Adding Extra Knowledge in Scalable Learning of Sparse Differential Gaussian Graphical Models

Arshdeep Sekhon, Beilun Wang, Yanjun Qi
Department of Computer Science
University of Virginia, Computer Science Department
http://jointnets.org/

Abstract

We focus on integrating different types of extra knowledge (other than the observed samples) for estimating the sparse structure change between two p-dimensional Gaussian Graphical Models (i.e. differential GGMs). Previous differential GGM estimators either fail to include additional knowledge or cannot scale up to a high-dimensional (large p) situation. This paper proposes a novel method KDiffNet that incorporates Additional Knowledge in identifying Differential Networks via an Elementary Estimator. We design a novel hybrid norm as a superposition of two structured norms guided by the extra edge information and the additional node group knowledge. KDiffNet is solved through a fast parallel proximal algorithm, enabling it to work in large-scale settings. KDiffNet can incorporate various combinations of existing knowledge without re-designing the optimization. Through rigorous statistical analysis we show that, while considering more evidence, KDiffNet achieves the same convergence rate as the state-of-the-art. Empirically on multiple synthetic datasets and one real-world fMRI brain data, KDiffNet significantly outperforms the cutting edge baselines with regard to the prediction performance, while achieving the same level of time cost or less.

1 Introduction

Learning the change of dependencies between random variables is an essential task in many real-world applications. For example, when analyzing functional MRI samples from different groups of human subjects, detecting the difference in brain connectivity networks can shed light on studying and designing treatments for psychiatric diseases [6]. In this paper, we consider Gaussian graphical models (GGMs) and focus on directly estimating changes in the dependency structures of two p-dimensional GGMs, based on n_c and n_d samples drawn from the two models (we call the task differential GGMs). In particular, we focus on estimating the structure change under a high-dimensional situation, where the number of variables p may exceed the number of observations: $p > \max(n_c, n_d)$. To conduct consistent estimation under high dimensional settings, we leverage the sparsity constraint. In the context of estimating structural changes between two GGMs, this translates into a differential network with few edges. We review the state-of-the-art estimators for differential GGMs in Section 2.1.

One significant caveat of previous differential GGM estimators is that little attention has been paid to incorporating extra knowledge of the nodes or of the edges. In addition to the observed samples, extra information is widely available in real-world applications. For instance, when estimating the functional connectivity networks among brain regions via fMRI measurements (i.e. observed samples), there exist considerable knowledge about the spatial and anatomical evidence of these regions. Adding such evidence will help the learned differential structure better reflect domain experts' knowledge like certain anatomical regions or spatially related regions are more likely to be connected [20].

Although being a strong evidence of structural patterns that we aim to discover, extra information has rarely been considered when estimating differential GGM from two sets of observed samples. To the authors' best knowledge, only two loosely-related studies exist in the literature: (1) One study with

Preprint. Under review.

the name NAK [2] (following ideas from [14]) proposed to integrate Additional Knowledge into the estimation of single-task graphical model using a weighted Neighbourhood selection formulation. (2) Another study with the name JEEK [18] (following [15]) considered edge-level evidence via a weighted objective formulation to estimate multiple dependency graphs from heterogeneous samples. Both studies only added edge-level extra knowledge in structural learning and neither of the approaches was designed for the direct structure estimation of differential GGM.

This paper fills the gap by proposing a novel method, namely KDiffNet , to add additional Knowledge in identifying DIFFerential networks via an Elementary Estimator. Our main objective is to make KDiffNet flexible enough to model various combinations of existing knowledge without re-designing the optimization. This is achieved by: (1) representing the edge-level domain knowledge as weights and using weights through a weighted ℓ_1 regularization constraint; and (2) describing the node-level knowledge as the variable groups and enforcing through a group norm constraint. Then KDiffNet designs a novel hybrid norm as the minimization objective and enforces the superposition of two aforementioned structured constraints. Our second main aim for KDiffNet is to achieve direct, scalable, and fast differential GGM estimations, and at the same time to guarantee the estimation error is well bounded. We achieve this goal through modeling KDiffNet in an elementary estimator based framework and solving it via parallel proximal based optimization. Briefly speaking, this paper makes the following contributions: [1]

- **Novel and Flexible:** KDiffNet is the first method to integrate different kinds of additional knowledge for structure learning of differential GGMs. KDiffNet proposes a flexible formulation to consider both the edge-level evidence and the node-group level knowledge (Section 2.3).
- **Fast and Scalable:** We optimize KDiffNet through a proximal algorithm making it scalable to large values of p. KDiffNet 's unified formulation avoids the need to design knowledge-specific optimization (Section 2.5).
- **Theoretically Sound:** We theoretically prove the convergence rate of KDiffNet as $O(\sqrt{\frac{\log p}{\min(n_c,n_d)}})$, achieving the same error bound as the state-of-the-art (Section 2.6).
- **Empirical Evaluation:** We evaluate KDiffNet using multiple synthetic datasets and one real-world task. Our experiments showcase how KDiffNet can integrate knowledge of spatial distances, known edges or anatomical grouping evidence in the proposed formulation, empirically showing its real-world adaptivity. KDiffNet improves the state-of-the-art baselines with consistently better prediction accuracy while maintaining the same or less time cost (Section 3).

