# Statistical Inference and Computational Methods for Large High-Dimensional Data with Network Structure

by

Sandipan Roy

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2015

Doctoral Committee:

Associate Professor Yves Atchadé, Co-Chair
Professor George Michailidis, University of Florida, Co-Chair
Assistant Professor Rajesh Rao Nadakuditi
Professor Ji Zhu

THE UNIVERSITY OF MICHIGAN

ARTES · SCIENTIA · VERITAS

1817

# ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my thesis advisors Yves Atchadé and George Michailidis whose continuous guidance and support has been the greatest source of encouragement throughout these five years. Several discussions and meetings with them on various research topics not only enriched my knowledge but also gave me a broader perspective of research. Their insights have certainly helped me to zoom out and see the big picture in the context of a particular research problem. I have benefited a lot from their clarity of thought and creative intellect.

I would like to thank my committee members Ji Zhu and Raj Rao Nadakuditi. Some helpful discussion with Ji Zhu on various topics has helped me quite a bit. Some of his work were definitely a great source of motivation for my research work and the thesis in general. Some of Raj Rao Nadakuditi's paper on Random matrix theory were definitely a source of motivation for my personal interest in that topic.

Moreover, I would like to express my gratitude to Professor Xuming He for his wonderful course on Research Appreciations in Statistics that provided a great learning experience for a young researcher like me; I would also like to thank Professor Susan Murphy for her great insights and in general encouragement for doing research during my research assistantship in the first semester at the University of Michigan. My sincere thanks to our graduate chair Liza Levina for some helpful discussions and same to professor Ambuj Tewari and Long Ngyuen for some light-hearted discussions.

Thanks to all my class mates here in Michigan for being great friends and special thanks to my roommate Naveen and some other college friends like Parichoy, Sayantan, Soumalya for their continuing support and helpful discussion. I would also like to express my sincere gratitude to Bhramardi, Mousumidi and Anandada for their love, affection and continuous encouragements. They were an integral part of this five-year period. Finally, a big thank you to my parents, my brother and all my family members for their love and encouragements.

# TABLE OF CONTENTS

**CHAPTER**

# LIST OF FIGURES

**Figure**

# LIST OF TABLES

Table

# LIST OF ABBREVIATIONS

**BIC** Bayesian Information Criterion

**BLB** Bag of Little Bootstraps

**EM** Expectation-Maximization

**GMM** Gaussian Mixture Model

**MCEM** Monte Carlo EM

**MLE** Maximum-Likelihood Estimator

**MRF** Markov Random Field

**MSE** Mean-Squared Error

**NMI** Normalized Mutual Information

**OIR** Out-In-Ratio

**RMSE** Root Mean-Squared Error

**SBM** Stochastic Block Model

# ABSTRACT

Statistical Inference and Computational Methods for Large High-Dimensional Data
with Network Structure

by

Sandipan Roy

Chair: Yves Atchadé and George Michailidis

New technological advancements have allowed collection of datasets of large volume and different levels of complexity. Many of these datasets have an underlying network structure. Networks are capable of capturing dependence relationship among a group of entities and hence analyzing these datasets unearth the underlying structural dependence among the individuals. Examples include gene regulatory networks, understanding stock markets, protein-protein interaction within the cell, online social networks etc.

The thesis addresses two important aspects of large high-dimensional data with network structure. The first one focuses on a high-dimensional data with network structure that evolves over time. Examples of such data sets include time course gene expression data, voting records of legislative bodies etc. The main task is to estimate the change-point as well as the network structures prior and post it. The network structures are obtained by $l_1$-penalized optimization method and we establish a finite sample estimation error bound for the change-point in the high-dimensional regime. The other aspect that we examine is about parameter estimation in large heterogeneous data with network structure. Our primary goal is to develop efficient computational techniques based on random subsampling and parallelization to estimate the parameters. We provide an analysis of rate of decay of bias and variance of our parallel implementation with a single round of communication after every iteration. We further show two applications of our methodology in the case of Gaussian Mixture Model (GMM) and Stochastic Block Model (SBM).

The emphasis is placed on developing new theoretical techniques and computational tools for network problems and applying the corresponding methodology in many fields, including biomedical and social science research, where network modeling and analysis plays an exceedingly important role.

# CHAPTER 1

# Introduction

Modern era has seen a explosion in the amount of information available. One of the main challenges in front of the researchers is how to deal with the massive amounts of data that are being generated frequently. Due to new technological advancements large amounts of very high-dimensional or unstructured data are continuously produced and stored with much cheaper cost than they used to be. These large high-dimensional datasets have brought forward a number of theoretical and computational challenges for the researchers in statistics, mathematics, computer science and various other fields. “Big Data” promise new levels of scientific discovery and economic value.

Some of the common examples of “Big Data” occur in networks. Networks are a collection of individuals or entities with possible relationships (Friendship, protein-protein interaction, relationship between stock prices etc.). It is not unusual for network data to be large, dynamic, heterogeneous, noisy, incomplete or even unobservable. Examples of time-varying network structures are ubiquitous in the nature and the increasing availability of data sets that evolve over time has accentuated the need for developing models for time varying networks. Examples of such data sets include time course gene expression data, voting records of legislative bodies etc.

*Heterogeneity* is another salient feature of large high-dimensional data. Mixture models are a prime example of generating large heterogeneous data. Gaussian Mixture Model (GMM) is a powerful tool for data clustering (McLachlan and Peel (2004)). We see its applications in the context of large heterogeneous data. In the next couple of sections we describe briefly the challenges one faces in dealing with large high-dimensional data.

## 1.1 Challenges of Analyzing Large High-Dimensional Data

Recent years have seen a surge in the volume and the size of the data being processed and analyzed in different applications. Modern technology has enabled collection of massive amount of data such as high-throughput biological assay data, large-scale genomic sequencing data, climate data, website transaction logs, online social network data etc. Such a "Big Data" movement is driven by the fact that massive amounts of very high-dimensional or unstructured data are continuously produced and stored with much cheaper cost than they used to be. Many of these large datasets have an underlying network structure. Further datasets with network structure that evolve over time are more ubiquitous in nature. This has emphasized the need to come up with model that can handle dynamic network or datasets with network structure varying over time. The time varying aspect of large networks has accentuated the need for developing time-varying network models. *Heterogeneity* is another common feature in these large datasets. To do efficient statistical analysis with large heterogeneous data is a challenging problem in this paradigm. Scientific advances are becoming more and more data-driven and researchers will more and more think of themselves as consumers of data. The massive amounts of high-dimensional data bring both opportunities and new challenges to data analysis. Hence valid statistical analysis for this large amount of data is becoming increasingly important.

## 1.2 Statistical Challenges

Massive sample size and the high-dimensionality of the dataset introduce unique statistical challenges that one encounters in "Big Data" regime. We need new statistical techniques to handle theoretical issues we might face in developing the methodology for dealing with large high-dimensional data. For example, many traditional methods that perform well for moderate sample sizes do not scale to massive data. Similarly, many statistical methods that perform well for low-dimensional data are facing significant challenges in analyzing high-dimensional data. Variable selection, dimension reduction are some of the common statistical problems one face with these "Big" data sets. As we discussed before many of these datasets have a time varying network structure and modeling the time varying aspect is another area of focus in the recent years. Kolar and Xing (2012); Kolar et al. (2010); Zhou et al. (2010) are some of the recent works on modeling dynamic network. One of the interesting features in a dataset with time evolving network structure is the locations where it undergoes changes in the structure viz. the *change-point*s. Investigating change-point in a high-dimensional network is a novel problem. Further theoretical analysis of consistency of the estimated change-point in high-dimensional regime is itself an intriguing problem.

There has been lot of research done in the statistics community for consistency of point estimates or even model selection consistency in high-dimensional regime but less work has been done so far in developing meaningful inference framework (confidence interval, hypothesis testing) for quantifying uncertainty. Hence the problem of constructing confidence interval or doing hypothesis testing for the change-point in this setting is still an open problem. The difficulty comes from the fact that the estimates of the underlying parameters in the model do not have a tractable limit distribution and the change-point estimate is typically a function of those parameters. We below describe a model that is widely used for relational structure over a fixed set of entities and can be used to model high-dimensional data with network structure.

**Markov Random Field**: A Markov Random field (MRF) (Wainwright and Jordan (2008)) is another term for a undirected graphical model. In physics, a field is an assignment of a physical quantity to points in space-time. For instance, a gravitational field is an assignment of a gravitational vector to points in space-time. Consider now a p-dimensional space, spanned by values of $p$ random variables instead of just the four of space and time. A random field is an assignment of a probability measure to points in the $p$-dimensional space. Just as a gravitational field describes a gravitational system, a random field describes a stochastic system. Thus a random field with a compact representation, and accessible inference procedures can be used as an interface layer for stochastic system applications.

Markov random fields use Markov assumptions to give compact representations for random fields. Let $G = (V, E)$ denote an undirected graph, with $V$ the set of nodes and $E$ the set of undirected edges. Let $X_i$ denote the variable associated with node i, for $i \in V$ ; giving a collated random vector $X = \{X1, ..., Xp\}$. The pairwise Markov property tells that

$$X_u \perp\!\!\!\perp X_v \mid X_{V\setminus\{u,v\}} \quad \text{if } \{u, v\} \notin E$$

. In the following figure $X_u$ and $X_v$ are independent given $X_w$ and $X_x$. We use high-dimensional MRF in Chapter 2 to model the underlying network structure for the given data.

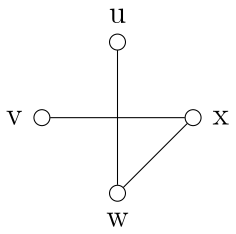


Figure 1.1: Example of Pairwise Markov Property

## 1.3 Computational Challenges

The current explosion in the size and amount of data available in statistical studies have motivated the development of new computational infrastructure and data-storage methods. Many standard algorithms for optimization that perform well on small datasets tend to perform inefficiently when applied to large datsets. Such a paradigm change has led to significant progresses on developments of fast algorithms that are scalable to massive data with high dimensionality. The study of some distributed and communication-efficient procedures for large scale optimization has come into forefront of dealing with the computational challenges posed by these large amount of data. Some of the recent works on distributed approaches to solving very large-scale statistical optimization problems are Nedic and Ozdaglar (2009); Ram et al. (2010); Johansson et al. (2009); Duchi et al. (2012); Dekel et al. (2012); Agarwal and Duchi (2011); Recht et al. (2011) etc. Many of these works depend on a common theme of splitting the original data into several small datasets, sending them to several machines to perform the optimization on those fractions of original data in a parallel manner and finally aggregating them via simple averaging. An interesting alternative to random splitting would be to use random subsampling which we focus on Chapter 3. Further, using a communication step among the machines after every iteration it is possible to show a bias reduction of the estimate relative to the parallel implementation without communication. Most of the parallelization schemes for solving large-scale optimization are certainly able to show the variance reduction relative to the serial implementation but the bias reduction of a parallel optimization procedure is a novel feature. We mention here a recent paper by Kleiner et al. (2014) that describes an automatic, accurate means of assessing estimator quality that is scalable to large dataset, known as Bag of Little Bootstraps (BLB). This work can also be seen as a way of constructing computationally efficient estimators in the “Big Data” setting.

As we have pointed out before *heterogeneity* is a common feature in large high-dimensional data. Those datasets are often created via aggregating many data sources corresponding to different subpopulations. Each subpopulation might exhibit some unique features not shared by others. To better illustrate this point we introduce two models that give rise to heterogeneous data.

**Gaussian Mixture Model**: A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system, such as vocal-tract related spectral features in a speaker recognition system. Although GMMs are widely used for clustering it is used for density estimation as well.

Let $\mathcal{D} = \{x_n, n = 1, 2, \ldots, N\}$ be $N$ iid observations obtained from a mixture model whose components are $d$-dimensional Gaussian distribution. The observations are assumed iid from the following model

$$p(x_n|\mu, \Sigma) = \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \tag{1.1}$$

where $f\left(x_n|\mu_i, \Sigma_i\right) = \frac{1}{(2\pi)^{\frac{m}{2}}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}\left(x_n - \mu_i\right)^T \Sigma_i^{-1}\left(x_n - \mu_i\right)\right]$. K is the number of mixture components. $\mu_i, \Sigma_i,\ i = 1, 2$ are the mean and the covariance matrix for the $i$th mixture component. $\pi_i$ is the mixing proportion for the $i$th component. The objective is to estimate the parameters $\{\pi_i, \mu_i, \Sigma_i\}_{i=1}^{K}$ of this mixture model in 3.39. The log-likelihood for the observed data is given by

$$\begin{aligned} l\left(\theta|\mathcal{D}\right) &= \sum_{n=1}^{N} \log p\left(x_n|\mu, \Sigma\right) \\ &= \sum_{n=1}^{N} \log \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \end{aligned} \tag{1.2}$$

GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm (See Dempster et al. (1977)). Typically in the context of a large high-dimensional data $N$ and $p$ would be both large and just applying the traditional EM for the entire dataset may not be feasible. Another source of heterogeneous data is the following

**Stochastic BlockModel**: Stochastic Blockmodels (SBM) are one of the prime examples of a latent variable model used for community detection in networks. Blockmodels and its stochastic versions have been popularly used for finding “Groups” or “Communities” in social networks. (See Lorrain and White (1971); Holland et al. (1983); Fienberg et al. (1985); Airoldi et al. (2008); Nowicki and Snijders (2001); Girvan and Newman (2002); Handcock et al. (2007)). The basic framework is the following: Let $A = (a_{ij})$ be the adjacency matrix of a network with n individuals. Suppose there are K groups. Assume $z_1, z_2, \ldots, z_n$ be the latent node labels for those n individuals. Let $P = (p_{ij})$ denote the link probability matrix of order $K \times K$. The probabilistic model is given by

$$\begin{aligned} A_{ij} &\overset{\text{ind}}{\sim} \text{Ber}\left(p_{z_i z_j}\right) \qquad (1.3) \\ z &\overset{\text{def}}{=} (z_1, \ldots, z_n) \sim \text{Mult}\,(\pi) \end{aligned}$$

where $\pi = (\pi_1, \ldots, \pi_K)$ are the class probabilities for the K-groups (classes). The main challenge comes in computing the parameter estimates in a SBM since the likelihood involving the latent membership of the nodes is not in general tractable. Monte Carlo EM (MCEM) may be employed for parameter estimation in SBM but with large $n$ each iteration of MCEM requiring $O(n^2)$ update, the algorithm becomes computationally infeasible.

## 1.4 Contributions of the Thesis

In this thesis, we address two specific aspects of large high-dimensional data. The two aspects are the following- (1) a change-point estimation problem arising from a large high dimensional data evolving over time and (2) a computational problem involving a communication-efficient algorithm for statistical optimization in large scale data.

In Chapter 2 we investigate a change-point estimation problem in the context of a high-dimensional MRF. Change-point estimation has a long history in the statistics literature (see Bai (1997), Carlstein (1988), Hinkley (1970), Loader et al. (1996), Lan et al. (2009) etc.) but its use in the context of a high-dimensional time evolving network is novel and supported by a Senate voting network data. Further we established a tight bound for the estimate, up to a logarithmic factor, even in settings where the number of possible edges in the network far exceeds the sample size. The technical details require a careful handling of model misspecification in Markov random fields

(Atchade (2014)), a novel aspect not present when estimating a single MRF from independent and identically distributed observations. The methodology we developed is also useful in other areas such as change-point estimation in a gene regulatory network or in a financial network that may undergo a significant change for some major economic announcements.

In Chapter 3 we propose a novel parallel optimization algorithm for large heterogeneous data. The algorithm is based on random subsampling and a single round of communication after every iteration of the optimization routine. The algorithm offers a fast computation of estimates of the model parameters relative to the serial implementation in a single machine with the entire data. Most of the existing parallel/distributed algorithms(Zinkevich et al. (2010); Zhang et al. (2013) etc.) ensure variance reduction of the final estimate relative to a serial implementation but does not reduce the bias of the estimate. Our parallel algorithm involves a communication step that results in a bias reduction of the final estimate relative to the parallel implementation without communication. We provide a sharp analysis on the rate of decay of bias and variance of our parallel scheme and compare it with a non-communication parallel scheme. The performance of the proposed algorithm is evaluated on large high-dimensional datasets generated from a GMM.

In Chapter 4 we examine the likelihood based inference in stochastic blockmodels for large network data. We focus on parameter estimation in stochastic blockmodels with covariate. Usual EM type algorithms do not scale well to large networks. Amini et al. (2013) developed a fast algorithm based on pseudo-likelihood approximation for community detection in large sparse networks. But in presence of covariate values in a blockmodel such approximation is hard to obtain. We present a computational algorithm based on case-control approximations of the likelihood (Raftery et al. (2012)) along with a parallel implementation of the Monte Carlo EM (MCEM) via the technique developed in Chapter 3. The performance of our algorithm is validated on synthetic datasets generated from large stochastic blockmodels with covariates. Further, to illustrate the performance of our methodology, we use a publicly available social network dataset that focuses on Facebook profiles of students in US colleges and Universities at a single point of time.

## CHAPTER 2

# Change-point Estimation in High-dimensional Markov Random Fields

## 2.1 Introduction

This chapter describes a change-point estimation problem in the context of high-dimensional Markov Random Field (MRF) models. Change-points represent a key feature in many dynamically evolving network structures. The change-point estimate is obtained by maximizing a profile penalized pseudo-likelihood function under a sparsity assumption. We also derive a tight bound for the estimate, up to a logarithmic factor, even in settings where the number of possible edges in the network far exceeds the sample size. The performance of the proposed estimator is evaluated on synthetic data sets and is also used to explore voting patterns in the US Senate in the 1979-2012 period.

## 2.2 Literature Review and Modeling Framework

Networks are capable of capturing dependence relationships and have been extensively employed in diverse scientific fields including biology, economics and the social sciences. A rich literature has been developed for static networks leveraging advances in estimating sparse graphical models. However, increasing availability of data sets that evolve over time has accentuated the need for developing models for time varying networks. Examples of such data sets include time course gene expression data, voting records of legislative bodies, etc.

In this work, we consider modeling the underlying network through a MRF that exhibits a change in its structure at some point in time. Specifically, suppose we have

$T$ observations $\left\{X^{(t)}, 1 \leq t \leq T\right\}$ over $p$-variables with $X^{(t)} = \left(X_1^{(t)}, \ldots, X_p^{(t)}\right)$ and $X_j^{(t)} \in \mathsf{X}$, for some finite set $\mathsf{X}$. Further, we assume that there exists a time point $1 \leq \tau_\star < T$ such that $\left\{X^{(t)}, 1 \leq t \leq \tau_\star\right\}$ is an independent and identically distributed sequence from a distribution $g_{\theta_\star^{(1)}}(\cdot)$ parametrized by a real symmetric matrix $\theta_\star^{(1)}$, while the remaining observations $\left\{X^{(t)}, \tau_\star + 1 \leq t \leq T\right\}$ forms also an independent and identically distributed sequence from a distribution $g_{\theta_\star^{(2)}}(\cdot)$ parametrized by another real symmetric matrix $\theta_\star^{(2)}$. We assume that the two distributions $g_{\theta_\star^{(1)}}(\cdot)$, $g_{\theta_\star^{(2)}}(\cdot)$ belong to a parametric family of MRF distributions given by

$$g_\theta(x) = \frac{1}{Z\left(\theta\right)} \exp\left(\sum_{j=1}^{p} \theta_{jj} B_0(x_j) + \sum_{1 \leq k < j \leq p} \theta_{jk} B(x_j, x_k)\right), \quad x \in \mathsf{X}^p, \tag{2.1}$$

for a function $B_0 : \mathsf{X} \to \mathbb{R}$, and a symmetric function $B : \mathsf{X} \times \mathsf{X} \to \mathbb{R}$ which encodes the interactions between the nodes. The term $Z\left(\theta\right)$ is the corresponding normalizing constant. Thus, the observations over time come from a MRF that exhibits a change in its structure at time $\tau_\star$ and the matrices $\theta_\star^{(1)}$ and $\theta_\star^{(2)}$ encode the dependence between the $p$ random variables respectively before and after the change-point.

The objective is to estimate the change-point $\tau_\star$, as well as the network structures $\theta_\star^{(1)}$ and $\theta_\star^{(2)}$. Although the problem of identifying a change point has a long history in statistics (see Bai (1997), Carlstein (1988), Hinkley (1970), Loader et al. (1996), Lan et al. (2009), Muller (1992), Raimondo (1998) and references therein), its use in a high-dimensional network problem is novel and motivated by the US Senate voting record application discussed in Section 2.7 Note that in a low-dimensional setting, the results obtained for the change-point depend on the regime considered; specifically, if there is a fixed shift then the asymptotic distribution of the change-point is given by the minimizer of a compound Poisson process (see Kosorok (2007)), while if the shift decreases to 0 as a function of the sample size, the distribution corresponds to that of Brownian motion with triangular drift (see Bhattacharya (1987), Muller (1992)).

Note that the methodology developed in this paper is useful in other areas, where similar problems occur. Examples include biological settings, where a gene regulatory network may exhibit a significant change at a particular dose of a drug treatment, or in finance where major economic announcements may disrupt financial networks.

Estimation of time invariant networks from independent and identically distributed data based on the MRF model has been a very active research area (see e.g. Banerjee et al. (2008); Höfling and Tibshirani (2009); Ravikumar et al. (2010); Xue et al.

(2012); Guo et al. (2010) and references therein). Sparsity (an often realistic assumption) plays an important role in this literature, and allows the recovery of the underlying network with relatively few observations (Ravikumar et al. (2010); Guo et al. (2010)).

On the other hand, there is significant less work on time varying networks (see Zhou et al. (2010), Kolar et al. (2010), Kolar and Xing (2012) etc.). The closest setting to our work is the one in Kolar and Xing (2012), which considers Gaussian graphical models where *each* node can exhibit multiple change points. In contrast, this paper focuses on a *single* change-point impacting the global network structure of the underlying Markov random field. In general, which setting is more appropriate depends on the application. In biological applications where the focus is on particular biomolecules (e.g. genes, proteins, metabolites), nodewise change-point analysis would typically be preferred, whereas is many social network applications (such as the political network example considered below), global structural changes in the network are of primary interest. Further, note that node-level changes detected at multiple nodes can be inconsistent, noisy and difficult to reconcile to extract global structural changes.

Another key difference with their work is the modeling framework employed. Specifically, in Kolar and Xing (2012) the number of nodes in the Gaussian graphical model is *fixed* and *smaller* than the available sample size. The high-dimensional challenge comes from the possible presence of multiple change-points per node, which leads to a large number of parameters to be estimated. To overcome this issue, a total variation penalty is introduced, a strategy that has worked well in regression modeling where the number of parameters is the same as the number of observations. On the other hand, this paper assumes a high-dimensional framework where the number of nodes (and hence the number of parameters of interest, namely the edges) grow with the number of time points and focuses on estimating a single change-point in a general MRF model.

To avoid the intractable normalizing constant issue in estimating the network structures, we employ a pseudo-likelihood framework. As customary in the analysis of change-point problems (Bai (1997); Lan et al. (2009)), we employ a profile pseudo-likelihood function to obtain the estimate $\hat{\tau}$ of the true change-point $\tau_{\star}$. Under a sparsity assumption, and some regularity conditions that allow the number of parameters $p(p+1)$ to be much larger than the sample size $T$, we establish that with high probability, $|\hat{\tau}-\tau_\star| = O(\log(pT))$, as $p \to \infty$. Note that in classical change-point problems with a fixed-magnitude change, it is well-known that the maximum likelihood estimator of the change-point satisfies $|\hat{\tau}-\tau_\star| = O_p(1)$ (see e.g. Bhattacharya (1987), Bai (1997)). The derivation of the result requires a careful handling of model misspecification in Markov random fields as explained in Section 2.4, a novel aspect not present when estimating a single MRF from independent and identically distributed observations. See also Atchade (2014) for another example of misspecification in Markov random fields. Further, to speed up the computation of the change-point estimator $\hat{\tau}$, we discuss a sampling strategy of the available observations, coupled with a smoothing procedure of the resulting likelihood function.

Last but not least, we employ the developed methodology to analyze the US Senate voting record from 1979 to 2012. In this application, each Senate seat represents a node of the network and the voting record of these 100 Senate seats on a given bill is viewed as a realization of an underlying MRF that captures dependencies between them. The analysis strongly points to the presence of a change-point around January, 1995, the beginning of the tenure of the 104th Congress. This change-point comes at the footsteps of the November 1994 election that witnessed the Republican Party capturing the US House of Representatives for the first time since 1956. Other analyses based on more ad hoc methods, also point to a significant change occurring after the November 1994 election (e.g. Moody and Mucha (2013)).

The remainder of the chapter is organized as follows. Modeling assumptions and the estimation framework are presented in Section 2.3, while Section 2.4 establishes the key technical results. Section 2.5 discusses computational issues and Section 2.6 evaluates the performance of the estimation procedure using synthetic data. Section 2.7 illustrates the procedure on the US Senate voting record. Finally, proofs are deferred to the Supplement.

## 2.3 Methodology

Let $\{X^{(t)},\ 1 \leq t \leq T\}$ be a sequence of independent random vector, where $X^{(t)} = (X_1^{(t)}, \ldots, X_p^{(t)})$ is a $p$-dimensional MRF whose $j$-th component $X_j^{(t)}$ takes values in a finite set $\mathsf{X}$. We assume that there exists a time point (change point) $\tau_\star \in \{1, \ldots, T-1\}$ and symmetric matrices $\theta_\star^{(1)}, \theta_\star^{(2)} \in \mathbb{R}^{p\times p}$, such that for all $x \in \mathsf{X}^p$,

$$\mathbb{P}\left(X^{(t)} = x\right) = g_{\theta_\star^{(1)}}(x), \quad \text{for } t = 1, \ldots, \tau_\star,$$

and

$$\mathbb{P}\left(X^{(t)} = x\right) = g_{\theta_\star^{(2)}}(x), \quad \text{for } t = \tau_\star + 1, \ldots, T,$$

where $g_\theta$ is the MRF distribution given in (2.1). The likelihood function of the observations $\{X^{(t)},\ 1 \leq t \leq T\}$ is then given by

$$\begin{aligned}
L_T\left(\tau, \theta^{(1)}, \theta^{(2)} | x^{1:T}\right) &= \prod_{t=1}^{\tau} g_{\theta^{(1)}}(x^{(t)}) \prod_{t=\tau+1}^{T} g_{\theta^{(2)}}(x^{(t)}) \\
&= \left(\frac{1}{Z(\theta^{(1)})}\right)^{\tau} \exp\left(\sum_{t=1}^{\tau}\sum_{j=1}^{p} \theta_{jj}^{(1)} B_0\left(x_j^{(t)}\right) + \sum_{t=1}^{\tau}\sum_{k\neq j} \theta_{jk}^{(1)} B\left(x_j^{(t)}, x_k^{(t)}\right)\right) \\
\times \left(\frac{1}{Z(\theta^{(2)})}\right)^{T-\tau} &\exp\left(\sum_{t=\tau+1}^{T}\sum_{j=1}^{p} \theta_{jj}^{(2)} B_0\left(x_j^{(t)}\right) + \sum_{t=\tau+1}^{T}\sum_{k\neq j} \theta_{jk}^{(2)} B\left(x_j^{(t)}, x_k^{(t)}\right)\right). \quad (2.2)
\end{aligned}$$

We write $\mathbb{E}$ to denote the expectation operator with respect to $\mathbb{P}$. For a symmetric matrix $\theta \in \mathbb{R}^{p\times p}$, we write $\mathbb{P}_\theta$ to denote the probability distribution on $\mathsf{X}^p$ with probability mass function $g_\theta$ and $\mathbb{E}_\theta$ its expectation operator.

We are interested in estimating both the change point $\tau_\star$, as well as the parameters $\theta_\star^{(1)}, \theta_\star^{(2)}$. Let $\mathcal{M}_p$ be the space of all $p \times p$ real symmetric matrices. We equip $\mathcal{M}_p$ with the Frobenius inner product $\langle\theta, \vartheta\rangle_{\mathsf{F}} \overset{\text{def}}{=} \sum_{k\leq j} \theta_{jk}\vartheta_{jk}$, and the associated norm $\|\theta\|_{\mathsf{F}} \overset{\text{def}}{=} \sqrt{\langle\theta,\theta\rangle}$. This is equivalent to identifying $\mathcal{M}_p$ with the Euclidean space $\mathbb{R}^{p(p+1)/2}$, and this identification prevails whenever we define gradients and Hessians of functions $f: \mathcal{M}_p \to \mathbb{R}$. For $\theta \in \mathcal{M}_p$ we also define $\|\theta\|_1 \overset{\text{def}}{=} \sum_{k\leq j} |\theta_{jk}|$, and $\|\theta\|_\infty \overset{\text{def}}{=} \sup_{k\leq j} |\theta_{jk}|$. If $u \in \mathbb{R}^d$, for some $d \geq 1$, and $A$ is an ordered subset of $\{1, \ldots, d\}$, we define $u_A \overset{\text{def}}{=} (u_j,\ j \in A)$, and $u_{-j}$ is a shortcut for $u_{\{1,\ldots,d\}\setminus\{j\}}$.

To avoid some of the computational difficulties in dealing with the normalizing constant of $g_\theta$, we take a pseudo-likelihood approach. For $\theta \in \mathcal{M}_p$ and $j \in \{1, 2, \ldots, p\}$, define $f_\theta^{(j)}(u|x) \overset{\text{def}}{=} \mathbb{P}_\theta(X_j = u | X_{-j} = x_{-j})$, for $u \in \mathsf{X}$, and $x \in \mathsf{X}^p$. From the expression of the joint distribution $g_\theta$ in (2.1), we have

$$f_\theta^{(j)}(u|x) = \frac{1}{Z_\theta^{(j)}(x)} \exp\left(\theta_{jj}B_0(u) + \sum_{k\neq j} \theta_{jk} B(u, x_k)\right),\ u \in \mathsf{X},\ x \in \mathsf{X}^p, \quad (2.3)$$

where

$$Z_\theta^{(j)}(x) \overset{\text{def}}{=} \int_{\mathsf{X}} \exp\left(\theta_{jj}B_0(z) + \sum_{k\neq j} \theta_{jk} B(z, x_k)\right) \mathrm{d}z. \quad (2.4)$$

The normalizing constant $Z_\theta^{(j)}(x)$ defined in (2.4) is actually a summation over $\mathsf{X}$, but for notational convenience we write it as an integral against the counting measure on $\mathsf{X}$. Next, we introduce

$$\phi(\theta, x) \overset{\text{def}}{=} -\sum_{j=1}^{p} \log f_\theta^{(j)}(x_j|x). \tag{2.5}$$

The negative log-pseudo-likelihood of the model (divided by $T$) is given by

$$\ell_T(\tau; \theta_1, \theta_2) \overset{\text{def}}{=} \frac{1}{T}\sum_{t=1}^{\tau} \phi(\theta_1, X^{(t)}) + \frac{1}{T}\sum_{t=(\tau+1)}^{T} \phi(\theta_2, X^{(t)}), \tag{2.6}$$

We propose to estimate the change point $\tau_\star$ using a profile pseudo-likelihood approach. More precisely our estimator $\hat{\tau}$ is defined as

$$\widehat{\tau} = \underset{\tau \in \mathcal{T}}{\mathsf{Argmin}}\ \ell_T(\tau; \widehat{\theta}_{1,\tau}, \widehat{\theta}_{2,\tau}), \tag{2.7}$$

for a search domain $\mathcal{T} \subset \{1, \ldots, T\}$ of the form $\{k_l, k_l + 1, \ldots, T - k_u\}$, where for each $\tau \in \mathcal{T}$, $\widehat{\theta}_{1,\tau}$ and $\widehat{\theta}_{2,\tau}$ are defined as

$$\widehat{\theta}_{1,\tau} \overset{\text{def}}{=} \underset{\theta \in \mathcal{M}_p}{\mathsf{Argmin}} \frac{1}{T}\sum_{t=1}^{\tau} \phi(\theta, X^{(t)}) + \lambda_{1,\tau}\|\theta\|_1,$$

and

$$\widehat{\theta}_{2,\tau} \overset{\text{def}}{=} \underset{\theta \in \mathcal{M}_p}{\mathsf{Argmin}} \frac{1}{T}\sum_{t=\tau+1}^{T} \phi(\theta, X^{(t)}) + \lambda_{2,\tau}\|\theta\|_1,$$

for some positive penalty parameters $\lambda_{1,\tau}, \lambda_{2,\tau}$. Since the network estimation errors at the boundaries of the time-line $\{1, \ldots, T\}$ are typically large, a restriction on the search domain is needed to guarantee the consistency of the method. This motivates the introduction of $\mathcal{T}$. We give more details on $\mathcal{T}$ below.

