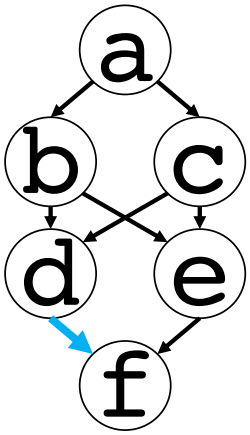
	a	b	c	d	e	f
a	0	0.5	-1	0	0	0
b	0	0	0	1	α_1	0
c	0	0	0	0.8	-2	0
d	0	0	0	0	0	β_1
e	0	1	0	0	0	-1
f	0	0	0	0	0	0

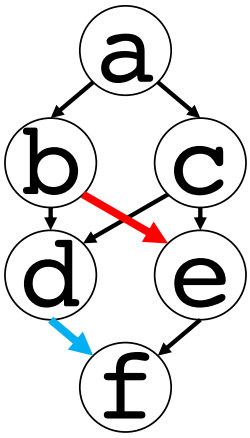
A′

	a	b	c	d	e	f
a	0	0.5	-1	0	0	0
b	0	0	0	1	α_2	0
c	0	0	0	0.8	-2	0
d	0	0	0	0	0	β_2
e	0	1	0	0	0	-1
f	0	0	0	0	0	0

A′′







	a	b	c	d	e	f
a	–	0	0	0	0	0
b	1	–	1	1	1	0
c	0	1	–	0	0	0
d	0	0	0	–	0	0
e	0	1	0	0	–	0
f	0	0	0	0	0	–

(a) $\alpha_1 \neq \alpha_2; \beta_1 = \beta_2$

	a	b	c	d	e	f
a	–	0	0	0	0	0
b	0	–	0	0	0	0
c	0	0	–	0	0	0
d	0	1	1	–	1	1
e	0	0	0	1	–	0
f	0	0	0	1	0	–

(b) $\alpha_1 \neq \alpha_2; \beta_1 = \beta_2$

	a	b	c	d	e	f
a	–	0	0	0	0	0
b	1	–	1	1	1	0
c	0	1	–	0	0	0
d	0	1	1	–	1	1
e	0	1	0	1	–	0
f	0	0	0	1	0	–

(c) $\alpha_1 \neq \alpha_2; \beta_1 \neq \beta_2$

Figure 3.2: A demonstration of the relationship between differential edge and DIM.

- $m_{ij} = m_{ji} = 1$,
- $\forall X_k \in mb(X_i) \backslash X_j$, $m_{ik} = 1$,
- $\forall X_t \in pa(X_j) \backslash X_i$, $m_{ti} = 1$,
- $\forall X_q \in mb(X_i) \backslash \{X_j, pa(X_j)\}, m_{qi} = m_{qj} = 0$.

If multiple *differential* edges exist, assuming that they do not cancel each other out, i.e. the effect of each *differential* edge can be observed in DIM, DIM is the logical disjunction of the separate DIMs obtained from assuming that merely a single *differential* edge exists.

Fig 3.2 gives an illustration of the relationship. In the figure the weighted adjacency matrices in two populations are $\mathbf{A}'$ and $\mathbf{A}''$ respectively, three different cases are demonstrated: (a) Assume only the red blocks differ in value in two populations, i.e. $\alpha_1 \neq \alpha_2$, $\beta_1 = \beta_2$, the difference of the node wise regression coefficients is shown below as a binary DIM. We can see that X_b is the parent of the *differential* edge, it has different regression coefficients to all other nodes in its Markov blanket. The child of the *differential* edge X_e has a different coefficient to X_b. X_c as the other parent of X_e also has a different coefficient to X_b. The rest of the coefficients are the same in two populations. (b) Assume only the blue blocks differ in two populations, i.e. $\alpha_1 = \alpha_2$, $\beta_1 \neq \beta_2$, then X_d has different coefficients towards its Markov blanket, and X_e, X_f are the variables with different coefficients on X_d. (c) Assume both the blue and red blocks are different, i.e. $\alpha_1 \neq \alpha_2$ and $\beta_1 \neq \beta_2$, the result is the combination of the above two cases, where the matrix can be seen as an element wise logical disjunction of the results of (a) and (b).

3.3 Edge orientation by difference indicator matrix

In Section 1.4, we show that under linear Gaussian DAG assumption, the true DAG is not identifiable from its Markov equivalence class from observational data. Whereas Assumption 3.1 introduces non-linearity to our model, and we can utilize the non-linearity incurred

from two populations to discover causal structure from cpDAGs. Based on the observation in Section 3.2, we propose an algorithm to orient edges in cpDAGs. We start with directing edges between single pair of vertices, then go through all the undirected edges and extend the orientation to the whole graph.

3.3.1 Single edge orientation

Under Assumption 3.1, the effect of the *differential* edges on the difference indicator matrix $\mathbf{M}$ has been explained in section 3.2. We can use the pattern of $\mathbf{M}$ and the graph structure G to determine edge direction. For a node V_i in a pDAG G, define set $C_i := \{X_k : m_{ki} = 1\}$, set of potential children $P_i := \{X_k : u_{ik} = 1\}$ where $U = (u_{ij})_{p\times p}$ is the unweighted (binary) adjacency matrix of G. Denote $mb(X_i)$ as the Markov blanket of X_i in G.

Algorithm 1: Single edge orientation

input : The target pDAG G, DIM $\mathbf{M}$, the target undirected edge $V_i - V_j$ in G.

output: Direction of edge $V_i - V_j$.

Initialize $f = 0$;

if $X_j \in C_i$ *and* $P_i \bigcap P_j \bigcap C_i = \emptyset$ *and* $\Sigma_{k:X_k\in mb(X_j)}(1 - m_{jk}) > 0$ **then**

  set $f = f + 1$;

if $X_i \in C_j$ *and* $P_i \bigcap P_j \bigcap C_j = \emptyset$ *and* $\Sigma_{k:X_k\in mb(X_i)}(1 - m_{ik}) > 0$ **then**

  set $f = f - 1$;

if $f > 0$ **then**

  return $V_i \to V_j$;

else if $f < 0$ **then**

  return $V_i \leftarrow V_j$;

else

  return $V_i - V_j$ (edge cannot be oriented).

end

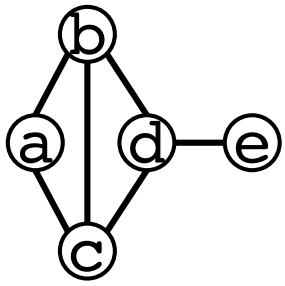


Figure 3.3: A cpDAG to be oriented.

Algorithm 1 can orient single pair of nodes and their neighbors, as well as provide information about the *differential* edges. We use the following example to illustrate how this algorithm works. Suppose we have a cpDAG as shown in Figure 3.3, which represents an equivalence class in Figure 3.4. Assume the true DAG is Figure 3.4 (III) and the only *differential* edge is $V_c \to V_a$. When the sample size goes to infinity, based on Section 3.2 the observed matrix should be

$$\mathbf{M} = \begin{array}{c} \begin{matrix} a & b & c & d & e \end{matrix} \\ \begin{pmatrix} \cdot & 0 & 1 & 0 & 0 \\ 0 & \cdot & 0 & 0 & 0 \\ 1 & 1 & \cdot & 1 & 0 \\ 0 & 0 & 0 & \cdot & 0 \\ 0 & 0 & 0 & 0 & \cdot \end{pmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix}.$$

With DIM $\mathbf{M}$ and the cpDAG G provided, we want to orient the edge V_a — V_c in G. Defined by the algorithm, $C_a = \{X_c\}$, and the potential children set of X_a is $P_a = \{X_b, X_c\}$. Similarly, $C_c = \{X_a\}$, $P_c = \{X_a, X_b, X_d\}$. Then we examine the Markov blanket of the two nodes. The Markov blanket consists of the neighboring nodes and the spouses, and in this example $mb(X_a) = \{X_b, X_c\}$, $mb(X_c) = \{X_a, X_b, X_d\}$, thus we have $\sum_{X_k \in mb(X_a)}(1 - m_{ak}) = \sum_{k \in \{b,c\}}(1 - m_{ak}) = 1$ and $\sum_{X_k \in mb(X_c)}(1 - m_{ck}) = \sum_{k \in \{a,b,d\}}(1 - m_{ck}) = 0$. These conditions guarantee that X_a is not the parent of the *differential* edge, since otherwise $\forall k : X_k \in mb(X_a)$, $m_{ak} = 1$, and thus $\sum_{X_k \in mb(X_a)}(1 - m_{ak}) = 0$. We can see that $X_c \in C_a$,

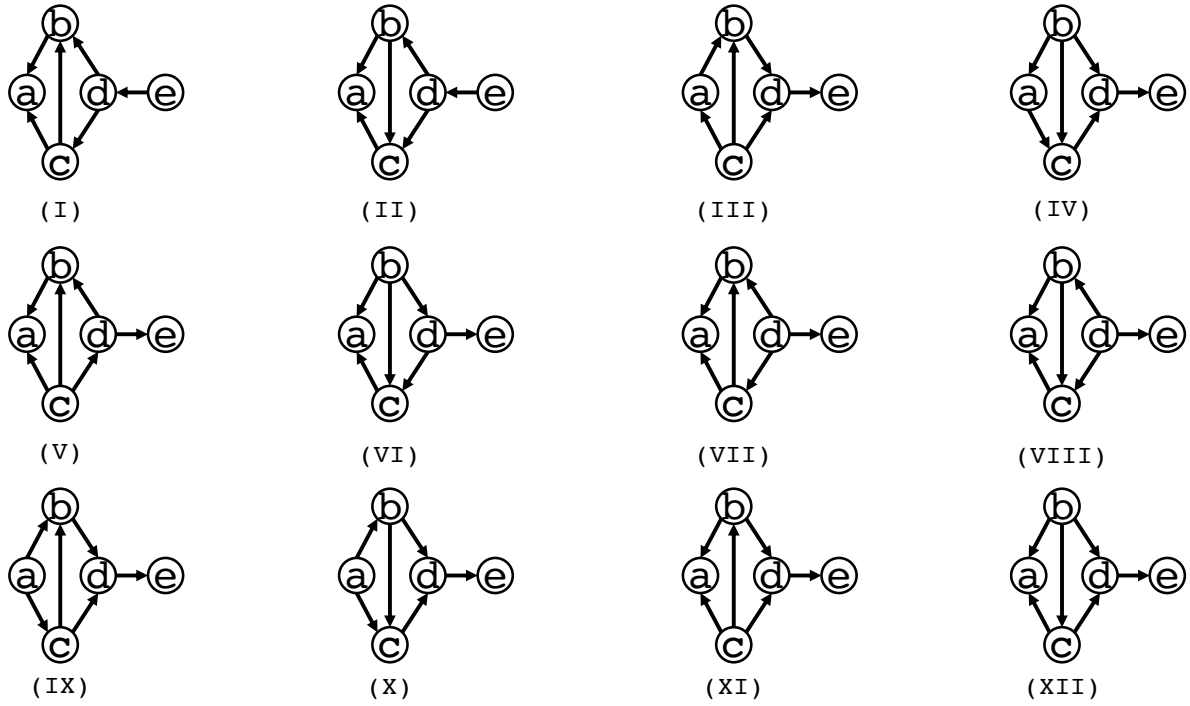


Figure 3.4: Markov equivalence class of the cpDAG in Figure 3.3.

