Fast and Scalable Joint Estimators for Learning Sparse Gaussian Graphical Models from Heterogeneous Data with Additional Knowledge

---

A Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment
of the requirements for the degree

Doctor of Philosophy

by

Beilun Wang

August 2018

APPROVAL SHEET

This Dissertation
is submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Author Signature: Beilun Wang

This Dissertation has been read and approved by the examining committee:

Advisor: Yanjun Qi

Committee Member: Mahmoody Mohammad

Committee Member: Xiaojin (Jerry) Zhu

Committee Member: Farzad Farnoud

Committee Member: Tingting Zhang

Committee Member:

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, School of Engineering and Applied Science

August 2018

Abstract

Understanding and quantifying variable graphs from heterogeneous samples is a fundamental and urgent analysis task thanks to the data explosion in many scientific domains. Such variable graphs can significantly improve network-driven studies like understanding genetic or neural pathways or providing valuable tools for the discovery of therapeutic targets or diagnostic markers. One typical approach is to jointly estimate K different but related conditional dependency graphs through a multi-task formulation of the sparse Gaussian Graphical Model (multi-sGGM). Most current studies of multi-sGGMs, however, involve expensive and difficult non-smooth optimizations, making them difficult to scale up to many dimensions (large p) or with many contexts (large K).

In this dissertation, we aim to fill the gap and have designed a category of novel estimators that can achieve fast and scalable joint structure estimation of multiple sGGMs.

Three crucial tasks exist when learning multi-sGGMs from heterogeneous samples: (1) to enforce graph relatedness through structural norms, (2) to estimate the change of variable dependencies directly, and (3) to incorporate existing knowledge of the variable nodes or about relationships among nodes. Targeting each, our work introduces fast and parallelizable estimators that largely improves the computational efficiency of the state-of-the-art. We have conducted rigorous statistical analysis and verified that surprisingly the proposed estimators achieve the same statistical convergence rates as the state-of-art solutions that are much harder to compute. Empirically, our estimators outperform the speed of the cutting edge significantly while achieving the same or better prediction accuracy. We have implemented all proposed estimators into publicly accessible tools in the R-CRAN repository. This suite of toolboxes can help users effectively translate aggregated data into knowledge that take the form of graphs.

To everyone who has been an influence, including my amazing and supportive wife and parents

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The past decade has seen a revolution in collecting large-scale heterogeneous data from many scientific fields. For instance, genomic technologies have delivered fast and accurate molecular profiling data across many cellular contexts (e.g., cell lines or stages) from national projects like ENCODE [3]. Given such data, understanding and quantifying variable graphs across multiple contexts is a fundamental analysis task. Such variable graphs can significantly simplify network-driven studies about diseases or treatments [4]. The number of contexts those applications need to consider grows extremely fast. For example, the ENCODE [3] project, being generated over ten years with contributions from bio-labs across the world, contains expression data from 147 different human cell types (i.e., the number of tasks $K = 147$) in 2016. Besides, the number of variables (denoted as p) is also quite large, ranging from thousands (e.g., gene) to hundreds of thousands (e.g., SNP [5]). In addition to the samples themselves, additional information is widely available in real-world applications. In fact, incorporating the knowledge is of great scientific interest. A prime example is when estimating the functional brain connectivity networks among brain regions based on fMRI samples, the spatial position of the regions are readily available. Neuroscientists have gathered considerable knowledge regarding the spatial and anatomical evidence underlying brain connectivity (e.g., short edges and certain anatomical regions are more likely to be connected [6]). Another important example is the problem of identifying gene-gene interactions from patients' gene expression profiles across multiple cancer types. Learning the statistical dependencies among genes from such heterogeneous datasets can help to understand how such dependencies vary from normal to abnormal and help to discover contributing markers that influence or cause the diseases. Besides the patient samples, state-of-the-art

bio-databases like HPRD [1] have collected a significant amount of information about direct physical interactions among corresponding proteins, regulatory gene pairs or signaling relationships collected from high-qualify bio-experiments.

For homogeneous data samples from a given condition, one typical approach to estimate such variable graphs in the literature is the sparse Gaussian Graphical Model (sGGM) [7,8]. sGGM assumes data samples are independently and identically drawn from $N_p(\mu, \Sigma)$, a multivariate normal distribution with mean μ and covariance matrix Σ. The graph structure G is encoded by the sparsity pattern of the inverse covariance matrix, also named precision matrix, Ω. $\Omega := \Sigma^{-1}$. In G an edge does not connect j-th node and k-th node (i.e., conditional independent) if and only if $\Omega_{jk} = 0$. sGGM imposes an sparsity penalty (typically ℓ_1 regularization) on the parameter Ω.

For heterogeneous data from many tasks with additional knowledge, there exist three possible formulations of translating aggregated data from multiple contexts into the knowledge of multiple connectivity graphs.

- First, we can formulate this data analysis problem as jointly estimating K conditional dependency graphs $G^{(1)}, G^{(2)}, \ldots, G^{(K)}$ from data samples accumulated from K distinct conditions. Each graph $G^{(i)}$ is decoded by the i-th precision matrix $\Omega^{(i)}$. Rather than estimating sGGM of each condition separately, a multi-task formulation that jointly estimates K different but related sGGMs with a penalty function $\mathcal{R}'(\cdot)$ can lead to a better generalization [9].

$$\widehat{\Omega}^{(1)}, \widehat{\Omega}^{(2)}, \ldots, \widehat{\Omega}^{(K)} = \underset{\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}}{\operatorname{argmin}} \text{Likelihood}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) + \text{Penalty}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) \tag{1.0.1}$$

- The second task is to learn both the shared (denote as Ω_S) and the context-specific (denote as $\Omega_I^{(i)}$) sub-graphs of multiple sGGMs explicitly and simultaneously from heterogeneous data samples. We formulate each precision matrix Ω^i as the summation of Ω_S and $\Omega_I^{(i)}$. Namely,

$$\Omega^{(i)} = \Omega_I^{(i)} + \Omega_S \tag{1.0.2}$$

- Third, many studies focus on estimating sparse changes in the dependency structure of two p-dimensional GGMs ($N_p(\mu_c, \Sigma_c)$ and $N_p(\mu_d, \Sigma_d)$). The goal is to estimate the structural change

Δ (defined by [10]) [1]:

$$\Delta = \Omega_d - \Omega_c \tag{1.0.3}$$

Where the precision matrix $\Omega_c := (\Sigma_c)^{-1}$ and $\Omega_d := (\Sigma_d)^{-1}$.

Thesis Statement In this research, we aim to design a novel category of estimators that can achieve fast and scalable joint structure estimation of multiple sGGMs in large-scale settings including additional knowledge.

With this purpose, we work on the following three tasks at large scale:

Method 1 – FASJEM: Speed-up and scale-up the joint estimation of K conditional dependency graphs Most previous studies [11–18] for joint estimation of multiple sGGMs relied on optimizing ℓ_1 regularized likelihood function plus an extra penalty function $\mathcal{R}'$. This extra regularizer $\mathcal{R}'$, which varies in different estimators, enforces similarity among multiple estimated networks. Since the penalized likelihood framework includes two regularization functions ($\ell_1 + \mathcal{R}'$), these approaches cannot avoid the steps like SVD [11] and matrix multiplication [11,12]. Both steps need $O(Kp^3)$ time complexity for computation. Besides, most studies in this category require all tasks' covariance matrices to locate in the main memory [11–13] (for their optimization). Storing all elements needs $O(Kp^2)$ memory space. As a result, this category of models are difficult to scale up when the dimension p or the number of tasks K are large.

we propose a novel model, namely fast and scalable joint estimator for multiple sGGM (FASJEM), for estimating multiple sGGMs jointly. Briefly speaking, this estimator presents a new way of learning multi-task sGGMs by extending the elementary estimator [19]. We optimize FASJEM through an entry-wise and group-entry-wise manner that can dramatically improve the time complexity to $O(Kp^2)$. The optimization of our estimators is scalable. We reduce the memory cost to $O(K)$ (i.e., requiring to store at most K entries in the main memory). We propose two variations of FASJEM: (1) FASJEM-G uses a group-2 norm to connect multiple sGGMs. (2) FASJEM-I uses a group-infinite norm to connect multiple related sGGMs. Both methods show better performance over their corresponded "Joint graphical lasso" (JGL) baselines. In addition, we theoretically prove the convergence rate of FASJEM as $O(\log(Kp)/n_{tot})$. This rate shows the benefit of joint estimation,

[1] Using which of the two sample sets as 'c' set (or 'd' set) does not affect the computational cost and the statistical convergence rates of our model. For instance, on samples from a controlled disease study, 'c' may represent the 'control' group and 'd' may represent the 'disease' group.

which significantly improves the convergence rate $O(\frac{\log p}{n})$ of single task sGGM (with n samples). FASJEM is evaluated using several synthetic datasets and four real-world biomedical datasets. It performs better than the baselines not only on accuracy but also with respect to the time and storage requirements.

Method 2 – JEEK: Speed-up and scale-up learning sparse Gaussian graphical models (sGGMs) with additional knowledge One significant caveat of state-of-the-art joint sGGM estimators is the fact that little attention has been paid to incorporating existing knowledge of the nodes or knowledge of the relationships among nodes in the models. Although being strong evidence of structural patterns we aim to discover, this type of information has rarely been considered in the joint sGGM formulation of such samples. To the our best knowledge, only one study named as W-SIMULE tried to extend the constrained ℓ_1 minimization in SIMULE into weighted ℓ_1 for considering spatial information of brain regions in the joint discovery of heterogeneous neural connectivity graphs [20]. This method was designed just for the neuroimaging samples and has $O(p^5K^4)$ time cost, making it not scalable for large-scale settings (more details in Section 5.1).

We aims to fill this gap by adding additional knowledge most effectively into scalable and fast joint sGGM estimations. We propose a novel model, namely Joint Elementary Estimator incorporating additional Knowledge (JEEK), that presents a principled and scalable strategy to include additional knowledge when estimating multiple related sGGMs jointly. JEEK presents a new way of integrating additional knowledge in learning multi-task sGGMs in a scalable way. We optimize JEEK through an entry-wise and group-entry-wise manner that can dramatically improve the time complexity to $O(p^2K^4)$. In addition, we theoretically prove the convergence rate of JEEK as $O(\log(Kp)/n_{tot})$. This rate shows the benefit of joint estimation and achieves the same convergence rate as the state-of-the-art that are much harder to compute. Finally, we evaluate JEEK using several synthetic datasets and two real-world data, one from neuroscience and one from genomics. It outperforms state-of-the-art baselines significantly regarding the speed.

Method 3 – DIFFEE: Speed-up and scale-up the estimation of sparse changes in the dependency structure of two p-dimensional GGMs A naive approach to detecting structural changes in GGMs is a two-step procedure in which we estimate $\widehat{\Omega}_d$ and $\widehat{\Omega}_c$ from two sets of samples separately and obtain $\widehat{\Delta} = \widehat{\Omega}_d - \widehat{\Omega}_c$. However, in a high-dimensional setting, this strategy needs to assume that both Ω_d and Ω_c are sparse (in order to achieve consistent estimation). This is not

necessarily true even if the change Δ is sparse. A motivating example s from identifying the difference in connectivity networks among brain regions (functional networks) of subjects from different groups. Recent literature in neuroscience has suggested functional networks are not sparse. On the other hand, differences in functional connections across subjects should be sparse [21]. In the application of estimating genetic networks of two conditions, each individual network might contain hub nodes and therefore not entirely sparse.

We propose a simple estimator, namely DIFFerential networks via an Elementary Estimator (DIFFEE) for fast and scalable learning of sparse structural change in high-dimensional GGMs. DIFFEE presents a novel way of structural change estimation by extending the elementary estimator for sparse GGM [19]. We optimize DIFFEE through a closed-form manner that can dramatically improve its entire time complexity to $O(p^3)$. The closed-form solution makes DIFFEE scalable to much larger values of p, compared to the aforementioned state-of-the-art. We theoretically prove that DIFFEE achieves the same sharp convergence rate as the aforementioned regularized convex programs. DIFFEE is evaluated using several simulated datasets and one real-world neuroscience dataset. It improves the state-of-the-art baselines with better estimation F-1 scores as well as significant computational advantages.

The remainder of the proposal is organized as follows: Chapter 2 reviews the Backgrounds, Chapter 3 describes the proposed approach Joint Elementary Estimators, Chapter 4 presents the Method and results of Method 1 JEEK, Chapter 5 presents the Method and results of Method 2 FASJEM, Chapter 6 presents the Method and results of Method 3 DIFFEE, Chapter 7 provide the Extensions, and finally, Chapter 8 lists the conclusions and potential future works.

Chapter 2

Background

List of Symbols

2.1 Sparse Gaussian Graphical Model

Sparse Gaussian Graphical Model(sGGM) [7, 8, 22] assumes data samples are independently and identically drawn from $N_p(\mu, \Sigma)$, a multivariate normal distribution with mean μ and covariance matrix Σ. The conditional dependency graph structure among its p random variables is encoded by the sparsity pattern of the inverse covariance matrix (precision matrix) Ω. $\Omega := (\Sigma)^{-1}$. An edge does not connect j-th node (variable) and k-th node (variable) if and only if $\Omega_{jk} = 0$ (i.e., conditionally independent). sGGM imposes an ℓ_1 penalty on the parameter Ω.

Regularized MLE: Over the past decade, significant progress has been made on estimating sGGMs based on samples drawn from the model. Most sGGM estimation [8, 23] are based on minimizing the ℓ_1-regularized Gaussian negative log likelihood:

$$\underset{\Omega}{\operatorname{argmin}} - \log\det(\Omega) + < \Omega, \Sigma > + \lambda_n ||\Omega||_1 \tag{2.1.1}$$

Friedman et al. [24] used a blockwise coordinate descent algorithm called the graphical lasso to efficiently solve the regularized MLE formulation. Alternatively, Meinshausen et al. [25] introduced a

Σ	Covariance Matrix
μ	Mean vector in the Gaussian distribution
Ω	Precision matrix
$\mathbf{X}$	Data sample matrix
$\Sigma^{(i)}$	ith Covariance matrix
$\Omega^{(i)}$	ith Precision matrix in a multi-task setting
Ω_S	Shared pattern among all precision matrices in a multi-task setting
$\Omega_I^{(i)}$	Individual part of ith Precision matrix in a multi-task setting
$\mathbf{X}^{(i)}$	ith data sample matrix in a multi-task setting
n_i	Number of samples in ith data matrix
n_{tot}	Total number of samples in a multi-task setting
$\mathbf{x}$	A p-dimensional sample
e_j	$(0, \ldots, 1, \ldots, 0)^T$
Σ^{tot}	$(\Sigma^{(1)}, \Sigma^{(2)}, \ldots, \Sigma^{(K)})$
Ω_I^{tot}	$(\Omega_I^{(1)}, \Omega_I^{(2)}, \ldots, \Omega^{(K)})_I$
Ω_S^{tot}	$(\Omega_S^{(1)}, \Omega_S^{(2)}, \ldots, \Omega^{(K)})_S$
Ω^{tot}	$(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$
Ω_c	The precision matrix for the control case
Ω_d	The precision matrix for the disease case
Σ_c	The covariance matrix for the control case
Σ_d	The covariance matrix for the disease case
λ_n	Tuning parameter
ϵ	Hyper-parameter
W^{tot}	Weight matrices for additional knowledge
W_I^{tot}	$(W_I^{(1)}, W_I^{(2)}, \ldots, W^{(K)})_I$
W_S^{tot}	$(W_S^{(1)}, W_S^{(2)}, \ldots, W^{(K)})_S$
W_S	Shared pattern among all Weight matrices in a multi-task setting
$W_I^{(i)}$	Individual part of ith Weight matrix in a multi-task setting
K	Total number of tasks
p	Total number of features

Table 2.1: List of Important Notations.

neighborhood selection approach that applies a lasso linear regression on each variable separately and combines the result to learn the conditional dependency structure.

CLIME: Later Cai et al. [26] proposed a constrained ℓ_1 minimization method for inverse matrix estimation (abbreviated as CLIME) formulated as follows:

$$\begin{aligned} &\underset{\Omega}{\text{argmin}} ||\Omega||_1 \\ &\text{subject to: } ||\Sigma\Omega - I||_\infty \leq \lambda_n \end{aligned} \tag{2.1.2}$$

The above formulation can be decomposed into column-wise linear programming. However the computational cost of this LP formulation gets significantly demanding as p increases.

EE-sGGM: Recently, Yang et al. [19] proposed a closed-form estimator for learning Gaussian graphical models through the following form:

$$\begin{aligned} &\underset{\Omega}{\operatorname{argmin}} ||\Omega||_{1,,\text{off}} \\ &\text{subject to:} ||\Omega - [T_v(\widehat{\Sigma})]^{-1}||_{\infty,\text{off}} \leq \lambda_n \end{aligned} \tag{2.1.3}$$

Eq. (2.1.3) is a special case of the elementary estimator for graphical models (GM) of exponential families proposed in [19], namely Elementary Estimators-GM. It has the following generic formulation:

$$\begin{aligned} &\underset{\theta}{\operatorname{argmin}} ||\theta||_1 \\ &\text{Subject to: } ||\theta - \mathcal{B}^*(\widehat{\phi})||_\infty \leq \lambda_n \end{aligned} \tag{2.1.4}$$

Here $\mathcal{B}^*(\widehat{\phi})$ is the so-called proxy of backward mapping for the target GM (more details in Section 2.3). λ_n is a regularization parameter. $\widehat{\phi}$ is the empirical mean of the sufficient statistics. For example, in the case of Gaussian GM, $\widehat{\phi}$ is the sample covariance matrix. More details are in Section 2.4.

M-Estimator with Decomposable Regularizer in High-Dimensional Situations: Recently the seminal study [27] proposed a unified framework for the high-dimensional analysis of the following general formulation: M-estimators with decomposable regularizers:

$$\underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\theta) + \lambda_n \mathcal{R}(\theta) \tag{2.1.5}$$

where $\mathcal{R}(\cdot)$ represents a decomposable regularization function and $\mathcal{L}(\cdot)$ represents a loss function (e.g., the negative log-likelihood function in sGGM $\mathcal{L}(\Omega) = -\log\det(\Omega) + < \Omega, \widehat{\Sigma} >$). Here $\lambda_n > 0$ is the tuning parameter.

2.2 Joint Learning sparse Gaussian Graphical Models (JsGGMs)

2.2.1 Penalized MLE based Estimators

Sparse GGM is an extremely active topic in the recent literature including notable studies like [28] and [23]. We can categories single-task sGGM estimators into three groups: (a) penalized likelihood (GLasso), (b) neighborhood approach and (c) CLIME estimator.
Most previous methods that estimate multiple sGGMs jointly (on the same set of variables from aggregated data samples) can be formulated as:

$$\begin{aligned} \underset{\Omega^{(i)}>0}{\operatorname{argmin}} \sum_i (-L(\Omega^{(i)}) + \lambda_n \sum_i ||\Omega^{(i)}||_1 \\ + \lambda_2 \mathcal{R}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) \end{aligned} \tag{2.2.1}$$

where $\Omega^{(i)}$ denotes the precision matrix for the i-th task. $L(\cdot)$ represents log-likelihood or pseudo-likelihood function. $\sum_i ||\Omega^{(i)}||_1$ adds sparsity constraints on each task. $P(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$ enforces certain joint properties among the tasks (like using group sparsity to enforce similarity among tasks).

The general purpose of the penalty function $\mathcal{R}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$ in Eq. (5.6.2) is to push the inference of multiple graphs toward a common pattern. Table 2.2 provides a representative list of penalty functions that have been used previously for the multi-sGGM setting. For example, JGL uses the fused norm (the 1st row of Table 2.2) to penalize the difference between two graphs with resulting variation named as JGL-fused. JGL-group uses a $\{\mathcal{G}, 2\}$ norm that pushes multiple graphs to have the same sparsity patterns (the 2nd row of Table 2.2). SIMONE provides a novel penalty proposed by the authors (shown as in the 3rd row of Table 2.2) to enforce similar sparsity pattern on multiple graphs.

There are a lot of existing multi-task sGGMs studies. For example, (a) Fused Joint graphical lasso (JGL-fused) [11], (b) Group Joint graphical lasso (JGL-group) [11] and (c) SIMONE [13]. JGL-fused and JGL-group are based on the popular "graphical lasso" estimator [8, 24]; (using $L(\Omega) = (\log det(\Omega) - < \Sigma, \Omega >)$ in Eq. (5.6.2)). SIMONE [13] follows neighborhood-selection based estimator. It can be viewed as using a pseudo-likelihood approximation instead of the full likelihood

as $L(\Omega)$ in Eq. (5.6.2). [1] In addition to these three works, a number of recent studies also perform multi-task learning of sGGM [14–18]. They all follow the same formulation as Eq. (5.6.2) but explore a different second penalty-function $P(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$. As an example, Node-based JGL proposed a novel penalty, namely $RCON$ [12] (shown as the 4th row of the Table 2.2) or "row-column overlap norm" for capturing special relationship among graphs. In two recent works, the penalty function at the 5th row is Table 2.2 has been used by [15] and the penalty function at the 6th row of Table 2.2 has been used by [14].

Furthermore, there exist studies that explore similar motivations as ours when learning multiple GGM models from data. (a) [29] proposed to estimate a population graph from multi-block data using a so-called "median-graph" idea. It is conceptually similar to Ω_S. However, they do not have $\Omega_I^{(i)}$ to model individual parts that are specific to each task. (b) Another recent study, CSSL-GGM [30] also tried to model both the shared and individual substructures in multi-sGGMs. Different from ours, their formulation is within the penalized likelihood framework as Eq. (5.6.2). They used ℓ1,p norm (see last row of Table 2.2) to regularize the task-specific parts, while SIMULE uses ℓ1 norm instead in Eq. (5.6.2). The ℓ1,p norm pushes the individual parts of multiple graphs to be similar which is contradictory to the original purpose of these parameters. [2] (c) More recently, [31] proposed to learn population and subject-specific brain connectivity networks via a so-called "Mixed Neighborhood Selection" (MSN) method. Following the neighborhood selection framework [25], for each node v, MSN tried to learn the neighborhood of each v. Similar to SIMULE, they estimated the neighborhood edges of a given node v in the i-task as $\beta^v + \widetilde{\boldsymbol{b}}^{(i),v}$. Here β^v represents the neighbor in the shared part and $\widetilde{\boldsymbol{b}}^{(i),v}$ represents the neighbors that are specific to the i-th graph. Since MSN is specially designed for brain imaging data, it assumes each individual graph is generated by random effects, i.e., $\widetilde{\boldsymbol{b}}^{(i),v} \sim N(0, \Phi^v)$. SIMULE does not have such strong assumptions on either task-specific or task-shared substructures. Our model is more general while MSN is designed for brain imaging data. (d) Another line of related studies [32, 33] prosed density-ratio based strategies to estimate a differential graph between two graphs. Even though this group of methods can handle the unbalance dataset (i.e., the numbers of samples in two datasets are quite different), they can only capture the difference between two graphs ($K = 2$). SIMULE does not have such a limitation on the number of tasks. (e) Moreover, several loosely related studies exist in settings different from ours. For example, for handling high-dimensional time series data a few recent papers have considered exploring multiple sGGMs by modeling relationships among networks; e.g., [34] [35].

[1] JGL [11] and SIMONE [13] are the two most-cited joint GGM estimators in the literature.
[2] We can not find CSSL-GGM implementation, therefore can not include it as a baseline.

Table 2.2: A list of representative multi-sGGM methods and the second penalty functions they have used.

	References	Penalty Function $P(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) =$
(1)	JGL-Fused [11]	$\sum_{ij,i>j} \|\|\Omega^{(i)} - \Omega^{(j)}\|\|_1$
(2)	JGL-Group [11]	$\|\|\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}\|\|_{\mathcal{G},2}$
(3)	SIMONE [13]	$\sum_{i\neq j}((\sum_{k=1}^{T}(\Omega_{ij}^{(k)})_+^2))^{\frac{1}{2}} + ((\sum_{k=1}^{K}(-\Omega_{ij}^{(k)})_+^2))^{\frac{1}{2}}$
(4)	Node JGL [12]	$\sum_{ij,i>j} RCON(\Omega^{(i)} - \Omega^{(j)})$
(5)	JEM-GM [15]	$\sum_{k=1}^{K} w_k \|\|\Omega^{(k)}\|\|_1$
(6)	MTL-GGM [14]	$\|\|\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}\|\|_{\mathcal{G},\infty}$
(7)	CSSL-GGM [30]	$\|\|\Omega_S\|\|_1 + \|\|\Omega_I^{(1)}, \Omega_I^{(2)}, \ldots, \Omega_I^{(K)}\|\|_{1,p}$

2.2.2 Constrained L1 Minimization based Estimators

One recent study [20] of multi-sGGMs (following ideas from [36]) also assumed that $\Omega^{(i)} = \Omega_I^{(i)} + \Omega_S$ and incorporated spatial distance knowledge in their convex formulation for joint discovery of heterogeneous neural connectivity graphs. This study, with name W-SIMULE (Weighted model for Shared and Individual parts of MULtiple graphs Explicitly) uses a weighted constrained ℓ_1 minimization:

$$\underset{\Omega_I^{(i)}, \Omega_S}{\operatorname{argmin}} \sum_i ||W \circ \Omega_I^{(i)}||_1 + \epsilon K ||W \circ \Omega_S||_1 \tag{2.2.2}$$

$$\text{Subject to: } ||\Sigma^{(i)}(\Omega_I^{(i)} + \Omega_S) - I||_\infty \leq \lambda_n,\ i = 1, \ldots, K$$

W-SIMULE simply includes the additional knowledge as a weight matrix W. [3]

[3] It can be solved by any linear programming solver and can be column-wise paralleled. However, it is very slow when $p > 200$ due to the expensive computation cost $O(K^4 p^5)$.

2.3 Backward Mapping

2.3.1 Backward mapping for Exponential Families

The solution of vanilla graphical model MLE can be expressed as a backward mapping [37] for an exponential family distribution. It estimates the model parameters (canonical parameter θ) from certain (sample) moments. We provide detailed explanations about backward mapping of exponential families, backward mapping for Gaussian special case and backward mapping for differential network of GGM in this section.

Backward mapping: Essentially the vanilla graphical model MLE can be expressed as a backward mapping that computes the model parameters corresponding to some given moments in an exponential family distribution. For instance, in the case of learning GGM with vanilla MLE, the backward mapping is $\widehat{\Sigma}^{-1}$ that estimates Ω from the sample covariance (moment) $\widehat{\Sigma}$.

Suppose a random variable $X \in \mathbb{R}^p$ follows the exponential family distribution:

$$\mathbb{P}(X;\theta) = h(X)\exp\{< \theta, \phi(\theta) > -A(\theta)\} \tag{2.3.1}$$

Where $\theta \in \Theta \subset \mathbb{R}^d$ is the canonical parameter to be estimated and Θ denotes the parameter space. $\phi(X)$ denotes the sufficient statistics as a feature mapping function $\phi : \mathbb{R}^p \to \mathbb{R}^d$, and $A(\theta)$ is the log-partition function. We then define mean parameters v as the expectation of $\phi(X)$: $v(\theta) := \mathbb{E}[\phi(X)]$, which can be the first and second moments of the sufficient statistics $\phi(X)$ under the exponential family distribution. The set of all possible moments by the moment polytope:

$$\mathcal{M} = \{v | \exists p \text{ is a distribution s.t. } \mathbb{E}_p[\phi(X)] = v\} \tag{2.3.2}$$

Mostly, the graphical model inference involves the task of computing moments $v(\theta) \in \mathcal{M}$ given the canonical parameters $\theta \in \mathcal{H}$. We denote this computing as **forward mapping** :

$$\mathcal{A} : \mathcal{H} \to \mathcal{M} \tag{2.3.3}$$

The learning/estimation of graphical models involves the task of the reverse computing of the forward

mapping, the so-called **backward mapping** [37]. We denote the interior of $\mathcal{M}$ as $\mathcal{M}^0$. **backward mapping** is defined as:

$$\mathcal{A}^* : \mathcal{M}^0 \to \mathcal{H} \tag{2.3.4}$$

which does not need to be unique. For the exponential family distribution,

$$\mathcal{A}^* : v(\theta) \to \theta = \nabla A^*(v(\theta)). \tag{2.3.5}$$

Where $A^*(v(\theta)) = \sup_{\theta \in \mathcal{H}} < \theta, v(\theta) > -A(\theta)$.

2.3.2 Backward Mapping: Gaussian Case

If a random variable $X \in \mathbb{R}^p$ follows the Gaussian Distribution $N(\mu, \Sigma)$. then $\theta = (\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1})$. The sufficient statistics $\phi(X) = (X, XX^T)$, $h(x) = (2\pi)^{-\frac{k}{2}}$, and the log-partition function

$$A(\theta) = \frac{1}{2}\mu^T\Sigma^{-1}\mu + \frac{1}{2}\log(|\Sigma|) \tag{2.3.6}$$

When performing the inference of Gaussian Graphical Models, it is easy to estimate the mean vector $v(\theta)$, since it equals to $\mathbb{E}[X, XX^T]$.

When learning the GGM, we estimate its canonical parameter θ through vanilla MLE. Because Σ^{-1} is one entry of θ we can use the backward mapping to estimate Σ^{-1}.

$$\begin{aligned} \theta &= (\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}) = \mathcal{A}^*(v) = \nabla A^*(v) \\ &= ((\mathbb{E}_\theta[XX^T] - \mathbb{E}_\theta[X]\mathbb{E}_\theta[X]^T)^{-1}\mathbb{E}_\theta[X], \\ &\quad -\frac{1}{2}(\mathbb{E}_\theta[XX^T] - \mathbb{E}_\theta[X]\mathbb{E}_\theta[X]^T)^{-1}). \end{aligned} \tag{2.3.7}$$

By plugging in Eq. (2.3.6) into Eq. (2.3.5), we get the backward mapping of Ω as $(\mathbb{E}_\theta[XX^T] - \mathbb{E}_\theta[X]\mathbb{E}_\theta[X]^T)^{-1}) = \widehat{\Sigma}^{-1}$, easily computable from the sample covariance matrix.

2.3.3 Backward Mapping for Differential Network of Two GGMs

When the random variables $X_c, X_d \in \mathbb{R}^p$ follows the Gaussian Distribution $N(\mu_c, \Sigma_c)$ and $N(\mu_d, \Sigma_d)$, their density ratio (defined by [32]) essentially is a distribution in exponential families:

$$\begin{aligned} r(x,\Delta) &= \frac{p_d(x)}{p_c(x)} \\ &= \frac{\sqrt{\det(\Sigma_c)}\exp\left(-\frac{1}{2}(x-\mu_d)^T\Sigma_d^{-1}(x-\mu_d)\right)}{\sqrt{\det(\Sigma_d)}\exp\left(-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)\right)} \\ &= \exp(-\frac{1}{2}(x-\mu_d)^T\Sigma_d^{-1}(x-\mu_d) \\ &+ \frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c) \\ &- \frac{1}{2}(\log(\det(\Sigma_d)) - \log(\det(\Sigma_c)))) \\ &= \exp\left(-\frac{1}{2}\Delta x^2 + \mu_\Delta x - A(\mu_\Delta,\Delta)\right) \end{aligned} \tag{2.3.8}$$

Here $\Delta = \Sigma_d^{-1} - \Sigma_c^{-1}$ and $\mu_\Delta = \Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c$.

The log-partition function

$$\begin{aligned} A(\mu_\Delta,\Delta) = \frac{1}{2}\mu_d^T\Sigma_d^{-1}\mu_d - \frac{1}{2}\mu_c^T\Sigma_c^{-1}\mu_c + \\ \frac{1}{2}\log(\det(\Sigma_d)) - \frac{1}{2}\log(\det(\Sigma_c)) \end{aligned} \tag{2.3.9}$$

The canonical parameter

$$\begin{aligned} \theta &= \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Sigma_d^{-1} - \Sigma_c^{-1})\right) \\ &= \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Delta)\right) \end{aligned} \tag{2.3.10}$$

The sufficient statistics $\phi([X_c, X_d])$ and the log-partition function $A(\theta)$:

$$\begin{aligned} &\phi([X_c, X_d]) = ([X_c, X_d], [X_cX_c^T, X_dX_d^T]) \\ &A(\theta) = \frac{1}{2}\mu_d^T\Sigma_d^{-1}\mu_d - \frac{1}{2}\mu_c^T\Sigma_c^{-1}\mu_c + \\ &\frac{1}{2}\log(\det(\Sigma_d)) - \frac{1}{2}\log(\det(\Sigma_c)) \end{aligned} \tag{2.3.11}$$

And $h(x) = 1$.

$$\begin{aligned}\theta =& \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Sigma_d^{-1} - \Sigma_c^{-1})\right) \\ =& \mathcal{A}^*(v) = \nabla A^*(v)\end{aligned} \tag{2.3.12}$$

The mean parameter vector $v(\theta)$ includes the moments of the sufficient statistics $\phi()$ under the exponential distribution. It can be easily estimated through $\mathbb{E}[([X_c, X_d], [X_c X_c^T, X_d X_d^T])]$.

Therefore the backward mapping of θ becomes,

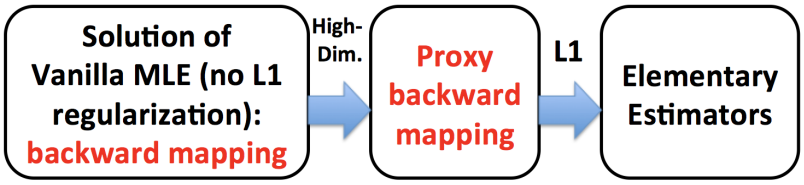
$$\begin{aligned}\widehat{\theta} =& (((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1}\mathbb{E}_\theta[X_d] \\ & - (\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1}\mathbb{E}_\theta[X_c]), \\ & - \frac{1}{2}((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1} - \\ & (\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1})).\end{aligned} \tag{2.3.13}$$

Because the second entry of the canonical parameter θ is $(\Sigma_d^{-1} - \Sigma_c^{-1})$, we get the backward mapping of Δ as

$$\begin{aligned}& ((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1} \\ & -(\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1}) \\ =& \widehat{\Sigma}_d^{-1} - \widehat{\Sigma}_c^{-1}\end{aligned} \tag{2.3.14}$$

This can be easily inferred from two sample covariance matrices $\widehat{\Sigma}_d$ and $\widehat{\Sigma}_c$ (Att: when under low-dimensional settings).

Figure 2.1: Basic idea of elementary estimators for graphical model.



2.4 Elementary Estimator

2.4.1 Elementary Estimators (EE):

Using the analysis framework from [27], recent studies [19,38,39] propose a new category of estimators named "Elementary estimator" (EE) with the following general formulation:

$$\begin{aligned} &\underset{\theta}{\operatorname{argmin}}\, \mathcal{R}(\theta) \\ &\text{subject to:} \mathcal{R}^*(\theta - \widehat{\theta}_n) \leq \lambda_n \end{aligned} \tag{2.4.1}$$

Where $\mathcal{R}^*(\cdot)$ is the dual norm of $\mathcal{R}(\cdot)$,

$$\mathcal{R}^*(v) := \sup_{u \neq 0} \frac{< u, v >}{\mathcal{R}(u)} = \sup_{\mathcal{R}(u) \leq 1} < u, v > . \tag{2.4.2}$$

The solution of Eq. (3.2.1) achieves the near optimal convergence rate as Eq. (2.1.5) when satisfying certain conditions. $\mathcal{R}(\cdot)$ represents a decomposable regularization function (e.g., ℓ_1-norm) and $\mathcal{R}^*(\cdot)$ is the dual norm of $\mathcal{R}(\cdot)$ (e.g., ℓ_∞-norm is the dual norm of ℓ_1-norm). λ_n is a regularization parameter.

The basic motivation of Eq. (3.2.1) is to build simpler and possibly fast estimators, that yet come with statistical guarantees that are nonetheless comparable to regularized MLE. $\widehat{\theta}_n$ needs to be carefully constructed, well-defined and closed-form for the purpose of simpler computations. The formulation defined by Eq. (3.2.1) is to ensure its solution having the desired structure defined by $\mathcal{R}(\cdot)$. For cases of high-dimensional estimation of linear regression models, $\widehat{\theta}_n$ can be the classical ridge estimator that itself is closed-form and with strong statistical convergence guarantees in high-dimensional situations. Related previous studies based on elementary estimators are summarized in Table 3.1.

2.4.2 EE-sGGM:

[19] proposed elementary estimators for graphical models (GM) of exponential families, in which $\widehat{\theta}_n$ represents so-called proxy of backward mapping for the target GM (more details in Section 2.3). The key idea (summarized in Figure 2.1) is to investigate the vanilla MLE and where it breaks down for estimating a graphical model of exponential families in the case of high-dimensions [19]. Essentially the vanilla graphical model MLE can be expressed as a backward mapping that computes the model parameters from some given moments in an exponential family distribution. For instance, in the case of learning Gaussian GM (GGM) with vanilla MLE, the backward mapping is $\widehat{\Sigma}^{-1}$ that estimates Ω from the sample covariance matrix (moment) $\widehat{\Sigma}$. We introduce the details of backward mapping in Section 2.3.

However, even though this backward mapping has a simple closed form for GGM, the backward mapping is normally not well-defined in high-dimensional settings. When given the sample covariance $\widehat{\Sigma}$, we cannot just compute the vanilla MLE solution as $[\widehat{\Sigma}]^{-1}$ for GGM since $\widehat{\Sigma}$ is rank-deficient when $p > n$. Therefore Yang et al. [19] used carefully constructed proxy backward maps as $\widehat{\theta}_n = [T_v(\widehat{\Sigma})]^{-1}$ that is both available in closed-form, and well-defined in high-dimensional settings for GGMs. We introduce the details of $[T_v(\widehat{\Sigma})]^{-1}$ and its statistical property in Section 2.3. Now Eq. (3.2.1) becomes the following closed-form estimator for learning sparse Gaussian graphical models [19]:

$$\begin{aligned} &\underset{\Omega}{\operatorname{argmin}} ||\Omega||_{1,,\text{off}} \\ &\text{subject to:} ||\Omega - [T_v(\widehat{\Sigma})]^{-1}||_{\infty,\text{off}} \leq \lambda_n \end{aligned} \tag{2.4.3}$$

Eq. (2.4.3) is a special case of Eq. (3.2.1), in which $\mathcal{R}(\cdot)$ is the off-diagonal ℓ_1-norm and the precision matrix Ω is the θ we search for. When $\mathcal{R}(\cdot)$ is the ℓ_1-norm, the solution of Eq. (3.2.1) (and Eq. (2.4.3)) just needs to perform entry-wise thresholding operations on $\widehat{\theta}_n$ to ensure the desired sparsity structure of its final solution.

Chapter 3

Joint Elementary Estimator and Theoretical Properties

3.1 Joint Elementary Estimator (JEE)

We aims to propose simple, scalable and theoretically-guaranteed joint estimators for estimating multiple sGGMs with additional knowledge in large-scale situations. We first propose to jointly estimate multiple related sGGMs from K data blocks using the following formulation:

$$\underset{\Omega^{(1)},\Omega^{(2)},\ldots,\Omega^{(K)}}{\operatorname{argmin}} \sum_{i=1}^{K} \mathcal{L}(\Omega^{(i)}) + \lambda_n \mathcal{R}(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) \tag{3.1.1}$$

where $\Omega^{(i)}$ denotes the precision matrix for i-th task. $\mathcal{L}(\Omega) = -\log\det(\Omega) + < \Omega, \widehat{\Sigma} >$ describes the negative log-likelihood function in sGGM. $\Omega^{(i)} \succ 0$ means that $\Omega^{(i)}$ needs to be a positive definite matrix. $\mathcal{R}(\cdot)$ represents a decomposable regularization function enforcing sparsity and structure assumptions.

For ease of notation, we denote that $\Omega^{tot} = (\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$ and $\Sigma^{tot} = (\Sigma^{(1)}, \Sigma^{(2)}, \ldots, \Sigma^{(K)})$. Ω^{tot} and Σ^{tot} are both $p \times Kp$ matrices (i.e., Kp^2 parameters to estimate). Now define an inverse function as $\text{inv}(A^{tot}) := ({A^{(1)}}^{-1}, {A^{(2)}}^{-1}, \ldots, {A^{(K)}}^{-1})$, where A_{tot} is a given $p \times Kp$ matrix with the

same structure as Σ_{tot}. Then we rewrite Eq. (3.1.1) into the following form:

$$\underset{\Omega^{tot}}{\operatorname{argmin}} \mathcal{L}(\Omega^{tot}) + \lambda_n \mathcal{R}(\Omega^{tot}) \tag{3.1.2}$$

Now connecting Eq. (3.1.2) to Eq. (2.1.5) and Eq. (3.2.1), we propose the following joint elementary estimator (JEE) for learning multiple sGGMs:

$$\begin{aligned} &\underset{\Omega^{tot}}{\operatorname{argmin}} \mathcal{R}(\Omega^{tot}) \\ &\text{subject to: } \mathcal{R}^*(\Omega^{tot} - \widehat{\Omega}^{tot}_{n_{tot}}) \leq \lambda_n \end{aligned} \tag{3.1.3}$$

The fundamental component in Eq. (3.2.1) for the single context sGGM was to use a well-defined proxy function to approximate the vanilla MLE solution (named as the backward mapping for exponential family distributions) [19]. The proposed proxy $\widehat{\theta}_n = [T_v(\widehat{\Sigma})]^{-1}$ is both well-defined under high-dimensional situations and also has a simple closed-form. Following a similar idea, when learning multiple sGGMs, we propose to use $inv(T_v(\widehat{\Sigma}^{tot}))$ for $\widehat{\Omega}^{tot}_{n_{tot}}$ and get the following joint elementary estimator:

$$\begin{aligned} &\underset{\Omega^{tot}}{\operatorname{argmin}} \mathcal{R}(\Omega^{tot}) \\ &\text{Subject to: } \mathcal{R}^*(\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot}))) \leq \lambda_n \end{aligned} \tag{3.1.4}$$

3.2 Theoretical Properties of EE and JEE

JEE formulation Eq. (3.1.4) and EE-sGGM Eq. (2.4.3) are special cases of the following generic formulation:

$$\begin{aligned} &\underset{\theta}{\operatorname{argmin}} \mathcal{R}(\theta) \\ &\text{subject to:} \mathcal{R}^*(\theta - \widehat{\theta}_n) \leq \lambda_n \end{aligned} \tag{3.2.1}$$

Where $\mathcal{R}^*(\cdot)$ is the dual norm of $\mathcal{R}(\cdot)$,

$$\mathcal{R}^*(v) := \sup_{u \neq 0} \frac{< u, v >}{\mathcal{R}(u)} = \sup_{\mathcal{R}(u) \leq 1} < u, v > . \tag{3.2.2}$$

Connecting Eq. (3.1.4) and Eq. (3.2.1), $\mathcal{R}()$ is the kw-norm. $\widehat{\theta}_n$ represents a close approximation of θ^*.