## 2.4 Theoretical Results

The recovery of $\tau_\star$ rests upon the ability of the estimators $\hat{\theta}_{j,\tau}$ to correctly estimate $\theta_\star^{(j)}$, $j \in \{1, 2\}$. Estimators for the static version of the problem where one has i.i.d. observations from a single MRF have been extensively studied; see Guo et al. (2010), Höfling and Tibshirani (2009), Meinshausen and Bühlmann (2006), Ravikumar et al. (2010) and references therein for computational and theoretical details. However, in the present setting one of the estimators $\hat{\theta}_{j,\tau}$, $j \in \{1,2\}$ is derived from a misspecified model. Hence, to establish the error bound for $\|\hat{\theta}_{j,\tau} - \theta_\star^{(j)}\|_2$, we borrow from the approach in Atchade (2014). For penalty terms $\lambda_{j,\tau}$ as in (2.8) and under some regularity assumptions, we derive a bound on the estimator errors $\|\hat{\theta}_{j,\tau} - \theta_\star^{(j)}\|_2$, for all $\tau \in \mathcal{T}$. We then use this result to show that the profile pseudo-log-likelihood estimator $\hat{\tau}$ is an approximate minimizer of $\tau \mapsto \ell_T(\tau; \theta_\star^{(1)}, \theta_\star^{(2)})$ and this allows us to establish a bound on the distance between $\hat{\tau}$ and the true change point $\tau_\star$.

We assume that the penalty parameters take the following specific form.

$$\lambda_{1,\tau} = \frac{16c_0\sqrt{\tau \log{(dT)}}}{T} \text{ and } \lambda_{2,\tau} = \frac{16c_0\sqrt{(T-\tau)\log{(dT)}}}{T}, \tag{2.8}$$

where $d \overset{\text{def}}{=} p(p+1)/2$, and

$$c_0 = \sup_{u,v \in \mathsf{X}} |B_0(u) - B_0(v)| \vee \sup_{x,u,v \in \mathsf{X}} |B(x,u) - B(x,v)|, \tag{2.9}$$

which serves as (an upper bound on the) standard deviation of the random variables $B_0(X)$, $B(X,Y)$. In practice, we use $\lambda_{1,\tau} = a_1 T^{-1} c_0 \sqrt{\tau \log(dT)}$, and $\lambda_{2,\tau} = a_2 T^{-1} c_0 \sqrt{(T-\tau)\log(dT)}$, where $a_1, a_2$ are chosen from the data by an analogue of the Bayesian Information Criterion (Schwarz et al. (1978)).

For $j = 1,2$, define $\mathcal{A}_j \overset{\text{def}}{=} \left\{1 \le k \le i \le p : \theta_{\star ik}^{(j)} \neq 0\right\}$, with $s_j = |\mathcal{A}_j|$ denoting the cardinality (and hence the sparsity) of the true model parameters. We also define

$$\mathbb{C}_j \overset{\text{def}}{=} \left\{\theta \in \mathcal{M}_p : \sum_{(k,i)\in\mathcal{A}_j^c} |\theta_{ik}^{(j)}| \le 3 \sum_{(k,i)\in\mathcal{A}_j} |\theta_{ik}^{(j)}|\right\}, \; j \in \{1,2\}, \tag{2.10}$$

used next in the definition of the restricted strong convexity assumption.

**Assumption 2.1.** [Restricted Strong Convexity] *For $j \in \{1,2\}$, and $X \sim g_{\theta_\star^{(j)}}$, there exists $\rho_j > 0$ such that for all $\Delta \in \mathbb{C}_j$,*

$$\sum_{i=1}^{p} \mathbb{E}_{\theta_\star^{(j)}}\left[\mathsf{Var}_{\theta_\star^{(j)}}\left(\sum_{k=1}^{p} \Delta_{ik} B_{ik}(X_i, X_k) | X_{-i}\right)\right] \ge 2\rho_j \|\Delta_{\mathcal{A}_j}\|_2^2, \tag{2.11}$$

*where $B_{ik}(x,y) = B_0(x)$ if $i = k$, and $B_{ik}(x,y) = B(x,y)$ if $i \neq k$.*

*Remark* 2.2. **Assumption** 2.1 is a restricted strong convexity assumption on the negative log-pseudo-likelihood function $\phi(\theta, x)$. This can be seen by noting that

(2.11) can also be written as

$$\Delta'\mathbb{E}\left[\nabla^{(2)}\phi(\theta_\star^{(j)}, X^{(j)})\right]\Delta \geq 2\rho_j\|\Delta_{\mathcal{A}_j}\|_2^2, \quad X^{(j)} \sim g_{\theta_\star^{(j)}}, \;\; \Delta \in \mathbb{C}_j, \;\; j \in \{1,2\}.$$

These restricted strong convexity assumptions of objective functions are more pertinent in high-dimensional problems and appear in one form or another in the analysis of high-dimensional statistical methods (see e.g. Neghaban et al. (2012) and references therein).

We impose the following condition on the change point and the sample size.

**Assumption 2.3.** [Sample size requirement] *We assume that there exists $\alpha_\star \in (0,1)$ such that $\tau_\star = \alpha_\star T$, and the sample size $T$ satisfies*

$$\min\left(\frac{T}{2^{11}\log(p)}, \frac{T}{48^2 \times 16^2 \log(dT)}\right) \geq c_0^2 \max\left(\frac{s_1^2}{\alpha_\star \rho_1^2}, \frac{s_2^2}{(1-\alpha_\star)\,\rho_2^2}\right),$$

*where $\rho_1$, and $\rho_2$ are as in* ***Assumption*** *2.1.*

*Remark* 2.4. Note that the constants $2^{11}$ and $48^2 \times 16^2$ required in **Assumption** 2.3 will typically yield a very conservative bound on the sample size $T$. We believe these large constants are mostly artifacts of our techniques, and can be improved. The key point of **Assumption** 2.3 is the fact that we require the sample $T$ to increase as a linear function of $\max(s_1^2, s_2^2)\log(p)$. This is in agreement with other results in high-dimensional sparse recovery.

The ability to detect the change-point requires that the change from $\theta_\star^{(1)}$ to $\theta_\star^{(2)}$ be identifiable. Define

$$\kappa_0 \stackrel{\text{def}}{=} \mathbb{E}_{\theta_\star^{(2)}}\left[\phi(\theta_\star^{(1)}, X) - \phi(\theta_\star^{(2)}, X)\right]. \tag{2.12}$$

**Assumption 2.5.** [Identifiability Condition] *Assume that $\theta_\star^{(1)} \neq \theta_\star^{(2)}$, and there also exists $\epsilon > 0$ that does not depend on $p, T$ such that*

$$\kappa_0 \geq \sqrt{\epsilon}\|\theta_*^{(2)} - \theta_*^{(1)}\|_1. \tag{2.13}$$

*Remark* 2.6. Obviously **Assumption** 2.5 is stronger than a mere identifiability condition $\kappa_0 > 0$. In the case where $\theta_\star^{(1)}$ and $\theta_\star^{(2)}$ have similar sparsity patterns, **Assumption** 2.5 can be shown to hold provided that most of the individual differences

$|\theta^{(2)}_{\star,ij} - \theta^{(1)}_{\star,ij}|$ are sufficiently large. To see this, notice that by a Taylor expansion one can show that

$$\kappa_0 \geq \frac{1}{2 + c_0 \|\theta^{(2)}_\star - \theta^{(1)}_\star\|_1} \sum_{i=1}^{p} \mathbb{E}_{\theta^{(2)}_\star} \left[ \mathsf{Var}_{\theta^{(2)}_\star} \left( \sum_{k=1}^{p} \left(\theta^{(1)}_{\star,ik} - \theta^{(2)}_{\star,ik}\right) B_{ik}(X_i, X_k) | X_{-i} \right) \right],$$

where $c_0$ is as in (2.9). Hence, if the restricted strong convexity assumption **Assumption** 2.1 holds and $\theta^{(1)}_\star$ and $\theta^{(2)}_\star$ have similar sparsity structures, in the sense that $\theta^{(1)}_\star - \theta^{(2)}_\star \in \mathbb{C}_2$, then using (2.11), we see that

$$\kappa_0 \geq \frac{2\rho_2 \|\theta^{(2)}_\star - \theta^{(1)}_\star\|_2^2}{2 + c_0 \|\theta^{(2)}_\star - \theta^{(1)}_\star\|_1}.$$

In this case (2.13) holds if the term $\|\theta^{(2)}_\star - \theta^{(1)}_\star\|_2 / \|\theta^{(2)}_\star - \theta^{(1)}_\star\|_1$ remains bounded away from zero as $p \to \infty$, which in turn holds true for instance if most of the differences $|\theta^{(2)}_{\star,ij} - \theta^{(1)}_{\star,ij}|$ are sufficiently large.

Finally, we define the search domain as the set

$$\mathcal{T} = \mathcal{T}_+ \cup \mathcal{T}_-, \tag{2.14}$$

where $\mathcal{T}_+$ is defined as the set of all time-points $\tau \in \{\tau_\star + 1, \ldots, T\}$ such that

$$c_0 b(\tau - \tau_\star) \leq 2\sqrt{\tau \log(dT)}, \quad \text{and} \quad 64 c_0^3 b s_1 (\tau - \tau_\star) \leq \rho_1 \tau, \tag{2.15}$$

and $\mathcal{T}_-$ is defined as the set of all time-point $\tau \in \{1, \ldots, \tau_\star\}$ such that

$$c_0 b(\tau_\star - \tau) \leq 2\sqrt{(T - \tau)\log(dT)}, \quad \text{and} \quad 64 c_0^3 b s_2 (\tau_\star - \tau) \leq \rho_2 (T - \tau), \tag{2.16}$$

where

$$b \overset{\text{def}}{=} \sup_{1 \leq j \leq p} \sum_{k=1}^{p} \left|\theta^{(2)}_{\star jk} - \theta^{(1)}_{\star jk}\right|. \tag{2.17}$$

*Remark* 2.7. Notice that $\mathcal{T}$ is of the form $\{k_l, k_l + 1, \ldots, \tau_\star, \tau_\star + 1, \ldots, T - k_u\}$, since for $\tau$ close to $\tau_\star$ both (2.15) and (2.16) hold provided that $T$ is large enough.

We can then establish the key result of this paper.

**Theorem 2.8.** *Under assumptions* ***Assumption*** *2.1-****Assumption*** *2.5, with $\alpha_\star$ as in* ***Assumption*** *2.3, for the model posited in (2.2), and for the estimator defined in*

*(2.7), we have that with probability tending to one as $p \to \infty$,*

$$\left|\frac{\widehat{\tau}}{T} - \alpha_\star\right| \lesssim \left(1 + \frac{c_0^2}{\epsilon} + \frac{M}{\kappa_0}\right) \frac{\log(dT)}{T}, \tag{2.18}$$

*where $M = \frac{s_1}{\rho_1}\left(1 + \frac{c_0^2 s_1}{\rho_1}\right) + \frac{s_2}{\rho_2}\left(1 + \frac{c_0^2 s_2}{\rho_2}\right)$, and the notation $a \lesssim b$ means that $a \le cb$ for some universal constant $c$.*

*Remark* 2.9. Theorem 2.8 gives a theoretical guarantee that even for large $p$ and for large enough sample size $T$, $\frac{1}{T}|\hat{\tau} - \tau_\star| = O(\frac{\log(pT)}{T})$ with high-probability. For fixed-parameter change-point problems, the maximum likelihood estimator of the change-point is known to satisfy $\frac{1}{T}|\hat{\tau} - \tau_\star| = O_P(\frac{1}{T})$ (see e. g. Bai (1997)).

Another nice feature of Theorem 2.8 is the fact that the constant $\left(1 + \frac{c_0^2}{\epsilon} + \frac{M}{\kappa_0}\right)$ describes the behavior of the change-point estimator as a function of the key parameters of the problem. In particular, the bound in (2.18) shows that the change-point estimator improves as $s_1, s_2$ (the number of non-zero entries of the matrices $\theta_\star^{(1)}, \theta_\star^{(2)}$ resp.), or $c_0$ (the maximum fluctuation of $B_0$ and $B$) decrease. The estimator also improves as the identifiability parameter $\kappa_0$ increases.

## 2.5 Algorithm and Implementation Issues

The key steps of the algorithm to compute the estimates $\left(\hat{\tau}, \hat{\theta}_{1,\hat{\tau}}, \hat{\theta}_{2,\hat{\tau}}\right)$ based on a sequence of observed $p$-dimensional vectors $\{x^{(t)}, 1 \le t \le T\}$ are described in the following algorithm.

**Algorithm 2.10** (**Basic Algorithm**). *Input: a sequence of observed $p$-dimensional vectors $\{x^{(t)}, 1 \le t \le T\}$, and $\mathcal{T} \subseteq \{1, \ldots, T\}$ the search domain.*

1. *For each $\tau \in \mathcal{T}$, estimate $\hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}$ using for instance the algorithm in Höfling and Tibshirani (2009) to obtain sparse estimates of the underlying network structures.*

2. *For each $\tau \in \mathcal{T}$, plug-in the estimates $\hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}$ in (2.6) and obtain the profile (negative) pseudo-log-likelihood function $\mathcal{P}\ell(\tau) \overset{\text{def}}{=} \ell_T(\tau; \hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau})$.*

3. *Identify $\hat{\tau}$ that achieves the minimum of $\mathcal{P}\ell(\tau)$ over the grid $\mathcal{T}$, and use $\hat{\theta}_{1,\hat{\tau}}, \hat{\theta}_{2,\hat{\tau}}$ as the estimates of $\theta_\star^{(1)}$ and $\theta_\star^{(2)}$, respectively.*

In our implementation of the Basic Algorithm, we choose the set $\mathcal{T}$ in such a way that we could avoid the large estimation errors at the boundaries. More specifically, we choose the search domain as $\mathcal{T} = \{k_l, k_l + 1, \ldots, T - k_l\}$ where $k_l$ is much larger than 1. Thus we ensure that the errors of estimation remain small by staying sufficiently away from both boundaries. For example, for a particular implementation with $T = 700$, we choose $k_l = 60$ as described in detail in Section 2.6.

Note that to identify the change-point $\hat{\tau}$ the algorithm requires a *full scan* of all the time points in the set $\mathcal{T}$, which can be expensive in the presence of a large number of them. To that end, we discuss a fast implementation that operates in two stages. In the first stage, a coarser grid $\mathcal{T}_1 \subset \mathcal{T}$ of time points is used and steps (a) and (b) of the Basic Algorithm are used to obtain $\ell_T(\tau; \hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}), \tau \in \mathcal{T}_1$. Subsequently, the profile likelihood function $\ell_T$ is smoothed using a Nadaraya-Watson kernel (Nadaraya (1965)). Based on this smoothed version of the profile likelihood, an initial estimate of the change-point is obtained. In the second stage, a new fine-resolution grid $\mathcal{T}_2$ is formed around the first stage estimate of $\hat{\tau}$. Then, the Basic Algorithm is used for the grid points in $\mathcal{T}_2$ to obtain the final estimate. This leads to a more practical algorithm summarized next.

**Algorithm 2.11** (**Fast Implementation Algorithm**). *Input: a sequence of observed p-dimensional vectors $\{x^{(t)}, 1 \le t \le T\}$, and $\mathcal{T} \subseteq \{1, \ldots, T\}$ the search domain.*

1. *Find a coarser grid $\mathcal{T}_1$ of time points.*
2. *For each $\tau \in \mathcal{T}_1$, use steps (a) and (b) of the Basic Algorithm to obtain $\mathcal{P}\ell_T(\tau), \quad \tau \in \mathcal{T}_1$.*
3. *Compute the profile negative pseudo-log-likelihood over the interval $[1, T]$ by Nadaraya-Watson kernel smoothing:*

$$\widetilde{\mathcal{P}\ell_{1s}}(\tau) \overset{\text{def}}{=} \frac{\sum_{\tau_i \in \mathcal{T}_1} K_{h_\nu}(\tau, \tau_i)\, \ell(\tau_i; \widehat{\theta}_{1,\tau_i}, \widehat{\theta}_{2,\tau_i})}{\sum_{\tau_i \in \mathcal{T}_1} \ell\left(\tau_i; \widehat{\theta}_{1,\tau_i}, \widehat{\theta}_{2,\tau_i}\right)}, \; 1 \le \tau \le T.$$

*The first stage change-point estimate is then obtained as*

$$\widehat{\tau} = \underset{1<\tau<T}{\mathit{Argmin}}\, \widetilde{\mathcal{P}\ell_{1s}}(\tau).$$

4. *Form a second stage grid $\mathcal{T}_2$ around the first stage estimate $\hat{\tau}$ and for each $\tau \in \mathcal{T}_2$, estimate $\widehat{\widehat{\theta}}_{1,\tau}$ and $\widehat{\widehat{\theta}}_{2,\tau}$ using steps (a) and (b) of the Basic Algorithm.*

5. *Construct the second stage smoothed profile pseudo-likelihood*

$$\widetilde{\mathcal{P}\ell_{2s}}(\tau) \overset{\text{def}}{=} \frac{\sum_{\tau_i \in \mathcal{T}_2} K_{h_\nu}(\tau, \tau_i)\, \ell\left(\tau_i; \widehat{\widehat{\theta}}_{1,\tau_i}, \widehat{\widehat{\theta}}_{2,\tau_i}\right)}{\sum_{\tau_i \in \mathcal{T}_2} \ell\left(\tau_i; \widehat{\widehat{\theta}}_{1,\tau_i}, \widehat{\widehat{\theta}}_{2,\tau_i}\right)}, \ \min(\mathcal{T}_2) \leq \tau \leq \max(\mathcal{T}_2).$$

*The final change-point estimate is then given by*

$$\widehat{\widehat{\tau}} = \underset{\min(\mathcal{T}_2) \leq \tau \leq \max(\mathcal{T}_2)}{\mathit{Argmin}} \ \widetilde{\mathcal{P}\ell_{2s}}(\tau).$$

## 2.6 Performance Assessment

### 2.6.1 Comparing Algorithm 1 and Algorithm 2

We start by examining the relative performance of both the Basic (Algorithm 1) and the Fast Implementation Algorithms (Algorithm 2). We use the so called Ising model; i.e. when (2.1) has $B_0(x_j) = x_j$, $B(x_j, x_k) = x_j x_k$ and $\mathsf{X} \equiv \{0, 1\}$. In all simulation setting the sample size is set to $T = 700$, and the true change-point is at $\tau_\star = 350$, while the network size $p$ varies from 40-100. All the simulation results reported below are based on 30 replications of Algorithm 1 and Algorithm 2.

The data are generated as follows. We first generate two $p \times p$ symmetric adjacency matrices each having density 10%; i.e. only ∼10% of the entries are different than zero. Each off-diagonal element of $\theta^{(i)}_{\star jk}$, $(i = 1, 2)$ is drawn uniformly from $[-1, -0.5] \cup [0.5, 1]$ if there is an edge between nodes $j$ and $k$, otherwise $\theta^{(i)}_{\star jk} = 0$. All the diagonal entries are set to zero. Given the two matrices $\theta^{(1)}_\star$ and $\theta^{(2)}_\star$, we generate the data $\left\{X^{(t)}\right\}_{t=1}^{\tau_\star} \overset{\text{iid}}{\sim} g_{\theta^{(1)}_*}$ and $\left\{X^{(t)}\right\}_{t=\tau_\star+1}^{T} \overset{\text{iid}}{\sim} g_{\theta^{(2)}_*}$ by Gibbs sampling.

Different "signal strenghts" are considered, by setting the degree of similarity between $\theta^{(1)}_\star$ and $\theta^{(2)}_\star$ to 0%, 20% and 40%. The degree of similarity is the proportion of equal off-diagonal elements between $\theta^{(1)}_\star$ and $\theta^{(2)}_\star$. Thus, the difference $\|\theta^{(2)}_\star - \theta^{(1)}_\star\|_1$ becomes smaller for higher degree of similarity and as can be seen from Assumption H3, the estimation problem becomes harder in such cases.

The choice of the tuning parameters $\lambda_{1,\tau}$ and $\lambda_{2,\tau}$ were made based on Bayesian Information Criterion (BIC) where we search $\lambda_{1,\tau}$ and $\lambda_{2,\tau}$ over a grid $\Lambda$ and for each penalty parameter the $\lambda$ value that minimizes the BIC score (defined below) over $\Lambda$ is selected. If we define $\lambda_1^{BIC}$ and $\lambda_2^{BIC}$ as the selected $\lambda$ values for $\lambda_1$ and $\lambda_2$ by BIC we have

$$\lambda_1^{BIC} = \underset{\lambda_1 \in \Lambda}{\mathsf{Argmin}} -2 \times \frac{1}{T} \sum_{t=1}^{\tau} \phi\left(\theta_1, X^{(t)}\right) + \log(\tau)|\widehat{\mathcal{A}}_1| \text{ and}$$

$$\lambda_2^{BIC} = \underset{\lambda_2 \in \Lambda}{\mathsf{Argmin}} -2 \times \frac{1}{T} \sum_{t=\tau+1}^{T} \phi\left(\theta_2, X^{(t)}\right) + \log(T-\tau)|\widehat{\mathcal{A}}_2|$$

where $\widehat{\mathcal{A}}_i = \left\{(j,k) : \hat{\theta}_{jk}^{(i)} \neq 0, j < k\right\}$, $i = 1, 2$.

For the fast algorithm (Algorithm 2), the first stage grid employed had a step size of 10 and ranged from 60 to 640, while the second stage grid was chosen in the interval $[\hat{\tau} - 30, \hat{\tau} + 30]$ with a step-size of 3.

We present the results for Algorithm 1 in Table 2.1 for the case $p = 40$. It can be seen that Algorithm 1 performs very well for stronger signals (0% and 20% similarity), while there is a small degradation for the 40% similarity setting. The results on the specificity, sensitivity and the relative error of the estimated network structures are given in Table 2.2. Specificity is defined as the proportion of true negatives and can also be interpretated as (1-Type 1 error). On the other hand sensitivity is the proportion of true positives and can be interpreted as the power of the method. The results for Algorithm 2 for $p = 40, 60$ and $p = 100$, for the change-point estimates are given in Table 2.4, while the specificity, sensitivity and relative error of the estimated network structures are given in Table 2.5. These results show that Algorithm 2 has about 20% higher mean-squared error (MSE) compared to Algorithm 1. However as pointed out in Section 2.5, Algorithm 2 is significantly faster. In fact in this particular simulation setting, Algorithm 2 is almost 5 times faster in a standard computing environment with 4 CPU cores. See also the results in Table 2.3 which reports the ratio of the run-time of a single iteration of Algorithm 1 and Algorithm 2.

Further, selected plots of the profile smoothed pseudo-log-likelihood functions $\widetilde{\mathcal{P}\ell_{1s}}(\tau)$ and $\widetilde{\mathcal{P}\ell_{2s}}(\tau)$ from the first and second stage of Algorithm 2 are given in Figure 2.1.

Table 2.1: Change-point estimation results using the Basic Algorithm, for different percentages of similarity.

| $p$ | % of Similarity | $\widehat{\tau}$ | RMSE | CV |
|---|---|---|---|---|
| 40 | 0 | 355 | 14.77 | 0.03 |
| | 20 | 362 | 24.65 | 0.06 |
| | 40 | 375 | 38.49 | 0.08 |

Table 2.2: Specificity, sensitivity and relative error in estimating $\theta_\star^{(1)}$ and $\theta_\star^{(2)}$ from the Basic Algorithm, with different percentages of similarity.

| $p$ | % of Similarity | Specificity | | Sensitivity | | Relative error | |
|---|---|---|---|---|---|---|---|
| | | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ |
| 40 | 0 | 0.78 | 0.87 | 0.79 | 0.89 | 0.70 | 0.63 |
| | 20 | 0.74 | 0.88 | 0.80 | 0.88 | 0.72 | 0.67 |
| | 40 | 0.71 | 0.80 | 0.77 | 0.81 | 0.75 | 0.72 |

Table 2.3: Ratio of the computing time of one iteration of Algorithm 1 and Algorithm 2.

| $p$ | Ratio of computing times |
|---|---|
| 40 | 4.93 |
| 60 | 4.82 |
| 100 | 4.81 |

Table 2.4: Change-point Estimation Results for different values of $p$ and different percentages of similarity for the Fast Implementation Algorithm.($T = 700$, $s_1 = s_2 = \frac{10p(p+1)}{2}\%$, $\tau^* = 354$)

| p | % of Similarity | $\widehat{\tau}$ | $\widehat{\widehat{\tau}}$ | RMSE | CV |
|---|---|---|---|---|---|
| 40 | 0 | 360 | 360 | 17.89 | 0.04 |
| | 20 | 363 | 361 | 30.07 | 0.08 |
| | 40 | 375 | 373 | 47.97 | 0.10 |
| 60 | 0 | 357 | 356 | 23.05 | 0.06 |
| | 20 | 388 | 386 | 43.20 | 0.08 |
| | 40 | 410 | 408 | 61.45 | 0.09 |
| 100 | 0 | 356 | 355 | 35.93 | 0.10 |
| | 20 | 408 | 401 | 62.89 | 0.10 |
| | 40 | 424 | 421 | 85.04 | 0.12 |

Table 2.5: Specificity, sensitivity and relative error of the two parameters for different values of $p$ and different percentages of similarity for the Fast Implementation Algorithm.

| p | % of Similarity | Specificity | | Sensitivity | | Relative error | |
|---|---|---|---|---|---|---|---|
| | | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ | $\theta_*^{(1)}$ | $\theta_*^{(2)}$ |
| 40 | 0 | 0.74 | 0.86 | 0.78 | 0.86 | 0.74 | 0.67 |
| | 20 | 0.74 | 0.81 | 0.76 | 0.82 | 0.73 | 0.71 |
| | 40 | 0.72 | 0.78 | 0.78 | 0.82 | 0.74 | 0.70 |
| 60 | 0 | 0.81 | 0.83 | 0.77 | 0.82 | 0.75 | 0.66 |
| | 20 | 0.82 | 0.87 | 0.70 | 0.72 | 0.79 | 0.73 |
| | 40 | 0.80 | 0.86 | 0.65 | 0.68 | 0.81 | 0.78 |
| 100 | 0 | 0.82 | 0.88 | 0.75 | 0.84 | 0.78 | 0.66 |
| | 20 | 0.81 | 0.87 | 0.66 | 0.70 | 0.81 | 0.78 |
| | 40 | 0.85 | 0.87 | 0.63 | 0.68 | 0.83 | 0.81 |

### 2.6.2 A community based network structure

Next, we examine a setting similar to the one that emerges from the US Senate analysis presented in the next Section. Specifically, there are two highly "connected" communities of size $p = 50$ that are more sparsely connected before the change-point, but exhibit fairly strong negative association between their members after the change-point. Further, the within community connections are increased for one of them and decreased for the other after the occurrence of the change-point. We keep

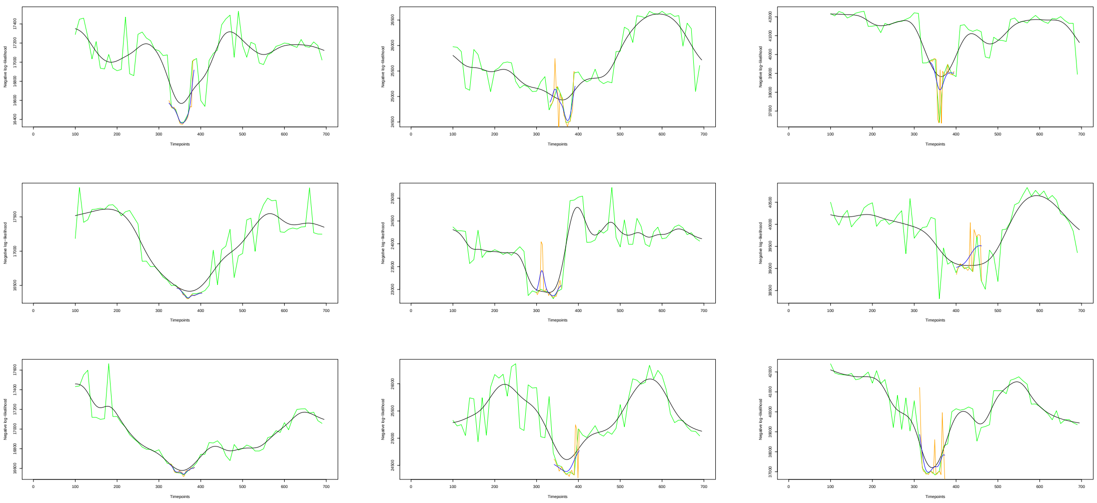


Figure 2.1: Smoothed profile pseudo-log-likelihood functions from one run of Algorithm 2. Different values of similarity (0%, 20% and 40%) in rows. Different values of $p$ ($p = 40, 60$ & $100$) in column. The green curve is the non-smoothed profile pseudo-log-likelihood from Stage 1 of Algorithm 2, and the black curve is its smoothed version. The orange and the blue curve are respectively the non-smoothed and the smoothed profile pseudo-log-likelihood functions from Stage 2 of Algorithm 2.

the density of the two matrices encoding the network structure before and after the true change-point at 10%. In the pre change-point regime, 40% of the non-zero entries are attributed to within group connections in community 1 (see Table 2.6), and 50% to community 2 (see Table 2.6), while the remaining 10% non-zeros represent between group connections and are negative. Note that the within group connections are all positive. In the post change-point regime, the community 1 within group connections slightly increase to 42% of the non-zero entries, whereas those of community 2 decrease to 17% of the non-zero entries. The between group connections increase to 41% of the non-zero entries in the post change-point regime. As before, each off-diagonal element $\theta^{(i)}_{jk}$, $i = 1, 2$ is drawn uniformly from $[-1, -0.5] \cup [0.5, 1]$ if nodes $j$ and $k$ are linked by an edge, otherwise $\theta^{(i)}_{*,jk} = 0$, $i = 1, 2$ and the diagonals for both the matrices are assigned as zeros. Given the two matrices $\theta^{(1)}_{*}$ and $\theta^{(2)}_{*}$, we generate data using the "BMN" package (Hoefling (2010)) as described earlier. The total sample size employed is $T = 1500$ and the true change-point is at $\tau^* = 750$. We choose the first stage grid comprising of 50 points with a step size of 27 and the second stage grid is chosen in a neighborhood of the first stage estimate with a step size of 3 with 20 points. We replicate the study 5 times and find that the estimated change-point averaged over the 5 replications as $\hat{\tau} = 768$. The relevant figure (see Figure 2.2) for this two community model is given below. The analysis indicates that our proposed methodology is able to estimate the true change-point sufficiently well in the presence of varying degrees of connections between two communities over two different time periods, a reassuring feature for the US Senate application presented next.