$P_a \cap P_c \cap C_a = \emptyset$ and $\sum_{X_k \in mb(X_a)}(1 - m_{ak}) > 0$. With all the criteria satisfied, the algorithm should return the edge orientation $V_c \to V_a$.

Theorem 3.1 (Correctness of orientation) *Under Assumption 3.1, if the input difference indicator matrix* $\mathbf{M}$ *is correct and pDAG* G *are consistent with the true DAG, then Algorithm 1 guarantees the correctness of the orientation.*

The correctness of the single edge orientation algorithm is summarized in Theorem 3.1. If the observed data satisfies Assumption 3.1, according to Section 3.2 we can obtain a difference indicator matrix $\mathbf{M}$. When the input matrix $\mathbf{M}$ correctly indicates differences between the population regression coefficients in all node wise regressions, we say $\mathbf{M}$ is correct. And we say the partial DAG G is consistent with the true DAG G^* if the edges in G^* are presented in G as either undirected edges or directed edges with correct directions. For example, G can be the skeleton or the cpDAG of G^*. Giving the conditions above, Algorithm 1 outputs

zero false case. The theorem can be proved by contradiction in Section 3.5.2.

Theorem 3.2 (Differential edges) *Under the same assumptions of Theorem 3.1, if a single edge V_a — V_b is oriented as $V_a \to V_b$ by Algorithm 1, then $V_a \to V_b$ is a differential edge, all other edges connected to V_b are non-differential, all edges between V_a and the common children of V_a and V_b are non-differential.*

Theorem 3.3 (Neighboring nodes orientation) *Under the same assumptions of Theorem 3.1, if a single edge V_a — V_b is oriented as $V_a \to V_b$ by Algorithm 1, then V_b is the parent of its other neighboring nodes.*

In fact, besides the orientation of the target edge, the algorithm provides further information about the underlying *differential* edges (Theorem 3.2) as well as the direction of the neighboring edges (Theorem 3.3). The proofs are provided in Section 3.5.2.

3.3.2 Graph orientation

In Section 3.3.1 we justified the correctness of the single edge orientation algorithm. Algorithm 2 (DIMEO) extends the orientation results from Algorithm 1 to the whole input cpDAG. The algorithm iteratively applies our single edge orientation algorithm and extends orientation using the output additional information as well as the Meek's rule, which was found by Meek (1995) to complete orientation based on the graph pattern. Figure 3.5 demonstrates the Meek orientation rule. When an edge is oriented by Algorithm 1, if one of the four patterns in figure 3.5 is formed, the graph can be further oriented. Katz et al. (2019) has summarized the Meek orientation rule as Lemma 3.1 and Lemma 3.2. The algorithm uses a loop to iteratively orient undirected edges. Once an edge is oriented, the additional information in Theorem 3.2 and Theorem 3.3 as well as Meek's rule are applied for extension, and the algorithm stops until no new orientation can be introduced after a traversal of all the undirected edges.

Algorithm 2: Graph edge orientation by DIM (DIMEO)

input : The target cpDAG G, DIM $\mathbf{M}$.

output: An oriented graph G.

Identify undirected edges $V_{a_1} - V_{b_1}, \cdots, V_{a_k} - V_{b_k}$ in G;

Initialize index $i = 1$;

while $i \leq k$ **do**

  if $V_{a_i} - V_{b_i}$ *is not oriented* **then**

    input G, $\mathbf{M}$, $V_{a_i} - V_{b_i}$ into Algorithm 1;

    if *Algorithm 1 outputs* $V_{a_i} \to V_{b_i}$ *or* $V_{a_i} \leftarrow V_{b_i}$ **then**

      update G by adding the orientation of $V_{a_i} - V_{b_i}$ and apply Meek's rule;

      set $i = 1$;

    else

      set $i = i + 1$;

    end

  else

    set $i = i + 1$;

  end

end

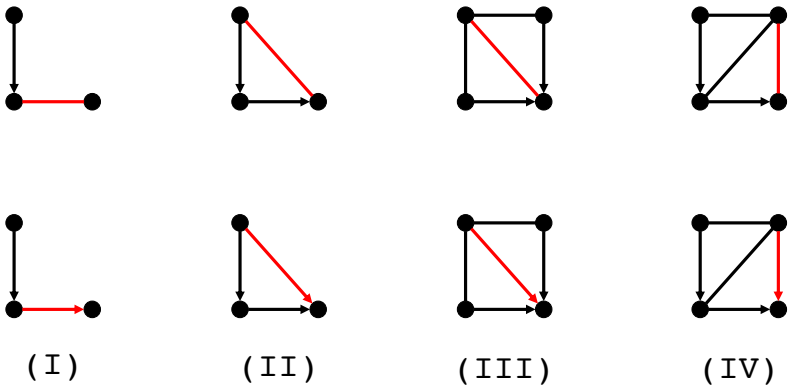


Figure 3.5: Motifs in Meek orientation rules.

Lemma 3.1 *If a node V is involved in any of the four Meek rules (pattern shown in Figure 3.5), and does not have an outgoing edge in the original graph G, then the oriented edge (in the four motifs on the bottom of figure 3.5) is an incoming edge to node V.*

Lemma 3.2 *If a node V is involved in any of the four Meek rules (pattern shown in Figure 3.5), then either V has an outgoing edge or an adjacent undirected edge (in the four motifs on the top of figure 3.5).*

Theorem 3.4 *Under Assumption 3.1, if the input difference indicator matrix* $\mathbf{M}$ *is correct and G is the true cpDAG, Algorithm 2 guarantees correctness of the oriented edges.*

The correctness of DIMEO algorithm is stated in Theorem 3.4. As DIMEO is essentially an assembly of single edge orientations and extension based on single edge directions, correctness of the algorithm is justified by correctness of the single edge orientation in Theorem 3.1. When the true distribution of the population data is given, matrix $\mathbf{M}$ can be easily calculated and is guaranteed to be correct. But when the method is applied on observed data with a finite sample size, hypothesis tests need to be conducted to generate $\mathbf{M}$ from regression coefficients in two populations. For a node V_m, when variable X_m is regressed on all other variables $\mathbf{X}_{-m}$, denote the coefficients and their standard errors as $(\mathbf{b}'_{X_m\sim\mathbf{X}_{-m}}, \mathbf{s}'_{X_m\sim\mathbf{X}_{-m}})$ and $(\mathbf{b}''_{X_m\sim\mathbf{X}_{-m}}, \mathbf{s}''_{X_m\sim\mathbf{X}_{-m}})$ respectively. Element wisely when comparing the coefficients of a single node X_k on X_m, the p-value of the difference is approximated by

$$p = 2\Phi\left(-\frac{|b'_{X_k|X_m\sim\mathbf{X}_{-m}} - b''_{X_k|X_m\sim\mathbf{X}_{-m}}|}{\sqrt{s'^2_{X_k|X_m\sim\mathbf{X}_{-m}} + s''^2_{X_k|X_m\sim\mathbf{X}_{-m}}}}\right), \tag{3.13}$$

where $\Phi(\cdot)$ is the standard Gaussian cumulative distribution function, assuming the sample sizes are large. The matrix of element wise p-values can be transformed into the binary DIM by thresholding. For a pre-determined threshold t, when the p-value is smaller than t, the corresponding element is 1, otherwise 0.

Paternoster et al. (1998) proposed (3.13) to test the equality of regression coefficients, the test is essentially a simple two sample z-test, thus when the sample size goes to infinity,

the type I and type II error converge to zero almost surely (Clogg et al., 1995), and the significance level t can be chosen to be close to zero on large samples, which guarantees the correctness of DIM $\mathbf{M}$. Based on Theorem 3.4, given that the input cpDAG G is correct and $\mathbf{M}$ is obtained from data $\mathbf{X}$ that satisfies Assumption 3.1, the number of mis-oriented edges by DIMEO will be zero as sample size goes to infinity.

Before applying DIMEO, a correct cpDAG is required to theoretically guarantee 100% accuracy of oriented edges. When applying DIMEO on real world data set, the true cpDAG is unknown, thus structural learning algorithms are required first to estimate the cpDAG from the observational data. As introduced in Section 1.4, constraint-based algorithms such as PC and score-based algorithms such as CCDr can be used to obtain a cpDAG. The consistency of PC algorithm is shown to hold on all causal graphs by Spirtes et al. (2001), i.e. $\lim_{n\to\infty} \mathbb{P}(\hat{G} \neq G) = 0$, where $\hat{G}$ is the cpDAG estimated by PC algorithm and G is the true cpDAG. Aragam and Zhou (2015) has provided the convergence rate of the CCDr algorithm. When these algorithms provide a correct estimate of the true cpDAG giving a sufficiently large sample size, the requirements of the consistency of DIMEO are satisfied.

3.4 Numerical results

In this section, we report the results of our algorithms on simulated graphs and a real world dataset. We first verified the results of the DIMEO algorithm given the population level data, i.e. the underlying distribution of $\mathbf{X}$ is known, thus $(\mathbf{A}, \mathbf{\Omega})$ in two populations can be used to calculate the difference indicator matrix. We also verified the effectiveness of the algorithm on simulated numerical data providing the true cpDAG structure. Finally we combined DIMEO with two existing structural learning algorithms and compared it to a recent nonlinear causal discovery method.