2 Proposed Method: KDiffNet

2.1 Previous Estimators for Structure Change between two GGMs (Differential GGMs)

The task of estimating differential GGMs assumes we are given two sets of observed samples (in the form of two matrices) $\mathbf{X}_c \in \mathbb{R}^{n_c \times p}$ and $\mathbf{X}_d \in \mathbb{R}^{n_d \times p}$, identically and independently drawn from two normal distributions $N_p(\mu_c, \Sigma_c)$ and $N_p(\mu_d, \Sigma_d)$ respectively. Here $\mu_c, \mu_d \in \mathbb{R}^p$ describe the mean vectors and $\Sigma_c, \Sigma_d \in \mathbb{R}^{p \times p}$ represent covariance matrices. The goal of differential GGMs is to estimate the structural change Δ defined by [27] [2].

$$\Delta = \Omega_d - \Omega_c \tag{2.1}$$

Here the precision matrices $\Omega_c := (\Sigma_c)^{-1}$ and $\Omega_d := (\Sigma_d)^{-1}$. The conditional dependency structure of a GGM is encoded by the sparsity pattern of its precision matrix. Therefore, one entry of Δ describes if the magnitude of conditional dependency of a pair of random variables changes between two conditions. A sparse Δ means few of its entries are non-zero, indicating a differential network with few edges.

A naive approach to estimate Δ is a two-step procedure in which we estimate $\widehat{\Omega}_d$ and $\widehat{\Omega}_c$ from two sets of samples separately and calculate $\widehat{\Delta}$ using Eq. (2.1). However, in a high-dimensional setting, the strategy needs to assume both Ω_d and Ω_c are sparse (to achieve consistent estimation of each), although the assumption is not necessarily true even if the change Δ is sparse (details in Section S:1).

[1] We put details of theoretical proofs, details of how we generate simulation datasets and concrete performance figures in the appendix. Notations with "S:" as the prefix indicate the corresponding contents are in the appendix.

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

Multiple recent studies have been motivated to directly estimate Δ from two sets of samples. We call these studies differential GGM estimators and group them to four kinds. (1) **Likelihood based**. Zhang et al. [25] used the fused norm for regularizing the maximum likelihood estimation (MLE) to simultaneously learn both two GGMs and the difference ($\lambda_2(||\Omega_c||_1 + ||\Omega_d||_1) + \lambda_n||\Delta||_1$). The resulting penalized MLE framework is a log-determinant program, which can be solved by block coordinate descent [25] or the alternating direction method of multipliers (ADMM) by the JGLFUSED package [5]. (2) **Density ratio based:**. Recently Liu et al. used density ratio estimation (SDRE) to directly learn Δ without having to identify the structures of Ω_c and Ω_d. The authors focused on exponential family-based pairwise Markov networks [10] and solved the resulting optimization using proximal gradient descent [9]. (3) **Constrained ℓ_1 minimization based**. Diff-CLIME, another regularized convex program, was proposed to directly learn structural changes Δ without going through the learning of each individual GGMs [26]. It uses an ℓ_1 minimization formulation constrained by the covariance-precision matching, reducing the estimation problem to solving linear programs. All three aforementioned groups have used ℓ_1 regularized convex formulation for estimating Δ. (4) **Elementary estimator based**. The last category extends the so-called Elementary Estimator proposed by [21, 23, 22] to achieve a closed-form estimation of differential GGM via the DIFFEE estimator [19] (more in the next section and Section 2.4).

2.2 Background: Elementary Estimators for Graphical Models

ℓ_1 Regularized MLE for GGM Estimation: Graphical Lasso (GLasso). The "GLasso" Estimator [24, 1] is the classic formulation for estimating sparse GGM from observations drawn from a single multivariate Gaussian distribution. It optimizes the following ℓ_1 penalized MLE objective:

$$\underset{\Omega>0}{\operatorname{argmin}} - \log\det(\Omega) + <\Omega, \widehat{\Sigma}> + \lambda_n||\Omega||_1 \tag{2.2}$$

Where $\lambda_n > 0$ is the sparsity regularization parameter. While state-of-the-art optimization methods have been developed to solve the optimization in Eq. (2.2), they are expensive for large-scale tasks.

ℓ_1 based Elementary Estimator for Graphical Model (EE-GM) Estimation: Yang et al. [23] proposed to learn sparse Gaussian graphical model via the following formulation instead:

$$\underset{\Omega}{\operatorname{argmin}} ||\Omega||_{1,,\text{off}}, \qquad \text{Subject to:} ||\Omega - [T_v(\widehat{\Sigma})]^{-1}||_{\infty,\text{off}} \le \lambda_n \tag{2.3}$$

Actually [23] proposed the following generic formulation to estimate graphical models (GM) of exponential families (GGM is a special case of GM with exponential distribution):

$$\underset{\theta}{\operatorname{argmin}} ||\theta||_1, \qquad \text{Subject to: } ||\theta - \mathcal{B}^*(\widehat{\phi})||_\infty \le \lambda_n \tag{2.4}$$

Here θ is the canonical parameter to be estimated and $\mathcal{B}^*(\widehat{\phi})$ is a so-called proxy of backward mapping for the target GM. $\widehat{\phi}$ is the empirical mean of the sufficient statistics of the underlying exponential distribution. For example, in the case of Gaussian, θ is the precision matrix, $\widehat{\phi}$ is the sample covariance matrix and the proxy backward mapping is $\mathcal{B}^*(\widehat{\phi}) = [T_v(\widehat{\Sigma})]^{-1}$ (We explain backward mapping, proxy backward mapping and the property and convergence rate of $[T_v(\widehat{\Sigma})]^{-1}$ in Section S:4.1).

The main advantage of Eq. (2.4) and Eq. (2.3) was that they are simple estimators with computationally easy solutions. Importantly their solutions achieve the same sharp convergence rate as the regularized convex formulation of Eq. (2.2) when under high-dimensional settings.