Table 2.6: Positive and negative edges before and after the true change-point for two community model

| Edges | Before | | | After | | |
|---|---|---|---|---|---|---|
| | comm 1 | comm 2 | between | comm 1 | comm 2 | between |
| positive | 50 | 63 | 0 | 52 | 21 | 0 |
| negative | 0 | 0 | 10 | 0 | 0 | 50 |
| Total | 50 | 63 | 10 | 52 | 21 | 50 |

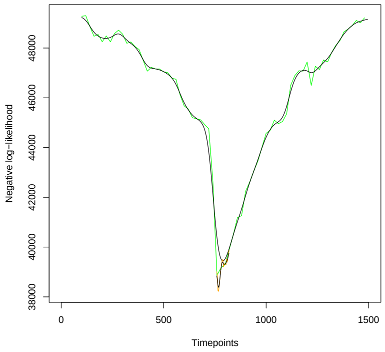


Figure 2.2: Change-point estimate for the two community model with $p = 50$, $T = 1500$ and $\tau^*$=754

## 2.7 Application to Roll Call Data of the US Senate

The data examined correspond to voting records of the US Senate covering the period 1979 (96th Congress) to 2012 (112th Congress) and were obtained from the website `www.voteview.com`. Specifically, for each of the 12129 votes cast during this period, the following information is recorded: the date that the vote occurred and the response to the bill/resolution under consideration -yes/no, or abstain- of the 100 Senate members. Due to the length of the time period under consideration, there was significant turnover of Senate members due to retirements, loss of re-election bids, appointments to cabinet or other administrative positions, or physical demise. In order to hold the number of nodes fixed to 100 (the membership size of the US Senate at any point in time), we considered Senate seats (e.g. Michigan 1 and Michigan 2) and carefully mapped the senators to their corresponding seats, thus creating a continuous record of the voting pattern of each Senate seat.

Note that a significant number of the 12129 votes deal with fairly mundane procedural matters, thus resulting in nearly unanimous outcomes. Hence, only votes exhibiting conformity less than 75% (yes/no) in either direction were retained, thus resulting in an effective sample size of $T = 7949$ votes. Further, missing values due to abstentions were imputed by the value (yes/no) of that member's party majority position on that particular vote. Note that other imputation methods of missing values were employed: (i) replacing all missing values by the value (yes/no) representing the winning majority on that bill and (ii) replacing the missing value of a Senator by the value that the majority of the opposite party voted on that particular bill. The results based on these two alternative imputation methods are given in the Supplement.

Finally, the yes/no votes were encoded as 1/0, respectively. Under the posited model, votes are considered as i.i.d. from the same underlying distribution pre and post any change-point. In reality, voting patterns are more complex and in all likelihood exhibit temporal dependence within the two year period that a Congress serves and probably even beyond that due to the slow turnover of Senate members. Nevertheless, the proposed model serves as a *working model* that captures essential features of the evolving voting dependency structure between Senate seats over time.

The likelihood function together with an estimate of a change-point are depicted in Figure 2.5 based on the Fast Implementation Algorithm presented in Section 2.5. We choose our first stage grid with a step-size of 50 that yields 157 points excluding time points close to both boundaries. In the second stage, we choose a finer-resolution grid with a step size of 20 in a neighborhood of the first stage change-point estimate. The vote corresponding to the change point occurred on January 17, 1995 at the beginning of the tenure of the 104th Congress. This change-point comes at the footsteps of the November 1994 election that witnessed the Republican Party capturing the US House of Representatives for the first time after 1956. As discussed in the political science literature, the 1994 election marked the end of the "Conservative Coalition", a bipartisan coalition of conservative oriented Republicans and Democrats on President Roosevelt's "New Deal" policies, which had often managed to control Congressional outcomes since the "New Deal" era. Note that other analyses based on fairly ad hoc methods (e.g. Moody and Mucha (2013)) also point to a significant change occurring after the November 1994 election.

Next, we examine more closely the pre and post change-point network structures, shown in the form of heatmaps of the adjacency matrices in Figure 2.6. To obtain stable estimates of the respective network structures, stability selection (Meinshausen and Bühlmann (2010)) was employed with edges retained if they were present in more than 90% of the 50 networks estimated from bootstrapped data. To aid interpretation, the 100 Senate seats were assigned to three categories: Democrat (blue), mixed (yellow) and Republican (red). Specifically, a seat was assigned to the Democrat or Republican categories if it were held for more than 70% of the time by the corresponding party within the pre or post change-point periods; otherwise, it was assigned to the mixed one. This means that if a seat was held for more than 5 out of the 8 Congresses in the pre change-point period and similarly 6 out of 9 Congresses in the post period by the Democrats, then it is assigned to that category and similarly for Republican assignments; otherwise, it is categorized as mixed.

In the depicted heatmaps, the ordering of the Senate seats in the pre and post change-point regimes are kept as similar as possible, since some of the seats changed their category membership completely across periods. Further, the green dots represent positive edge weights, mostly corresponding to within categories interactions, while black dots represent negative edge weights, mostly between category interactions. It can be clearly seen an emergence of a significant number of black dots in the post change-point regimes, indicative of sharper disagreements between political parties and thus increased polarization. Further, it can be seen that in the post change-point regime the mixed group becomes more prominent, indicating that it contributes to the emergence of a change-point.

To further explore the reasons behind the presence of a change-point, we provide some network statistics in Figure 2.3 and Figure 2.4. Specifically, the two figures present the proportion of positive and negative edges, before and after the estimated change-point using two different methods for selecting the penalty tuning parameters; an analogue of the Bayesian Information Criterion and threshold 0.8 for the stability selection method respectively. The patterns shown across the figures for the two different methods are very similar- high proportion of positive edges within groups and very low or almost negligible proportion of negative edges within the "republican" or "democrat" groups in both pre and post-change-point periods. Further, a large proportion of negative edges can be accounted for "republican" and "democrat" group interactions, which tend to increase in the post regime. One noticeable fact is that the proportion of positive edges within the "republican" and "democrat" groups remain almost same from pre to post change-point regime under BIC and stability selection both whereas the proportion of positive edges between the two groups decrease and the proportion of negative edges between them tend to increase from pre to post change-point regime for both the methods. It can also be observed that the "mixed" and the "democrat" groups exhibit a large proportion of positive edges between them in the pre regime, as gleaned from their overlap in the corresponding heatmap.

We also present some other network statistics, such as average degree, centrality scores and average clustering coefficients for the three groups "republican", "democrat" and "mixed" in Table 2.7. We observe that in terms of centrality scores the "democrat" group is more influential than the "republican" one, in both the pre and post change-point network structures, whereas in terms of clustering coefficient values the "republican" group is ahead of the "democrat" one and the gap increases from pre to post change-point regime, also reflected in the finding that the number of edges within the "republican" group mostly remains the same from pre to post regimes, whereas for the democrats it decreases. These results suggest that the Republicans form a tight cluster, whereas the Democrats not to the same extent.

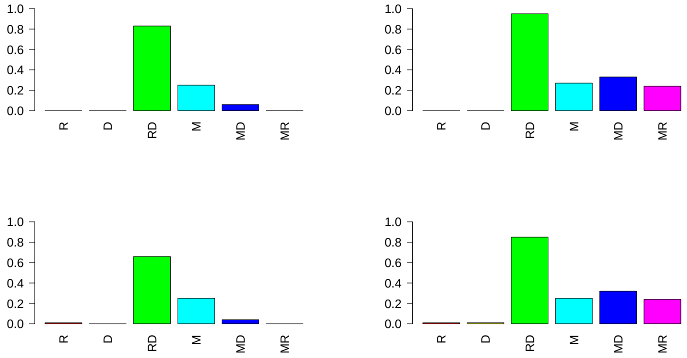


Figure 2.3: Proportion of negative edges for network structures before (left figure) and after (right figure) the estimated change-point for BIC and stability selection with threshold=0.8

Table 2.7: Different network statistic values for stability selection with threshold=0.9 and 0.8 respectively

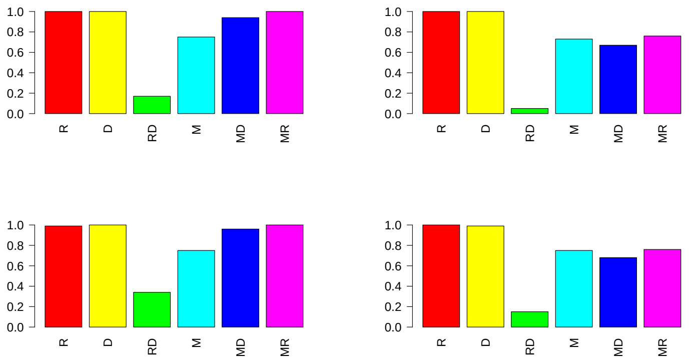


Figure 2.4: Proportion of positive edges for network structures before (left figure) and after (right figure) the estimated change-point for BIC and stability selection with threshold=0.8

| Methods | Network Statistic | Before | | | After | | |
|---|---|---|---|---|---|---|---|
| | | Rep | Dem | Mixed | Rep | Dem | Mixed |
| Stable (0.9) | Centrality Score | 0.004 | 0.368 | 0.054 | 0.001 | 0.483 | 0.034 |
| | Clustering Coefficient | 0.346 | 0.311 | 0.339 | 0.334 | 0.251 | 0.391 |
| | | | | | | | |
| Stable (0.8) | Centrality Score | 0.004 | 0.378 | 0.055 | 0.001 | 0.481 | 0.078 |
| | Clustering Coefficient | 0.366 | 0.371 | 0.360 | 0.378 | 0.307 | 0.364 |

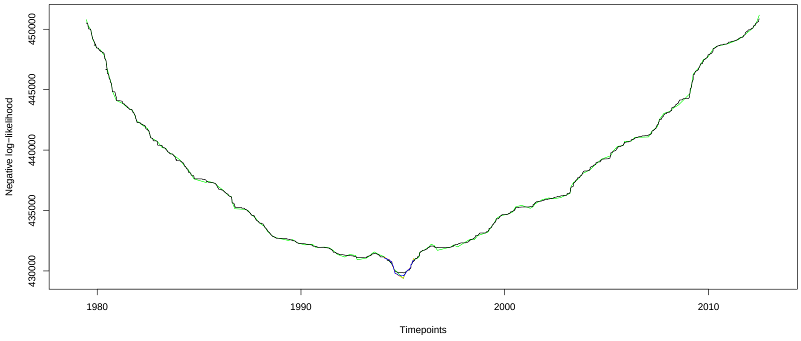


Figure 2.5: Estimate of the change-point for the combined US senate data from 1979-2012

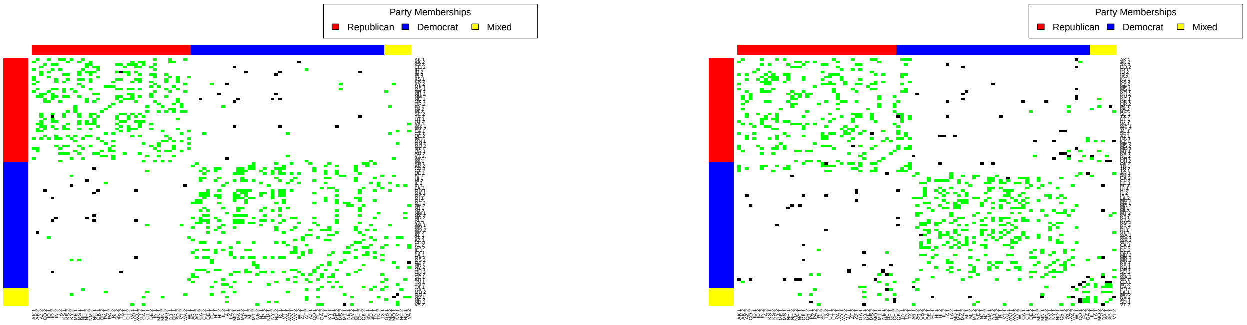


Figure 2.6: Heatmap of the stable network structures before and after the estimated change-point

## 2.8 Proof of Main Theorem 2.8 and Associated lemmas

We organize the proofs as follows. We start with some preliminary lemmas in Section 2.8.1. In particular under assumptions **Assumption** 2.1-**Assumption** 2.3, we derive a bound on the estimation errors $\|\hat{\theta}_{j,\tau} - \theta_\star^{(j)}\|_2$ and this yields a control on the term $\max_\tau |\ell_T(\tau, \hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}) - \ell_T(\tau, \theta_\star^{(1)}, \theta_\star^{(2)})|$, which allows us to conclude that $\hat{\tau}$ is an approximate minimizer of $\tau \mapsto \ell_T(\tau, \theta_\star^{(1)}, \theta_\star^{(2)})$. Using these results we establish Theorem 1 in Section 2.8.2. The proofs of the preliminary lemmas are postponed to Section 2.8.3-2.8.5.

We recall some of the notation defined above. $\mathcal{M}_p$ denote the set of all $p \times p$ real symmetric matrices, equipped with the (modified) Frobenius inner product $\langle \theta, \vartheta \rangle_{\mathsf{F}} \overset{\text{def}}{=} \sum_{k \leq j} \theta_{jk} \vartheta_{jk}$, and the associated norm $\|\theta\|_{\mathsf{F}} \overset{\text{def}}{=} \sqrt{\langle \theta, \theta \rangle}$. With this inner product, we identify $\mathcal{M}_p$ with the Euclidean space $\mathbb{R}^{p(p+1)/2}$, and we systematically use this identification when defining first and second order derivative of functions $f : \mathcal{M}_p \to \mathbb{R}$. For $\theta \in \mathcal{M}_p$ we also define $\|\theta\|_1 \overset{\text{def}}{=} \sum_{k \leq j} |\theta_{jk}|$, and $\|\theta\|_\infty \overset{\text{def}}{=} \sup_{k \leq j} |\theta_{jk}|$. If $u \in \mathbb{R}^d$, for some $d \geq 1$, and $A$ is an ordered subset of $\{1, \ldots, d\}$, we define $u_A \overset{\text{def}}{=} (u_j,\, j \in A)$, and $u_{-j}$ is a shortcut for $u_{\{1,\ldots,d\} \setminus \{j\}}$. Finally we also recall that $\mathcal{T} = \mathcal{T}_+ \cup \mathcal{T}_-$ denotes the search domain as defined in (2.14)-(2.16).

The following properties of the conditional distribution (2.3) will be used below. It is well known (and easy to prove using Fisher's identity) that the function $\theta \mapsto \phi(\theta, x)$ is Lispchitz and

$$|\phi(\theta, x) - \phi(\vartheta, x)| \leq 2c_0 \|\theta - \vartheta\|_1, \quad \theta, \vartheta \in \mathcal{M}_p,\ x \in \mathsf{X}^p, \tag{2.19}$$

where $c_0$ is as in (2.9).

From the expression (2.3) of the conditional densities, using straightforward algebra, it is easy to show that the negative log-pseudo-likelihood function $\phi(\theta, x)$ satisfies the following. For all $\theta, \Delta \in \mathcal{M}_p$, and $x \in \mathsf{X}^p$,

$$\begin{aligned}\phi(\theta+\Delta, x) - \phi(\theta, x) - \langle \nabla_\theta \phi(\theta, x), \Delta\rangle_{\mathsf{F}} \\ = \sum_{j=1}^{p} \left[\log Z^{(j)}_{\theta+\Delta}(x) - \log Z^{(j)}_{\theta}(x) - \sum_{k=1}^{p} \Delta_{jk} \frac{\partial}{\partial \theta_{jk}} \log Z^{(j)}_{\theta}(x)\right].\end{aligned} \tag{2.20}$$

Furthermore by Taylor expansion, we have

$$\begin{aligned}\log Z^{(j)}_{\theta+\Delta}(x) - \log Z^{(j)}_{\theta}(x) - \sum_{k=1}^{p} \Delta_{jk} \frac{\partial}{\partial \theta_{jk}} \log Z^{(j)}_{\theta}(x) \\ = \int_0^1 (1-t)\mathsf{Var}_{\theta+t\Delta}\left(\sum_{k=1}^{p} \Delta_{jk} B_{jk}(X_j, X_k) | X_{-j}\right) \mathrm{d}t \le \frac{c_0^2}{2}\left(\sum_{k=1}^{p} |\Delta_{jk}|\right)^2.\end{aligned} \tag{2.21}$$

### 2.8.1 Preliminary results

We introduce

$$\mathcal{V}^1(\tau, \Delta) \overset{\text{def}}{=} \frac{1}{\tau} \sum_{t=1}^{\tau} \sum_{j=1}^{p} \mathsf{Var}_{\theta_\star^{(1)}}\left(\sum_{k=1}^{p} \Delta_{jk} B_{jk}(X_j, X_k) | X_{-j}\right), \quad \Delta \in \mathcal{M}_p, \ \tau \in \mathcal{T},$$

which is the sample version of the left hand side of (2.11). Similarly we define

$$\mathcal{V}^2(\tau, \Delta) \overset{\text{def}}{=} \frac{1}{T-\tau} \sum_{t=\tau+1}^{T} \sum_{j=1}^{p} \mathsf{Var}_{\theta_\star^{(2)}}\left(\sum_{k=1}^{p} \Delta_{jk} B_{jk}(X_j, X_k) | X_{-j}\right), \quad \Delta \in \mathcal{M}_p, \ \tau \in \mathcal{T}.$$

We introduce

$$G^1_\tau \overset{\text{def}}{=} \frac{1}{T} \sum_{t=1}^{\tau} \nabla\phi\left(\theta_\star^{(1)}; X^{(t)}\right) \text{ and } G^2_\tau \overset{\text{def}}{=} \frac{1}{T} \sum_{t=\tau+1}^{T} \nabla\phi\left(\theta_\star^{(2)}; X^{(t)}\right).$$

For $\tau > 1$, $\rho > 0$, $\lambda > 0$, and for $j = 1, 2$ we work with the event

$$\mathcal{E}^j_\tau(\rho, \lambda) \overset{\text{def}}{=} \left\{\mathcal{V}^j(\tau, \Delta) \ge \rho \sum_{(i,k)\in\mathcal{A}_j} |\Delta_{ik}|^2 \text{ for all } \Delta \in \mathbb{C}_j \text{ and } \|G^j_\tau\|_\infty \le \frac{\lambda}{2}\right\}.$$

The following key lemma is a straightforward variant of lemma 2.2 of Atchade (2014), which itself follows closely Neghaban et al. (2012). For brevity we omit the details here.

**Lemma 2.12.** *Fix $\tau \in \{1, 2, \dots, T-1\}$. Suppose that there exists $\check{\rho}_{1,\tau} > 0$ and $\check{\rho}_{2,\tau} > 0$ such that the event $\mathcal{E}_\tau^1 (\check{\rho}_{1,\tau}, \lambda_{1,\tau}) \cap \mathcal{E}_\tau^2 (\check{\rho}_{2,\tau}, \lambda_{2,\tau})$ holds, where $\lambda_{1,\tau}$ and $\lambda_{2,\tau}$ are as in equation* (2.8)*. Suppose also that*

$$\frac{\tau}{T}\check{\rho}_{1,\tau} \geq 48\lambda_{1,\tau}s_1 \text{ and } \frac{T-\tau}{T}\check{\rho}_{2,\tau} \geq 48\lambda_{2,\tau}s_2. \tag{2.22}$$

*Then $\hat{\theta}_{j,\tau} - \theta_\star^{(j)} \in \mathbb{C}_j$, ($j = 1, 2$), where $\mathbb{C}_j$ is defined in (2.10), and*

$$\|\hat{\theta}_{1,\tau} - \theta_\star^{(1)}\|_2 \leq \frac{T}{\tau}\frac{24s_1^{1/2}\lambda_{1,\tau}}{\check{\rho}_{1,\tau}} \text{ and } \|\hat{\theta}_{2,\tau} - \theta_\star^{(2)}\|_2 \leq \frac{T}{T-\tau}\frac{24s_2^{1/2}\lambda_{2,\tau}}{\check{\rho}_{2,\tau}}. \tag{2.23}$$

The next result follows easily.

**Lemma 2.13.** *Fix $\tau \in \{1, 2, \dots, T-1\}$. Under the assumptions of lemma 2.12,*

$$\left|\ell_T(\tau, \hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}) - \ell_T(\tau, \theta_\star^{(1)}, \theta_\star^{(2)})\right| \lesssim M\frac{\log(dT)}{T},$$

*where $M = \frac{s_1}{\check{\rho}_{1,\tau}}\left(1 + \frac{c_0^2 s_1}{\check{\rho}_{1,\tau}}\right) + \frac{s_2}{\check{\rho}_{2,\tau}}\left(1 + \frac{c_0^2 s_2}{\check{\rho}_{2,\tau}}\right)$.*

*Proof.* See Section 2.8.3. □

The next two lemmas imply that under **Assumption** 2.1 and **Assumption** 2.3, the event $\mathcal{E}_\tau^1 (\check{\rho}_{1,\tau}, \lambda_{1,\tau}) \cap \mathcal{E}_\tau^2 (\check{\rho}_{2,\tau}, \lambda_{2,\tau})$ holds with high probability. This is explicitly stated in the following corollary.

**Lemma 2.14.** *Assume **Assumption** 2.1 and **Assumption** 2.3, and $\mathcal{T} \neq \emptyset$. With $\lambda_{1,\tau}, \lambda_{2,\tau}$ as in equation* (2.8)*,*

$$\mathbb{P}\left[\max_{\tau\in\mathcal{T}} 2\lambda_{1,\tau}^{-1}\|G_\tau^1\|_\infty > 1\right] \leq \frac{2}{Td}, \text{ and } \mathbb{P}\left[\max_{\tau\in\mathcal{T}} 2\lambda_{2,\tau}^{-1}\|G_\tau^2\|_\infty > 1\right] \leq \frac{2}{Td}$$

*where $d = p(p+1)/2$.*

*Proof.* See Section 2.8.4. □

**Lemma 2.15.** *Assume* ***Assumption*** *2.1 and* ***Assumption*** *2.3, and $\mathcal{T} \neq \emptyset$. With probability at least $1 - \frac{4}{d}$ the following holds: for all $\tau \in \mathcal{T}$, for all $\Delta^{(1)} \in \mathbb{C}_1$, and for all $\Delta^{(2)} \in \mathbb{C}_2$,*

$$\mathcal{V}^{(1)}\left(\tau, \Delta^{(1)}\right) \geq \rho_1 \|\Delta^{(1)}_{\mathcal{A}_1}\|_2^2, \textit{ and } \mathcal{V}^{(2)}\left(\tau, \Delta^{(2)}\right) \geq \rho_2 \|\Delta^{(2)}_{\mathcal{A}_2}\|_2^2.$$

*Proof.* See Section 2.8.5. □

We combine the last two lemmas to obtain the following.

**Corollary 2.16.** *Assume* ***Assumption*** *2.1 and* ***Assumption*** *2.3, and $\mathcal{T} \neq \emptyset$. Let $\lambda_{1,\tau}$ and $\lambda_{2,\tau}$ as in equation (2.8). Then the event $\bigcap_{\tau \in \mathcal{T}} \left[\mathcal{E}^1_\tau\left(\rho_1, \lambda_{1,\tau}\right) \cap \mathcal{E}^2_\tau\left(\rho_2, \lambda_{2,\tau}\right)\right]$ holds with probability at least $1 - \frac{8}{d}$.*

### 2.8.2 Proof of Theorem 2.8

*Proof.* We use $\ell_T(\tau)$ instead of $\ell_T\left(\tau; \hat{\theta}_{1,\tau}, \hat{\theta}_{2,\tau}\right)$ for notational convenience, and we define $r_T(\tau) \overset{\text{def}}{=} \ell_T(\tau) - \ell_T\left(\tau, \theta_\star^{(1)}, \theta_\star^{(2)}\right)$. Fix $\tau \in \mathcal{T}_+$. We have

$$\begin{aligned} \ell_T(\tau) &= \ell_T\left(\tau, \theta_\star^{(1)}, \theta_\star^{(2)}\right) + r_T(\tau), \\ &= \left[\ell_T\left(\tau, \theta_\star^{(1)}, \theta_\star^{(2)}\right) - \ell_T\left(\tau_\star, \theta_\star^{(1)}, \theta_\star^{(2)}\right)\right] + \ell_T\left(\tau_\star, \theta_\star^{(1)}, \theta_\star^{(2)}\right) + r_T(\tau). \end{aligned} \tag{2.24}$$

It is straightforward to check that

$$\ell_T\left(\tau, \theta_\star^{(1)}, \theta_\star^{(2)}\right) - \ell_T\left(\tau_\star, \theta_\star^{(1)}, \theta_\star^{(2)}\right) = \frac{1}{T} \sum_{t=\tau_\star+1}^{\tau} \left(\phi(\theta_\star^{(1)}, X^{(t)}) - \phi(\theta_\star^{(2)}, X^{(t)})\right).$$

Recall that $\kappa_0$ is defined in **Assumption** 2.5 as

$$\kappa_0 = \mathbb{E}_{\theta_\star^{(2)}}\left[\phi(\theta_\star^{(1)}, X^{(t)}) - \phi(\theta_\star^{(2)}, X^{(t)})\right].$$

We then define $Z^{(t)} = \phi(\theta_\star^{(1)}, X^{(t)}) - \phi(\theta_\star^{(2)}, X^{(t)}) - \kappa_0$. Hence $\mathbb{E}_{\theta_*^{(2)}}\left(Z^{(t)}\right) = 0$, and (2.24) becomes

$$\begin{aligned}\ell_T(\tau) - \ell_T(\tau_\star) &= \frac{(\tau-\tau_\star)\kappa_0}{T} + \frac{1}{T}\sum_{t=\tau_\star+1}^{\tau} Z_t + \left[\ell_T\left(\tau_\star, \theta_\star^{(1)}, \theta_\star^{(2)}\right) - \ell_T(\tau_\star)\right] + r_T(\tau) \\ &= \frac{(\tau-\tau_\star)\kappa_0}{T} + \frac{1}{T}\sum_{t=\tau_\star+1}^{\tau} Z_t + r_T(\tau) - r_T(\tau_*)\end{aligned} \tag{2.25}$$

We conclude from lemma 2.13 that on the event $\bigcap_{\tau\in\mathcal{T}}\left[\mathcal{E}_\tau^1(\rho_1, \lambda_{1,\tau}) \cap \mathcal{E}_\tau^2(\rho_2, \lambda_{2,\tau})\right]$,

$$\ell_T(\tau) - \ell_T(\tau_\star) = \frac{(\tau-\tau_\star)\kappa_0}{T} + \frac{1}{T}\sum_{t=\tau_\star+1}^{\tau} Z_t + \epsilon_T(\tau),$$
$$\text{where } \;\epsilon_T(\tau) = r_T(\tau) - r_T(\tau_*) \text{ and } \;\max_{\tau\in\mathcal{T}}|\epsilon_T(\tau)| \le CM\frac{\log(dT)}{T}, \tag{2.26}$$

for some universal constant $C$, and where $M$ is as defined in the statement of Theorem 1. For $\delta > 0$, we set $B \overset{\text{def}}{=} \frac{CM(1+\delta)}{\kappa_0}\log(dT)$. Notice that the event

$$\{\hat{\tau} > \tau_\star + B\} \subset \bigcup_{j\ge 0, \{\tau_\star+\lceil B\rceil+j\}\in\mathcal{T}} \{\hat{\tau} = \tau_\star + \lceil B\rceil + j\}.$$

Equation (2.26) implies that on the event $\bigcap_{\tau\in\mathcal{T}}\left[\mathcal{E}_\tau^1(\rho_1, \lambda_{1,\tau}) \cap \mathcal{E}_\tau^2(\rho_2, \lambda_{2,\tau})\right]$, and for $\tau_\star + \lceil B\rceil + j \in \mathcal{T}$, the event $\{\hat{\tau} = \tau_\star + \lceil B\rceil + j\}$ is also a subset of

$$\{\ell_T(\tau_\star + \lceil B\rceil + j) \le \ell_T(\tau_\star)\} \subseteq \left\{-\sum_{t=\tau_\star+1}^{\tau_\star+\lceil B\rceil+j} Z_t \ge (\lceil B\rceil + j)\,\kappa_0 - CM\log(dT)\right\}. \tag{2.27}$$

However by Corollary 2.16, the event $\cap_{\tau\in\mathcal{T}}\left[\mathcal{E}_\tau^1(\rho_1, \lambda_{1,\tau}) \cap \mathcal{E}_\tau^2(\rho_2, \lambda_{2,\tau})\right]$ occurs with probability at least $1 - 8/d$. This, together with (2.26) and (2.27) imply that

$$\mathbb{P}\left[\hat{\tau} > \tau_\star + B\right] \le \frac{8}{d} + \sum_{j\ge 0}\mathbb{P}\left[-\sum_{t=\tau_\star+1}^{\tau_\star+\lceil B\rceil+j} Z^{(t)} > \kappa_0 j + CM\delta\log(dT)\right] \tag{2.28}$$

We set

$$A \overset{\text{def}}{=} CM\delta\log(dT), \quad\text{and}\quad L \overset{\text{def}}{=} \frac{1}{8c_0^2\|\theta_\star^{(2)} - \theta_\star^{(1)}\|_1^2},$$

where $c_0$ is as in (2.9). Using (2.19), Hoeffding's inequality and the inequality $\frac{(A+\kappa_0 j)^2}{(\lceil B\rceil+j)} \geq (A+\kappa_0 j)\frac{A}{\lceil B\rceil}$, we deduce that

$$\begin{aligned}
\mathbb{P}\left[\hat{\tau} > \tau_\star + B\right] &\leq \frac{8}{d} + \sum_{j\geq 0}\exp\left(-\frac{L\left(A+\kappa_0 j\right)^2}{\lceil B\rceil + j}\right)\\
&\leq \frac{8}{d} + \exp\left(-\frac{LA^2}{\lceil B\rceil}\right)\sum_{j\geq 0}\exp\left(-\frac{LA\kappa_0 j}{\lceil B\rceil}\right)\\
&\leq \frac{8}{d} + \exp\left(-\frac{LA^2}{\lceil B\rceil}\right)\left(1-\exp\left(-\frac{LA\kappa_0}{\lceil B\rceil}\right)\right)^{-1}. \qquad (2.29)
\end{aligned}$$

Using **Assumption** 2.5 and the monotonicity of $x \mapsto ax/(bx+c)$ for $ac > 0$ we write

$$\frac{L\kappa_0 A}{\lceil B\rceil} \geq \frac{L\kappa_0^2 CM\delta}{CM(1+\delta)+\kappa_0} \geq \frac{8}{c_0^2}\frac{\epsilon CM\delta}{CM\delta+\kappa_0+CM} \geq \frac{4\epsilon}{c_0^2},$$

provided that $\delta \geq 1 + \frac{\kappa_0}{CM}$. Using again the fact that $x \mapsto ax/(bx+c)$ is increasing for $ac > 0$, we get

$$\frac{LA^2}{\lceil B\rceil} \geq \left(\frac{\kappa_0 LC^2M^2\delta^2}{CM\delta + CM + \kappa_0}\right)\log(dT).$$