3.4.1 Population level results on true cpDAG

We use the cpDAG in Figure 3.3 as an example to verify the DIMEO algorithm's results. The graph has 5 nodes and 6 undirected edges. Despite the simple structure, it is representative and contains typical motifs in DAGs. While each edge can be either *differential* or *non-differential*, there are $2^6 - 1$ combinations in total (assume that at least one *differential* edge exists). We considered all the possible combinations of the *differential* edges in each of the DAGs in Figure 3.4, giving a total of 12 DAGs. We simulated data from each distinct DAG and checked if the edge directions in each DAG could be identified by the algorithm. In the simulation for both populations, the weights of the *non-differential* edges were sampled from a uniform distribution $\mathcal{U}(-1, 1)$, and the weights of *differential* edges were sampled from $\mathcal{U}(-1, 0.1)$ and $\mathcal{U}(0.5, 1.5)$ respectively. The standard deviation of each error was sampled from $\mathcal{U}(0.1, 0.2)$ independently.

Difference indicator matrix $\mathbf{M}$ could be obtained directly from the regression coefficients. Considering the computational accuracy, we took the element as 1 if the absolute difference between two coefficients was greater than 10^{-5} and 0 otherwise. Matrix $\mathbf{M}$ and the cpDAG structure in Figure 3.3 were the input to DIMEO. Table 3.1 has enumerated all the identifiable cases among the combinations. In the table the first column represents the corresponding true DAG in the equivalence class shown in Figure 3.4, the second column "underlying *diff*" represents the true *differential* edge, the third column "*indiff*" are the edges that can either be *differential* or *non-differential*, which have no effect in the orientation process. Except the *differential* and *indifferent* edges, the rest of the edges have to be *non-differential* to guarantee a successful orientation of DIMEO. Once the results from Algorithm 1 were generated, most of the other edges in the graph could be oriented by the Meek's rule. The columns "Algo 1 ori" and "Final ori" show the edges oriented by Algorithm 1 alone, and the final extended orientation results by DIMEO respectively. From Theorem 3.3, the single edge orientation output from Algorithm 1 could be extended in its neighborhood, for example, in DAG (III), when the *differential* edge $V_c \to V_a$ was oriented, all the other nodes connected to

V_a should be its children, thus we could write it as $V_c \to V_a \to \cdot$ (with no self loops formed).

Additionally, shown in column "Algo 1 additional", Algorithm 1 provided information about whether edges were *differential* according to Theorem 3.2. For example, when the direction was determined as $V_c \to V_a$ in DAG (III), we could know that the edge between V_c and V_a were *differential* while $V_a \to V_b$ and $V_c \to V_b$ were *non-differential*. Moreover, when the orientation was extended by Meek's rule, other *differential* oriented edges could also be detected through linear regression from the target nodes to their parents. In the case of DAG (III), we could further check whether $V_c \to V_d$ and $V_b \to V_d$ were *differential* by regressing V_d on V_b and V_c respectively.

Table 3.1: Population level results on Figure 3.3 graph.

DAG	underlying *diff*	*indiff*	Algo 1 ori	Algo 1 additional	Final ori
I	$V_d \to V_c$	$V_e \to V_d$	$V_d \to V_c \to \cdot$	$a'_{dc} \neq a''_{dc}, a'_{ca} = a''_{ca}, a'_{cb} = a''_{cb}, a'_{db} = a''_{db}$	All except $V_e \to V_d$
I	$V_e \to V_d$	$V_b \to V_a$, $V_c \to V_a$, $V_c \to V_b$	$V_e \to V_d \to \cdot$	$a'_{ed} \neq a''_{ed}, a'_{db} = a''_{db}, a'_{dc} = a''_{dc}$	All except $V_b \to V_c$
II	$V_d \to V_b$	$V_e \to V_d$	$V_d \to V_b \to \cdot$	$a'_{db} \neq a''_{db}, a'_{ba} = a''_{ba}, a'_{bc} = a''_{bc}, a'_{dc} = a''_{dc}$	All except $V_e \to V_d$
II	$V_e \to V_d$	$V_b \to V_a$, $V_c \to V_a$, $V_c \to V_b$	$V_e \to V_d \to \cdot$	$a'_{ed} \neq a''_{ed}, a'_{db} = a''_{db}, a'_{dc} = a''_{dc}$	All except $V_b \to V_c$
III	$V_c \to V_a$	$V_d \to V_e$	$V_c \to V_a \to \cdot$	$a'_{ca} \neq a''_{ca}, a'_{ab} = a''_{ab}, a'_{cb} = a''_{cb}$	All
IV	$V_b \to V_a$	$V_d \to V_e$	$V_b \to V_a \to \cdot$	$a'_{ba} \neq a''_{ba}, a'_{bc} = a''_{bc}, a'_{ac} = a''_{ac}$	All
V	$V_c \to V_d$	None	$V_c \to V_d \to \cdot$	$a'_{cd} \neq a''_{cd}, a'_{cb} = a''_{cb}, a'_{db} = a''_{db}, a'_{de} = a''_{de}$	All
VI	$V_b \to V_d$	None	$V_b \to V_d \to \cdot$	$a'_{bd} \neq a''_{bd}, a'_{bc} = a''_{bc}, a'_{dc} = a''_{dc}, a'_{de} = a''_{de}$	All
VII	$V_c \to V_a$	$V_d \to V_e$	$V_c \to V_a \to \cdot$	$a'_{ca} \neq a''_{ca}, a'_{ab} = a''_{ab}, a'_{cb} = a''_{cb}$	All
VIII	$V_d \to V_b$	$V_d \to V_e$	$V_d \to V_b \to \cdot$	$a'_{db} \neq a''_{db}, a'_{ba} = a''_{ba}, a'_{bc} = a''_{bc}, a'_{dc} = a''_{dc}$	All except $V_d \to V_e$
IX	$V_a \to V_c$	$V_d \to V_e$	$V_a \to V_c \to \cdot$	$a'_{ac} \neq a''_{ac}, a'_{ab} = a''_{ab}, a'_{cb} = a''_{cb}, a'_{cd} = a''_{cd}$	All
X	$V_a \to V_b$	$V_d \to V_e$	$V_a \to V_b \to \cdot$	$a'_{ab} \neq a''_{ab}, a'_{ac} = a''_{ac}, a'_{bc} = a''_{bc}, a'_{bd} = a''_{bd}$	All
XI	$V_c \to V_b$	None	$V_c \to V_b \to \cdot$	$a'_{cb} \neq a''_{cb}, a'_{ba} = a''_{ba}, a'_{ca} = a''_{ca}, a'_{cd} = a''_{cd}, a'_{bd} = a''_{bd}$	All
XII	$V_b \to V_c$	None	$V_b \to V_c \to \cdot$	$a'_{bc} \neq a''_{bc}, a'_{ba} = a''_{ba}, a'_{ca} = a''_{ca}, a'_{cd} = a''_{cd}, a'_{bd} = a''_{bd}$	All

We explain the results using the first row in Table 3.1 as an example. The true DAG structure is Figure 3.3 (I), when the underlying *differential* edge is $V_d \to V_c$ and all other edges except $V_e \to V_f$ are *non-differential* ($V_e \to V_f$ can be either *differential* or *non-differential*, the orientation results will be the same), provided the correct DIM $\mathbf{M}$ and cpDAG G. DIMEO first oriented edge V_d — V_c, and further provided orientation of V_c to other neighbors as well as the information on *differential* edges. From the output of single edge orientation

Algorithm 1, we have the orientation result $V_d \to V_c \to \cdot$ and information that $V_d \to V_c$ is a *differential* edge and $V_c \to V_a$, $V_c \to V_b$, $V_d \to V_b$ are all *non-differential*. DIMEO further extended the orientation from Algorithm 1 by Meek's rule and finally output orientation of all edges except $V_e \to V_d$.

We see that each DAG can be oriented, under some cases of *differential* edges combination. The power of the algorithm varies on the true DAG structure. For cases like DAG (I) and (II), 10 out of 63 cases could be oriented by DIMEO, while for cases like (V), (VI), (XI) and (XII), DIMEO only worked when there was only one *differential* edge. In total, 192 out of $63 \times 6 \times 12 = 4536$ (4%) undirected edges were oriented by DIMEO. However, when the number of *differential* edges was less than or equal to two, which is highly possible in a small network with only 6 edges, 152 out of 1512 (10%) edges could be oriented. And when the number of *differential* edge was only one, 78 out of 432 (18%) undirected edges were oriented.

3.4.2 Numerical results on true cpDAG

The population level, or the case when sample size goes to infinity, is the ideal case where we can achieve orientation with zero error. In this section we compare the performance of DIMEO on observed samples to the ideal case to verify the feasibility of the algorithm on numerical data.

Again we simulated data from the 5-node cpDAG (Figure 3.3) using the same settings as in Section 3.4.1. DIMEO was applied given the true cpDAG structure. In the simulation, we tuned the number of differential edges d from 1 to 6. When $d = 1$, there were 6 combinations because of the 6 undirected graph in the cpDAG, and when $d = 2$, there were $\binom{6}{2} = 15$ combinations and so on. We set the number of samples in each population as $N = \{5000, 10000, 50000\}$, and the p-value cut-off threshold $t = 0.005$. Since the weights were sampled randomly from uniform distributions, we repeated the process 100 times and report the average of the output.

Table 3.2: Numerical results on Figure 3.3 graph.

d	U	Ideal	N	TP	OC	R	M	acc_1(%)	diff	acc_2(%)
			5000	0.972	0.917	0.006	0.167	99	0.653	97
1	6	1.080	10000	1.028	1.000	0.000	0.083	100	0.694	98
			50000	1.000	1.000	0.000	0.083	100	0.781	100
			5000	0.619	0.556	0.020	0.048	97	0.398	93
1–2	6	0.603	10000	0.619	0.579	0.020	0.024	97	0.464	93
			50000	0.595	0.579	0.004	0.024	99	0.438	96
			5000	0.224	0.208	0.025	0.046	90	0.174	90
1–6	6	0.254	10000	0.245	0.213	0.015	0.041	94	0.178	92
			50000	0.251	0.245	0.001	0.009	99	0.188	94

Table 3.2 shows the accuracy of numerical results compared to the ideal case. In the table, d represents the number of *differential* edges in the true DAG, we illustrate three cases: $d = 1, \ldots, 6$, $d = 1, 2$, and $d = 1$. As the graph only has 5 vertices and 6 edges, it is likely that very few edges will be significantly different in two populations. Column "U" shows the average number of undirected edges in each cpDAG. Column "Ideal" is the number of oriented edges given the population level information ($N = \infty$), which is the number of orientations that can be justified by Theorem 3.4 in each graph. For the results on samples, the table provides: (1) TP: the number of total correct orientations, i.e. edges correctly oriented by DIMEO from the observed data; (2) OC: the number of overlapping edges that were correctly oriented in both ideal case and simulated case, which represents the number of orientations that can be justified by the algorithm; (3) R: the number of reversed edges, which were oriented by the algorithm but with wrong directions; (4) M: the number of missing orientations, which should be oriented in the ideal case but were failed to be detected by the algorithm from simulated data; (5) acc_1 = TP/(TP+R): accuracy of

oriented edges that have correct directions; (6) diff: average number of reported *differential* edges in each graph; (7) acc_2: accuracy of the reported differential edges, in which the false cases are the number of *non-differential* edges that were identified as *differential* edges by the algorithm. We can see that in all cases, the accuracy of orientation and information about differential edges were above 90%. And when $d = 1$, our algorithm achieved the best performance. Despite the fact that the overall power of the algorithm was relatively low, the high accuracy validates the usefulness of the algorithm. And compared to the ideal case, the application on samples had comparable performance, as both R and M were very small compared to OC and TP.