$\mathcal{R}(\cdot)$ norm based Elementary Estimators: Recently multiple studies [21, 22, 19, 18] followed [23] and expanded EE-GM into a more general framework "Elementary estimators" (EE):

$$\underset{\theta}{\operatorname{argmin}} \mathcal{R}(\theta), \qquad \text{Subject to: } \mathcal{R}^*(\theta - \widehat{\theta}_n) \le \lambda_n \tag{2.5}$$

Where $\mathcal{R}(\cdot)$ represents a decomposable regularization function. $\mathcal{R}^*(\cdot)$ is the dual norm of $\mathcal{R}(\cdot)$,

$$\mathcal{R}^*(v) := \sup_{u\neq 0} \frac{<u,v>}{\mathcal{R}(u)} = \sup_{\mathcal{R}(u)\le 1} <u,v> . \tag{2.6}$$

Eq. (2.4) and Eq. (2.3) are special cases of Eq. (2.5). $\widehat{\theta}_n$ needs to be carefully constructed, well-defined and closed-form for the purpose of simplified computations. For example, [21] conduct the high-dimensional estimation of linear regression models by using the classical ridge estimator as $\widehat{\theta}_n$ in Eq. (2.5). When $\widehat{\theta}_n$ itself is closed-form and comes with strong statistical convergence guarantees in high-dimensional situations, we can use the unified framework proposed by the recent seminal study from [11] to prove that the solution of Eq. (2.5) achieves the near optimal convergence rate as comparable to regularized convex formulations when satisfying certain conditions.

2.3 Integrating additional knowledge and Δ with a Novel Function: kEV norm

Section 2.1 points out that none of the previous Δ estimators have designed to integrate extra evidence beyond two sets of observed samples. Differently our Δ estimator aims to achieve two goals: (1) the new estimator should be flexible enough to describe various kinds of real-world knowledge, including like spatial distance, hub knowledge, known interactions or how multiple variables function as groups (see below). (2) the new estimator should work well in high-dimensional situations (large p) and is computationally practical. Eq. (2.5) provides an intriguing formulation to build simpler and possibly fast estimators accompanied by statistical guarantees, as long as $\widehat{\theta}_n$ can be carefully constructed, well-defined and closed-form. We adapt it to design KDiffNet in the next Section 2.4.

In order to use Eq. (2.5) for estimating our target parameter $\theta = \Delta$, we need to design $\mathcal{R}(\Delta)$.

(1) Knowledge as Weight Matrix: We can describe the edge-level knowledge as positive weight matrices like $W_E \in \mathbb{R}^{p \times p}$. For example, when estimating the functional brain connectivity networks among brain regions W_E can describe spatial distance among brain regions that are publicly available through projects like openfMRI [12]. Another important example is when identifying gene-gene interactions from patients' gene expression profiles. Besides the patient samples, state-of-the-art bio-databases like HPRD [13] have collected a significant amount of information about direct physical interactions among corresponding proteins, regulatory gene pairs or signaling relationships collected from high-quality bio-experiments. Here W_E can describe existing known edges as the knowledge, like those from interaction databases for discovering gene networks (a semi-supervised setting for such sample based network estimations).

The positive matrix-based representation provides a powerful and flexible strategy that allows integration of many possible forms of existing knowledge to improve differential structure estimation, as long as they can be represented into edge-level weights. We can combine W_E knowledge and the sparse regularization of Δ into a weighted ℓ_1 norm $||W_E \circ \Delta||_1$, enforcing prior known importance of edges in the differential graph through weights. The larger a weight entry in W_E, the less likely the corresponding edge belongs to the true differential graph. As mentioned in Section 1, NAK and JEEK estimators have tried similar weight matrix based strategy to add extra knowledge in identifying single-task GGM and in discovering multiple GGMs. None of the previous differential GGM estimators have explored this though.

(2) Knowledge as Node Groups: In many real-world applications, there exist known group knowledge about random variables. For example, when working with genomics samples, biologists have collected a rich set of group evidence about how genes belong to various biological pathways or exist in the same or different cellular locations [4]. Such knowledge of node grouping provides solid biological bias like genes belonging to the same biology pathway tend to have interactions among them (shared dependency pattern) in one cellular context or tend to not interact with each other (shared sparsity) at some other cellular conditions. However, this type of group evidence cannot be described via the weight matrix W_E based formulation.

This is because even though it is safe to assume nodes in the same group share similar interaction patterns, but we do not know beforehand if the nodes in the group are collectively part of the differential network (group dependency) or not (group sparsity). To mitigate this issue, we use a flexible known node-group norm to include such extra knowledge. We represent the group knowledge as a set of groups on feature variables (vertices) $\mathcal{G}_p$. Formally, $\forall g_k \in \mathcal{G}_p$, $g_k = \{i\}$ where i indicates that the i-th node belongs to the group k. Integrating $\mathcal{G}_p$ knowledge into Δ means to enforce a group sparsity regularization on Δ. We generate edge-group index $\mathcal{G}_V$ from the node group index $\mathcal{G}_p$. This is done via defining $\mathcal{G}_V := \{g'_k | (i,j) \in g'_k, \forall i, \forall j \in g_k\}$. For vertex nodes in each node group g_k, all possible pairs between these nodes belong to an edge-group g'_k. We propose to use the group,2 norm $||\Delta||_{\mathcal{G}_V,2}$ to enforce group-wise sparse structure on Δ. None of the previous differential GGM estimators have explored this knowledge-integration strategy before.

kEV norm: Now we propose a novel norm $\mathcal{R}(\Delta)$ to combine the two strategies above. We assume that the true parameter $\Delta^* = \Delta^*_e + \Delta^*_g$ is a superposition of two "clean" structures, a sparse structured Δ^*_e and a group-structured Δ^*_g. We propose a new norm, knowledge for Edges and Vertex norm (kEV-norm), as the superposition of the edge-weighted ℓ_1 norm and the group structured norm:

$$\mathcal{R}(\Delta) = ||W_E \circ \Delta_e||_1 + \epsilon||\Delta_g||_{\mathcal{G}_V,2} \tag{2.7}$$

Our target parameter $\Delta = \Delta_e + \Delta_g$. The Hadamard product $\circ$ is the element-wise product between two matrices i.e. $[A \circ B]_{ij} = A_{ij}B_{ij}$ and $||\cdot||_{\mathcal{G}_V,2} = \sum_k ||\Delta_{g'_k}||_2$ where k is the k-th group. $W_E \in R^{p \times p}$ is the aforementioned edge-level additional knowledge. $W_{E_{ij}} > 0$, $\forall\, i,j \in \{1 \ldots p\}$. $\epsilon > 0$ is a hyperparameter. kEV-norm has the following three properties (proofs in Section S:3).