We also use the fact that for $a, b, c > 0$, $\frac{ax^2}{bx+c} \geq 1$ for $x \geq \frac{b+\sqrt{b^2+4ac}}{2a}$ to deduce that $\frac{\kappa_0 LC^2M^2\delta^2}{CM\delta+CM+\kappa_0} \geq 1$, for $\delta = 1 + \frac{\kappa_0}{CM} + \frac{1}{L\kappa_0 CM}$. Hence for $\delta = 1 + \frac{\kappa_0}{CM} + \frac{1}{L\kappa_0 CM}$

$$\mathbb{P}\left[\hat{\tau} > \tau_\star + B\right] \leq \frac{8}{d} + \frac{1}{dT}\left(1-\exp\left(-\frac{4\epsilon}{c_0^2}\right)\right)^{-1}. \qquad (2.30)$$

A similar bound holds for $\mathbb{P}\left[\hat{\tau} < \tau_\star - B\right]$. Thus we conclude that with a probability tending to one as $p \to \infty$, $|\hat{\tau} - \tau_\star| \leq \left(1 + \frac{2CM}{\kappa_0} + \frac{1}{L\kappa_0^2}\right)\log(dT)$, as claimed. □

### 2.8.3 Proof of lemma 2.13

*Proof.* Set $\hat{\Delta}_\tau^{(j)} \overset{\text{def}}{=} \hat{\theta}_{j,\tau} - \theta_*^{(j)}$. From (2.20) we have

$$\frac{1}{T}\sum_{t=1}^{\tau}\left[\phi\left(\theta_*^{(1)} + \hat{\Delta}_\tau^{(1)}, X^{(t)}\right) - \phi\left(\theta_*^{(1)}, X^{(t)}\right)\right] = \frac{1}{T}\sum_{t=1}^{\tau}\left\langle \nabla_\theta\phi\left(\theta_*^{(1)}, X^{(t)}\right), \hat{\Delta}_\tau^{(1)}\right\rangle_{\mathsf{F}}$$
$$+\frac{1}{T}\sum_{t=1}^{\tau}\left[\sum_{j=1}^{p}\log Z^{(j)}_{\theta_\star^{(1)}+\hat{\Delta}_\tau^{(1)}}(X^{(t)}) - \log Z^{(j)}_{\theta_\star^{(1)}}(X^{(t)}) - \sum_{k=1}^{p}\Delta_{jk}\frac{\partial}{\partial\theta_{jk}}\log Z^{(j)}_{\theta_\star^{(1)}}(X^{(t)})\right].$$

On $\mathcal{E}_\tau^1\left(\check{\rho}_{1,\tau},\lambda_{1,\tau}\right)$, $\left\|T^{-1}\sum_{t=1}^{\tau}\nabla\phi\left(\theta_*^{(1)},X^{(t)}\right)\right\|_\infty \leq \dfrac{\lambda_{1,\tau}}{2}$ and $\hat{\Delta}_\tau^{(1)}\in\mathbb{C}_1$. Hence

$$
\begin{aligned}
\left|\frac{1}{T}\sum_{t=1}^{\tau}\left\langle\nabla_\theta\phi\left(\theta_*^{(1)},X^{(t)}\right),\hat{\Delta}_\tau^{(1)}\right\rangle_{\mathsf{F}}\right| &\leq \left\|\frac{1}{T}\sum_{t=1}^{\tau}\nabla_\theta\phi\left(\theta_*^{(1)},X^{(t)}\right)\right\|_\infty\|\hat{\Delta}_\tau^{(1)}\|_1\\
&\leq \frac{\lambda_{1,\tau}}{2}\|\hat{\theta}_{1,\tau}-\theta_*^{(1)}\|_1\\
&\leq 2\lambda_{1,\tau}s_1^{1/2}\left\|\hat{\theta}_{1,\tau}-\theta_*^{(1)}\right\|_{\mathsf{F}}\\
&\leq C\frac{s_1}{\check{\rho}_{1,\tau}}\frac{\log\left(dT\right)}{T},
\end{aligned}
$$

where $C$ can be taken as $2\cdot 24\cdot 48^2$. On $\mathcal{E}_\tau^1\left(\check{\rho}_{1,\tau},\lambda_{1,\tau}\right)$, using (2.21) and lemma 2.12,

$$
\begin{aligned}
&\left|\frac{1}{T}\sum_{t=1}^{\tau}\left[\sum_{j=1}^{p}\log Z^{(j)}_{\theta_\star^{(1)}+\hat{\Delta}_\tau^{(1)}}(X^{(t)})-\log Z^{(j)}_{\theta_\star^{(1)}}(X^{(t)})-\sum_{k=1}^{p}\Delta_{jk}\frac{\partial}{\partial\theta_{jk}}\log Z^{(j)}_{\theta_\star^{(1)}}(X^{(t)})\right]\right|\\
&\qquad\leq\frac{c_0^2\tau}{2T}\sum_{j=1}^{p}\left(\sum_{k=1}^{p}|\hat{\Delta}_{jk}^{(1)}|\right)^2\leq\frac{c_0^2\tau}{T}\|\hat{\Delta}_\tau^{(1)}\|_1^2\leq 8\cdot 48^2\cdot 24^2c_0^2\left(\frac{s_1}{\check{\rho}_{1,\tau}}\right)^2\frac{\log(dT)}{T}.
\end{aligned}
$$

We combine these two bound to conclude that on $\mathcal{E}_\tau^1\left(\check{\rho}_{1,\tau},\lambda_{1,\tau}\right)$

$$
\left|\frac{1}{T}\sum_{t=1}^{\tau}\left[\phi\left(\hat{\theta}_{1,\tau},X^{(t)}\right)-\phi\left(\theta_*^{(1)},X^{(t)}\right)\right]\right|\lesssim M_1\frac{\log\left(dT\right)}{T},
$$

where $M_1=\frac{s_1}{\check{\rho}_{1,\tau}}\left(1+\frac{c_0^2s_1}{\check{\rho}_{1,\tau}}\right)$. A similar bound holds for the second term

$$
\frac{1}{T}\sum_{t=\tau+1}^{T}\left[\phi\left(\hat{\theta}_{2,\tau},X^{(t)}\right)-\phi\left(\theta_*^{(2)},X^{(t)}\right)\right],
$$

and the lemma follows. □

### 2.8.4 Proof of lemma 2.14

*Proof.* We carry the details for the first bound. The second is done similarly. For $\tau \in \mathcal{T}_+$, we calculate that for $1 \leq j \leq i \leq p$,

$$\frac{\partial}{\partial \theta_{ij}} \left[ -\frac{1}{T} \sum_{t=1}^{\tau} \phi\left(\theta_\star^{(1)}, X^{(t)}\right) \right]$$

$$= \begin{cases} -\frac{1}{T} \sum_{t=1}^{\tau} \left[ B_0(X_i^{(t)}) - \mathbb{E}_{\theta_\star^{(1)}}(B_0(X_i | X_{-i}^{(t)}) \right] & \text{if } i = j \\ -\frac{1}{T} \sum_{t=1}^{\tau} \left[ 2B(X_i^{(t)}, X_j^{(t)}) - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-i}^{(t)} \right) \right. & \\ \qquad \left. - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-j}^{(t)} \right) \right] & \text{if } \ j < i \end{cases}$$

In the above display the notation $\mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-i}^{(t)} \right)$ is defined as the function $z \mapsto \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-i} = z_{-i} \right)$ evaluated on $X^{(t)}$.

Fix a pair of nodes $j < i$ (the argument is similar for $i = j$). Set

$$\mu_{ij}^{(t)} \stackrel{\text{def}}{=} \mathbb{E} \left[ 2B(X_i^{(t)}, X_j^{(t)}) - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-i}^{(t)} \right) - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-j}^{(t)} \right) \right],$$

and

$$V_{ij}^{(t)} \stackrel{\text{def}}{=} 2B(X_i^{(t)}, X_j^{(t)}) - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-i}^{(t)} \right) - \mathbb{E}_{\theta_\star^{(1)}} \left( B(X_i, X_j) | X_{-j}^{(t)} \right) - \mu_{ij}^{(t)},$$

so that $\mathbb{E}\left( V_{ij}^{(t)} \right) = 0$ and

$$\frac{\partial}{\partial \theta_{ij}} \left[ \frac{1}{T} \sum_{t=1}^{\tau} \phi\left(\theta_*^{(1)}, X^{(t)}\right) \right] = \frac{1}{T} \sum_{t=1}^{\tau} V_{ij}^{(t)} + \frac{1}{T} \sum_{t=1}^{\tau} \mu_{ij}^{(t)}.$$

The important point to notice is that for $t \leq \tau_\star$, $\mu_{ij}^{(t)} = 0$. For $t > \tau_\star$, we can bound $\mu_{ij}^{(t)}$ by comparing the conditional expectation of $B(X_i, X_j)$ under $g_{\theta_\star^{(1)}}$ and $g_{\theta_\star^{(2)}}$. To this end, we use lemma 2.17 which gives that for $t > \tau_\star$,

$$\begin{aligned}
&\left|\mathbb{E}\left[B(X_i^{(t)}, X_j^{(t)}) - \mathbb{E}_{\theta_\star^{(1)}}\left(B(X_i, X_j)|X_{-i}^{(t)}\right)\right]\right| \\
&\quad = \left|\mathbb{E}\left[\int_{\mathsf{X}} B(u, X_j^{(t)}) f_{\theta_\star^{(2)}}(u|X_{-i}^{(t)})\mathrm{d}u - \int_{\mathsf{X}} B(u, X_j^{(t)}) f_{\theta_\star^{(1)}}(u|X_{-i}^{(t)})\mathrm{d}u\right]\right| \\
&\qquad \leq c_0^2 \sum_{j=1}^{p} |\theta_{\star,ij}^{(2)} - \theta_{\star,ij}^{(1)}| \leq bc_0^2,
\end{aligned}$$

where $b$ is as in (2.17). Hence

$$\left|\mu_{ij}^{(t)}\right| \leq 2\max_{j\leq i}\left|\mathbb{E}_{\theta_\star^{(2)}}\left(B(X_i^{(t)}, X_j^{(t)})|X_{-i}^{(t)}\right) - \mathbb{E}_{\theta_\star^{(1)}}\left(B(X_i^{(t)}, X_j^{(t)})|X_{-j}^{(t)}\right)\right| \leq 2bc_0^2.$$

Using the fact that for $\tau \in \mathcal{T}_+$, $8bc_0^2\left(\tau - \tau_*\right) \leq \lambda_{1,\tau}T$ we conclude that

$$\frac{1}{T}\left|\sum_{t=\tau_*+1}^{\tau}\mu_{ij}^{(t)}\right| \leq \frac{\left(\tau - \tau_*\right)c_0^2 b}{T} \leq \frac{\lambda_{1,\tau}}{4}.$$

Now by Hoeffding's inequality,

$$\begin{aligned}
\mathbb{P}_*\left[\left|\frac{\partial}{\partial\theta_{ij}}\left[\frac{1}{T}\sum_{t=1}^{\tau}\phi\left(\theta_*^{(1)}, X^{(t)}\right)\right]\right| > \frac{\lambda_{1,\tau}}{2}\right] &\leq \mathbb{P}_*\left[\left|\sum_{t=1}^{\tau}V_{ij}^{(t)}\right| > \frac{T\lambda_{1,\tau}}{4}\right] \\
&\leq 2\exp\left(-\frac{T^2\lambda_{1,\tau}^2}{2^7c_0^2\tau}\right) \\
&\leq 2\exp\left(-2\log\left(Td\right)\right).
\end{aligned}$$

A similar bound holds when $i = j$, and for $\tau \in \mathcal{T}_-$. We conclude by a union-sum inequality that

$$\mathbb{P}_*\left[\max_{\tau\in\mathcal{T}} 2\lambda_{1,\tau}^{-1}\left\|G_\tau^1\right\|_\infty > 1\right] \leq 2\exp\left(\log\left(Td\right) - 2\log\left(Td\right)\right) \leq \frac{2}{Td}.$$

□ □

### 2.8.5 Proof of lemma 2.15

*Proof.* We prove the first bound, the second bound is similar, if not simpler since there is no misspecification. We define

$$W_{jkk'}^{(t)} \overset{\mathrm{def}}{=} \mathsf{Cov}_{\theta_\star^{(1)}}\left(B(X_j^{(t)}, X_k^{(t)}), B(X_j^{(t)}, X_{k'}^{(t)})|X_{-j}^{(t)}\right) \\ - \mathbb{E}\left[\mathsf{Cov}_{\theta_\star^{(1)}}\left(B(X_j^{(t)}, X_k^{(t)}), B(X_j^{(t)}, X_{k'}^{(t)})|X_{-j}^{(t)}\right)\right]$$

Then for $\Delta^{(1)} = \Delta \in \mathbb{C}_1 \setminus \{0\}$,

$$\begin{aligned} \mathcal{V}^1(\tau, \Delta) &= \frac{1}{\tau}\sum_{t=1}^{\tau}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}W_{jkk'}^{(t)} \qquad (2.31)\\ &\quad + \frac{1}{\tau}\sum_{t=1}^{\tau}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}\mathbb{E}\left[\mathsf{Cov}_{\theta_\star^{(1)}}\left(B(X_j^{(t)}, X_k^{(t)}), B(X_j^{(t)}, X_{k'}^{(t)})|X_{-j}^{(t)}\right)\right]. \end{aligned}$$

Using **Assumption** 2.1, we deduce that

$$\begin{aligned} \mathcal{V}^1(\tau, \Delta) &\geq 2\rho_1\|\Delta_{\mathcal{A}_1}\|_2^2 + \frac{1}{\tau}\sum_{t=1}^{\tau}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}W_{jkk'}^{(t)} \\ &\quad + \frac{\tau-\tau_*}{\tau}\sum_{j=1}^{p}\mathbb{E}_{\theta_\star^{(2)}}\left[\mathsf{Var}_{\theta_\star^{(1)}}\left(\sum_{k=1}^{p}\Delta_{jk}B_{ik}(X_j, X_k)|X_{-j}\right)\right] \\ &\quad - \frac{\tau-\tau_*}{\tau}\sum_{j=1}^{p}\mathbb{E}_{\theta_\star^{(1)}}\left[\mathsf{Var}_{\theta_\star^{(1)}}\left(\sum_{k=1}^{p}\Delta_{jk}B_{ik}(X_j, X_k)|X_{-j}\right)\right]. \qquad (2.32) \end{aligned}$$

By the comparison lemma 2.17

$$\begin{aligned} &\Bigg|\mathbb{E}_{\theta_\star^{(2)}}\left[\mathsf{Var}_{\theta_\star^{(1)}}\left(\sum_{k=1}^{p}\Delta_{jk}B_{ik}(X_j, X_k)|X_{-j}\right)\right] \\ &\qquad - \mathbb{E}_{\theta_\star^{(1)}}\left[\mathsf{Var}_{\theta_\star^{(1)}}\left(\sum_{k=1}^{p}\Delta_{jk}B_{ik}(X_j, X_k)|X_{-j}\right)\right]\Bigg| \\ &\qquad\qquad \leq c_0^3\left(\sum_{k=1}^{p}|\Delta_{jk}|\right)^2\sum_{k=1}^{p}|\theta_{\star jk}^{(1)} - \theta_{\star jk}^{(2)}| \leq c_0^3 b\left(\sum_{k=1}^{p}|\Delta_{jk}|\right)^2, \end{aligned}$$

which implies that

$$\mathcal{V}^1(\tau, \Delta) \geq \left(2\rho_1 - 32\left(\frac{\tau - \tau_\star}{\tau}\right) s_1 c_0^3 b\right) \|\Delta_{\mathcal{A}_1}\|_2^2 + \frac{1}{\tau}\sum_{t=1}^{\tau}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}W_{jkk'}^{(t)}.$$

Given that on $\mathcal{T}_+$, $64(\tau - \tau_\star)s_1 c_0^3 b \leq \rho_1 \tau$, it follows that

$$\mathcal{V}^1(\tau, \Delta) \geq \frac{3}{2}\rho_1 \|\Delta_{\mathcal{A}_1}\|_2^2 + \frac{1}{\tau}\sum_{t=1}^{\tau}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}W_{jkk'}^{(t)} \tag{2.33}$$

Set $Z_{jkk'}^{\tau} \overset{\text{def}}{=} \frac{1}{\tau}\sum_{t=1}^{\tau} W_{jkk'}^{(t)}$. We conclude from equation (2.33) that if for some $\Delta \in \mathbb{C}_1 \setminus \{0\}$, and for some $\tau \geq \tau^*$,

$$\mathcal{V}^1(\tau, \Delta) \leq \rho_1 \|\Delta_{\mathcal{A}_1}\|_2^2 \tag{2.34}$$

then

$$\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}Z_{jkk'}^{(\tau)} \leq -\frac{\rho_1}{2}\|\Delta_{\mathcal{A}_1}\|_2^2.$$

But on the other hand, using the fact that $\Delta \in \mathbb{C}_1$,

$$\begin{aligned}\sum_{j=1}^{p}\sum_{k,k'=1}^{p} \Delta_{jk}\Delta_{jk'}Z_{jkk'}^{(\tau)} &\geq -\left(\sup_{j,k,k'} |Z_{jkk'}^{(\tau)}|\right)\left(\sum_{i=1}^{p}\sum_{k=1}^{p}|\Delta_{ik}|\right)^2 \\ &\geq -\left(\sup_{j,k,k'} |Z_{jkk'}^{(\tau)}|\right) 16\|\Delta_{\mathcal{A}_1}\|_1^2 \\ &\geq -16 s_1 \left(\sup_{j,k,k'} |Z_{jkk'}^{(\tau)}|\right)\|\Delta_{\mathcal{A}_1}\|_2^2.\end{aligned}$$

Therefore if there exists a non-zero $\Delta \in \mathbb{C}_1$ and $\tau \geq \tau_*$ such that equation (2.34) holds then $\left(\sup_{j,k,k'} |Z_{jkk'}^{(\tau)}|\right) \geq \rho_1/32s_1$. But by Hoeffding's inequality and a union-sum bound,

$$\mathbb{P}\left[\sup_{j,k,k'} |Z_{jkk'}^{(\tau)}| \geq \frac{\rho_1}{32 s_1}\right] \leq 2\exp\left(3\log p - \frac{\tau\rho_1^2}{2^9 c_0^2 s_1^2}\right) \leq \frac{2}{p},$$

since for $\tau \in \mathcal{T}$, $\tau \geq 2^{11}c_0^2 s_1^2 \rho_1^{-2}\log p$. □

**Lemma 2.17.** *Let $(\mathsf{Y}, \mathcal{A}, \nu)$ be a measure space where $\nu$ is a finite measure. Let $g_1, g_2, f_1, f_2 : \mathsf{Y} \to \mathbb{R}$ be bounded measurable functions. Set $Z_{g_i} \overset{\text{def}}{=} \int_{\mathsf{Y}} e^{g_i(y)}\nu(dy)$,*

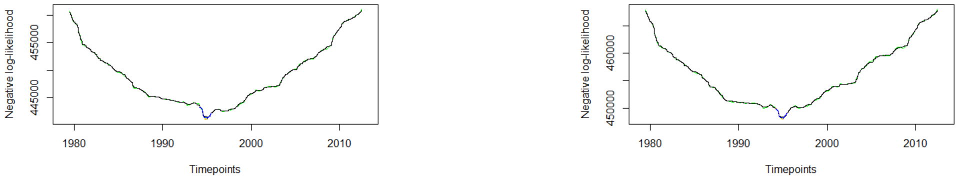


Figure 2.7: Estimated Change-points via imputation technique (i) and (ii) respectively

*$i \in \{1, 2\}$. Then*

$$\left| \frac{1}{Z_{g_1}} \int f_1(y) e^{g_1(y)} \nu(dy) - \frac{1}{Z_{g_2}} \int f_2(y) e^{g_2(y)} \nu(dy) \right| \\ \leq \|f_2 - f_1\|_\infty + \frac{1}{2} \mathsf{osc}(g_2 - g_1) \left( \mathsf{osc}(f_1) + \mathsf{osc}(f_2) \right),$$

*where $\|f\|_\infty = \sup_{x \in \mathsf{Y}} |f(x)|$, and $\mathsf{osc}(f) \overset{\text{def}}{=} \sup_{x,y \in \mathsf{Y}} |f(x) - f(y)|$ is the oscillation of $f$.*

*Proof.* The proof follows from Atchade (2014) lemma A.4. □

## 2.9 Different Methods of Missing Data Imputation for the Real Data Application

In the main paper we replaced the missing votes by the value (yes/no) of that member's party majority position on that particular vote. Here we employed two other missing data imputation techniques viz. (i) replacing all missing values by the value (yes/no) representing the winning majority on that bill and (ii) replacing the missing value of a Senator by the value that the majority of the opposite party voted on that particular bill. The estimated change-point obtained following these two imputation methods are not much different . The imputation technique (i) results in a estimated change-point at January 19, 1995 and the technique (ii) yields estimated change-point at January 17, 1995 respectively. The change-point estimate we obtained in the main paper was January 17, 1995. Clearly there is not much difference between the different imputation techniques and Fig. 2.7 also conveys the same message.

## 2.10 Discussion

We analyzed a change-point estimation problem in the context of a high dimensional MRF and established the rate of convergence of $O\left(\log(dT)\right)$ for the estimated change-point to the truth. Recall that the usual rate of convergence in the low dimensional setting is $O(1)$, as discussed in Bai (1997); Kosorok (2007). The logarithmic factor seems to be the cost that one has to account for the high dimensionality of the problem. Another key aspect that we investigate in our theoretical analysis is the model misspecification in high dimensional setting which can be of independent interest in some other problems as well.

# CHAPTER 3

# Parallel Optimization Algorithm for Large Heterogeneous Data

## 3.1 Introduction

*Heterogeneity* is one of the important features exhibited in a large high-dimensional data. In this chapter, we consider a computational problem involving efficient parameter estimation in a large heterogeneous data. The two main problems one faces in dealing with these large datasets are -(1) the size of the data is so large that it may be infeasible to store all of the data on a single computer and (2) the standard algorithms used to solve the optimization problem for parameter estimation becomes extremely slow. To tackle these issues one of the common strategies that most of the distributed algorithms (Agarwal and Duchi (2011), Zinkevich et al. (2010), Zhang et al. (2013) etc.) utilize is to split the large data into small parts and use some optimization algorithm to perform parameter estimation on those fractions of data using several machines ("Divide and Conquer"). The final parameter estimate is obtained by taking a simple average of the individual estimates from different machines. The only communication step in these algorithms is the final aggregation step combining estimates from different machines.

The averaging step at the end of the parallel implementation ensure variance reduction of the estimate relative to a serial implementation. But the bias of the estimate is not reduced by simple averaging. We provide a parallel algorithm that involves random subsampling and a communication step among different machines after each iteration of the optimization algorithm. Instead of data splitting, we use a random subsample of the full data on individual machines. The communication is done in such a way that the estimates in each machine can utilize all the subsamples on different machines over the iterations. This yields bias reduction of the estimate relative to a parallel implementation where we do not communicate among the machines. As before, the final estimate is obtained by taking a simple average of all the different estimates at the end of the iterations. We provide a sharp analysis on the rate of decay of bias and variance of our parallel implementation and compare it with the parallel scheme without communication.

Gaussian Mixture Model (GMM) is a prime example that can be used to generate heterogeneous data. We provide experimental evaluation of our method by simulating large dataset from GMM with high dimension and different levels of overlap among the mixture components. We compare the performance of our parallel implementation with communication with the one that is employed in a parallel manner but without communication.

## 3.2 Parallel Algorithms for Large Dataset

Many procedures for statistical estimation involves minimizing a objective function iteratively with respect to the parameters. Given the current explosion in the size and amount of data available in statistical studies, a central challenge is to design efficient algorithms for solving large-scale problem instances. In a centralized setting there are many procedures for performing iterative optimization of a objective function viz. EM algorithm and its variants (Dempster et al. (1977); Wei and Tanner (1990); Nielsen (2000)), Stochastic approximation and optimization algorithms (Robbins and Monro (1951); Spall (1998)), gradient descent (Boyd and Vandenberghe (2004)), coorodinate descent (Boyd and Vandenberghe (2004)) etc. When the size of the dataset becomes extremely large, however, it may be infeasible to store all of the data on a single computer, or at least to keep the data in memory. Accordingly, the focus of this paper is the study of some parallel communication-efficient procedures for iterative optimization of a objective function.

Recent years have witnessed a flurry of research on distributed approaches to solving very large-scale statistical optimization problems. To name a few Nedic and Ozdaglar (2009); Ram et al. (2010); Johansson et al. (2009); Duchi et al. (2012); Dekel et al. (2012); Agarwal and Duchi (2011); Recht et al. (2011) etc. It can be difficult within a purely optimization theoretic setting to show explicit benefits arising from parallel computation. In statistical settings, however, parallel computation can lead to gains in computational efficiency, as shown by a number of authors (Agarwal and Duchi

(2011); Dekel et al. (2012); Recht et al. (2011); Duchi et al. (2012)). Within the family of distributed algorithms, there can be significant differences in communication complexity: different computers must be synchronized, and when the dimensionality of the data is high, communication can be prohibitively expensive. It is thus interesting to study parallel estimation algorithms that require fairly limited synchronization and communication while still enjoying the greater statistical accuracy that is usually associated with a larger dataset.

With this context, perhaps the simplest algorithm is the *average mixture*(AVGM) algorithm considered in Zhang et al. (2013). It is an appealingly simple method: given m machines and a dataset of size N, first assign to each machine a dataset (distinct) of size $n = N/m$, then have each machine perform the iterative optimization on its fraction of the data to compute estimates $\theta_i$ and then average all the parameter estimates $\theta_i$ across machines. This approach has been studied for some classification and estimation problems by Mcdonald et al. (2009) and McDonald et al. (2010), as well as for certain stochastic approximation methods by Zinkevich et al. (2010). Mcdonald et al. (2009) showed the variance reduction via this parallelization scheme compared to the single processor solution. Zinkevich et al. (2010) showed that the bias reduction is possible with a stochastic gradient descent algorithm which digests not a fixed fraction of data but rather a random fixed subset of data (See Algorithm 3 in Zinkevich et al. (2010)).

However we introduce a parallel algorithm that does not require distinct datasets in different machines i.e. instead of splitting the dataset as in Zhang et al. (2013) or Zinkevich et al. (2010) we use a random subsampling from the full dataset. The paper makes two contributions. First in Section 3.3 we introduce our parallel algorithm based on random subsampling and with a single round of communication after every iteration. Our second contribution is to provide a detailed analysis of our parallel algorithm with communication and to compare the results with the one without communication. We show that the bias reduction is possible in the communication case by comparing the two algorithms after running a finite number of iterations with same number of machines as the number of iterations. Notice that, in both communication and non-communication case we employ random subsampling rather than random splitting of the data. Further in Zhang et al. (2013) the mean squared error (MSE) is considered for the parameter vector minimizing the population risk whereas in our case we consider the MSE of our estimates that approximates the maximum-likelihood

estimator (MLE). In other words the only randomness in our case is induced by the random subsampling and we assume that we have a fixed dataset of size N.
We present the two parallel algorithms in Section 3.3. We provide a brief review of the EM algorithm implementation in GMM in Section 3.5. Also, some convergence results related to EM in GMM is also discussed in section 3.6. Finally in Section 3.7 we provide an illustration of our algorithm in the case of parameter estimation in GMM and compare the two parallel algorithms. The empirical results validate that parallel implementation with communication does reduce the bias relative to the parallel scheme without communication. The proof of the technical results are deferred to the Appendix.

## 3.3 Optimization via parallel random subsampling

Consider a dataset $\mathcal{D}$ containing N observations where N typically would be very large. Assume that we have $T$ machines available for parallel computation and the iterative optimization is also run for $T$ iterations in each machine. We repeat drawing random subsample of size $m$ from the pool of $N$ observations $T$ times and then send them to the $T$ machines available. An optimization routine with a communication step after every iteration is then performed with the fraction of data available in each machine. The final estimate is obtained by taking a simple average of the estimates obtained from individual machines after running $T$ iterations. We now describe some notations useful for presenting the algorithm and the theoretical analysis thereafter. Let $S_1, S_2, \ldots, S_T$ be the subsamples in the respective machines. We also have $|S_i| = m$, for all $i = 1, 2, \ldots, T$. For any vector $U$ and any subsample $S$ we denote $U_S = \{u_i, i \in S\}$. Let $\theta \in \Theta \subset \mathbb{R}^p$ be the parameters in the model. We denote $\mathcal{M}_{i,j}(\theta) : \Theta \mapsto \Theta$ as a random iterative map in $i$th machine at the $j$th iteration. This is the approximate version of the true map $\mathcal{M}$ which can be used with the entire dataset $D$. The notation $\mathcal{M}_{i,j}$ indicates that the random map depends on the subsample $s_{i,j}$ which denotes the subsample in $i$th machine at the $j$th iteration. Here $s_{i,j} \in S = \{S_1, S_2, \ldots, S_T\}$. We associate the usual uniform norm (sup norm) with the notation $\|f\|$. For any real-valued function $f$ defined on a set S, the uniform norm (sup norm) is defined as

$$\|f\|_\infty = \|f\|_{\infty,S} = \sup\{\, |f(x)| : x \in S \,\}$$

In particular, for the case of a vector $x = (x_1, \ldots, x_n)$ in finite dimensional coordinate space, it takes the form

$$\|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\}$$

which we again simply denote as $\|x\|$. Further we use the notation $\theta_j^{(i)}$ to denote the estimate in the $i$th core at $j$th iteration and $\bar{\theta}_T$ as the final estimate averaged across $T$ machines. We desribe the steps of both the parallel algorithm with communication and without communication below.

**Algorithm 3.1** Parallel Optimization via Random Subsampling with communication(PORSWC)

```
Input: Data $D$ of size $N$, Number of machines $T$, subsamples $S = \{S_1, S_2, \ldots, S_T\}$
    each of size $m$, initial estimates $\left\{\theta_0^{(i)}\right\}_{i=1}^{T}$ in $T$ machines, Number of iterations $T$
Output: $\bar{\theta}_T = \frac{1}{T}\sum_{i=1}^{T}\theta_T^{(i)}$
procedure PORSWC$\left(D, N, T, S, m, \left\{\theta_0^{(i)}\right\}_{i=1}^{T}\right)$
loop:
    for $j = 1$ to $T$ do
loop:
        for each $i = 1$ to $T$ do
            $\theta_j^{(i)} = \mathcal{M}_{i,j}\left(\theta_{j-1}^{(i)}\right)$
        end                                        ▷ communication among machines
loop:
        for $i = 1$ to $(T-1)$ do
            $s_{i+1,j} \leftarrow s_{i,j}$
        end
        $s_{1,j} \leftarrow s_{T,j}$
    end
```

The type of communication that we have proposed in **Algorithm** 3.1 allows each initial estimate $\theta_0^{(i)}, i = 1, 2, \ldots, T$ to update itself through different subsample over each iteration. In practice when we implement **Algorithm** 3.1, rather than transferring the subsamples among the machines (which is a more expensive communication scheme) we transfer the estimates among the machines in the following manner: for

$i = 1$ to $(T-1)$

$$\theta_j^{(i+1)} \leftarrow \theta_j^{(i)} \text{ and}$$

$$\theta_j^{(1)} \leftarrow \theta_j^{(T)}$$

Nevertheless, the description in **Algorithm** 3.1 is useful for mathematical analysis of the algorithm. We show later in the theoretical analysis that this communication scheme reduces the bias of the final estimate $\bar{\theta}_T$ compared to a parallel implementation without communication. We describe the non-communication scheme below in detail.