In summary, the DIMEO algorithm can detect *differential* edges and orient them in some cases. While it did not identify causal structure in all cpDAGs, it achieved a very high accuracy among the oriented edges in the simulation. And the information about *differential* edges was also overall reliable, which is another major advantage of DIMEO.

3.4.3 Comparison to other algorithms

In this section we implemented DIMEO on cpDAGs estimated from PC (Spirtes et al., 2001) and CCDr (Aragam and Zhou, 2015). We also compared this approach to a recent non-linear causal learning method named regression with subsequent independence test (RESIT) (Peters et al., 2014) developed from additive noise models (Hoyer et al., 2009). RESIT can identify causal relations from observational data by assuming non-linear noises. The first step of the algorithm is to iteratively identify and sink nodes to yield a topological order of the nodes. During this process, each remaining variable is regressed on all other variables and the dependence between the residuals and the regressors is measured through hypothesis tests. A variable will be sinked or removed if it has the least independence level among all variables. In the next step, based on the estimated topological order, edges are removed by further conditional independence tests. The RESIT algorithm was implemented in R by Peters et al. (2014).

Both PC and CCDr algorithms are available as R packages. We use the package *bnlearn* (Scutari, 2010) to perform PC algorithm, more details about the algorithm can be found in Colombo and Maathuis (2014). The R package *sparsebn* (Aragam et al., 2019) contains the CCDr algorithm, which is a score-based method that is achieved by maximizing a regularized likelihood under a concave penalty. The algorithm randomly outputs a DAG out of the equivalence class, we used the *cpdag* function in *bnlearn* package to convert the DAG to its corresponding cpDAG for causal discovery.

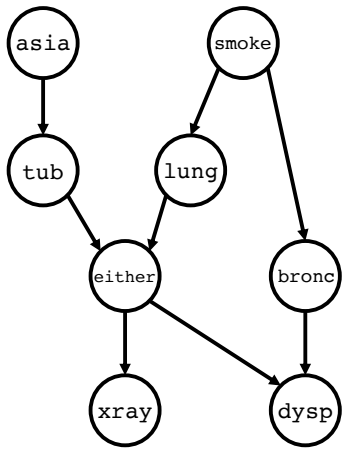


Figure 3.6: Network *asia*.

Due to the long computational time of RESIT, we ran these methods on a small network *Asia* (or sometimes called *lung cancer*) from *bnlearn* network repository (Scutari, 2010). The network has 8 nodes and 8 directed edges, corresponding to a cpDAG with 3 undirected edges, which belongs to an equivalence class with 6 distinct DAGs (Figure 3.6). We simulated data with number of *differential* edges varying from 1 to 8, the weights of the edges were sampled with the same settings as in Section 3.4.1. The sample size was $N = 1000$ for each population, and the experiments were repeated 100 times with different edge weights.

Table 3.3 provides the numerical results of the simulation. The columns have the same meaning as in Table 3.2. We applied DIMEO on cpDAGs estimated from PC and CCDr, the results are shown as "PC-DIMEO" and "CCDr-DIMEO" in the table respectively. CCDr in

general had more undirected edges in estimated cpDAGs compared to PC. Column "TP" represents the number of correct oriented edges in each graph, and "R" represents the reversed orientation. The accuracy is very high ($\geq 90\%$) when d is small, such as $d = 1, 2, 3$. And when d is large, i.e. most of the edges were *differential*, DIMEO still successfully oriented some edges, while the accuracy was slightly lower. Column "diff" shows the number of reported *differential* edges in each graph, the accuracy of this information was generally over 90% except for a few cases. From the table, it is clear to see the effectiveness of DIMEO in identification of causal relations from observational data.

Table 3.3: Numerical results on *asia* network simulation.

	PC-DIMEO						CCDr-DIMEO					
d	U	TP	R	$\text{acc}_1(\%)$	diff	$\text{acc}_2(\%)$	U	TP	R	$\text{acc}_1(\%)$	diff	$\text{acc}_2(\%)$
1	5.55	0.43	0.00	100	0.35	100	6.49	0.37	0.00	100	0.29	100
2	5.64	0.37	0.02	95	0.31	96	6.46	0.34	0.01	98	0.28	98
3	5.70	0.32	0.02	94	0.26	93	6.45	0.33	0.02	95	0.28	98
4	5.70	0.31	0.03	90	0.24	90	6.46	0.34	0.02	94	0.29	97
5	5.58	0.36	0.04	89	0.29	89	6.39	0.37	0.03	92	0.33	97
6	5.42	0.38	0.06	87	0.29	89	6.29	0.38	0.40	91	0.36	99
7	5.04	0.36	0.05	87	0.30	94	6.13	0.33	0.05	86	0.35	99
8	5.26	0.39	0.01	98	0.35	100	6.28	0.28	0.07	80	0.32	100

In order to evaluate and compare the performance of the algorithms, we used structural Hamming distance (SHD) and Jaccard index (JI). The SHD measures the shortest edit distance between the estimated graph and the true graph, it is the total number of additions, deletions and reversions of the edges needed to be taken to match the two DAGs. A lower SHD indicates a better performance of structural and causal estimation. JI measures the similarity between two graphs, it is the ratio of the number of common edges (with same direction) over the number of edges in the union of all edges from two DAGs. A higher JI

indicates higher similarity, thus better performance.

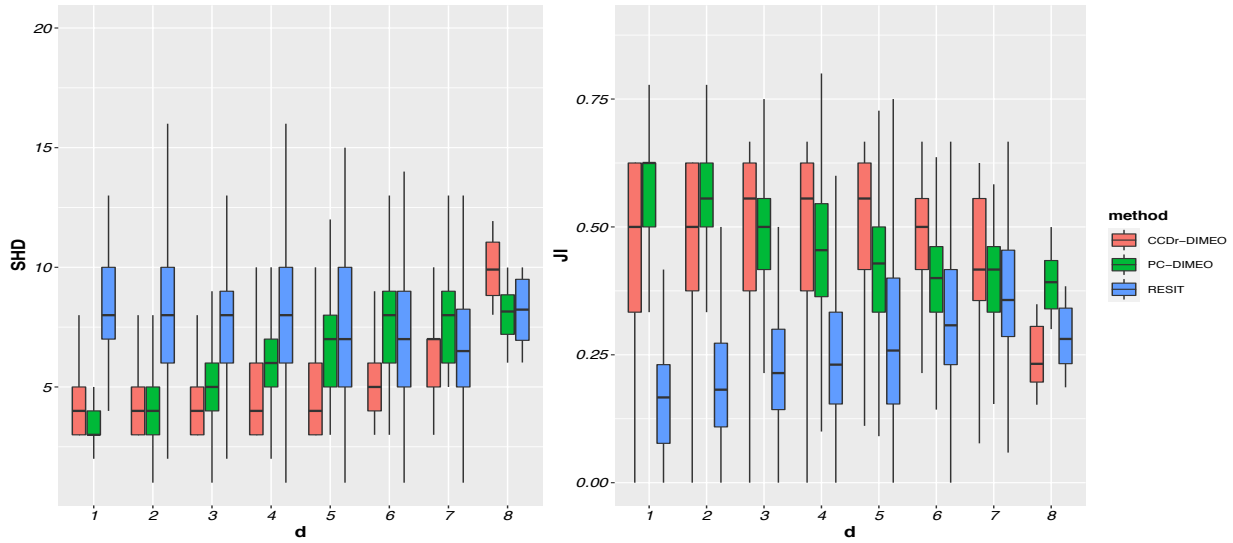


Figure 3.7: Comparison of algorithms on *asia* network.

Figure 3.7 is the box plot of SHD and JI of three methods: PC-DIMEO, CCDr-DIMEO and RESIT. The results were generated by 100 simulations on each d. PC-DIMEO and CCDr-DIMEO represent the methods first applying PC and CCDr to estimate a cpDAG and then running DIMEO on the cpDAG yielded from the structural learning step. For all algorithms the p-value cutoff threshold was set to 0.001. In the figure, the x-axis is d that represents the number of *differential* edges in the simulation, ranging from 1 to 8. We can see that among three methods, RESIT had a relatively poor performance on the mixed data, with the highest SHD and lowest JI in almost all cases, especially when $d \leq 4$. The reason could be that in the simulation the relationships between the variables were linear functions with Gaussian noises within one population, though the mixed data from two populations was essentially non-linear, RESIT failed to model this kind of non-linearity. PC-DIMEO had slightly better performance than CCDr-DIMEO in terms of SHD, while CCDr-DIMEO outperformed the others with respect to JI. These results show the effectiveness of DIMEO to deal with data generated from multiple populations, as the performance of both PC-DIMEO and CCDr-DIMEO were generally better than RESIT.

Meanwhile, as listed in Table 3.4 (performed on a single 2.6GHz Intel i7 core), the additional computational time required by including DIMEO was negligible. In summary, when the data came from different populations with an existence of *differential* edges, the DIMEO algorithm can efficiently improve the causal structural estimation results. The algorithm is simple to implement and requires little computational power, thus can be a very useful tool in causal learning.

Table 3.4: Algorithms average running time.

Method	PC	CCDr	PC-DIMEO	CCDr-DIMEO	RESIT
Time (seconds)	0.050	0.108	0.056	0.116	168

3.4.4 Application on protein-signaling network

A bayesian network has been established from the multidimensional flow cytometry data by Sachs et al. (2005). The authors have discovered 20 possible molecular pathways among 11 proteins/phospholipid, which constructed the true DAG structure (Figure 3.8) in our experiment. Their dataset has $N = 7466$ observations and was obtained under 5 experimental interventions on different nodes including *akt* ($N = 911$), *mek* ($N = 799$), *pip*2 ($N = 810$), *pka* ($N = 707$) and *pkc* ($N = 1636$), as well as the control group ($N = 2603$).