- (i) kEV-norm is a norm function if ϵ and entries of W_E are positive.
- (ii) If the condition in (i) holds, kEV-norm is a decomposable norm.
- (iii) The dual norm of kEV-norm is

$$\mathcal{R}^*(u) = \max(||(1 \oslash W_E) \circ u||_\infty, \frac{1}{\epsilon}||u||^*_{\mathcal{G}_V,2}) \tag{2.8}$$

2.4 KDiffNet : kEV Norm based Elementary Estimator for identifying Differential Net

Our goal is to achieve simple, scalable and theoretically sound estimation. EE in Eq. (2.5) provides such a formulation as long as we can construct $\widehat{\theta}_n$ well. Now we have $\mathcal{R}(\Delta)$ as Eq. (2.7) and its dual norm $\mathcal{R}^*(\cdot)$ in Eq. (2.8). We just need to find $\widehat{\theta}_n$ for Δ that is carefully constructed, theoretically well-behaved when high-dimensional, and closed-form for the purpose of simplified computations.

One key insight of differential GGM is that the density ratio of two Gaussian distributions is naturally an exponential-family distribution (see proofs in Section S:4.2). The differential network Δ is one entry of the canonical parameter for this distribution. The MLE solution of estimating vanilla (i.e. no sparsity and not high-dimensional) graphical model in an exponential family distribution can be expressed as a backward mapping that computes the target model parameters from certain given moments. When using vanilla MLE to learn the exponential distribution about differential GGM (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 S:4.2). Even though this backward mapping has a simple closed form, it is not well-defined when high-dimensional because $\widehat{\Sigma}_c$ and $\widehat{\Sigma}_d$ are rank-deficient (thus not invertible) when $p > n$. Using Eq. (2.3) to estimate Δ, Wang et. al. [19] proposed the DIFFEE estimator for EE-based differential GGM estimation and used only the sparsity assumption on Δ. This study proposed a proxy backward mapping as $\widehat{\theta}_n = [T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$. Here $[T_v(A)]_{ij} := \rho_v(A_{ij})$ and $\rho_v(\cdot)$ is chosen as a soft-threshold function.

We borrow the idea to use $\widehat{\theta}_n = [T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$. In Section S:4.3 and Section S:4.4 we prove that $\widehat{\theta}_n$ is both available in closed-form, and well-defined in high-dimensional settings. Now by plugging $\mathcal{R}(\Delta)$, its dual $\mathcal{R}^*(\cdot)$ and $\widehat{\theta}_n$ into Eq. (2.5), we get the formulation of KDiffNet :

$$\begin{aligned}
&\operatorname*{argmin}_{\Delta} ||W_E \circ \Delta_e||_1 + \epsilon||\Delta_g||_{\mathcal{G}_V,2} \\
\text{s.t.: } &||(1 \oslash W_E) \circ \Big(\Delta - \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)\Big)||_\infty \le \lambda_n \\
&||\Delta - \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)||^*_{\mathcal{G}_V,2} \le \epsilon\lambda_n \\
&\Delta = \Delta_e + \Delta_g
\end{aligned} \tag{2.9}$$

2.5 Solving KDiffNet

We then propose to use a proximal parallel based optimization to solve Eq. (2.9), inspired by its distributed and parallel nature [3]. To simplify notations, we add a new notation $\Delta_{tot} := [\Delta_e; \Delta_g]$, where ; denotes the row wise concatenation. We also add three operator notations including $L_e(\Delta_{tot}) = \Delta_e$, $L_g(\Delta_{tot}) = \Delta_g$ and $L_{tot}(\Delta_{tot}) = \Delta_e + \Delta_g$. Now we obtain the following re-formulation of KDiffNet :

$$\begin{aligned}
&\operatorname*{argmin}_{\Delta_{tot}} ||W_E \circ (L_e(\Delta_{tot}))||_1 + \epsilon||L_g(\Delta_{tot})||_{\mathcal{G}_V,2} \\
\text{s.t.: } &||(1 \oslash W_E) \circ \Big(L_{tot}(\Delta_{tot}) - \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)\Big)||_\infty \le \lambda_n \\
&||L_{tot}(\Delta_{tot}) - \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)||^*_{\mathcal{G}_V,2} \le \epsilon\lambda_n
\end{aligned} \tag{2.10}$$

Actually the three added operators are affine mappings: $L_e = A_e\Delta_{tot}$, $L_g = A_g\Delta_{tot}$, and $L_{tot} = A_{tot}\Delta_{tot}$, where $A_e = [\boldsymbol{I}_{p\times p} \quad \mathbf{0}_{p\times p}]$, $A_g = [\mathbf{0}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$ and $A_{tot} = [\boldsymbol{I}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$.