Following the unified framework [27], we first decompose the parameter space into a subspace pair$(\mathcal{M}, \bar{\mathcal{M}}^{\perp})$, where $\bar{\mathcal{M}}$ is the closure of $\mathcal{M}$. Here $\bar{\mathcal{M}}^{\perp} := \{v \in \mathbb{R}^p | < u, v >= 0, \forall u \in \bar{\mathcal{M}}\}$. $\mathcal{M}$ is the **model subspace** that typically has a much lower dimension than the original high-dimensional space. $\bar{\mathcal{M}}^{\perp}$ is the **perturbation subspace** of parameters. For further proofs, we assume the regularization function in Eq. (3.2.1) is **decomposable** w.r.t the subspace pair $(\mathcal{M}, \bar{\mathcal{M}}^{\perp})$.

(C1) $\mathcal{R}(u+v) = \mathcal{R}(u) + \mathcal{R}(v)$, $\forall u \in \mathcal{M}, \forall v \in \bar{\mathcal{M}}^{\perp}$.

[27] showed that most regularization norms are decomposable corresponding to a certain subspace pair.

Definition 3.2.1. ***Subspace Compatibility Constant***

Subspace compatibility constant is defined as $\Psi(\mathcal{M}, |\cdot|) := \sup_{u \in \mathcal{M} \setminus \{0\}} \frac{\mathcal{R}(u)}{|u|}$ *which captures the relative value between the error norm* $|\cdot|$ *and the regularization function* $\mathcal{R}(\cdot)$.

For simplicity, we assume there exists a true parameter θ^* which has the exact structure w.r.t a certain subspace pair. Concretely:

(C2) $\exists$ a subspace pair $(\mathcal{M}, \bar{\mathcal{M}}^{\perp})$ such that the true parameter satisfies $\text{proj}_{\mathcal{M}^{\perp}}(\theta^*) = 0$

Then we have the following theorem.

Theorem 3.2.2. *Suppose the regularization function in Eq. (3.2.1) satisfies condition* ***(C1)****, the true parameter of Eq. (3.2.1) satisfies condition* ***(C2)****, and* λ_n *satisfies that* $\lambda_n \geq \mathcal{R}^*(\widehat{\theta}_n - \theta^*)$*. Then, the optimal solution* $\widehat{\theta}$ *of Eq. (3.2.1) satisfies:*

$$\mathcal{R}^*(\widehat{\theta} - \theta^*) \leq 2\lambda_n \tag{3.2.3}$$

$$||\widehat{\theta} - \theta^*||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}) \tag{3.2.4}$$

$$\mathcal{R}(\widehat{\theta} - \theta^*) \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2 \tag{3.2.5}$$

3.2.1 Proof of Theorem (3.2.2)

Proof. Let $\delta := \widehat{\theta} - \theta^*$ be the error vector that we are interested in.

$$\begin{aligned}\mathcal{R}^*(\widehat{\theta}-\theta^*) &= \mathcal{R}^*(\widehat{\theta}-\widehat{\theta}_n+\widehat{\theta}_n-\theta^*)\\ &\leq \mathcal{R}^*(\widehat{\theta}_n-\widehat{\theta})+\mathcal{R}^*(\widehat{\theta}_n-\theta^*)\leq 2\lambda_n\end{aligned} \tag{3.2.6}$$

By the fact that $\theta^*_{\mathcal{M}^\perp}=0$, and the decomposability of $\mathcal{R}$ with respect to $(\mathcal{M},\bar{\mathcal{M}}^\perp)$

$$\begin{aligned}&\mathcal{R}(\theta^*)\\ &=\mathcal{R}(\theta^*)+\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]-\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]\\ &=\mathcal{R}[\theta^*+\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]-\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]\\ &\leq \mathcal{R}[\theta^*+\Pi_{\bar{\mathcal{M}}^\perp}(\delta)+\Pi_{\bar{\mathcal{M}}}(\delta)]+\mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)]\\ &-\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]\\ &=\mathcal{R}[\theta^*+\delta]+\mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)]-\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]\end{aligned} \tag{3.2.7}$$

Here, the inequality holds by the triangle inequality of norm. Since Eq. (3.2.1) minimizes $\mathcal{R}(\widehat{\theta})$, we have $\mathcal{R}(\theta^*+\Delta)=\mathcal{R}(\widehat{\theta})\leq\mathcal{R}(\theta^*)$. Combining this inequality with Eq. (6.3.2), we have:

$$\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]\leq\mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \tag{3.2.8}$$

Moreover, by Hlder's inequality and the decomposability of $\mathcal{R}(\cdot)$, we have:

$$\begin{aligned}&||\Delta||_2^2=\langle\delta,\delta\rangle\leq\mathcal{R}^*(\delta)\mathcal{R}(\delta)\leq 2\lambda_n\mathcal{R}(\delta)\\ &=2\lambda_n[\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta))+\mathcal{R}(\Pi_{\bar{\mathcal{M}}^\perp}(\delta))]\leq 4\lambda_n\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta))\\ &\leq 4\lambda_n\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2\end{aligned} \tag{3.2.9}$$

where $\Psi(\bar{\mathcal{M}})$ is a simple notation for $\Psi(\bar{\mathcal{M}},||\cdot||_2)$.

Since the projection operator is defined in terms of $||\cdot||_2$ norm, it is non-expansive: $||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2\leq||\Delta||_2$. Therefore, by Eq. (6.3.4), we have:

$$||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}), \tag{3.2.10}$$

and plugging it back to Eq. (6.3.4) yields the error bound Eq. (3.2.4).

Finally, Eq. (3.2.5) is straightforward from Eq. (6.3.3) and Eq. (6.3.5).

$$\begin{aligned} \mathcal{R}(\delta) &\leq 2\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 2\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2. \end{aligned} \tag{3.2.11}$$

□

3.2.2 Connecting to Previous Studies:

Most previous studies of multi-sGGMs follow the penalized MLE framework. Few works of Multi-task sGGM follow the CLIME formulation, since it is not easy to transfer two regularizers into the CLIME formulation (summarized in Table 3.1). Based on the authors' knowledge, no previous multi-sGGM studies have followed the elementary estimators(EE) formulation. As a simple soft-thresholding based estimator, elementary estimators (EE) have been used for other tasks as well. Table 3.1 summarizes three different types of previous tasks for which EE can be applied: high-dimensional regression, single sGGM and multi-sGGM. For comparison, we show how these tasks have been solved through the penalized likelihood framework in the second column and use the the third column to show studies following the CLIME formulation.

Table 3.1: Two categories of relevant studies differ over learning based on "penalized log-likelihood" or learning based on"elementary estimator"

Problems	**Penalized Likelihood**	**Elementary Estimator**
High dimensional linear regression	Lasso: $\underset{\beta}{\operatorname{argmin}} \|Y - \beta X\|_F + \lambda\|\beta\|_1$	$\underset{\beta}{\operatorname{argmin}} \|\beta\|_1$ subject to : $\|\beta - (X^T X + \epsilon I)^{-1} X^T y\|_\infty \leq \lambda_n$
sparse Gaussian Graphical Model	GLasso: $\underset{\Omega \geq 0}{\operatorname{argmin}} -logdet(\Omega) + < \Omega, \Sigma > + \lambda\|\Omega\|_1$	$\underset{\Omega \geq 0}{\operatorname{argmin}} \|\Omega\|_1$ subject to: $\|\Omega - [T_v(\Sigma)]^{-1}\|_\infty \leq \lambda_n$
Multi-task sGGM	Different Choices for Penalty $\mathcal{R}'$ $\underset{\Omega > 0}{\operatorname{argmin}} \sum_i (-L(\Omega_{tot}) + \lambda_1 \sum_i \|\Omega^{(i)}\|_1 + \lambda_2 \mathcal{R}'(\Omega_{tot})$	**Our methods: FASJEM, JEEK, DIFFEE**

Chapter 4

Method I: FASJEM – JEE for Enforcing Structural Assumptions

4.1 Method: A fast and scalable joint estimator for multi-sGGM

In this Chapter, we propose another novel model, namely fast and scalable joint estimator for multiple sGGM (FASJEM), for estimating multiple sGGMs jointly.

The penalized likelihood framework for multi-task sGGMs in Eq. (5.6.2) involves a hybrid of two regularization functions ($\ell_1 + \mathcal{R}'$). Studies in this direction cannot avoid the expensive steps like SVD and matrix multiplication and also require to store K covariance matrices in the main memory. Since this paper aims to design a scalable joint estimator for multi-sGGM under large-scale settings, extending the elementary estimator of single-task sGGM [19] to multi-task formulation becomes a natural choice.

For multi-task sGGMs, we can denote that $\Omega_{tot} = (\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)})$ and $\Sigma_{tot} = (\Sigma^{(1)}, \Sigma^{(2)}, \ldots, \Sigma^{(K)})$. Ω_{tot} and Σ_{tot} are both $p \times Kp$ matrices (i.e., Kp^2 parameters to estimate). Now define an inverse

function as $\text{inv}(A_{tot}) := ({A^{(1)}}^{-1}, {A^{(2)}}^{-1}, \ldots, {A^{(K)}}^{-1})$, where A_{tot} is a given $p \times Kp$ matrix with the same structure as Σ_{tot}. Furthermore, we add a new hyperparameter variable $\epsilon = \frac{\lambda'_n}{\lambda_n}$.

Let $I = \{1, 2\}$ and $\theta_1 = \theta_2 = \frac{1}{2}\Omega_{tot}$. We can clearly tell that Eq. (5.6.2) is a special case of the superposition structured estimation in Eq. (4.4.1). The ESS (elementary superposition structured) moment estimator (Eq. (4.4.2)) extends the elementary estimator of structured covariance matrix to elementary superposition-structured estimator for estimating covariance matrices with a hybrid structure (e.g., sparse + low rank). This motivates us to propose the following elementary superposition estimator for learning multi-task sGGM:

$$\begin{aligned} &\underset{\Omega_{tot}}{\text{argmin}}\, ||\Omega_{tot}||_1 + \epsilon\mathcal{R}'(\Omega_{tot}) \\ &s.t. ||\Omega_{tot} - \text{inv}(T_v(\widehat{\Sigma}_{tot}))||_\infty \leq \lambda_n \\ &\mathcal{R}'^*(\Omega_{tot} - \text{inv}(T_v(\widehat{\Sigma}_{tot}))) \leq \epsilon\lambda_n \end{aligned} \tag{4.1.1}$$

Here $||\cdot||_1^* = ||\cdot||_\infty$ (the dual norm of l_1-norm is l_∞-norm). $\mathcal{R}'(\cdot)$ represents a regularizer on Ω_{tot} to enforce that $\{\Omega^{(i)}\}$ share certain similarity. $\mathcal{R}'^*(\cdot)$ is the dual norm of $\mathcal{R}'(\cdot)$. We name this novel formulation as FASJEM. By varying $\mathcal{R}'(\cdot)$, we can get a variety of FASJEM estimators.

Section 4.2 theoretically proves the convergence rate of FASJEM as $O(\log(Kp)/n_{tot})$. Our theory proof is inspired by the ESS moment estimator [39], the SS estimator [40] and the EE-sGGM [19].

4.1.1 Method I: FASJEM-G

For multi-task regularization, the first $\mathcal{R}'(\cdot)$ we try is the $\mathcal{G}, 2$-norm (i.e., $\mathcal{R}'(\cdot) = ||\cdot||_{\mathcal{G},2}$). This norm is inspired by JGL-group lasso [11]. $\mathcal{G}, 2$-norm constrains the parameters in the same group to have the same level of sparsity. In multi-task sGGMs, group set $\mathcal{G} := \{g_{j,k}\}$, where $g_{j,k} = \{\Omega^{(i)}_{j,k} | i = 1, \ldots, K\}$. Suppose g is an arbitrary group in group set $\mathcal{G}$ and totally we have p^2 groups. $||\Omega_{tot}||_{\mathcal{G},2} = \sum_{j=1}^{p}\sum_{k=1}^{p} ||(\Omega^{(1)}_{j,k}, \Omega^{(2)}_{j,k}, \ldots, \Omega^{(i)}_{j,k}, \ldots, \Omega^{(K)}_{j,k})||_2$. When $\mathcal{R}'(\cdot) = ||\cdot||_{\mathcal{G},2}$, we name Eq. (4.1.1) as FASJEM-G (short form of FASJEM-Group2). We solve FASJEM-G using a parallel proximal based optimization formulation from [41]. Algorithm 1 summarizes the detailed optimization steps and the four proximity operators implemented on GPU are listed in Table 4.1[1]. The optimization sequence of Algorithm 1 converges Q-linearly (See Eq. (4.1.13)).

[1]The non-GPU version of the four proximity operators are in Eq. (4.1.5) to Eq. (4.1.8).

4.1.2 Proximal Algorithm for Optimization

Eq. (4.1.1) includes a convex programming task since the norms we choose are convex. By simplifying notations and adding another parameter, we reformulate it to:

$$\begin{aligned} &\underset{\theta_1,\theta_2}{\operatorname{argmin}}\ f_1(\theta_1) + f_2(\theta_2) \\ \text{subject to }: &||\theta_1 - \text{inv}(T_v(\widehat{\Sigma}_{tot}))||_\infty \leq \lambda_n \\ &\mathcal{R}'^*(\theta_2 - \text{inv}(T_v(\widehat{\Sigma}_{tot}))) \leq \epsilon\lambda_n \\ &\theta_1 = \theta_2 \end{aligned} \tag{4.1.2}$$

Where $f_1(\cdot) = ||\cdot||_1$ and $f_2(\cdot) = \epsilon||\cdot||_{\mathcal{G},2}$. Then we convert Eq. (4.1.2) to the following equivalent and distributed formulation:

$$\begin{aligned} &\underset{\theta_1,\theta_2,\theta_3,\theta_4}{\operatorname{argmin}}\ f_1(\theta_1) + f_2(\theta_2) + f_3(\theta_3) + f_4(\theta_4) \\ &\text{subject to: } \theta_1 = \theta_2 = \theta_3 = \theta_4 \end{aligned} \tag{4.1.3}$$

Here $f_3(\theta) = \mathcal{I}_{\{||\theta - inv(T_v(\Sigma_{tot}))||_\infty \leq \lambda_n\}}(\theta)$ and $f_4(\theta) = \mathcal{I}_{\{||\theta - inv(T_v(\Sigma_{tot}))||^*_{\mathcal{G},2} \leq \epsilon\lambda_n\}}(\theta)$. $\mathcal{I}_C(\cdot)$ represents the indicator function of a convex set C as $\mathcal{I}_C(x) = 0$ when $x \in C$. Otherwise $\mathcal{I}_C(x) = \infty$. To solve Eq. (4.1.3), we choose a parallel proximal based algorithm [41] summarized in Algorithm 1. Besides the distributed nature, the proximal algorithm also bring in the benefit that many proximity operators are entry-wise operators for the targeted parameters. The four proximal operators for four functions $\{f_1, f_2, f_3, f_4\}$ (for CPU platform implementation) are included in the Equations Eq. (4.1.5) to Eq. (4.1.8) in Section 4.1. With the benefits as proximal operators, Eq. (4.1.5) and Eq. (4.1.7) are entry-wise and Eq. (4.1.6) and Eq. (4.1.8) are group entry-wise.

Proximal Optimization: The proximal algorithm only needs to calculate the proximity operator of the parameters to be optimized. The proximity operator in proximal algorithms is defined as:

$$\text{prox}_{\gamma f}(x) = \underset{y}{\operatorname{argmin}}(f(y) + (\frac{1}{2\gamma}||x - y||_2^2)). \tag{4.1.4}$$

The benefit of the proximal algorithm is that many proximity operators are entry-wise operators for the targeted parameters. The parallel proximal (initially called proximity splitting) algorithm [41] belongs to the general family of distributed convex optimization that optimizes in such a way that

each term (in this case, each proximity operator) can be handled by its own processing element, such as a thread or processor.

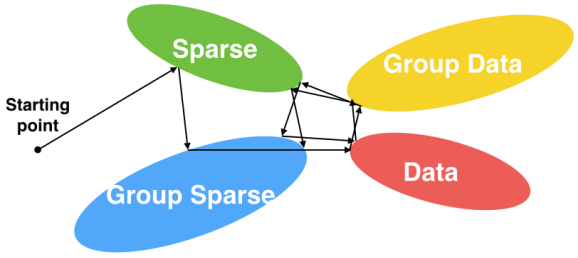


Figure 4.1: A simple figure to show how our optimization method works. Our optimization approach is a method with linear convergence rate in finding the optimal point. It considers four properties : (1) information from the raw data; (2) information from the group data; (3)sparsity property; (4) group sparsity property.

Four proximity operators for CPU implementation of FASJEM-G: In the following, we denote $x = \Omega_{tot}$, $a = \Sigma_{tot}$ and $g \in \mathcal{G}$ to simply notations. Eq. (4.1.5) and Eq. (4.1.7) are entry-wise operators and Eq. (4.1.6) and Eq. (4.1.8) are group entry-wise. Group entry-wise means in calculation, the operator can compute each group of entries independently from other groups. Entry-wise means the calculation of each entry is only related to itself). The optimization process of Algorithm 1 iterating among four proximal operators is visualized by Figure 4.1.

For $f_1(\cdot) = ||\cdot||_1$.

$$\begin{aligned} \text{prox}_{\gamma f_1}(x) &= \text{prox}_{\gamma||\cdot||_1}(x) \\ &= \begin{cases} x_{j,k}^{(i)} - \gamma, & x_{j,k}^{(i)} > \gamma \\ 0, & |x_{j,k}^{(i)}| \leq \gamma \\ x_{j,k}^{(i)} + \gamma, & x_{j,k}^{(i)} < -\gamma \end{cases} \end{aligned} \tag{4.1.5}$$

Eq. (4.1.5) is the closed form solution of Eq. (4.1.4) when $f = |\cdot|_1$. Here $j, k = 1, \ldots, p$ and $i = 1, \ldots, K$. This is an entry-wise operator (i.e., the calculation of each entry is only related to itself).

Similarly, $f_2(\cdot) = ||\cdot||_{\mathcal{G},2}$

$$\begin{aligned} \text{prox}_{\gamma f_2}(x_g) &= \text{prox}_{\gamma||\cdot||_{\mathcal{G},2}}(x_g) \\ &= \begin{cases} x_g - \gamma \frac{x_g}{||x_g||_2}, & ||x_g||_2 > \gamma \\ 0, & ||x_g||_2 \leq \gamma \end{cases} \end{aligned} \tag{4.1.6}$$

Here $g \in \mathcal{G}$. This is a group entry-wise operator (computing a group of entries is not related to other groups).

$f_3(\cdot)$ and $f_4(\cdot)$ include function forms of $\mathcal{I}_{f(\cdot)<D}$ and $\text{prox}_{\mathcal{I}_{\{f(\cdot)<D\}}} = \text{proj}_{\{f(\cdot)<D\}}$, where proj_C means the projection function to the convex set C. We can obtain

$$\begin{aligned} \text{prox}_{\gamma f_3}(x) &= \text{proj}_{||x-a||_\infty \leq \lambda} \\ &= \begin{cases} x_{,k}^{(i)}, \ |x_{j,k}^{(i)} - a_{j,k}^{(i)}| \leq \lambda \\ a_{,k}^{(i)} + \lambda, \ x_{j,k}^{(i)} > a_{j,k}^{(i)} + \lambda \\ a_{,k}^{(i)} - \lambda, \ x_{j,k}^{(i)} < a_{j,k}^{(i)} - \lambda \end{cases} \end{aligned} \tag{4.1.7}$$

where $j, k = 1, \ldots, p$ and $i = 1, \ldots, K$. This operator is entry-wise (i.e., only related to each entry of x and a).

$$\begin{aligned} \text{prox}_{\gamma f_4}(x_g) &= \text{proj}_{||x-a||_{\mathcal{G},2}^* \leq \lambda} \\ &= \begin{cases} x_g, \ ||x_g - a_g||_2 \leq \lambda \\ \lambda \frac{x_g - a_g}{||x_g - a_g||_2} + a_g, \ ||x_g - a_g||_2 > \lambda \end{cases} \end{aligned} \tag{4.1.8}$$

This operator is group entry-wise.

Four proximity operators for GPU parallel implementation of FASJEM-G: The four proximity operators on GPU are summarized in Table 4.1. More details as following:

For Eq. (4.1.5),

$$\begin{aligned} &\text{prox}_{\gamma f_1}(x) = \text{prox}_{\gamma||\cdot||_1}(x) \\ &= \max((x_{j,k}^{(i)} - \gamma), 0) + \min(0, (x_{j,k}^{(i)} + \gamma)) \end{aligned} \tag{4.1.9}$$

For Eq. (4.1.6)

$$\begin{aligned} \text{prox}_{\gamma f_2}(x_g) &= \text{prox}_{\gamma||\cdot||_{\mathcal{G},2}}(x_g) \\ &= x_g \max((1 - \frac{\gamma}{||x_g||_2}), 0) \end{aligned} \tag{4.1.10}$$

Table 4.1: Four proximity operators implemented on GPU platform.

$[\text{prox}_{\gamma f_1}(x)]^{(i)}_{j,k}$	$\max((x^{(i)}_{j,k} - \gamma), 0) + \min(0, (x^{(i)}_{j,k} + \gamma))$
$\text{prox}_{\gamma f_2}(x_g)$	$x_g \max((1 - \frac{\gamma}{\|\|x_g\|\|_2}), 0)$
$[\text{prox}_{\gamma f_3}(x)]^{(i)}_{j,k}$	$\min(\max(x^{(i)}_{j,k} - a^{(i)}_{j,k}, -\lambda_n), \lambda_n) + a^{(i)}_{j,k}$
$\text{prox}_{\gamma f_4}(x_g)$	$\max(\frac{\lambda_n}{\|\|x_g - a_g\|\|_2}, 1)(x_g - a_g) + a_g$

For Eq. (4.1.7)

$$\begin{aligned} \text{prox}_{\gamma f_3}(x) &= \text{proj}_{||x-a||_\infty \leq \lambda} \\ &= \min(\max(x^{(i)}_{j,k} - a^{(i)}_{j,k}, -\lambda), \lambda) + a^{(i)}_{j,k} \end{aligned} \tag{4.1.11}$$

For Eq. (4.1.8)

$$\begin{aligned} \text{prox}_{\gamma f_4}(x) &= \text{proj}_{||x-a||^*_{\mathcal{G},2} \leq \lambda} \\ &= \max(\frac{\lambda}{||x_g - a_g||_2}, 1)(x_g - a_g) + a_g \end{aligned} \tag{4.1.12}$$

Here $j, k = 1, \ldots, p$, $i = 1, \ldots, K$ and $g \in \mathcal{G}$.

Q-linearly Convergence of Optimization: The proposed optimization is a first-order method. Based on the recent study [42], the optimization sequence $\{\Omega^i\}$(for $i = 1$ to t iteration) converges Q-linearly. Q-linearly means:

$$\limsup_{k \to \infty} \frac{||\Omega^{k+1} - \Omega^*||}{||\Omega^k - \Omega^*||} \leq \rho \tag{4.1.13}$$

Algorithm 1 Parallel proximal algorithm[2]

input K given data blocks $X^{(1)}, X^{(2)}, \ldots, X^{(K)}$. Hyper-parameter:$\alpha$, ϵ, v, λ_n and γ. Learning rate: $0 < \rho < 2$. Max iteration number $iter$.

output Ω_{tot}

Compute Σ_{tot} from $X^{(1)}, X^{(2)}, \ldots, X^{(K)}$

Initialize $\theta^0 = \text{inv}(T_v(\Sigma_{tot})), \theta_j^0 = \text{inv}(T_v(\Sigma_{tot}))$ for $j \in \{1, 2, 3, 4\}$ and $a = \text{inv}(T_v(\Sigma_{tot}))$.

for $i = 0$ **to** $iter$ **do**

  $p_1^i = \text{prox}_{4\gamma f_1}\theta_1^i$

  $p_2^i = \text{prox}_{4\gamma f_2}\theta_2^i$

  $p_3^i = \text{prox}_{4\gamma f_3}\theta_3^i$

  $p_4^i = \text{prox}_{4\gamma f_4}\theta_4^i$

  $p^i = \frac{1}{4}(\sum_{j=1}^{4}\theta_j^i)$

  for $j = 1, 2, 3, 4$ **do**

    $\theta_j^{i+1} = \theta_j^i + \rho(2p^i - \theta^i - p_j^i)$

  end for

  $\theta^{i+1} = \theta^i + \rho(p^i - \theta^i)$

end for

$\Omega_{tot} = \theta^{iter}$

output Ω_{tot}

4.1.3 Method II: FASJEM-I

As shown in Section 4.4, most previous models for multi-task sGGMs varied the second norm R' to obtain different models. Similarly we can easily change $R'(\cdot)$ in Eq. 4.1.1 into any other desired norm to extend our FASJEM. For instance, we can change $R'(\cdot)$ to group-infinity norm $||\cdot||_{\mathcal{G},\infty}$.

$||\Omega_{tot}||_{\mathcal{G},\infty} = \sum_{j=1}^{p}\sum_{k=1}^{p}||(\Omega_{j,k}^{(1)}, \Omega_{j,k}^{(2)}, \ldots, \Omega_{j,k}^{(i)}, \ldots, \Omega_{j,k}^{(K)})||_\infty$. This norm is inspired by a multi-task sGGM proposed by [14]. When using group-infinity norm, we get FASJEM-I(short for FASJEM-Groupinf). We can derive the optimization for FASJAM-I by changing two proximities in Algorithm 1. Considering that the original formulation in [14] is similar with JGL [11], in the rest of this paper, we call the model from [14] as JGL-groupinf or JGL-I (the corresponded baseline for FASJEM-I).

[2] Four proximity operators used on GPU are defined in Table 4.1. Hyperparameters are explained in Section 4.5. Here $j, k = 1, \ldots, p$, $i = 1, \ldots, K$ and $g \in \mathcal{G}$.

4.1.4 FASJEM with Additional Knowledge

We can also extend FASJEM by adding the additional knowledge into the formulation. By replacing the weighted-ℓ_1 norm into the formulation, we have the FASJEM-K as follows:

$$\begin{aligned} &\underset{\Omega_{tot}}{\operatorname{argmin}} ||W_{tot} \circ \Omega_{tot}||_1 + \epsilon \mathcal{R}'(\Omega_{tot}) \\ &s.t. ||W_{tot} \circ (\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot})))||_\infty \leq \lambda_n \\ &\mathcal{R}'^*(\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot}))) \leq \epsilon \lambda_n \end{aligned} \tag{4.1.14}$$

4.2 Theoretical Analysis

In this section, we prove that our estimator can be optimized asynchronously in a group entry-wise manner. We also provide the proof of the theoretical error bounds of FASJEM.

4.2.1 Group entry-wise and parallelizing optimizable

Theorem 4.2.1. *(**FASJEM is Group entry-wise optimizable**) Suppose we use FASJEM to infer multiple inverse of covariance matrices summarized as $\widehat{\Omega}_{tot}$. $\{\widehat{\Omega}^{(i)}_{j,k} | i = 1, \ldots, K\}$ describes a group of K entries at (j,k) position. Varying $j \in \{1, 2, \ldots, p\}$ and $k \in \{1, 2, \ldots, p\}$, we have totally $p \times p$ groups. If these groups are independently estimated by FASJEM, then we have,*

$$\bigcup_{j=1}^{p} \bigcup_{k=1}^{p} \{\widehat{\Omega}^{(i)}_{j,k} | i = 1, \ldots, K\} = \widehat{\Omega}_{tot}. \tag{4.2.1}$$

Proof. Eq. (4.1.9) and Eq. (4.1.11) are soft-thresholding based operators on each entry. Eq. (4.1.10) and Eq. (4.1.12) are soft-thresholding operators on each group of entries. □

Corollary 4.2.2. *We can decompose FASJEM into $p \times p$ subproblems that are independent from each other, and solve each subproblem at a time. Therefore our estimator only requires $O(K)$ memory storage for computation.*

This corollary proves the claims we showed in section 4.4. Through Theorem (5.2.1), it is important to notice that the optimization on multiple groups of entries can be totally **parallelized**.

4.2.2 Theoretical error bounds

In this subsection, we first provide the error bounds for elementary super-position estimator (ESS estimator) under $I = \{1, 2\}$. We then use this general bound to prove the error bound for FASJEM-G. All the proofs are included in Section 4.3. We also include the error bounds for elementary estimator (EE) in Section 2.4.

Extension to ESS: For the multiple-task case, we need to consider two or more regularization functions. For instance, in FASJEM-G we assume the sparsity of parameter and the group sparsity among tasks. Since we only consider the models with two regularization function, we consider the error bounds of the following elementary super-position estimator formulation in the rest of the section.

$$\begin{aligned} &\underset{\theta_1,\theta_2}{\operatorname{argmin}} \lambda_1 \mathcal{R}_1(\theta_1) + \lambda_2 \mathcal{R}_2(\theta_2) \\ &\text{subject to:} \mathcal{R}_i^*(\widehat{\theta}_n - (\theta_1 + \theta_2)) \leq \lambda_i \ , i = 1, 2 \end{aligned} \tag{4.2.2}$$

This equation restricts the number of penalty functions to 2. Similar to the single-task error bounds (in Section 2.4) , we naturally extend condition **(C2)** to the following condition:

(C3) $\text{proj}_{\mathcal{M}_i^{\perp}}(\theta_i^*) = 0$, $i = 1, 2$.

We borrow the following condition from [40], which is a structural incoherence condition ensuring that the non-interference of different structures.

(C4) Let $\Phi := \max\{2 + \frac{3\lambda_1 \Psi_1(\bar{\mathcal{M}}_1)}{\lambda_2 \Psi_2(\bar{\mathcal{M}}_2)}, 2 + \frac{3\lambda_2 \Psi_2(\bar{\mathcal{M}}_2)}{\lambda_1 \Psi_1(\bar{\mathcal{M}}_1)}\}$.
$\max\{\sigma_{\max}(\mathcal{P}_{\bar{\mathcal{M}}_1} \mathcal{P}_{\bar{\mathcal{M}}_2}),$
$\sigma_{\max}(\mathcal{P}_{\bar{\mathcal{M}}_1} \mathcal{P}_{\bar{\mathcal{M}}_2^{\perp}}) \sigma_{\max}(\mathcal{P}_{\bar{\mathcal{M}}_1^{\perp}} \mathcal{P}_{\bar{\mathcal{M}}_2^{\perp}})\} \leq \frac{1}{16\Phi^2}$

where $\mathcal{P}_{\bar{\mathcal{M}}}$ is the matrix corresponding to the projection operator for the subspace $\bar{\mathcal{M}}$. The definition of $\Psi(\cdot)$ are included in Definition (3.2.1).

With these two conditions, we have the following theorem:

Theorem 4.2.3. *Suppose that the true parameter θ^* satisfies the conditions **(C3)(C4)** and $\lambda_i \geq \mathcal{R}_i^*(\widehat{\theta} - \theta^*)$, then the optimal point $\widehat{\theta}$ of Eq. (4.2.2) has the following error bounds:*

$$\mathcal{R}_i^*(\widehat{\theta} - \theta^*) \leq 2\lambda_i,\ i = 1, 2 \tag{4.2.3}$$

$$\mathcal{R}_i(\widehat{\theta} - \theta^*) \leq \frac{32}{\lambda_i}(\max_i \lambda_i \Psi(\bar{\mathcal{M}}_i))^2,\ i = 1, 2 \tag{4.2.4}$$

$$||\widehat{\theta} - \theta^*||_F \leq 8 \max_i \lambda_i \Psi(\bar{\mathcal{M}}_i) \tag{4.2.5}$$

Notice that for FASJEM-G model, $\mathcal{R}_1 = ||\cdot||_1$ and $\mathcal{R}_2 = ||\cdot||_{\mathcal{G},2}$. Based on the results in [27], $\Psi(\bar{\mathcal{M}}_1) = \sqrt{s}$ and $\Psi(\bar{\mathcal{M}}_2) = \sqrt{s_{\mathcal{G}}}$, where s is the number of nonzero entries in Ω_{tot} and s_g is the number of groups in which there exists at least one nonzero entry. Clearly $s > s_g$. Also in practice, to utilize group information, we have to choose hyperparameter $\lambda_n > \lambda_n'$ ($\lambda_1 > \lambda_2$ in Eq. (4.2.5)). Therefore by Theorem (4.2.3), we have the following theorem,

Theorem 4.2.4. *Suppose that $\mathcal{R}_1 = ||\cdot||_1$ and $\mathcal{R}_2 = ||\cdot||_{\mathcal{G},2}$ and the true parameter Ω_{tot}^* satisfies the conditions **(C3)(C4)** and $\lambda_i \geq \mathcal{R}_i^*(\widehat{\Omega}_{tot} - \Omega_{tot}^*)$, then the optimal point $\widehat{\Omega}_{tot}$ of Eq. (4.1.1) has the following error bounds: $||\widehat{\Omega}_{tot} - \Omega_{tot}^*||_F \leq 8\sqrt{s}\lambda_n$.*

We then derive a corollary of Theorem (6.3.1) for FASJEM-G. A prerequisite is to show that $inv(T_v(\widehat{\Sigma}_{tot}))$ is well-defined. The following conditions define a broad class of sGGM that satisfy the requirement. Similar results are also introduced by [19].

Conditions for elementary estimator of sGGM: **C-MinInfΣ** The true parameter Ω_{tot}^* of Eq. (4.2.2) has bounded induced operator norm, i.e., $|||{\Omega^{(i)}}^*|||_\infty := \sup_{w \neq 0 \in \mathbb{R}^p} \frac{||{\Sigma^{(i)}}^* w||_\infty}{|w|_\infty} \leq \kappa_1 \forall i$.

C-SparseΣ The true multiple covariance matrices $\Sigma_{tot}^* := inv(\Omega_{tot}^*)$ are "approximately sparse" along the lines [43] : for some positive constant D, ${\Sigma_{j,j}^{(i)}}^* \leq D$ for all diagonal entries. Moreover, for some $0 \leq q < 1$ and $c_0(p)$, $\max_i \sum_{j=1}^{p} |{\Sigma_{j,k}^{(i)}}^*|^q \leq c_0(p) \forall i$. If $q = 0$, then this condition reduce to Σ^* being sparse. We additionally require $\inf_{w \neq 0 \in \mathbb{R}^p} \frac{|{\Omega^{(i)}}^* w|_\infty}{|w|_\infty} \geq \kappa_2$.

Error bounds of FASJEM-group: In FASJEM, $\theta_1^* = \theta_2^* = \frac{1}{2}\theta^*$. θ_i is the parameter w.r.t a subspace pair$(\mathcal{M}_i, \bar{\mathcal{M}}_i^\perp)$, where $i = 1, 2$.

Here $\mathcal{R}_1 = ||\cdot||_1$ and $\mathcal{R}_2 = ||\cdot||_{\mathcal{G},2}$. We assume the true parameter θ^* satisfies **C-MinInf**Σ and **C-Sparse**Σ conditions. Using the above theorems, we have the following corollary:

Corollary 4.2.5. *If we choose hyperparameters $\lambda'_n < \lambda_n$. Let $v := a\sqrt{\frac{\log p'}{n_{tot}}}$ for $p' = \max(Kp, n_{tot})$. Then for $\lambda_n := \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_{tot}}}$ and $n_{tot} > c\log p'$, with a probability of at least $1-2C_1\exp(-C_2 Kp\log(Kp))$, the estimated optimal solution $\widehat{\Omega}_{tot}$ has the following error bound:*
$||\widehat{\Omega}_{tot} - \Omega^*_{tot}||_F \leq 32\frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{s\log p'}{n_{tot}}}\}$

where a, c, κ_1 and κ_2 are constants.

The convergence rate of single-task sGGM is $O(\log p/n_i)$. In high-dimensional setting, $p' = Kp$ since $Kp > n_{tot}$. Assuming $n_i = \frac{n_{tot}}{K}$, the convergence rate of single sGGM is $O(K\log p/n_{tot})$. Clearly, since $K\log p > \log(Kp)$, the convergence rate of FASJEM is better than single-task sGGM.

4.3 Proof

Proof of Theorem (3.2.2)

Proof. Let $\Delta := \widehat{\theta} - \theta^*$ be the error vector that we are interested in.

$$\begin{aligned}\mathcal{R}^*(\widehat{\theta} - \theta^*) &= \mathcal{R}^*(\widehat{\theta} - \widehat{\theta}_n + \widehat{\theta}_n - \theta^*) \\ &\leq \mathcal{R}^*(\widehat{\theta}_n - \widehat{\theta}) + \mathcal{R}^*(\widehat{\theta}_n - \theta^*) \leq 2\lambda_n\end{aligned} \tag{4.3.1}$$

By the fact that $\theta^*_{\mathcal{M}^\perp} = 0$, and the decomposability of $\mathcal{R}$ with respect to $(\mathcal{M}, \bar{\mathcal{M}}^\perp)$

$$\begin{aligned}&\mathcal{R}(\theta^*) \\ &= \mathcal{R}(\theta^*) + \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] \\ &= \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] \\ &\leq \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\Delta) + \Pi_{\bar{\mathcal{M}}}(\Delta)] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\Delta)] \\ &- \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] \\ &= \mathcal{R}[\theta^* + \Delta] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\Delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)]\end{aligned} \tag{4.3.2}$$

Here, the inequality holds by the triangle inequality of norm. Since EE minimizes $\mathcal{R}(\widehat{\theta})$, we have $\mathcal{R}(\theta^* + \Delta) = \mathcal{R}(\widehat{\theta}) \leq \mathcal{R}(\theta^*)$. Combining this inequality with Eq. (6.3.2), we have:

$$\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\Delta)] \leq \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\Delta)] \tag{4.3.3}$$

Moreover, by Hlder's inequality and the decomposability of $\mathcal{R}(\cdot)$, we have:

$$\begin{aligned} &||\Delta||_2^2 = \langle \Delta, \Delta \rangle \leq \mathcal{R}^*(\Delta)\mathcal{R}(\Delta) \leq 2\lambda_n \mathcal{R}(\Delta) \\ &= 2\lambda_n[\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\Delta)) + \mathcal{R}(\Pi_{\bar{\mathcal{M}}^\perp}(\Delta))] \leq 4\lambda_n \mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\Delta)) \\ &\leq 4\lambda_n \Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \end{aligned} \tag{4.3.4}$$

where $\Psi(\bar{\mathcal{M}})$ is a simple notation for $\Psi(\bar{\mathcal{M}}, ||\cdot||_2)$.

Since the projection operator is defined in terms of $||\cdot||_2$ norm, it is non-expansive: $||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \leq ||\Delta||_2$. Therefore, by Eq. (6.3.4), we have:

$$||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}), \tag{4.3.5}$$

and plugging it back to Eq. (6.3.4) yields the error bound Eq. (3.2.4).

Finally, Eq. (3.2.5) is straightforward from Eq. (6.3.3) and Eq. (6.3.5).

$$\begin{aligned} &\mathcal{R}(\Delta) \leq 2\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\Delta)) \\ &\leq 2\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2. \end{aligned} \tag{4.3.6}$$

□

Proof of Theorem (4.2.3)

Proof. In this proof, we consider the matrix parameter such as the covariance. $I = \{1, 2\}$ in the following contents. Basically, the Frobenius norm can be simply replaced by ℓ_2 norm for the vector parameters. Let $\Delta_i := \widehat{\theta}_i - \theta_i^*$, and $\Delta = \widehat{\theta} - \theta^* = \Sigma_{i \in I} \Delta_i$. The error bound Eq. (4.2.3) can be easily shown from the assumption in the statement with the constraint of Eq. (4.2.2). For every $i \in I$,

$$\begin{aligned} &\mathcal{R}_i^*(\Delta) = \mathcal{R}_i^*(\widehat{\theta} - \theta^*) = \mathcal{R}_i^*(\widehat{\theta} - \widehat{\theta}_n + \widehat{\theta}_n - \theta^*) \\ &\leq \mathcal{R}_i^*(\widehat{\theta}_n - \widehat{\theta}) + \mathcal{R}_i^*(\widehat{\theta}_n - \theta^*) \leq 2\lambda_i. \end{aligned} \tag{4.3.7}$$

By the similar reasoning as in Eq. (6.3.2) with the fact that $\Pi_{\mathcal{M}_i^{\perp}}(\theta_i^*) = 0$ in **C3**, and the decomposability of $\mathcal{R}_i$ with respect to $(\mathcal{M}_i, \widehat{\mathcal{M}}_i^{\perp})$, we have:

$$\begin{aligned} \mathcal{R}_i(\theta_i^*) \leq &\mathcal{R}_i[\theta_i^* + \Delta_i] + \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] \\ &- \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i^{\perp}}(\Delta_i)]. \end{aligned} \tag{4.3.8}$$

Since $\left\{\widehat{\theta}_i\right\}_{i \in I}$ minimizes the objective function of Eq. (4.2.2),

$$\begin{aligned} \sum_{i \in I} \lambda_i \mathcal{R}_i(\widehat{\theta}_i) \leq \sum_{i \in I} \lambda_i \{&\mathcal{R}_i(\theta_i^* + \Delta_i) \\ &\mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] - \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i^{\perp}}(\Delta_i)]\}, \end{aligned} \tag{4.3.9}$$

Which implies

$$\sum_{i \in I} \lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i^{\perp}}(\Delta_i)] \leq \sum_{i \in I} \lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] \tag{4.3.10}$$

Now, for each structure $i \in I$, we have an application for Hlder's inequality: $|\langle \Delta, \Delta_i \rangle| \leq \mathcal{R}_i^*(\Delta)\mathcal{R}_i(\Delta_i) \leq 2\lambda_i \mathcal{R}_i(\Delta_i)$ where the notation $\langle\langle A, B \rangle\rangle$ denotes the trace inner product, $\text{trace}(A^T B) = \Sigma_i \Sigma_j A_{ij} B_{ij}$, and we use the pre-computed bound in Eq. (4.3.7). Then, the Frobenius error $||\Delta||_F$ can be upper-bounded as follows:

$$
\begin{aligned}
&||\Delta||_F^2 = \langle\langle \Delta, \Delta \rangle\rangle = \sum_{i \in I} \langle\langle \Delta, \Delta_i \rangle\rangle \leq \sum_{i \in I} |\langle\langle \Delta, \Delta_i \rangle\rangle| \\
&\leq 2 \sum_{i \in I} \lambda_i \mathcal{R}_i(\Delta_i) \leq 2 \sum_{i \in I} \{\lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] + \\
&\lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i^{\perp}}(\Delta_i)]\} \leq 4 \sum_{i \in I} \lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] \\
&\leq 4 \sum_{i \in I} \lambda_i \Psi(\bar{\mathcal{M}}_i) ||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F
\end{aligned}
\tag{4.3.11}
$$

where $\Psi(\bar{\mathcal{M}}_i)$ denotes the compatibility constant of space $\bar{\mathcal{M}}_i$ with respect to the Frobenius norm: $\Psi(\bar{\mathcal{M}}_i, ||\cdot||_F)$.