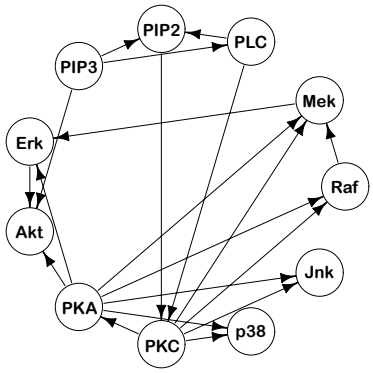


Figure 3.8: Causal protein-signaling network.

In order to conduct our experiment, we took the data under each intervention as a separate population, thus we had 6 populations in total. And in order to keep the same graph structure, when comparing the control group to other groups, we excluded the node that was under intervention from the graph. For example, when comparing the control group to group *akt*, the node *akt* and its connecting edges were removed from the true DAG structure, and we only estimated the structure among the other 10 nodes. In total, we have 5 pairs of groups to compare, i.e. control group versus 5 experimental groups. And when we input the data in the experimental groups, we did not use the information about which node was manipulated, consequently the data we used in causal learning were essentially purely observational.

We first applied PC and CCDr on the data and ran DIMEO on the estimated cpDAGs. The numerical results are shown in Table 3.5. In three groups: control vs. *akt*, control vs. *pip*2 and control vs. *pkc*, DIMEO failed to orient undirected edges, a possible reason could be that the intervened nodes in these groups did not result in *differential* edges required in our assumption. In the group control vs. *mek*, DIMEO successfully oriented 13% of the undirected edges from PC's output and 8% of the undirected edges from CCDr's output. Both orientation achieved very high accuracy over 98%. Similarly, for the group control vs. *pka*, DIMEO oriented 21% and 28% of the edges for PC and CCDr respectively, with accuracy rate 89% and 73%.

Figure 3.9 is the box plot of SHD and JI of the three methods. Some boxes have very concentrated values, which means the estimated graph structures were very similar across the 100 groups of samples. Both PC-DIMEO and CCDR-DIMEO had clear advantage compared to RESIT in terms of SHD, while three methods were relatively comparable evaluated by JI in groups control vs. *akt* and *mek*. PC-DIMEO clearly outperformed RESIT by JI in all the other three groups. Overall the two methods with DIMEO generated better structure learning results.

Table 3.5: Numerical results on flow cytometry data.

	PC-DIMEO				CCDr-DIMEO			
Group	U	TP	R	acc (%)	U	TP	R	acc (%)
control vs. *akt*	2.40	0	0	–	4.68	0	0	–
control vs. *mek*	6.74	0.87	0	100	4.90	0.39	0.01	98
control vs. *pip2*	2.16	0	0	–	3.25	0	0	–
control vs. *pka*	7.30	0.17	0.02	89	5.23	1.46	0.53	73
control vs. *pkc*	7.18	0	0	–	3.46	0	0	–

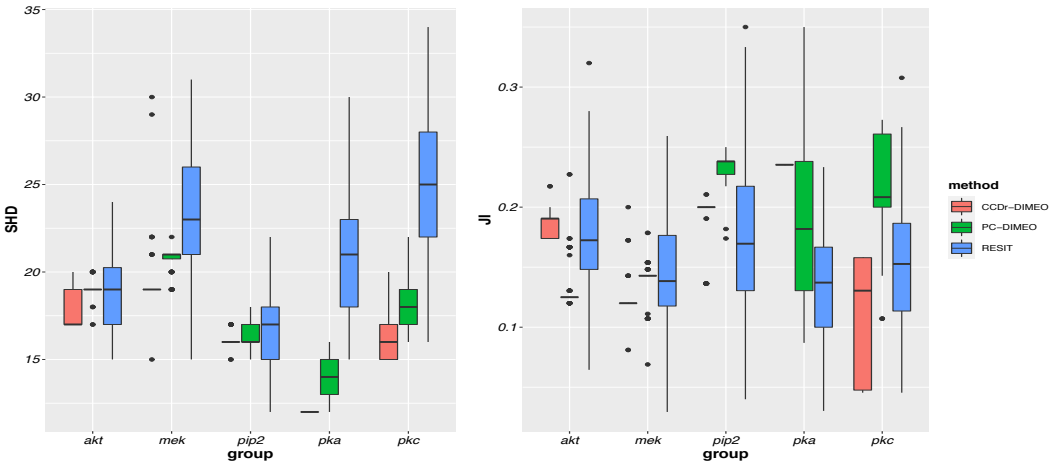


Figure 3.9: Comparison of algorithms on flow cytometry data.

3.5 Proofs

3.5.1 Differential edges and difference indicator matrix

As shown in Figure 3.10, in this section we justify the effect of *differential* edges on the difference indicator matrix. In each case, the red node is regressed on its Markov blanket, which includes the blue and gray nodes. We use the blue node to represent the target variable we are interested in. The edge in red color represents the *differential* edge. We discuss three

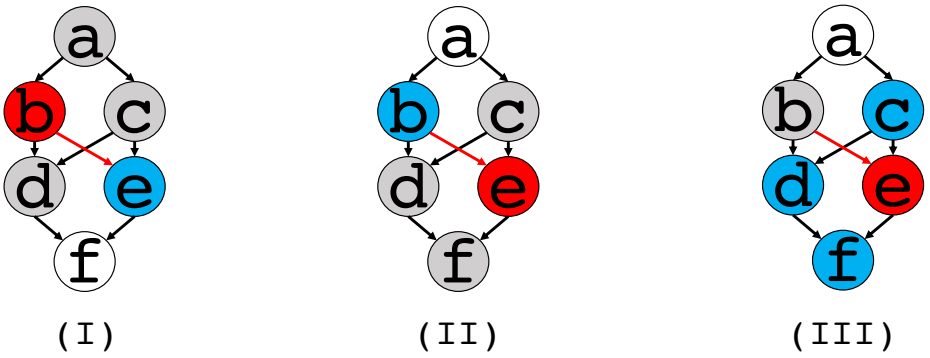


Figure 3.10: Target and regressor nodes.

situations (I), (II), (III) in the figure and show the effect of the *differential* edges on DIM.

In (3.10), the regression coefficient is the ratio of two elements in the precision matrix $\mathbf{\Theta}$. Since we have $\mathbf{\Omega}' = \alpha\mathbf{\Omega}''$ in two populations, the scaling factor α gets cancelled out, and the value of α and $\mathbf{\Omega}'$, $\mathbf{\Omega}''$ do not affect the change of coefficients.

Case I. Regression coefficients of $X_e|X_b \sim mb(X_b)$

Denote the coefficients of regressing X_b on $mb(X_b)$ in two population as $\mathbf{b}'$ and $\mathbf{b}''$. First let us check the change of $b_{X_e|(X_b\sim mb(X_b))}$ when $X_e \in mb(X_b)$. Assume $ch^*(X_b) \neq \emptyset$, taking the set of weights $\{a_{bk}\}_{k:X_k\in ch^*(X_b)}$ as variables in the function, then (3.12) can be written as

$$b_{X_e|(X_b\sim mb(X_b))}(\{a_{bk}\}_{k:X_k\in ch^*(X_b)}) = -\frac{\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}a_{ek} + C_1}{\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}^2 + C_2}, \tag{3.14}$$

where C_1, C_2 are constant with respect to $\{a_{bk}\}_{k:X_k\in ch^*(X_b)}$. By taking the first order partial derivative with respect to a single variable a_{bk}, we have

$$\frac{\partial b_{X_e|(X_b\sim mb(X_b))}}{\partial a_{bk}} = -\frac{\frac{1}{\omega_k^2}a_{ek}(\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}^2 + C_2) - \frac{2}{\omega_k^2}a_{bk}(\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}a_{ek} + C_1)}{(\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}^2 + C_2)^2}, \tag{3.15}$$

which is essentially a quadratic polynomial of a_{bk} divided by the square of another polynomial of a_{bk}. It is clear that $\frac{\partial b_{X_e|(X_b\sim mb(X_b))}}{\partial a_{bk}}$ has finite zero points as the numerator is a non-zero polynomial. Consequently, given that $\sum_{X_k\in ch^*(X_b)} \frac{1}{\omega_k^2} a_{bk}a_{ek} + C_1 \neq 0$, $b_{X_e|(X_b\sim mb(X_b))}$

does not have constant piece with respect to a_{bk}, which also applies to other variables in $\{a_{bk}\}_{k:X_k\in ch^*(X_b)}$. In conclusion, $b'_{X_e|(X_b\sim mb(X_b))} \neq b''_{X_e|(X_b\sim mb(X_b))}$ for $\forall X_e \in mb(X_b)$ given $ch^*(X_b) \neq \emptyset$, i.e. when at least one of the edges from X_b to $ch(X_b)$ is *differential*, the regression coefficients $\mathbf{b}'_{X_b\sim mb(X_b)}$ are element-wisely different from $\mathbf{b}''_{X_b\sim mb(X_b)}$.

In the figure, as $mb(X_b) = \{X_a, X_c, X_d, X_e\}$, we have $m_{ba} = m_{bc} = m_{bd} = m_{be} = 1$ in DIM.

Case II. Regression coefficients of $X_b|X_e \sim mb(X_e)$

Under the same assumption that $ch^*(X_b) \neq \emptyset$, for a node $X_e \in mb(X_b)$, suppose outside $mb(X_b)$ there is no other *differential* edges, i.e. $mb^*(X_e)\backslash mb(X_b) = \emptyset$. When X_e is regressed on its Markov blanket $mb(X_e)$, the coefficient of X_b can be written as

$$b_{X_b|(X_e\sim mb(X_e))} = -\frac{-\frac{1}{\omega_e^2}a_{eb} - \frac{1}{\omega_b^2}a_{be} + \sum_{k:X_k\in ch(X_b)\cap ch(X_e)} \frac{1}{\omega_k^2}a_{bk}a_{ek}}{\frac{1}{\omega_e^2} + \sum_{k:X_k\in ch(X_e)} \frac{1}{\omega_k^2}a_{ek}^2}, \tag{3.16}$$

which can also be seen as a function of $\{a_{bk}\}_{k:X_k\in ch^*(X_b)}$. The behavior of the function depends on the value of a_{be}, a_{eb} and $a_{bk}a_{ek}$ for $X_k \in ch(X_b) \cap ch(X_e)$.