Algorithm 1 summarizes the Parallel Proximal algorithm [3, 22] we propose for optimizing Eq. (2.10). In Section S:1.3 we further prove that its computational cost is $O(p^3)$. More concretely in Algorithm 1, we simplify the notations by denoting $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c) := [T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}$, and reformulate Eq. (2.10) to the following equivalent and distributed formulation:

$$\begin{aligned} &\underset{\Delta_{tot}}{\operatorname{argmin}}\, F_1(\Delta_{tot_1}) + F_2(\Delta_{tot_2}) + G_1(\Delta_{tot_3}) + G_2(\Delta_{tot_4}) \\ &\text{subject to: } \Delta_{tot_1} = \Delta_{tot_2} = \Delta_{tot_3} = \Delta_{tot_4} = \Delta_{tot} \end{aligned} \tag{2.11}$$

Where $F_1(\cdot) = ||W_E \circ (L_e(\cdot))||_1$, $G_1(\cdot) = \mathcal{I}_{||(1 \oslash W_E)\circ\left(L_{tot}(\cdot) - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)\right)||_\infty \le \lambda_n}$, $F_2(\cdot) = \epsilon||L_g(\cdot)||_{\mathcal{G}_V,2}$ and $G_2(\cdot) = \mathcal{I}_{||L_{tot}(\cdot) - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||^*_{\mathcal{G}_V,2} \le \epsilon\lambda_n}$. Here $\mathcal{I}_C(\cdot)$ represents the indicator function of a convex set C denoting that $\mathcal{I}_C(x) = 0$ when $x \in C$ and otherwise $\mathcal{I}_C(x) = \infty$. The detailed solution of each proximal operator is summarized in Table S:1 and Section S:2.

Algorithm 1 A Parallel Proximal Algorithm to optimize KDiffNet

```
input  Two data matrices $\mathbf{X}_c$ and $\mathbf{X}_d$, The weight matrix $W_E$ and $\mathcal{G}_\mathcal{V}$.
       Hyperparameters: $\alpha$, $\epsilon$, $v$, $\lambda_n$ and $\gamma$. Learning rate: $0 < \rho < 2$. Max iteration number $iter$.
output  $\Delta$
Compute $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$ from $\mathbf{X}_d$ and $\mathbf{X}_c$
Initialize $A_e = [\boldsymbol{I}_{p\times p} \quad \mathbf{0}_{p\times p}]$, $A_g = [\mathbf{0}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$, $A_{tot} = [\boldsymbol{I}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$,
Initialize $\Delta_{tot_1}, \Delta_{tot_2}, \Delta_{tot_3}, \Delta_{tot_4}$ and $\Delta_{tot} = \dfrac{\Delta_{tot_1} + \Delta_{tot_2} + \Delta_{tot_3} + \Delta_{tot_4}}{4}$
for $i = 0$ to $iter$ do
    $p_1^i = \text{prox}_{4\gamma F_1}\Delta_{tot_1}^i$; $p_2^i = \text{prox}_{4\gamma F_2}\Delta_{tot_2}^i$; $p_3^i = \text{prox}_{4\gamma G_1}\Delta_{tot_3}^i$; $p_4^i = \text{prox}_{4\gamma G_2}\Delta_{tot_4}^i$
    $p^i = \frac{1}{4}(\sum_{j=1}^{4} p_j^i)$
    for $j = 1, 2, 3, 4$ do
        $\Delta_{tot_j}^{i+1} = \Delta_{tot}^i + \rho(2p^i - \Delta_{tot}^i - p_j^i)$
    end for
    $\Delta_{tot}^{i+1} = \Delta_{tot}^i + \rho(p^i - \Delta_{tot}^i)$
end for
$\widehat{\Delta} = A_{tot}\Delta_{tot}^{iter}$
output  $\widehat{\Delta}$
```

2.6 Analysis of Error Bounds

Based on Theorem S:5.3 and conditions in Section S:5, we have the following corollary about the convergence rate of KDiffNet . See its proof in Section S:5.2.2.

Corollary 2.1. *In 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* $\lambda_n := \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p}{\min(n_c, n_d)}}$ *and* $\min(n_c, n_d) > c\log p$*, with a probability of at least* $1 - 2C_1\exp(-C_2 p\log(p))$*, the estimated optimal solution* $\widehat{\Delta}$ *has the following error bound:*

$$||\widehat{\Delta} - \Delta^*||_F \le \frac{16\kappa_1 a \max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}), \epsilon\sqrt{s_G})}{\min_{i,j}(W_{E_{i,j}})\kappa_2}\sqrt{\frac{\log p}{\min(n_c, n_d)}} \tag{2.12}$$

where C_1,C_2,a, c, κ_1 *and* κ_2 *are constants. See* s *and* s_G *in Definition S:3.4.*

3 Experiments

We aim to empirically show that KDiffNet is adaptive and flexible in incorporating different kinds of available evidence for improved differential network estimation. **Data:** This is accomplished by evaluating KDiffNet and baselines on two sets of datasets: (1) A total of 126 different synthetic datasets representing various combinations of additional knowledge (details see Section 3.1); and (2) one real-world fMRI dataset ABIDE for functional brain connectivity estimation (Section 3.2). We obtain the edge-level knowledge from three different human brain atlas [7, 8, 16] about brain connectivity, resulting in three different W_E with $p = \{116, 160, 246\}$. For each atlas we compute W_E using the spatial distance between its brain Region of Interests (ROIs). At the same time, we explore two different types of group knowledge about brain regions from Dosenbach Atlas[7] (Section 3.2). **Baselines:** We compare KDiffNet to JEEK[18] and NAK[2], that use the extra edge