**Algorithm 3.2** Parallel Optimization via Random Subsampling without communication(PORSWOC)

```
Input: Data $D$ of size $N$, Number of machines $T$, subsamples $S = \{S_1, S_2, \ldots, S_T\}$
    each of size $m$, initial estimates $\left\{\theta_0^{(i)}\right\}_{i=1}^{T}$ in $T$ machines, Number of iterations $T$
Output: $\bar{\theta}_T = \frac{1}{T}\sum_{i=1}^{T}\theta_T^{(i)}$
procedure PORSWOC$\left(D, N, T, S, m, \left\{\theta_0^{(i)}\right\}_{i=1}^{T}\right)$
loop:
    for $j = 1$ to $T$ do
loop:
        for each $i = 1$ to $T$ do
            $\theta_j^{(i)} = \mathcal{M}_{i,i}\left(\theta_{j-1}^{(i)}\right)$
        end
    end
```

## 3.4 Theoretical Results

### 3.4.1 Algorithm with Parallel Communication

For the communication scheme in each machine we provide new subsamples in every iteration. This allows initial estimate in each core to use different subsamples over the entire length of the optimization routine. We make assumption about contraction property of the true map $\mathcal{M}$ and the approximate one $\mathcal{M}_{i,j}$ below.

**Assumption 3.1.** [Contraction property of the approximate EM map] *Suppose for* $\lambda \in (0,1)$*, the map* $\mathcal{M}$ *and* $\mathcal{M}_{i,j}$ *satisfies*

$$\|\mathcal{M}(\theta) - \mathcal{M}(\theta')\| \leq \lambda\|\theta - \theta'\| \tag{3.1}$$

*and*

$$\|\mathcal{M}_{i,j}(\theta) - \mathcal{M}_{i,j}(\theta')\| \leq \lambda\|\theta - \theta'\| \tag{3.2}$$

Further, we make the following assumption on the unbiasedness of the approximate map $\mathcal{M}_{i,j}$

**Assumption 3.2.** [Unbiasedness of the approximate EM map] *For the approximate map $\mathcal{M}_{i,j}$ we assume,*

$$\mathbb{E}\left[\mathcal{M}_{i,j}(\theta)\right] = \mathcal{M}(\theta) \tag{3.3}$$

We define the following quantity which can be interpreted as the variance of the approximate random map $\mathcal{M}_{i.j}$ based on the iid random subsamples $S_1, S_2, \ldots, S_T$.

$$\sigma^2 \overset{\text{def}}{=} \mathbb{E}\left[\|\mathcal{M}_{i,j}(\theta) - \mathcal{M}(\theta)\|^2\right] \ , i, j = 1, 2, \ldots, T \tag{3.4}$$

We now state a theorem regarding the bias and the variance in the parallel algorithm with communication.

**Theorem 3.3.** *Assume* ***Assumption*** *3.1. Let us assume an implementation of our parallel algorithm* ***Algorithm*** *3.1 with $T$ machines and $T$ number of iterations starting from $\theta_0^{(i)} \in \Theta$ in ith machine, $i = 1, 2, \ldots, T$ and $\bar{\theta}_T \overset{def}{=} \frac{1}{T}\sum\limits_{i=1}^{T} \theta_T^{(i)}$. Then*

$$\left\|\mathbb{E}(\bar{\theta}_T) - \theta^\star\right\| \leq \frac{1}{T}\lambda^T \sum_{i=1}^{T} \left\|\theta_0^{(i)} - \theta^\star\right\| \tag{3.5}$$

*and*

$$\mathbb{V}ar\left(\bar{\theta}_T\right) \leq \frac{1}{T}\left(\frac{\sigma}{1-\lambda}\right)^2 \frac{1+\lambda}{1-\lambda}\left(1 - \lambda^T\right) \tag{3.6}$$

*Proof.* We start writing the update after $T$ iterations in the $l$th machine following **Algorithm** 3.1 as

$$\theta_T^{(l)} = \mathcal{M}_{l,T}(\mathcal{M}_{l,T-1}(\ldots\mathcal{M}_{l,1}(\theta_0^{(l)}))) \tag{3.7}$$

We can subtract the update obtained via the true map and write Eq.(3.7) using a telescoping sum in the following way

$$\begin{aligned}\theta_T^{(l)} = \mathcal{M}^T(\theta_0^{(l)}) + \mathcal{M}_{l,T}\ldots\mathcal{M}_{l,2}(\mathcal{M}_{l,1} - \mathcal{M})\theta_0^{(l)} \\ + \mathcal{M}_{l,T}\ldots\mathcal{M}_{l,3}(\mathcal{M}_{l,2} - \mathcal{M})\mathcal{M}\theta_0^{(l)} \\ + \ldots + \mathcal{M}_{l,T}(\mathcal{M}_{l,T-1} - \mathcal{M})\mathcal{M}^{T-2}\theta_0^{(l)} + (\mathcal{M}_{l,T} - \mathcal{M})\mathcal{M}^{T-1}\theta_0^{(l)} \quad (3.8)\end{aligned}$$

Hence we can write the above expression as

$$\theta_T^{(l)} = \mathcal{M}^T(\theta_0^{(l)}) + \sum_{j=1}^{T}\mathcal{M}_{l,T}\ldots\mathcal{M}_{l,j+1}(\mathcal{M}_{l,j} - \mathcal{M})\mathcal{M}^{j-1}\theta_0^{(l)} \quad (3.9)$$

Therefore,

$$\begin{aligned}&\mathbb{E}\left[\theta_T^{(l)}\middle|\,\{s_{l,r}\}_{r=j+1}^T\right] \\ &= \mathcal{M}^T(\theta_0^{(l)}) + \sum_{j=1}^{T}\mathcal{M}_{l,T}\ldots\mathcal{M}_{l,j+1}\mathbb{E}\left[(\mathcal{M}_{l,j} - \mathcal{M})\mathcal{M}^{j-1}\theta_0^{(l)}\middle|\,\{s_{l,r}\}_{r=j+1}^T\right] \quad (3.10)\end{aligned}$$

But since each machine uses a new subsample every iteration and the subsamples $S_1, \ldots, S_T$ are drawn independently we have,

$$\begin{aligned}\mathbb{E}\left[(\mathcal{M}_{l,j} - \mathcal{M})\mathcal{M}^{j-1}\theta_0^{(l)}\middle|\,\{s_{l,r}\}_{r=j+1}^T\right] &= \mathbb{E}\left[(\mathcal{M}_{l,j} - \mathcal{M})\mathcal{M}^{j-1}\theta_0^{(l)}\right] \\ &\overset{(i)}{=} \mathcal{M}^j\theta_0^{(l)} - \mathcal{M}\mathcal{M}^{j-1}(\theta_0^{(l)}) \\ &= 0\end{aligned}$$

Here $(i)$ follows from assumption **Assumption** 3.2. Hence from Eq. (3.10) we arrive at

$$\mathbb{E}\left(\theta_T^{(l)}\right) = \mathcal{M}^T(\theta_0^{(l)}) \quad (3.11)$$

Therefore we can write

$$\begin{aligned}\left\|\mathbb{E}(\bar{\theta}_T)-\theta^\star\right\| &= \left\|\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}(\theta_T^{(i)})-\theta^\star\right\| \\ &= \left\|\frac{1}{T}\sum_{i=1}^{T}\mathcal{M}^T(\theta_0^{(i)})-\mathcal{M}^T(\theta^\star)\right\| \\ &\overset{(i)}{\leq} \frac{1}{T}\lambda^T\sum_{i=1}^{T}\left\|\theta_0^{(i)}-\theta^\star\right\|\end{aligned}$$

where in inequality $(i)$ we used contraction property of the map $\mathcal{M}$.

Let's now look at the variance term i.e.

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left(\bar{\theta}_T\right) &= \frac{1}{T^2}\mathbb{V}\mathrm{ar}\left(\sum_{i=1}^{T}\theta_T^{(i)}\right) \\ &= \frac{1}{T^2}\sum_i\sum_j\mathbb{C}\mathrm{ov}\left(\theta_T^{(i)},\theta_T^{(j)}\right) \\ &= \sum_{i=1}^{T}\sum_{k=1}^{T}\mathbb{C}\mathrm{ov}\left(\theta_T^{(i)},\theta_T^{(i+k)}\right) \qquad (3.12)\end{aligned}$$

Here we identify index $(T+k)$ as $(T+k)\equiv\operatorname{mod}(T+k,T)=k, k=1,2,\ldots,T$ according to our parallel scheme of communication.

Let us look at the covariance term first and consider $j=i+k$ where $k=1,2,\ldots,T$. We can write

$$\mathbb{C}\mathrm{ov}\left(\theta_T^{(i)},\theta_T^{(i+k)}\right)=\mathbb{E}\left[\theta_T^{(i)}-\mathbb{E}\left(\theta_T^{(i)}\right)\right]\left[\theta_T^{(i+k)}-\mathbb{E}\left(\theta_T^{(i+k)}\right)\right]' \qquad (3.13)$$

Now using $\mathbb{E}\left(\theta_T^{(l)}\right)=\mathcal{M}^T(\theta_0^{(l)})$ from the first part of the proof we have

$$\begin{aligned}&\mathbb{E}\left[\theta_T^{(i)}-\mathbb{E}\left(\theta_T^{(i)}\right)\right]\left[\theta_T^{(i+k)}-\mathbb{E}\left(\theta_T^{(i+k)}\right)\right]' \\ &=\sum_{j_1=1}^{T}\sum_{j_2=1}^{T}\mathbb{E}\left[C_{i,j_1}^{\mathcal{M}}\left(C_{i+k,j_2}^{\mathcal{M}}\right)'\right] \qquad (3.14)\end{aligned}$$

where

$$C_{i,j_1}^{\mathcal{M}}=\mathcal{M}_{i,T}\ldots\mathcal{M}_{i,j_1+1}(\mathcal{M}_{i,j_1}-\mathcal{M})\tilde{\theta}_{j_1-1}^{(i)}$$

where $\tilde{\theta}_r^{(s)}=\mathcal{M}^r(\theta_0^{(s)})$, $r=0,1,\ldots,T-1$ and $s=1,2,\ldots,T$

Consider the following diagram that explains which subsamples the estimates in machine $i$ and $i+k$ uses over the iterations respectively.

$$i : \overbrace{i \to i+1 \to i+2 \to \ldots \to i+k}^{j_1} \to \ldots \to T \to 1 \to 2 \to i-1$$

$$i : \underbrace{i+k \to i+k+1 \to i+k+2 \to \ldots \to T \to 1 \to 2 \to \ldots \to i}_{j_2} \to \ldots \to i+k-1$$

Hence for $j_1 \geq k+1$ and $j_2 \geq (T-k+1)$,

$$\mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right] \overset{(i)}{=} \mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\right]\mathbb{E}\left[C^{\mathcal{M}}_{i+k,j_2}\right] \overset{(ii)}{=} 0 \tag{3.15}$$

where $(i)$ follows from the fact that estimates in machine $i$ and $i+k$ goes through independent subsamples for $j_1 \geq k+1$ and $j_2 \geq (T-k+1)$ and $(ii)$ utilizes **Assumption** 3.2 with the conditioning argument shown in Eq. (3.10). Therefore we can write

$$\mathbb{C}\mathrm{ov}\left(\theta_T^{(i)}, \theta_T^{(i+k)}\right) = \sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right] + \sum_{j_1=k+1}^{T}\sum_{j_2=1}^{T-k}\mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right] \tag{3.16}$$

Considering the first term in Eq. (3.16) we can write

$$\begin{aligned}
&\sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right] \qquad (3.17)\\
&\leq \sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\mathbb{E}\left[\left|C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right|\right]\\
&\overset{(i)}{\leq} \sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\mathbb{E}\left[\lambda^{T-j_1}\|\mathcal{M}_{i,j_1}(\tilde{\theta}^{(i)}_{j_1-1}) - \mathcal{M}(\tilde{\theta}^{(i)}_{j_1-1})\|\lambda^{T-j_2}\|\mathcal{M}_{i+k,j_2}(\tilde{\theta}^{(i+k)}_{j_2-1}) - \mathcal{M}(\tilde{\theta}^{(i+k)}_{j_2-1})\|\right]\\
&= \sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\lambda^{T-j_1}\lambda^{T-j_2}\mathbb{E}\left[\|\mathcal{M}_{i,j_1}(\tilde{\theta}^{(i)}_{j_1-1}) - \mathcal{M}(\tilde{\theta}^{(i)}_{j_1-1})\|\|\mathcal{M}_{i+k,j_2}(\tilde{\theta}^{(i+k)}_{j_2-1}) - \mathcal{M}(\tilde{\theta}^{(i+k)}_{j_2-1})\|\right]\\
&\overset{(ii)}{\leq} \sigma^2\sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\lambda^{T-j_1}\lambda^{T-j_2} \qquad (3.18)
\end{aligned}$$

where $(i)$ follows from repeated use of **Assumption** 3.1, $(ii)$ is obtained by Cauchy-Schwarz inequality and Eq.(3.4). Similar algebraic calculations for the second term in Eq.(3.16) yields

$$\sum_{j_1=k+1}^{T}\sum_{j_2=1}^{T-k}\mathbb{E}\left[C^{\mathcal{M}}_{i,j_1}\left(C^{\mathcal{M}}_{i+k,j_2}\right)'\right] \leq \sigma^2 \sum_{j_1=k+1}^{T}\sum_{j_2=1}^{T-k}\lambda^{T-j_1}\lambda^{T-j_2} \tag{3.19}$$

Hence using Eq.(3.18) and Eq.(3.19) in Eq.(3.16) we have

$$\begin{aligned}\mathbb{C}\text{ov}\left(\theta_T^{(i)},\theta_T^{(i+k)}\right) &\leq \sigma^2\left(\sum_{j_1=1}^{k}\sum_{j_2=1}^{T}\lambda^{T-j_1}\lambda^{T-j_2} + \sum_{j_1=k+1}^{T}\sum_{j_2=1}^{T-k}\lambda^{T-j_1}\lambda^{T-j_2}\right)\\ &\leq \left(\frac{\sigma}{1-\lambda}\right)^2\left(\lambda^{T-k}+\lambda^k\right) \end{aligned}\tag{3.20}$$

Now from Eq.(3.12) we derive

$$\begin{aligned}\mathbb{V}\text{ar}\left(\bar{\theta}_T\right) &\leq \frac{1}{T^2}\left(\frac{\sigma}{1-\lambda}\right)^2\sum_{i=1}^{T}\sum_{k=1}^{T}\left(\lambda^{T-k}+\lambda^k\right)\\ &= \frac{1}{T^2}\left(\frac{\sigma}{1-\lambda}\right)^2(1+\lambda)T\frac{1-\lambda^T}{1-\lambda}\\ &= \frac{1}{T}\left(\frac{\sigma}{1-\lambda}\right)^2\frac{1+\lambda}{1-\lambda}\left(1-\lambda^T\right)\end{aligned}\tag{3.21}$$

Hence the proof.

□

### 3.4.2 Parallel Algorithm without Communication

Since for the non-communication scheme the subsamples in any machine are kept same over the iterations we can simplify the notations $\mathcal{M}_{i,j}$ as $\mathcal{M}_i$ where $i$ stands for the machine $i$, $i = 1, 2, \ldots, T$. The subsample $S_i$ is used in core $i$ ($i = 1, 2, \ldots, T$) through out the iterations. We now state a theorem regarding the bias and the variance in the parallel algorithm without communication.

**Theorem 3.4.** *Assume* ***Assumption*** *3.1. Let us assume an implementation of our parallel algorithm without communication (**Algorithm** 3.2) with $T$ machines and $T$ number of iterations starting from $\theta_0^{(i)} \in \Theta$ in ith machine, $i = 1, 2, \ldots, T$ and $\bar{\theta}_T \overset{def}{=} \frac{1}{T}\sum_{i=1}^{T}\theta_T^{(i)}$. Then*

$$\left\|\mathbb{E}(\bar{\theta}_T)-\theta^\star\right\| \leq \frac{1}{T}\lambda^T\sum_{i=1}^{T}\|\theta_0^{(i)}-\theta^\star\| + \sigma\frac{1-\lambda^T}{1-\lambda} \tag{3.22}$$

*and*

$$\mathbb{V}ar\left(\bar{\theta}_T\right) \leq \frac{4}{T}\left(\frac{\sigma}{1-\lambda}\right)^2(1-\lambda^T)^2 \tag{3.23}$$

*Proof.* We start by writing the update after $T$ iterations in any machine $l$, $l = 1, 2, \ldots, T$ following **Algorithm** 3.2

$$\theta_T^{(l)} = \mathcal{M}_l(\theta_{T-1}^{(l)}) \tag{3.24}$$

Iterating this update over $T-1, T-2, \ldots, 1$ and using a telescoping sum we have

$$\theta_T^{(l)} = \mathcal{M}^T(\theta_0^{(i)}) + \sum_{j=1}^{T} \mathcal{M}_l^{T-j}(\mathcal{M}_l - \mathcal{M})\mathcal{M}^{j-1}(\theta_0^{(l)}) \tag{3.25}$$

Taking expectation on both sides of Eq. (3.25) we get

$$\mathbb{E}(\theta_T^{(i)}) = \mathcal{M}^T(\theta_0^{(l)}) + \sum_{j=1}^{T} \mathbb{E}\left[\mathcal{M}_l^{T-j}(\mathcal{M}_l - \mathcal{M})\mathcal{M}^{j-1}(\theta_0^{(l)})\right] \tag{3.26}$$

Hence the bias of this parallel algorithm can be formalized as

$$\begin{aligned}
&\left\|\mathbb{E}(\bar{\theta}_T) - \theta^\star\right\| \\
&= \left\|\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}(\theta_T^{(i)}) - \theta^\star\right\| \\
&= \left\|\frac{1}{T}\sum_{i=1}^{T}\left(\mathcal{M}^T(\theta_0^{(i)}) + \sum_{j=1}^{T}\mathbb{E}\left[\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\mathcal{M}^{j-1}(\theta_0^{(i)})\right]\right) - \mathcal{M}^T(\theta^\star)\right\| \\
&\overset{(i)}{\leq} \frac{1}{T}\lambda^T\sum_{i=1}^{T}\|\theta_0^{(i)} - \theta^\star\| + \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{T}\mathbb{E}\|\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\mathcal{M}^{j-1}(\theta_0^{(i)})\| \\
&\overset{(ii)}{\leq} \frac{1}{T}\lambda^T\sum_{i=1}^{T}\|\theta_0^{(i)} - \theta^\star\| + \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{T}\lambda^{T-j}\mathbb{E}\|\mathcal{M}_i(\tilde{\theta}_{j-1}^{(l)}) - \mathcal{M}(\tilde{\theta}_{j-1}^{(l)})\| \\
&\overset{(iii)}{\leq} \frac{1}{T}\lambda^T\sum_{i=1}^{T}\|\theta_0^{(i)} - \theta^\star\| + \sigma\frac{1-\lambda^T}{1-\lambda}
\end{aligned} \tag{3.27}$$

where $(i)$ follows from triangle inequality, $(ii)$ follows from repeated use of **Assumption** 3.1 and we use $\tilde{\theta}_r^{(s)} = \mathcal{M}^r(\theta_0^{(s)})$, $r = 0, 1, \ldots, T-1$ and $s = 1, 2, \ldots, T$. Further in $(iii)$ we use Eq. (3.4).

Let us now look at the variance term i.e.

$$\mathbb{V}\text{ar}\left(\bar{\theta}_T\right) \overset{(i)}{=} \frac{1}{T^2}\sum_{i=1}^{T}\mathbb{V}\text{ar}\left(\theta_T^{(i)}\right) \tag{3.28}$$

Here $(i)$ follows from the fact that the estimates $\theta_T^{(i)}$ in each machine $i$ are based on independent random subsample. Now

$$\mathbb{V}\text{ar}\left(\theta_T^{(i)}\right) = \mathbb{E}\left[\theta_T^{(i)} - \mathbb{E}(\theta_T^{(i)})\right]\left[\theta_T^{(i)} - \mathbb{E}(\theta_T^{(i)})\right]' \tag{3.29}$$

Using Eq. (3.25) and Eq. (3.26) we can write

$$\left[\theta_T^{(i)} - \mathbb{E}(\theta_T^{(i)})\right] = \sum_{j=1}^{T}\left[\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\tilde{\theta}_{j-1} - \mathbb{E}(\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\tilde{\theta}_{j-1})\right] \tag{3.30}$$

Therefore

$$\begin{aligned}
&\left|\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\tilde{\theta}_{j-1} - \mathbb{E}(\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\tilde{\theta}_{j-1})\right| \\
&\overset{(i)}{\leq} \lambda^{T-j}\|\mathcal{M}_i(\tilde{\theta}_{j-1}) - \mathcal{M}(\tilde{\theta}_{j-1})\| + \mathbb{E}\|\mathcal{M}_i^{T-j}(\mathcal{M}_i - \mathcal{M})\tilde{\theta}_{j-1}^{(i)}\| \\
&\overset{(ii)}{\leq} \lambda^{T-j}\|\mathcal{M}_i(\tilde{\theta}_{j-1}) - \mathcal{M}(\tilde{\theta}_{j-1})\| + \lambda^{T-j}\sigma
\end{aligned} \tag{3.31}$$

where $(i)$ follows from repeated use of **Assumption** 3.1 and $(ii)$ uses Eq. (3.4). Let us denote

$$V_{i,j}(\sigma) = \|\mathcal{M}_i(\tilde{\theta}_{j-1}) - \mathcal{M}(\tilde{\theta}_{j-1})\| + \sigma$$

Hence

$$\begin{aligned}
&\mathbb{E}\left[\theta_T^{(i)} - \mathbb{E}(\theta_T^{(i)})\right]\left[\theta_T^{(i)} - \mathbb{E}(\theta_T^{(i)})\right]' \\
&\leq \sum_{j_1=1}^{T}\sum_{j_2=1}^{T} \lambda^{T-j_1}\lambda^{T-j_2}\mathbb{E}\left[V_{i,j_1}(\sigma)V_{i,j_2}(\sigma)\right] \\
&\overset{(i)}{\leq} 4\sigma^2 \sum_{j_1=1}^{T}\sum_{j_2=1}^{T} \lambda^{T-j_1}\lambda^{T-j_2} \\
&= 4\sigma^2 \sum_{j_1=1}^{T} \lambda^{T-j_1} \sum_{j_2=1}^{T} \lambda^{T-j_2} \\
&= 4\left(\frac{\sigma}{1-\lambda}\right)^2 \left(1-\lambda^T\right)^2 \qquad (3.32)
\end{aligned}$$

where $(i)$ follows from Cauchy-Schwarz inequality and Eq. (3.4). Therefore from Eq. (3.28) we have

$$\mathbb{V}\mathrm{ar}\left(\bar{\theta}_T\right) \leq \frac{4}{T}\left(\frac{\sigma}{1-\lambda}\right)^2 \left(1-\lambda^T\right)^2$$

Hence the proof.

□

### 3.4.3 Discussion of Results of Theorem 3.3 and 3.4

We first make a comparison of the bias term for the two parallel schemes. From Theorem 3.3 we have,

$$\left\|\mathbb{E}(\bar{\theta}_T) - \theta^\star\right\| \leq \frac{1}{T}\lambda^T \sum_{i=1}^{T} \left\|\theta_0^{(i)} - \theta^\star\right\| \qquad (3.33)$$

On the other hand, Theorem 3.4 yields

$$\left\|\mathbb{E}(\bar{\theta}_T) - \theta^\star\right\| \leq \frac{1}{T}\lambda^T \sum_{i=1}^{T} \|\theta_0^{(i)} - \theta^\star\| + \sigma\frac{1-\lambda^T}{1-\lambda} \qquad (3.34)$$

Looking at Eq. (3.33) and Eq. (3.34) we observe as T becomes large , R.H.S. of Eq. (3.33) tends to zero whereas R.H.S. of Eq. (3.34) tends to $\frac{\sigma}{1-\lambda}$ which clearly highlights the gain of **Algorithm** 3.1 over **Algorithm** 3.2 in reducing the bias.
Now let us look at the two variance terms. From Theorem 3.3 we have,

$$\mathbb{V}\text{ar}\left(\bar{\theta}_T\right) \leq \frac{1}{T}\left(\frac{\sigma}{1-\lambda}\right)^2 \frac{1+\lambda}{1-\lambda}\left(1-\lambda^T\right) \tag{3.35}$$

Theorem 3.4 tells that

$$\mathbb{V}\text{ar}\left(\bar{\theta}_T\right) \leq \frac{1}{T}4\left(\frac{\sigma}{1-\lambda}\right)^2 \left(1-\lambda^T\right)^2 \tag{3.36}$$

Therefore

$$\begin{aligned}
&\text{R.H.S. of 3.35} < \text{R.H.S. of 3.36}\\
&\Rightarrow \frac{1+\lambda}{1-\lambda}\left(1-\lambda^T\right) < 4\left(1-\lambda^T\right)^2\\
&\Rightarrow \frac{1+\lambda}{1-\lambda} < 4\left(1-\lambda^T\right) \qquad (3.37)
\end{aligned}$$

Since $\lambda \in (0,1)$, for large T we have from Eq. (3.37),

$$\begin{aligned}
&\frac{1+\lambda}{1-\lambda} < 4\\
&\Rightarrow 5\lambda < 3\\
&\Rightarrow \lambda < 0.6 \qquad (3.38)
\end{aligned}$$

Therefore for large $T$, variance of the parallel scheme with communication is smaller than the variance of the non-communication scheme if $\lambda < 0.6$.

## 3.5 An application: EM Algorithm for Gaussian Mixture Model

The EM algorithm is an iterative algorithm used for maximum likelihood (ML) or maximum a posteriori (MAP) estimation. EM and it's several variants have been popular in parameter estimation in latent-variable models (see Dempster et al. (1977); Wei and Tanner (1990); Nielsen (2000) etc.). EM has been used for parameter estimation in Gaussian Mixture Model (GMM) as well (see Ghahramani and Jordan (1994); Nowlan (1991); Xu and Jordan (1993b,a); Tresp et al. (1994); Redner and Walker (1984); Xu and Jordan (1996); Ma et al. (2000) etc.). GMM is a popular tool for clustering data (McLachlan & Peal(2000)). It models the data as a mixture of multiple Gaussian distributions where each Gaussian component corresponds to one cluster. Let $\mathcal{D} = \{x_n, n = 1, 2, \ldots, N\}$ be $N$ iid observations obtained from a mixture model whose components are $d$-dimensional Gaussian distribution. The observations are assumed iid from the following model

$$p(x_n|\mu, \Sigma) = \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \tag{3.39}$$

where $f\left(x_n|\mu_i, \Sigma_i\right) = \frac{1}{(2\pi)^{\frac{m}{2}} |\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}\left(x_n - \mu_i\right)^T \Sigma_i^{-1}\left(x_n - \mu_i\right)\right]$. K is the number of mixture components. $\mu_i, \Sigma_i,\ i = 1, 2$ are the mean and the covariance matrix for the $i$th mixture component. $\pi_i$ is the mixing proportion for the $i$th component. The objective is to estimate the parameters $\{\pi_i, \mu_i, \Sigma_i\}_{i=1}^{K}$ of model (3.39). The log-likelihood for the observed data is given by

$$\begin{aligned} l\left(\theta|\mathcal{D}\right) &= \sum_{n=1}^{N} \log p\left(x_n|\mu, \Sigma\right) \\ &= \sum_{n=1}^{N} \log \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \end{aligned} \tag{3.40}$$

The above log-likelihood can be optimized via the following iterative algorithm (See Dempster et.al. (1977)):

$$\begin{aligned} \pi_i^{(t+1)} &= \frac{1}{N} \sum_{n=1}^{N} \tau_n^{i(t)} \qquad (3.41) \\ \mu_i^{(t+1)} &= \frac{\sum_{n=1}^{N} \tau_n^{i(t)} x_n}{\sum_{n=1}^{N} \tau_n^{i(t)}} \\ \Sigma_i^{(t+1)} &= \frac{\sum_{n=1}^{N} \tau_n^{i(t)} \left(x_n - \mu_i^{(t+1)}\right)\left(x_n - \mu_i^{(t+1)}\right)^T}{\sum_{n=1}^{N} \tau_n^{i(t)}} \end{aligned}$$

where the posterior probabilities are defined as follows:

$$\tau_n^{i(t)} \overset{def}{=} \frac{\pi_i^{(t)} f\left(x_n|\mu_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_{j=1}^{K} \pi_j^{(t)} f\left(x_n|\mu_j^{(t)}, \Sigma_j^{(t)}\right)} \tag{3.42}$$

### 3.5.1 Derivation of EM Algorithm for GMM

EM algorithm has been employed for successful analysis of GMM for heterogeneous data. See Ghahramani and Jordan (1994); Nowlan (1991); Xu and Jordan (1993b,a) etc. The computational complexity for computing the GMM likelihood is $\mathcal{O}(nKp^2)$ which for a fixed number of clusters K, grows with large number of observations and increasing dimension of the feature space. Hence applying EM to the entire dataset becomes computationally infeasible. Therefore, we use **Algorithm** 3.1 to overcome the large size of the data and implement EM on random subsamples (much smaller size then the full data) in each machine with a communication step in every iteration. For completeness, we describe here the derivation of the EM updates given in Eq. (3.41). First we take the derivative of Eq. (3.40) with respect to $\mu_i$:

$$\frac{\partial l}{\partial \mu_i} = \frac{\partial}{\partial \mu_i} \left\{ \sum_{n=1}^{N} \log \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \right\} \tag{3.43}$$

$$= \sum_{n=1}^{N} \frac{\pi_i f\left(x_n|\mu_i, \Sigma_i\right)}{\sum_{j=1}^{K} \pi_j f\left(x_n|\mu_j, \Sigma_j\right)} \frac{\partial}{\partial \mu_i} \log f\left(x_n|\mu_i \Sigma_i\right) \tag{3.44}$$

$$= \sum_{n=1}^{N} \tau_n^i \frac{\partial}{\partial \mu_i} \log f\left(x_n|\mu_i \Sigma_i\right) \tag{3.45}$$

$$= \sum_{n=1}^{N} \tau_n^i \Sigma_i^{-1} \left(x_n - \mu_i\right) \tag{3.46}$$

where in Eq. (3.45) we use definition of $\tau_n^i$ as given in Eq. (3.42). Setting to zero yields

$$\mu_i = \frac{\sum_{n=1}^{N} \tau_n^i x_n}{\sum_{n=1}^{N} \tau_n^i} \tag{3.47}$$

at a stationary point of the log-likelihood. A very similar calculations yield the following condition for the covariance matrices

$$\Sigma_i = \frac{\sum_{n=1}^{N} \tau_n \left(x_n - \mu_i\right)\left(x_n - \mu_i\right)^T}{\sum_{n=1}^{N} \tau_n^i} \tag{3.48}$$

and the mixing proportions

$$\pi_i = \frac{1}{N} \sum_{n=1}^{N} \tau_n^i \tag{3.49}$$

where in the latter case we use Lagrange multipliers. These equations certainly do not constitute an explicit solution since the posterior probabilities are themselves functions of the parameters and so equations (3.47), (3.48) and (3.49) constitute a system of coupled non-linear equations. We try to solve these system of equations iteratively as given in Eq. (3.41). To connect with the usual E and the M-step of the EM algorithm we explain the arguments in the following paragraphs.