- If $X_e \in ch(X_b)\backslash sp(X_b)$, $b_{X_b|(X_e\sim mb(X_e))} = Ca_{be}$, thus $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$ if and only if $a'_{be} \neq a''_{be}$, i.e. if $X_e \in ch^*(X_b)$ then $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$.
- If $X_e \in pa(X_b)\backslash sp(X_b)$, $b_{X_b|(X_e\sim mb(X_e))} = \frac{C_1 a_{eb}}{C_2+\frac{1}{\omega_b^2}a_{eb}^2}$, $\frac{\partial b_{X_b|(X_e\sim mb(X_e))}}{\partial a_{eb}}$ has finite zero points, thus $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$ if and only if $a'_{eb} \neq a''_{eb}$, i.e. if $X_e \in pa^*(X_b)$ then $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$.
- If $X_e \in sp(X_b)$, $b_{X_b|(X_e\sim mb(X_e))} = C + \sum_{X_e\in sp(X_b)} \frac{1}{\omega_k^2}a_{bk}a_{ek}$, thus $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$ if and only if $a'_{bk}a'_{ek} \neq a''_{bk}a''_{ek}$, i.e. if $X_e \in sp^*(X_b)$ then $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$.

In conclusion, $b'_{X_b|(X_e\sim mb(X_e))} \neq b''_{X_b|(X_e\sim mb(X_e))}$ for $\forall X_e \in pa^*(X_b) \cup ch^*(X_b) \cup sp^*(X_b)$, and $b'_{X_b|(X_e\sim mb(X_e))} = b''_{X_b|(X_e\sim mb(X_e))}$ for all other nodes X_e in $mb(X_b)$.

In the figure, as $X_e, X_c \in pa^*(X_b) \cup ch^*(X_b) \cup sp^*(X_b)$, we have $m_{eb} = m_{cb} = 1$ in DIM.

Case III. Regression coefficients of $X_d|X_e \sim mb(X_e)$

Assuming the same situation above where $ch^*(X_b) \neq \emptyset$ and $mb^*(X_e)\backslash mb(X_b) = \emptyset$ for $\forall X_c \in mb(X_b)$, we now prove that the coefficients of X_c in regression of X_e on $mb(X_e)$ are the same in two populations except for case II mentioned above ($b_{X_b|(X_e \sim mb(X_e))}$ in the figure). For a node $X_d \in mb(X_e)\backslash pa^*(X_e)$, we have

$$b_{X_d|(X_e \sim mb(X_e))} = -\frac{-\frac{1}{\omega_e^2}a_{ed} - \frac{1}{\omega_d^2}a_{de} + \sum_{k:X_k \in ch(X_d) \cap ch(X_e)} \frac{1}{\omega_k^2}a_{dk}a_{ek}}{\frac{1}{\omega_e^2} + \sum_{k:X_k \in ch(X_e)} \frac{1}{\omega_k^2}a_{ek}^2}, \tag{3.17}$$

as $mb^*(X_e)\backslash mb(X_b) = \emptyset$, we have $a'_{ed} = a''_{ed}$, and $a'_{ek} = a''_{ek}$ as well as $a'_{dk}a'_{ek} = a''_{dk}a''_{ek}$ for $\forall X_k \in mb(X_e)\backslash pa^*(X_e)$. Thus $b'_{X_d|(X_e \sim mb(X_e))} = b''_{X_d|(X_e \sim mb(X_e))}$, i.e. the *differential* edges connected to X_b does not affect coefficients of regression among other nodes in $mb(X_b)$.

In the figure, if $V_b \to V_e$ is the only *differential* edge, for all other elements m_{ij} in DIM except case I and II explained above, $m_{ij} = 0$.

3.5.2 Single edge orientation

Proof of Theorem 3.1

Given the correct DIM $\mathbf{M} = (m_{ij})$, we aim to orient an edge between two nodes V_a and V_b. Suppose the true direction is $V_a \to V_b$, then the correct $\mathbf{M}$ should have $m_{ab} = 1$, i.e. $X_a \in C_b$. We have the following possible situations about the *differential* edge:

(1) V_a has a *differential* edge to at least one of the nodes in $ch(X_a)$ and/or

(2) V_b has a *differential* edge to a common child of V_a and V_b, i.e. nodes in $ch(X_a) \cap ch(X_b)$.

If (1) holds, we should have $\sum_{X_k \in mb(X_a)}(1 - m_{ak}) = 0$, since $\forall X_k \in mb(X_a)$, $m_{ak} = 1$.

If (2) holds, the common child $X_k \in P_a \cap P_b$, also $X_k \in C_b$ since it is in $mb(X_b)$, thus $\{X_k\} \subseteq P_a \cap P_b \cap C_b$.

In order to orient the edges as $V_a \leftarrow V_b$, three requirements have to be met: $X_a \in C_b$, graph G has $P_a \cap P_b \cap C_b = \emptyset$, and $\sum_{X_k \in mb(X_a)}(1 - m_{ak}) > 0$. As situation (1) and (2)

are both contradictory to the requirements to orient the edge as $V_a \leftarrow V_b$, the algorithm guarantees no false orientation on $V_a \rightarrow V_b$.

Proof of Theorem 3.2

If an edge $V_a \rightarrow V_b$ is oriented, three requirements in Algorithm 1 have to be satisfied. As $\sum_{X_k \in mb(V_b)}(1 - m_{bk}) > 0$, all outgoing edges of node V_b are *non-differential*. And $X_b \in C_a$, i.e. $m_{ba} = 1$, either $V_a \rightarrow V_b$ is *differential* or there exists a *collider* V_c so that $V_a \rightarrow V_c \leftarrow V_b$, and $V_a \rightarrow V_c$ is *differential*. However, the later situation violates the requirement that $P_a \cap P_b \cap C_a = \emptyset$ because $X_c \in P_a \cap P_b$ and $m_{ca} = 1$, which means that $X_c \in P_a \cap P_b \cap C_a$. This observation is summarized in Theorem 3.2.

Proof of Theorem 3.3

If an edge $V_a \rightarrow V_b$ is determined, we can further orient other edges involving V_b. For a node V_c that is not connected to V_a, the edge direction can be determined since no new v-structure should be introduced given the cpDAG. For a node V_c connected to both V_a and V_b, if V_c is the parent node of V_b, since $X_b \in P_a \cap P_c$ and $X_b \in C_a$ (requirement for this orientation), we should have $X_b \in P_a \cap P_b \cap C_a$, which is contradictory to the requirement that $P_a \cap P_b \cap C_a = \emptyset$. Consequently, as written in Theorem 3.3, when an edge $V_a \rightarrow V_b$ is oriented by Algorithm 1, all other edges should be incoming edges to V_b as well.

CHAPTER 4

Summary

The dissertation works on two problems associated with graphical models. The first one is focused on undirected SBM and graphon models, we proposed an empirical Bayes (EB) estimate as an alternative to the classic maximum likelihood estimate (MLE). Compared to MLE and variational Bayes EM, our EB method achieved much higher accuracy on various networks generated by stochastic blockmodels and graphons and on two well-annotated real world networks. Besides, we have proposed a likelihood-based model selection criterion, which generally outperforms other popular existing criteria. The second problem is about causal discovery from multiple populations, we proposed a simple algorithm (DIMEO) based on node wise regression within different populations. Several experiments show the fact that DIMEO can efficiently deal with mixed data generated from different underlying populations, providing a great boost to the existing structural learning methods.

4.1 Contributions

In Chapter 2, we developed an empirical Bayes estimate for the probabilities of edge connections between communities in a network. While empirical Bayes (EB) under a hierarchical model is a well-established method, its application to SBMs is very limited before our work. Our method is a natural fit to the SBM and the idea is generally applicable to different community detection methods. It does not require complicated algorithms or heavy computation, yet can effectively improve the estimation accuracy of model parameters. For the large volume of published community detection or network clustering algorithms, our pa-

rameter estimation method can be adopted as a superior alternate after the node clustering step. SBM approximation to graphons could result in a large number of blocks, for which case the EB often shows substantial advantage over the MLE, and this was a key motivation for our generalization to graphon estimation.

In Chapter 3, we developed an algorithm using difference indicator matrix to discover causal relations when the observed data were generated from different populations. Our proposed algorithm DIMEO orients undirected edges in cpDAG by using difference indicator matrix obtained from node wise regression. Several simulation studies and an application on protein-signaling network data verified the effectiveness of the DIMEO algorithm. The experiments on population level data (ideal case) and numerical simulations demonstrated the high accuracy of the method. DIMEO applied after PC and CCDr algorithms successfully oriented edges in cpDAGs, and outperformed a popular non-linear causal learning algorithm RESIT in both simulation and real data experiments. Another major advantage of the algorithm is its simplicity for implementation and low computation complexity, which makes it a handy choice for causal learning when dealing with observational data generated from different populations.

4.2 Discussion

For graphon estimation, though shrinkage in empirical Bayes approach leads to more accurate estimate of the connectivity probabilities, the improvement depends on the variability of the underlying connectivity matrix or graphon function. Typically, a higher variance reduces its improvement relative to the MLE. Therefore, for some graphon functions with high volatility, EB cannot guarantee a better estimate, but from our simulation results, EB estimate and MLE are usually comparable for such cases. A key observation from the numerical comparisons is that the shrinkage estimation in our EB method improves the accuracy of the estimator regardless of the accuracy in the node clustering step. A main reason for this

observation is that EB estimate uses a very small number of hyperparameters, which effectively reduces the model complexity and greatly minimizes the risk of overfitting the data. This also helps the development of a good model selection criterion based on the marginal likelihood.

We put a beta conjugate prior on connection probability Θ, and the estimates of the hyperparameters $(\alpha_d, \beta_d)_{d=\{0,1\}}$ will not be 0. Thus when there is a true connectivity $\theta_{ab} = 0$ in block (a, b), which is likely to happen in sparse networks, our hierarchical model introduces bias to the estimate of θ_{ab}. However, since the empirical Bayes estimator is pooling data in all the blocks, the overall accuracy should still be higher. To alleviate this biased fitting problem, we can build the likelihood only on blocks with observed connections, or consider adding only a proportion α of zero connectivity blocks. This method can be tested with more experiments to find out which α works the best under different assumptions of SBM and graphon.