Method	Data-EG(Time)	Data-EG(F1-Score)			Data-G(Time)	Data-G(F1-Score)
	W2($p = 246$)	W1($p = 116$)	W2($p = 246$)	W3($p = 160$)	W2($p = 246$)	W2($p = 246$)
KDiffNet-EG	4.591±0.08	**0.717±0.05**	**0.927±0.01**	**0.934±0.07**	*	*
KDiffNet-G	**0.013±0.01**	0.578±0.09	0.565±0.10	0.575±0.09	0.006±0.00	**0.891±0.06**
KDiffNet-E	**0.017±0.01**	0.692±0.06	0.872±0.02	0.916±0.02	*	*
JEEK [18]	10.998±0.11	0.572±0.10	0.581±0.09	0.580±0.09	*	*
NAK[2]	3.800±0.15	0.226±0.08	0.204±0.07	0.207±0.08	*	*
SDRE[10]	27.487±2.59	0.573±0.11	0.568±0.11	0.574±0.11	11.764±1.23	0.318±0.10
DIFFEE[19]	**0.004±0.00**	0.570±0.11	0.560±0.11	0.570±0.11	0.004±0.00	0.135±0.02
JGLFUSED[5]	68.128±1.66	0.502±0.08	0.481±0.08	0.495±0.08	144.470±59.68	0.055±0.01

Table 1: Mean Performance(F1-Score) and Computation Time(seconds) with standard deviation given in parentheses for the same setting of n_c and n_d of KDiffNet-EG , KDiffNet-E , KDiffNet-G and baselines for simulated data. $*$ indicates that the method is not applicable for a data setting.

knowledge, two direct differential estimators (SDRE[9], DIFFEE[19]) and MLE based JGLFUSED[5] (Section 2.1 and detailed equations of each in Section S:1). We also extend KDiffNet to data situations with only edge knowledge (KDiffNet-E) or only group knowledge (KDiffNet-G). Both variations (KDiffNet-E and KDiffNet-G) can be solved by fast closed form solutions (Section S:2.2).

Additional details of setup, metrics and hyper-parameters are in Section S:6.1. **Hyperparameters:** The key hyper-parameters are tuned as follows:

- v : To compute the proxy backward mapping, we vary v in $\{0.001i|i = 1, 2, \dots, 1000\}$ (to make $T_v(\Sigma_c)$ and $T_v(\Sigma_d)$ invertible).
- λ_n : According to our convergence rate analysis in Section 2.6, $\lambda_n \geq C\sqrt{\frac{\log p}{\min(n_c,n_d)}}$, we choose λ_n from a range of $\{0.01 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i|i \in \{1, 2, 3, \dots, 100\}\}$ using cross-validation. For KDiffNet-G , we tune over λ_n from a range of $\{0.1 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i|i \in \{1, 2, 3, \dots, 100\}\}$[3].
- ϵ: For KDiffNet-EG experiments, we tune $\epsilon \in \{0.0001, 0.01, 1, 100\}$.

3.1 Experiment: Simulated Data about Brain Connectivity using Three Real-World Brain Spatial Matrices and Anatomic Group Evidence from Neuroscience as Knowledge

In this section, we show the effectiveness of KDiffNet in integrating additional evidence through a comprehensive set of many simulation datasets. Our simulated data settings mimic three possible types of additional knowledge in the real-world: with both edge and known node group knowledge (Data-EG), with only edge-level evidence (Data-E) or with only known node groups (Data-G). For the edge knowledge, we consider three cases of W_E with $p = \{116, 160, 246\}$ computed from three human brain atlas about brain regions [7, 8, 16]. For the group knowledge, we simulate groups to represent related anatomic regions inspired by the atlas [7]. For each simulation dataset, two blocks of data samples are generated following Gaussian distribution using $N(0, \Omega_c^{-1})$ and $N(0, \Omega_d^{-1})$ via the simulated Ω_c and Ω_d. Each simulated dataset includes a pair of data blocks to estimate its differential GGM. We conduct a comprehensive evaluation over a total of 126 different simulated datasets by varying (p), varying the number of samples (n_c and n_d), changing the proportion of edges controlled by W_E (s) and by varying the number of known groups s_G. The details of the simulation framework are in Section S:6.2.

We present a summary of our results (partial) in Table 1 using columns showing two cases of data generation settings (Data-EG and Data-G). Table 1 uses the mean F1-score and the computational time cost to compare methods (rows). Results about simulated datasets under Data-E case are in Section S:6. We can make several conclusions:

(1) **KDiffNet outperforms those baselines not considering knowledge.** Clearly KDiffNet and its variations achieve the highest F1-score across all the 126 datasets. SDRE and DIFFEE are direct differential network estimators but perform poorly indicating that adding additional knowledge improves differential GGM estimation. MLE based JGLFUSED performs the worst in all cases.

(2) **KDiffNet outperforms those baselines considering knowledge, especially when group knowledge exist.** When under the Data-EG setting, while JEEK and NAK include the extra edge information, they cannot integrate group information and are not for differential estimation. This

[3] We use the same range to tune λ_1 for SDRE and λ_2 for JGLFUSED. We use $\lambda_1 = 0.0001$(a small value) for JGLFUSED to ensure only the differential network is sparse. Tuning NAK is done by the package itself.

results in lower F1-Score(0.581 and 0.204 for W2) compared to KDiffNet-EG (0.927 for W2). The advantage of modeling both edge and node groups evidence is also indicated by the higher F1-Score of KDiffNet-EG with respect to KDiffNet-E and KDiffNet-G on the Data-EG setting (Top 3 rows in Table 1). On Data-G cases, none of the baselines can model node group evidence. On average KDiffNet-G performs $6.4\times$ better than the baselines for $p = 246$ with respect to F1.