In the EM algorithm generally E-step is defined as the "Calculation of complete data log-likelihood" and the M-step amounts to maximization of the expected complete data log-likelihood with respect to the parameters. Let us denote

$$\mathcal{D}_c = \{(x_n, z_n) : n = 1, 2, \ldots, N\}$$

to be the complete data for the GMM. We also denote $\theta = (\mu, \Sigma)$. The complete data log-likelihood is given by

$$\begin{aligned} l_c(\theta|D_c) &= \sum_{n=1}^{N} \log p\left(x_n, z_n|\theta\right) && (3.50)\\ &= \sum_{n=1}^{N} \prod_{i=1}^{K} \left[\pi f\left(x_n|\mu_i, \Sigma_i\right)\right]^{z_n^i} && (3.51)\\ &= \sum_{n=1}^{N} \sum_{i=1}^{K} z_n^i \log\left[\pi_i f\left(x_n|\mu_i, \Sigma_i\right)\right] && (3.52) \end{aligned}$$

One can clearly observe the difference between this log-likelihood and the one in Eq. (3.40). We repeat that here for convenience

$$l\left(\theta|\mathcal{D}\right) = \sum_{n=1}^{N} \log \sum_{i=1}^{K} \pi_i f\left(x_n|\mu_i, \Sigma_i\right) \tag{3.53}$$

In the latter log is outside the sum over i which reflects that it is a marginal probability. The complete data log-likelihood, on the other hand is not a marginal probability and hence the log is inside the sum. Since we do not know the latent variables $Z_n$, the next step is to compute the conditional expectation of the latent variables given the data $X_n$ and fixing the parameter $\theta$ to a particular value $\theta^{(t)}$. Using the operator notation $\langle . \rangle_{\theta^{(t)}}$ to denote these conditional expectations we define the expected complete data log-likelihood as

$$\begin{aligned}
\langle l_c\left(\theta|\mathcal{D}_c\right)\rangle_{\theta^{(t)}} &= \Big\langle \sum_{n=1}^{N} \sum_{i=1}^{K} z_n^i \log\left[\pi_i f\left(x_n|\mu_i, \Sigma_i\right)\right] \Big\rangle_{\theta^{(t)}} && (3.54)\\
&= \sum_{n=1}^{N} \sum_{i=1}^{K} \langle z_n^i \rangle_{\theta^{(t)}} \log\left[\pi_i f\left(x_n|\mu_i, \Sigma_i\right)\right] && (3.55)\\
&= \sum_{n=1}^{N} \sum_{i=1}^{K} \tau_n^{i(t)} \log\left[\pi_i f\left(x_n|\mu_i, \Sigma_i\right)\right] && (3.56)
\end{aligned}$$

The M-step requires maximization of the expected complete data log-likelihood in Eq. (3.56) with respect to the parameters. Let us first consider the update for the means. Collecting terms that involve $\mu_i$ in 3.56 and writing as $J(\mu_i)$ we obtain:

$$J(\mu_i) = -\frac{1}{2} \sum_{n=1}^{N} \tau_n^{i(t)} \left(x_n - \mu_i\right)^T \Sigma_i^{-1} \left(x_n - \mu_i\right) \tag{3.57}$$

Notice that this is a weighted least-squares problem. Calculating the derivative of $J(\mu_i)$ with respect to $\mu_i$ and setting to zero yields:

$$\mu_i^{(t+1)} = \frac{\displaystyle\sum_{n=1}^{N} \tau_n^{i(t)} x_n}{\displaystyle\sum_{n=1}^{N} \tau_n^{i(t)}} \tag{3.58}$$

which is same as the update for $\mu$ in Eq. (3.41) and one can also see that it is identical with the Eq. (3.47).

Similarly, collecting together terms that reference the covariance matrix $\Sigma_i$ in Eq. (3.56) we have

$$J(\Sigma_i) = -\frac{1}{2}\sum_{n=1}^{N} \tau_n^{i(t)} \left\{ \log|\Sigma_i| + (x_n - \mu_i)^T \Sigma_i^{-1} (x_n - \mu_i) \right\} \tag{3.59}$$

This is the weighted variant of the problem of estimating the covariance matrix of a multivariate Gaussian. Taking the derivative with respect to $\Sigma_i$ and setting it to zero yields:

$$\Sigma_i^{(t+1)} = \frac{\sum_{n=1}^{N} \tau_n^{i(t)} \left(x_n - \mu_i^{(t+1)}\right)\left(x_n - \mu_i^{(t+1)}\right)^T}{\sum_{n=1}^{N} \tau_n^{i(t)}} \tag{3.60}$$

which is the update for $\Sigma$ in Eq. (3.41). Again one can also observe the similarity of Eq. (3.60) and Eq. (3.48).

Finally the terms in the expected complete data log-likelihood that reference $\pi$ are:

$$J(\pi) = \sum_{n=1}^{N}\sum_{i=1}^{K} \tau_n^{i(t)} \log \pi_i \tag{3.61}$$

Adding a Lagrangian term to account for the constraint that $\sum_{i=1}^{K} \pi_i = 1$, taking derivatives and setting to zero yields:

$$\pi_i^{(t+1)} = \frac{1}{N}\sum_{n=1}^{N} \tau_n^{i(t)} \tag{3.62}$$

which is the update for $\pi$ in Eq. (3.41) and coincides with Eq. (3.49) as well.

## 3.6 Review of Convergence of EM in GMM

The EM algorithm is guaranteed to monotonically converge to local optima under mild continuity conditions (See Dempster et al. (1977); Wu (1983)). Redner and Walker (1984) show that EM has linear rate of convergence. Xu and Jordan (1996) provides some insight into the convergence rate of EM in the setting of Gaussian mixtures. For the convenience of mathematical analyses, they studied a variant of the original EM algorithm for Gaussian mixtures and showed that the condition number associated with this variant EM algorithm is guaranteed to be smaller than the condition number associated with gradient ascent, providing a general guarantee of the dominance of this variant EM algorithm over the gradient algorithm.They forge a connection between the EM algorithm and gradient ascent and prove that rate of convergence of the EM algorithm depends on the condition number of a projected Hessian matrix $E^T P(\Theta^*)H(\Theta^*)E$ where $\Theta^*$ is the optimum parameter value. $E = [e_1, \ldots, e_m]$ is a set of unit basis vectors spanning the constrained parameter space (satisfying the constraint $\sum_{j=1}^{K} \alpha_j = 1$). $P(\Theta^*)$ is a projection matrix, and $H(\Theta^*)$ is the Hessian of the log-likelihood function. Moreover, in cases in which the mixture components are well separated, they showed that the condition number for this EM algorithm approximately converges to one, corresponding to a local superlinear convergence rate. Thus, in this restrictive case,this type of EM algorithm has the favorable property of showing quasi-Newton behavior as it nears the ML or MAP solution. Xu (1997) further showed that the original EM algorithm has the same convergence properties as this variant EM algorithm.

### 3.6.1 Identifying true Map $\mathcal{M}$ and the approximate map $\mathcal{M}_{i,j}$

We discuss here identifying the true map $M$ and the approximate map $M_{i,j}$ which will be necessary for understanding **Algorithm** 3.1 and **Algorithm** 3.2 as well as comprehending the theoretical results in Theorem 3.3 and Theorem 3.4 repectively. We follow the proof of Theorem 1 in Xu and Jordan (1996). This theorem illustrates the connection between EM and gradient ascent algorithm. This connection helps us to identify map $M$ and $M_{i,j}$

**Theorem 3.5.** *(Theorem 1 of Xu and Jordan, 1996) At each iteration of the EM algorithm we have*

$$\pi^{(t+1)} - \pi^{(t)} = P_{\pi}^{(t)} \frac{\partial l}{\partial \pi}\bigg|_{\pi=\pi^{(t)}} \tag{3.63}$$

$$\mu_j^{(t+1)} - \mu_j^{(t)} = P_{\mu_j}^{(t)} \frac{\partial l}{\partial \mu_j}\bigg|_{\mu_j=\mu_j^{(t)}} \tag{3.64}$$

$$vec\left[\Sigma_j^{(t+1)}\right] - vec\left[\Sigma_j^{(t)}\right] = P_{\Sigma_j}^{(t)} \frac{\partial l}{\partial \Sigma_j}\bigg|_{\Sigma_j=\Sigma_j^{(t)}} \tag{3.65}$$

*where*

$$P_\pi^{(t)} = \frac{1}{N}\left\{diag\left[\pi_1^{(t)}, \ldots, \pi_K^{(t)}\right] - \pi^{(t)}(\pi^{(t)})^T\right\} \tag{3.66}$$

$$P_{\mu_j}^{(t)} = \frac{\Sigma_j^{(t)}}{\sum_{n=1}^{N} \tau_n^{j(t)}} \tag{3.67}$$

$$P_{\Sigma_j}^{(t)} = \frac{2}{\sum_{n=1}^{N} \tau_n^{j(t)}} \left(\Sigma_j^{(t)} \otimes \Sigma_j^{(t)}\right) \tag{3.68}$$

*where $\pi$ denotes the vector of mixing proportions $[\pi_1, \ldots, \pi_K]^T$, $j$ indexes the mixture components ($j = 1, 2, \ldots, K$), $k$ denotes the iteration number, "vec$[B]$" is defined as the vectors obtained by stacking the column vectors of matrix $B$, $\otimes$ denotes the Kronecker product. Moreover given the constraints $\sum_{j=1}^{K} \pi_j^{(t)} = 1$ and $\pi_j^{(t)} \geq 0$, $P_\pi^{(t)}$ is positive definite matrix and the matrices $P_{\mu_j}^{(t)}$ and $P_{\Sigma_j}^{(t)}$ are positive definite matrices with probability one for $N$ sufficiently large.*

*Proof.* We start by looking at the update for the mixing proportions. Using Eq. 3.40 we have

$$\frac{\partial l}{\partial \pi}\bigg|_{\pi=\pi^{(t)}} = \sum_{n=1}^{N} \frac{\left[f(x_n|\theta_1^{(t)}, \ldots, f(x_n|\theta_K^{(t)}\right]^T}{\sum_{i=1}^{K} \pi_i^{(t)} f(x_n|\theta_i^{(t)})}$$

where $\theta_j^{(t)} = \left(\mu_j^{(t)}, \Sigma_j^{(t)}\right)$ for $j = 1, 2, \ldots, K$. Now premultiplying the above by $P_\pi^{(t)}$ we obtain

$$P_\pi^{(t)} \frac{\partial l}{\partial \pi}\bigg|_{\pi=\pi^{(t)}} = \frac{1}{N} \sum_{n=1}^{N} \left[\tau_n^{1(t)}, \ldots, \tau_n^{K(t)}\right]^T - \pi^{(t)}$$

Thus update formula for $\pi$ in Eq. 3.41 can be rewritten as

$$\pi^{(t+1)} = \pi^{(t)} + \frac{1}{N} \sum_{n=1}^{N} \left[\tau_n^{1(t)}, \ldots, \tau_n^{K(t)}\right]^T - \pi^{(t)}$$

Combining the last two equations establish the update rule for $\pi$ in Eq. 3.63. Furthermore for an arbitrary vector $u$, we have $Nu^T P_\pi^{(t)} u = u^T \text{diag}\left[\pi_1^{(t)}, \ldots, \pi_K^{(t)}\right] u -$

$\left(u^T\pi^{(t)}\right)^2$. Using Jensen's inequality we have

$$\begin{aligned} u^T \text{diag}\left[\pi_1^{(t)}, \ldots, \pi_K^{(t)}\right] u &= \sum_{j=1}^{K} \pi_j^{(t)} u_j^2 \\ &> \left(\sum_{j=1}^{K} \pi_j^{(t)} u_j\right)^2 \\ &= \left(u^T \pi^{(t)}\right)^2 \end{aligned}$$

Thus $u^T P_\pi^{(t)} u > 0$ and $P_\pi^{(t)}$ is positive definite given the constraints $\sum_{j=1}^{K} \pi_j^{(t)} = 1$ and $\pi_j^{(t)} \geq 0$ for all $j$.

We now consider the update for the means $\mu_j$. Notice that from Eq. 3.40 we can write

$$\left.\frac{\partial l}{\partial \mu_j}\right|_{\mu_j=\mu_j^{(t)}} = \sum_{n=1}^{N} \tau_n^{j(t)} (\Sigma_j^{(t)})^{-1} \left(x_n - \mu_j^{(t)}\right)$$

Premultiplying by $P_{\mu_j}^{(t)}$ yields,

$$\begin{aligned} P_{\mu_j}^{(t)} \left.\frac{\partial l}{\partial \mu_j}\right|_{\mu_j=\mu_j^{(t)}} &= \frac{1}{\sum_{n=1}^{N} \tau_n^{j(t)}} \sum_{n=1}^{N} \tau_n^{j(t)} \left(x_n - \mu_j^{(t)}\right) \\ &= \mu_j^{(t+1)} - \mu_j^{(t)} \end{aligned}$$

From Eq. (3.41), $\sum_{n=1}^{N} \tau_n^{j(t)} > 0$; moreover $\Sigma_j^{(t)}$ is positive definite assuming $N$ is large enough such that the matrix is of full rank. Thus it follows from Eq. 3.67 that $P_{\mu_j}^{(t)}$ is positive definite with probability one.

Finally we look at the update for $\Sigma$. As before from Eq. 3.40 we have

$$\left.\frac{\partial l}{\partial \Sigma_j}\right|_{\Sigma_j=\Sigma_j^{(t)}} = -\frac{1}{2} \sum_{n=1}^{N} \tau_n^{j(t)} \left(\Sigma_j^{(t)}\right)^{-1} \left\{\Sigma_j^{(t)} - \left(x_n - \mu_j^{(t)}\right)\left(x_n - \mu_j^{(t)}\right)^T\right\} \left(\Sigma_j^{(t)}\right)^{-1}$$

With this in mind, we rewrite the update formula for $\Sigma$ in Eq. 3.41 as

$$\begin{aligned}\Sigma_j^{(t+1)} &= \Sigma_j^{(t)} + \frac{1}{\sum\limits_{n=1}^{N} \tau_n^{j(t)}} \sum_{n=1}^{N} \tau_n^{j(t)} \left(x_n - \mu_j^{(t)}\right)\left(x_n - \mu_j^{(t)}\right)^T - \Sigma_j^{(t)} \\ &= \Sigma_j^{(t)} + \frac{2\Sigma_j^{(t)}}{\sum\limits_{n=1}^{N} \tau_n^{j(t)}} V_{\Sigma_j} \Sigma_j^{(t)}\end{aligned}$$

where

$$\begin{aligned}V_{\Sigma_j} &= -\frac{1}{2}\sum_{n=1}^{N} \tau_n^{j(t)} \left(\Sigma_j^{(t)}\right)^{-1} \left\{\Sigma_j^{(t)} - \left(x_n - \mu_j^{(t)}\right)\left(x_n - \mu_j^{(t)}\right)^T\right\}\left(\Sigma_j^{(t)}\right)^{-1} \\ &= \left.\frac{\partial l}{\partial \Sigma_j}\right|_{\Sigma_j = \Sigma_j^{(t)}}\end{aligned}$$

That is we have,

$$\Sigma_j^{(t+1)} = \Sigma_j^{(t)} + \frac{2\Sigma_j^{(t)}}{\sum\limits_{n=1}^{N} \tau_n^{j(t)}} \left.\frac{\partial l}{\partial \Sigma_j}\right|_{\Sigma_j = \Sigma_j^{(t)}} \Sigma_j^{(t)}$$

Utilizing the identity $\operatorname{vec}[ABC] = (C \otimes A)\operatorname{vec}[B]$ we obtain

$$\operatorname{vec}\left(\Sigma_j^{(t+1)}\right) = \operatorname{vec}\left(\Sigma_j^{(t)}\right) + \frac{2}{\sum\limits_{n=1}^{N} \tau_n^{j(t)}} \left(\Sigma_j^{(t)} \otimes \Sigma_j^{(t)}\right) \left.\frac{\partial l}{\partial \Sigma_j}\right|_{\Sigma_j = \Sigma_j^{(t)}} \Sigma_j^{(t)}$$

Thus $P_{\Sigma_j}^{(t)} = \frac{2}{\sum\limits_{n=1}^{N} \tau_n^{j(t)}} \left(\Sigma_j^{(t)} \otimes \Sigma_j^{(t)}\right)$, moreover for an arbitrary matrix $U$ we have

$$\begin{aligned}\operatorname{vec}[U]^T \left(\Sigma_j^{(t)} \otimes \Sigma_j^{(t)}\right)\operatorname{vec}[U] &= \operatorname{trace}\left(\Sigma_j^{(t)} U \Sigma_j^{(t)} U^T\right) \\ &= \operatorname{trace}\left(\left(\Sigma_j^{(t)} U\right)^T \left(\Sigma_j^{(t)} U\right)\right) \\ &\operatorname{vec}\left[\Sigma_j^{(t)} U\right]^T \operatorname{vec}\left[\Sigma_j^{(t)} U\right] \\ &\geq 0\end{aligned}$$

where equality holds only when $\Sigma_j^{(t)} U = 0$. Equality is impossible since $\Sigma_j^{(t)}$ is positive definite with probability one for $N$ sufficiently large. Thus it follows from Eq. 3.68 and $\sum_{n=1}^{N} \tau_n^{j(t)} > 0$ that $P_{\Sigma_j}^{(t)}$ is postive definite with probability one. □

Notice that we can write the updates in Eq. (3.63), (3.64) and (3.65) in the combined form as the following

$$\Theta^{(t+1)} = \Theta^{(t)} + P(\Theta^{(t)}) \frac{\partial l}{\partial \Theta}|_{\Theta=\Theta^{(t)}} \tag{3.69}$$

where $\Theta = \left[\mu_1^T, \ldots, \mu_K^T, \ldots, \text{vec}(\Sigma_1)^T, \ldots, \text{vec}(\Sigma_K)^T, \ldots, \pi^T\right]^T$, the combined parameters in the model. "vec($B$)" stands for the vector obtained by stacking the column vectors of matrix $B$ and $P(\Theta) = \text{diag}\left[P_{\mu_1}, \ldots, P_{\mu_K}, P_{\Sigma_1}, \ldots, P_{\Sigma_k}, P_\pi\right]$ is the combined projection matrix for $\Theta$. That is, the EM algorithm can be viewed as a variable metric gradient ascent algorithm for which the projection matrix $P(\Theta^{(k)})$ changes at each iteration as a function of the current parameter value $\Theta^{(k)}$.
Now notice that we can write Eq. (3.69) as

$$\Theta^{(t+1)} = \mathcal{M}(\Theta^{(t)}) \tag{3.70}$$

where the true EM map is

$$\mathcal{M}(\theta) = \theta + P(\theta)\nabla l(\theta)$$

. Ma et al. (2000) showed linear convergence for EM in GMM locally around the true solution $\theta^\star$ but in **Assumption** 3.1 we require a global property of the map $\mathcal{M}$. Further, for any subsample $s$ of size $m$ drawn from data $\mathcal{D}$ of size N, we can write subsampled log-likelihood following Eq. (3.40) as

$$l_s(\theta) = \frac{N}{m} \sum_{n=1}^{m} \log \sum_{i=1}^{K} \pi_i f\left(x_n^s | \mu_i, \Sigma_i\right) \tag{3.71}$$

where $\{x_1^s, \ldots, x_m^s\} \in s$. Now let $\theta$ be the parameter value at $(j-1)$th iteration in machine $i$ and $s$ denote the subsample in $i$th machine at $j$th iteration. Then we can write the approximate map $M_{i,j}$ as

$$\mathcal{M}_{i,j}(\theta) = \theta + P(\theta)\nabla l_s(\theta)$$

which indicates that **Assumption** 3.2 holds here as $l_s(\theta)$ is a simple random sample estimate of $l(\theta)$.

## 3.7 Numerical Results

We start by examining the relative performance of **Algorithm** 3.1 and **Algorithm** 3.2. We clarify here that in practice we use a little modified version of **Algorithm** 3.1 and **Algorithm** 3.2 due to limitation in resources. In practice, we may not have a large number of machines available (number of machines equal to the number of iterations) and hence we may have to reuse subsampled datasets after a certain number of iterations has passed in **Algorithm** 3.1. For **Algorithm** 3.2 we have to use lesser number of machines relative to the number of iterations but here we don't have to reuse subsamples since there is no communication and each machine performs the EM algorithm with the subsample provided for all the iterations. We provide in Table 3.2 a comparative analysis (root-mean-square error (RMSE) and bias in parenthesis) of implementing the parallel algorithm with communication in two different ways where in method 1 every iteration uses a new subsampled dataset but in method 2 a limited pool (equivalently limited number of machines) of subsampled datasets are there and we have to reuse the same subsample after a certain number of iterations. Here we use a GMM with number of components $K = 2$, dimension $p = 2$ and number of observations $n = 1000$. We draw a random subsample of size 250 in each machine. The number of machines we have used for the two methods are given in the parenthesis in second column of Table 3.2. One can observe that when method 2 uses small number of machines compared to method 1 the performance gap between them is much larger and with increment in number of machines for method 2 the gap in performance reduces.

For comparing the parallel schemes with communication and without communication we use a GMM with number of components $K = 3$, dimension $p = 10$ and number of observations $n = 50000$. We vary the number of machines as 4,8, 16 and 32. The subsample size in each machine is 2000. Different percentages of average overlap of the mixture components are considered viz. 20%, 30% and 40% respectively. All the simulation results are based on 50 replications of both the algorithms. The data are generated as follows. We use the *MixSim* package available in R (Melnykov et al. (2012b)) which allows simulating mixtures of Gaussian distributions with different levels of overlap between mixture components. The key quantity in such data generation mechanism is the pairwise overlap which is the sum of misclassification probabilities between any two clusters. It measures the degree of interaction between the clusters and can be used to control the clustering complexity of the data simulated from the mixtures. The mixture model parameters in their package are generated in the following way.

Mean vectors of the K-component GMM are obtained as K independent realizations from a uniform p-variate hypercube with bounds specified by the user. Covariance matrices $\Sigma_K$ are drawn from the Wishart distributions with parameter $p$ and $(p+1)$ degrees of freedom. Finally, mixing proportions $\pi_K$ are generated on the $[0, 1]$ interval subject to the restriction $\sum_{i=1}^{K} \pi_K = 1$ with the lower bound pre-specified by the user. To simulate a dataset from a generated mixture, first, cluster sizes are obtained as a draw from a multinomial distribution based on mixing proportions. Then, the corresponding number of realizations are obtained from each multivariate normal component.

Melnykov et al. (2010) used two algorithms to generate dataset from GMM based on controlling the average or maximum pairwise overlap. After initial drawing of the model parameters they use an iterative algorithm to obtain the desired level of average or maximum overlap. For details see Melnykov et al. (2010) and Melnykov et al. (2012a). To be specific we provide in Table 3.1 the pairwise overlaps among the three cluster components that we've used for the simulations. We use the average pairwise overlap to control the complexity of the problem and as one can infer the problem of clustering becomes more challenging with higher percentage of overlap.

Nevertheless, here our aim is to compare the two parallel implementations in terms of the model parameter estimation and we focus on comparing the performance of the two parallel methods based on relative bias and variance respectively. The ideal implementation of **Algorithm** 3.1 allows initial estimates in each machine to use different subsamples over the iterations and thereby bias reduction is possible. In our modified implementation of **Algorithm** 3.1 although we reuse subsamples after a certain iteration, we can still gain in reducing estimation bias compared to **Algorithm** 3.2 and will be evident from the following figures.

Table 3.1: Pairwise overlaps among clusters 1, 2 and 3

| Avg. Overlap | 1 and 2 | 1 and 3 | 2 and 3 |
|---|---|---|---|
| 20% | 0.11 | 0.27 | 0.21 |
| 30% | 0.23 | 0.37 | 0.30 |
| 40% | 0.38 | 0.44 | 0.38 |

Table 3.2: RMSE Comparison of $\mu$ and $\Sigma$ for method 1 and method 2 with 40% average overlap

| Methods | rmse($\mu_1$) | rmse($\mu_2$) | rmse($\Sigma_1$) | rmse($\Sigma_2$) |
|---|---|---|---|---|
| method 1 | 0.1186(0.0173) | 0.0999(0.0198) | 0.1025(0.0098) | 0.1272(0.0187) |
| method 2 (4) | 0.2612(0.1116) | 0.2039(0.0880) | 0.1701(0.0444) | 0.2557(0.0833) |
| method 2 (8) | 0.2292(0.1005) | 0.1700(0.0705) | 0.1674(0.0362) | 0.2474(0.0826) |
| method 2 (16) | 0.1836(0.0447) | 0.1237(0.0314) | 0.1370(0.0135) | 0.1667(0.0305) |

We define

$$\theta^\star = \underset{\theta}{\mathsf{Argmax}}\, l(\theta|\mathcal{D})$$

where $l(\theta|\mathcal{D})$ as defined in Eq. (3.40). The bias and variance for comparing the two parallel methods are computed as following:

$$\text{MSE} = \mathbb{E}\left[\|\bar{\theta}_T - \theta^\star\|^2\right] \ ,$$

$$\text{bias} = \|\mathbb{E}\left(\bar{\theta}_T\right) - \theta^\star\|$$

and

$$\text{variance} = \text{MSE} - (\text{bias})^2$$

where $\bar{\theta}_T = \frac{1}{T}\sum_{i=1}^{T}\theta_T^{(i)}$ is the combined estimate obtained after $T$ iterations by simple averaging the estimates in individual machines $i = 1, 2, \ldots, T$. We provide the comparison of the two methods in terms of relative bias in Figure 3.1-3.3. The figures for the corresponding relative standard deviations are given in Figure 3.4-3.6. It can be seen from the figures that our parallel algorithm with communication significantly reduces the bias compared to the one without communication. Further the gap between the two bias curves for the two methods widens with increase in number of machines. Another noticeable fact is that with increase in the overlap percentage our parallel algorithm with communication reduces the bias to a greater extent in comparison to the method without communication. To this end when the pairwise overlap between the two clusters is maximum for example as can be observed between cluster 1 and 3 from Table 3.1, the gap between the two bias curves for the parameters corresponding to those clusters is widest. We also provide results of the time comparison for the two parallel methods in Table 3.3 and the results there show that cost of communication is not too large and additionally we gain in terms of bias reduction if we do the parallel implementation with communication. We plot the square root of variance i.e. standard deviation of the two parallel methods in Figure 3.4-3.6. We expect the variance for both the methods to go down over the number of machines and Figure 3.4-3.6 suggest the same. However the variance for the parallel communication scheme ( shown as “par1” in figure) still remains a bit smaller than the variance for the parallel non-communication scheme (shown as “par2” in figure).

Table 3.3: Ratio of the computing times of **Algorithm** 3.1 and **Algorithm** 3.2 averaged over 50 replication

| Number of Machines | Ratio of computing times |
|---|---|
| 4 | 1.01 |
| 8 | 1.02 |
| 16 | 1.09 |
| 32 | 1.22 |

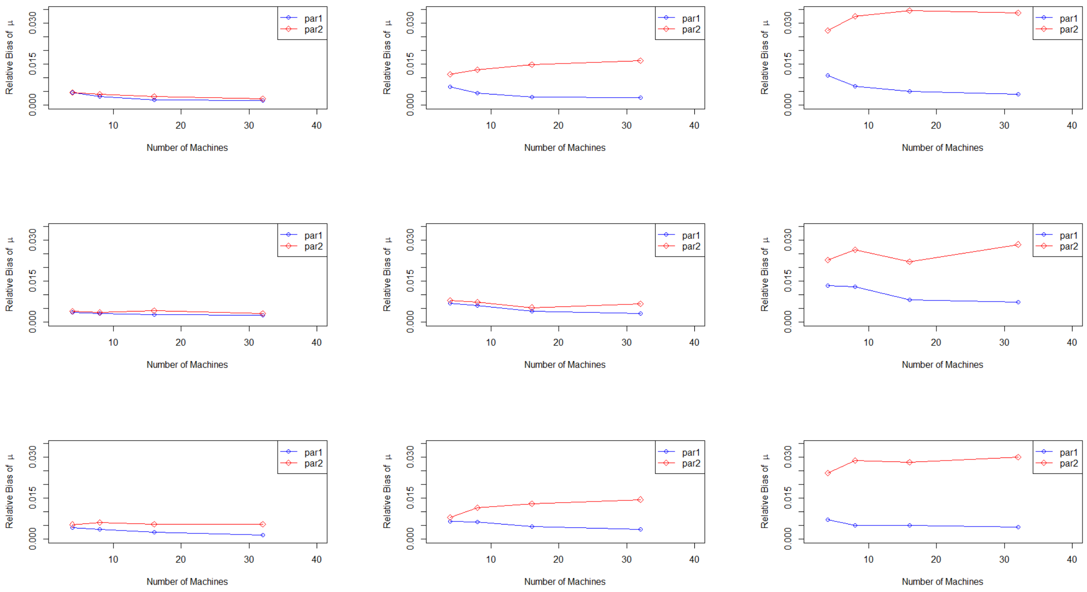


Figure 3.1: Bias comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\mu_1$, $\mu_2$ and $\mu_3$. “par1” is the estimation bias for parallel algorithm with communication and “par2” is the estimation bias for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

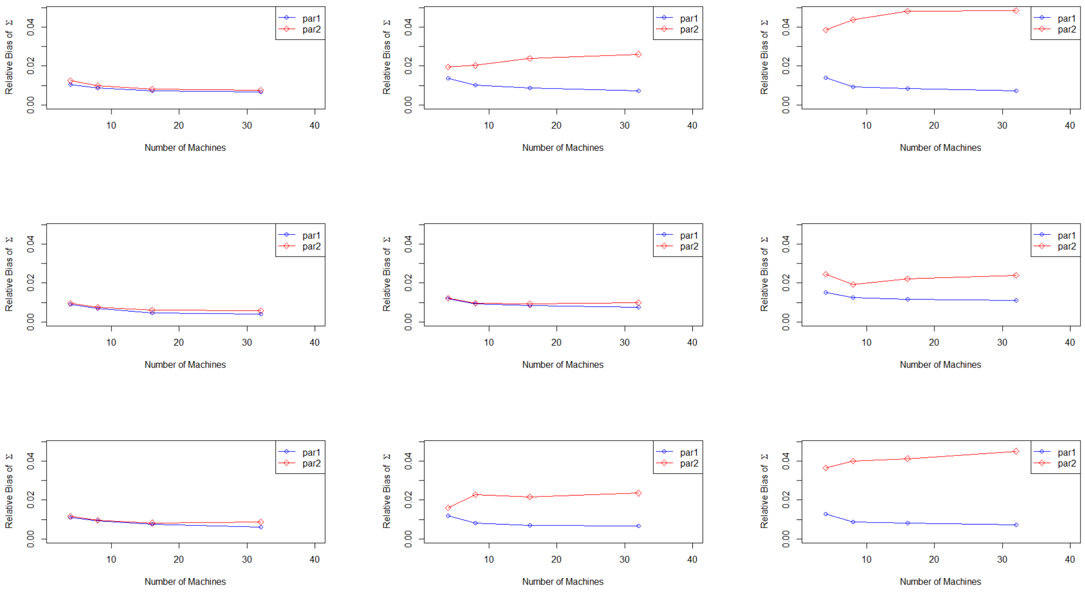


Figure 3.2: Bias comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\Sigma_1$, $\Sigma_2$ and $\Sigma_3$. “par1” is the estimation bias for parallel algorithm with communication and “par2” is the estimation bias for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

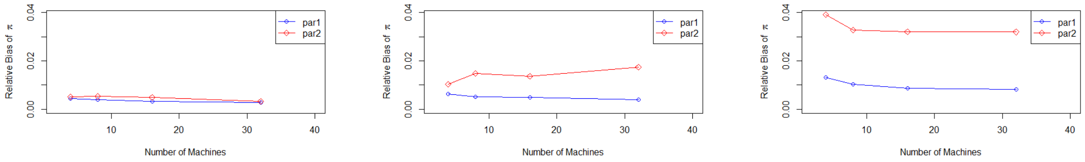


Figure 3.3: Bias comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\pi$. “par1” is the estimation bias for parallel algorithm with communication and “par2” is the estimation bias for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