Here, we define a key notation in the error bound:

$$
\Phi := \max_{i \in I} \lambda_i \Psi(\bar{\mathcal{M}}_i). \tag{4.3.12}
$$

Armed with this notation, Eq. (4.3.11) can be written as

$$
||\Delta||_F^2 \leq 4\Phi \sum_{i \in I} ||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F \tag{4.3.13}
$$

At this point, we directly appeal to the result in Proposition 2 of [40] with a small modification:

Proposition 4. Suppose that the structural incoherence condition **(C4)** as well as the condition **(C3)** hold. Then, we have

$$
2| \sum_{i<j} \langle\langle \Delta_i, \Delta_j \rangle\rangle| \leq \frac{1}{2} \sum_{i \in I} ||\Delta_i||_F^2. \tag{4.3.14}
$$

By this proposition, we have

$$\begin{aligned}\sum_{i\in I}||\Delta_i||_F^2 &\leq ||\Delta||_F^2 + 2|\sum_{i<j}\langle\langle\Delta_i,\Delta_j\rangle\rangle| \\ &\leq ||\Delta||_F^2 + \frac{1}{2}\sum_{i\in I}||\Delta_i||_F^2,\end{aligned} \tag{4.3.15}$$

which implies $\Sigma_{i\in I}||\Delta_i||_F^2 \leq 2||\Delta||_F^2$.

Moreover, since the projection operator is defined in terms of the Frobenius norm, it is non-expansive for all $i : ||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F \leq ||\Delta_i||_F$. Hence, we finally obtain:

$$\begin{aligned}(\sum_{i\in I}||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F)^2 &\leq (\sum_{i\in I}||\Delta_i||_F)^2 \\ &\leq |I|\sum_{i\in I}||\Delta_i||_F^2 \leq 8|I|\Phi\sum_{i\in I}||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F\end{aligned} \tag{4.3.16}$$

and therefore,

$$\sum_{i\in I}||\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)||_F \leq 8|I|\Phi \tag{4.3.17}$$

The Frobenius norm error bound Eq. (4.2.5) can be derived by plugging Eq. (4.3.17) back into Eq. (4.3.13):

$$||\Delta||_F^2 \leq 32|I|\Phi^2. \tag{4.3.18}$$

Therefore, we have

$$||\Delta||_F \leq 8\Phi \tag{4.3.19}$$

Which is exactly Eq. (4.2.5)

The proof of the final error bound Eq. (4.2.4) is straightforward from Eq. (4.3.10) and Eq. (4.3.17) as follows: for each fixed $i \in I$,

$$\begin{aligned}
&\mathcal{R}_i(\Delta_i) \\
&\leq \frac{1}{\lambda_i}\{\lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] + \lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i^{\perp}}(\Delta_i)]\} \\
&\leq \frac{1}{\lambda_i}\{\lambda_i \mathcal{R}_i[\Pi_{\bar{\mathcal{M}}_i}(\Delta_i)] + \sum_{j\in I} \lambda_j \mathcal{R}_j[\Pi_{\bar{\mathcal{M}}_j}(\Delta_j)]\} \\
&\leq \frac{2}{\lambda_i}\sum_{j\in I} \lambda_j \mathcal{R}_j[\Pi_{\bar{\mathcal{M}}_j}(\Delta_j)] \\
&\leq \frac{2}{\lambda_i}\sum_{j\in I} \lambda_j \Psi(\bar{\mathcal{M}}_j)||\Pi_{\bar{\mathcal{M}}_j}(\Delta_j)||_F \\
&\leq \frac{2\Phi}{\lambda_i}\sum_{j\in I} ||\Pi_{\bar{\mathcal{M}}_j}(\Delta_j)||_F \leq \frac{16|I|\Phi^2}{\lambda_i} = \frac{32\Phi^2}{\lambda_i}
\end{aligned} \tag{4.3.20}$$

which completes the proof. □

Proof of Theorem (6.3.1)

Proof. Since $\lambda_n > \lambda_n'$ and $\sqrt{s} > \sqrt{s_{\mathcal{G}}}$, We have that

$\lambda_n\sqrt{s} > \lambda_n'\sqrt{s_{\mathcal{G}}}$.

By Theorem (4.2.3),

$||\widehat{\Omega}_{tot} - \Omega^*_{tot}||_F \leq 8\max(\lambda_n\sqrt{s}, \lambda_n'\sqrt{s_{\mathcal{G}}}) \leq 8\sqrt{s}\lambda_n$. □

4.3.1 Useful lemma(s)

Lemma 4.3.1. *(Theorem 1 of [44]). Let δ be $\max_{ij}|[\frac{X^TX}{n}]_{ij} - \Sigma_{ij}|$. Suppose that $\nu > 2\delta$. Then, under the conditions (C-SparseΣ), and as $\rho_v(\cdot)$ is a soft-threshold function, we can deterministically guarantee that the spectral norm of error is bounded as follows:*

$$|||T_v(\widehat{\Sigma}) - \Sigma|||_\infty \leq 5\nu^{1-q}c_0(p) + 3\nu^{-q}c_0(p)\delta \tag{4.3.21}$$

Lemma 4.3.2. *(Lemma 1 of [45]). Let $\mathcal{A}$ be the event that*

$$||\frac{X^T X}{n} - \Sigma||_\infty \leq 8(\max_i \Sigma_{ii})\sqrt{\frac{10\tau \log p'}{n}} \tag{4.3.22}$$

where $p' := \max n, p$ and τ is any constant greater than 2. Suppose that the design matrix X is i.i.d. sampled from Σ-Gaussian ensemble with $n \geq 40 \max_i \Sigma_{ii}$. Then, the probability of event $\mathcal{A}$ occurring is at least $1 - 4/p'^{\tau-2}$.

Proof of Corollary (6.2.2)

Proof. In the following proof, we re-denote the following two notations: $\Sigma_{tot} := \begin{pmatrix} \Sigma^{(1)} & 0 & \cdots & 0 \\ 0 & \Sigma^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma^{(K)} \end{pmatrix}$

and

$$\Omega_{tot} := \begin{pmatrix} \Omega^{(1)} & 0 & \cdots & 0 \\ 0 & \Omega^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Omega^{(K)} \end{pmatrix}$$

The condition (C-SparseΣ) and condition (C-MinInfΣ) also hold for Ω^*_{tot} and Σ^*_{tot}. In order to utilize Theorem (6.3.1) for this specific case, we only need to show that $||\Omega^*_{tot} - [T_\nu(\widehat{\Sigma}_{tot})]^{-1}||_\infty \leq \lambda_n$ for the setting of λ_n in the statement:

$$\begin{aligned} ||\Omega^*_{tot} - [T_\nu(\widehat{\Sigma}_{tot})]^{-1}||_\infty &= ||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}(T_\nu(\widehat{\Sigma}_{tot})\Omega^*_{tot} - I)||_\infty \\ &\leq |||[T_\nu(\widehat{\Sigma}_{tot})w]|||_\infty ||T_\nu(\widehat{\Sigma}_{tot})\Omega^*_{tot} - I||_\infty \\ &= |||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty ||\Omega^*_{tot}(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot})||_\infty \\ &\leq |||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty |||\Omega^*_{tot}|||_\infty ||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty. \end{aligned} \tag{4.3.23}$$

We first compute the upper bound of $|||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty$. By the selection ν in the statement, Lemma (6.3.2) and Lemma (6.3.3) hold with probability at least $1-4/p'^{\tau-2}$. Armed with Eq. (6.3.7), we use the triangle inequality of norm and the condition (C-SparseΣ): for any w,

$$\begin{aligned}
&||T_\nu(\widehat{\Sigma}_{tot})w||_\infty = ||T_\nu(\widehat{\Sigma}_{tot})w - \Sigma w + \Sigma w||_\infty \\
&\geq ||\Sigma w||_\infty - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty \\
&\geq \kappa_2||w||_\infty - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty \\
&\geq (\kappa_2 - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty)||w||_\infty
\end{aligned} \tag{4.3.24}$$

Where the second inequality uses the condition (C-SparseΣ). Now, by Lemma (6.3.2) with the selection of ν, we have

$$|||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma|||_\infty \leq c_1(\frac{\log p'}{n_{tot}})^{(1-q)/2}c_0(p) \tag{4.3.25}$$

where c_1 is a constant related only on τ and $\max_i \Sigma_{ii}$. Specifically, it is defined as $6.5(16(\max_i \Sigma_{ii})\sqrt{10\tau})^{1-q}$. Hence, as long as $n_{tot} > (\frac{2c_1c_0(p)}{\kappa_2})^{\frac{2}{1-q}} \log p'$ as stated, so that $|||T_\nu(\widehat{\Sigma}_{tot})-\Sigma|||_\infty \leq \frac{\kappa_2}{2}$, we can conclude that $||T_\nu(\widehat{\Sigma}_{tot})w||_\infty \geq \frac{\kappa_2}{2}||w||_\infty$, which implies $|||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty \leq \frac{2}{\kappa_2}$.

The remaining term in Eq. (6.3.9) is $||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty$; $||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty \leq ||T_\nu(\widehat{\Sigma}_{tot}) - \widehat{\Sigma}_{tot}||_\infty + ||\widehat{\Sigma}_{tot} - \Sigma^*_{tot}||_\infty$. By construction of $T_\nu(\cdot)$ in (C-Thresh) and by Lemma (6.3.3), we can confirm that $||T_\nu(\widehat{\Sigma}_{tot}) - \widehat{\Sigma}_{tot}||_\infty$ as well as $||\widehat{\Sigma}_{tot} - \Sigma^*_{tot}||_\infty$ can be upper-bounded by ν.

By combining all together, we can confirm that the selection of λ_n satisfies the requirement of Theorem (6.3.1), which completes the proof. □

4.4 Related works

Superposition structured estimator (SS estimator) : The above Eq. (5.6.2) is a special case (explained in Section 4.1) of the following superposition structured estimators [27]:

$$\underset{(\theta_\alpha)_{\alpha\in I}}{\operatorname{argmin}} \mathcal{L}(\sum_{\alpha\in I}\theta_\alpha) + \sum_{\alpha\in I}\lambda_\alpha \mathcal{R}_\alpha(\theta_\alpha). \tag{4.4.1}$$

$\{\mathcal{R}_\alpha(\cdot)|\alpha\in I\}$ are a set of regularization functions and $(\lambda_\alpha)_{\alpha\in I}$ are the regularization penalties. The target parameter is $\theta = \sum_{\alpha\in I}\theta_\alpha$, a superposition of θ_α.

Elementary superposition-structured moment estimator (ESS moment estimator): Similar to Eq. (3.2.1), a recent study [39] extends the elementary estimator for sparse covariance matrices to the case of superposition-structured moments and named this extension as "Elem-Super-Moment" (ESM) estimator. [3]

$$\begin{aligned} &\underset{\theta_1,\theta_2,\ldots,\theta_{|I|}}{\operatorname{argmin}} \sum_{\alpha\in I}\lambda_\alpha \mathcal{R}_\alpha(\theta_\alpha) \\ &\text{Subject to: } \mathcal{R}^*_\alpha(\widehat{\theta} - \sum_{\alpha\in I}\theta_\alpha) \le \lambda_\alpha \qquad \forall \alpha \in I. \end{aligned} \tag{4.4.2}$$

Optimization and Computational Comparison: We use JGL-group and the model proposed by [14] (we name it as JGL-groupInf) as baselines in our experiments. As we mentioned in Section 1, the bottleneck of optimizing multi-sGGM in JGL-group is the step of SVD that needs $O(Kp^3)$ time complexity and requires storing K covariance matrix ($O(Kp^2)$ memory cost). Differently, JGL-Groupinf chose a coordinate descent method and proved that their optimization is equivalent to p sequences of quadratic subproblems, each of which costs $O(K^3p^3)$ computation. Therefore the total computational complexity of JGL-Groupinf is $O(K^3p^4)$. Besides, this coordinate descent method needs to store all K covariance matrices in the main memory($O(Kp^2)$ memory cost). Table 4.2 compares our model with two baselines in terms of time and space cost. Solving our model relies totally on entry-wise and group-entry-wise procedures. Its time complexity is $O(Kp^2)$. This is much faster than the baselines, especially in high-dimensional settings (Table 4.2). [4] Moreover, in our

[3] [39] has proved that this class of ESM estimators achieves the same convergence rate as the corresponding estimators (with the same superposition of structures) using the penalized MLE formulation under certain conditions.

[4] Note that the discussion of time complexity is for each iteration in optimization. We show the Q-linear convergence for all first-order multi-task sGGM estimators in Eq. (4.1.13). Since the baselines and our methods all use first-order optimization, we assume the number of iterations is the same among all methods.

Table 4.2: Comparison to Previous multi-sGGM methods

References	**Computational Complexity**	**Memory Cost**
JGL-Group [11]	$O(Kp^3)$	$O(Kp^2)$
JGL-GroupInf [14]	$O(K^3p^4)$	$O(Kp^2)$
FASJEM Models	$O(Kp^2)$ (if paralleling completely, $O(K)$)	$O(K)$

optimization, learning the parameters for each group $\{\Omega^{(i)}_{j,k}|i = 1, \ldots, K\}$ does not rely on other groups. This means we only need to store K entries of the same group in the memory for computing Eq. (4.1.7) and Eq. (4.1.8). The space complexity $O(K)$ is much smaller than previous methods' $O(Kp^2)$ requirement. [5] The comparisons are in Table 4.2.

Convergence Rate Analysis: Although previous joint sGGMs work well on datasets whose K and p are relatively small, two important questions remain unanswered: (1) what's the statistical convergence rate of these joint estimators? and (2) what's the benefits of joint learning? The convergence rate of estimating single-task sGGM has been well investigated [8, 23, 46, 47]. These studies proved that the estimator of single-task sGGM holds a consistent convergence rate $O(\sqrt{\frac{\log p}{n}})$ if given n data samples. However, none of the previous joint-sGGM studies provide such theoretical analysis. Experimental evaluations in previous joint-sGGM papers have shown better performance of running joint estimators over running single-task sGGM estimators on each dataset separately. However, it hasn't been proven that theoretically this joint estimation is better. We successfully answer these two remaining questions in Section 4.2.

4.5 Experiment

Multiple simulated datasets and four real-world biomedical datasets are used to evaluate FASJEM.

[5] We have provided a GPU implementation of FASJEM in Section 4.1.2. Although SVD or matrix inversion can also be speed up by GPU parallelization, these method cannot avoid the $O(Kp^2)$ memory cost, which is a huge bottleneck for large-scale problems. In Section 4.2 we prove that our estimator is completely group entry-wise and asynchronously optimizable, this makes FASJEM only require $O(K)$ memory storage.

4.5.1 Experimental Settings

Baseline: We compare (1)FASJEM-G versus JGL-group [11]; (2)FASJEM-I versus JGL-groupinf [14]. This is because the specific FASJEM estimator and its baseline share the same second-penalty function. [6] Three evaluation metrics are used for such comparisons.

- **Precision:** We use the edge-level false positive rate (FPR) and true positive rate (TPR) to measure the predicted graphs versus true graph. Repeating the process 10 times, we obtain average metrics for each method we tests. Here, FPR $= \frac{\text{FP}}{\text{FP + TN}}$ and TPR $= \frac{\text{TP}}{\text{TP + FN}}$. TP (true positive) and TN (true negative) mean the number of true nonzero entries and the number of true zero entries estimated by the predicted precision matrices. The FPR vs. TPR curve shows multi-point performance of a method over a range of the tuning parameter. The bigger the area under a FPR-TPR curve, the better a method has achieved overall.
- **Speed:** The time (log(second)) between the whole program's start and end indicates the speed of a certain method under a specific configuration of hyper-parameters. To be fair, we set up two types of comparisons. The first one fixes the number of tasks (K) but varies the dimension (p). This shows the performance of each method under a high-dimensional setting. The other type fixes the dimension (p) but varies the number of tasks (K). This measures the performance of each method when having a large number of tasks.
- **Memory:** For each method, we vary the number of tasks (K) and the dimension (p) until a specific method terminates due to the "out of memory" error. This measures the memory capacity of the corresponding method.

Our implementation: We implement FASJEM on two different architectures: (1)CPU only and (2)GPU [7]. Similar to the JGL-group from [11], we implement the CPU version FASJEM-G and FASJEM-I with R. We choose torch7 [48] (LUA based) to program FASJEM on GPU machine. [8]

[6] Since single-sGGM EE has a closed-form solution (i.e., no iterative steps are needed in optimization), we do not include it as baseline.

[7] Information of Experiment Machines: The machine that we use for experiments includes Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz with a 8GB memory. The GPU that we use for experiments is Nvidia Tesla K40c with 2880 cores and 12GB memory.

[8] Though the ideal memory requirement of FASJEM is only $O(K)$, IO costs should also be taken into account in real implementations. As being proved, Ω_{tot} is group-entry-wise optimizable. The parameter groups are independently estimated in the parallelized style. When implementing FASJEM in a single machine (our experimental setting), we prefer to choose smaller m to make full use of the main memory, where m is the number of parameter groups which are estimated at the same time.

Selection of hyper-parameters: In this experiment, we need to choose the value of three hyper-parameters. The first one v is unique for elementary-estimator based sGGM models. The second λ_n(in some models also noted as λ_1) is the main hyper-parameter we need to tune. The third ϵ equals to $\frac{\lambda'_n}{\lambda_n}$ (The notation λ_2 is normally used in related works instead of λ'_n).

- v: We pre-choose v in the set $\{0.001i|i = 1, 2, \ldots, 1000\}$ to guarantee $T_v(\Sigma_{tot})$ is invertible.
- λ_n[9]: Recent research studies from [27] and [19] conclude that the regularization parameter λ_{n_i} of a single task with n_i samples should be chosen with $\lambda_{n_i} \propto \sqrt{\frac{\log p}{n_i}}$. Combining this result and our convergence rate analysis in Section 4.2, we choose $\lambda_n = \alpha\sqrt{\frac{\log Kp}{n_{tot}}}$ where α is a hyper-parameter. The hyperparameter γ in Algorithm 1 equals to λ_n.
- ϵ: We select the best ϵ from the set $\{0.1i|i = 1, 2, \ldots, 10\}$ using cross-validation.

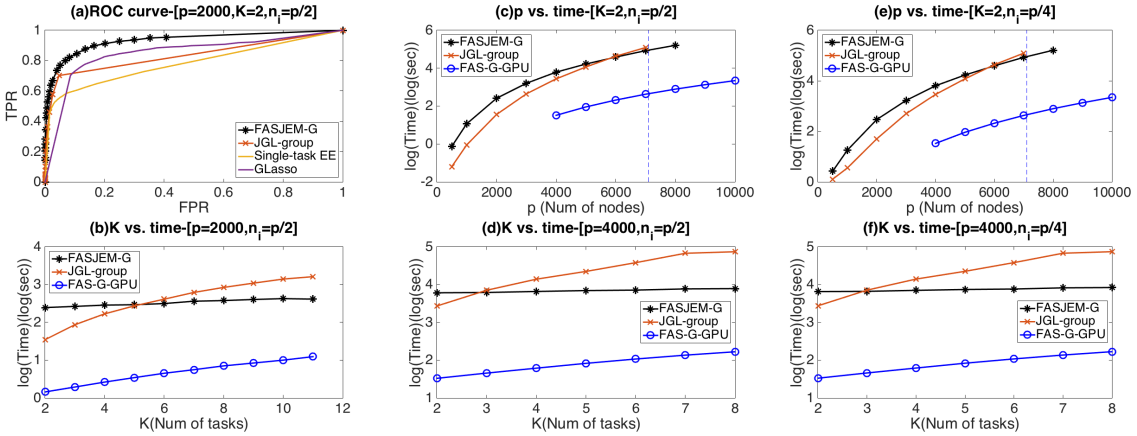


Figure 4.2: FASJEM-G versus JGL-group with respect to accuracy, speed and memory capacity. **(a):** FPR-TPR curves of two methods and two single-sGGM baselines on the simulated dataset using Random Graph Model when $p = 2000$ and $K = 2$. (AUC number–FASJEM-G:0.9332, JGL-group:0.5803, EE for sGGM:0.7852, GLasso:0.8504) **(c) and (e):** Time versus p(the number of variables) curves from FASJEM-G, JGL-group and FASJEM-G's GPU implementation. (c) uses $n_i = p/2$ and (e)$n_i = p/4$. **(b), (d) and (f):** the time versus K(the number of tasks) curves for two methods plus FASJEM-G-GPU. (b) uses $p = 2000$ and $n_i = p/2$, (d) uses $p = 4000$ and $n_i = p/2$ and (f) uses $p = 4000$ and $n_i = p/4$.

4.5.2 Experiments on simulated datasets

Using the following "Random Graph Model"(RGM), we first generate a set of synthetic multivariate Gaussian datasets, each of which includes samples of K tasks described by p variables. From [47], this "Random Graph Model" assumes $\Omega^{(i)} = B^{(i)} + \delta^{(i)}I$, where each off-diagonal entry in $B^{(i)}$ is generated independently, equals to 0.5 with probability $0.05i$ and, equals to 0 with probability $1 - 0.05i$. $\delta^{(i)}$ is selected large enough to guarantee the positive definiteness of precision matrix.

[9] $\lambda_n = 0.1$ used for time and memory experiments

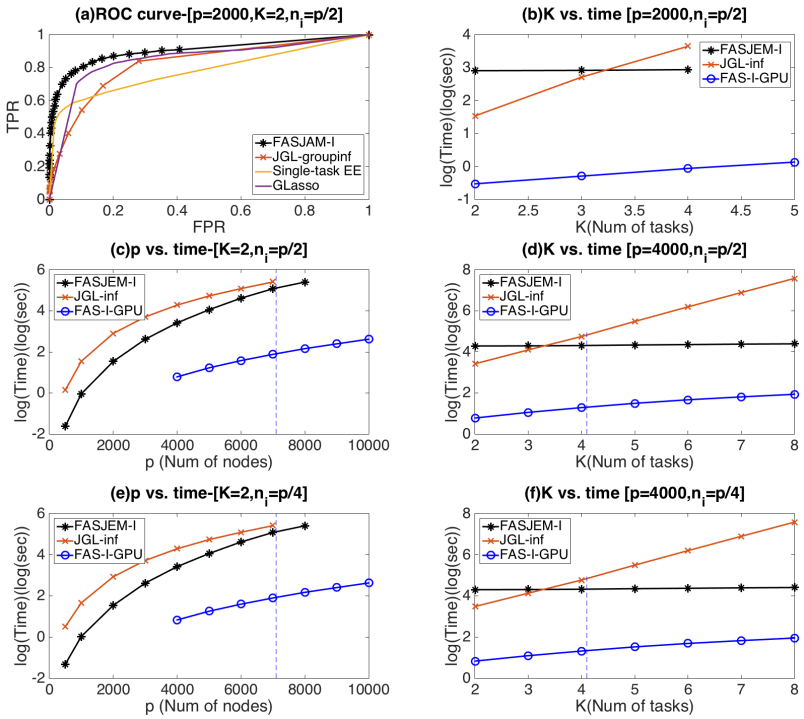


Figure 4.3: Comparison between FASJEM-I and JGL-groupinf using accuracy, speed and memory capacity. (a) FPR-TPR curves of two methods on the simulated dataset using Random Graph Model when $p = 2000$ and $K = 2$. (c) and (e) Time versus p(the number of variables) curves from FASJEM-G, JGL-group and FASJEM-I's GPU implementation. (c) uses $n_i = p/2$ and (e)$n_i = p/4$ (b), (d) and (f) include the time versus K(the number of tasks) curves for two methods plus FASJEM-I-GPU. (b) uses $p = 2000$ and $n_i = p/2$, (d) uses $p = 4000$ and $n_i = p/2$ and (f) uses $p = 4000$ and $n_i = p/4$.

For each case of p, we use this model to generate K random sparse graphs. For each graph (task), $n = p/2$ data samples are generated randomly by following $N(0, (\Omega^{(i)})^{-1})$. For each (K, p) parameter setting we test in the experiment, we use this RGM process to generate 10 different datasets (with different random seeds). Then we apply our methods and baseline methods on these datasets to obtain estimated sGGM networks. All results or curves we show in the rest of this section are average scores/curves over 10 trials for each case of parameter configuration.

Figure 4.2(a) and 4.3(a) present FPR vs. TPR curves of two proposed methods: FASJEM-G and FASJEM-I versus their corresponding baselines: JGL-group and JGL-groupinf, on the simulated datasets. We choose $p = 2000$ and $K = 2$. FPR-TPR curve plots are obtained by varying its tuning parameter λ_n over a range of $\{0.05 \times \sqrt{\frac{\log Kp}{n_{tot}}} \times i | i \in \{1, 2, 3, \ldots, 30\}\}$ and interpolating the obtained performance points (We pre-choose v and ϵ and the methods are introduced in Section 4.5.1). The two subfigures of "ROC curve" clearly show that FASJEM-G and FASJEM-I obtain better under-plot areas than corresponding JGL-group and JGL-groupinf.

Then in Figure 4.2(c)(e) and 4.3(c)(e) we show the curves of Time vs. Dimension p comparing

FASJEM-G and FASJEM-I versus their baselines. Sub-figure 4.2(c) and 4.3(c) choose $n_i = p/2$. Sub-Figures 4.2(e) and 4.3(e) use $n_i = p/4$. The CPU curves are obtained by varying p in the set of $\{1000i | i = 0.5, 1, 2, 3, \ldots, 8\}$. GPU curves are obtained by varying p in the set of $\{1000i | i = 4, 5, 6, \ldots, 10\}$. The subfigure (c) "$p$ versus time-$[K = 2,\ n_i = p/2]$" and subfigure (e) "p versus time-$[K = 2,\ n_i = p/4]$" in Figure 4.2 show that though JGL-group obtains a slightly better performance than our method under lower-dimension cases, when reaching high dimensional stages, FASJEM-G performs similarly and trains much faster than the baseline method. Figure 4.3(c) and Figure 4.3(e) provide similar conclusions for FASJEM-I vs JGL-groupinf. In addition, the baselines cannot handle $p \geq 8000$ because these approaches require too much memory. Clearly our proposed FASJEM methods can still perform reasonable well for the large-scale cases. This shows that our methods makes better usage of memory. Moreover, both FASJEM-G-GPU and FASJEM-I-GPU implementations spend only $\frac{1}{10}$ of train time against its CPU implementations. This proves that GPU parallelization can speed up FASJEM significantly.

Figure 4.2(b)(d)(f) and 4.3(b)(d)(f) show the curves about "Time vs. Number of tasks-K" comparing our methods FASJEM-G and FASJEM-I versus two baseline methods JGL-group and JGL-groupinf respectively. These sub-figures use the varying K as the x-axis over a range of $\{2, 3, \ldots, 8\}$. Sub-figures (b) use $p = 2000$, $n_i = p/2$, sub-figures (d) use $p = 4000$, $n_i = p/2$ and sub-figures (f) choose $p = 4000$, $n_i = p/4$. These figures show that the JGL-group and JGL-groupinf obtain a slightly better speed than two FASJEM, under small K cases. For larger K, our methods perform faster than the baseline methods. The conclusion hold across three cases with different pairs of (p, n_i), indicating that the advantage of our methods do not change by working on graphs and datasets of different sizes. In addition, when $p = 4000$, JGL-group and JGL-groupinf cannot handle $K \geq 5$ (i.e., the R program died) due to the memory issue on our experiment machine, while both FASJEM-G and FASJEM-I can. This proves that FASJEM requires a lower memory cost than the baselines. In all subfigures (b), (d) and (f), FASJEM curves are roughly linear. The experimental results match with the computational complexity analysis we have performed in Table 4.2 (the computation cost of FASJEM is linear to K). Moreover, subfigures (b), (d) and (f) show that both FASJEM-G-GPU and FASJEM-I-GPU implementations spend only $\frac{1}{10}$ time of their CPU implementations respectively. This confirms that GPU-parallelization can speed up FASJEM significantly.

Figure 4.4 represents a comparison between the single-task EE estimator for sGGM and GLasso estimator. We choose the $\Omega^{(i)}$ in the random graph model as the true graph. We obtain the two subfigures by varying p in a set of $\{100, 200, 300, 400, 500\}$. The left subfigure is "AUC vs. p (number

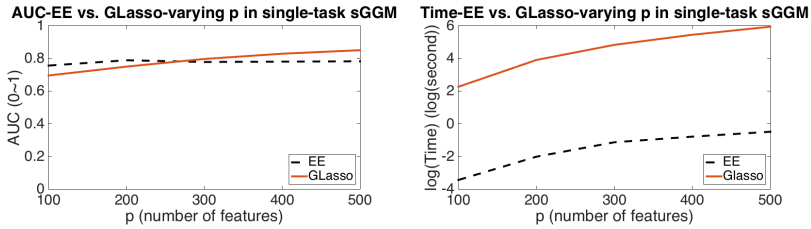


Figure 4.4: Comparison between elementary estimator for sGGM and GLasso for single-task sGGM. The left figure is the curve of AUC number by varying p. The number of sample $n = p/2$. The right figure is the curve of computation time by varying p. Other settings are the same as the left one. Clearly, elementary estimator has the similar accuracy performance as GLasso but is much faster and scalable than it.

of features)" while the right subfigure is "Time vs. p (number of features)". Figure 4.4 shows that the elementary estimator has achieved similar performance of GLasso among different p while the computation time of EE is much less than the GLasso.

Furthermore in Section 4.5.3, we compare FASJEM-G and JGL-group on four different real-world datasets. FASJEM-G consistently outperforms JGL-group on all four datasets in recovering more known edges.

4.5.3 Experiments on Real-world Datasets

We apply FASJEM-G and JGL-group on four different real-world datasets: (1) the breast/colon cancer data [49] (with 2 cell types and 104 samples, each having 22283 features); (2) Crohn's disease data [50] (with 3 cell types, 127 samples and 22283 features) , (3) the myeloma and bone lesions data set [51] (with 2 cell types, 173 samples and 12625 features) and (4) Encode project dataset [3] (with 3 cell types, 25185 samples and 27 features). For the first three datasets, we select its top 500 features based on the variance of the variables. After obtaining estimated dependency networks, we compare all methods using two major existing databases [1, 2] archiving known gene interactions. The number of known gene-gene interactions predicted by each method has been shown as bar graphs in Figure 4.5. These graphs clearly show that FASJEM-G outperforms JGL-group on all three datasets and across all cell conditions within each of the three datasets. This leads us to believe that the proposed FASJEM-G is very promising for identifying variable interactions in a wider range of applications as well.

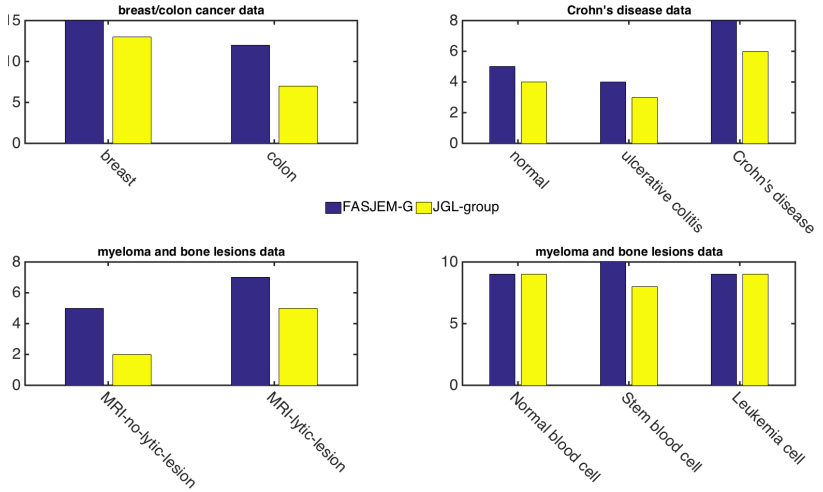


Figure 4.5: Compare predicted dependencies among genes or proteins using existing databases [1, 2] with known interactions (biologically validated) in human. The number of matches among predicted interactions and known interactions is shown as bar lines.

Chapter 5

Method II: JEEK – JEE for Adding Knowledge Explicitly

5.1 Proposed Method: JEEK

In applications of Gaussian graphical models, we typically have more information than just the data samples themselves.

5.1.1 Knowledge as Weight (KW-Norm)

The main goal of this work is to design a principled strategy to incorporate existing knowledge (other than samples or structured assumptions) into the multi-sGGM formulation. We consider two factors in such a design:

(1) When learning multiple sGGMs jointly from real-world applications, it is often of great scientific interests to model and learn context-specific graph variations explicitly, because such variations can "fingerprint" important markers in domains like cognition [4] or pathology [52]. Therefore we design to share parameters between different contexts. Mathematically, we model $\Omega^{(i)}$ as two parts:

$$\Omega^{(i)} = \Omega_I^{(i)} + \Omega_S \tag{5.1.1}$$

where $\Omega_I^{(i)}$ is the individual precision matrix for context i and Ω_S is the shared precision matrix between contexts. Again, for ease of notation we denote $\Omega_I^{tot} = (\Omega_I^{(1)}, \Omega_I^{(2)}, \ldots, \Omega_I^{(K)})$ and $\Omega_S^{tot} = (\Omega_S, \Omega_S, \ldots, \Omega_S)$.

(2) We represent additional knowledge as positive weight matrices from $\mathbb{R}^{p \times p}$. More specifically, we represent the knowledge of the task-specific graph as weight matrix $\{W^{(i)}\}$ and W_S representing existing knowledge of the shared network. The positive matrix-based representation is a powerful and flexible strategy that can describe many possible forms of existing knowledge. In Section (5.3), we provide four different designs of $\{W^{(i)}\}$ and W_S for real-world applications. In total, we have weight matrices $\{W_I^{(1)}, W_I^{(2)}, \ldots, W_I^{(K)}, W_S\}$ to represent additional knowledge. To simplify notations, we denote $W_I^{tot} = (W_I^{(1)}, W^{(2)}, \ldots, W_I^{(K)})$ and $W_S^{tot} = (W_S, W_S, \ldots, W_S)$.

Now we propose the following knowledge as weight norm (kw-norm) combining the above two:

$$\mathcal{R}(\Omega^{tot}) = ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}||_1 \tag{5.1.2}$$

Here the Hadamard product $\circ$ is the element-wise product between two matrices i.e. $[A \circ B]_{ij} = A_{ij}B_{ij}$.

The kw-norm(Eq. (5.1.2)) has the following three properties:

- (i) kw-norm is a norm function if and only if any entries in W_I^{tot} and W_S^{tot} do not equal to 0.
- (ii) If the condition in (i) holds, kw-norm is a decomposable norm.
- (iii) If the condition in (i) holds, the dual norm of kw-norm is $\mathcal{R}^*(u) = \max(||W_I^{tot} \circ u||_\infty, ||W_S^{tot} \circ u||_\infty)$.

Section 5.5.1 provides proofs of the above claims.

5.1.2 JEE with Knowledge (JEEK)

Plugging Eq. (5.1.2) to Eq. (3.1.4), we obtain the following formulation of JEEK for learning multiple related sGGMs from heterogerous samples:

$$\begin{aligned} &\underset{\Omega_I^{tot},\Omega_S^{tot}}{\operatorname{argmin}} ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}|| \\ \text{Subject to: } &||W_I^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \\ &||W_S^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \end{aligned} \tag{5.1.3}$$

In Section 4.2, we theoretically prove that the statistical convergence rate of JEEK achieves the same sharp convergence rate as the state-of-the-art estimators for multi-task sGGMs. Our proofs are inspired by the unified framework of the high-dimensional statistics [27].

5.2 Solution of JEEK:

A huge computational advantage of JEEK (Eq. (5.1.3)) is that it can be decomposed into $p \times p$ independent small linear programming problems. To simplify notations, we denote ${\Omega_I^{(i)}}_{j,k}$ (the $\{j,k\}$-th entry of $\Omega^{(i)}$) as a_i. Similarly we denote $\Omega_{S\,j,k}$ as b and $[T_v(\widehat{\Sigma}^{(i)})]_{j,k}^{-1}$ be c_i. Similarly we denote $W_{j,k}^{(i)} = w_i$ and $W_{j,k}^{S} = w_s$. "A group of entries" means a set of parameters $\{a_1, \ldots, a_K, b\}$ for certain j, k.

JEEK (Eq. (5.1.3)) can be decomposed into the following formulation for a certain j, k to estimate $\{a_1, \ldots, a_K, b\}$:

$$\begin{aligned} &\underset{a_i,b}{\operatorname{argmin}} \sum_i |w_i a_i| + K|w_s b| \\ \text{Subject to: } &|a_i + b - c_i| \leq \frac{\lambda_n}{\min(w_i, w_s)}, \\ &i = 1, \ldots, K \end{aligned} \tag{5.2.1}$$

Eq. (5.2.1) can be easily converted into a linear programming form of Eq. (5.2.2) with only $K+1$ variables. The time complexity of Eq. (5.2.1) is $O(K^4)$. Considering JEEK has a total $p(p-1)/2$ of

Algorithm 2 Joint Elementary Estimator with additional knowledge (JEEK) for Multi-task sGGMs

Input: Data sample matrix $\mathbf{X}^{(i)}$ ($i = 1$ **to** K), regularization hyperparameter λ_n, Knowledge weight matrices $\{W_I^{(i)}, W_S\}$ and **LP(.)** (a linear programming solver)

Output: $\{\Omega^{(i)}\}$ ($i = 1$ **to** K)

for $i = 1$ **to** K **do**
  Initialize $\widehat{\Sigma}^{(i)} = \frac{1}{n_i - 1} \sum_{s=1}^{n_i} (\mathbf{X}_{s,}^{(i)} - \widehat{\mu}^{(i)})(\mathbf{X}_{s,}^{(i)} - \widehat{\mu}^{(i)})^T$ (the sample covariance matrix of $\mathbf{X}^{(i)}$)
  Initialize $\Omega^{(i)} = \mathbf{0}_{p \times p}$
  Calculate the proxy backward mapping $[T_v(\widehat{\Sigma}^{(i)})]^{-1}$
end for
for $j = 1$ **to** p **do**
  for $k = 1$ **to** j **do**
    $a_i = \Omega_{I\ j,k}^{(i)}$
    $b = \Omega_{S\,j,k}$
    $[T_v(\widehat{\Sigma}^{(i)})]_{j,k}^{-1} = c_i$
    $w_i = W_{j,k}^{(i)}$
    $w_s = W_{S\,j,k}$
    $a_i, b = \mathbf{LP}(w_i, w_s, c_i, \lambda_n)$ where $i = 1, \ldots, K$ and **LP(.)** solves Eq. (5.2.1)
    for $i = 1$ **to** K **do**
      $\Omega^{(i)}{}_{j,k} = \Omega^{(i)}{}_{k,j} = a_i + b$
    end for
  end for
end for

such subproblems to solve, the computational complexity of JEEK (Eq. (5.1.3)) is therefore $O(p^2K^4)$. We summarize the algorithm of JEEK in Algorithm 2 (details in Section (5.2.2)).

5.2.1 Detailed solution

Notations: $X_{n_i \times p}^{(i)}$ is the data matrix for the i-th task, which includes n_i data samples being described by p different feature variables. Then $n_{tot} = \sum_{i=1}^{K} n_i$ is the total number of data samples. We use notation $\Omega^{(i)}$ for the precision matrices and $\widehat{\Sigma}^{(i)}$ for the estimated covariance matrices. Given a p-dimensional vector $\mathbf{x} = (x_1, x_2, \ldots, x_p)^T \in \mathbb{R}^p$, we denote the l_1-norm of x as $||\mathbf{x}||_1 = \sum_i |x_i|$. $||\mathbf{x}||_\infty = \max_i |x_i|$ is the l_∞-norm of $\mathbf{x}$. Similarly, for a matrix X, let $||X||_1 = \sum_{i,j} |X_{i,j}|$ be the ℓ_1-norm of X and $||X||_\infty = \max_{i,j} |X_{i,j}|$ be the ℓ_∞-norm of X.