(3) **KDiffNet achieves reasonable time cost versus the baselines and is scalable to large p.** Figure 1(a) shows each method's time cost per λ_n for large $p = 2000$. Consistently KDiffNet-EG is faster than JEEK, JGLFUSED and SDRE (Column 1 in Table 1). KDiffNet-E and KDiffNet-G are faster than KDiffNet-EG owing to closed form solutions. On Data-G dataset and Data-E datasets (Section S:6.2), our faster closed form solutions achieve much more significant computational all the baselines. For example on datasets using W2 $p = 246$, KDiffNet-E and KDiffNet-G are on an average $21000\times$ and $7400\times$ faster (Column 5 in Table 1) than the baselines, respectively. We have all detailed results and figures about F1-Score and time cost for all 126 data settings in Section S:6.2. Besides F1-Score, we also present the ROC curves from all methods when varying λ_n. KDiffNet achieves the highest Area under Curve (AUC) in comparison to all other baselines.

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

In this experiment, we evaluate KDiffNet in a real-world downstream classification task on a publicly available resting-state fMRI dataset: ABIDE[6]. This aims to understand how functional dependencies among brain regions vary between normal and abnormal and help to discover contributing markers that influence or cause the neural disorders [17]. ABIDE includes two groups of human subjects: autism and control. We utilize three types of additional knowledge: W_E based on the spatial distance between 160 regions of the brain[7] and two types of available node groups from Dosenbach Atlas[7]: one with 40 unique groups about macroscopic brain structures (G1) and another with 6 higher level node groups having the same functional connectivity(G2). We use Quadratic Discriminant Analysis (QDA) in downstream classification to assess the ability of the estimators to learn the differential patterns about the connectome structures. (Details of the ABIDE dataset, baselines, design of the additional knowledge W_E matrix, cross-validation and the QDA classification method are in Section S:6.4.) Figure 1(b) compares KDiffNet-EG , KDiffNet-E , KDiffNet-G and baselines on ABIDE, using the y axis for classification test accuracy (the higher the better) and the x axis for the computation speed (negative log seconds, the more right the better). KDiffNet -EG1, incorporating both edge(W_E) and (G1) group knowledge, achieves the highest accuracy of 57.2% for distinguishing the autism subjects versus the control subjects without sacrificing computation speed [4].

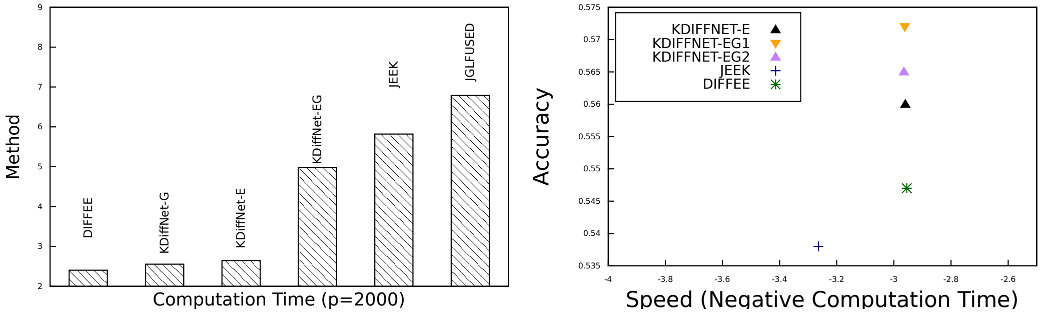

Figure 1: (a)(LEFT) Computation Time (log milliseconds) per λ_n for large $p = 2000$: KDiffNet-EG has reasonable time cost with respect to baseline methods. KDiffNet-E and KDiffNet-G are fast closed form solutions. (b) (RIGHT) ABIDE Dataset: KDiffNet-EG achieves highest Accuracy without sacrificing computation speed (points towards top right are better).

4 Conclusions

We propose a novel method, KDiffNet , to incorporate additional knowledge in estimating differential GGMs. KDiffNet elegantly formulates existing knowledge based on the problem at hand and avoids the need to design knowledge-specific optimization. We sincerely believe the scalability and flexibility provided by KDiffNet can make differential structure learning of GGMs feasible in many real-world tasks. We plan to generalize KDiffNet from Gaussian to semi-parametric distributions or to Ising Model structures. As node group knowledge is particularly important and abundant in genomics, we plan to evaluate KDiffNet on more real-world genomics data with multiple types of group information.

[4] We cannot compare to NAK and SDRE because they do not provide precision matrix required for QDA

Acknowledgement: *This work was supported partly by the National Science Foundation under NSF CAREER award No. 1453580. Any Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.*

References

[1] O. Banerjee, L. El Ghaoui, and A. 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.

[2] Y. Bu and J. Lederer. Integrating additional knowledge into estimation of graphical models.

[3] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[4] B. T. S. Da Wei Huang and R. A. Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature protocols*, 4(1):44–57, 2008.

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

[6] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. 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.

[7] N. U. Dosenbach, B. Nardos, A. L. Cohen, D. A. Fair, J. D. Power, J. A. Church, S. M. Nelson, G. S. Wig, A. C. Vogel, C. N. Lessov-Schlaggar, et al. Prediction of individual brain maturity using fmri. *Science*, 329(5997):1358–1361, 2010.

[8] L. Fan, H. Li, J. Zhuo, Y. Zhang, J. Wang, L. Chen, Z. Yang, C. Chu, S. Xie, A. R. Laird, et al. The human brainnetome atlas: a new brain atlas based on connectional architecture. *Cerebral cortex*, 26(8):3508–3526, 2016.

[9] S. Liu, K. Fukumizu, and T. Suzuki. Learning sparse structural changes in high-dimensional markov networks. *Behaviormetrika*, 44(1):265–286, 2017.

[10] S. Liu, J. A. Quinn, M. U. Gutmann, T. Suzuki, and M. Sugiyama. Direct learning of sparse changes in markov networks by density ratio estimation. *Neural computation*, 26(6):1169–1197, 2014.