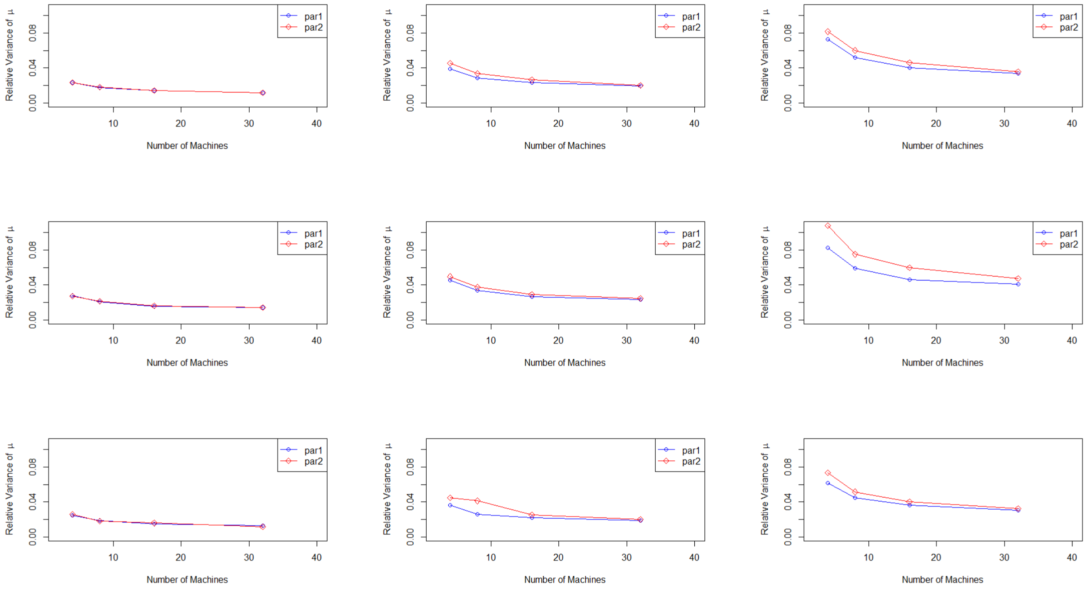


Figure 3.4: Variance comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\mu_1$, $\mu_2$ and $\mu_3$. "par1" is the Variance for parallel algorithm with communication and "par2" is the Variance for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

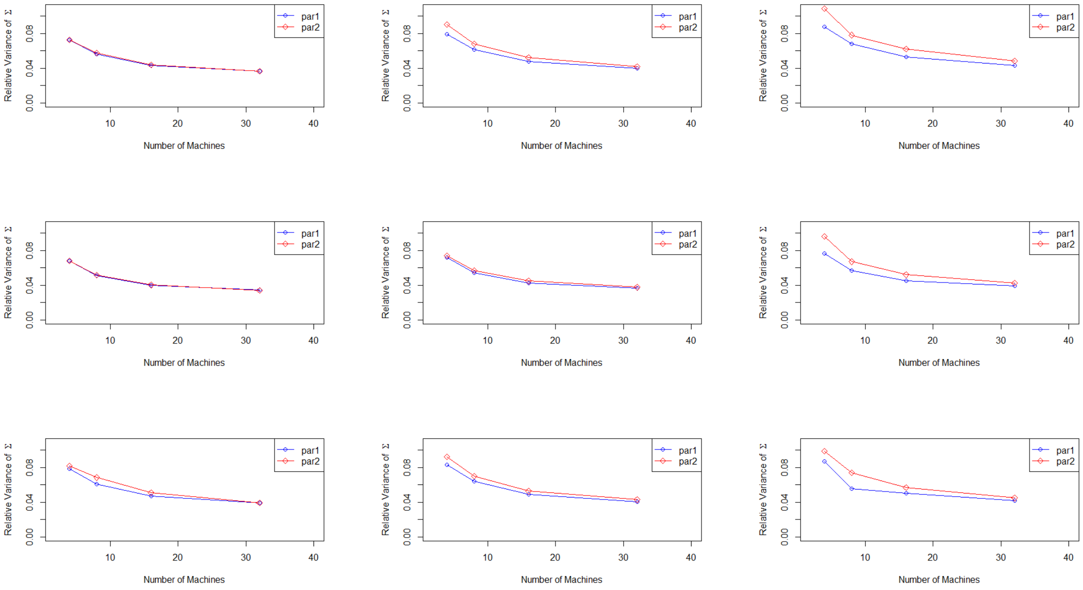


Figure 3.5: Variance comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\Sigma_1$, $\Sigma_2$ and $\Sigma_3$. "par1" is the Variance for parallel algorithm with communication and "par2" is the Variance for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

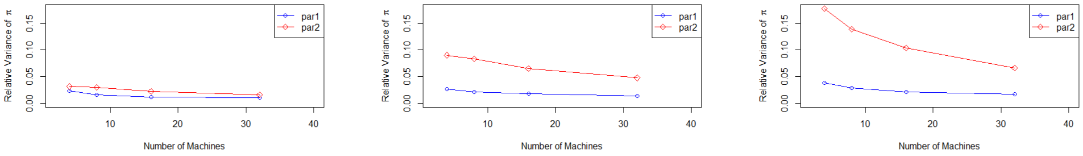


Figure 3.6: Variance comparison of two parallel algorithms over number of machines and varying overlap percentages for estimating $\pi$. "par1" is the Variance for parallel algorithm with communication and "par2" is the Variance for parallel algorithm without communication over different number of machines. The average overlap percentage is varied along the columns in the figure as 20%, 30% and 40% respectively.

# CHAPTER 4

# Likelihood Inference for Large Stochastic Blockmodels with Covariates

## 4.1 Introduction

Stochastic Blockmodels are used to model relational structure among a group of individuals. It has been widely used to model relationships in a social network (Holland et al. (1983), Nowicki and Snijders (2001), Hoff (2008) etc.). Typically parameter estimation in a large Stochastic Block Model (SBM) is a difficult problem. EM can be used for parameter estimation in these kind of latent variable models (mixture models). But EM requires $O(n^2)$ update in every iteration and hence will be computationally infeasible for large $n$ (size of the blockmodel). Amini et al. (2013) provided a pseudo-likelihood method for community detection in large sparse networks. This can be used for fast parameter estimation in a regular SBM but it is not readily applicable to settings when a blockmodel has covariate values. The pseudo-likelihood approximation is not simple to obtain in a covariate blockmodeling framework.

We introduce a model that captures observations coming from a large SBM along with certain number of covariates. We model the log odds of link probability between any two individuals as a composition of the latent block effect and the covariate effect. Our implementation depends on the parallel algorithm (**Algorithm** 3.1) introduced in Chapter 4. The algorithm is based on a case-control approximation of the log-likelihood combined with a subsampling approach. Following **Algorithm** 3.1 we subsample the original adjacency matrix several times and send those small *chunks* of the original matrix to several machines. We use a parallel Monte Carlo EM (MCEM) type algorithm with a communication step among the machines after each iteration. Final estimate is obtained by taking a simple average of the estimates from each machine at the end of the iterations.

The performance of our algorithm is shown on datasets simulated from large SBM with covariates and we also provide a network data example comprising a collection of Facebook profiles of individuals in different US colleges with few specific covariates.

## 4.2 Stochastic BlockModel in Network Data Analysis

Networks are used to model relational data among a set of individuals or a group of entities. With the recent explosion of large datasets, analyzing large network structure involving groups or communities becoming more and more common in practice. Some common examples of large networks with groups or communities are online social networks such as Facebook, Twitter etc. , gene-gene interaction network where group of genes behave in a similar manner due to some external stimulus, co-authorship networks etc.

In analyzing the relational structure among a group of individuals, blockmodels are used to analyze the group structure and position of an individual in that group. They were first introduced in a deterministic sense in the pioneering work by Lorrain and White (1971) in explaining the concept of structural equivalence in social network analysis. Relative to the deterministic model the stochastic one allows us a theoretical model for the relations among the individuals or the actors and assign a edge or link probability for relation between two individuals depending on their memberships. Some initial works by Holland et al. (1983) and Fienberg et al. (1985) led the foundation for analyzing a SBM. Later on the model and its variants have been applied in a variety of disciplines (Airoldi et al. (2008); Hoff (2008); Nowicki and Snijders (2001); Girvan and Newman (2002); Handcock et al. (2007); Copic et al. (2009); Mariadassou et al. (2010); Karrer and Newman (2011)).

But so far there has been less work on introducing covariates into the SBM set up and analyzing the covariate blockmodel as a whole. There are generally two ways of introducing covariates in a network viz. (1) individual/actor level (node specific covariate) and (2) dyadic level (edge specific covariate). Some of the works that has been done towards incorporating covariates in a SBM ( or some variant of SBM) are Tallberg (2004); Mariadassou et al. (2010); Choi et al. (2012); Airoldi et al. (2008) etc. But many of these works incorporate covariates in a individual level in the model whereas we introduce dyadic level covariates that is the probability of presence or absence of an edge is combined effect of the latent part plus the edge specific covariate value.

To this end we can mention the work of Hoff et al. (2002) who introduced dyadic-level covariate in the context of a *Latent Space Model*. Mariadassou et al. (2010) extended general SBM to valued graphs and use a general mixture model describing the intensities of the connections between nodes spread among a certain number of classes. They used a variational approximation for the likelihood and used a variational EM algorithm to estimate the parameters. Choi et al. (2012) worked with the general SBM and used likelihood based inference for the independent Bernoulli data. They considered the log-likelihood as a function of latent variables $z$ and parameter $\theta$ where they treat the latent variables $z$ as fixed parameters. They employed Gibbs sampling to explore the function $\max_\theta L(A; z, \theta)$ and recorded the best value of $z$ visited by the sampler. $L(A; z, \theta)$ is the log-likelihood for the blockmodel. They allow incorporation of covariates in the model such that the log-odds ratio of connection probability between two nodes follows a linear model. In this context they introduce a edge-specific covariate blockmodel. Airoldi et al. (2008) introduced a variant of SBM known as the "Mixed Membership Stochastic Block Model (MMSBM)" which allows inclusion of covariates for the actors belonging to different possible clusters. In all these models the difficulty comes in computing the parameter estimates even without covariate since the likelihood involving the latent membership of the nodes is not in general tractable. The EM algorithm generally is not efficient for large network sizes. The incorporation of the covariates adds another layer of computational hurdle. The objective of this work is to perform likelihood inference for data coming from large stochastic blockmodels with covariates in a efficient way.

We consider a general SBM that incorporates the covariates in a manner similar to that used in Choi et al. (2012) i.e. the logit of the edge probability is a linear model with latent membership part plays the role of the intercept and $X(i, j)$ are the covariate values for each pair of individuals $i$ and $j$. For a K-class SBM the logit of the edge probability is given as following

$$\log \frac{P_{ij}}{1 - P_{ij}} = \tilde{\theta}_{z_i z_j} + \beta^T X(i, j) \; i = 1, \ldots, n; j = i + 1, \ldots, n \tag{4.1}$$

where $P$ is the matrix describing the probability of the edges between any two individuals in the network and the probability of a link between $i$ and $j$ is assumed to be composed of the "latent" part given by $\tilde{\theta}_{z_i z_j}$ and the "covariate" part given by $\beta^T X(i, j)$ where $X(i, j)$ a vector of covariates of the same order indicating shared group membership. We then subsample the entire data available into several machines and perform a MCEM type algorithm to estimate the model parameters in each individual machine with a communication step after every iteration of the optimization algorithm. Finally at the end of all the iterations, we combine the estimates from different machines via simple averaging and the latent membership of the nodes is given by majority voting across machines. The methodology and the algorithm is discussed in detail in Section 4.3. In Section 4.3.1, we describe the general K-class SBM with covariates and Monte-Carlo EM in estimating the parameters of the model; In section 4.3.2, we propose our generic parallel algorithm involving data subsampling using several machines; In section 4.3.3, we discuss the implementation of the parallel algorithm specific to the SBM case. In section 4.5, we show some numerical results of our algorithm. We conclude the chapter with a real data application involving Facebook networks of US colleges with a specific number of covariates.

## 4.3 Data Subsampling for Parameter Estimation in SBM

### 4.3.1 K-class Stochastic Blockmodel with Covariates

We consider independent Bernoulli data $\{A_{ij}\}\,(i = 1, 2, \ldots, n; j = 1, 2, \ldots, n)$, which are the entries of a symmetric adjacency matrix $A = ((a_{ij}))$ of order $n \times n$ defined on a undirected graph with n nodes. We model the Bernoulli success probabilities or the link probabilities $\{P_{ij}\}$ as following

$$P_{ij} = \theta_{z_i z_j} \tag{4.2}$$

for some symmetric matrix $\theta \in [0, 1]^{K \times K}$ and latent membership vector

$$z \in \{1, 2, \ldots, K\}^n$$

. Thus the probability of an edge between any two nodes is assumed to depend only on the class memberships of each of them. The true node labels $z = (z_1, z_2, \ldots, z_n)$ are assumed to be drawn independently from multinomial distribution with parameter $\pi = (\pi_1, \pi_2, \ldots, \pi_K)$ where $\pi_i > 0$ for all $i$. Suppose in addition to the independent Bernoulli data $\{a_{ij}\}$ we have some covariate values observed for each pair of nodes in the undirected graph with n nodes. Instead of model (4.2) we then model the odds of Bernoulli success probability as below

$$\log \frac{P_{ij}}{1-P_{ij}} = \theta_{z_i z_j} + \beta^T X(i,j) \tag{4.3}$$

where $X(i,j)$ are the covariate values observed for each pair of individuals $i$ and $j$ where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. $\beta$ is the corresponding coefficient vector for the covariates. The parameters in the model we want to estimate are $\theta$ , $\beta$ and $\pi$ (class probabilities for the latent membership vector). We model the entries of the adjacency matrix $A = ((a_{ij}))$ as follows

$$A_{ij} \overset{\text{ind}}{\sim} \text{Ber}(P_{ij}) \tag{4.4}$$

The log-likelihood for the observed data is given by

$$\ell(\theta, \beta, \pi) = \log \int_{z \in \mathcal{Z}} L(\theta, \beta | A, z) \prod_{i=1}^{n} \prod_{k=1}^{K} \pi_k^{1(z_i = k)} dz_i \tag{4.5}$$

Although $\mathcal{Z}$ is a discrete set we write it as a integration against the counting measure. When $n$ is large computing the maximum-likelihood estimate (MLE)

$$(\hat{\theta}, \hat{\beta}, \hat{\pi}) = \underset{\theta, \beta, \pi}{\mathsf{Argmax}}\, \ell(\theta, \beta, \pi)$$

is a difficult computational problem. Here $L(\theta, \beta | A, z)$ is the complete data likelihood given by

$$L(\theta, \beta | A, z) = \prod_{i,j} \left( \frac{\mathrm{e}^{\theta_{z_i z_j} + \beta^T X(i,j)}}{1 + \mathrm{e}^{\theta_{z_i z_j} + \beta^T X(i,j)}} \right)^{a_{ij}} \left( \frac{1}{1 + \mathrm{e}^{\theta_{z_i z_j} + \beta^T X(i,j)}} \right)^{1 - a_{ij}} \tag{4.6}$$

Fitting blockmodel is nontrivial, especially for large networks, since in principle the problem of optimizing over all possible class memberships is NP-hard. The implementation of EM for (4.2) or for (4.3) is also computationally time consuming for large networks. Typically MCMC is needed to update the latent variables z, and because the adjacency matrix $A$ is not sparse in general, each iteration of EM to find $(\hat{\theta}, \hat{\beta}, \hat{\pi})$ is $\mathcal{O}(n^2)$, and can be very slow.

The Q-function for implementing EM for SBM with covariate is given by

$$Q(\theta, \beta, \pi; \theta^0, \beta^0, \pi^0) = \int \log L(\theta, \beta|A, z) p(z|A, \theta_0, \beta_0, \pi_0) dz \tag{4.7}$$

Here $\theta_0$, $\beta_0$, $\pi_0$ are some given values of the parameters. Since the integral on the RHS of (4.7) is hard to compute the usual procedure is to approximate the Q-function by sampling latent variables $\{z_{j,r+1}\}$, $j = 1, 2, \ldots, n_g$( $n_g$ is the number of Gibbs samples ) at the $(r + 1)$th iteration from the conditional distribution of the latent variables given the observed data and the parameters i.e. from $p(z|A, \theta_r, \beta_r, \pi_r)$ ( usual E-step of MCEM). The next step (usual M-step of MCEM) that follows then is the maximization of the approximated Q-function i.e.

$$\hat{\theta}^{(r+1)} = \underset{\theta}{\mathsf{Argmax}} \, \widehat{Q} \left(\theta, \beta, \pi; \theta_r, \beta_r, \pi_r\right) \tag{4.8}$$

where $\widehat{Q} \left(\theta, \beta, \pi; \theta_r, \beta_r, \pi_r\right) = \frac{1}{n_g} \sum\limits_{j=1}^{n_g} \log L \left(\theta, \beta|A, z_j\right)$

### 4.3.2 Approximate Parallel Optimization method by Data Sampling

Many optimization algorithms when applied to a large dataset becomes reasonably slow with increase of the size of the data. The common cause being either the update of the parameters in every iterations involves MCMC sampling which slows down with increase in size of the data ( a standard example is the parameter update in SBM where each involves MCMC sampling with $\mathcal{O}(n^2)$, n is the number of observations) or the method in general is sensitive to the size of the datamatrix (a typical example would be EM in the context of Mixture model with large number of observations and high dimension of the parameters within the mixture components, or even with large number of mixing components). We apply **Algorithm** 3.1 discussed in Chapter 3 to perform the parameter estimation in such instances when we are faced with large datasets.

We explore the use of parallel computing for faster computation where only a subset of the data is sent to any processors. We perform an optimization routine on each of the subsampled data in individual machines with a single communication step after every iteration of the optimization algorithm. At the end of the iterations we combine the estimates by simple averaging of the estimates from different machines. For the sake of completeness we once more describe our generic algorithm exploiting parallel computation.

**Algorithm 4.1** Parallel Optimization via Random Subsampling with communication(PORSWC)

```
Input: Data $D$ of size $N$, Number of machines $T$, subsamples $S = \{S_1, S_2, \ldots, S_T\}$
    each of size $m$, initial estimates $\left\{\theta_0^{(i)}\right\}_{i=1}^{T}$ in $T$ machines, Number of iterations $T$
Output: $\bar{\theta}_T = \frac{1}{T}\sum_{i=1}^{T}\theta_T^{(i)}$
procedure PORSWC$\left(D, N, T, S, m, \left\{\theta_0^{(i)}\right\}_{i=1}^{T}\right)$
loop:
    for $j = 1$ to $T$ do
loop:
        parfor $i = 1$ to $T$ do
            $\theta_j^{(i)} = \mathcal{M}_{i,j}\left(\theta_{j-1}^{(i)}\right)$
        end                                    ▷ communication among machines
loop:
        for $i = 1$ to $(T-1)$ do
            $s_{i+1,j} \leftarrow s_{i,j}$
        end
        $s_{1,j} \leftarrow s_{T,j}$
    end
```

We use the above algorithm specific to our problem involving parameter estimation in SBM with covariate. In the next section we describe the specific implementation of the Monte-Carlo EM in parallel for parameter estimation in SBM. The latent node labels that define the communities are estimated by majority voting across several machines.

### 4.3.3 Approximate Parallel Monte Carlo EM

We first describe the approximate MCEM that we perform in individual machines.

#### 4.3.3.1 Approximate Monte Carlo EM

We use spectral clustering with perturbation(Amini et al. (2013)) to obtain an initial cluster labels for the actors in the network. Then based on the initial labeling we sample only a few individuals from each of the cluster. We then approximate the log-likelihood $\log L(\theta, \beta|A, z)$ appearing in Eq. (4.7) via initial random sampling followed by a case-control approximation (Raftery et al. (2012)). The basic idea of case-control approximation is as follows. Typically an adjacency matrix corresponding to a large network will tend to have more "0"s than the number of "1"s. We will treat the "1"s as *cases* and for each individual contribution of the "1"s in the likelihood would be unaltered where as "0"s will be treated as *controls* and we will select( via simple random sampling without replacement) only a few of them for any particular individual. We write the approximate complete data log-likelihood as

$$\log \tilde{L}(\theta, \beta|A, z) \approx \sum_{k=1}^{K} \frac{N_k}{M_k} \sum_{i:i\in\mathcal{M}_k(s)} \tilde{l}_i^{(k)} \tag{4.9}$$

where

$$\tilde{l}_i^{(k)} = \sum_{j:A_{ij}=1} \left(\theta_{kz_j} + \beta^T X(i,j)\right) - \log\left(1 + \mathrm{e}^{\theta_{kz_j}+\beta^T X(i,j)}\right)$$
$$+ \frac{n_{i0}^{(k)}}{m_{i0}^{(k)}} \sum_{j=1}^{m_{i0}^{(k)}} - \log\left(1 + \mathrm{e}^{\theta_{kz_j}+\beta^T X(i,j)}\right)$$

$$\mathcal{M}_k(s) \subset C_k \text{ and } C_k = \left\{i : z_i^0 = k\right\} \text{ , k=1,2,...,K}$$

Here $z_i^0$ is the cluster label of the $i$th individual at the initial stage via spectral clustering with perturbation. $N_k = \sum_i 1(z_i^0 = k) = |C_k|$ for $k = 1, 2, \ldots, K$ and $M_k$ are the number of individuals sampled from the $k$th class i.e. $|\mathcal{M}_k(s)| = M_k$ and denote $\mathcal{M}(s) = \cup_{k=1}^K \mathcal{M}_k(s)$ as the total number of individuals selected combining all the clusters. Here $n_{i0}^{(k)}$ is the number of "zero connections" for the $i$th individual selected from the $k$th class and $m_{i0}^{(k)}$ is the case-control sample size for the $i$th individual selected from the $k$th class, $i = 1, 2, \ldots, M_k$. For simplicity, we keep $m_{i0}^{(k)} = m_0$ for each $i$ and $k$ and choose $m_0 = \lambda r$ where $\lambda$ is the average node degree of the network and $r$ is the global case-to-control rate.

Now using the approximation Eq. (4.9) in Eq. (4.7) we can write the E-step of the MCEM as the following

$$\widehat{\widehat{Q}}\left(\theta, \beta, \pi; \theta_0, \beta_0, \pi_0\right) = \frac{1}{n_g} \sum_{j=1}^{n_g} \log \tilde{L}\left(\theta, \beta|A, z_j\right) \tag{4.10}$$

The latent variables $z_j$ are drawn from the distribution $p(z|A, \theta_0, \beta_0, \pi_0)$ using Gibbs sampling and $n_g$ is the number of Gibbs samples. The approximate Monte-Carlo EM that we perform here differs from the usual E-step in the sense that we only update those latent variables which are selected in the random set $\mathcal{M}(s)$ i.e. $\{z_{j,r+1}\}_{j\in\mathcal{M}(s)}$ are being updated. The M-step then follows as

$$\left(\hat{\theta}_{r+1}, \beta_{r+1}, \pi_{r+1}\right) = \underset{\theta}{\mathsf{Argmax}}\, \widehat{\widehat{Q}}\left(\theta, \beta, \pi; \theta_r, \beta_r, \pi_r\right) \tag{4.11}$$

We repeat the two steps for $r = 1, 2, \dots,$ till convergence.

#### 4.3.3.2 Parallel Implementation of the Approximate Monte-Carlo EM

We now describe the implementation of the approximate Monte-Carlo EM described in section 4.3.3.1 using parallel computation with several machines. The key idea is that for a large network instead of working with the entire adjacency matrix in a single core we draw several subsamples from the adjacency matrix based on some initial labeling of the nodes. Typically as discussed in section 4.3.3.1 we use spectral clustering with perturbation as a initialization method. Then we perform our approximate Monte-Carlo EM in each machine on the subsampled data with a single round of communication among the machines after every iteration. **Algorithm** 4.2 summarizes the details of the proposed approximate parallel MCEM based on random subsampling.

Before describing the specific algorithm we present some notations needed for explaining the algorithm. Let $T$ denote the total number of machines as well as the total number of iterations. $S_1, \dots, S_T$ denote the $T$ subsamples. $z_0$, $\theta_0$, $\beta_0$ and $\pi_0$ denote the initial node label, initial link probability matrix, initial covariate parameter and initial class probability values respectively. Initial node labels $z_0$ are calculate via spectral clustering with perturbation (See Amini et al. (2013)). Let $z_t^{(s)}$ denote the node label of the individuals in the $s$th machine at the $t$th iteration, where $s = 1, 2, \dots, T$. Similarly $\pi_t^{(s)}$, $\theta_t^{(s)}$ and $\beta_t^{(s)}$ denote the parameter estimates at the $t$th iteration in the $s$th machine. As we have pointed it out before in Chapter 3 that transferring the subsamples among machines is a expensive communication step, we do the communication in a equivalent manner by transferring the estimates among the machines in every iteration.

**Algorithm 4.2** Approximate Parallel MCEM

```
procedure Parallel Implementation
    Compute $z_0$, $\theta_0$, $\beta_0$ and $\pi_0$
    Use initial node label $z_0$ to draw $T$ random subsamples $S_1, \dots, S_T$ and send them to $T$ machines available
loop:
    for $t = 1$ to $T$ do
loop:
        parfor $s = 1$ to $T$ do
            $\widehat{\widehat{Q}}_t^{(s)}\left(\theta, \beta, \pi; \theta_{t-1}^{(s)}, \beta_{t-1}^{(s)}, \pi_{t-1}^{(s)}\right) = \frac{1}{n_g}\sum_{j=1}^{n_g} \log \tilde{L}\left(\theta, \beta | A, z_{j,t-1}^{(s)}\right)$
            $\hat{\theta}_t^{(s)} = \underset{\theta}{\mathsf{Argmax}}\, \widehat{\widehat{Q}}\left(\theta, \beta, \pi; \theta_{t-1}^{(s)}, \beta_{t-1}^{(s)}, \pi_{t-1}^{(s)}\right)$
        end
loop:
        for $s = 1$ to $(T-1)$ do
            $\left(\theta_t^{(s+1)}, \beta_t^{(s+1)}, \pi_t^{(s+1)}\right) \leftarrow \left(\theta_t^{(s)}, \beta_t^{(s)}, \pi_t^{(s)}\right)$
        end
        $\left(\theta_t^{(1)}, \beta_t^{(1)}, \pi_t^{(1)}\right) \leftarrow \left(\theta_t^{(T)}, \beta_t^{(T)}, \pi_t^{(T)}\right)$
    end
    Compute $\bar{\theta}_T = \frac{1}{T}\sum_{s=1}^{T} \theta_T^{(s)}$, $\bar{\beta}_T = \frac{1}{T}\sum_{s=1}^{T} \beta_T^{(s)}$ and $\bar{\pi}_T = \frac{1}{T}\sum_{s=1}^{T} \pi_T^{(s)}$
```

## 4.4 Discussion about the true EM map $\mathcal{M}$ and the approximate random map $\mathcal{M}_{i,j}$

We first present the true EM map $\mathcal{M}$ in case of SBM without covariates.The sequence of EM iterates $\{\theta_n\}$ can be written as the following map

$$\begin{aligned}\theta_{n+1} &= \mathcal{M}\left(\theta_n\right)\\ &= \underset{u}{\mathsf{Argmax}}\, Q\left(u|\theta_n\right)\\ &= \underset{u}{\mathsf{Argmax}} \int H_u\left(z\right)\pi_{\theta_n}\left(dz\right) \qquad (4.12)\end{aligned}$$

where $Z = (z_1, \ldots, z_p)$ are the latent variables with $z_i \in \{1, \ldots, K\}$, $\pi_{\theta_n}$ is the conditional probability measure of the latent variables given the observed data at the nth iteration and

$$H_\theta\left(z\right) = \sum_{i<j}\left[A_{ij}\theta_{z_i z_j} - \log\left(1 + \mathrm{e}^{\theta_{z_i z_j}}\right)\right] \qquad (4.13)$$

After some algebraic calculations we arrive at

$$\left[\mathrm{e}^{\mathcal{M}(\theta_n)}\right]_{rs} = \frac{\sum\limits_{i<j} A_{ij} S_{ij}^{rs}}{\sum\limits_{i<j}\left(1 - A_{ij}\right) S_{ij}^{rs}},\ r = 1, \ldots, K;\ s = 1, \ldots, K$$

$$\Rightarrow \mathcal{M}\left(\theta_n\right) = \log\left(\frac{\sum\limits_{i<j} A_{ij} S_{ij}^{rs}}{\sum\limits_{i<j}\left(1 - A_{ij}\right) S_{ij}^{rs}}\right)\quad n = 1, 2, \ldots \qquad (4.14)$$

where

$$S_{ij}^{rs} = \int 1\left(z_i = r, z_j = s\right)\pi_{\theta_n}\left(dz\right)\ .$$

In case of blockmodel with covariate such a compact representation of the true map $\mathcal{M}$ is hard to find since the parameters $\theta$ and $\beta$ are entangled in the complete data log-likelihood given in Eq. (4.6). Further, the approximate random map $\mathcal{M}_{i,j}$ based on random subsampling in the $i$th machine involves two other layers of approximation -(1) due to case-control approximation of the log-likelihood given in Eq. (4.9) and (2) due to Monte Carlo sampling of the latent node labels in the E-step of the MCEM. This additional approximations make derivation of result such as Theorem 3.3 difficult.

### 4.4.1 Review of Some Convergence Results related to MCEM

We discuss here some of the results available in the literature related to the convergence of MCEM. The first serious effort in establishing convergence properties of MCEM is that of Chan and Ledolter (1995) , who treat the data as fixed, and hold the Monte Carlo sample size m constant across MCEM iterations. They then let m go to infinity, and study the asymptotic properties of the MCEM sequence as a Monte Carlo approximation to the ordinary EM sequence with the same starting value (whose convergence properties are well understood). For the sake of completeness we will discuss Chan and Ledolter (1995) result below.
Chan and Ledolter (1995) showed that, given a suitable starting value, a sequence of parameter values generated by the MCEM algorithm will get arbitrarily close to a maximizer of the observed likelihood with high probability. Their main result is given as Theorem 4.1 below.
We denote $\Theta$ to be the parameter space and $\theta \in \Theta$ be the underlying parameters in the model. Let $M_{EM} : \Theta \to \Theta$ denote the mapping given by the deterministic EM update rule, that is, $M_{EM}(\tilde{\theta}) = \mathsf{Argmax}\, Q(\theta|\tilde{\theta}; y)$.