In Eq. (5.2.1), let $a_i = a_i^+ - a_i^-$ and $b = b^+ - b^-$. If $a_i \geq 0$, then $a_i^+ = a_i$ and $a_i^- = 0$. If $a_i < 0$, then $a_i^+ = 0$ and $a_i^- = -a_i$. The b^+ and b^- have the similar definition. Then Eq. (5.2.1) can be

solved by the following small linear programming problem.

$$\begin{aligned}
&\underset{a_i,b}{\operatorname{argmin}} \sum_i (w_i a_i^+ + w_i a_i^-) + K w_s b^+ + K w_s b^- \\
\text{Subject to: } & a_i^+ - a_i^- + b^+ - b^- \leq c_i + \frac{\lambda_n}{\min(w_i, w_s)}, \\
& a_i^+ - a_i^- + b^+ - b^- \geq c_i - \frac{\lambda_n}{\min(w_i, w_s)}, \\
& a_i^+, a_i^-, b^+, b^- \geq 0 \\
& i = 1, \ldots, K
\end{aligned}$$

5.2.2 JEEK is Group entry-wise and parallelizing optimizable

JEEK can be easily paralleled. Essentially we just need to revise the "For loop" of step 6 and step 7 in Algorithm 2 into, for instance, "entry per machine" "entry per core". Now We prove that JEEK is group entry-wise and parallelizing optimizable. We prove that our estimator can be optimized asynchronously in a group entry-wise manner.

Theorem 5.2.1. *(**JEEK is Group entry-wise optimizable**) Suppose we use JEEK to infer multiple inverse of covariance matrices summarized as* $\widehat{\Omega}_{tot}$. $\{[\widehat{\Omega}_I^{(i)}]_{j,k}, [\widehat{\Omega}_S]_{j,k} | i = 1, \ldots, K\}$. *describes a group of* $K+1$ *entries at* (j,k) *position. Varying* $j \in \{1, 2, \ldots, p\}$ *and* $k \in \{1, 2, \ldots, p\}$, *we have a total of* $p \times p$ *groups. If these groups are independently estimated by JEEK, then we have,*

$$\bigcup_{j=1}^{p} \bigcup_{k=1}^{p} \{([\widehat{\Omega}_I^{(i)}]_{j,k} + [\widehat{\Omega}_S]_{j,k}) | i = 1, \ldots, K\} = \widehat{\Omega}_{tot}. \tag{5.2.2}$$

Proof. Eq. (5.2.1) are the small sub-linear programming problems on each group of entries. □

5.2.3 Difficulties in the JEEK's extension

We can also add more flexibility into the JEEK such as the second normalization function in JGL. It has the following formulation:

$$\begin{aligned} \underset{\Omega_I^{tot},\Omega_S^{tot}}{\operatorname{argmin}} & ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}|| + \epsilon \mathcal{R}'(\Omega^{tot}) \\ \text{Subject to: } & ||W_I^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \\ & ||W_S^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \\ & \mathcal{R}^{*\prime}(\Omega^{tot}) \leq \epsilon \lambda_n \end{aligned} \tag{5.2.3}$$

We have two ways to solve Eq. (7.2.1). The first one is the parallelized proximal algorithm. However, this algorithm requires the kw-norm has a closed-form proximity, which has not been discovered. The other way assume each ℓ_1 norm as the indepedent regularizer. However, this increases the number of proximities need to calculate to $K+1$. Moreover, $\mathcal{R}'$ is for Ω^{tot} while each ℓ_1 is for either $\Omega^{(i)_I}$ or Ω_S. Therefore, none of these solutions can keep the algorithm be fast and scalable. We choose not to introduce this work.

5.3 Design W_S and $W_I^{(i)}$: connections with related work and real-world applications

In this section, we showcase with specific examples that our proposed model JEEK can easily incorporate edge-level (like distance) as well as node-based (like hubs or groups) knowledge for the joint estimation of multiple graphs. To this end, we introduce four different choices of W_S^{tot} and W_I^{tot} in our formulation Eq. (5.1.3). By simply designing different choices of W_S^{tot} and W_I^{tot}, we can express different kinds of additional knowledge explicitly without changing the optimization algorithm.

Specifically, we design W_S and $W_I^{(i)}$ for cases like:

- (1) the additional knowledge is available in the form of a $p * p$ matrix W. For instance distance matrix among brain regions in neuroscience study belongs to this type;

- (2) the existing knowledge is not in the form of matrix about nodes. We need to design W for such cases, for example the information of known hub nodes or the information of how nodes fall into groups (e.g., genes belonging to the same pathway or locations).

For the second kind, we showcase three different designs of weight matrices for representing (a) known co-Hub nodes, (b) perturbed hub nodes, and (c) node grouping information.

The design of knowledge matrices is loosely related to the different structural assumptions used by he JGL studies as ([12], [11]). For example, JGL can use specially designed norms like the one proposed in [12] to push multiple graphs to have a similar set of nodes as hubs. However JGL can not model additional knowledge like a specific set of nodes are hub nodes (like we know node j is a hub node). Differently, JEEK can design $\{W_I^{(i)}, W_S\}$ for incorporating such knowledge. Essentially JEEK is complementary to JGL because they capture different type of prior information.

5.3.1 Case study I: Knowledge as matrix form like a distance matrix or some known edges

The first example we consider is exploiting a spatial prior to jointly estimate brain connectivity for multiple subject groups. Over time, neuroscientists have gathered considerable knowledge regarding the spatial and anatomical information underlying brain connectivity (*i.e.* short edges and certain anatomical regions are more likely to be connected [6]). Previous studies enforce these priors via a matrix of weights, W, corresponding to edges. To use our proposed model JEEK for such tasks, we can similarly choose $W = W_I^{(i)} = W_S$ in Eq. (5.1.3)).

5.3.2 Case study II: Knowledge of co-hub nodes

The structure assumption we consider is graphs with co-hub nodes. Namely, there exists a set of nodes $NId = \{j | j \in \{1, 2, \ldots, p\}\}$ such that $\Omega_{j,k}^{(i)} \neq 0, \forall i \in \{1, 2, \ldots, K\}$ and $k \in \{1, \ldots, p\}$. The above sub-figure of Figure 5.6 is an example of the co-hub nodes.

A so-called JGL-hub [12] estimator chooses $\mathcal{R}'(\cdot) = \sum_{i<i'} P_q(\Omega^{(i)} - \Omega^{(i')})$ in Eq. (5.6.2) to account for the co-hub structure assumption. Here $P_q(\Theta_1, \Theta_2, \ldots, \Theta_k) = 1/2||\Theta_1, \ldots, \Theta_k||_{\ell_1,\ell_q}$. Θ_i is a symmetric matrix and $||\cdot||_{\ell_1,\ell_q}$ is the notation of ℓ_1, ℓ_q-norm. JGL-hub formulation needs a complicated ADMM solution with computationally expensive SVD steps.

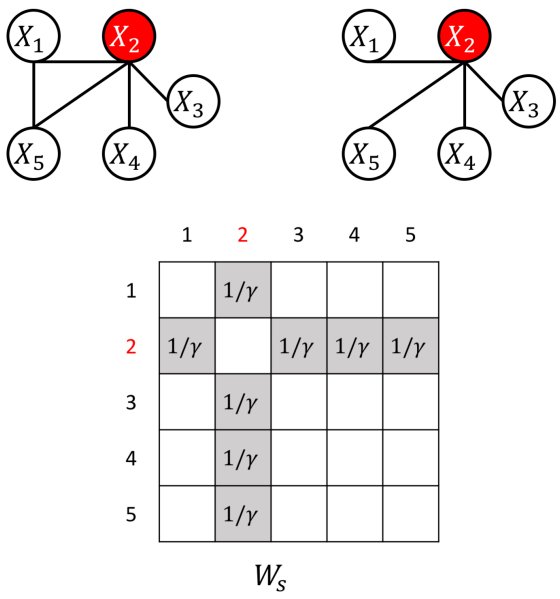


Figure 5.1: co-hub. Top: An example of the co-hub node structure. Bottom: The designed W_S for the co-hub structure case.

We design W_S and $W_I^{(i)}$ for the co-hub type knowledge in JEEK via: (1) We initialize $\{W_I^{(i)}, W_S\}$ with $\mathbf{1}_{p\times p}$; (2) $W_{S_{j,k}} = \frac{1}{\gamma}, \forall j \in NId$ and $k \in 1, \ldots, p$ where γ is a hyperparameter. Therefore, the smaller weights for the edge connecting to the node j of all the graphs enforce the co-hub structure.; (3). After this process, each entry of $\{W_I^{(i)}, W_S\}$ equals to either $\frac{1}{\gamma}$ or 1. The below sub-figure of Figure 5.6 is an example of the designed W_S.

5.3.3 Case study III: Knowledge of the perturbed hub nodes

Another structure assumption we study is graphs with perturbed nodes. Namely, there exists a set of nodes $NId = \{j | j \in \{1, 2, \ldots, p\}\}$ so that there exists i, i' $\Omega_{j,k}^{(i)} \neq 0$, and $\Omega_{j,k}^{(i')} = 0, \forall k \in \{1, \ldots, p\}$. The above sub-figure of Figure 5.7 is an example of the perturbed nodes. A so-called JGL-perturb [12] estimator chose $\mathcal{R}'(\cdot) = \sum_{i<i'} P_q((\Omega^{(1)} - \text{diag}(\Omega^{(1)})), \ldots, (\Omega^{(K)} - \text{diag}(\Omega^{(K)})))$ in Eq. (5.6.2). Here $P_q(\cdot)$ has the same definition as mentioned previously. This JGL-perturb formulation also needs a complicated ADMM solution with computationally expensive SVD steps.

To design W_S and $W_I^{(i)}$ for this type of knowledge in JEEK, we use a similar strategy as the above strategy: (1) We initialize $\{W_I^{(i)}, W_S\}$ with $\mathbf{1}_{p\times p}$; we let ${W_I^{(i)}}_{j,k} = \frac{1}{\gamma}, {W_I^{(i')}}_{j,k} = \gamma, \forall j \in NId$ and $k \in 1, \ldots, p$. Therefore, the different weights for the edge connecting to the node j in different $W_I^{(i)}$

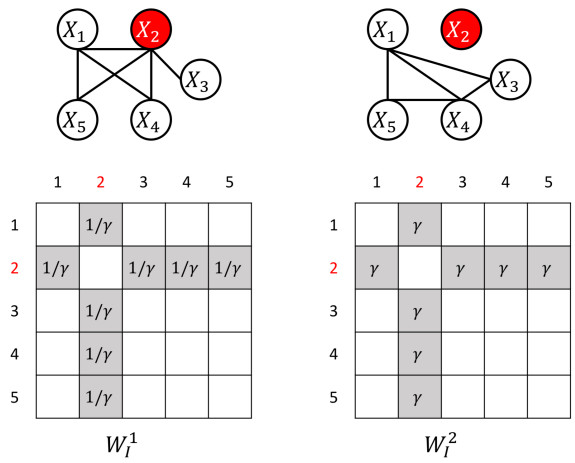


Figure 5.2: Perturb hub nodes. Top: An example of the perturbed node structure. Bottom: The designed W_I for the perturbed case.

enforce the node-perturbed structure. ; (3). After this process, each entry of $\{W_I^{(i)}, W_S\}$ equals to either $\frac{1}{\gamma}$,γ or 1. The below sub-figure of Figure 5.7 is an example of the designed $\{W_I^{(i)}\}$.

5.3.4 Case study IV: Knowledge of group information about nodes

To design W_S and $W_I^{(i)}$ for the group information about a set of nodes, we use a simple three-step strategy: (1) We initialize $\{W_I^{(i)}, W_S\}$ with $\mathbf{1}_{p\times p}$; (2) We let $W_{S_{j,k}} = \frac{1}{\gamma}, \forall (j,k) \in Id$ where γ is a hyperparameter. Therefore, the smaller weights for the edge (j,k) in all the graphs favors the edges among nodes in the same group. ; (3). After this process, each entry of $\{W_I^{(i)}, W_S\}$ equals to either $\frac{1}{\gamma}$ or 1. The below sub-figure of Figure 5.3 is an example of the designed W_S (extra knowledge is that X_2, X_3, X_4 belong to the same group).

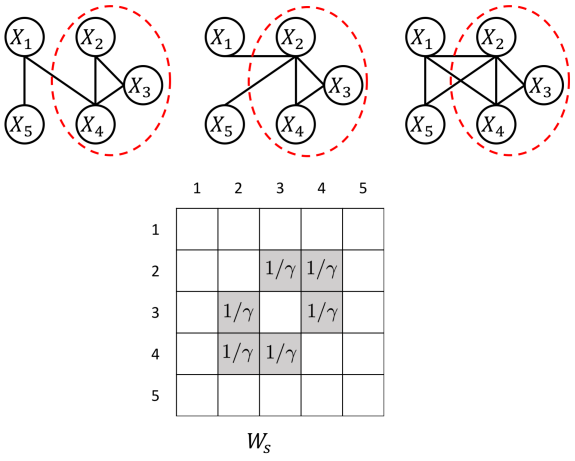
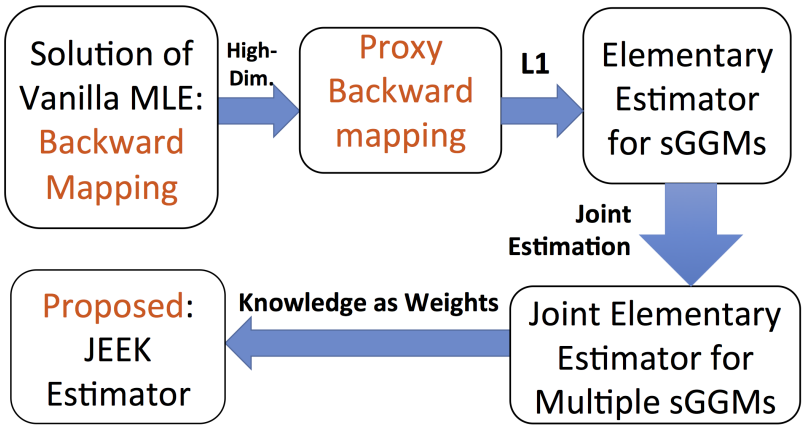


Figure 5.3: Co-group

Figure 5.4: Basic idea of JEEK.



5.4 Theoretical Analysis

KW-Norm: We presented the three properties of kw-norm in Section 5.1.1. The proofs of these three properties are included in the next Section.

Theoretical error bounds of Proxy Backward Mapping: [19] proved that when $(p \geq n)$, the proxy backward mapping $[T_v(\widehat{\Sigma})]^{-1}$ used by EE-sGGM achieves the sharp convergence rate to its truth (i.e., by proving $||T_v(\widehat{\Sigma}))^{-1} - \Sigma^{*-1}||_\infty = O(\sqrt{\frac{\log p}{n}})$). The proof was extended from the previous study [44] that devised $T_v(\widehat{\Sigma})$ for estimating covariance matrix consistently in high-dimensional situations. To derive the statistical error bound of JEEK, we need to assume that $inv(T_v(\widehat{\Sigma}^{tot}))$ are well-defined. This is ensured by assuming that the true $\Omega^{(i)^*}$ satisfy the conditions defined in Section (5.5.1).

Theoretical error bounds of JEEK: We now use the high-dimensional analysis framework from [27], three properties of kw-norm, and error bounds of backward mapping from [19, 44] to derive the statistical convergence rates of JEEK. Detailed proofs of the following theorems are in Section 5.4

.

Before providing the theorem, we need to define the structural assumption, the IS-Sparsity, we assume for the parameter truth.

(IS-Sparsity): The 'true' parameter of Ω^{tot^*} can be decomposed into two clear structures–$\Omega_I^{tot^*}$ and $\Omega_S^{tot^*}$. $\Omega_I^{tot^*}$ is exactly sparse with k_i non-zero entries indexed by a support set S_I and $\Omega_S^{tot^*}$ is exactly sparse with k_s non-zero entries indexed by a support set S_S. $S_I \bigcap S_S = \emptyset$. All other elements

equal to 0 (in $(S_I \bigcup S_S)^c$).

Theorem 5.4.1. *Consider Ω^{tot} whose true parameter Ω^{tot*} satisfies the **(IS-Sparsity)** assumption. Suppose we compute the solution of Eq. (5.1.3) with a bounded λ_n such that $\lambda_n \geq \max(||W_I^{tot} \circ (\Omega^{tot*} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty, ||W_S^{tot} \circ (\Omega^{tot*} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty)$, then the optimal solution $\widehat{\Omega}^{tot}$ satisfies the following error bounds:*

$$\begin{aligned} &||\widehat{\Omega}^{tot} - \Omega^{tot*}||_F \leq 4\sqrt{k_i + k_s}\lambda_n \\ &\max(||W_I^{tot} \circ (\widehat{\Omega}^{tot} - \Omega^{tot*})||_\infty, ||W_S^{tot} \circ (\widehat{\Omega}^{tot} - \Omega^{tot*}||_\infty) \\ &\qquad\qquad \leq 2\lambda_n \\ &||W_I^{tot} \circ (\widehat{\Omega}_I^{tot} - \Omega_I^{tot*})||_1 + ||W_S^{tot} \circ (\widehat{\Omega}_S^{tot} - \Omega_S^{tot*})||_1 \\ &\qquad\qquad \leq 8(k_i + k_s)\lambda_n \end{aligned} \tag{5.4.1}$$

Proof. See detailed proof in Section 5.5.2 □

Theorem (5.4.1) provides a general bound for any selection of λ_n. The bound of λ_n is designed by the distance between Ω^{tot*} and $inv(T_v(\widehat{\Sigma}^{tot}))$. We then use Theorem (5.4.1) to derive the statistical convergence rate of JEEK. This gives us the following corollary:

Corollary 5.4.2. *Suppose the high-dimensional setting, i.e., $p > \max(n_i)$. Let $v := a\sqrt{\frac{\log(Kp)}{n_{tot}}}$. Then for $\lambda_n := \frac{8\kappa_1 a}{\kappa_2}\sqrt{\frac{\log(Kp)}{n_{tot}}}$ and $n_{tot} > c \log Kp$, with a probability of at least $1 - 2C_1 \exp(-C_2 Kp \log(Kp))$, the estimated optimal solution $\widehat{\Omega}^{tot}$ has the following error bound:*

$$\begin{aligned} &||\widehat{\Omega}^{tot} - \Omega^{tot*}||_F \\ &\qquad \leq \frac{16\kappa_1 a \max_{j,k}(W_I^{tot}{}_{j,k}, W_S^{tot}{}_{j,k})}{\kappa_2}\sqrt{\frac{(k_i + k_s)\log(Kp)}{n_{tot}}} \end{aligned} \tag{5.4.2}$$

where a, c, κ_1 and κ_2 are constants.

Proof. See detailed proof in Section 6.3 (especially from Eq. (6.3.2) to Eq. (5.5.14)). □

5.5 Proofs

5.5.1 Theorems and Proofs of three properties of kw-norm

In this sub-section, we prove the three properties of kw-norm used in Section 5.1.1. We then provide the convergence rate of our estimator based on these three properties.

- (i) kw-norm is a norm function if and only if any entries in W_I^{tot} and W_S^{tot} do not equal to 0.
- (ii) If the condition in (i) holds, kw-norm is a decomposable norm.
- (iii) If the condition in (i) holds, the dual norm of kw-norm is $\mathcal{R}^*(u) = \max(||W_I^{tot} \circ u||_\infty, ||W_S^{tot} \circ u||_\infty)$.

Norm:

First we prove the correctness of the argument that kw-norm is a norm function by the following theorem:

Theorem 5.5.1. *Eq. (5.1.2) is a norm if and only if* $\forall 1 \geq j, k \leq p, {W_I^{(i)}}_{jk} \neq 0$, *and* $W_{Sj,k} \neq 0$.

This theorem gives the sufficient and necessary conditions to make kw-norm (Eq. (5.1.2)) a norm function.

Decomposable Norm:

Then we show that kw-norm is a decomposable norm within a certain subspace. Before providing the theorem, we give the structural assumption of the parameter.

(IS-Sparsity): The 'true' parameter for Ω^{tot*} (multiple GGM structures) can be decomposed into two clear structures–Ω_I^{tot*} and Ω_S^{tot*}. Ω_I^{tot*} is exactly sparse with k_i non-zero entries indexed by a support set S_I and Ω_S^{tot*} is exactly sparse with k_s non-zero entries indexed by a support set S_S. $S_I \bigcap S_S = \emptyset$. All other elements equal to 0 (in $(S_I \bigcup S_S)^c$).

Definition 5.5.2. *(IS-subspace)*

$$\mathcal{M}(S_I \bigcup S_S) = \{\theta_j = 0 | \forall j \notin S_I \bigcup S_S\} \tag{5.5.1}$$

Theorem 5.5.3. *Eq. (5.1.2) is a decomposable norm with respect to $\mathcal{M}$ and $\bar{\mathcal{M}}^{\perp}$*

Dual Norm of kw-Norm:

To obtain the final formulation Eq. (5.1.3) and its statistical convergence rate, we need to derive the dual norm formulation of kw-norm.

Theorem 5.5.4. *The dual norm of kw-norm (Eq. (5.1.2)) is*

$$\mathcal{R}^*(u) = \max(||W_I^{tot} \circ u||_\infty, ||W_S^{tot} \circ u||_\infty) \tag{5.5.2}$$

The details of the proof are as follows.

Proof of Theorem (5.5.1)

Lemma 5.5.5. *For kw-norm, ${W_I^{tot}}_{j,k} \neq 0$ and ${W_S^{tot}}_{j,k} \neq 0$ equals to ${W_I^{tot}}_{j,k} > 0$ and ${W_S^{tot}}_{j,k} > 0$.*

Proof. If ${W_I^{tot}}_{j,k} < 0$, then $|{W_I^{tot}}_{j,k}{\Omega_I^{tot}}_{j,k}| = |{W_I^{tot}}_{j,k}||{\Omega_I^{tot}}_{j,k}| = |-{W_I^{tot}}_{j,k}{\Omega_I^{tot}}_{j,k}|$. Notice that $-{W_I^{tot}}_{j,k} > 0$. □

Proof. To prove the kw-norm is a norm, by Lemma (6.3.2) the only thing we need to prove is that $f(x) = ||W \circ x||_1$ is a norm function if $W_{i,j} > 0$. 1. $f(ax) = ||aW \circ x||_1 = |a|||W \circ x||_1 = |a|f(x)$. 2. $f(x+y) = ||W \circ (x+y)||_1 = ||W \circ x + W \circ y||_1 \leq ||W \circ x||_1 + ||W \circ y||_1 = f(x) + f(y)$. 3. $f(x) \geq 0$ 4. If $f(x) = 0$, then $\sum |W_{i,j}x_{i,j}| = 0$. Since $W_{i,j} \neq 0$, $x_{i,j} = 0$. Therefore, $x = 0$. Based on the above, $f(x)$ is a norm function. Since summation of norm is still a norm function, kw-norm is a norm function. □

Futhurmore, we have the following Lemma:

Lemma 5.5.6. *The dual norm of $f(x)$ is $||W \circ x||_\infty$.*

Proof. $f^*(u) = \sup_x \frac{<u,x>}{||W \circ x||_1} = \sup_x \{< u, x > | ||W \circ x||_1 \leq 1\} = ||W \circ u||_\infty$. □

Proof of Theorem (5.5.3)

Proof. Assume $u \in \mathcal{M}$ and $v \in \bar{\mathcal{M}}^{\perp}$, $\mathcal{R}(u+v) = ||W_I^{tot} \circ (u_I + v_I)||_1 + ||W_S^{tot} \circ (u_S + v_S)||_1 = ||W_I^{tot} \circ u_I||_1 + ||W_S^{tot} \circ u_S||_1 + ||W_I^{tot} \circ v_I||_1 + ||W_S^{tot} \circ v_S||_1 = \mathcal{R}(u) + \mathcal{R}(v)$. Therefore, kw-norm is a decomposable norm with respect to the subspace pair $(\mathcal{M}, \bar{\mathcal{M}}^{\perp})$. □

Proof of Theorem (5.5.4)

Proof. Suppose $\mathcal{R}(\theta) = \sum_{\alpha \in I} c_\alpha \mathcal{R}_\alpha(\theta_\alpha)$, where $\sum_{\alpha \in I} \theta_\alpha = \theta$. Then the dual norm $\mathcal{R}^*(\cdot)$ can be derived by the following equation.

$$\begin{aligned}
\mathcal{R}^*(u) &= \sup_{\theta} \frac{< \theta, u >}{\theta} \\
&= \sup_{\theta_\alpha} \frac{\sum\limits_{\alpha} < u, \theta_\alpha >}{\sum\limits_{\alpha} c_\alpha \mathcal{R}_\alpha(\theta_\alpha)} \\
&= \sup_{\theta_\alpha} \frac{\sum\limits_{\alpha} < u/c_\alpha, \theta_\alpha >}{\sum\limits_{\alpha} \mathcal{R}_\alpha(\theta_\alpha)} \\
&\leq \sup_{\theta_\alpha} \frac{\sum\limits_{\alpha} \mathcal{R}^*_\alpha(u/c_\alpha) \mathcal{R}(\theta_\alpha)}{\sum\limits_{\alpha} \mathcal{R}_\alpha(\theta_\alpha)} \\
&\leq \max_{\alpha \in I} \mathcal{R}^*_\alpha(u)/c_\alpha.
\end{aligned} \tag{5.5.3}$$

Therefore by Lemma (5.5.6), the dual norm of kw-norm is $\mathcal{R}^*(u) = \max(||W_I^{tot} \circ u||_\infty, ||W_S^{tot} \circ u||_\infty)$. □

5.5.2 Proofs of Theorems about All Error Bounds of JEEK

For the proposed JEEK model, $\mathcal{R}(\Omega^{tot}) = ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}||_1$. Based on the results in [27], $\Psi(\bar{\mathcal{M}}) = \sqrt{k_i + k_s}$, where k_i and k_s are the total number of nonzero entries in Ω_I^{tot} and Ω_S^{tot}. Using $\mathcal{R}(\Omega^{tot}) = ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}||_1$ in Theorem (3.2.2), we have the following theorem (the same as Theorem (5.4.1)),

Theorem 5.5.7. *Suppose that* $\mathcal{R}(\Omega^{tot}) = ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}||_1$ *and the true parameter* Ω^{tot*} *satisfy the conditions* ***(C1)(C2)*** *and* $\lambda_n \geq \mathcal{R}^*(\widehat{\Omega}^{tot} - \Omega^{tot*})$, *then the optimal point* $\widehat{\Omega}^{tot}$ *of*

Eq. (5.1.3) has the following error bounds:

$$\begin{aligned}
&\max(||W_I^{tot} \circ (\widehat{\Omega}^{tot} - \Omega^{tot^*})||_\infty, ||W_S^{tot} \circ (\widehat{\Omega}^{tot} - \Omega^{tot^*}||_\infty) \\
&\qquad\qquad \leq 2\lambda_n \\
&||\widehat{\Omega}^{tot} - \Omega^{tot^*}||_F \leq 4\sqrt{k_i + k_s}\lambda_n \\
&||W_I^{tot} \circ (\widehat{\Omega}_I^{tot} - \Omega_I^{tot^*})||_1 + ||W_S^{tot} \circ (\widehat{\Omega}_S^{tot} - \Omega_S^{tot^*})||_1 \\
&\qquad\qquad \leq 8(k_i + k_s)\lambda_n
\end{aligned} \tag{5.5.4}$$

Proof of Theorem (3.2.2)

Proof. Let $\delta := \widehat{\theta} - \theta^*$ be the error vector that we are interested in.

$$\begin{aligned}
&\mathcal{R}^*(\widehat{\theta} - \theta^*) = \mathcal{R}^*(\widehat{\theta} - \widehat{\theta}_n + \widehat{\theta}_n - \theta^*) \\
&\leq \mathcal{R}^*(\widehat{\theta}_n - \widehat{\theta}) + \mathcal{R}^*(\widehat{\theta}_n - \theta^*) \leq 2\lambda_n
\end{aligned} \tag{5.5.5}$$

By the fact that $\theta^*_{\mathcal{M}^\perp} = 0$, and the decomposability of $\mathcal{R}$ with respect to $(\mathcal{M}, \bar{\mathcal{M}}^\perp)$

$$\begin{aligned}
&\mathcal{R}(\theta^*) \\
&= \mathcal{R}(\theta^*) + \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&= \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&\leq \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\delta) + \Pi_{\bar{\mathcal{M}}}(\delta)] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \\
&- \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&= \mathcal{R}[\theta^* + \delta] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]
\end{aligned} \tag{5.5.6}$$

Here, the inequality holds by the triangle inequality of norm. Since Eq. (3.2.1) minimizes $\mathcal{R}(\widehat{\theta})$, we have $\mathcal{R}(\theta^* + \Delta) = \mathcal{R}(\widehat{\theta}) \leq \mathcal{R}(\theta^*)$. Combining this inequality with Eq. (6.3.2), we have:

$$\mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \leq \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \tag{5.5.7}$$

Moreover, by Hlder's inequality and the decomposability of $\mathcal{R}(\cdot)$, we have:

$$\begin{aligned} &||\Delta||_2^2 = \langle \delta, \delta \rangle \leq \mathcal{R}^*(\delta)\mathcal{R}(\delta) \leq 2\lambda_n \mathcal{R}(\delta) \\ &= 2\lambda_n[\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) + \mathcal{R}(\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta))] \leq 4\lambda_n \mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 4\lambda_n \Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \end{aligned} \tag{5.5.8}$$

where $\Psi(\bar{\mathcal{M}})$ is a simple notation for $\Psi(\bar{\mathcal{M}}, ||\cdot||_2)$.

Since the projection operator is defined in terms of $||\cdot||_2$ norm, it is non-expansive: $||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \leq ||\Delta||_2$. Therefore, by Eq. (6.3.4), we have:

$$||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}), \tag{5.5.9}$$

and plugging it back to Eq. (6.3.4) yields the error bound Eq. (3.2.4).

Finally, Eq. (3.2.5) is straightforward from Eq. (6.3.3) and Eq. (6.3.5).

$$\begin{aligned} &\mathcal{R}(\delta) \leq 2\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 2\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2. \end{aligned} \tag{5.5.10}$$

□

Conditions of Proving Error Bounds of JEEK

JEEK achieves similar convergence rates as the SIMULE [36] (W-SIMULE with no additional knowledge) and FASJEM estimator [53]. The other multiple sGGMs estimation methods have not provided such convergence rate analysis.

To derive the statistical error bound of JEEK, we need to assume that $inv(T_v(\widehat{\Sigma}^{tot}))$ are well-defined. This is ensured by assuming that the true ${\Omega^{(i)}}^*$ satisfy the following conditions [19]:

(C-MinInf$-\Sigma$): The true ${\Omega^{(i)}}^*$ Eq. (5.1.3) have bounded induced operator norm, i.e., $|||{\Omega^{(i)}}^*|||_\infty := \sup_{w \neq 0 \in \mathbb{R}^p} \frac{||{\Sigma^{(i)}}^* w||_\infty}{||w||_\infty} \leq \kappa_1$.

(C-Sparse-Σ): The true covariance matrices ${\Sigma^{(i)}}^*$ are "approximately sparse" (following [43]). For some constant $0 \leq q < 1$ and $c_0(p)$, $\max_i \sum_{j=1}^{p} |[{\Sigma^{(i)}}^*]_{ij}|^q \leq c_0(p)$. [1]

We additionally require $\inf_{w \neq 0 \in \mathbb{R}^p} \frac{||{\Omega^{(i)}}^* w||_\infty}{||w||_\infty} \geq \kappa_2$.

Proof of Corollary (6.2.2)

Proof. In the following proof, we re-denote the following two notations: $\Sigma_{tot} := \begin{pmatrix} \Sigma^{(1)} & 0 & \cdots & 0 \\ 0 & \Sigma^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma^{(K)} \end{pmatrix}$

and

$$\Omega_{tot} := \begin{pmatrix} \Omega^{(1)} & 0 & \cdots & 0 \\ 0 & \Omega^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Omega^{(K)} \end{pmatrix}$$

The condition (C-SparseΣ) and condition (C-MinInfΣ) also hold for Ω^*_{tot} and Σ^*_{tot}. In order to utilize Theorem (6.3.1) for this specific case, we only need to show that $||\Omega^*_{tot} - [T_\nu(\widehat{\Sigma}_{tot})]^{-1}||_\infty \leq \lambda_n$ for the setting of λ_n in the statement:

[1] This indicates for some positive constant d, $[{\Sigma^{(i)}}^*]_{jj} \leq d$ for all diagonal entries. Moreover, if $q = 0$, then this condition reduces to ${\Sigma^{(i)}}^*$.

$$
\begin{aligned}
&||\Omega^*_{tot} - [T_\nu(\widehat{\Sigma}_{tot})]^{-1}||_\infty \\
&= ||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}(T_\nu(\widehat{\Sigma}_{tot})\Omega^*_{tot} - I)||_\infty \\
&\leq |||[T_\nu(\widehat{\Sigma}_{tot})w]|||_\infty ||T_\nu(\widehat{\Sigma}_{tot})\Omega^*_{tot} - I||_\infty \\
&= |||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty ||\Omega^*_{tot}(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot})||_\infty \\
&\leq |||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty |||\Omega^*_{tot}|||_\infty ||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty.
\end{aligned}
\tag{5.5.11}
$$

We first compute the upper bound of $|||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty$. By the selection ν in the statement, Lemma (6.3.2) and Lemma (6.3.3) hold with probability at least $1 - 4/p'^{\tau-2}$. Armed with Eq. (6.3.7), we use the triangle inequality of norm and the condition (C-SparseΣ): for any w,

$$
\begin{aligned}
&||T_\nu(\widehat{\Sigma}_{tot})w||_\infty = ||T_\nu(\widehat{\Sigma}_{tot})w - \Sigma w + \Sigma w||_\infty \\
&\geq ||\Sigma w||_\infty - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty \\
&\geq \kappa_2 ||w||_\infty - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty \\
&\geq (\kappa_2 - ||(T_\nu(\widehat{\Sigma}_{tot}) - \Sigma)w||_\infty)||w||_\infty
\end{aligned}
\tag{5.5.12}
$$

Where the second inequality uses the condition (C-SparseΣ). Now, by Lemma (6.3.2) with the selection of ν, we have

$$
|||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma|||_\infty \leq c_1 (\frac{\log(Kp')}{n_{tot}})^{(1-q)/2} c_0(p) \tag{5.5.13}
$$

where c_1 is a constant related only on τ and $\max_i \Sigma_{ii}$. Specifically, it is defined as $6.5(16(\max_i \Sigma_{ii})\sqrt{10\tau})^{1-q}$. Hence, as long as $n_{tot} > (\frac{2c_1 c_0(p)}{\kappa_2})^{\frac{2}{1-q}} \log p'$ as stated, so that $|||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma|||_\infty \leq \frac{\kappa_2}{2}$, we can conclude that $||T_\nu(\widehat{\Sigma}_{tot})w||_\infty \geq \frac{\kappa_2}{2}||w||_\infty$, which implies $|||[T_\nu(\widehat{\Sigma}_{tot})]^{-1}|||_\infty \leq \frac{2}{\kappa_2}$.

The remaining term in Eq. (6.3.9) is $||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty$; $||T_\nu(\widehat{\Sigma}_{tot}) - \Sigma^*_{tot}||_\infty \leq ||T_\nu(\widehat{\Sigma}_{tot}) - \widehat{\Sigma}_{tot}||_\infty + ||\widehat{\Sigma}_{tot} - \Sigma^*_{tot}||_\infty$. By construction of $T_\nu(\cdot)$ in (C-Thresh) and by Lemma (6.3.3), we can confirm that $||T_\nu(\widehat{\Sigma}_{tot}) - \widehat{\Sigma}_{tot}||_\infty$ as well as $||\widehat{\Sigma}_{tot} - \Sigma^*_{tot}||_\infty$ can be upper-bounded by ν.

Table 5.1: Compare JEEK versus baselines. Here T is the number of iterations.

Method	JEEK	W-SIMULE	JGL	FASJEM	NAK (run K times)
Time Complexity	$O(K^4p^2)$ ($\Rightarrow O(K^4)$ if parallelizing completely)	$O(K^4p^5)$	$O(T \times Kp^3)$	$O(T \times Kp^2)$	$O(Knp^3 + Kp^4)$
Additional Knowledge	YES	YES	NO	NO	YES

Therefore,

$$\begin{aligned} &\max(||W_I^{tot} \circ (\Omega^{tot*} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty, \\ &||W_S^{tot} \circ (\Omega^{tot*} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty) \\ &\leq O(\max_{j,k}(W_I^{tot}{}_{j,k}, W_S^{tot}{}_{j,k})\sqrt{\frac{\log(Kp)}{n_{tot}}}) \end{aligned} \tag{5.5.14}$$

By combining all together, we can confirm that the selection of λ_n satisfies the requirement of Theorem (6.3.1), which completes the proof. □

5.6 Related works

5.6.1 Connecting to Relevant Studies

JEEK is closely related to a few state-of-the-art studies summarized in Table 6.1. We compare the time complexity and functional properties of JEEK versus these studies.

NAK: [54] For the single task sGGM, one recent study [54] (following ideas from [55]) proposed to integrating Additional Knowledge (NAK)into estimation of graphical models through a weighted Neighbourhood selection formulation (NAK) as: $\widehat{\beta}^j = \underset{\beta,\beta_j=0}{\operatorname{argmin}} \frac{1}{2}||X^j - X\beta||_2^2 + ||\mathbf{r}_j \circ \beta||_1$. NAK is designed for estimating brain connectivity networks from homogeneous samples and incorporate distance knowledge as weight vectors. [2] In experiments, we compare JEEK to NAK (by running NAK R package K times) on multiple synthetic datasets of simulated samples about brain regions. The data simulation strategy was suggested by [54]. Same as the NAK [54], we use the spatial distance among brain regions as additional knowledge in JEEK.

[2] Here $\widehat{\beta}^j$ indicates the sparsity of j-th column of a single $\widehat{\Omega}$. Namely, $\widehat{\beta}^j_k = 0$ if and only if $\widehat{\Omega}_{k,j} = 0$. $\mathbf{r}_j$ is a weight vector as the additional knowledge The NAK formulation can be solved by a classic Lasso solver like glmnet.

W-SIMULE: [20] Like JEEK, one recent study [20] of multi-sGGMs (following ideas from [36]) also assumed that $\Omega^{(i)} = \Omega_I^{(i)} + \Omega_S$ and incorporated spatial distance knowledge in their convex formulation for joint discovery of heterogeneous neural connectivity graphs. This study, with name W-SIMULE (Weighted model for Shared and Individual parts of MULtiple graphs Explicitly) uses a weighted constrained ℓ_1 minimization:

$$\begin{aligned} &\underset{\Omega_I^{(i)},\Omega_S}{\operatorname{argmin}} \sum_i ||W \circ \Omega_I^{(i)}||_1 + \epsilon K||W \circ \Omega_S||_1 \\ &\text{Subject to: } ||\Sigma^{(i)}(\Omega_I^{(i)} + \Omega_S) - I||_\infty \leq \lambda_n,\ i = 1, \ldots, K \end{aligned} \tag{5.6.1}$$

W-SIMULE simply includes the additional knowledge as a weight matrix W. [3]

Different from W-SIMULE, JEEK separates the knowledge of individual context and the shared using different weight matrices. While W-SIMULE also minimizes a weighted $\ell 1$ norm, its constraint optimization term is entirely different from JEEK. The formulation difference makes the optimization of JEEK much faster and more scalable than W-SIMULE (Section (4.5)). We have provided a complete theoretical analysis of error bounds of JEEK, while W-SIMULE provided no theoretical results. Empirically, we compare JEEK with W-SIMULE R package from [20] in the experiments.

JGL: [11]: Regularized MLE based multi-sGGMs Studies mostly follow the so called joint graphical lasso (JGL) formulation as Eq. (5.6.2):

$$\begin{aligned} \underset{\Omega^{(i)} \succ 0}{\operatorname{argmin}} &\sum_{i=1}^{K}(-L(\Omega^{(i)}) + \lambda_n \sum_{i=1}^{K} ||\Omega^{(i)}||_1 \\ &+ \lambda_n' \mathcal{R}'(\Omega^{(1)}, \Omega^{(2)}, \ldots, \Omega^{(K)}) \end{aligned} \tag{5.6.2}$$

$\mathcal{R}'(\cdot)$ is the second penalty function for enforcing some structural assumption of group property among the multiple graphs. One caveat of JGL is that $\mathcal{R}'(\cdot)$ cannot model explicit additional knowledge. For instance,it can not incorporate the information of a few known hub nodes shared by the contexts. In experiments, we compare JEEK to JGL-co-hub and JGL-perturb-hub toolbox provided by [12].

[3] It can be solved by any linear programming solver and can be column-wise paralleled. However, it is very slow when $p > 200$ due to the expensive computation cost $O(K^4 p^5)$.

FASJEM: [53] One very recent study extended JGL using so-called Elementary superposition-structured moment estimator formulation as Eq. (5.6.3):

$$\begin{aligned} &\underset{\Omega_{tot}}{\operatorname{argmin}} ||\Omega_{tot}||_1 + \epsilon\mathcal{R}'(\Omega_{tot}) \\ &s.t. ||\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot}))||_\infty \leq \lambda_n \\ &\mathcal{R}'^*(\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot}))) \leq \epsilon\lambda_n \end{aligned} \tag{5.6.3}$$

FASJEM is much faster and more scalable than the JGL estimators. However like JGL estimators it can not model additional knowledge and its optimization needs to be carefully re-designed for different $\mathcal{R}'(\cdot)$. [4]

Both NAK and W-SIMULE only explored the formulation for estimating neural connectivity graphs using spatial information as additional knowledge. Differently our experiments (Section (4.5)) extend the weight-as-knowledge formulation on weights as distance, as shared hub knowledge, as perturbed hub knowledge, and as nodes' grouping information (e.g., multiple genes are known to be in the same pathway). This has largely extends the previous studies in showing the real-world adaptivity of the proposed formulation. JEEK elegantly formulates existing knowledge based on the problem at hand and avoid the need to design knowledge-specific optimization.