We compared the model estimation accuracy by their mean squared error, which is a gold standard criterion to evaluate parameter estimation. However several other metrics such as KL-divergence of the estimated graphon function to the truth, deviation of the estimated number of motifs in the graph to the true value, and divergence of degree distributions can also be considered. For the application on real data, the goodness of fit of SBM or graphon model to the dataset should be checked by comparison to other exisiting network modeling methods. A decent fit of the stochastic blockmodel and graphon to the chosen dataset will strengthen the persuasiveness of the usefulness of our method.

For the causal discovery work in the dissertation, our algorithm provides stable output of orientation, and the potential uncertainty of the output is essentially from the data itself. From the output of DIMEO we receive no information about the confidence level of the estimation. In order to provide a measure of uncertainty of the output, we can use bootstrap sampling from the data to yield a non-parametric distribution of the orientation results in the graph, thus have a bootstrap confidence level of the estimated direction of each oriented

edge.

4.3 Future works

As for the SBM and graphon estimation project, we have focused on parameter estimation for binary and assortative stochastic blockmodels and graphons. In fact, this idea can be generalized to more sophisticated random graph models, such as SBM with mixed memberships (Airoldi et al., 2008), SBM with weighted edges (Aicher et al., 2014), Nested (hierarchical) SBM (Peixoto, 2014), and bipartite SBM (Larremore et al., 2014) etc. In particular, the empirical Bayes method can be applied after node clustering to improve the estimation accuracy and to identify a proper number of clusters for these models. This is left as future work.

As for the second project, the current DIMEO algorithm is based on an assumption of Gaussian noise and relies on linear regression to calculate difference indicator matrix. While in many real world networks the noise distribution are non-Gaussian, and the weights of the edges are not linear functions of the parent variables, in some cases the functions are discrete. The idea can be extended to discrete DAGs and linear regression should be replaced with generalized linear models. Usually in these cases a weighted adjacency matrix does not have a simple form, and the relationship between the graph structure and DIM cannot be easily found as in the Gaussian DAGs.

Here we consider a simple binary data case where the effect of variables can still be quantified with simple numerical values. Assuming a large scale of data, the probability distribution of X_i given its Markov blanket $mb(X_j) = \{X_i, X_k\}$ can be listed as a contingency table. Assume that each edge $V_i \to V_j$ has a value β_{ij} that is similar to the edge weight in our method. From the observed data, in each population we are able to estimate probabilities

$\mathbb{P}(X_j = 1)$ with logistic regression

$$\log \frac{\mathbb{P}(X_j = 1)}{1 - \mathbb{P}(X_j = 1)} = \beta_0 + \beta_{ij} x_i + \beta_{kj} x_k, \tag{4.1}$$

where $x_i, x_k = \{0, 1\}$ and β_{ij}, β_{kj} are the regression coefficients. If a *differential* edge $V_i \to V_j$ is interpreted as an edge with different β_{ij} in two populations, the difference indicator matrix $\mathbf{M}$ is obtained from the comparison of β_{ij} for $i, j = \{1, \ldots, p\}$ in two populations.

The development of DIMEO based on other generalized linear model remains as a major future extension of our work.

Meanwhile, the GRN construction still remains to be a challenging task, especially when the gene expression data is obtained from single cell sequencing, which is known to suffer from missing values and typically follows bi-model distributions that are more difficult to model. While DIMEO deals with the *differential* edges in GRN, it does not solve the problem of fitting the non-linear functions between genes. If a generalized version of DIMEO can be developed and implemented we could hope to see a big step forward in the field of GRN inference.

Bibliography

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

Aicher C, Jacobs AZ, Clauset A (2014) Learning latent block structure in weighted networks. Journal of Complex Networks 3(2):221–248

Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. J Mach Learn Res 9:1981–2014

Airoldi EM, Costa TB, Chan SH (2013) Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In: Burges CJC, Bottou L, Welling M, Ghahramani Z, Weinberger KQ (eds) Advances in Neural Information Processing Systems 26, Curran Associates, Inc., pp 692–700

Albert R, Barabási AL (2002) Statistical mechanics of complex networks. Reviews of Modern Physics 74(1):47–97

Andersson SA, Madigan D, Perlman MD (1997) A characterization of Markov equivalence classes for acyclic digraphs. The Annals of Statistics 25(2):505 – 541

Aragam B, Zhou Q (2015) Concave penalized estimation of sparse gaussian bayesian networks. Journal of Machine Learning Research 16(69):2273–2328

Aragam B, Gu J, Zhou Q (2019) Learning large-scale Bayesian networks with the sparsebn package. Journal of Statistical Software 91(11):1–38

Bang-Jensen J, Gutin GZ (2002) Digraphs - theory, algorithms and applications. Springer

Barabási AL, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509–512

Beal M, Ghahramani Z (2003) The variational bayesian em algorithm for incomplete data : with application to scoring graphical model structures. Bayesian Statistics 7:453–464

Bickel PJ, Chen A (2009) A nonparametric view of network models and newman–girvan and other modularities. Proceedings of the National Academy of Sciences 106(50):21068–21073

Bickel PJ, Chen A, Levina E (2011) The method of moments and degree distributions for network models. The Annals of Statistics 39(5):2280 – 2301

Boppana RB (1987) Eigenvalues and graph bisection: An average-case analysis. In: 28th Annual Symposium on Foundations of Computer Science (sfcs 1987), pp 280–285

Borgs C, Chayes J (2017) Graphons: A nonparametric method to model, estimate, and design algorithms for massive networks. In: Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17, p 665672

Borgs C, Chayes J, Cohn H, Holden N (2016) Sparse exchangeable graphs and their limits via graphon processes. Journal of Machine Learning Research 18

Van den Broeck L, Gordon M, Inz D, Williams C, Sozzani R (2020) Gene regulatory network inference: Connecting plant biology and mathematical modeling. Frontiers in Genetics 11:457

Byrd R, Lu P, Nocedal J, Zhu C (1995) A limited memory algorithm for bound constrained optimization. SIAM Journal of Scientific Computing 16:1190–1208

Caron F, Rousseau J (2017) On sparsity and power-law properties of graphs based on exchangeable point processes. arXiv: Statistics Theory

Channarond A, Daudin JJ, Robin S (2012) Classification and estimation in the stochastic blockmodel based on the empirical degrees. Electron J Statist 6:2574–2601

Chaudhuri K, Chung F, Tsiatas A (2012) Spectral clustering of graphs with general degrees in the extended planted partition model. In: Proceedings of the 25th Annual Conference on Learning Theory, Proceedings of Machine Learning Research, vol 23, pp 35.1–35.23

Chickering DM (2003) Optimal structure identification with greedy search. J Mach Learn Res 3:507554

CHOI DS, WOLFE PJ, AIROLDI EM (2012) Stochastic blockmodels with a growing number of classes. Biometrika 99(2):273–284

Chu LF, Leng N, Zhang J, Hou Z, Mamott D, Vereide D, Choi J, Kendziorski C, Stewart R, Thomson J (2016) Single-cell rna-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. Genome Biology 17

Clogg CC, Petkova E, Haritou A (1995) Statistical methods for comparing regression coefficients between models. American Journal of Sociology 100(5):1261–1293

COJA-OGHLAN A (2010) Graph partitioning via adaptive spectral techniques. Combinatorics, Probability and Computing 19(2):227284

Colombo D, Maathuis MH (2014) Order-independent constraint-based causal structure learning. Journal of Machine Learning Research 15(116):3921–3962

Cooper G, Herskovits E (2004) A bayesian method for the induction of probabilistic networks from data. Machine Learning 9:309–347

Crick FH (1958) On protein synthesis. Symp Soc Exp Biol 12:138–163

Crick FH (1970) Central dogma of molecular biology. Nature 227(5258):561–563

D M, Heckerman D (1997) Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. Machine Learning 29:181–212

Danon L, Díaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment 2005(09):P09008–P09008

Daudin JJ, Picard F, Robin S (2008) A mixture model for random graphs. Statistics and Computing 18(2):173–183

Efron B (2010) Large-Scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction. Institute of Mathematical Statistics Monographs, Cambridge University Press

EH D (2012) In: Gene Activity in Early Development, Elsevier

Emmert-Streib F, Dehmer M, Haibe-Kains B (2014) Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. Frontiers in Cell and Developmental Biology 2:38

Fan X, Cao L, Da Xu RY (2015) Dynamic infinite mixed-membership stochastic blockmodel. IEEE Transactions on Neural Networks and Learning Systems 26(9):2072–2085

Freeman LC (1983) Spheres, cubes and boxes: Graph dimensionality and network structure. Social Networks 5(2):139 – 156

Fu F, Zhou Q (2013) Learning sparse causal gaussian networks with experimental intervention: Regularization and coordinate descent. Journal of the American Statistical Association 108(501):288–300

Funke T, Becker T (2019) Stochastic block models: A comparison of variants and inference methods. PLOS ONE 14(4):1–40

Gao B, Cui Y (2015) Learning directed acyclic graphical structures with genetical genomics data. Bioinformatics 31(24):3953–3960

Gao C, Lu Y, Zhou HH (2015) Rate-optimal graphon estimation. The Annals of Statistics 43(6):2624–2652

Garber M, Grabherr M, Guttman M, Trapnell C (2011) Computational methods for transcriptome annotation and quantification using rna-seq. Nature methods 8:469–77

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99(12):7821–7826

Glymour C, Zhang K, Spirtes P (2019) Review of causal discovery methods based on graphical models. Frontiers in Genetics 10:524

Gopalan PK, Gerrish S, Freedman M, Blei D, Mimno D (2012) Scalable inference of overlapping communities. In: Advances in Neural Information Processing Systems, vol 25

Greenland S, Pearl J, Robins J (1999) Causal diagrams for epidemiologic research. Epidemiology (Cambridge, Mass) 10(1):3748

Handcock M, Raftery A, Tantrum J (2007) Model-based clustering for social networks. Journal of the Royal Statistical Society: Series A (Statistics in Society) 170:301 – 354

Hauser A, Bühlmann P (2012) Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. J Mach Learn Res 13(1):24092464

Heckerman D, Geiger D, Chickering D (1995) Learning bayesian networks: The combination of knowledge and statistical data. Machine Learning 20:197–243

Hoff PD, Raftery AE, Handcock MS (2002) Latent space approaches to social network analysis. Journal of the American Statistical Association 97(460):1090–1098

Hoyer P, Janzing D, Mooij JM, Peters J, Schölkopf B (2009) Nonlinear causal discovery with additive noise models. In: Advances in Neural Information Processing Systems, vol 21

Hu JX, Thomas CE, Brunak S (2016) Network biology concepts in complex disease comorbidities. Nature reviews Genetics 17(10):615629