**Theorem 4.1.** *(Theorem 1 of Chan and Ledolter, 1995). Let $\left\{\theta^{(t)}\right\}$ denote a Monte Carlo EM sequence based on Monte Carlo sample sizes $m_t \equiv m$, and suppose that the MCEM update $\mathcal{M}_m(\tilde{\theta}) := \mathsf{Argmax}\, Q_m(\theta|\tilde{\theta}; y)$ converges in probability to $M_{EM}(\tilde{\theta})$ as $m \to \infty$. Further suppose that this convergence is uniform on compact subsets of $\Theta$. Let $\theta^*$ be an isolated local maximizer of $l(\theta; y)$, a continous function of $\theta$. Then there exists a neighborhood of $\theta^*$ such that for any starting value $\theta^{(0)}$ in that neighborhood and for any $\varepsilon > 0$, there exists $T_0$ such that*

$$\Pr\left\{||\theta^{(t)} - \theta^*|| < \varepsilon \text{ for some } t \leq T_0\right\} \;\to\; 1 \qquad (4.15)$$

*as the Monte Carlo sample size $m \to \infty$.*

The conclusion of Theorem 4.1, while interesting, is unsatisfying in at least one respect: It does not guarantee the convergence of an MCEM sequence in any meaningful sense. Practically, what this theorem tells us is that if you run the algorithm long enough (at least $T_0$ iterations), the resulting sequence will, with high probability, *at some point* get arbitrarily close to the MLE. A more powerful result would be one that specifies conditions under which the algorithm gets close to the MLE and stays there.

Fort and Moulines (2003) treat the data as fixed, the Monte Carlo sample size as increasing (deterministically) across MCEM iterations, and establish a.s. convergence of the sequence as the iteration count goes to infinity. We consider this the strongest known result on the asymptotic properties of MCEM, as this notion of convergence seems the most consistent with that of ordinary (deterministic) EM. Further under certain assumptions on the fluctuations of the $L_p$ norm of the Monte Carlo approximations of the EM map $M_{EM}$ they showed linear rate of convergence for MCEM.

## 4.5 Numerical Results

Here we investigate the performance of both our parallel MCEM algorithm applied to SBM with covariate. We simulate observations from the SBM Eq. (4.3). We initialize with spectral clustering with perturbations and evaluate the performance on estimating the link probability matrix ('$\theta$'), class probabilities ('$\pi$'), covariate paramter ('$\beta$') and the latent node labels ('$z$').

Throughout this section we fix number of communities $K$=3, network size $n = 1000$. We vary the "out-in-ratio"(OIR) (Decelle et al. (2011)) $\beta$ as $(0.04, 0.08, 0.2)$ which we term as *low* OIR,*medium* OIR and *high* OIR respectively.. Similarly average degree $\lambda$ is varied as $(4, 8, 14)$ which we term as *low* degree, *medium* degree and *high* degree respectively. We also experiment with two different class probabilities for the 3 communities viz. $\pi = (1/3, 1/3, 1/3)$ (balanced community size) and $\pi = (0.5, 0.3, 0.2)$ (unbalanced community size) We choose global case-to-control rate $r = 7$ so that case-control sample size for our MCEM algorithm is $\lambda r$.

The link probability matrix $\theta$ is generated as discussed in the numerical results section in Amini et al. (2013). For our algorithm we choose subsample size $n_s = 50$. We evaluate the performance of our algorithm by tabulating relative mean squared error (MSE) of the parameters and the Normalized Mutual Information (NMI) values (Amini et al. (2013)) for the latent node labels. All the simulations were performed over 30 replications.

Table 4.1 and 4.2 shows performance of the algorithm by varying the OIR over balanced and unbalanced community size. For Table 4.1 and 4.2 average degree is kept at 7. It clearly shows that as one moves from *low* to *high* regime in OIR value, there is a clear decrease in the NMI values for the latent node labels which is intuitive because with increase in the number of connections among groups clustering becomes harder and hence NMI becomes smaller. On the other hand, the estimation errors for the parameter is affected by lesser extent with increase in OIR value. Further NMI is decreased to a larger extent in unbalanced community size relative to balanced community size when OIR is *high*. Table 4.3 and 4.4 shows performance of **Algorithm** 4.2 when average degree $\lambda$ is varied from *low* to *high* regime. The OIR is kept at 0.04 for Table 4.3 and 4.4. We expect clustering problem to be easier for large $\lambda$ and one sees that improvement in NMI values for larger $\lambda$. For smaller $\lambda$, the NMI is decreased to a larger extent in unbalanced community size (Table 4.4) compared to balanced community size (Table 4.3). The estimation errors for the parameters is affected by lesser extent compared to the NMI values with the decrement of $\lambda$ values.

Table 4.1: Estimation Errors and NMI Values for Balanced Community Size with Varying OIR

| p | OIR | estimation error($\pi$) | estimation error($\theta$) | estimation error ($\beta$) | NMI(c) |
|---|---|---|---|---|---|
| 1000 | 0.04 | 0.0340 | 0.0987 | 0.0232 | 1.000 |
|  | 0.08 | 0.0349 | 0.1042 | 0.0320 | 0.9830 |
|  | 0.2 | 0.0406 | 0.1061 | 0.0476 | 0.7596 |

Table 4.2: Estimation Errors and NMI Values for Unbalanced Community Size with Varying OIR

| p | OIR | estimation error($\pi$) | estimation error($\theta$) | estimation error ($\beta$) | NMI(c) |
|---|---|---|---|---|---|
| 1000 | 0.04 | 0.0704 | 0.0762 | 0.0644 | 0.9327 |
|  | 0.08 | 0.0786 | 0.0778 | 0.1032 | 0.8852 |
|  | 0.2 | 0.0803 | 0.1243 | 0.1149 | 0.6068 |

Table 4.3: Estimation Errors and NMI Values for Balanced Community Size with Varying $\lambda$

| p | $\lambda$ | estimation error($\pi$) | estimation error($\theta$) | estimation error ($\beta$) | NMI(c) |
|---|---|---|---|---|---|
| 1000 | 4 | 0.0508 | 0.0948 | 0.0516 | 0.8240 |
|  | 8 | 0.0451 | 0.0721 | 0.0487 | 0.9670 |
|  | 14 | 0.0340 | 0.0540 | 0.0354 | 0.9868 |

Table 4.4: Estimation Errors and NMI Values for Unbalanced Community Size with Varying $\lambda$

| p | $\lambda$ | estimation error($\pi$) | estimation error($\theta$) | estimation error ($\beta$) | NMI(c) |
|---|---|---|---|---|---|
| 1000 | 4 | 0.0853 | 0.1637 | 0.0706 | 0.7343 |
|  | 8 | 0.0628 | 0.1329 | 0.0612 | 0.8337 |
|  | 14 | 0.0433 | 0.1147 | 0.0478 | 0.9668 |

## 4.6 Application to Collegiate Facebook Data

To illustrate the performance of our algorithm we use a publicly available social network data set (`https://archive.org/details/oxford-2005-facebook-matrix`) containing the social structure of Facebook friendship networks at one hundred American colleges and universities at a single point in time. This data set was analyzed by Traud et al. (2012) . The focus of their study was to illustrate how the relative importance of different characteristics of individuals vary across different institutions. They examine the influence of the common attributes at the dyad level in terms of assortativity coefficients and regression models. We on the other hand pick a data set corresponding to a particular university and show the performance of our algorithm and Pseudo-likelihood based method on it. We also fit a SBM with covariate via our algorithm and compare the clusters obtained from it with the ones obtained in case of fitting SBM without covariate.

We examine the Rice University data set from the list of one hundred American Colleges and Universities and use our K-class SBM with and without covariate to identify group/community structures in the data set. We examine the role of the user attributes- dorm/house number, gender and class year along with the latent structure.

Dorm/house number is a multi-category variable taking values as 202, 203, 204 etc., gender is a binary ($\{0, 1\}$) variable and class year is a integer valued variable (e.g. "2004", "2005", "2006" etc.). We evalauate the performance of **Algorithm** 4.2 fitted to SBM with covariate viz. (4.3).

There are some missing values in the dataset although it is only around 5%. Since the network size is 4087 i.e. which is large enough, we discard the missing value cases. We also consider the covariate values only between year 2004 to 2010. Further, we drop those nodes with degree less than or equal to 1. After this initial cleaning up the adjacency matrix in this case is of order $3160 \times 3160$. We choose number of communities $K = 20$. The choice of the number of the communities is made by employing Bayesian

Information Criterion (BIC) (Schwarz et al. (1978)) where the observed data likelihood is computed by path sampling (Gelman and Meng (1998)). The corresponding figure is given in Figure 4.1 where the possible number of communities are plotted along x-axis and the BIC values are along y-axis.

The K-class SBM with covariate as per (4.3) is the following

$$\log \frac{P_{ij}}{1-P_{ij}} = \tilde{\theta}_{z_i z_j} + \beta^T X(i,j) \; i = 1,\ldots,N; j = i+1,\ldots,n \tag{4.16}$$

where $P$ is the matrix describing the probability of the edges between any two individuals in the network and the probability of a link between $i$ and $j$ is assumed to be composed of the "latent" part given by $\tilde{\theta}_{z_i z_j}$ and the "covariate" part given by $\beta^T X(i,j)$ where $\beta$ is a parameter of size $20 \times 1$ and $X(i,j)$ a vector of covariates of the same order indicating shared group membership. The vector $\beta$ is implemented here with sum to zero identifiability constraints. We first do a basic plot (see Figure4.2) of the degree distribution of the network which clearly shows that the network has a skewed degree distribution. We apply **Algorithm** 4.2 to fit model 4.16 to the Rice university facebook network with three covariates dorm/house number, gender and class year. In the following figure we present the heatmap of the edge probabilities in estimated $\theta$ (latent part in Eq. (4.16)) matrix and a bar diagram showing the estimated class probabilities. We observe from Figure 4.3 that the block 9 has the largest proportion of individuals but do not have a strong tie among the individuals present there as the corresponding entry in the diagonal of the $\theta$ matrix is very small. We also plot the communities found by fitting a SBM without covariate and a blockmodel with covariate (model (4.2) to the given data. We arrange the adjacency matrix rows according to the clusters/communities found by the two methods.

Further we use a information based criterion generally used to compare two different sets of clustering when the ground truth is not known. We use a metric called variation of information (VI) to compare the two sets of clusters. ( For any two sets $C$ and $C'$ of clusters, VI is given as $VI = H(C) + H(C') - I(C, C')$ where $H(.)$ is the entropy function and $I(.,.)$ is the mutual information between the two sets of clusters.). We now present a table which describes how similar the two cluster labels are viz. the one obtained via fitting without covariate blockmodel and the other obtained via fitting the covariate blockmodel. We also indicate in the last column of the table the effect of the possible covariates if the similarity percentage drops below 70%. The

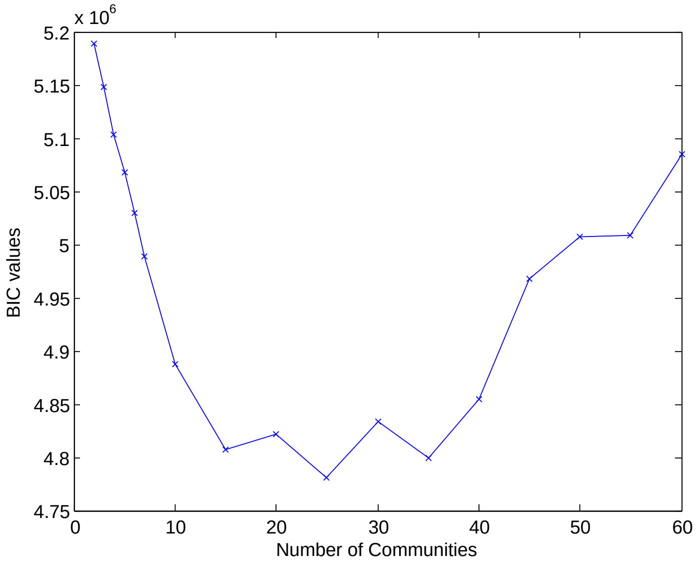


Figure 4.1: Plot of BIC values for possible number of communities in the Rice University dataset

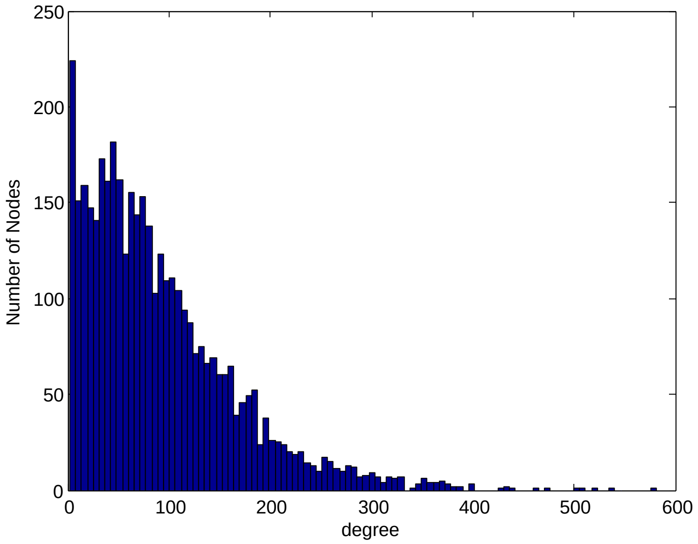


Figure 4.2: Plot of the degree distribution of the Rice University network

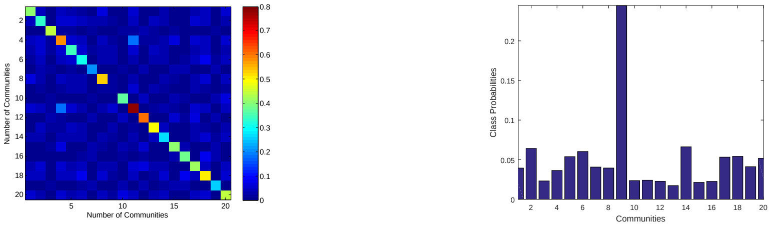


Figure 4.3: Heatmap plots for the edge probability matrix and the bar plot of the class probabilities for parallel MCEM applied to SBM with covariate

estimate of the parameter beta linked with the covariate effects is given by

$$\hat{\beta} = [0.7956, -0.1738, -0.6218]'$$

We compute the metric VI and it is calculated to be 0.1245 which tells us that the two sets of clustering do not differ much. Further Figure 4.5 also indicates that two sets of clustering obtained from without covariate and with covariate model differs only in few specific instances.

## 4.7 Discussion

We present here a covariate blockmodeling framework in the class of blockmodels that has been widely used in analyzing social networks. The edge probability among individuals in the network is modeled by combining the block effect with the covariate effect. To do likelihood inference in large stochastic blockmodels we devise a novel algorithm based on case-control approximation of the log-likelihood along with a subsampling approach. The numerical examples and the real data application validate the use of our parallel algorithm in the context of analyzing large networks by blockmodels with covariates.

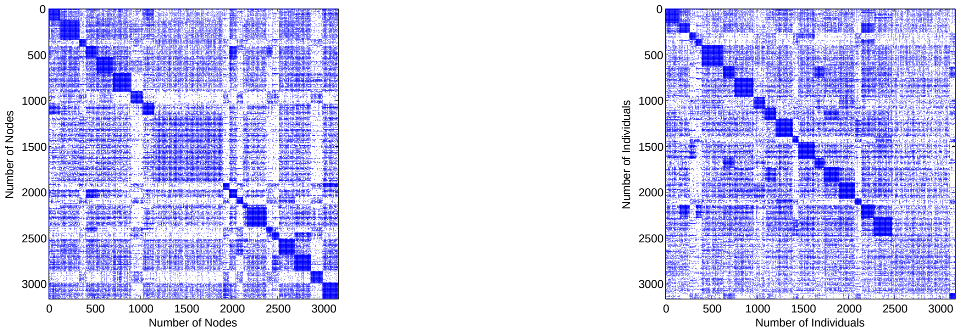
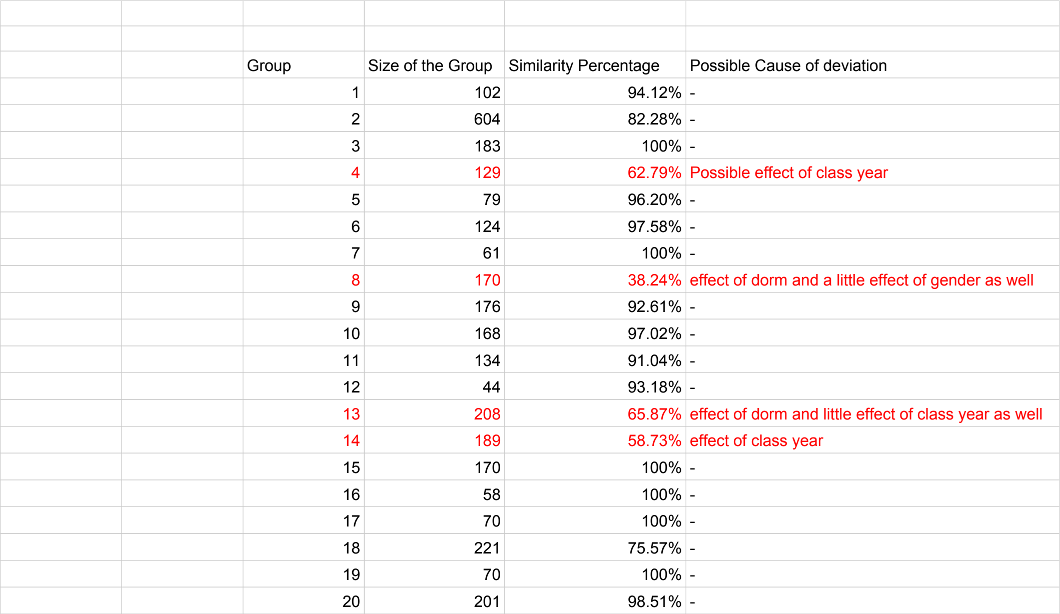


Figure 4.4: Community detection plots for parallel MCEM with and without covariate respectively.



| Group | Size of the Group | Similarity Percentage | Possible Cause of deviation |
|---|---|---|---|
| 1 | 102 | 94.12% | - |
| 2 | 604 | 82.28% | - |
| 3 | 183 | 100% | - |
| 4 | 129 | 62.79% | Possible effect of class year |
| 5 | 79 | 96.20% | - |
| 6 | 124 | 97.58% | - |
| 7 | 61 | 100% | - |
| 8 | 170 | 38.24% | effect of dorm and a little effect of gender as well |
| 9 | 176 | 92.61% | - |
| 10 | 168 | 97.02% | - |
| 11 | 134 | 91.04% | - |
| 12 | 44 | 93.18% | - |
| 13 | 208 | 65.87% | effect of dorm and little effect of class year as well |
| 14 | 189 | 58.73% | effect of class year |
| 15 | 170 | 100% | - |
| 16 | 58 | 100% | - |
| 17 | 70 | 100% | - |
| 18 | 221 | 75.57% | - |
| 19 | 70 | 100% | - |
| 20 | 201 | 98.51% | - |

Figure 4.5: Table showing difference in the communities found by without covariate and with covariate SBM

# BIBLIOGRAPHY

# BIBLIOGRAPHY

Agarwal, A. and J. C. Duchi (2011). Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pp. 873–881.

Airoldi, E. M., D. M. Blei, S. E. Fienberg, and E. P. Xing (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research 9*, 1981–2014.

Amini, A. A., A. Chen, P. J. Bickel, and E. Levina (2013, 08). Pseudo-likelihood methods for community detection in large sparse networks. *Ann. Statist. 41*(4), 2097–2122.

Atchade, Y. F. (2014). Estimation of high-dimensional partially-observed discrete markov random fields. *Electron. J. Statist. 8*(2), 2242–2263.

Bach, F. et al. (2010). Self-concordant analysis for logistic regression. *Electronic Journal of Statistics 4*, 384–414.

Bai, J. (1997). Estimation of a change point in multiple regression models. *Review of Economics and Statistics 79*(4), 551–563.

Banerjee, O., L. El Ghaoui, and A. d'Aspremont (2008). Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research 9*, 485–516.

Basu, S. and G. Michailidis (2015). Estimation in high-dimensional vector autoregressive models. *Ann. Statist.(to appear)*.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 192–236.

Bhattacharya, P. K. (1987). Maximum likelihood estimation of a change-point in the distribution of independent random variables: general multiparameter case. *Journal of Multivariate Analysis 23*(2), 183–208.

Bickel, P. J. and E. Levina (2008). Regularized estimation of large covariance matrices. *The Annals of Statistics*, 199–227.

Bickel, P. J., Y. Ritov, and A. B. Tsybakov (2009). Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, 1705–1732.

Boyd, S. and L. Vandenberghe (2004). *Convex optimization.* Cambridge university press.

Buja, A., T. Hastie, and R. Tibshirani (1989). Linear smoothers and additive models. *The Annals of Statistics*, 453–510.

Carlstein, E. (1988). Nonparametric change-point estimation. *The Annals of Statistics*, 188–197.

Chan, K. and J. Ledolter (1995). Monte carlo em estimation for time series models involving counts. *Journal of the American Statistical Association 90*(429), 242–252.

Choi, D. S., P. J. Wolfe, and E. M. Airoldi (2012). Stochastic blockmodels with a growing number of classes. *Biometrika*, asr053.

Copic, J., M. O. Jackson, and A. Kirman (2009). Identifying community structures from network data via maximum likelihood methods. *The BE Journal of Theoretical Economics 9*(1).

Decelle, A., F. Krzakala, C. Moore, and L. Zdeborová (2011). Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E 84*(6), 066106.

Dekel, O., R. Gilad-Bachrach, O. Shamir, and L. Xiao (2012). Optimal distributed online prediction using mini-batches. *The Journal of Machine Learning Research 13*(1), 165–202.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, 1–38.

Drton, M. and M. D. Perlman (2004). Model selection for gaussian concentration graphs. *Biometrika 91*(3), 591–602.

Duchi, J. C., A. Agarwal, and M. J. Wainwright (2012). Dual averaging for distributed optimization: convergence analysis and network scaling. *Automatic Control, IEEE Transactions on 57*(3), 592–606.

Duchi, J. C., P. L. Bartlett, and M. J. Wainwright (2012). Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization 22*(2), 674–701.

Fan, J., F. Han, and H. Liu (2014). Challenges of big data analysis. *National science review 1*(2), 293–314.

Fienberg, S. E., M. M. Meyer, and S. S. Wasserman (1985). Statistical analysis of multiple sociometric relations. *Journal of the american Statistical association 80*(389), 51–67.

Fithian, W. and T. Hastie (2014). Local case-control sampling: Efficient subsampling in imbalanced data sets. *Annals of statistics 42*(5), 1693.

Fort, G. and E. Moulines (2003, 08). Convergence of the monte carlo expectation maximization for curved exponential families. *Ann. Statist. 31*(4), 1220–1259.

Friedman, J., T. Hastie, and R. Tibshirani (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software 33*(1), 1.

Gelman, A. and X.-L. Meng (1998). Simulating normalizing constants: From importance sampling to bridge sampling to path sampling. *Statistical science*, 163–185.

Ghahramani, Z. and M. I. Jordan (1994). Function approximation via density estimation using the em approach. In *Advances in Neural Information Processing Systems*, pp. 120–127.

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

Guo, J., E. Levina, G. Michailidis, and J. Zhu (2010). Joint structure estimation for categorical markov networks. *Unpublished manuscript 3*(5.2), 6.

Han, F. and H. Liu (2013). A direct estimation of high dimensional stationary vector autoregressions. *arXiv preprint arXiv:1307.0293*.

Handcock, M. S., A. E. Raftery, and J. M. Tantrum (2007). Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society) 170*(2), 301–354.

Hanneke, S., W. Fu, E. P. Xing, et al. (2010). Discrete temporal models of social networks. *Electronic Journal of Statistics 4*, 585–605.

Hinkley, D. (1972). Time-ordered classification. *Biometrika 59*(3), 509–523.

Hinkley, D. V. (1970). Inference about the change-point in a sequence of random variables. *Biometrika 57*(1), 1–17.

Hoefling, H. (2010). *BMN: The pseudo-likelihood method for pairwise binary markov networks.* R package version 1.02.

Hoff, P. (2008). Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, pp. 657–664.

Hoff, P. D., A. E. Raftery, and M. S. Handcock (2002). Latent space approaches to social network analysis. *Journal of the american Statistical association 97*(460), 1090–1098.

Höfling, H. and R. Tibshirani (2009). Estimation of sparse binary pairwise markov networks using pseudo-likelihoods. *The Journal of Machine Learning Research 10*, 883–906.

Holland, P. W., K. B. Laskey, and S. Leinhardt (1983). Stochastic blockmodels: First steps. *Social networks 5*(2), 109–137.

Hurvich, C. M., J. S. Simonoff, and C.-L. Tsai (1998). Smoothing parameter selection in nonparametric regression using an improved akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 60*(2), 271–293.

Johansson, B., M. Rabi, and M. Johansson (2009). A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization 20*(3), 1157–1170.

Karrer, B. and M. E. Newman (2011). Stochastic blockmodels and community structure in networks. *Physical Review E 83*(1), 016107.

Kleiner, A., A. Talwalkar, P. Sarkar, and M. I. Jordan (2014). A scalable bootstrap for massive data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 76*(4), 795–816.

Kolar, M., L. Song, A. Ahmed, and E. P. Xing (2010). Estimating time-varying networks. *The Annals of Applied Statistics*, 94–123.

Kolar, M. and E. P. Xing (2012). Estimating networks with jumps. *Electron. J. Statist. 6*, 2069–2106.

Kosorok, M. R. (2007). *Introduction to empirical processes and semiparametric inference*. Springer Science & Business Media.

Lam, C. and J. Fan (2009). Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of statistics 37*(6B), 4254.

Lan, Y., M. Banerjee, G. Michailidis, et al. (2009). Change-point estimation under adaptive sampling. *The Annals of Statistics 37*(4), 1752–1791.

Loader, C. R. et al. (1996). Change point estimation using nonparametric regression. *The Annals of Statistics 24*(4), 1667–1678.

Lorrain, F. and H. C. White (1971). Structural equivalence of individuals in social networks. *The Journal of mathematical sociology 1*(1), 49–80.

Ma, J., L. Xu, and M. I. Jordan (2000). Asymptotic convergence rate of the em algorithm for gaussian mixtures. *Neural Computation 12*(12), 2881–2907.

Mariadassou, M., S. Robin, and C. Vacher (2010). Uncovering latent structure in valued graphs: a variational approach. *The Annals of Applied Statistics*, 715–742.

McDonald, R., K. Hall, and G. Mann (2010). Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 456–464. Association for Computational Linguistics.

Mcdonald, R., M. Mohri, N. Silberman, D. Walker, and G. S. Mann (2009). Efficient large-scale distributed training of conditional maximum entropy models. In *Advances in Neural Information Processing Systems*, pp. 1231–1239.

McLachlan, G. and D. Peel (2004). *Finite mixture models.* John Wiley & Sons.

Meinshausen, N. and P. Bühlmann (2006). High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 1436–1462.

Meinshausen, N. and P. Bühlmann (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 72*(4), 417–473.

Melnykov, V., W.-C. Chen, and R. Maitra (2012a). Mixsim: An r package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software 51*(12), 1–25.

Melnykov, V., W.-C. Chen, and R. Maitra (2012b). MixSim: An R package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software 51*(12), 1–25.

Melnykov, V., R. Maitra, et al. (2010). Finite mixture models and model-based clustering. *Statistics Surveys 4*, 80–116.

Moody, J. and P. J. Mucha (2013). Portrait of political party polarization. *Network Science 1*(01), 119–121.

Muller, H.-G. (1992). Change-points in nonparametric regression analysis. *The Annals of Statistics*, 737–761.

Nadaraya, E. (1965). On non-parametric estimates of density functions and regression curves. *Theory of Probability & Its Applications 10*(1), 186–190.

Neath, R. C. et al. (2013). On convergence properties of the monte carlo em algorithm. In *Advances in Modern Statistical Theory and Applications: A Festschrift in Honor of Morris L. Eaton*, pp. 43–62. Institute of Mathematical Statistics.

Nedic, A. and A. Ozdaglar (2009). Distributed subgradient methods for multi-agent optimization. *Automatic Control, IEEE Transactions on 54*(1), 48–61.

Neghaban, S., P. Ravikumar, M. Wainwright, and B. Yu (2012). A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science 27*(4), 538–557.

Nielsen, S. F. (2000). The stochastic em algorithm: estimation and asymptotic results. *Bernoulli*, 457–489.

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

Nowlan, S. J. (1991). Soft competitive adaptation: neural network learning algorithms based on fitting statistical mixtures.

Raftery, A. E., X. Niu, P. D. Hoff, and K. Y. Yeung (2012). Fast inference for the latent space network model using a case-control approximate likelihood. *Journal of Computational and Graphical Statistics 21*(4), 901–919.

Raimondo, M. (1998). Minimax estimation of sharp change points. *Annals of statistics*, 1379–1397.

Ram, S. S., A. Nedić, and V. V. Veeravalli (2010). Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications 147*(3), 516–545.

Ravikumar, P., M. J. Wainwright, J. D. Lafferty, et al. (2010). High-dimensional ising model selection using l1-regularized logistic regression. *The Annals of Statistics 38*(3), 1287–1319.

Recht, B., C. Re, S. Wright, and F. Niu (2011). Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 693–701.

Redner, R. A. and H. F. Walker (1984). Mixture densities, maximum likelihood and the em algorithm. *SIAM review 26*(2), 195–239.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *The annals of mathematical statistics*, 400–407.

Rothman, A. J., P. J. Bickel, E. Levina, J. Zhu, et al. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics 2*, 494–515.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics 6*(2), 461–464.

Spall, J. C. (1998). Implementation of the simultaneous perturbation algorithm for stochastic optimization. *Aerospace and Electronic Systems, IEEE Transactions on 34*(3), 817–823.

Tallberg, C. (2004). A bayesian approach to modeling stochastic blockstructures with covariates. *Journal of Mathematical Sociology 29*(1), 1–23.

Traud, A. L., P. J. Mucha, and M. A. Porter (2012). Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications 391*(16), 4165–4180.

Tresp, V., S. Ahmad, R. Neuneier, et al. (1994). Training neural networks with deficient data. *Advances in neural information processing systems*, 128–128.

Van De Geer, S. A., P. Bühlmann, et al. (2009). On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics 3*, 1360–1392.

Wainwright, M. J. and M. I. Jordan (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning 1*(1-2), 1–305.

Wei, G. C. and M. A. Tanner (1990). A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association 85*(411), 699–704.

Wu, C. J. (1983). On the convergence properties of the em algorithm. *The Annals of statistics*, 95–103.

Xu, L. (1997). Comparative analysis on convergence rates of the em algorithm and its two modifications for gaussian mixtures. *Neural Processing Letters 6*(3), 69–76.

Xu, L., M. Jordan, and G. Hinton (1994). A modified gating network for the mixtures of experts architecture. In *World Congress on Neural Networks II, San Diego CA, June*, pp. 405–410.

Xu, L. and M. I. Jordan (1993a). Em learning on a generalized finite mixture model for combining multiple classifiers. In *Proc. of WCNN*, pp. 227–230.

Xu, L. and M. I. Jordan (1993b). Unsupervised learning by em algorithm based on finite mixture of gaussians. In *Proc. of WCNN*, pp. 431–434.

Xu, L. and M. I. Jordan (1996). On convergence properties of the em algorithm for gaussian mixtures. *Neural computation 8*(1), 129–151.

Xue, L., H. Zou, T. Cai, et al. (2012). Nonconcave penalized composite conditional likelihood estimation of sparse ising models. *The Annals of Statistics 40*(3), 1403–1429.

Yuan, M. and Y. Lin (2007). Model selection and estimation in the gaussian graphical model. *Biometrika 94*(1), 19–35.

Zhang, Y., J. C. Duchi, and M. J. Wainwright (2013). Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research 14*, 3321–3363.

Zhou, S., J. Lafferty, and L. Wasserman (2010). Time varying undirected graphs. *Machine Learning 80*(2-3), 295–319.

Zinkevich, M., M. Weimer, L. Li, and A. J. Smola (2010). Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 2595–2603.