5.6.2 Connecting to the Bayesian statistics

Our approach has a close connection to a hierarchical Bayesian model perspective. We show that the additional knowledge weight matrices are also the parameters of the prior distribution of $\Omega_I^{(i)}, \Omega_S$. In our formulationEq. (5.1.3), $W_I^{(i)}, W_S$ are the additional knowledge weight matrices. From a hierarchical Bayesian view, the first level of the prior is a Gaussian distribution and the second level is a Laplace distribution. In the following section, we show that $W_I^{(i)}, W_S$ are also the parameters of Laplace distributions, which is a prior distribution of $\Omega_I^{(i)}, \Omega_S$.

Since by the definition, ${\Omega_I^{(i)}}_{j,k}\Omega_{S\,j,k} = 0$. There are only two possible situations:

Case I (${\Omega_I^{(i)}}_{j,k} = 0$):

$$X^{(i)}|\mu^{(i)}, \Omega^{(i)} \sim N(\mu^{(i)}, (\Omega^{(i)})^{-1}) \tag{5.6.4}$$

[4] FASJEM extends JGL into multiple independent group-entry wise optimization just like JEEK. Here $\mathcal{R}'^*(\cdot)$ is the dual norm of $\mathcal{R}'(\cdot)$. Because [53] only designs the optimization of two cases (group,2 and group,inf), we can not use it as a baseline.

$$\Omega^{(i)}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}, W_{Sj,k} = \Omega_{Sj,k}|\mu^{(i)}, W_{Sj,k} \tag{5.6.5}$$

$$\begin{aligned} &p(\Omega_{Sj,k}|\mu^{(i)}, W_{Sj,k}) \\ &\propto e^{-(W_{Sj,k}|\Omega_{Sj,k}|)} \end{aligned} \tag{5.6.6}$$

Here $\Omega_{Sj,k}|\mu^{(i)}, W_{Sj,k}$ follows a Laplace distribution with mean 0. $1/W_{Sj,k} > 0$ is the diversity parameter. The larger $W_{Sj,k}$ is, the distribution of $\Omega_{Sj,k}|\mu^{(i)}, W_{Sj,k}$ more likely concentrate on the 0. Namely, there will be the higher density for $\Omega_{Sj,k} = 0|\mu^{(i)}, W_{Sj,k}$.

Case II ($\Omega_{Sj,k} = 0$):

$$X^{(i)}|\mu^{(i)}, \Omega^{(i)} \sim N(\mu^{(i)}, (\Omega^{(i)})^{-1}) \tag{5.6.7}$$

$$\Omega^{(i)}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}, W_{Sj,k} = {\Omega_I^{(i)}}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k} \tag{5.6.8}$$

$$\begin{aligned} &p({\Omega_I^{(i)}}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}) \\ &\propto e^{-({W_I^{(i)}}_{j,k}|{\Omega_I^{(i)}}_{j,k}|)} \end{aligned} \tag{5.6.9}$$

Here ${\Omega_I^{(i)}}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}$ follows a Laplace distribution with mean 0. $1/{W_I^{(i)}}_{j,k} > 0$ is the diversity parameter. The larger ${W_I^{(i)}}_{j,k}$ is, the distribution of ${\Omega_I^{(i)}}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}$ more likely concentrate on the 0. Namely, there will be the higher density for ${\Omega_I^{(i)}}_{j,k} = 0|\mu^{(i)}, {W_I^{(i)}}_{j,k}$.

Therefore, we can combine the above two cases into the following one equation.

$$\begin{aligned} &p(\Omega^{(i)}_{j,k}|\mu^{(i)}, {W_I^{(i)}}_{j,k}, W_{Sj,k}) \\ &\propto e^{-({W_I^{(i)}}_{j,k}|{\Omega_I^{(i)}}_{j,k}| + W_{Sj,k}|\Omega_{Sj,k}|)} \end{aligned} \tag{5.6.10}$$

Our final hierarchical Bayesian formulation consists of the Eq. (5.6.4) and Eq. (5.6.10). This model

is a generalization of the model considered in the seminal paper on the Bayesian lasso [56]. The parameters ${W_I^{(i)}}_{j,k}, W_{Sj,k}$ in our general model are hyper-parameters that specify the shape of the prior distribution of each edges in $\Omega^{(i)}$. The negative log-posterior distribution of $\Omega^{(i)}$ is now given by:

$$\begin{aligned} & -\log(\mathbb{P}(\Omega^{(i)}|X^{(i)}, \mu^{(i)}, {W_I^{(i)}}_{j,k}, W_{Sj,k})) \\ & \propto -\log(det({\Omega^{(i)}}^{-1})) + < \Omega^{(i)}, \widehat{\Sigma}^{(i)} > \\ & + \sum_{j,k} ({W_I^{(i)}}_{j,k} |{\Omega_I^{(i)}}_{j,k}| + W_S |\Omega_{Sj,k}|) \end{aligned} \tag{5.6.11}$$

Eq. (5.6.11) follows a weighted variation of Eq. (5.6.2).

5.7 Experiments

We empirically evaluate JEEK and baselines on four types of datasets, including two groups of synthetic data, one real-world fMRI dataset for brain connectivity estimation and one real-world genomics dataset for estimating interaction among regulatory genes. In order to incorporating various types of knowledge, we provide five different designs of the weight matrices in Section 5.3. Details of experimental setup, metrics and hyper-parameter tuning are included in Section (5.7.1). Baselines used in our experiments have been explained in details by Section (5.6.1). We also use JEEK with no additional knowledge (JEEK-NK) as a baseline.

JEEK is available as the R package 'jeek' in CRAN.

5.7.1 Experimental Setup

On four types of datasets, we focus on empirically evaluating JEEK with regard to three aspects: (i) effectiveness, computational speed and scalability in brain connectivity simulation data; (ii) flexibility in incorporating different types of knowledge of known hub nodes in graphs; (iii) effectiveness and computational speed for brain connectivity estimation from real-world fMRI.

5.7.2 Evaluation Metrics

- AUC-score: The edge-level false positive rate (FPR) and true positive rate (TPR) are used to measure the difference between the true graphs and the predicted graphs. We obtain FPR vs. TPR curve for each method by tuning over a range of its regularization parameter. We use the area under the FPR -TPR curve (AUC-Score) to compare the predicted versus true graph. Here, $\text{FPR} = \frac{\text{FP}}{\text{FP}+\text{TN}}$ and $\text{TPR} = \frac{\text{TP}}{\text{FN}+\text{TP}}$. TP (true positive) and TN (true negative) means the number of true edges and non-edges correctly estimated by the predicted network respectively. FP (false positive) and FN (false negative) are the number of incorrectly predicted nonzero entries and zero entries respectively.

- F1-score: We first use the edge-level F1-score to compare the predicted versus true graph. Here, $\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}}$, where $\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}$ and $\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$. The better method achieves a higher F1-score.

- Time Cost: We use the execution time (measured in seconds or log(seconds)) for a method as a measure of its scalability. To ensure a fair comparison, we try 30 different λ_n (or λ_2) and measure the total time of execution for each method. The better method uses less time.[5]

Evaluations: For the first experiment on brain simulation data, we evaluate JEEK and the baseline methods on F1-score and running time cost. For the second experiment, we use AUC-score and running time cost.[6] For the third experiment, our evaluation metrics include classification accuracy, likelihood and running time cost.

- The first set of experiments evaluates the speed and scalability of our model JEEK on simulation data imitating brain connectivity. We compare both the estimation performance and computational time of JEEK with the baselines in multiple simulated datasets.

- In the second experiment, we show JEEK's ability to incorporate knowledge of known hubs in multiple graphs. We also compare the estimation performance and scalability of JEEK with the baselines in multiple simulated datasets.

[5] The machine that we use for experiments is an AMD 64-core CPU with a 256GB memory.

[6] We cannot use AUC-score for the first set of experiments as the baseline NAK only gives us the best adjacency matrix after tuning over their hyperparameters. It does not provide an option for tuning the λ_n.

- Thirdly, we evaluate the ability to import additional knowledge for enhancing graph estimation in a real world dataset. The dataset used in this experiment is a human brain fMRI dataset with two groups of subjects: autism and control. Our choice of this dataset is motivated by recent literature in neuroscience that has suggested many known weights between different regions in human brain as the additional knowledge.

5.7.3 Hyper-parameters:

We need to tune four hyper-parameters v, λ_n, λ_2 and γ:

- v is used for soft-thresholding in JEEK. We choose v from the set $\{0.001i | i = 1, 2, \ldots, 1000\}$ and pick a value that makes $T_v(\widehat{\Sigma}^{(i)})$ invertible.

- λ_n is the main hyper-parameter that controls the sparsity of the estimated network. Based on our convergence rate analysis in Section 5.4, $\lambda_n \geq C\sqrt{\frac{\log Kp}{n_{tot}}}$ where $n_{tot} = Kn$ and $n = n_i$. Accordingly, we choose λ_n from a range of $\{0.01 \times \sqrt{\frac{\log Kp}{n_{tot}}} \times i | i \in \{1, 2, 3, \ldots, 30\}\}$.

- λ_2 controls the regularization of the second penalty function in JGL-type estimators. We tune λ_2 from the set $\{0.01, 0.05, 0.1\}$ for all experiments and pick the one that gives the best results.

- γ is a hyperparameter used to design the $W_I^{(i)}, W_S$ (5.6.1). The value of γ intuitively indicates the confidence of the additional knowledge weights. In the second experiment, we choose $\gamma = \{2, 4, 10\}$.

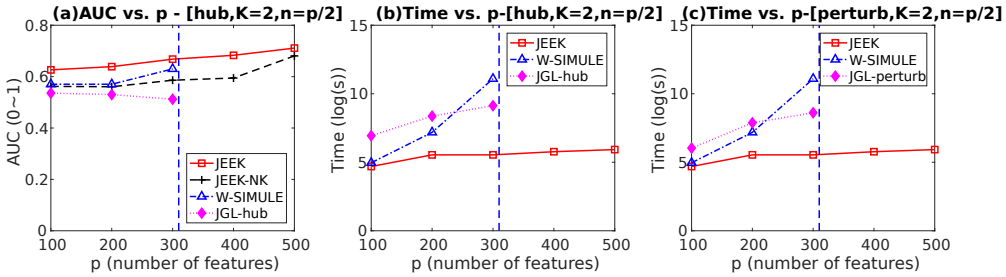


Figure 5.5: Performance comparison on simulation Datasets using co-Hub Knowledge: AUC vs. Time when varying number of nodes p.

5.7.4 Experiment I:Simulate Samples with Known Hubs as Knowledge

In this set of experiments, we show empirically JEEK's ability to model knowledge of known hub nodes across multiple sGGMs and its advantages in scalability and effectiveness. We generate multiple simulated Gaussian datasets for both the co-hub and perturbed-hub graph structures.

Simulation Protocol to generate simulated datasets: We generate multiple sets of synthetic multivariate-Gaussian datasets. First, we generate random graphs following the Random Graph Model [47]. This model assumes $\Omega^{(i)} = \mathbf{B}_I^{(i)} + \mathbf{B}_S + \delta I$, where each off-diagonal entry in $\mathbf{B}^{(i)}$ is generated independently and equals 0.5 with probability $0.1i$ and 0 with probability $1 - 0.1i$. The shared part $\mathbf{B}_S$ is generated independently and equal to 0.5 with probability 0.1 and 0 with probability 0.9. δ is selected large enough to guarantee positive definiteness. We generate cohub and perturbed structure simulations, using the following data generation models:

- **Random Graphs with cohub nodes:** After we generate the random graphs using the aforementioned Random Graph Model, we randomly generate a set of nodes $NId = \{j | j \in \{1, 2, \ldots, p\}\}$ as the cohub nodes among all the random graphs. The cardinal number of this set equals to $5\%p$. For each of these nodes j, we randomly select 90% edges $E_j = \{(j, k) | k \in \{1, 2, \ldots, p\}\}$ to be included in the graph. Then we set $\Omega_{j,k}^{(i)} = \Omega_{k,j}^{(i)} = 0.5, \forall i \in \{1, 2, \ldots, K\}$ and $(j, k) \in E_j$.

- **Random Graphs with perturbed nodes:** After we generate the random graphs using the aforementioned Random Graph Model, we randomly generate a set of nodes $NId = \{j | j \in \{1, 2, \ldots, p\}\}$ as the perturbed hub nodes for the random graphs. The cardinal number of this set equals to $5\%p$. For all graphs $\{\Omega^{(i)} | \text{i is odd}\}$, for each of these nodes $j \in NId$, we randomly select 90% edges $E_j = \{(j, k) | k \in \{1, 2, \ldots, p\}\}$ to be included in the graph. We set $\Omega_{j,k}^{(i)} = \Omega_{k,j}^{(i)} = 0.5, \forall$ odd $i \in \{1, 2, \ldots, K\}$ and $(j, k) \in E_j$. For all graphs $\{\Omega^{(i)} | \text{i is even}\}$ and nodes $j \in NId$, we randomly select 10% edges $E_j^{'} = \{(j, k) | k \in \{1, 2, \ldots, p\}\}$ to be included in the graph. We set $\Omega_{j,k}^{(i)} = \Omega_{k,j}^{(i)} = 0.5, \forall$ even $i \in \{1, 2, \ldots, K\}$ and $(j, k) \in E_j^{'}$. This creates a perturbed node structure in the multiple graphs.

Experimental baselines: We employ JGL-node for cohub and perturbed hub node structure (JGL-hub and JGL-perturb respectively) and W-SIMULE as the baselines for this set of experiments. The weights in $\{W_I^{tot}, W_S^{tot}\}$ are designed by the strategy mentioned in Section 5.3.

Experiment Results: We assess the performance of JEEK in terms of effectiveness (AUC score) and scalability (computational time cost) through baseline comparison as follows:

(a) Effectiveness: We plot the AUC-score for a number of multiple simulated datasets generated by varying the number of features p, the number of tasks K and the number of samples n. We

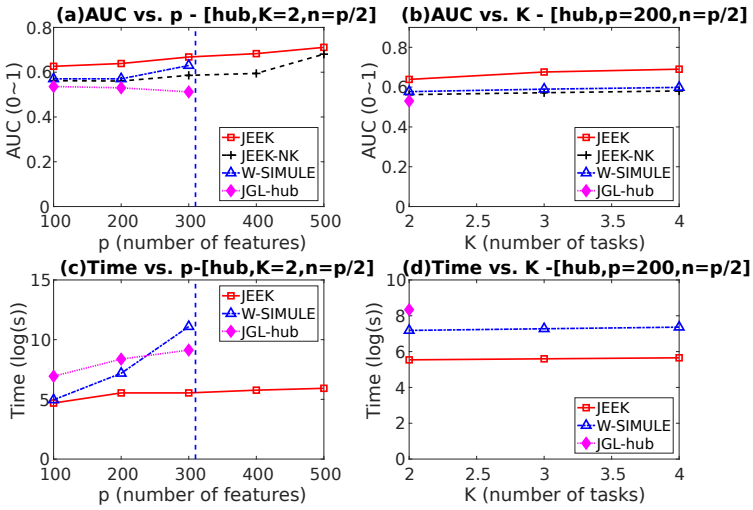


Figure 5.6: Cohub node structure: (a) AUC-score vs the number of features (p). (b) AUC-score vs the number of tasks (K). (c) Time cost (log(seconds)) vs the number of features (p). (d) Time cost (log(seconds)) vs the number of tasks (K). For $p > 300$ and $n = p/2$ W-SIMULE takes more than one month and JGL takes more than one day (indicated by dotted blue line). JGL package can only run for $K = 2$.

calculate AUC by varying λ_n. For the JGL estimator, we additionally vary λ_2 and select the best AUC (section 5.7.1). In Figure 5.6 (a) and Figure 5.6 (b), we plot the AUC-Score for the cohub node structure vs varying p and K, respectively. Figure 5.7 (a) and Figure 5.7 (b) plot the same for the perturbed node structure. In Figure 5.6 (a) and Figure 5.7 (a), we vary p in the set $\{100, 200, 300, 400, 500\}$ and set $K = 2$ and $n = p/2$. For $p > 300$ and $n = p/2$, W-SIMULE takes more than one month and JGL takes more than one day. Therefore we can not show their results for $p > 300$. For both the cohub and perturbed node structures, JEEK consistently achieves better AUC-score than the baseline methods as p is increased. For Figure 5.6(b) and Figure 5.7 (b), we vary K in the set $\{2, 3, 4\}$ and set $p = 200$ and $n = p/2$. JEEK consistently has a higher AUC-score than the baselines JGL and W-SIMULE as K is increased.

(b) Scalability: In Figure 5.6 (c) and (d), we plot the computational time cost for the cohub node structure vs the number of features p and the number of tasks K, respectively. Figure 5.7 (c) and (d) plot the same for the perturbed node structure. We interpolate the points of computation time of each estimator into curves. For each simulation case, the computation time for each estimator is the summation of a method's execution time over all values of λ_n. In Figure 5.6(c) and Figure 5.7(c), we vary p in the set $\{100, 200, 300, 400, 500\}$ and set $K = 2$ and $n = p/2$. When $p > 300$ and $n = p/2$, W-SIMULE takes more than one month and JGL takes more than one day. Hence, we have omitted their results for $p > 300$. For both the cohub and perturbed node structures, JEEK is consistently

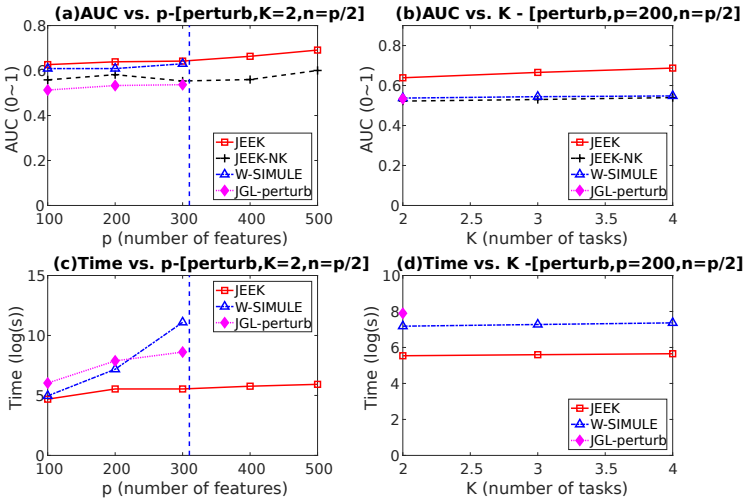


Figure 5.7: Perturbed node structure: (a) AUC-score vs the number of features (p). (b) AUC-score vs the number of tasks (K). (c) Time cost (log(seconds)) of JEEK and the baseline methods vs the number of features (p). (d) Time cost (log(seconds)) vs the number of tasks (K). for $p > 300$ and $n = p/2$, W-SIMULE takes more than one month and JGL takes more than one day (indicated by dotted blue line). JGL package can only run for $K = 2$.

more than 5 times faster as p is increased. In Figure 5.6(d) and Figure 5.7 (d), we vary K in the set $\{2, 3, 4\}$ and fix $p = 200$ and $n = p/2$. JEEK is 50 times faster than the baselines for all cases with $p = 200$ and as K is increased. In summary, JEEK is on an average more than 10 times faster than all the baselines.

(c) Stability of Results when varying W matrices: Additionally, to account for JEEK's explicit structure assumption, we also vary the ratio of known hub nodes to the total number of hub nodes. The known hub nodes are used to design the $\{W_I^i, W_S\}$ matrices(details in Section 5.6.1). In Figure 5.8(a) and (b), AUC for JEEK increases as the ratio of the number of known to total hub nodes increases. The initial increase in AUC is particularly significant as it confirms that JEEK is effective in harvesting additional knowledge for multiple sGGMs. The increase in AUC is particularly significant in the perturbed node case (Figure 5.8(b)). The AUC for the hub case does not have a correspondingly large increase with an increase in ratio because the total number of hub nodes are only 5% of the total nodes. In comparison, an increase in this ratio leads to a more significant increase in AUC because the perturbed node assumption has more information than the cohub node structure. We show in Figure 5.8(c) and (d) that the computational cost is largely unaffected by this ratio for both the cohub and perturbed node structure.

We also empirically check how the parameter r in the designed knowledge weight matrices influences the performance. In Figure 5.9(a) and (b), we show that the designed strategy for including additional

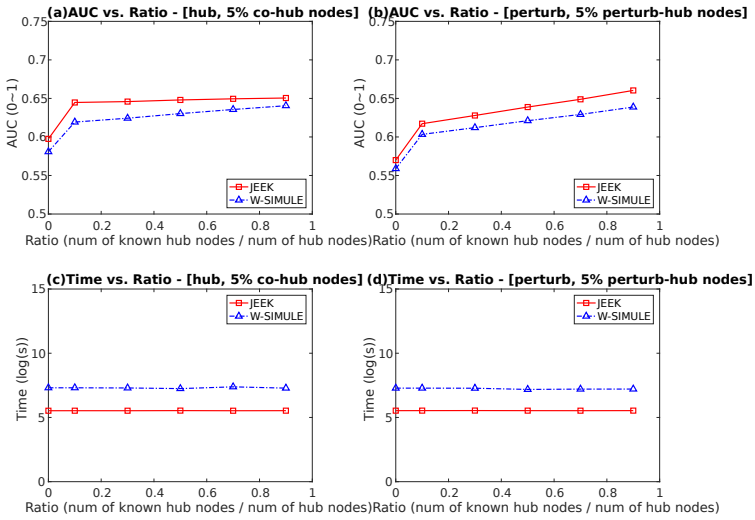


Figure 5.8: AUC-Score vs. ratio of number of known hub nodes to number of total hub nodes for (a) Cohub node structure (b) perturbed node structure. Computational Time Cost vs. ratio of number of known hub nodes to number of total hub nodes for (a) Cohub node structure (b) perturbed node structure.

knowledge as W is not affected by variations of γ. We vary γ in the set of $\{2, 4, 10\}$. In summary, the AUC-score(Figure 5.9(a),(b)) and computational time cost(Figure 5.9(c),(d)) remains relatively unaffected by the changes in γ for both co-hub and perturbed-hub case.

5.7.5 Experiment II: Gene Interaction Network from Real-World Genomics Data

Next, we apply JEEK and the baselines on one real-world biomedical data about gene expression profiles across two different cell types. We explored two different types of knowledge: (1) Known edges and (2) Known group about genes. Figure 5.10(c) shows that JEEK has lower time cost and recovers more interactions than baselines (higher number of matched edges to the existing bio-databases.).

Advancements in genome-wide monitoring have resulted in enormous amounts of data across most of the common cell contexts, like multiple common cancer types [57]. Complex diseases such as cancer are the result of multiple genetic and epigenetic factors. Thus, recent research has shifted towards the identification of multiple genes/proteins that interact directly or indirectly in contributing to certain disease(s). Structure learning of sGGMs on such heterogeneous datasets can uncover statistical dependencies among genes and understand how such dependencies vary from normal to abnormal or

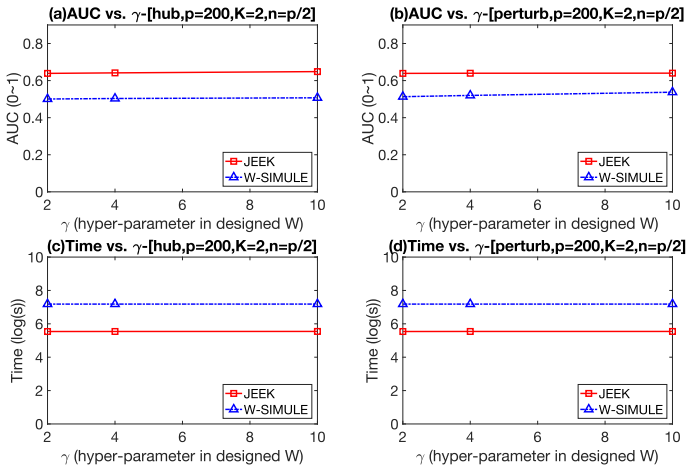


Figure 5.9: AUC-Score vs. γ (a) Cohub node structure for (b) perturbed node structure. Computational Time Cost vs. γ for (a) Cohub node structure (b) perturbed node structure.

across different diseases. These structural variations are highly likely to be contributing markers that influence or cause the diseases.

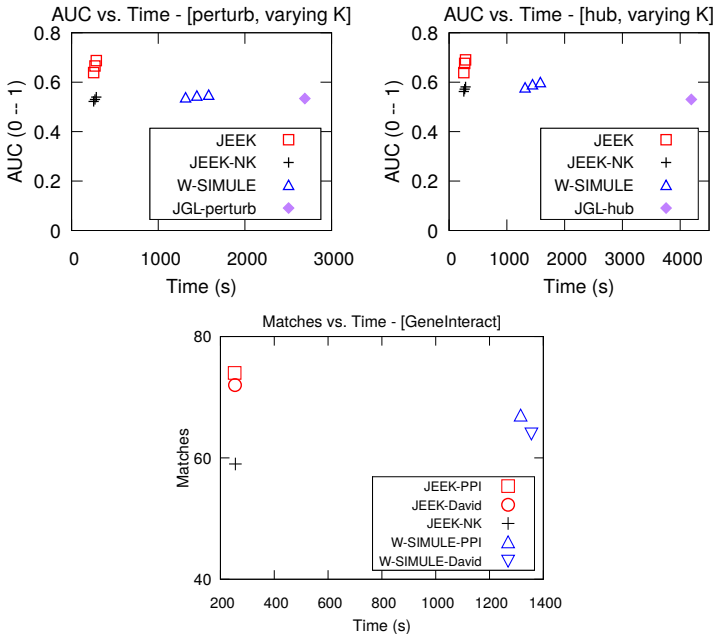
Two major cell contexts are selected from the human expression dataset provided by [58]: leukemia cells (including 895 samples and normal blood cells (including 227 samples). Then we choose the top 1000 features from the total 12,704 features (ranked by variance) and perform graph estimation on this two-task dataset. We explore two type of knowledge in the experiments.

The first kind (DAVID) is about the known group information about nodes, such as genes belonging to the same biological pathway or cellular location. We use the popular "functional enrichment" analysis tool DAVID [59] to get a set of group information about the 1000 genes. Multiple different types of groups are provided by DAVID and we pick the co-pathway. We only use the grouping information covering 20% of the nodes (randomly picked from 1000). The derived dependency graphs are compared by using the number of predicted edges being validated by three major existing protein/gene interaction databases [1,2,60] (average over both cell contexts).

The second type (PPI) is using existing known edges as the knowledge, like the known protein interaction databases for discovering gene networks (a semi-supervised setting for such estimations). We use three major existing protein/gene interaction databases [1,2,60]. We only use the known interaction edge information covering 20% of the nodes (randomly picked from 1000). The derived dependency graphs are compared by using the number of predicted edges that are not part of the known knowledge and are being validated by three major existing protein/gene interaction

databases [1, 2, 60] (average over both cell contexts).

We would like to point out that the interactions JEEK and baselines find represent statistical dependencies between genes that vary across multiple cell types. There exist many possibilities for such interactions, including like physical protein-protein interactions, regulatory gene pairs or signaling relationships. Therefore, we combine multiple existing databases for a joint validation. The numbers of matches between interactions in databases and those edges predicted by each method have been shown as the y-axis in Figure 5.10(c). It clearly shows that JEEK consistently outperforms two baselines.



(a) (b) (c)

Figure 5.10: (a)(b) Performance comparison on simulation Datasets about hubs: AUC vs. Time when varying number of tasks K. (a) is the perturbed hub cases and (b) is for the co-hub cases. (c) Performance comparison on one real-world gene expression dataset with two cell types. Two type knowledge are used to cover one fifth of the nodes, therefore each method corresponds to two performance points.

5.7.6 Experiment III: Simulated Data about Brain Connectivity with Distance as Knowledge

Following [54], we use one known Euclidean distance between human brain regions as additional knowledge W and use it to generate multiple simulated datasets. We compare JEEK with the baselines regarding (a) Scalability (computational time cost), and (b) effectiveness (F1-score, because

NAK package does not allow AUC calculation). For each simulation case, the computation time for each estimator is the summation of a method's execution time over all values of λ_n.

In this set of experiments, we confirm JEEK's ability to harvest additional knowledge using brain connectivity simulation data. Following [54], we employ the known Euclidean distance between brain regions as additional knowledge W to generate simulated datasets. To generate the simulated graphs, we use $p_{j,k} = inv.logit(10 - W_{j,k}/3)$ as the probability of an edge between nodes j and k in the graphs, where $W_{j,k}$ is the Euclidean distance between regions j and k of the brain.

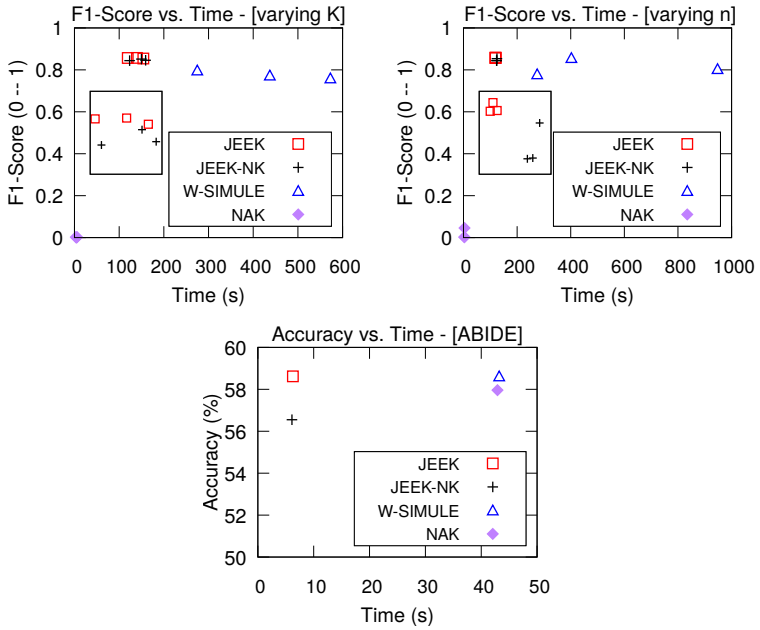
The generate datasets all have $p = 116$ corresponding to the number of brain regions in the distance matrix shared by [54]. We vary K from the set $\{2, 3, 4\}$ with $n = p/2$. The F1-scores for JEEK, JEEK-NK and W-SIMULE is the best F1-score after tuning over λ_n. The hyperparameter tuning for NAK is done by the package itself.

Simulated brain data generation model: We generate multiple sets of synthetic multivariate-Gaussian datasets. To imitate brain connectivity, we use the Euclidean distance between the brain regions as additional knowledge W where $W_{j,k}$ is the Euclidean distance between regions j and k. We fix $p = 116$ corresponding to the number of brain regions [54]. We generate the graph $\Omega^{(i)}$ following $\Omega^{(i)} = \mathbf{B}_I^{(i)} + \mathbf{B}_S + \delta I$, where each off-diagonal entry in $\mathbf{B}_I^{(i)}$ is generated independently and equals 0.5 with probability $p_{j,k} = inv.logit(10 - W_{j,k}/3)$ and 0 with probability $1 - p_{j,k}$ [54]. Similarly, the shared part $\mathbf{B}_S$ is generated independently and equal to 0.5 with probability $p_{j,k} = inv.logit(10 - W_{j,k}/3)$ and 0 with probability $1 - p_{j,k}$. δ is selected large enough to guarantee the positive definiteness. This choice ensures there are more direct connections between close regions, effectively simulating brain connectivity. For each case of simulated data generation, we generate K blocks of data samples following the distribution $N(0, (\Omega^{(i)})^{-1})$. Details see Section 5.7.1.

Experimental baselines: We choose W-SIMULE, NAK and JEEK with no additional knowledge(JEEK-NK) as the baselines. (see Section 5.6.1).

Experiment Results: We compare JEEK with the baselines regarding two aspects– (a) Scalability (Computational time cost), and (b) Effectiveness (F1-score). Figure 5.11(a) and Figure 5.11(b) respectively show the F1-score vs. computational time cost with varying number of tasks K and the number of samples n. In these experiments, $p = 116$ corresponding to the number of brain regions in

the distance matrix provided by [54]. In Figure 5.11(a), we vary K in the set $\{2, 3, 4\}$ with $n = p/2$. In Figure 5.11(b), we vary n in the set $\{p/2, p, 2p\}$ and fix $K = 2$. The F1-score plotted for JEEK, JEEK-NK and W-SIMULE is the best F1-score after tuning over λ_n. The hyperparameter tuning for NAK is done by the package itself. For each simulation case, the computation time for each estimator is the summation of a method's execution time over all values of λ_n. The points in the top left region of Figure 5.11 indicate higher F1-score and lower computational cost. Clearly, JEEK outperforms its baselines as all JEEK points are in the top left region of Figure 5.11. JEEK has a consistently higher F1-Score and is almost 6 times faster than W-SIMULE in the high dimensional case. JEEK performs better than JEEK-NK, confirming the advantage of integrating additional knowledge in graph estimation. While NAK is fast, its F1-Score is nearly 0 and hence, not useful for multi-sGGM estimation.



(a) (b) (c)

Figure 5.11: Experimental Results on Simulated Brain Datasets and on ABIDE. (a) Performance obtained on simulated brain samples with respect to F1-score vs. computational time cost when varying the number of tasks K. (b) Performance obtained on simulated brain samples with respect to F1-score vs. computational time cost when varying the number of samples n. In both (a) and (b) the smaller box shows an enlarged view comparing JEEK and JEEK-NK points. All JEEK points are in the top left region indicating higher F1-score and lower computational cost. (c). On ABIDE, JEEK outperforms the baseline methods in both classification accuracy and running time cost. JEEK and JEEK-NK points in the top left region and JEEK points are higher in terms of y-axis positions.

5.7.7 Experiment IV: Functional Connectivity Estimation from Real-World Brain fMRI Data

We evaluate JEEK and relevant baselines for a classification task on one real-world publicly available resting-state fMRI dataset: ABIDE [61]. The ABIDE data aims to understand human brain connectivity and how it reflects neural disorders [62]. ABIDE includes two groups of human subjects: autism and control, and therefore we formulate it as $K = 2$ graph estimation. We utilize the spatial distance between human brain regions as additional knowledge for estimating functional connectivity edges among brain regions. We use Linear Discriminant Analysis (LDA) for a downstream classification task aiming to assess the ability of a graph estimator to learn the differential patterns of the connectome structures.

Figure 5.11(c) compares JEEK and three baselines: JEEK-NK, W-SIMULE and W-SIMULE with no additional knowledge (W-SIMULE-NK). JEEK yields a classification accuracy of 58.62% for distinguishing the autism subjects versus the control subjects, clearly outperforming JEEK-NK and W-SIMULE-NK. JEEK is roughly 7 times faster than the W-SIMULE estimators, locating at the top left region in Figure 5.11(c) (higher classification accuracy and lower time cost). We also experimented with variations of the W matrix and found the classification results are fairly robust to the variations of W.

Experimental Baselines: We choose W-SIMULE as the the baseline in this experiment. We also compare JEEK to JEEK-NK and W-SIMULE-NK to demonstrate the need for additional knowledge in graph estimation.

ABIDE Dataset: This data is from the Autism Brain Imaging Data Exchange (ABIDE) [61], a publicly available resting-state fMRI dataset. The ABIDE data aims to understand human brain connectivity and how it reflects neural disorders [62]. The data is retrieved from the Preprocessed Connectomes Project [63], where preprocessing is performed using the Configurable Pipeline for the Analysis of Connectomes (CPAC) [64] without global signal correction or band-pass filtering. After preprocessing with this pipeline, 871 individuals remain (468 diagnosed with autism). Signals for the 160 (number of features $p = 160$) regions of interest (ROIs) in the often-used Dosenbach Atlas [65] are examined.

Distance as Additional Knowledge: To select the weights $\{W_I^{(i)}, W_S\}$, two separate spatial distance matrices W were derived from the Dosenbach atlas. The first, referred to as *anatomical*^{i}, gives each ROI one of 40 well-known, anatomic labels (*e.g.* "basal ganglia", "thalamus"). Weights $W_{j,k}$ take the low value i if two ROIs have the same label, and the high value $10 - i$ otherwise. The second additional knowledge matrix, referred to as *dist*^{i}, sets the weight of each edge ($W_{j,k}$) to its spatial length, in MNI space[7], raised to the power i. Then $W_I^{(i)} = W_S = W$.

Cross-validation: Classification is performed using the 3-fold cross-validation suggested by the literature [66] [67]. The subjects are randomly partitioned into three equal sets: a training set, a validation set, and a test set. Each estimator produces $\widehat{\Omega}^{(1)} - \widehat{\Omega}^{(2)}$ using the training set. Then, these differential networks are used as inputs to linear discriminant analysis (LDA), which is tuned via cross-validation on the validation set. Finally, accuracy is calculated by running LDA on the test set. This classification process aims to assess the ability of an estimator to learn the differential patterns of the connectome structures. We cannot use NAK to perform classification for this task, as NAK outputs only an adjacency matrix, which cannot be used for estimation using LDA.

Parameter variation: The results are fairly robust to variations of the W. (see Table 5.2). The effect of changing W seems to have a fairly small effect on the log-likelihood of the model. This is likely because both penalize picking physically long edges, which agrees with observations from neuroscience. The *dist* W effectively encourages the selection of short edges, and the *anatomical* W also has substantial spatial localization.

Table 5.2: Variations of the W and multi-task component yield fairly stable results.

Prior	Sparsity=8%		Sparsity=16%	
	Log-Likelihood	Test Accuracy	Log-Likelihood	Test Accuracy
No Additional Knowledge	-294.34	0.56	-283.27	0.55
dist	-289.12	0.53	-285.69	0.55
dist^2	-283.78	0.54	-282.92	0.54
anatomical^1	-292.42	0.56	-289.34	0.57
anatomical^2	-291.29	0.58	-285.63	0.56

[7] MNI space is a coordinate system used to refer to analagous points on different brains.

Chapter 6

Method III: DIFFEE – JEE for Learning Sparse Structural Change Directly

In last line of the works, we propose a simple estimator, namely DIFFerential networks via an Elementary Estimator (DIFFEE) for fast and scalable learning of sparse structural change in high-dimensional GGMs.

6.1 Proposed Method: DIFFEE

The aforementioned studies cannot avoid certain steps involving expensive computation in their iterative optimization, such as SVD operations in the FusedGLasso, linear programming in the Diff-CLIME, and calculating the normalization term in the Density-Ratio estimator. We aim to propose a scalable and theoretically-guaranteed estimator for estimating sparse differential network under large-scale settings.

Differential Network by Elementary Estimators (DIFFEE): Computationally elementary estimators are much faster than their regularized convex program peers for graphical model estimation.

Therefore we extend it to the following general estimator for estimating sparse change in GGM structure:

$$\begin{aligned} &\underset{\Delta}{\operatorname{argmin}} \, ||\Delta||_1 \\ &\text{Subject to: } ||\Delta - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty \leq \lambda_n \end{aligned} \tag{6.1.1}$$

The basic idea in Eq. (6.1.1) is to use a well-defined proxy function $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$ to approximate the backward mapping (the vanilla graphical model MLE solution), so that $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$ is both well-defined under high-dimensional situations and also has a simple closed-form.

As shown by Figure 2.1, there are three components in the estimation pipeline of elementary estimator for GM: (1) Backward mapping that is the vanilla MLE solution for estimating an exponential graphical model; (2) Proxy backward mapping $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c$ for dimensional settings; and (3) The closed-form solution of Eq. (6.1.1) as the final estimator.

(1) Backward Mapping: The density ratio of two Gaussian distributions is naturally an exponential-family distribution (see Section 2.3.3). Based on [37], learning an exponential family distribution from data means to estimate its canonical parameter. For an exponential family distribution, computing the canonical parameter through vanilla graphical model MLE can be expressed as a backward mapping (the first step in Figure 2.1). Through simple derivations in Eq. (2.3.8), we can easily conclude that the differential network Δ is one entry of the canonical parameter for this distribution. When using vanilla MLE to learn this exponential distribution (i.e., estimating canonical parameter), the backward mapping of Δ can be easily inferred from the two sample covariance matrices using $(\widehat{\Sigma}_d^{-1} - \widehat{\Sigma}_c^{-1})(Section\ 2.3)$.

(2) Proxy Backward Mapping: Now the key is to find a closed-form and statistical guaranteed estimator as proxy backward mapping of Δ under high-dimensional cases. Inspired by the elementary estimator for sGGM, we choose $[T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1})$ as the proxy backward mapping for Δ. Here

$$[T_v(A)]_{ij} := \rho_v(A_{ij}) \tag{6.1.2}$$

where $\rho_v(\cdot)$ is chosen to be a soft-thresholding function. We therefore obtain the following DIFFEE objective function for estimating sparse changes in GGM structure:

$$\begin{aligned} &\underset{\Delta}{\operatorname{argmin}} \, ||\Delta||_1 \\ &\text{Subject to: } ||\Delta - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||_\infty \leq \lambda_n \end{aligned} \tag{6.1.3}$$

Here $\lambda_n > 0$ is the tuning parameter.

The optimization in Eq. (6.1.3) seeks an estimator with minimum complexity with regard to the ℓ_1 regularization, at the same time being close enough to the 'initial estimator' $[T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$ according to the element-wise ℓ_∞ norm. This formulation ensures that the final estimator (solution of Eq. (6.1.3)) has the desired sparse structure.

Theoretically, the choice of ℓ_1 and ℓ_∞ in Eq. (6.1.1) connects to the asymptotic error bounds of the final estimators. In Section 6.2, we theoretically prove that the statistical convergence rate of DIFFEE achieves the same sharp convergence rate as the state-of-the-art estimators for differential network. Our proofs are inspired by the unified framework of the high-dimensional statistics [27] and EE for sGGM [19].