Huynh-Thu VA, Sanguinetti G (2019) Gene Regulatory Network Inference: An Introductory Survey, Springer New York, New York, NY, pp 1–23

Jeffreys H (1946) An invariant form for the prior probability in estimation problems. Proceedings of the Royal Society of London Series A Mathematical and Physical Sciences 186(1007):453–461

Judea P (2010) An Introduction to Causal Inference. The International Journal of Biostatistics 6(2):1–62

Kallenberg O (1999) Multivariate sampling and the estimation problem for exchangeable arrays. Journal of Theoretical Probability 12:859–883

Karrer B, Newman MEJ (2011) Stochastic blockmodels and community structure in networks. Phys Rev E 83:016107

Katz D, Shanmugam K, Squires C, Uhler C (2019) Size of interventional markov equivalence classes in random dag models. In: Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, PMLR, Proceedings of Machine Learning Research, vol 89, pp 3234–3243

Klopp O, Tsybakov AB, Verzelen N (2017) Oracle inequalities for network models and sparse graphon estimation. The Annals of Statistics 45(1):316–354

Kuwabara PE (2003) Dna microarrays and gene expression: From experiments to data analysis and modeling. Briefings in Functional Genomics 2(1):80–81

Larremore DB, Clauset A, Jacobs AZ (2014) Efficiently inferring community structure in bipartite networks. Phys Rev E 90:012805

Latouche P, Robin S (2016) Variational bayes model averaging for graphon functions and motif frequencies inference in w-graph models. Statistics and Computing 26(6):1173–1185

Latouche P, Birmelé E, Ambroise C (2011) Overlapping stochastic block models with application to the french political blogosphere. The Annals of Applied Statistics 5(1):309–336

Latouche P, Birmele E, Ambroise C (2012) Variational bayesian inference and complexity control for stochastic block models. Statistical Modelling 12(1):93–115

Lawrence N (2005) Probabilistic non-linear principal component analysis with gaussian process latent variable models. Journal of Machine Learning Research 6(60):1783–1816

Lee C, Wilkinson DJ (2019) A review of stochastic block models and extensions for graph clustering. Applied Network Science 4(1):122

Leskovec J, Krevl A (2014) SNAP Datasets: Stanford large network dataset collection

Li KC (2002) Genome-wide coexpression dynamics: Theory and application. Proceedings of the National Academy of Sciences 99(26):16875–16880

Li W, Ahn S, Welling M (2015) Scalable mcmc for mixed membership stochastic blockmodels. 1510.04815

Li W, Ahn S, Welling M (2016) Scalable mcmc for mixed membership stochastic blockmodels. pp 723–731

Lloyd J, Orbanz P, Ghahramani Z, Roy DM (2012) Random function priors for exchangeable arrays with applications to graphs and relational data. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) Advances in Neural Information Processing Systems 25, Curran Associates, Inc., pp 998–1006

Lloyd S (1982) Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2):129–137

Lovasz L, Szegedy B (2006) Limits of dense graph sequences. Journal of Combinatorial Theory, Series B 96(6):933 – 957

Lu X, Szymanski B (2019) A regularized stochastic block model for the robust community detection in complex networks. Scientific Reports 9

Madhamshettiwar P, Maetschke S, Davis M, Reverter A, Ragan M (2012) Gene regulatory network inference: Evaluation and application to ovarian cancer allows the prioritization of drug targets. Genome medicine 4:41

Mariadassou M, Robin S, Vacher C (2010) Uncovering latent structure in valued graphs: A variational approach. Ann Appl Stat 4(2):715–742

Meek C (1995) Causal inference and causal explanation with background knowledge. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, p 403410

Monti RP, Zhang K, Hyvärinen A (2020) Causal discovery with general non-linear relationships using non-linear ica. In: Proceedings of The 35th Uncertainty in Artificial Intelligence Conference, PMLR, Proceedings of Machine Learning Research, vol 115, pp 186–195

Mooij J, Peters J, Janzing D, Zscheischler J, Schölkopf B (2016) Distinguishing cause from effect using observational data: Methods and benchmarks. Journal of Machine Learning Research 17

Mortazavi A, Williams B, Mccue K, Schaeffer L, Wold B (2008) Mapping and quantifying mammalian transcriptomes by rna-seq. Nature methods 5:621–8

Mørup M, Schmidt MN, Hansen LK (2011) Infinite multiple membership relational modeling for complex networks. 1101.5097

Nagalakshmi U, Waern K, Snyder M (2010) Rna-seq: A method for comprehensive transcriptome analysis. Current protocols in molecular biology Chapter 4:Unit 4.11.1–13

Newman MEJ (2004) Fast algorithm for detecting community structure in networks. Physical Review E 69(6)

Newman MEJ, Watts DJ, Strogatz SH (2002) Random graph models of social networks. Proceedings of the National Academy of Sciences 99(suppl 1):2566–2572

Ng AY, Jordan MI, Weiss Y (2001) On spectral clustering: Analysis and an algorithm. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, MIT Press, pp 849–856

Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. Journal of the American Statistical Association 96(455):1077–1087

Olhede SC, Wolfe PJ (2014) Network histograms and universality of blockmodel approximation. Proceedings of the National Academy of Sciences 111(41):14722–14727

Orbanz P, Roy DM (2015) Bayesian models of graphs, arrays and other exchangeable random structures. IEEE Transactions on Pattern Analysis and Machine Intelligence 37(2):437–461

Palla K, Knowles DA, Ghahramani Z (2012) An infinite latent attribute model for network data. In: Proceedings of the 29th International Coference on International Conference on Machine Learning, p 395402

Passador-Gurgel G, Hsieh WP, Hunt P, Deighton N, Gibson G (2007) Quantitative trait transcripts for nicotine resistance in drosophila melanogaster. Nature Genetics 39:264–268

Paternoster R, BRAME R, Mazerolle P, Piquero A (1998) Using the correct statistical test for equality of regression coefficients. Criminology 36:859 – 866

Pearl J (1988) Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference

Pearl J (2009) Causality, 2nd edn. Cambridge University Press

Peixoto TP (2014) Hierarchical block structures and high-resolution model selection in large networks. Phys Rev X 4:011047

Pennisi E (2012) Single-cell sequencing tackles basic and biomedical questions. Science 336:976–977

Peters J, Bhlmann P (2013) Identifiability of Gaussian structural equation models with equal error variances. Biometrika 101(1):219–228

Peters J, Mooij JM, Janzing D, Schölkopf B (2014) Causal discovery with continuous additive noise models. Journal of Machine Learning Research 15(58):2009–2053

Robinson RW (1977) Counting unlabeled acyclic digraphs. In: Combinatorial Mathematics V, pp 28–43

Sachs K, Perez O, Pe'er D, Lauffenburger DA, Nolan GP (2005) Causal protein-signaling networks derived from multiparameter single-cell data. Science 308(5721):523–529

Scutari M (2010) Learning bayesian networks with the bnlearn r package. Journal of Statistical Software, Articles 35(3):1–22

Shimizu S, Hoyer P, Hyvärinen A, Kerminen AJ (2006) A linear non-gaussian acyclic model for causal discovery. J Mach Learn Res 7:2003–2030

Shimizu S, Inazumi T, Sogawa Y, Hyvarinen A, Kawahara Y, Washio T, Hoyer P, Bollen K (2011) Directlingam: A direct method for learning a linear non-gaussian structural equation model. Journal of Machine Learning Research 12

Spirtes P, Glymour C, Scheines R (2001) Causation, Prediction, and Search, 2nd Edition, MIT Press Books, vol 1

Suwan S, Lee DS, Tang R, Sussman DL, Tang M, Priebe CE (2016) Empirical bayes estimation for the stochastic blockmodel. Electron J Statist 10(1):761–782

Tang X, Yang CC (2014) Detecting social media hidden communities using dynamic stochastic blockmodel with temporal dirichlet process. ACM Trans Intell Syst Technol 5(2)

Tang X, Huang Y, Lei J, Luo H, Zhu X (2019) The single-cell sequencing: New developments and medical applications. Cell & Bioscience 9

Tarka P (2018) An overview of structural equation modeling: its beginnings, historical development, usefulness and controversies in the social sciences. Quality & Quantity 52:313 – 354

Tennant PWG, Murray EJ, Arnold KF, Berrie L, Fox MP, Gadd SC, Harrison WJ, Keeble C, Ranker LR, Textor J, Tomova GD, Gilthorpe MS, Ellison GTH (2020) Use of directed acyclic graphs (dags) to identify confounders in applied health research: review and recommendations. International Journal of Epidemiology 50(2):620–632

Tsamardinos I, Brown L, Aliferis C (2006) The max-min hill-climbing bayesian network structure learning algorithm. Machine Learning 65:31–78

Veitch V, Roy D (2015) The class of random graphs arising from exchangeable random measures

Velikova M, van Scheltinga JT, Lucas PJ, Spaanderman M (2014) Exploiting causal functional relationships in bayesian network modelling for personalised healthcare. International Journal of Approximate Reasoning 55(1, Part 1):59–73

Verma T, Pearl J (1990a) Causal networks: Semantics and expressiveness. In: Uncertainty in Artificial Intelligence, vol 9, pp 69–76

Verma T, Pearl J (1990b) Equivalence and synthesis of causal models. In: Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence, UAI '90, p 255270

Wang B, Zhou Q (2021) Causal network learning with non-invertible functional relationships. Computational Statistics & Data Analysis 156:107141

Wen L, Tang F (2018) Boosting the power of single-cell analysis. Nat Biotechnol 36:408–409

Williams T, Bach C, Matthiesen N, Henriksen T, Gagliardi L (2018) Directed acyclic graphs: a tool for causal studies in pediatrics. Pediatric Research 84

Yan X, Jenson JE, Krzakala F, Moore C, Shalizi C, Zdeborova L, Zhang P, Zhu Y (2018) Model selection for degree-corrected block models

Young SJ, Scheinerman ER (2007) Random dot product graph models for social networks. In: Algorithms and Models for the Web-Graph, Springer Berlin Heidelberg, pp 138–149

Zanghi H, Ambroise C, Miele V (2008) Fast online graph clustering via erdös-rényi mixture. Pattern Recognition 41(12):3592 – 3599

Zhang K, Hyvärinen A (2009) On the identifiability of the post-nonlinear causal model. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, AUAI Press, p 647655

Zhang K, Wang Z, Zhang J, Schölkopf B (2015) On estimation of functional causal models: General results and application to the post-nonlinear causal model. ACM Trans Intell Syst Technol 7(2)