[19] proved that when (p¿n), the proxy backward mapping $[T_v(\widehat{\Sigma})]^{-1}$ in their EE-sGGM achieves the sharp convergence rate to its truth (i.e., by proving $||T_v(\widehat{\Sigma}))^{-1} - \Sigma^{*-1}||_\infty = O(\sqrt{\frac{\log p}{n}})$). The proof was extended from the previous study [44] who devised $T_v(\widehat{\Sigma})$ for estimating covariance matrix consistently under high-dimensional cases. We use the convergence results from [44] and [19] in Section 6.2 for deriving the statistical convergence rates of DIFFEE (details in Section 6.3).

(3) Closed Form Solution: To solve Eq. (6.1.3), we get the following closed form solution:

$$\widehat{\Delta} = S_{\lambda_n}([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}) \tag{6.1.4}$$

Where $[T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$ is the pre-computed proxy backward mapping. Here $[S_\lambda(A)]_{ij} = \text{sign}(A_{ij}) \max(|A_{ij}| - \lambda, 0)$ is the soft-thresholding function. Algorithm 3 shows the detailed steps of the DIFFEE estimator. Being non-iterative, the closed form solution helps DIFFEE achieve significant computational advantages over other estimators.

Algorithm 3 DIFFEE

input Two data matrices $\mathbf{X}_c$ and $\mathbf{X}_d$.
input Hyper-parameter: λ_n and v
output Δ
1: Compute $[T_v(\widehat{\Sigma}_c)]^{-1}$ and $[T_v(\widehat{\Sigma}_d)]^{-1}$ from $\widehat{\Sigma}_c$ and $\widehat{\Sigma}_d$.
2: Compute $\Delta = S_{\lambda_n}([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1})$
output Δ

6.1.1 Analysis of Computational Complexity

The closed form solution (Eq. (6.1.4)) brings significant advantages in hyper-parameter tuning. This is because we only need to compute the proxy backward mapping $[T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$ once. Then the model selection just executes a fast and simple element-wise soft-thresholding operator using different values of hyper-parameter λ_n (Eq. (6.1.4)).

In details, DIFFEE includes four non-iterative operations in its computation:

1. Estimating two covariance matrices. The computational complexity is $O(\max(n_c, n_d)p^2)$.
2. The element-wise soft-thresholding operations $[T_v(\cdot)]$, that cost $O(p^2)$.
3. The matrix inversions [1] $[T_v(\cdot)]^{-1}$ to get the proxy backward mapping, that cost $O(p^3)$.
4. The element-wise soft-thresholding operation S_{λ_n} that costs $O(p^2)$.

Therefore, the total asymptotic computational complexity of DIFFEE estimator is $O(p^3)$.

In Table 6.1, we compare the asymptotic computational complexity of our method to the baselines. DIFFEE achieves the best computational complexity compared to the state-of-the-art baselines. This is because:

- All existing estimators for differential network estimation have used an iterative optimization procedure to find the solution. In each iteration, their estimations require at least $O(p^3)$ computational cost.
- For tuning the sparsity hyperparameter λ_n, DIFFEE only needs to re-run its element-wise soft-thresholding operation S_{λ_n} that cost $O(p^2)$. In contrast, all the baselines have to re-run the whole algorithm for each value of the hyper-parameter λ_n.

[1] Many faster algorithms exist for speeding up matrix inversion and matrix multiplication. The best known asymptotic cost of matrix inversion is $O(p^{2.373})$ (Wikipedia). Besides both operations can be further improved up by parralelization

Table 6.1: Compare the asymptotic time complexity. DIFFEE is the best among all the estimators. Here T is the number of iterations.

DIFFEE	FusedGLasso	Density Ratio	Diff-CLIME
$O(p^3)$	$O(T * p^3)$	$O((n_c + p^2)^3)$	$O(p^8)$

- Most estimators have two hyperparameters for tuning. FusedGlasso (Eq. (6.4.1)) and DensityRatio (Eq. (6.4.3)) both need to tune the hyperparameter λ_2 [2]. Both tuning are much more expensive than DIFFEE in computation. DIFFEE needs to tune the hyperparamter v, but it costs only $O(p^2)$.
- Diff-CLIME has one hyperparameter λ_n for tuning, however, its asymptotic time cost ($O(p^8)$) is significantly more demanding than DIFFEE [3]. In summary, Diff-CLIME can not handle large-scale cases, like $p > 100$. For example, in our experiments Diff-CLIME can not even finish on a case of $p = 200$ after two days of running.

6.1.2 DIFFEE-K: DIFFEE with additional knowledge

Similar to JEEK, We can also extend our model DIFFEE by adding additional knowledge into the model.

Incorporating knowledge with a new norm function: kEV norm

The main goal of this paper is to design a principled strategy to incorporate existing knowledge (other than samples or structured assumptions) into the differential network estimation formulation. We consider two alternative choices in such a design:

(1) Knowledge as Weight Matrix: We represent additional knowledge as positive weight matrices from $\mathbb{R}^{p\times p}$. More specifically, we represent the edge-level knowledge of the differential graph as weight matrix W_E. We use the weights to increase or decrease the sparsity penalty on certain edges. The larger a weight, the more likely its corresponding edge to be sparse.

[2] The optimization problem of DensityRatio is a quadratic programming problem with $n_c + p^2$ variables. Based on the result from [68], the computational complexity of quadratic problem with b variables is $O(b^3)$. Therefore, the time complexity of DensityRatio is $O((n_c + p^2)^3)$.

[3] The optimization problem of Diff-CLIME is a linear programming problem with p^2 variables. Based on the result from [69], the computational complexity of linear problem with b variables is $O(b^4)$. Therefore, the time complexity of Diff-CLIME is $O((p^2)^4)$.

The positive matrix-based representation is a powerful and flexible strategy that can describe many possible forms of existing knowledge. For instance, it can describe spatial or anatomy knowledge about brain regions. Over time, neuroscientists have gathered considerable knowledge regarding the spatial and anatomical information underlying brain connectivity (*i.e.* short edges and certain anatomical regions are more likely to be connected [70]). Such a weight matrix W_E can also include existing known edges as the knowledge, like the known protein interaction databases for discovering gene networks (a semi-supervised setting for such estimations).

(2) Knowledge as Node Groups: In many real-world applications, there exists known group knowledge about feature variables. For example, when working with genomics data "functional enrichment" analysis [59] can provide a rich set of group information about genes belonging to the same biological pathway or cellular locations. Genes in the same biology pathway tend to have interactions among them in one cellular context or tend to not interact with each other (shared sparsity) at some other cellular condition.

This type of node group information can not be represented through the weight matrix because even though it is safe to assume nodes in the same group share similar interaction pattern, but we do not know if we should enforce more sparsity or enforce less sparsity on the whole group of nodes. Therefore this work uses a node-level group norm to describe such extra knowledge. We represent the group knowledge as a set of feature variable (vertex in the graph) groups $\mathcal{G}_p$. $\forall g_k \in \mathcal{G}_p$, $g_k = \{i\}$ where i indicates the i-th node in the graph. Moreover, we can generate an edge (matrix index) set group $\mathcal{G}_V$ from $\mathcal{G}_p$. Concretely, $\mathcal{G}_V = \{g'_k | (i,j) \in g'_k, i,j \in g_k\}$. We also denote E is the whole edge set.

Now we propose the following knowledge for Edges and Vertex norm (kEV-norm) combining the above two choices:

$$\mathcal{R}(\Delta) = ||W_E \circ \Delta_{E \setminus \mathcal{G}_V}||_1 + \epsilon ||\Delta_{\mathcal{G}_V}||_{\mathcal{G}_V,2} \tag{6.1.5}$$

Here $\Delta = \Delta_{\mathcal{G}_V \bigcup E \setminus \mathcal{G}_V}$. The Hadamard product $\circ$ is the element-wise product between two matrices i.e. $[A \circ B]_{ij} = A_{ij} B_{ij}$ and $||\cdot||_{\mathcal{G}_V,2} = \sum_k ||\Delta_{g'_k}||_2$.

Notice that when $\epsilon > 0$, we can simplify the Eq. (6.1.5) into $\mathcal{R}(\Delta) = ||\frac{W_E}{\epsilon} \circ \Delta_{E \setminus \mathcal{G}_V}||_1 + ||\Delta_{\mathcal{G}_V}||_{\mathcal{G}_V,2}$ and we let the $\frac{W_E}{\epsilon}$ as the new W_E. If $\epsilon = 0$, $\mathcal{R}(\Delta)$ reduces to a weighted ℓ_1 norm. Therefore, in the experiments we set ϵ either 1 or 0.

The kEV-norm(Eq. (6.1.5)) has the following three properties:

- (i) kEV-norm is a norm function.
- (ii) kEV-norm is a decomposable norm.
- (iii) The dual norm of kEV-norm is $\mathcal{R}^*(u) = \max(||W_E \circ u||_\infty, \epsilon||u||^*_{\mathcal{G}_V,2})$.

Where $||u||^*_{\mathcal{G},2} := \max_{g\in\mathcal{G}} ||u_g||_2$.

DIFFEE with Knowledge (DIFFEE-K)

Plugging Eq. (6.1.5) to Eq. (3.2.1), we obtain the following formulation of DIFFEE-K for learning sparse changes between two GGMs:

$$\begin{aligned} &\underset{\Delta}{\operatorname{argmin}} ||W_E \circ \Delta_{E\setminus\mathcal{G}_V}||_1 + \epsilon||\Delta_{\mathcal{G}_V}||_{\mathcal{G}_V,2} \\ \text{Subject to: } &||W_E \circ \left(\Delta - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)\right)||_\infty \leq \lambda_n \\ &\epsilon||\Delta - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||^*_{\mathcal{G}_V,2} \leq \lambda_n \end{aligned} \tag{6.1.6}$$

6.2 Strong Statistical Guarantees of DIFFEE

In this section, we provide a statistical convergence analysis of DIFFEE Eq. (6.1.1) under the following structural assumption:

(C-Sparsity): The 'true' canonical exponential family parameter for Δ^* (sparse change between two GGM structures) is exactly sparse with k non-zero entries indexed by a supported set S. All other elements equal to 0 (in S^c).

Theorem 6.2.1. *Consider any differential network in Eq. (1.0.3) whose sparse canonical parameter Δ^* satisfies the **(C-Sparsity)** assumption. Suppose we compute the solution of Eq. (6.1.1) with a bounded λ_n such that $\lambda_n \geq ||\Delta^* - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty$, then the optimal solution $\widehat{\Delta}$ satisfies the following error bounds:*

$$
\begin{aligned}
||\widehat{\Delta} - \Delta^*||_\infty &\leq 2\lambda_n \\
||\widehat{\Delta} - \Delta^*||_F &\leq 4\sqrt{k}\lambda_n \\
||\widehat{\Delta} - \Delta^*||_1 &\leq 8k\lambda_n
\end{aligned} \tag{6.2.1}
$$

Proof. See detailed proof in Section 6.3 □

Theorem (6.2.1) provides a general bound for any selection of λ_n and $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$. We then use Theorem (6.2.1) to derive the statistical convergence rate of DIFFEE whose choice of the proxy backward mapping is $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c) = [T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$. This gives us the following corollary:

Corollary 6.2.2. *Suppose the high-dimensional setting, i.e., $p > \max(n_c, n_d)$. Let $v := a\sqrt{\frac{\log p}{\min(n_c,n_d)}}$. Then for $\lambda_n := \frac{8\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p}{\min(n_c,n_d)}}$ and $\min(n_c, n_d) > c \log p$, with a probability of at least $1 - 2C_1 \exp(-C_2 K p \log(Kp))$, the estimated optimal solution $\widehat{\Delta}$ has the following error bound:*

$$
\begin{aligned}
||\widehat{\Delta} - \Delta^*||_\infty &\leq \frac{16\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p}{\min(n_c, n_d)}} \\
||\widehat{\Delta} - \Delta^*||_F &\leq \frac{32\kappa_1 a}{\kappa_2}\sqrt{\frac{k \log p}{\min(n_c, n_d)}} \\
||\widehat{\Delta} - \Delta^*||_1 &\leq \frac{64\kappa_1 a}{\kappa_2} k\sqrt{\frac{\log p}{\min(n_c, n_d)}}
\end{aligned} \tag{6.2.2}
$$

where a, c, κ_1 and κ_2 are constants.

Proof. See detailed proof in Section 6.3 (especially from Eq. (6.3.12) to Eq. (6.3.17)). □

DIFFEE has achieved the same convergence rates as the Diff-CLIME [10] and the DensityRatio estimator [71]. The FusedGLasso estimator has not provided such convergence rate analysis.

To derive the statistical error bound of DIFFEE, we need to assume that $[T_v(\widehat{\Sigma}_c)]^{-1}$ and $[T_v(\widehat{\Sigma}_d)]^{-1}$ are well-defined. This is ensured by assuming that the true Ω_c^* and Ω_d^* satisfy the following conditions [19]:

(C-MinInf$-\Sigma$): The true Ω_c^* and Ω_d^* of Eq. (1.0.3) have bounded induced operator norm, i.e., $|||{\Omega_c}^*|||_\infty := \sup_{w \neq 0 \in \mathbb{R}^p} \frac{||{\Sigma_c}^* w||_\infty}{||w||_\infty} \leq \kappa_1$ and $|||{\Omega_d}^*|||_\infty := \sup_{w \neq 0 \in \mathbb{R}^p} \frac{||{\Sigma_d}^* w||_\infty}{||w||_\infty} \leq \kappa_1$.

(C-Sparse-Σ): The two true covariance matrices Σ_c^* and Σ_d^* are "approximately sparse" (following [43]). For some constant $0 \leq q < 1$ and $c_0(p)$, $\max_i \sum_{j=1}^{p} |[\Sigma_c^*]_{ij}|^q \leq c_0(p)$ and $\max_i \sum_{j=1}^{p} |[\Sigma_d^*]_{ij}|^q \leq c_0(p)$. [4]

We additionally require $\inf_{w \neq 0 \in \mathbb{R}^p} \frac{||\Omega_c^* w||_\infty}{||w||_\infty} \geq \kappa_2$ and $\inf_{w \neq 0 \in \mathbb{R}^p} \frac{||\Omega_d^* w||_\infty}{||w||_\infty} \geq \kappa_2$.

6.3 Proofs

For the proposed DIFFEE model, $\mathcal{R} = ||\cdot||_1$. Based on the results in [27], $\Psi(\bar{\mathcal{M}}) = \sqrt{k}$, where k is the total number of nonzero entries in Δ. Using $\mathcal{R} = ||\cdot||_1$ in Theorem (3.2.2), we have the following theorem (the same as Theorem (6.2.1)),

Theorem 6.3.1. *Suppose that $\mathcal{R} = ||\cdot||_1$ and the true parameter Δ^* satisfy the conditions **(C1)(C2)** and $\lambda_n \geq \mathcal{R}^*(\widehat{\Delta} - \Delta^*)$, then the optimal point $\widehat{\Delta}$ of Eq. (6.1.3) has the following error bounds: $||\widehat{\Delta} - \Delta^*||_\infty \leq 2\lambda_n$, $||\widehat{\Delta} - \Delta^*||_2 \leq 4\sqrt{k}\lambda_n$, and $||\widehat{\Delta} - \Delta^*||_1 \leq 8k\lambda_n$*

Proof of Theorem (3.2.2)

Proof. Let $\delta := \widehat{\theta} - \theta^*$ be the error vector that we are interested in.

$$\begin{aligned} &\mathcal{R}^*(\widehat{\theta} - \theta^*) = \mathcal{R}^*(\widehat{\theta} - \widehat{\theta}_n + \widehat{\theta}_n - \theta^*) \\ &\leq \mathcal{R}^*(\widehat{\theta}_n - \widehat{\theta}) + \mathcal{R}^*(\widehat{\theta}_n - \theta^*) \leq 2\lambda_n \end{aligned} \tag{6.3.1}$$

By the fact that $\theta^*_{\mathcal{M}^\perp} = 0$, and the decomposability of $\mathcal{R}$ with respect to $(\mathcal{M}, \bar{\mathcal{M}}^\perp)$

[4] This indicates for some positive constant d, $[\Sigma_c^*]_{jj} \leq d$ and $[\Sigma_d^*]_{jj} \leq d$ for all diagonal entries. Moreover, if $q = 0$, then this condition reduces to Σ_d^* and Σ_c^* being sparse.

$$
\begin{aligned}
&\mathcal{R}(\theta^*) \\
&= \mathcal{R}(\theta^*) + \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&= \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&\leq \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^\perp}(\delta) + \Pi_{\bar{\mathcal{M}}}(\delta)] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \\
&- \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \\
&= \mathcal{R}[\theta^* + \delta] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)]
\end{aligned}
\tag{6.3.2}
$$

Here, the inequality holds by the triangle inequality of norm. Since Eq. (3.2.1) minimizes $\mathcal{R}(\widehat{\theta})$, we have $\mathcal{R}(\theta^* + \Delta) = \mathcal{R}(\widehat{\theta}) \leq \mathcal{R}(\theta^*)$. Combining this inequality with Eq. (6.3.2), we have:

$$
\mathcal{R}[\Pi_{\bar{\mathcal{M}}^\perp}(\delta)] \leq \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \tag{6.3.3}
$$

Moreover, by Hlder's inequality and the decomposability of $\mathcal{R}(\cdot)$, we have:

$$
\begin{aligned}
&||\Delta||_2^2 = \langle \delta, \delta \rangle \leq \mathcal{R}^*(\delta)\mathcal{R}(\delta) \leq 2\lambda_n \mathcal{R}(\delta) \\
&= 2\lambda_n[\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) + \mathcal{R}(\Pi_{\bar{\mathcal{M}}^\perp}(\delta))] \leq 4\lambda_n \mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\
&\leq 4\lambda_n \Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2
\end{aligned}
\tag{6.3.4}
$$

where $\Psi(\bar{\mathcal{M}})$ is a simple notation for $\Psi(\bar{\mathcal{M}}, ||\cdot||_2)$.

Since the projection operator is defined in terms of $||\cdot||_2$ norm, it is non-expansive: $||\Pi_{\bar{\mathcal{M}}}(\Delta)||_2 \leq ||\Delta||_2$. Therefore, by Eq. (6.3.4), we have:

$$
||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}), \tag{6.3.5}
$$

and plugging it back to Eq. (6.3.4) yields the error bound Eq. (3.2.4).

Finally, Eq. (3.2.5) is straightforward from Eq. (6.3.3) and Eq. (6.3.5).

$$\begin{aligned}\mathcal{R}(\delta) &\leq 2\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 2\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 8\lambda_n\Psi(\bar{\mathcal{M}})^2.\end{aligned} \tag{6.3.6}$$

□

Useful lemma(s)

Lemma 6.3.2. *(Theorem 1 of [44]). Let δ be $\max_{ij}|[\frac{X^TX}{n}]_{ij} - \Sigma_{ij}|$. Suppose that $v > 2\delta$. Then, under the conditions (C-SparseΣ), and as $\rho_v(\cdot)$ is a soft-threshold function, we can deterministically guarantee that the spectral norm of error is bounded as follows:*

$$|||T_v(\widehat{\Sigma}) - \Sigma|||_\infty \leq 5v^{1-q}c_0(p) + 3v^{-q}c_0(p)\delta \tag{6.3.7}$$

Lemma 6.3.3. *(Lemma 1 of [45]). Let $\mathcal{A}$ be the event that*

$$||\frac{X^TX}{n} - \Sigma||_\infty \leq 8(\max_i \Sigma_{ii})\sqrt{\frac{10\tau\log p'}{n}} \tag{6.3.8}$$

where $p' := \max n, p$ and τ is any constant greater than 2. Suppose that the design matrix X is i.i.d. sampled from Σ-Gaussian ensemble with $n \geq 40\max_i \Sigma_{ii}$. Then, the probability of event $\mathcal{A}$ occurring is at least $1 - 4/p'^{\tau-2}$.

To prove the bound of $||\Delta^* - ([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1})||_\infty$, we first prove the bound of $||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty$. In the following proof, we first derive the inequality $||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \leq |||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty|||\Omega_c^*|||_\infty||T_v(\widehat{\Sigma}_c) - \Sigma_c^*||_\infty$, which is bounded by multiplication of three parts. Then we use the above Lemmas and two conditions to prove the bound of each part. Finally, we combine the three results to have the whole bound of $||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty$.

Proof of Corollary (6.2.2)

Proof. In the following proof, we first prove $||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \leq \lambda_{n_c}$. Here $\lambda_{n_c} = \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}}$ and $p' = \max(p, n_c)$

The condition (C-SparseΣ) and condition (C-MinInfΣ) also hold for Ω_c^* and Σ_c^*. In order to utilize Theorem (6.3.1) for this specific case, we only need to show that $||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \leq \lambda_{n_c}$ for the setting of $\lambda_{n_c} = \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}}$:

$$\begin{aligned}
&||\Omega_c^* - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty = |||[T_v(\widehat{\Sigma}_c)]^{-1}(T_v(\widehat{\Sigma}_c)\Omega_c^* - I)||_\infty \\
&\leq |||[T_v(\widehat{\Sigma}_c)w]|||_\infty ||T_v(\widehat{\Sigma}_c)\Omega_c^* - I||_\infty \\
&= |||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty ||\Omega_c^*(T_v(\widehat{\Sigma}_c) - \Sigma_c^*)||_\infty \\
&\leq |||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty |||\Omega_c^*|||_\infty ||T_v(\widehat{\Sigma}_c) - \Sigma_c^*||_\infty.
\end{aligned} \tag{6.3.9}$$

We first compute the upper bound of $|||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty$. By the selection v in the statement, Lemma (6.3.2) and Lemma (6.3.3) hold with probability at least $1 - 4/p'^{\tau-2}$. Armed with Eq. (6.3.7), we use the triangle inequality of norm and the condition (C-SparseΣ): for any w,

$$\begin{aligned}
&||T_v(\widehat{\Sigma}_c)w||_\infty = ||T_v(\widehat{\Sigma}_c)w - \Sigma w + \Sigma w||_\infty \\
&\geq ||\Sigma w||_\infty - ||(T_v(\widehat{\Sigma}_c) - \Sigma)w||_\infty \\
&\geq \kappa_2 ||w||_\infty - ||(T_v(\widehat{\Sigma}_c) - \Sigma)w||_\infty \\
&\geq (\kappa_2 - ||(T_v(\widehat{\Sigma}_c) - \Sigma)w||_\infty)||w||_\infty
\end{aligned} \tag{6.3.10}$$

Where the second inequality uses the condition (C-SparseΣ). Now, by Lemma (6.3.2) with the selection of v, we have

$$|||T_v(\widehat{\Sigma}_c) - \Sigma|||_\infty \leq c_1(\frac{\log p'}{n_c})^{(1-q)/2} c_0(p) \tag{6.3.11}$$

where c_1 is a constant related only on τ and $\max_i \Sigma_{ii}$. Specifically, it is defined as $6.5\times(16(\max_i \Sigma_{ii})\sqrt{10\tau})^{1-q}$. Hence, as long as $n_c > (\frac{2c_1 c_0(p)}{\kappa_2})^{\frac{2}{1-q}} \log p'$ as stated, so that $|||T_v(\widehat{\Sigma}_c) - \Sigma|||_\infty \leq \frac{\kappa_2}{2}$, we can conclude that $||T_v(\widehat{\Sigma}_c)w||_\infty \geq \frac{\kappa_2}{2}||w||_\infty$, which implies $|||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty \leq \frac{2}{\kappa_2}$.

The remaining term in Eq. (6.3.9) is $||T_v(\widehat{\Sigma}_c) - \Sigma_c^*||_\infty$; $||T_v(\widehat{\Sigma}_c) - \Sigma_c^*||_\infty \leq ||T_v(\widehat{\Sigma}_c) - \widehat{\Sigma}_c||_\infty + ||\widehat{\Sigma}_c - \Sigma_c^*||_\infty$. By construction of $T_v(\cdot)$ in (C-Thresh) and by Lemma (6.3.3), we can confirm that

$||T_v(\widehat{\Sigma}_c) - \widehat{\Sigma}_c||_\infty$ as well as $||\widehat{\Sigma}_c - \Sigma_c^*||_\infty$ can be upper-bounded by v.

Similarly, the $[T_v(\widehat{\Sigma}_d)]^{-1}$ has the same result.

Finally,

$$||\Delta^* - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||_\infty \tag{6.3.12}$$

$$\leq ||\Omega_d - [T_v(\widehat{\Sigma}_d)]^{-1}||_\infty + ||\Omega_c - [T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \tag{6.3.13}$$

$$\leq \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}} + \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}} \tag{6.3.14}$$

Suppose $p > \max(n_c, n_d)$, we have that

$$||\Delta^* - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||_\infty \leq \frac{8\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p}{\min(n_c, n_d)}} \tag{6.3.15}$$

Similarly, we also have that

$$||\Delta^* - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||_F \leq \frac{32\kappa_1 a}{\kappa_2}\sqrt{\frac{k\log p}{\min(n_c, n_d)}} \tag{6.3.16}$$

, and

$$||\Delta^* - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||_1 \leq \frac{64\kappa_1 a}{\kappa_2}k\sqrt{\frac{\log p}{\min(n_c, n_d)}} \tag{6.3.17}$$

By combining all together, we can confirm that the selection of λ_n satisfies the requirement of Theorem (6.3.1), which completes the proof. □

6.3.1 Support Recovery Analysis

Theorem 6.3.4. *The support set of $\widehat{\Delta}$ is a subset of the support set of Δ^*. Moreover, under the additional assumption that $\min_{s\in supp(\Delta^*)} |\Delta_s^*| \geq 3||\Delta^* - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty$, it correctly includes all non-zero*

coordinates of Δ^*

Proof. In Theorem (6.3.1), we prove that $||\Delta^* - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty \leq \lambda_n$. Therefore, if $\Delta^*_{i,j} = 0$, then $|\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)_{i,j}| \leq \lambda_n$.

Thus, if $\Delta^*_{i,j} = 0$, then $\widehat{\Delta}_{i,j} = 0$. In another word the zero set of Δ^* is the subset of the zero set of $\widehat{\Delta}$.

Finally, we have that $supp(\widehat{\Delta}) \subseteq supp(\Delta^*)$

If $\min_{s \in supp(\Delta^*)} |\Delta^*_s| \geq 3||\Delta^* - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty$, then $supp(\Delta^*) \subseteq supp(\widehat{\Delta})$.

Thus, $supp(\Delta^*) = supp(\widehat{\Delta})$ □

6.4 Related Works

6.4.1 Previous Estimators for Change Estimation in GGM Structure

Multiple estimators have been proposed to estimate sparse differential network from two sets of samples.

FusedGLasso (Regularized MLE): The most straightforward estimator for differential network was to extend the classic Graphical lasso estimator [8] for sparse GGM with an added sparsity penalty on the differential network (i.e., fused norm).

$$\begin{aligned} \underset{\Omega_c, \Omega_d \succ 0, \Delta}{\operatorname{argmin}} \; & n_c(-\log\det(\Omega_c) + < \Omega_c, \widehat{\Sigma}_c >) \\ & + n_d(-\log\det(\Omega_d) + < \Omega_d, \widehat{\Sigma}_d >) \\ & + \lambda_2(||\Omega_c||_1 + ||\Omega_d||_1) + \lambda_n||\Delta||_1 \end{aligned} \tag{6.4.1}$$

This was solved by block coordinate descent algorithms in [16]. Later the alternating direction method of multipliers (ADMM) was used to solve Eq. (6.4.1) that needs to run SVD in one sub-procedure [11].

Diff-CLIME: Another recent study [10] extended the CLIME estimator to directly learn the Δ through a constrained optimization formulation.

$$\begin{aligned} &\underset{\Delta}{\operatorname{argmin}} \, ||\Delta||_1 \\ &\text{Subject to: } ||\widehat{\Sigma}_c \Delta \widehat{\Sigma}_d - (\widehat{\Sigma}_c - \widehat{\Sigma}_d)||_\infty \leq \lambda_n \end{aligned} \tag{6.4.2}$$

This reduces the estimation to solving multiple linear programming problems.

DensityRatio: The third category of estimators optimizes the following loss:

$$\underset{\Delta}{\operatorname{argmax}} \, \mathcal{L}_{\text{KLIEP}}(\Delta) - \lambda_n \parallel \Delta \parallel_1 - \lambda_2 \parallel \Delta \parallel_2 \tag{6.4.3}$$

Here KLIEP minimizes the KL divergence between the true probability density $p_d(x)$ and the estimated $\widehat{p}_d(x) = r(x;\Delta)p_c(x)$ without explicitly modeling the true $p_c(x)$ and $p_d(x)$. Its key idea is the formulation of density ratio term $r(x;\Delta)$ for directly estimating sparse differential network of graphical models in exponential families. This DensityRatio estimator uses the elastic-net penalty for enforcing Δ to be sparse. The resulting optimization was solved using proximal gradient descent methods in [71].

6.5 Experiments

We use two models of simulated datasets as well as a real world dataset for empirical comparisons.

- The first model mimics real world networks with a sparse differential network containing only hub nodes. This model can evaluate whether the method can efficiently infer the hub nodes in the differential network or not. In [10], the authors claim that if the change estimator also assumes the sparsity structure in Ω_c and Ω_d, then the estimator cannot achieve a good result on datasets generated by this data model.
- The second data simulation model, in contrast, generates random graphs that differ by a sparse random differential network. It evaluates the estimation performance of a certain estimator for inferring the randomly-generated differential networks.

- The real world dataset is a human brain fMRI dataset with two groups of subjects: autism and control. Our choice of this dataset is motivated by the recent literature in neuroscience that has suggested functional networks are not sparse. On the other hand, differences in functional connections across subjects should be sparse [21].

The two simulation models allow for a thorough evaluation of DIFFEE vs the baseline methods. The real-world data allows us to compare DIFFEE versus the baselines through classification using the estimated differential graph.

6.5.1 Experimental Setup

Baselines: We compare DIFFEE with (1) FusedGLasso [11], (2) DensityRatio [32], and (3) Diff-CLIME [10].

Hyper-parameters: We need to tune the value of three hyper-parameters in these experiments: v, λ_n and λ_2. In detail:

- v is used for soft-thresholding in DIFFEE. We choose v from the set $\{0.001i | i = 1, 2, \ldots, 1000\}$ and pick a value that makes $T_v(\Sigma_c)$ and $T_v(\Sigma_d)$ invertible.
- λ_n is the main hyper-parameter that control the sparsity of the estimated differential network. Based on our convergence rate analysis in Section 6.2, $\lambda_n \geq C\sqrt{\frac{\log p}{\min(n_c, n_d)}}$. Accordingly, we choose λ_n from a range of $\{0.01 \times \sqrt{\frac{\log p}{\min(n_c, n_d)}} \times i | i \in \{1, 2, 3, \ldots, 30\}\}$. The λ_n in the DensityRatio is tuned by their package.
- λ_2 controls individual graph's sparsity in FusedGLasso. We choose $\lambda_1 = 0.0001$ (a very small value) for all experiments to ensure only the differential network is sparse. λ_2 in the DensityRatio is set to 0.2 according to their package.

Evaluation Metrics: We evaluate DIFFEE and the baseline methods on both contexts of effectiveness and scalability.

- F1-score: We first use the edge-level F1-score to compare the predicted versus true differential graph. Here, F1 $= \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$, where Precision $= \frac{\text{TP}}{\text{TP+FP}}$ and Recall $= \frac{\text{TP}}{\text{TP+FN}}$. TP (true positive) means the number of true edges correctly estimated by the predicted differential network. FP (false positive) and FN (false negative) are the number of incorrectly predicted nonzero entries

and zero entries respectively. We repeat the experiment 10 times for each method and use the average metrics for comparison. The better method achieves a higher F1-score.

- Time Cost: We use the execution time (measured in seconds or log(seconds)) for a method as a measure of its scalability. To ensure a fair comparison, we try 30 different λ (or λ_2) and measure the total time of execution for each method. The better method uses less time[5].

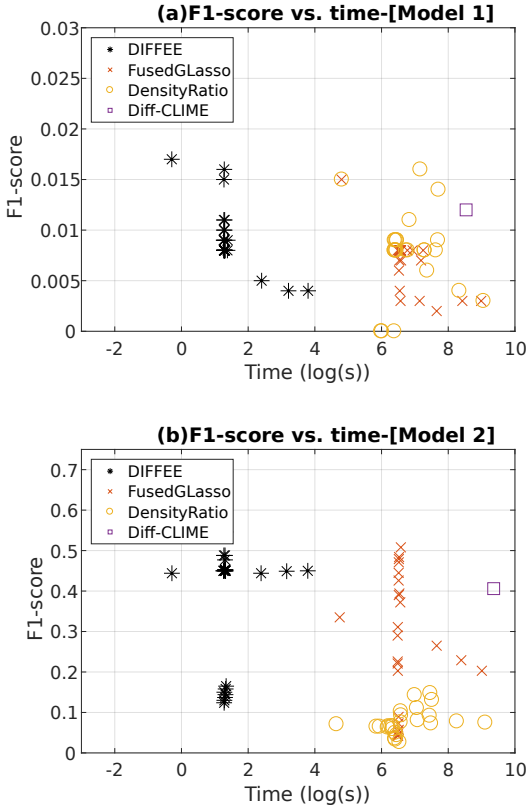
- Low F1 values on Model 1 datasets: The F1-score of all cases in Figure 6.2(a) appear quite low. This is due to the fact that simulated differential networks from Model 1 are extremely sparse (e.g., only 0.1% non-zero edges among all possible edges). For example, if the estimated $\widehat{\Delta}$ only predicts 5% zero entries incorrectly (i.e., FP=5%) and correctly predicts all the rest entries (TP = 0.1%, TN = 94.9%). The precision equals to $\frac{\text{TP}}{\text{TP + FP}} = \frac{0.1\%}{0.1\%+5\%} \approx 0.02$, which is a small number. The recall equals to $\frac{\text{TP}}{\text{TP + FN}} = \frac{0.1\%}{0.1\%+0\%} = 1$. Then F1 $= \frac{\text{precision}\cdot\text{recall}}{2(\text{precision}+\text{recall})} \approx 0.01$, which is also a relatively small number. However, the estimator only wrongly inferred 5% zero entries, which is still a good result. Therefore, low F1-score doesn't mean that the estimator is bad when the differential network is extremely sparse.

  This extreme sparsity also influences other evaluation metrics. For instance, if the estimated $\widehat{\Delta}$ only includes 1% zero entries and 0.05% non-zero entries incorrectly (i.e., FP=5% and FN=0.05%) and correctly predicts all the rest entries (TP=0.05% and TN=94.9%). The TPR $= \frac{0.05\%}{0.05\%+0.05\%} = 0.5$ and FPR $= \frac{5\%}{5\%+94.9\%} \approx 0.2$. If you plot this point in the FPR vs. TPR curve, it is not good. However from the angle of accuracy, this method only predicts wrongly around 5% edges, which indicates that it performs well.

Simulated Data Generation: We first simulate precision matrices Ω_c and Ω_d by Model 1 or Model 2. To simulate data for the control block, we generate n_c data samples following multivariate gaussian distribution with mean 0 and covariance matrix $(\Omega_c)^{-1}$. We use the multivariate distribution method from stochastic simulation [72] to sample the simulated data blocks. In our implementation, we directly use the R function "**mvrnorm**" in **MASS** package. We repeat the same process for the case group with Ω_d. Then, we apply DIFFEE and baseline methods to obtain the estimated differential networks.

[5]The machine that we use for experiments is an Intel(R) Core(TM) i7-6850k CPU @ 3.60GHz with a 64GB memory.

Figure 6.1: F1-score versus Time Cost(log(seconds)) for different methods and synthetic data models (a) F1-score vs. Time for Model 1. (b)F1-score vs. Time for Model 2.



Two models to generate simulated datasets: Using the following two graph models, we generate multiple sets of synthetic multivariate-Gaussian datasets.

- **Model 1 – mimic real-world networks with hub nodes:** Inspired by [10], this model assumes that the graphs mimic real-world networks [73]. We first generate Ω_d as a network with $s \cdot \frac{p(p-1)}{2}$ edges following a power-law degree distribution with an expected power parameter of 2. Here s is a parameter that controls the sparsity of the two graphs. A larger value of s corresponds to denser graphs. Next, the value of each nonzero entry of Ω_d is generated from a uniform distribution with $[-10/p, -4/p] \cup [4/p, 10/p]$, where division by p ensures the positive definiteness of Ω_c and Ω_d. The diagonals are then set to 1 and Ω_d is symmetrized by averaging it with its transpose $(\frac{1}{2}(\Omega_d + \Omega_d^T))$. The differential network Δ is generated by the top 20% edges of the top 2 hub nodes in Ω_d. $\Omega_c = \Omega_d - \Delta$.
- **Model 2 – random graph model:** Following [47], this model assumes $\Omega_c = \mathbf{B}_c + \mathbf{B}_S + \delta_c I$ and $\Omega_d = \mathbf{B}_d + \mathbf{B}_S + \delta_d I$, where each off-diagonal entry in $\mathbf{B}_c$ and $\mathbf{B}_d$ are generated independently

and equals 0.5 with probability 0.1 and 0 with probability 0.9. The shared part $\mathbf{B}_S$ is generated independently and equal to 0.5 with probability $0.1s$ and 0 with probability $1 - 0.1s$. Similar to Model 1, s controls the sparsity of the two graphs. δ_c and δ_d are selected large enough to guarantee the positive definiteness. A clear differential network structure $\Delta = \mathbf{B}_d - \mathbf{B}_c$ exists between these two graphs.

Following Model 1 or Model 2, for each case of simulated data generation, we generate two blocks of data samples following the distribution $N(0, (\Omega_c)^{-1})$ and $N(0, (\Omega_d)^{-1})$.

6.5.2 Experiments on Simulated Datasets

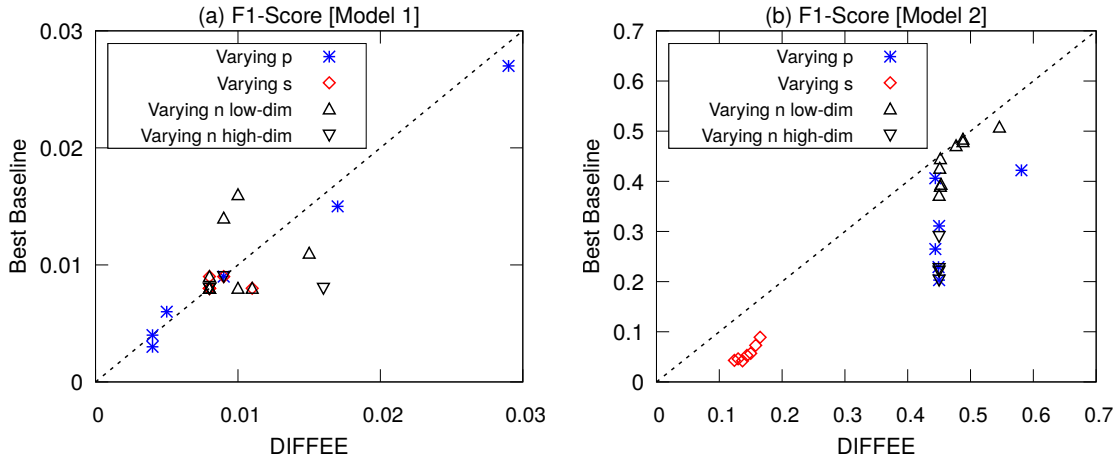
Experimental Design: By varying the number of features p, amount of sparsity s, and the number of samples (n_c,n_d), we can generate many cases of simulated datasets. This allows us to comprehensively evaluate DIFFEE across a wide range of data situations. To this end, we design the following three sets of synthetic experiments by varying p, s, n_c, and n_d:

- p (the number of features): The first set of experiments varies p in the set of $\{50, 100, 200, 300, 400, 500\}$ while setting n_c and n_d as $p/2$ and the sparsity parameter $s = 0.2$.
- s (the sparsity): In the second set of experiments, we vary the value of the sparsity parameter s in the set of $\{0.1, 0.2, \ldots, 0.7\}$, while using $p = 200$ and $n_c = n_d = p/2$.
- n_c and n_d (the number of samples): In the third set, we vary the number of samples in both groups and set $p = 200$ and $s = 0.2$. We group this set of experiments into two categories: low-dimensional cases, and high-dimensional cases. For the high dimensional case, we vary n_c and n_d from the value set of $\{p/4, p/2\}$. Similarly, for the low dimensional case, we vary n_c and n_d from the value set of $\{p, 2p, 3p\}$.

Experiment Results: We compare DIFFEE with the baselines regarding two aspects– (a) Effectiveness, and (b) Scalability.

(a) Effectiveness : We evaluate the prediction effectiveness of using F1-Score. Figure 6.2 presents the summarized results of our DIFFEE versus baselines on all 50 cases of simulated datasets. As explained above, the simulated datasets are generated by varying the parameters p, s, n_c, and n_d by data Model 1 and Model 2. In Figure 6.2 (a) and (b), we plot the F1-Score of DIFFEE vs

Figure 6.2: F1-Score of DIFFEE vs the F1-Score of the best performing baseline. The more points below the diagonal line, the better. (a) On simulated datasets from Model 1 (b) On simulated datasets from Model 2. (Black up-triangles describe 'varying (n_c, n_d) in low dimensions'; Black down-triangles describe 'varying (n_c, n_d) in high-dimensions; Red diamonds represent 'varying s'; and Blue stars represent 'varying p'.)



(a) (b)

the F1-Score of the best performing baseline on each simulated case from Model 1 and Model 2, respectively. Each point in the two figures is obtained by comparing DIFFEE vs. the best baseline among all baselines on one simulated case. Each point below the line $y = x$ indicates that DIFFEE achieves better performance over baselines. Overall Figure 6.2 shows that DIFFEE outperforms the corresponding best baseline in almost all cases. The only two points for which DIFFEE doesn't do as well as the corresponding best baseline DensityRatio are two low dimensional cases. This is as expected because the design of DIFFEE is for high-dimensional cases (i.e., the choices of proxy backward mapping). Details of F1-Scores from all simulation cases and discussions of low F1 values on Model 1 are in Appendix.

(b) Scalability : To evaluate DIFFEE and the baselines on scalability, Figure 6.3 presents the time cost vs. varying p, varying sparsity (s) and varying number of samples in the 'c' group (n_c). Figure 6.3 (a),(c) and (e) show time results from data Model 1. Figure 6.3 (b),(d) and (f) correspond to datasets from Model 2. We interpolate the points of computation time from each estimator into curves. For each simulation case, the computation time for each estimator is the summation of a method's execution time over all values of λ_n. Figure 6.3 shows that in general the time costs of FusedGLasso and DensityRatio are roughly comparable. DIFFEE is about 100 times better than both (detailed numbers are provided in Table 6.3 to Table 6.10). Diff-CLIME is extremely slow when p increases. Because Figure 6.3 (c),(d),(e) and (f) are about data cases with $p = 200$, we can not run Diff-CLIME on these cases (it cannot finish any $p = 200$ case for a single value of λ_n by one day).

Interestingly, the empirical time results match the computational analysis in Table 6.1. Especially DensityRatio's time cost grows quickly when n_c increases. In contrast the running time of DIFFEE and FusedGLasso are not connected strongly to the size of samples. Overall DIFFEE costs much less computation time than the baselines and can significantly scale up to larger p.

6.5.3 A Real-World Dataset about Functional Connectivity among Brain Regions

We then use DIFFEE for a classification task on a well-known human brain fMRI dataset: ABIDE [61].

ABIDE Dataset: This data is from the Autism Brain Imaging Data Exchange (ABIDE) [61], a publicly available resting-state fMRI dataset. The ABIDE data aims to understand human brain connectivity and how it reflects neural disorders [62]. The data is retrieved from the Preprocessed Connectomes Project [63], where preprocessing is performed using the Configurable Pipeline for the Analysis of Connectomes (CPAC) [64] without global signal correction or band-pass filtering. After preprocessing with this pipeline, 871 individuals remain (468 diagnosed with autism). Signals for the 160 (number of features $p = 160$) regions of interest (ROIs) in the often-used Dosenbach Atlas [65] are examined.

Cross-validation: Classification is performed using the 3-fold cross-validation suggested by the literature [66] [67]. The subjects are randomly partitioned into three equal sets: a training set, a validation set, and a test set. Each estimator produces $\widehat{\Delta}$ using the training set. Then, these differential networks are used as inputs to linear discriminant analysis (LDA), which is tuned via cross-validation on the validation set. Finally, accuracy is calculated by running LDA on the test set. This classification process aims to assess the ability of an estimator to learn the differential patterns of the connectome structures. Notably, the DensityRatio method cannot be compared on this data, because the method does not provide the precision matrices necessary for LDA.

Classification Results: Table 6.2 displays the maximum accuracy achieved by DIFFEE, FusedGLasso, and Diff-CLIME, after tuning over hyperparameters. DIFFEE yields a classification

Table 6.2: Classification accuracy obtained on the ABIDE dataset using DIFFEE, FusedGLasso, and Diff-CLIME. DIFFEE achieves the highest classification accuracy.

Method	DIFFEE	FusedGLasso	Diff-CLIME
Accuracy (%)	**57.58%**	56.90%	53.79%

accuracy of 57.58% distinguishing the autism and control groups, outperforming the FusedGLasso and Diff-CLIME estimators.

6.5.4 Detailed Empirical Results

Figure 6.1 (a) and (b) summarize DIFFEE's better performance in both scalability and effectiveness for all experiment settings in Model 1 and Model 2, respectively. Each point in Figure6.1 represents both the F1-Score and Time Cost of a method. Most of the DIFFEE points lie in the top left area, indicating lesser Time Cost and higher F1-scores compared to the other baselines.

Table 6.3 and Table 6.4 present the detailed results on the simulated datasets, comparing the scalability to p of our proposed method DIFFEE with the baselines FusedGLasso, Density Ratio, and Diff-CLIME. The Table 6.3 and Table 6.4 are obtained by experimental settings under Model 1 and Model 2 respectively. We vary number of features p in the set of $\{100, 200, 300, 400, 500\}$. The computation time for each case is the summation of the computational time for the method over a range of $\lambda_n \in \{0.01 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i | i \in \{1, 2, 3, \ldots, 30\}\}$. The F1-score for each case is the best result over a range of $\lambda_n \in \{0.01 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i | i \in \{1, 2, 3, \ldots, 30\}\}$. The Diff-CLIME cannot finish any tasks in one day. So all the results in the column "Diff-CLIME" are indicated by "NA". In most of the synthetic datasets, DIFFEE achieves a higher F1-Score and less computation time than other baselines. This proves that DIFFEE outperforms the baselines in both effectiveness and scalability.

Table 6.5 and Table 6.6 present the detailed performance results of our proposed method DIFFEE and others by varying the sparsity level s. The Table 6.5 and Table 6.6 are obtained by Model 1 and Model 2 respectively. We vary the sparsity parameter s in the set of $\{0.1, 0.2, \ldots, 0.7\}$. The computation time and F1-Score are measured similar to Table 6.3 and Table 6.4. In all of the synthetic datasets, DIFFEE performs better as indicated by its higher F1-score and lesser computation time than other baselines.

Table 6.7 and Table 6.8 present the detailed results of our proposed method–DIFFEE versus the corresponding baselines FusedGLasso, Density Ratio, and Diff-CLIME on the simulated datasets

Table 6.3: Model 1 varying p

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower	Diff-CLIME	Slower
6*F1-score	p = 50	**0.029**	0		0.027		0.016	
	p = 100	**0.017**	0.015		0.015		0.012	
	p = 200	**0.009**	0.008		0.009		NA	
	p = 300	0.005	0.002		**0.006**		NA	
	p = 400	**0.004**	0.003		0.004		NA	
	p = 500	**0.004**	0.003		0.003		NA	
6*Time (s)	p = 50	**0.296**	45.61	154×	24.903	84×	56.37	190×
	p = 100	**0.748**	121.537	162×	122.596	163×	5094.796	6811×
	p = 200	**3.645**	715.672	196×	611.341	167×	NA	
	p = 300	**11.064**	2106.681	190×	1584.262	143×	NA	
	p = 400	**24.763**	4551.419	183×	4159.019	167×	NA	
	p = 500	**44.54**	8008.809	179×	8575.529	192×	NA	

Table 6.4: Model 2 varying p

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower	Diff-CLIME	Slower
6*F1-score	p = 50	**0.581**	0.401		0.082		0.422	
	p = 100	**0.444**	0.335		0.071		0.406	
	p = 200	**0.45**	0.311		0.066		NA	
	p = 300	0.444	0.265		0.073		NA	
	p = 400	**0.449**	0.229		0.078		NA	
	p = 500	**0.45**	0.203		0.075		NA	
6*Time (s)	p = 50	**0.274**	43.57	159×	19.35	70×	116.712	425×
	p = 100	**0.751**	115.049	153×	104.53	139×	11640.82	15500×
	p = 200	**3.528**	657.147	186×	538.842	152×	NA	
	p = 300	**10.887**	2106.415	193×	1780.176	163×	NA	
	p = 400	**23.462**	4406.156	187×	3859.082	164×	NA	
	p = 500	**44.163**	8164.19	184×	9054.507	205×	NA	

Table 6.5: Model 1 varying sparsity

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower
7*F1-score	s = 0.1	0.008	0.003		**0.009**	
	s = 0.2	**0.009**	0.008		0.009	
	s = 0.3	**0.008**	0.008		0.008	
	s = 0.4	**0.011**	0.008		0.008	
	s = 0.5	**0.008**	0.006		0.008	
	s = 0.6	**0.008**	0.008		0.008	
	s = 0.7	**0.008**	0.007		0.008	
7*Time (s)	s = 0.1	**3.606**	712.682	197×	631.582	175×
	s = 0.2	**3.993**	712.365	178×	598.191	149×
	s = 0.3	**3.97**	719.859	181×	595.246	149×
	s = 0.4	**3.65**	721.785	197×	598.009	163×
	s = 0.5	**3.632**	679.94	187×	631.062	173×
	s = 0.6	**3.693**	679.263	183×	608.358	164×
	s = 0.7	**3.679**	686.979	186×	624.632	169×

varying different n_c and n_d in a high-dimensional setting ($p > \max(n_c, n_d)$). The Table 6.7 and Table 6.8 are obtained by Model 1 and Model 2, respectively. We vary the number of samples n_c and n_d in the set of $\{p/2, p/4\}$. The computation time and F1-Score are measured similar to

Table 6.6: Model 2 varying sparsity

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower
7*F1-score	s = 0.1	**0.165**	0.089		0.066	
	s = 0.2	**0.158**	0.073		0.059	
	s = 0.3	**0.15**	0.057		0.05	
	s = 0.4	**0.144**	0.053		0.044	
	s = 0.5	**0.137**	0.042		0.036	
	s = 0.6	**0.13**	0.046		0.033	
	s = 0.7	**0.124**	0.043		0.027	
7*Time (s)	s = 0.1	**3.817**	671.255	175×	564.679	147×
	s = 0.2	**3.763**	671.499	178×	559.455	148×
	s = 0.3	**3.62**	674.941	186×	609.633	168×
	s = 0.4	**3.741**	664.363	177×	635.302	169×
	s = 0.5	**3.691**	662.802	179×	603.838	163×
	s = 0.6	**3.619**	659.336	182×	611.441	168×
	s = 0.7	**3.596**	648.885	180×	689.137	191×

Table 6.7: model1 varying n_c and n_d in high-dimensional setting

	Model	**DIFFEE**	FusedGLasso	Slower	Density Ratio	Slower
4*F1-score	$n_c = p/4, n_d = p/4$	**0.008**	0.008		0	
	$n_c = p/4, n_d = p/2$	**0.008**	0.008		0	
	$n_c = p/2, n_d = p/4$	**0.016**	0.008		0	
	$n_c = p/2, n_d = p/2$	**0.009**	0.008		0.009	
4*Time (s)	$n_c = p/4, n_d = p/4$	**3.647**	696.742	191×	398.226	109×
	$n_c = p/4, n_d = p/2$	**3.61**	704.943	195×	590.044	163×
	$n_c = p/2, n_d = p/4$	**3.609**	697.858	193×	408.149	113×
	$n_c = p/2, n_d = p/2$	**3.582**	654.147	182×	642.168	179×

Table 6.8: model2 varying n_c and n_d in high-dimensional setting

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower
4*F1-score	$n_c = p/4, n_d = p/4$	**0.45**	0.221		0.065	
	$n_c = p/4, n_d = p/2$	**0.45**	0.226		0.063	
	$n_c = p/2, n_d = p/4$	**0.45**	0.29		0.065	
	$n_c = p/2, n_d = p/2$	**0.45**	0.203		0.066	
4*Time (s)	$n_c = p/4, n_d = p/4$	**3.74**	654.227	174×	381.686	102×
	$n_c = p/4, n_d = p/2$	**3.748**	654.822	174×	484.77	129×
	$n_c = p/2, n_d = p/4$	**3.717**	653.657	175×	346.148	93×
	$n_c = p/2, n_d = p/2$	**3.528**	657.147	186×	494.066	140×

Table 6.3 and Table 6.4. In most of the synthetic datasets, DIFFEE achieves a higher F1-Score and less computation time than other baselines.

Table 6.9 and Table 6.10 present the performance of our proposed method–DIFFEE and other methods with varying n_c and n_d in a low-dimensional setting ($p > \max(n_c, n_d)$). The Table 6.9 and Table 6.10 correspond to Model 1 and Model 2, respectively. We vary the number of samples n_c and n_d in the set of $\{p, 2p, 3p\}$. The computation time and F1-Score are measured similar to Table 6.3 and Table 6.4. In most of the synthetic datasets, DIFFEE achieves a higher F1-Score and

less computation time than other baselines.

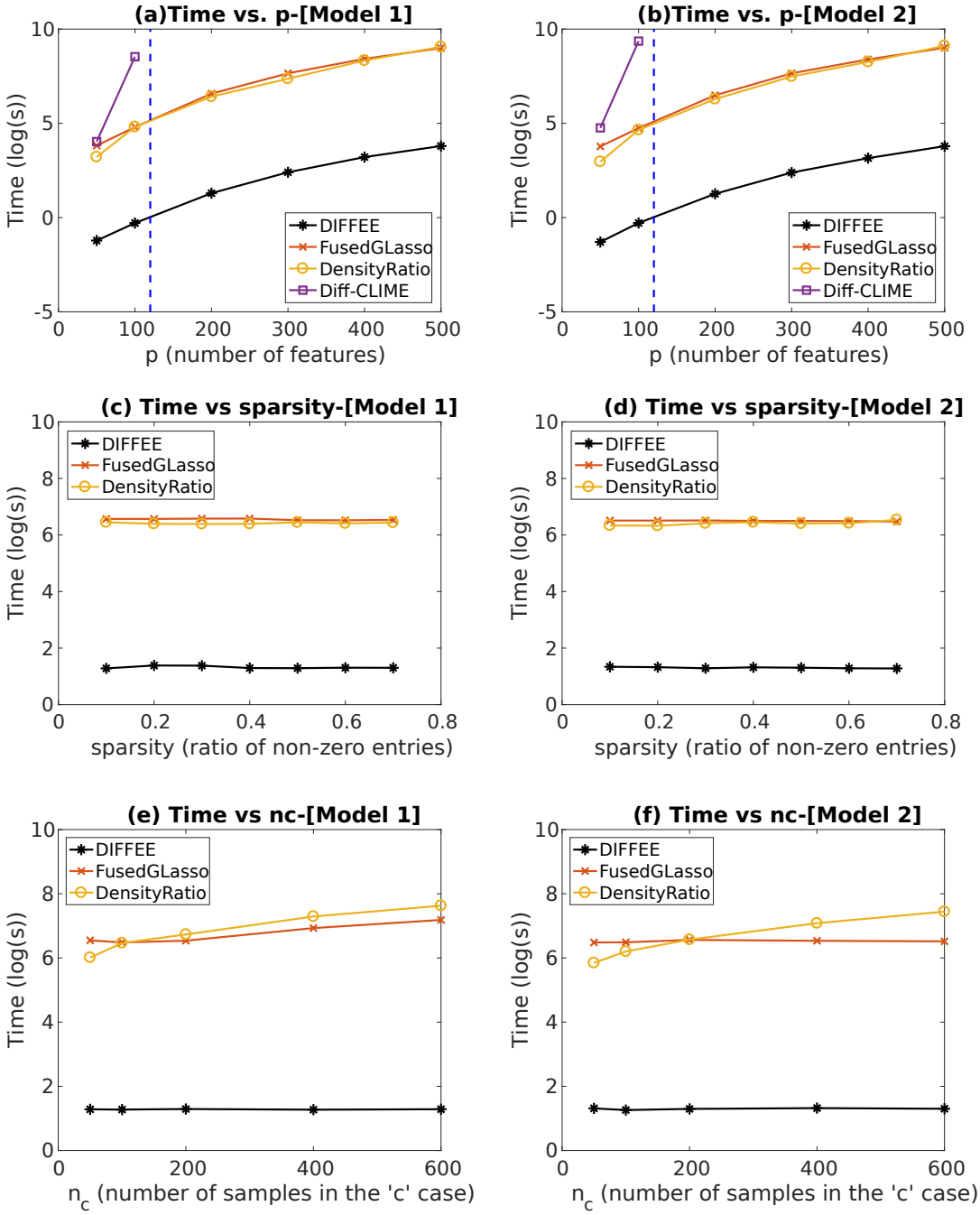
Table 6.9: model1 varying n_c and n_d in low-dimensional setting

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower
9*F1-score	$n_c = p, n_d = p$	**0.01**	0.008		0.008	
	$n_c = p, n_d = 2p$	**0.011**	0.008		0.008	
	$n_c = p, n_d = 3p$	**0.008**	0.007		0.008	
	$n_c = 2p, n_d = p$	**0.015**	0.008		0.011	
	$n_c = 2p, n_d = 2p$	0.01	0.008		**0.016**	
	$n_c = 2p, n_d = 3p$	0.009	0.008		**0.014**	
	$n_c = 3p, n_d = p$	**0.008**	0.004		0.008	
	$n_c = 3p, n_d = 2p$	**0.008**	0.007		0.008	
	$n_c = 3p, n_d = 3p$	0.008	0.003		**0.009**	
9*Time (s)	$n_c = p, n_d = p$	**3.643**	691.581	189×	838.863	230×
	$n_c = p, n_d = 2p$	**3.569**	1023.507	286×	1468.593	411×
	$n_c = p, n_d = 3p$	**3.62**	1319.354	364×	2054.228	567×
	$n_c = 2p, n_d = p$	**3.578**	700.539	195×	932.511	260×
	$n_c = 2p, n_d = 2p$	**3.568**	875.55	245×	1291.795	362×
	$n_c = 2p, n_d = 3p$	**3.553**	1406.44	395×	2224.744	626×
	$n_c = 3p, n_d = p$	**3.587**	696.087	194×	882.885	246×
	$n_c = 3p, n_d = 2p$	**3.578**	725.195	202×	1464.343	409×
	$n_c = 3p, n_d = 3p$	**3.592**	1264.346	351×	2191.003	609×

Table 6.10: model2 varying n_c and n_d in low-dimensional setting

	Model	DIFFEE	FusedGLasso	Slower	Density Ratio	Slower
9*F1-score	$n_c = p, n_d = p$	**0.45**	0.372		0.076	
	$n_c = p, n_d = 2p$	**0.453**	0.394		0.081	
	$n_c = p, n_d = 3p$	**0.452**	0.39		0.092	
	$n_c = 2p, n_d = p$	**0.451**	0.426		0.093	
	$n_c = 2p, n_d = 2p$	**0.477**	0.471		0.111	
	$n_c = 2p, n_d = 3p$	**0.488**	0.479		0.131	
	$n_c = 3p, n_d = p$	**0.452**	0.445		0.103	
	$n_c = 3p, n_d = 2p$	**0.488**	0.484		0.143	
	$n_c = 3p, n_d = 3p$	**0.546**	0.508		0.148	
9*Time (s)	$n_c = p, n_d = p$	**3.658**	707.735	193×	714.371	195×
	$n_c = p, n_d = 2p$	**3.746**	688.608	183×	1192.792	318×
	$n_c = p, n_d = 3p$	**3.673**	676.806	184×	1707.516	464×
	$n_c = 2p, n_d = p$	**3.69**	673.112	182×	723.656	196×
	$n_c = 2p, n_d = 2p$	**3.691**	676.597	183×	1164.175	315×
	$n_c = 2p, n_d = 3p$	**3.57**	677.65	189×	1830.678	512×
	$n_c = 3p, n_d = p$	**3.692**	673.364	182×	717.752	194×
	$n_c = 3p, n_d = 2p$	**3.692**	682.499	184×	1090.64	295×
	$n_c = 3p, n_d = 3p$	**3.732**	719.733	192×	1739.274	466×

Figure 6.3: Time Cost(log(seconds)) of DIFFEE versus the baseline methods (a) Time vs. number of features(p) for Model 1. (b)Time vs. number of features(p) for Model 2. (c) Time vs. sparsity(s) for Model 1. (d) Time vs. sparsity(s) for Model 2. (e)Time vs. number of samples in 'c' case (n_c) for Model 1. (f) Time vs. number of samples in the 'c' case (n_c) for Model 2.



Chapter 7

Variations and Extensions

7.1 Nonparanormal Graphical Models

Though sGGM is powerful, its normality assumption is commonly violated in real applications (e.g., for the TF ChIP-Seq data). the histogram of one of its TF variables is clearly not following Gaussian distribution (across samples, shown as the right distribution graph in Figure 7.1). After a univariate log-transformation of the same feature, we obtain its distribution histogram as the left graph in Figure 7.1. The transformed data samples are approximately normally distributed. This motivates us to adopt a more generalized UGM (recently proposed in [74]) to overcome the limitation of sGGM. This so-called "nonparanormal graphical model" (NGM) [74] assumes that data samples follow a multivariate nonparanormal distribution, which is a strict superset of the Gaussian distribution. We extend our models to the nonparanormal family and name these novel variations "NJEEK", "NFASJEM", and "NDIFFEE".

7.1.1 Background: Nonparanormal Graphical Model

A random variable $Z = (Z_1, \ldots, Z_p)^T$ is said to follow a nonparanormal distribution

$$Z \sim NPN_p(\mu, \mathbf{S}; f_1, \ldots, f_p)$$

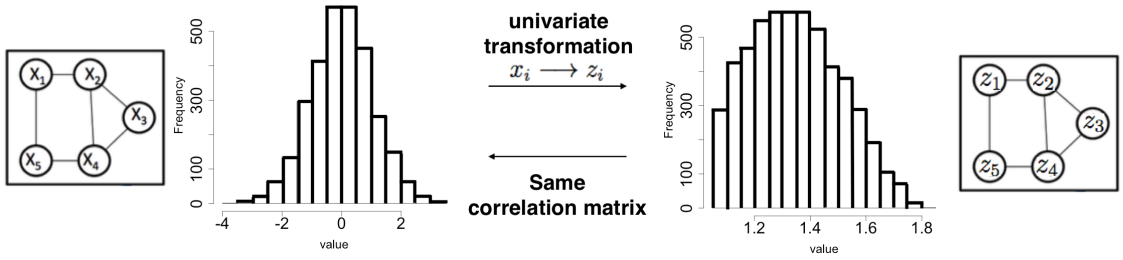


Figure 7.1: A simple example showing nonparanormal graphical model and its unobserved latent Gaussian graphical model. The leftmost sub-figure shows $X \sim N_p(\mu, \mathbf{S})$. The rightmost sub-figure shows $Z \sim NPN_p(\mu, \mathbf{S}; f_1, \ldots, f_p)$. The right distribution graph shows the histogram of one feature $\mathbf{z}_i$ (one TF variable) from a real TF binding dataset. The left histogram graph shows the distribution of a log-transformation of the same feature ($\mathbf{z}_i$). Because we can clearly see that the left histogram roughly follows a Gaussian distribution, this indicates $\mathbf{z}_i$ follows a nonparanormal distribution. This shows the need to extend SIMULE to the nonparanormal distribution that is a strict superset of the Gaussian distribution.

if and only if there exists a set of univariate strictly increasing transformations $f = \{f_j\}_{j=1}^p$ such that:

$$f(Z) = (f_1(Z_1), \ldots, f_p(Z_p))^T := X \sim N(\mu, \mathbf{S})$$

Figure 7.1 shows a simple example of nonparanormal graphical model (NGM) (inside the rightmost sub-figure) and its unobserved latent Gaussian graphical model (inside the leftmost sub-figure). Assume that we are given a dataset including n observations that are independently and identically drawn from $NPN_p(\mu, \mathbf{S}; f_1, \ldots, f_p)$, a multivariate nonparanormal distribution. The conditional independence graph G among Z_i variables can be modeled with a corresponding NGM [74]. The graph structure of NGM is encoded through the sparsity pattern of the inverse covariance matrix $\Phi := \mathbf{S}^{-1}$, where $\mathbf{S}$ denotes the covariance matrix of NPN_p.

7.1.2 Background: Estimate S through Rank-based Measures of Correlation Matrix $\mathbf{S}_0$

Since the direct estimation of covariance matrix $\mathbf{S}$ is difficult in nonparanormal distribution, recent studies have proposed an efficient nonparametric estimator [74] for $\mathbf{S}$. This estimator is derived from the correlation matrix $\mathbf{S}_0$. Because the covariance matrix $\mathbf{S} = diag(\mathbf{S}_i)\mathbf{S}_0 diag(\mathbf{S}_i)$, $\mathbf{S}^{-1} = diag(\mathbf{S}_i)^{-1}\mathbf{S}_0^{-1}diag(\mathbf{S}_i)^{-1}$. Here $\mathbf{S}_i = \sqrt{Cov(Z_i, Z_i)}$ and $diag(\mathbf{S}_i) = diag(\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_p)$. Therefore, the inverse of correlation matrix ($\mathbf{S}_0^{-1}$) and the inverse of covariance matrix ($\mathbf{S}^{-1}$) have the same nonzero and zero entries. Based on this observation, [74] proposed a nonparametric method to

estimate the correlation matrix $\mathbf{S}_0$, instead of estimating the covariance matrix $\mathbf{S}$ for the purpose of structure inference.

In [74] the authors proposed using the population Kendall's tau correlation coefficients τ_{jk} to estimate $\mathbf{S}_0$, based upon the explicit relationship between this rank-based measure τ_{jk} and the correlation measure $(\mathbf{S}_{jk})_0$ for a given nonparanormal dataset $Z \sim NPN_p(\mu, \mathbf{S}, f_1, \ldots, f_p)$ (discussed in [75]). Figure 7.1 presents the simple relationship between $Z \sim NPN_p(\mu, S; f_1, \ldots, f_p)$ and its latent $X \sim N(\mu, S)$. To simplify notations, we use $\mathbf{S}$ to represent the correlation matrix for the remainder of this paper.

Theorem 7.1.1. *Given $Z \sim NPN_p(\mu, S, f_1, \ldots, f_p)$, a nonparanormal distribution, we have that*

$$\mathbf{S}_{jk} = sin(\frac{\pi}{2}\tau(Z_j, Z_k)). \tag{7.1.1}$$

where the Kendall's tau can be estimated as:

$$\widehat{\tau}_{jk} = \frac{1}{n(n-1)} \sum_{1 \le i \le i' \le n} \text{sign}((\mathbf{z}_j^i - \mathbf{z}_j^{i'})(\mathbf{z}_k^i - \mathbf{z}_k^{i'}))$$

Proof. The proof is provided in [74]. □

Therefore, the correlation matrix $\mathbf{S}$ can be estimated as:

$$\widehat{\mathbf{S}}_{jk} = sin(\frac{\pi}{2}\widehat{\tau}_{jk}).$$

We can then plug in the estimated $\widehat{\mathbf{S}}$ for learning the dependency graph structure in the corresponding NGM.

7.1.3 JEE for Learning Multiple Nonparanormal Graphical Models

We can now substitute each sample covariance matrix $\widehat{\Sigma}^{(i)}$ or $\widehat{\Sigma}_{c,d}$ used in our works from each task with its corresponding correlation matrix $\mathbf{S}^{(i)}$ as estimated above. The rest of the computations are the same as our estimators. We refer to this whole process as "NJEEK", "NFASJEM", and "NDIFFEE".

Theorem 7.1.2. *If X, Y are two independent random variables and f,$g : \mathbb{R} \to \mathbb{R}$ are two measurable functions, then $f(X)$ and $g(Y)$ are also independent.*

Through the above theorem, the monotone functions f in NPN_p will not change the conditional dependency among variables. As proved in [74], the conditional dependency network among the latent Gaussian variables X (in this NPN_p) is the same as the conditional dependency network among the nonparanormal variables Z_i, with a parametric asymptotic convergence rate. Therefore, we can use the estimated correlation matrices $\mathbf{S}^{(i)}$ for the joint network inference of multiple sNGMs in our estimators. This is also because we have shown that the inverse of the correlation matrix and the inverse of the covariance matrix share the same nonzero and zero patterns.

7.2 Difficulty in combining the above estimators

We can also add more flexibility into the JEEK such as the second normalization function in FASJEM. It has the following formulation:

$$\begin{aligned} \underset{\Omega_I^{tot}, \Omega_S^{tot}}{\operatorname{argmin}} & ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}|| + \epsilon \mathcal{R}'(\Omega^{tot}) \\ \text{Subject to: } & ||W_I^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \\ & ||W_S^{tot} \circ (\Omega^{tot} - inv(T_v(\widehat{\Sigma}^{tot})))||_\infty \leq \lambda_n \\ \mathcal{R}^{*\prime}(\Omega^{tot}) \leq \epsilon \lambda_n & \end{aligned} \tag{7.2.1}$$

We have two ways to solve Eq. (7.2.1). The first one is the parallelized proximal algorithm. However, this algorithm requires the kw-norm has a closed-form proximity, which has not been discovered. The other way assume each ℓ_1 norm as the indepedent regularizer. However, this increases the number of proximities need to calculate to $K + 1$. Moreover, $\mathcal{R}'$ is for Ω^{tot} while each ℓ_1 is for either $\Omega^{(i)_I}$ or Ω_S. Therefore, none of these solutions can keep the algorithm be fast and scalable. We choose not to introduce this work in this paper.

Chapter 8

Conclusion and Future Works

8.1 Intellectual Merit

The three proposed models – JEEK, FASJEM, and DIFFEE are novel approaches speeding-up and scaling-up the joint estimation of multiple sGGMs from large-scale data. To the best of our knowledge, these methods are the first suite of tools using the elementary estimator mechanism to achieve a scalable and fast estimation of multi-sGGMs. The expected outcome will be a powerful toolkit that not only provides accurate edge detection but also allows fast computation on the datasets with a large number of features p.

8.2 Broader Impact

The proposed methods and tools are expected to impact scientific domains that need to manipulate massive high-dimensional and heterogeneous data sets. This set of potential tools will enable researchers like those working in network biology or brain connectivity to effectively extract novel network-driven hypotheses or knowledge from their scientific data-sets. In our experiments of finding connectivity among different genes, proteins or regions of the human brain from data, our method FASJEM and JEEK find more existing interactions between important entities when compared with the state-of-the-art baselines. On one human brain imaging dataset, our proposed method DIFFEE and JEEK achieved better classification accuracy than the state-of-art methods using the searched

edges as evidence in classification. We hope to provide a suite of faster and more scalable "data → connectivities" tools when data sets in many application fields become more heterogeneous and grow at a faster scale than computational capabilities.

8.3 Future works

As we stated above, the next step we make is including additional knowledge for the FASJEM. Furthermore, we are also interested in overcoming the difficulty stated in the Section 7.2.

8.4 Related works in a broader context

In this section, we also introduce some loosely related works.

8.4.1 Partial Correlation

Partial correlation measures the degree of association between two random variables, with the effect of a set of chosen random variables removed. If given the correlation matrix S, the partial correlation $\rho_{i,j}$ without other variables can be calculated by $P = S^{-1}$.

$$\rho_{i,j} = -\frac{P_{ij}}{\sqrt{P_{ii}P_{jj}}} \tag{8.4.1}$$

Therefore, the partial correlation matrix has the same sparsity pattern as the precision matrix.

8.4.2 Learning other Graphical Models

We focus on the sparse Gaussian Graphical model. However, there still exist other types of graphical models. For example, the Bayesian Network (BN) also captures the conditional dependency structure. Unlike the sGGM, it represents the conditional dependency relationships via a directed acyclic graph (DAG). It is important to point out that the BN and sGGM are similar, but they both have a certain type of relationship that the other type cannot represent. The classic BN model is the topic model, which discovers the abstract "topics" that occur in a collection of documents.

The inference algorithms of BN include stochastic MCMC simulation, loopy belief propagation, generalized belief propagation, and variational methods.

8.4.3 Learning Graphical Models from Temporal Data

Our works do not consider the temporal dependencies among variables. There exist many other works considering the temporal relationship. For example, a few recent papers have considered exploring multiple sGGMs by modeling relationships among networks; e.g., [34] [35]. [35] estimates the sparse Gaussian Graphical model at the first time step.

8.4.4 Discrete Markov Random Field

The discrete Markov Random Field (MRF) [76] is similar to the sparse Gaussian Graphical Model. However, they consider the random vector $\mathbf{X}$ as a binary random vector $\mathbf{X} \in \{0,1\}^p$. The density function of $\mathbf{X}$ can be reformulated as:

$$q_\theta(x_1, \ldots, x_p) = \exp\{\sum_{s \in V} \theta_s x_s + \sum_{s,t \in E} \theta_s t x_s x_t - \Phi(\theta)\} \tag{8.4.2}$$

The Graph structure of discrete MRF can be estimated by estimation of a generalized covariance matrix [76].

Bibliography

[1] TS Keshava Prasad, Renu Goel, Kumaran Kandasamy, Shivakumar Keerthikumar, Sameer Kumar, Suresh Mathivanan, Deepthi Telikicherla, Rajesh Raju, Beema Shafreen, Abhilash Venugopal, et al. Human protein reference database?2009 update. *Nucleic acids research*, 37(suppl 1):D767–D772, 2009.

[2] Sandra Orchard, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H Campbell, Gayatri Chavali, Carol Chen, Noemi Del-Toro, et al. The MIntAct project IntAct as a common curation platform for 11 molecular interaction databases. *Nucleic acids research*, page gkt1115, 2013.

[3] ENCODE Project Consortium et al. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57–74, 2012.

[4] Trey Ideker and Nevan J Krogan. Differential network biology. *Molecular systems biology*, 8(1):565, 2012.

[5] Tim Beck, Robert K Hastings, Sirisha Gollapudi, Robert C Free, and Anthony J Brookes. Gwas central: a comprehensive resource for the comparison and interrogation of genome-wide association studies. *European Journal of Human Genetics*, 22(7):949–952, 2014.

[6] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[7] Kantilal Varichand Mardia, John T Kent, and John M Bibby. Multivariate analysis. 1980.

[8] Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

[9] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[10] Sihai Dave Zhao, T Tony Cai, and Hongzhe Li. Direct estimation of differential networks. *Biometrika*, page asu009, 2014.

[11] Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2013.

[12] Karthik Mohan, Palma London, Maryam Fazel, Su-In Lee, and Daniela Witten. Node-based learning of multiple gaussian graphical models. *arXiv preprint arXiv:1303.5145*, 2013.

[13] Julien Chiquet, Yves Grandvalet, and Christophe Ambroise. Inferring multiple graphical structures. *Statistics and Computing*, 21(4):537–553, 2011.

[14] Jean Honorio and Dimitris Samaras. Multi-task learning of gaussian graphical models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 447–454, 2010.

[15] Jian Guo, Elizaveta Levina, George Michailidis, and Ji Zhu. Joint estimation of multiple graphical models. *Biometrika*, page asq060, 2011.

[16] Bai Zhang and Yue Wang. Learning structural changes of gaussian graphical models in controlled experiments. *arXiv preprint arXiv:1203.3532*, 2012.

[17] Yi Zhang and Jeff G Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, pages 2550–2558, 2010.

[18] Yunzhang Zhu, Xiaotong Shen, and Wei Pan. Structural pursuit over multiple undirected graphs. *Journal of the American Statistical Association*, 109(508):1683–1696, 2014.

[19] Eunho Yang, Aurelie C Lozano, and Pradeep Ravikumar. Elementary estimators for graphical models. In *Advances in Neural Information Processing Systems*, pages 2159–2167, 2014.

[20] Chandan Singh, Beilun Wang, and Yanjun Qi. A constrained, weighted-l1 minimization approach for joint discovery of heterogeneous neural connectivity graphs. *arXiv preprint arXiv:1709.04090*, 2017.

[21] Eugene Belilovsky, Gaël Varoquaux, and Matthew B Blaschko. Testing for differences in gaussian graphical models: applications to brain connectivity. In *Advances in Neural Information Processing Systems*, pages 595–603, 2016.

[22] Steffen L Lauritzen. *Graphical models*, volume 17. Clarendon Press, 1996.

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

[24] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[25] Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, pages 1436–1462, 2006.

[26] Tony Cai, Weidong Liu, and Xi Luo. A constrained l1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

[27] Sahand Negahban, Bin Yu, Martin J Wainwright, and Pradeep K Ravikumar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009.

[28] Martin J Wainwright and Michael I Jordan. Log-determinant relaxation for approximate inference in discrete markov random fields. *Signal Processing, IEEE Transactions on*, 54(6):2099–2109, 2006.

[29] Fang Han, Han Liu, and Brian Caffo. Sparse median graphs estimation in a high dimensional semiparametric model. *arXiv preprint arXiv:1310.3223*, 2013.

[30] Satoshi Hara and Takashi Washio. Learning a common substructure of multiple graphical gaussian models. *Neural Networks*, 38:23–38, 2013.

[31] Ricardo Pio Monti, Christoforos Anagnostopoulos, and Giovanni Montana. Learning population and subject-specific brain connectivity networks via mixed neighborhood selection. *arXiv preprint arXiv:1512.01947*, 2015.

[32] Song Liu, John A Quinn, Michael U Gutmann, Taiji Suzuki, and Masashi Sugiyama. Direct learning of sparse changes in markov networks by density ratio estimation. *Neural computation*, 26(6):1169–1197, 2014.

[33] Song Liu, Makoto Yamada, Nigel Collier, and Masashi Sugiyama. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83, 2013.

[34] Mladen Kolar, Le Song, Amr Ahmed, Eric P Xing, et al. Estimating time-varying networks. *The Annals of Applied Statistics*, 4(1):94–123, 2010.

[35] Huitong Qiu, Fang Han, Han Liu, and Brian Caffo. Joint estimation of multiple graphical models from high dimensional time series. *arXiv preprint arXiv: 1311.0219*, 2013.

[36] Beilun Wang, Ritambhara Singh, and Yanjun Qi. A constrained l1 minimization approach for estimating multiple sparse gaussian or nonparanormal graphical models. *Machine Learning*, 106(9-10):1381–1417, 2017.

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

[38] Eunho Yang, Aurelie Lozano, and Pradeep Ravikumar. Elementary estimators for high-dimensional linear regression. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 388–396, 2014.

[39] Eunho Yang, Aurelie C Lozano, and Pradeep Ravikumar. Elementary estimators for sparse covariance matrices and other structured moments. In *ICML*, pages 397–405, 2014.

[40] Eunho Yang and Pradeep K Ravikumar. Dirty statistical models. In *Advances in Neural Information Processing Systems*, pages 611–619, 2013.

[41] Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[42] Anthony Man-Cho So Zirui Zhou, Qi Zhang. Error bounds and convergence rate analysis of first-order methods. In *Proceedings of the 32th International Conference on Machine learning*, 2015.

[43] Peter J Bickel and Elizaveta Levina. Covariance regularization by thresholding. *The Annals of Statistics*, pages 2577–2604, 2008.

[44] Adam J Rothman, Elizaveta Levina, and Ji Zhu. Generalized thresholding of large covariance matrices. *Journal of the American Statistical Association*, 104(485):177–186, 2009.

[45] Pradeep Ravikumar, Martin J Wainwright, Garvesh Raskutti, Bin Yu, et al. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

[46] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

[47] Adam J Rothman, Peter J Bickel, Elizaveta Levina, Ji Zhu, et al. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

[48] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011.

[49] Dondapati Chowdary, Jessica Lathrop, Joanne Skelton, Kathleen Curtin, Thomas Briggs, Yi Zhang, Jack Yu, Yixin Wang, and Abhijit Mazumder. Prognostic gene expression signatures can be measured in tissues collected in rnalater preservative. *The journal of molecular diagnostics*, 8(1):31–39, 2006.

[50] Michael E Burczynski, Ron L Peterson, Natalie C Twine, Krystyna A Zuberek, Brendan J Brodeur, Lori Casciotti, Vasu Maganti, Padma S Reddy, Andrew Strahs, Fred Immermann, et al. Molecular classification of crohn's disease and ulcerative colitis patients using transcriptional profiles in peripheral blood mononuclear cells. *The journal of molecular diagnostics*, 8(1):51–61, 2006.

[51] Erming Tian, Fenghuang Zhan, Ronald Walker, Erik Rasmussen, Yupo Ma, Bart Barlogie, and John D Shaughnessy Jr. The role of the wnt-signaling antagonist dkk1 in the development of osteolytic lesions in multiple myeloma. *New England Journal of Medicine*, 349(26):2483–2494, 2003.

[52] Clare Kelly, Bharat B Biswal, R Cameron Craddock, F Xavier Castellanos, and Michael P Milham. Characterizing variation in the functional connectome: promise and pitfalls. *Trends in cognitive sciences*, 16(3):181–188, 2012.

[53] Beilun Wang, Ji Gao, and Yanjun Qi. A fast and scalable joint estimator for learning multiple related sparse gaussian graphical models. In *Artificial Intelligence and Statistics*, pages 1168–1177, 2017.

[54] Yunqi Bu and Johannes Lederer. Integrating additional knowledge into estimation of graphical models. *arXiv preprint arXiv:1704.02739*, 2017.

[55] Teppei Shimamura, Seiya Imoto, Rui Yamaguchi, and Satoru Miyano. Weighted lasso in graphical gaussian modeling for large gene network estimation based on microarray data. In *Genome Informatics 2007: Genome Informatics Series Vol. 19*, pages 142–153. World Scientific, 2007.

[56] Trevor Park and George Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103(482):681–686, 2008.

[57] Cancer Genome Atlas Research Network et al. Integrated genomic analyses of ovarian carcinoma. *Nature*, 474(7353):609–615, 2011.

[58] Matthew N McCall, Karan Uppal, Harris A Jaffee, Michael J Zilliox, and Rafael A Irizarry. The gene expression barcode: leveraging public data repositories to begin cataloging the human and murine transcriptomes. *Nucleic acids research*, 39(suppl 1):D1011–D1015, 2011.

[59] Brad T Sherman Da Wei Huang and Richard A Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature protocols*, 4(1):44–57, 2008.

[60] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. Biogrid: a general repository for interaction datasets. *Nucleic acids research*, 34(suppl_1):D535–D539, 2006.

[61] Adriana Di Martino, Chao-Gan Yan, Qingyang Li, Erin Denio, Francisco X Castellanos, Kaat Alaerts, Jeffrey S Anderson, Michal Assaf, Susan Y Bookheimer, Mirella Dapretto,

et al. The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. *Molecular psychiatry*, 19(6):659–667, 2014.

[62] David C Van Essen, Stephen M Smith, Deanna M Barch, Timothy EJ Behrens, Essa Yacoub, Kamil Ugurbil, WU-Minn HCP Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[63] Cameron Craddock. Preprocessed connectomes project: open sharing of preprocessed neuroimaging data and derivatives. In *61st Annual Meeting*. AACAP, 2014.

[64] C Craddock, S Sikka, B Cheung, R Khanuja, SS Ghosh, C Yan, Q Li, D Lurie, J Vogelstein, R Burns, et al. Towards automated analysis of connectomes: The configurable pipeline for the analysis of connectomes (c-pac). *Front Neuroinform*, 42, 2013.

[65] Nico UF Dosenbach, Binyam Nardos, Alexander L Cohen, Damien A Fair, Jonathan D Power, Jessica A Church, Steven M Nelson, Gagan S Wig, Alecia C Vogel, Christina N Lessov-Schlaggar, et al. Prediction of individual brain maturity using fmri. *Science*, 329(5997):1358–1361, 2010.

[66] Russell A Poldrack, Paul C Fletcher, Richard N Henson, Keith J Worsley, Matthew Brett, and Thomas E Nichols. Guidelines for reporting an fmri study. *Neuroimage*, 40(2):409–414, 2008.

[67] Gaël Varoquaux, Alexandre Gramfort, Jean-Baptiste Poline, and Bertrand Thirion. Brain covariance selection: better individual functional connectivity models using population prior. In *Advances in neural information processing systems*, pages 2334–2342, 2010.

[68] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[69] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

[70] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. 393(6684):440–442.

[71] Song Liu, Kenji Fukumizu, and Taiji Suzuki. Learning sparse structural changes in high-dimensional markov networks. *Behaviormetrika*, 44(1):265–286, 2017.

[72] Brian D Ripley. *Stochastic simulation*, volume 316. John Wiley & Sons, 2009.

[73] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[74] Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *Journal of Machine Learning Research*, 10(Oct):2295–2328, 2009.

[75] Han Liu, Fang Han, and Cun-hui Zhang. Transelliptical graphical models. In *Advances in Neural Information Processing Systems*, pages 809–817, 2012.

[76] Po-Ling Loh, Martin J Wainwright, et al. Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses. *The Annals of Statistics*, 41(6):3022–3049, 2013.

[77] Russell A Poldrack, Deanna M Barch, Jason Mitchell, Tor Wager, Anthony D Wagner, Joseph T Devlin, Chad Cumba, Oluwasanmi Koyejo, and Michael Milham. Toward open

sharing of task-based fmri data: the openfmri project. *Frontiers in neuroinformatics*, 7:12, 2013.

[78] Satoshi Hara and Takashi Washio. Common substructure learning of multiple graphical gaussian models. In *Machine learning and knowledge discovery in databases*, pages 1–16. Springer, 2011.

[79] Martin Renqiang Min, Xia Ning, Chao Cheng, and Mark Gerstein. Interpretable sparse high-order boltzmann machines. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 614–622, 2014.

[80] Theodoros Evgeniou, Charles A Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *The Journal of Machine Learning Research*, 6:615–637, 2005.

[81] Fang Han, Tuo Zhao, and Han Liu. Coda: High dimensional copula discriminant analysis. *Journal of Machine Learning Research*, 2012.

[82] Michael E Tipping and Christopher M Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999.

[83] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

[84] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[85] Cho-Jui Hsieh, Matyas A Sustik, Inderjit S Dhillon, and Pradeep D Ravikumar. Sparse inverse covariance matrix estimation using quadratic approximation. In *NIPS*, pages 2330–2338, 2011.

[86] Patrick L Harrington Jr and Alfred O Hero III. Spatio-temporal graphical model selection. *arXiv preprint arXiv:1004.2304*, 2010.

[87] Yunlong He, Yanjun Qi, Koray Kavukcuoglu, and Haesun Park. Learning the dependency structure of latent factors. In *NIPS*, pages 2375–2383, 2012.

[88] Nick Martin and Hermine Maes. *Multivariate analysis*. Academic press, 1979.

[89] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

[90] Emmanuel Candes and Terence Tao. The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

[91] Haotian Pang, Han Liu, and Robert Vanderbei. The fastclime package for linear programming and large-scale precision matrix estimation in r. *Journal of Machine Learning Research*, 15:489–493, 2014.

[92] ENCODE Project Consortium et al. A user's guide to the encyclopedia of dna elements (encode). *PLoS biology*, 9(4):e1001046, 2011.

[93] Mark B Gerstein, Anshul Kundaje, Manoj Hariharan, Stephen G Landt, Koon-Kiu Yan, Chao Cheng, Xinmeng Jasmine Mu, Ekta Khurana, Joel Rozowsky, Roger Alexander, et al. Architecture of the human regulatory network derived from encode data. *Nature*, 489(7414):91–100, 2012.

[94] Chao Cheng, Koon-Kiu Yan, Woochang Hwang, Jiang Qian, Nitin Bhardwaj, Joel Rozowsky, Zhi John Lu, Wei Niu, Pedro Alves, Masaomi Kato, et al. Construction and analysis of an integrated regulatory network derived from high-throughput sequencing data. *PLoS computational biology*, 7(11):e1002190, 2011.

[95] Da Wei Huang, Brad T Sherman, and Richard A Lempicki. Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic acids research*, 37(1):1–13, 2009.

[96] S.I. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of markov networks using l1 regularization. In *In NIPS*. Citeseer, 2006.

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

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

[99] M.J. Wainwright, P. Ravikumar, and J.D. Lafferty. High-dimensional graphical model selection using $\tilde{l1}$-regularized logistic regression. *Advances in neural information processing systems*, 19:1465, 2007.

[100] M. Schmidt, A. Niculescu-Mizil, and K. Murphy. Learning graphical model structure using l1-regularization paths. In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, volume 22, page 1278. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[101] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968.

[102] J.H. Friedman and B.E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, pages 916–954, 2008.

[103] D. Heckerman, D.M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *The Journal of Machine Learning Research*, 1:49–75, 2001.

[104] M. Schmidt, K. Murphy, G. Fung, and R. Rosales. Structure learning in random fields for heart motion abnormality detection. *CVPR. IEEE Computer Society*, 2008.

[105] J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. *The Journal of Machine Learning Research*, 999888:3371–3412, 2011.

[106] M. Schmidt. *Graphical Model Structure Learning with l1-Regularization*. PhD thesis, UNIVERSITY OF BRITISH COLUMBIA, 2010.

[107] M. Schmidt and K. Murphy. Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[108] David Buchman, Mark Schmidt, Shakir Mohamed, David Poole, and Nando De Freitas. On sparse, spectral and other parameterizations of binary probabilistic models. In *Artificial Intelligence and Statistics*, pages 173–181, 2012.

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

[110] C. Dahinden, G. Parmigiani, M.C. Emerick, and P. Bühlmann. Penalized likelihood for sparse contingency tables with an application to full-length cdna libraries. *BMC bioinformatics*, 8(1):476, 2007.

[111] Shilin Ding, Grace Wahba, and Xiaojin (Jerry) Zhu. Learning higher-order graph structure with features by structure penalty. In *NIPS*, pages 253–261, 2011.

[112] Han Liu, Xi Chen, John Lafferty, and Larry Wasserman. Graph-valued regression. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1423–1431. 2010.

[113] S. Zhou, J. Lafferty, and L. Wasserman. Time varying undirected graphs. *Arxiv preprint arXiv:0802.2758*, 2008.

[114] J.K. Bradley and C. Guestrin. Learning tree conditional random fields. In *Proceedings of the 27th International Conference on Machine learning*, pages 127–134, 2010.

[115] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166–176, 2003.

[116] N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science's STKE*, 303(5659):799, 2004.

[117] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. *Journal of computational biology*, 7(3-4):601–620, 2000.

[118] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with cuda. *Queue*, 6(2):40–53, 2008.

[119] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*, pages 583–598, 2014.

[120] Sorathan Chaturapruek, John C Duchi, and Christopher Ré. Asynchronous stochastic convex optimization: the noise is in the noise and sgd don't care. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1531–1539. Curran Associates, Inc., 2015.

[121] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2719–2727. Curran Associates, Inc., 2015.

[122] Padhraic Smyth, Max Welling, and Arthur U. Asuncion. Asynchronous distributed learning of topic models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 81–88. Curran Associates, Inc., 2009.

[123] Brooks Paige, Frank Wood, Arnaud Doucet, and Yee Whye Teh. Asynchronous anytime sequential monte carlo. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3410–3418. Curran Associates, Inc., 2014.

[124] Karl W Broman and Terence P Speed. A model selection approach for the identification of quantitative trait loci in experimental crosses. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):641–656, 2002.

[125] Elías Moreno, F Javier Girón, and George Casella. Consistency of objective bayes factors as the model dimension grows. *The Annals of Statistics*, pages 1937–1952, 2010.

[126] Yuhong Yang and Andrew R Barron. An asymptotic property of model selection criteria. *Information Theory, IEEE Transactions on*, 44(1):95–116, 1998.

[127] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[128] Pan Xu and Quanquan Gu. Semiparametric differential graph models. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1064–1072. Curran Associates, Inc., 2016.

[129] Beilun Wang, Ritambhara Singh, and Yanjun Qi. A constrained l1 minimization approach for estimating multiple sparse gaussian or nonparanormal graphical models. *Machine Learning*, 2017.

[130] Alexandre Abraham, Michael P Milham, Adriana Di Martino, R Cameron Craddock, Dimitris Samaras, Bertrand Thirion, and Gael Varoquaux. Deriving reproducible biomarkers from multi-site resting-state data: An autism-based example. *NeuroImage*, 147:736–745, 2017.

[131] Sarah Parisot, Sofia Ira Ktena, Enzo Ferrante, Matthew Lee, Ricardo Guerrerro Moreno, Ben Glocker, and Daniel Rueckert. Spectral graph convolutions on population graphs for disease prediction. *arXiv preprint arXiv:1703.03020*, 2017.

[132] Gaël Varoquaux, Alexandre Gramfort, Jean Baptiste Poline, and Bertrand Thirion. Markov models for fmri correlation structure: is brain functional connectivity small world, or decomposable into networks? *Journal of physiology-Paris*, 106(5):212–221, 2012.

[133] Nicole J Rinehart, Bruce J Tonge, Robert Iansek, Jenny McGinley, Avril V Brereton, Peter G Enticott, and John L Bradshaw. Gait function in newly diagnosed children with autism: cerebellar and basal ganglia related motor disorder. *Developmental Medicine & Child Neurology*, 48(10):819–824, 2006.

[134] Peter Mundy. Annotation: The neural basis of social impairments in autism: the role of the dorsal medial-frontal cortex and anterior cingulate system. *Journal of Child Psychology and psychiatry*, 44(6):793–809, 2003.

[135] Vladimir L Cherkassky, Rajesh K Kana, Timothy A Keller, and Marcel Adam Just. Functional connectivity in a baseline resting-state network in autism. *Neuroreport*, 17(16):1687–1690, 2006.

[136] Baxter P Rogers, Victoria L Morgan, Allen T Newton, and John C Gore. Assessing functional connectivity in the human brain by fmri. *Magnetic resonance imaging*, 25(10):1347–1357, 2007.

[137] Michael D Greicius, Ben Krasnow, Allan L Reiss, and Vinod Menon. Functional connectivity in the resting brain: a network analysis of the default mode hypothesis. *Proceedings of the National Academy of Sciences*, 100(1):253–258, 2003.

[138] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM, 2004.

[139] Haotian Pang, Han Liu, and Robert J Vanderbei. The fastclime package for linear programming and large-scale precision matrix estimation in r. *Journal of Machine Learning Research*, 15(1):489–493, 2014.

[140] Emmanuel Candes and Terence Tao. The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

[141] L Grosenick, B Klingenberg, B Knutson, and JE Taylor. A family of interpretable multivariate models for regression and classification of whole-brain fmri data. *arXiv preprint arXiv:1110.4139*, 2011.

[142] Bernard Ng and Rafeef Abugharbieh. Generalized sparse regularization with application to fmri brain decoding. In *Biennial International Conference on Information Processing in Medical Imaging*, pages 612–623. Springer, 2011.

[143] Esteban Real, Hiroki Asari, Tim Gollisch, and Markus Meister. Neural circuit inference from function to structure. *Current Biology*, 2017.

[144] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted 1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

[145] Bertrand Thirion, Gaël Varoquaux, Elvis Dohmatob, and Jean-Baptiste Poline. Which fmri clustering gives good brain parcellations? *Frontiers in neuroscience*, 8:167, 2014.

[146] R Cameron Craddock, Rosalia L Tungaraza, and Michael P Milham. Connectomics and new approaches for analyzing human brain functional connectivity. *GigaScience*, 4(1):13, 2015.

[147] Bernard Ng, Gaël Varoquaux, Jean Baptiste Poline, and Bertrand Thirion. A novel sparse group gaussian graphical model for functional connectivity estimation. In *International Conference on Information Processing in Medical Imaging*, pages 256–267. Springer, 2013.

[148] Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Muller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gäel Varoquaux. Machine learning for neuroimaging with scikit-learn. *arXiv preprint arXiv:1412.3919*, 2014.

[149] Qiang Liu and Alexander T Ihler. Learning scale free networks by reweighted l1 regularization. In *AISTATS*, pages 40–48, 2011.

[150] Petra E Vértes, Aaron F Alexander-Bloch, Nitin Gogtay, Jay N Giedd, Judith L Rapoport, and Edward T Bullmore. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences*, 109(15):5868–5873, 2012.

[151] Stephen M Smith, Diego Vidaurre, Christian F Beckmann, Matthew F Glasser, Mark Jenkinson, Karla L Miller, Thomas E Nichols, Emma C Robinson, Gholamreza Salimi-Khorshidi, Mark W Woolrich, et al. Functional connectomics from resting-state fmri. *Trends in cognitive sciences*, 17(12):666–682, 2013.

[152] Christopher Baldassano, Marius Cătălin Iordan, Diane M Beck, and Li Fei-Fei. Voxel-level functional connectivity using spatial regularization. *Neuroimage*, 63(3):1099–1106, 2012.

[153] Lucina Q Uddin, Kaustubh Supekar, Charles J Lynch, Amirah Khouzam, Jennifer Phillips, Carl Feinstein, Srikanth Ryali, and Vinod Menon. Salience network–based classification and prediction of symptom severity in children with autism. *JAMA psychiatry*, 70(8):869–879, 2013.

[154] D Koller, N Freedman, L Getoor, and B Taskar. Graphical models in a nutshell in introduction to statistical relational learning. *Stanford*, 2007.

[155] Jianqing Fan and Han Liu. Statistical analysis of big data on pharmacogenomics. *Advanced drug delivery reviews*, 65(7):987–1000, 2013.

[156] Beilun Wang, Ritambhara Singh, and Yanjun Qi. A constrained l1 minimization approach for estimating multiple sparse gaussian or nonparanormal graphical models. *arXiv preprint arXiv:1605.03468*, 2016.

[157] Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.

[158] H Sebastian Seung. Neuroscience: towards functional connectomics. *Nature*, 471(7337):170–172, 2011.

[159] Stephen M Smith, Christian F Beckmann, Jesper Andersson, Edward J Auerbach, Janine Bijsterbosch, Gwenaëlle Douaud, Eugene Duff, David A Feinberg, Ludovica Griffanti, Michael P Harms, et al. Resting-state fmri in the human connectome project. *Neuroimage*, 80:144–168, 2013.

[160] Michael P Milham, Damien Fair, Maarten Mennes, Stewart HMD Mostofsky, et al. The adhd-200 consortium: a model to advance the translational potential of neuroimaging in clinical neuroscience. *Frontiers in systems neuroscience*, 6:62, 2012.

[161] Shlomi Haar, Sigal Berman, Marlene Behrmann, and Ilan Dinstein. Anatomical abnormalities in autism? *Cerebral Cortex*, page bhu242, 2014.

[162] Jared A Nielsen, Brandon A Zielinski, P Thomas Fletcher, Andrew L Alexander, Nicholas Lange, Erin D Bigler, Janet E Lainhart, and Jeffrey S Anderson. Multisite functional connectivity mri classification of autism: Abide results. *Frontiers in Human Neuroscience*, 7, 2013.

[163] Sina Ghiassian, Russell Greiner, Ping Jin, and M Brown. Learning to classify psychiatric disorders based on fmr images: Autism vs healthy and adhd vs healthy. In *Proceedings of 3rd NIPS Workshop on Machine Learning and Interpretation in NeuroImaging*, 2013.

[164] Tetsuya Iidaka. Resting state functional magnetic resonance imaging and neural network classified autism and control. *Cortex*, 63:55–67, 2015.

[165] Heng Chen, Xujun Duan, Feng Liu, Fengmei Lu, Xujing Ma, Youxue Zhang, Lucina Q Uddin, and Huafu Chen. Multivariate classification of autism spectrum disorder using frequency-specific resting-state functional connectivity—a multi-center study. *Progress in Neuro-Psychopharmacology and Biological Psychiatry*, 64:1–9, 2016.

[166] Mark Plitt, Kelly Anne Barnes, and Alex Martin. Functional connectivity classification of autism identifies highly predictive brain features but falls short of biomarker standards. *NeuroImage: Clinical*, 7:359–366, 2015.

[167] Colleen P Chen, Christopher L Keown, Afrooz Jahedi, Aarti Nair, Mark E Pflieger, Barbara A Bailey, and Ralph-Axel Müller. Diagnostic classification of intrinsic functional connectivity highlights somatosensory, default mode, and visual regions in autism. *NeuroImage: Clinical*, 8:238–245, 2015.

[168] Thierry Bouwmans, Necdet Serhat Aybat, and El-hadi Zahzah. Handbook of robust low-rank and sparse matrix decomposition: Applications in image and video processing, 2016.

[169] Diane Bouchacourt, Pawan K Mudigonda, and Sebastian Nowozin. DISCO nets : DISsimilarity COefficients networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 352–360. Curran Associates, Inc.

[170] Satoshi Hara and Takashi Washio. Learning a common substructure of multiple graphical gaussian models. 38:23–38.

[171] Gael Varoquaux, Alexandre Gramfort, Jean-baptiste Poline, and Bertrand Thirion. Brain covariance selection: better individual functional connectivity models using population prior. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2334–2342. Curran Associates, Inc.

[172] Russell Lyons et al. Distance covariance in metric spaces. *The Annals of Probability*, 41(5):3284–3305, 2013.

[173] Aaron F. Alexander-Bloch, Petra E. Vértes, Reva Stidd, François Lalonde, Liv Clasen, Judith Rapoport, Jay Giedd, Edward T. Bullmore, and Nitin Gogtay. The anatomical distance of functional connections predicts brain network topology in health and schizophrenia. 23(1):127–138.

[174] Parietal. Covariance and graphical models of brain connectivity, 2016.

[175] Yunqi Bu and Johannes Lederer. Integrating additional knowledge into estimation of graphical models.

[176] Nicole J Rinehart, Bruce J Tonge, Robert Iansek, Jenny McGinley, Avril V Brereton, Peter G Enticott, and John L Bradshaw. Gait function in newly diagnosed children with autism: cerebellar and basal ganglia related motor disorder. 48(10):819–824.

[177] Peter Mundy. Annotation: The neural basis of social impairments in autism: the role of the dorsal medial-frontal cortex and anterior cingulate system. 44(6):793–809.

[178] Vladimir L Cherkassky, Rajesh K Kana, Timothy A Keller, and Marcel Adam Just. Functional connectivity in a baseline resting-state network in autism. 17(16):1687–1690.

[179] David C Van Essen, Stephen M Smith, Deanna M Barch, Timothy EJ Behrens, Essa Yacoub, Kamil Ugurbil, WU-Minn HCP Consortium, and others. The WU-minn human connectome project: an overview. 80:62–79.

[180] Baxter P Rogers, Victoria L Morgan, Allen T Newton, and John C Gore. Assessing functional connectivity in the human brain by fMRI. 25(10):1347–1357.

[181] Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. 10:2295–2328.

[182] Michael D Greicius, Ben Krasnow, Allan L Reiss, and Vinod Menon. Functional connectivity in the resting brain: a network analysis of the default mode hypothesis. 100(1):253–258.

[183] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM.

[184] Haotian Pang, Han Liu, and Robert J Vanderbei. The fastclime package for linear programming and large-scale precision matrix estimation in r. 15(1):489–493.

[185] Emmanuel Candes and Terence Tao. The dantzig selector: Statistical estimation when p is much larger than n. pages 2313–2351.

[186] Thomas H Cormen. *Introduction to algorithms*. MIT press.

[187] L Grosenick, B Klingenberg, B Knutson, and JE Taylor. A family of interpretable multivariate models for regression and classification of whole-brain fMRI data.

[188] Bernard Ng and Rafeef Abugharbieh. Generalized sparse regularization with application to fMRI brain decoding. In *Biennial International Conference on Information Processing in Medical Imaging*, pages 612–623. Springer.

[189] Esteban Real, Hiroki Asari, Tim Gollisch, and Markus Meister. Neural circuit inference from function to structure.

[190] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted 1 minimization. 14(5):877–905.

[191] Teppei Shimamura, Seiya Imoto, Rui Yamaguchi, and Satoru Miyano. Weighted lasso in graphical gaussian modeling for large gene network estimation based on microarray data. 19:142–153.

[192] Cameron Craddock. Preprocessed connectomes project: open sharing of preprocessed neuroimaging data and derivatives. In *61st Annual Meeting*. AACAP.

[193] Bertrand Thirion, Gaël Varoquaux, Elvis Dohmatob, and Jean-Baptiste Poline. Which fMRI clustering gives good brain parcellations? 8:167.

[194] R Cameron Craddock, Rosalia L Tungaraza, and Michael P Milham. Connectomics and new approaches for analyzing human brain functional connectivity. 4(1):13.

[195] Sihai Dave Zhao, T Tony Cai, and Hongzhe Li. Direct estimation of differential networks. page asu009.

[196] Bernard Ng, Gaël Varoquaux, Jean Baptiste Poline, and Bertrand Thirion. A novel sparse group gaussian graphical model for functional connectivity estimation. In *International Conference on Information Processing in Medical Imaging*, pages 256–267. Springer.

[197] Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Muller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gäel Varoquaux. Machine learning for neuroimaging with scikit-learn.

[198] Qiang Liu and Alexander T Ihler. Learning scale free networks by reweighted l1 regularization. In *AISTATS*, pages 40–48.

[199] Tony Cai, Weidong Liu, and Xi Luo. A constrained 1 minimization approach to sparse precision matrix estimation. 106(494):594–607.

[200] Petra E Vértes, Aaron F Alexander-Bloch, Nitin Gogtay, Jay N Giedd, Judith L Rapoport, and Edward T Bullmore. Simple models of human brain functional networks. 109(15):5868–5873.

[201] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. 9(3):432–441.

[202] Stephen M Smith, Diego Vidaurre, Christian F Beckmann, Matthew F Glasser, Mark Jenkinson, Karla L Miller, Thomas E Nichols, Emma C Robinson, Gholamreza Salimi-Khorshidi, Mark W Woolrich, and others. Functional connectomics from resting-state fMRI. 17(12):666–682.

[203] Christopher Baldassano, Marius Cătălin Iordan, Diane M Beck, and Li Fei-Fei. Voxel-level functional connectivity using spatial regularization. 63(3):1099–1106.

[204] Ricardo Pio Monti, Christoforos Anagnostopoulos, and Giovanni Montana. Learning population and subject-specific brain connectivity networks via mixed neighborhood selection.

[205] Lucina Q Uddin, Kaustubh Supekar, Charles J Lynch, Amirah Khouzam, Jennifer Phillips, Carl Feinstein, Srikanth Ryali, and Vinod Menon. Salience network–based classification and prediction of symptom severity in children with autism. 70(8):869–879.

[206] Steffen L Lauritzen. *Graphical models*, volume 17. Clarendon Press.

[207] D Koller, N Freedman, L Getoor, and B Taskar. Graphical models in a nutshell in introduction to statistical relational learning.

[208] Jianqing Fan and Han Liu. Statistical analysis of big data on pharmacogenomics. 65(7):987–1000.

[209] Beilun Wang, Ritambhara Singh, and Yanjun Qi. A constrained l1 minimization approach for estimating multiple sparse gaussian or nonparanormal graphical models.

[210] Julien Chiquet, Yves Grandvalet, and Christophe Ambroise. Inferring multiple graphical structures. 21(4):537–553.

[211] Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. 76(2):373–397.

[212] Nico UF Dosenbach, Binyam Nardos, Alexander L Cohen, Damien A Fair, Jonathan D Power, Jessica A Church, Steven M Nelson, Gagan S Wig, Alecia C Vogel, Christina N Lessov-Schlaggar, and others. Prediction of individual brain maturity using fMRI. 329(5997):1358–1361.

[213] C Craddock, S Sikka, B Cheung, R Khanuja, SS Ghosh, C Yan, Q Li, D Lurie, J Vogelstein, R Burns, and others. Towards automated analysis of connectomes: The configurable pipeline for the analysis of connectomes (c-pac). 42.

[214] Adriana Di Martino, Chao-Gan Yan, Qingyang Li, Erin Denio, Francisco X Castellanos, Kaat Alaerts, Jeffrey S Anderson, Michal Assaf, Susan Y Bookheimer, Mirella Dapretto, and others. The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. 19(6):659–667.

[215] H Sebastian Seung. Neuroscience: towards functional connectomics. 471(7337):170–172.

[216] Stephen M Smith, Christian F Beckmann, Jesper Andersson, Edward J Auerbach, Janine Bijsterbosch, Gwenaëlle Douaud, Eugene Duff, David A Feinberg, Ludovica Griffanti,

Michael P Harms, and others. Resting-state fMRI in the human connectome project. 80:144–168.

[217] Michael P Milham, Damien Fair, Maarten Mennes, Stewart HMD Mostofsky, and others. The ADHD-200 consortium: a model to advance the translational potential of neuroimaging in clinical neuroscience. 6:62.

[218] Shlomi Haar, Sigal Berman, Marlene Behrmann, and Ilan Dinstein. Anatomical abnormalities in autism? page bhu242.

[219] Jared A Nielsen, Brandon A Zielinski, P Thomas Fletcher, Andrew L Alexander, Nicholas Lange, Erin D Bigler, Janet E Lainhart, and Jeffrey S Anderson. Multisite functional connectivity MRI classification of autism: ABIDE results. 7.

[220] Sina Ghiassian, Russell Greiner, Ping Jin, and M Brown. Learning to classify psychiatric disorders based on fMR images: Autism vs healthy and ADHD vs healthy. In *Proceedings of 3rd NIPS Workshop on Machine Learning and Interpretation in NeuroImaging*.

[221] Tetsuya Iidaka. Resting state functional magnetic resonance imaging and neural network classified autism and control. 63:55–67.

[222] Heng Chen, Xujun Duan, Feng Liu, Fengmei Lu, Xujing Ma, Youxue Zhang, Lucina Q Uddin, and Huafu Chen. Multivariate classification of autism spectrum disorder using frequency-specific resting-state functional connectivity—a multi-center study. 64:1–9.

[223] Mark Plitt, Kelly Anne Barnes, and Alex Martin. Functional connectivity classification of autism identifies highly predictive brain features but falls short of biomarker standards. 7:359–366.

[224] Colleen P Chen, Christopher L Keown, Afrooz Jahedi, Aarti Nair, Mark E Pflieger, Barbara A Bailey, and Ralph-Axel Müller. Diagnostic classification of intrinsic functional connectivity highlights somatosensory, default mode, and visual regions in autism. 8:238–245.

[225] Edward T. Bullmore and Danielle S. Bassett. Brain graphs: graphical models of the human brain connectome. 7:113–140.

[226] Raymond Salvador, John Suckling, Christian Schwarzbauer, and Ed Bullmore. Undirected graphs of frequency-dependent functional connectivity in whole brain networks. 360(1457):937–946.

[227] Pedro A. Valdés-Sosa, Jose M. Sánchez-Bornot, Agustín Lage-Castellanos, Mayrim Vega-Hernández, Jorge Bosch-Bayard, Lester Melie-García, and Erick Canales-Rodríguez. Estimating brain functional connectivity with sparse multivariate autoregression. 360(1457):969–981.

[228] Karl J. Friston. Functional and effective connectivity: a review. 1(1):13–36.

[229] Shuai Huang, Jing Li, Liang Sun, Jieping Ye, Adam Fleisher, Teresa Wu, Kewei Chen, Eric Reiman, Alzheimer's Disease NeuroImaging Initiative, and others. Learning brain connectivity of alzheimer's disease by sparse inverse covariance estimation. 50(3):935–949.

[230] Karl Friston. Causal modelling and brain connectivity in functional magnetic resonance imaging. 7(2):e1000033.

[231] Ivor Cribben, Ragnheidur Haraldsdottir, Lauren Y. Atlas, Tor D. Wager, and Martin A. Lindquist. Dynamic connectivity regression: determining state-related changes in brain connectivity. 61(4):907–920.

[232] Jagath C. Rajapakse and Juan Zhou. Learning effective brain connectivity with dynamic bayesian networks. 37(3):749–760.

[233] Michael Eichler. A graphical approach for evaluating effective connectivity in neural systems. 360(1457):953–967.

[234] Shuai Huang, Jing Li, Liang Sun, Jun Liu, Teresa Wu, Kewei Chen, Adam Fleisher, Eric Reiman, and Jieping Ye. Learning brain connectivity of alzheimer's disease from neuroimaging data. In *Advances in Neural Information Processing Systems*, pages 808–816.

[235] Guillaume Marrelec, Alexandre Krainik, Hugues Duffau, Mélanie Pélégrini-Issac, Stéphane Lehéricy, Julien Doyon, and Habib Benali. Partial correlation for functional brain interactivity investigation in functional MRI. 32(1):228–237.

[236] Geoffrey JM Parker, Claudia AM Wheeler-Kingshott, and Gareth J. Barker. Estimating distributed anatomical connectivity using fast marching methods and diffusion tensor imaging. 21(5):505–512.

[237] Mikail Rubinov and Olaf Sporns. Complex network measures of brain connectivity: uses and interpretations. 52(3):1059–1069.

[238] Danielle S. Bassett and Michael S. Gazzaniga. Understanding complexity in the human brain. 15(5):200–209.

[239] Sean L. Simpson, Malaak N. Moussa, and Paul J. Laurienti. An exponential random graph modeling approach to creating group-based representative whole-brain connectivity networks. 60(2):1117–1126.

[240] David Meunier, Renaud Lambiotte, and Edward T. Bullmore. Modular and hierarchically modular organization of brain networks. 4:200.

[241] Will D. Penny, Klaas E. Stephan, Jean Daunizeau, Maria J. Rosa, Karl J. Friston, Thomas M. Schofield, and Alex P. Leff. Comparing families of dynamic causal models. 6(3):e1000709.

[242] Sean L. Simpson, Satoru Hayasaka, and Paul J. Laurienti. Exponential random graph modeling for complex brain networks. 6(5):e20039.

[243] Stephen M. Smith. The future of FMRI connectivity. 62(2):1257–1266.

[244] Vinod Menon. Large-scale brain networks and psychopathology: a unifying triple network model. 15(10):483–506.

[245] Tarek Medkour, Andrew T. Walden, and Adrian Burgess. Graphical modelling for brain connectivity via partial coherence. 180(2):374–383.

[246] Pedro A. Valdes-Sosa, Alard Roebroeck, Jean Daunizeau, and Karl Friston. Effective connectivity: influence, causality and biophysical modeling. 58(2):339–361.

[247] Aaron F. Alexander-Bloch, Nitin Gogtay, David Meunier, Rasmus Birn, Liv Clasen, Francois Lalonde, Rhoshel Lenroot, Jay Giedd, and Edward T. Bullmore. Disrupted modularity and local connectivity of brain functional networks in childhood-onset schizophrenia. 4:147.

[248] Raymond Salvador, John Suckling, Martin R. Coleman, John D. Pickard, David Menon, and E. D. Bullmore. Neurophysiological architecture of functional magnetic resonance images of human brain. 15(9):1332–1342.

[249] Elmar Wolfgang Lang, Ana Maria Tomé, Ingo R. Keck, J. M. Górriz-Sáez, and Carlos García Puntonet. Brain connectivity analysis: a short survey. 2012:8.

[250] João Ricardo Sato, Edson Amaro Junior, Daniel Yasumasa Takahashi, Marcelo de Maria Felix, Michael John Brammer, and Pedro Alberto Morettin. A method to produce evolving functional connectivity maps during the course of an fMRI experiment using wavelet-based time-varying granger causality. 31(1):187–196.

[251] Elena A. Allen, Eswar Damaraju, Sergey M. Plis, Erik B. Erhardt, Tom Eichele, and Vince D. Calhoun. Tracking whole-brain connectivity dynamics in the resting state. page bhs352.

[252] Stefan Haufe, Ryota Tomioka, Guido Nolte, Klaus-Robert Müller, and Motoaki Kawanabe. Modeling sparse connectivity between underlying brain sources for EEG/MEG. 57(8):1954–1963.

[253] Andrew Zalesky, Alex Fornito, Ian H. Harding, Luca Cocchi, Murat Yücel, Christos Pantelis, and Edward T. Bullmore. Whole-brain anatomical networks: does the choice of nodes matter? 50(3):970–983.

[254] Kaustubh Supekar, Vinod Menon, Daniel Rubin, Mark Musen, and Michael D. Greicius. Network analysis of intrinsic functional brain connectivity in alzheimer's disease. 4(6):e1000100.

[255] Guo-Rong Wu, Wei Liao, Sebastiano Stramaglia, Ju-Rong Ding, Huafu Chen, and Daniele Marinazzo. A blind deconvolution approach to recover effective connectivity brain networks from resting state fMRI data. 17(3):365–374.

[256] Liang Sun, Rinkal Patel, Jun Liu, Kewei Chen, Teresa Wu, Jing Li, Eric Reiman, and Jieping Ye. Mining brain region connectivity for alzheimer's disease study via sparse inverse covariance estimation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1335–1344. ACM.

[257] Raymond Salvador, Salvador Sarró, Jesús J. Gomar, Jordi Ortiz-Gil, Fidel Vila, Antoni Capdevila, Ed Bullmore, Peter J. McKenna, and Edith Pomarol-Clotet. Overall brain connectivity maps show cortico-subcortical abnormalities in schizophrenia. 31(12):2003–2014.

[258] Bernadette CM Van Wijk, Cornelis J. Stam, and Andreas Daffertshofer. Comparing brain networks of different size and connectivity density using graph theory. 5(10):e13701.

[259] Heike Tost, Edda Bilek, and Andreas Meyer-Lindenberg. Brain connectivity in psychiatric imaging genetics. 62(4):2250–2260.

[260] Nicolas A. Crossley, Andrea Mechelli, Petra E. Vértes, Toby T. Winton-Brown, Ameera X. Patel, Cedric E. Ginestet, Philip McGuire, and Edward T. Bullmore. Cognitive relevance of the community structure of the human brain functional coactivation network. 110(28):11583–11588.

[261] Olaf Sporns. From simple graphs to the connectome: networks in neuroimaging. 62(2):881–886.

[262] Srikanth Ryali, Tianwen Chen, Kaustubh Supekar, and Vinod Menon. Estimation of functional connectivity in fMRI data using stability selection-based sparse partial correlation with elastic net penalty. 59(4):3852–3861.

[263] Farras Abdelnour, Henning U. Voss, and Ashish Raj. Network diffusion accurately models the relationship between structural and functional brain connectivity networks. 90:335–347.

[264] Jagath C. Rajapakse, Yang Wang, Xuebin Zheng, and Juan Zhou. Probabilistic framework for brain connectivity from functional MR images. 27(6):825–833.

[265] Gopikrishna Deshpande, Stephan LaConte, George Andrew James, Scott Peltier, and Xiaoping Hu. Multivariate granger causality analysis of fMRI data. 30(4):1361–1373.

[266] Luiz A. Baccalá and Koichi Sameshima. Partial directed coherence. In *Methods in Brain Connectivity Inference through Multivariate Time Series Analysis*, pages 57–73. CRC Press.

[267] Saad Jbabdi, M. W. Woolrich, J. L. R. Andersson, and T. E. J. Behrens. A bayesian framework for global tractography. 37(1):116–129.

[268] Lei Zhang, Dimitris Samaras, Nelly Alia-Klein, Nora Volkow, and Rita Goldstein. Modeling neuronal interactivity using dynamic bayesian networks. 18:1593.

[269] Manfred G. Kitzbichler, Richard NA Henson, Marie L. Smith, Pradeep J. Nathan, and Edward T. Bullmore. Cognitive effort drives workspace configuration of human brain functional networks. 31(22):8259–8270.

[270] Kaiming Li, Lei Guo, Jingxin Nie, Gang Li, and Tianming Liu. Review of methods for functional brain connectivity detection using fMRI. 33(2):131–139.

[271] Junning Li, Z. Jane Wang, Samantha J. Palmer, and Martin J. McKeown. Dynamic bayesian network modeling of fMRI: a comparison of group-analysis methods. 41(2):398–407.

[272] Luca Cocchi, Andrew Zalesky, Alex Fornito, and Jason B. Mattingley. Dynamic cooperation and competition between brain systems during cognitive control. 17(10):493–501.

[273] Gaël Varoquaux, Alexandre Gramfort, Jean Baptiste Poline, and Bertrand Thirion. Markov models for fMRI correlation structure: is brain functional connectivity small world, or decomposable into networks? 106(5):212–221.

[274] Pedro A. Valdes-Sosa. Spatio-temporal autoregressive models defined over brain manifolds. 2(2):239–250.

[275] Bin He, Yakang Dai, Laura Astolfi, Fabio Babiloni, Han Yuan, and Lin Yang. eConnectome: A MATLAB toolbox for mapping and imaging of brain functional connectivity. 195(2):261–269.

[276] G. Marrelec, J. Kim, J. Doyon, and B. Horwitz. Large-scale neural model validation of partial correlation analysis for effective connectivity investigation in functional MRI. 30(3):941–950.

[277] Pierre-Olivier Amblard and Olivier JJ Michel. On directed information theory and granger causality graphs. 30(1):7–16.

[278] Chenhui Hu, Lin Cheng, Jorge Sepulcre, Georges El Fakhri, Yue M. Lu, and Quanzheng Li. A graph theoretical regression model for brain connectivity learning of alzheimer's disease. In *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*, pages 616–619. IEEE.

[279] Narender Ramnani, Timothy EJ Behrens, Will Penny, and Paul M. Matthews. New approaches for exploring anatomical and functional connectivity in the human brain. 56(9):613–619.

[280] Mark Fiecas and Hernando Ombao. The generalized shrinkage estimator for the analysis of functional connectivity of brain signals. pages 1102–1125.

[281] Jurriaan M. Peters, Maxime Taquet, Clemente Vega, Shafali S. Jeste, Iván Sánchez Fernández, Jacqueline Tan, Charles A. Nelson, Mustafa Sahin, and Simon K. Warfield. Brain functional networks in syndromic and non-syndromic autism: a graph theoretical study of EEG connectivity. 11(1):54.

[282] Ting Xu, Zhi Yang, Lili Jiang, Xiu-Xia Xing, and Xi-Nian Zuo. A connectome computation system for discovery science of brain. 60(1):86–95.

[283] Andrew Arnold, Yan Liu, and Naoki Abe. Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 66–75. ACM.

[284] Gaël Varoquaux and R. Cameron Craddock. Learning and comparing functional connectomes across subjects. 80:405–415.

[285] David Papo, Javier M. Buldú, Stefano Boccaletti, and Edward T. Bullmore. *Complex network theory and the brain*. The Royal Society.

[286] Karl J. Friston. Models of brain function in neuroimaging. 56:57–87.

[287] Kathleen M. Gates and Peter CM Molenaar. Group search algorithm recovers effective connectivity maps for individuals in homogeneous and heterogeneous samples. 63(1):310–319.

[288] Olaf Sporns. Structure and function of complex brain networks. 15(3):247–262.

[289] J. D. Kruschwitz, D. List, L. Waller, M. Rubinov, and H. Walter. GraphVar: A user-friendly toolbox for comprehensive graph analyses of functional brain connectivity. 245:107–115.

[290] Huitong Qiu, Fang Han, Han Liu, and Brian Caffo. Joint estimation of multiple graphical models from high dimensional time series. 78(2):487–504.

[291] Tommi S. Jaakkola and Michael I. Jordan. Variational methods for inference and estimation in graphical models.

[292] Demian Battaglia, Annette Witt, Fred Wolf, and Theo Geisel. Dynamic effective connectivity of inter-areal brain circuits. 8(3):e1002438.

[293] Xi-Nian Zuo, Ting Xu, Lili Jiang, Zhi Yang, Xiao-Yan Cao, Yong He, Yu-Feng Zang, F. Xavier Castellanos, and Michael P. Milham. Toward reliable characterization of functional homogeneity in the human brain: preprocessing, scan duration, imaging resolution and computational space. 65:374–386.

[294] Klaas Enno Stephan and Alard Roebroeck. A short history of causal modeling of fMRI data. 62(2):856–863.

[295] Anil K. Seth. A MATLAB toolbox for granger causal connectivity analysis. 186(2):262–273.

[296] Guillaume Marrelec, Pierre Bellec, and Habib Benali. Exploring large-scale brain networks in functional MRI. 100(4):171–181.

[297] Elena Rykhlevskaia, Monica Fabiani, and Gabriele Gratton. Lagged covariance structure models for studying functional connectivity in the brain. 30(4):1203–1218.

[298] Andreas Horn, Dirk Ostwald, Marco Reisert, and Felix Blankenburg. The structural–functional connectome and the default mode network of the human brain. 102:142–151.

[299] Susan Whitfield-Gabrieli and Alfonso Nieto-Castanon. Conn: a functional connectivity toolbox for correlated and anticorrelated brain networks. 2(3):125–141.

[300] Ivor Cribben, Tor Wager, and Martin Lindquist. Detecting functional connectivity change points for single-subject fMRI data. 7:143.

[301] Sen Yang, Qian Sun, Shuiwang Ji, Peter Wonka, Ian Davidson, and Jieping Ye. Structural graphical lasso for learning mouse brain connectivity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1385–1394. ACM.

[302] Kriti Puniyani and Eric P. Xing. NP-MuScL: Unsupervised global prediction of interaction networks from multiple data sources. 20(11):892–904.

[303] Mladen Kolar. Uncovering structure in high-dimensions: Networks and multi-task learning problems.

[304] Diane Oyen and Terran Lane. Leveraging domain knowledge in multitask bayesian network structure learning. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 1091–1097. AAAI Press.

[305] Yun Zhou, Norman Fenton, Timothy M. Hospedales, and Martin Neil. Probabilistic graphical models parameter learning with transferred prior and constraints. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, pages 972–981. AUAI Press.

[306] Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep K Ravikumar. A dirty model for multi-task learning. In *Advances in neural information processing systems*, pages 964–972, 2010.