[11] S. Negahban, B. Yu, M. J. Wainwright, and P. 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.

[12] R. A. Poldrack, D. M. Barch, J. Mitchell, T. Wager, A. D. Wagner, J. T. Devlin, C. Cumba, O. Koyejo, and M. Milham. Toward open sharing of task-based fmri data: the openfmri project. *Frontiers in neuroinformatics*, 7:12, 2013.

[13] T. K. Prasad, R. Goel, K. Kandasamy, S. Keerthikumar, S. Kumar, S. Mathivanan, D. Telikicherla, R. Raju, B. Shafreen, A. Venugopal, et al. Human protein reference database 2009 update. *Nucleic acids research*, 37(suppl 1):D767–D772, 2009.

[14] T. Shimamura, S. Imoto, R. Yamaguchi, and S. Miyano. Weighted lasso in graphical gaussian modeling for large gene network estimation based on microarray data. 19:142–153.

[15] C. Singh, B. Wang, and Y. Qi. A constrained, weighted-l1 minimization approach for joint discovery of heterogeneous neural connectivity graphs. *arXiv preprint arXiv:1709.04090*, 2017.

[16] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

[17] D. C. Van Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, K. Ugurbil, W.-M. H. Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[18] B. Wang, A. Sekhon, and Y. Qi. A fast and scalable joint estimator for integrating additional knowledge in learning multiple related sparse gaussian graphical models. *arXiv preprint arXiv:1806.00548*, 2018.

[19] B. Wang, A. Sekhon, and Y. Qi. Fast and scalable learning of sparse changes in high-dimensional gaussian graphical model structure. In *Proceedings of The 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018. [PS].

[20] D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’networks. 393(6684):440–442.

[21] E. Yang, A. C. Lozano, and P. Ravikumar. Elementary estimators for high-dimensional linear regression. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 388–396, 2014.

[22] E. Yang, A. C. Lozano, and P. D. Ravikumar. Elementary estimators for sparse covariance matrices and other structured moments. In *ICML*, pages 397–405, 2014.

[23] E. Yang, A. C. Lozano, and P. K. Ravikumar. Elementary estimators for graphical models. In *Advances in Neural Information Processing Systems*, pages 2159–2167, 2014.

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

[25] B. Zhang and Y. Wang. Learning structural changes of gaussian graphical models in controlled experiments. *arXiv preprint arXiv:1203.3532*, 2012.

[26] S. D. Zhao, T. T. Cai, and H. Li. Direct estimation of differential networks. page asu009.

[27] S. D. Zhao, T. T. Cai, and H. Li. Direct estimation of differential networks. *Biometrika*, 101(2):253–268, 2014.

Appendix
Adding Extra Knowledge in Scalable Learning of Sparse Differential Gaussian Graphical Models

S:1 Connecting to Relevant Studies

S:1.1 Differential GGM Estimation

JGLFUSED[6]: This study extends the previously mentioned MLE based GLasso(Section 2.2) estimator for sparse differential GGM estimation. An additional sparsity penalty on the differential network, called the fused norm, is included as part of the optimization objective:

$$\begin{aligned}\underset{\Omega_c,\Omega_d \succ 0,\Delta}{\operatorname{argmin}}\ & n_c(-\log\det(\Omega_c) + <\Omega_c, \widehat{\Sigma}_c>) \\ & + n_d(-\log\det(\Omega_d) + <\Omega_d, \widehat{\Sigma}_d>) \\ & + \lambda_2(||\Omega_c||_1 + ||\Omega_d||_1) + \lambda_n||\Delta||_1\end{aligned} \tag{S:1.1}$$

The alternating direction method of multipliers (ADMM) method was used to solve Eq. (S:1.1) that needs to run expensive SVD in one sub-procedure [6].

DIFFEE[23]: Computationally, EEs are much faster than their regularized convex program peers for GM estimation. [23] proposed the so-called DIFFEE for estimating sparse changes in high-dimensional GGM structure using EE:

$$\begin{aligned}&\underset{\Delta}{\operatorname{argmin}}\ ||\Delta||_1 \\ &\text{Subject to: } ||\Delta - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||_\infty \le \lambda_n\end{aligned} \tag{S:1.2}$$

[23] use a closed form and well-defined proxy function $\widehat{\theta}_n = \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c) = \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)$ to approximate the backward mapping (the vanilla MLE solution) for differential sGGMs. We explain the proxy backward mapping and its statistical properties in Section S:4.1. The DIFFEE solution is a closed-form entry-wise thresholding operation on $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$ to ensure the desired sparsity structure of its final estimate. Here $\lambda_n > 0$ is the tuning parameter. Eq. (S:1.2) is a special case of Eq. (2.5), in which $\mathcal{R}(\cdot)$ is the ℓ_1-norm for sparsity and the differential network Δ is the θ we aim to estimate.

As claimed by [10] direct estimation of differential GGMs can be more efficient both in terms of the number of required samples as well as the computation time cost. Besides, it does not require to assume each precision matrix as sparse. For instance recent literature in neuroscience has suggested that each subject's functional brain connection network may not be sparse, even though differences across subjects may be sparse [1]. When identifying how genetic networks vary between two conditions, each individual network may contain hub nodes, therefore not entirely sparse [11].

SDRE[12]: [12] proposed to estimate Sparse differential networks in exponential families by Density Ratio Estimation using the following formulation:

$$\underset{\Delta}{\operatorname{argmax}}\ \mathcal{L}_{\text{KLIEP}}(\Delta) - \lambda_n \parallel \Delta \parallel_1 - \lambda_2 \parallel \Delta \parallel_2 \tag{S:1.3}$$

$\mathcal{L}_{\text{KLIEP}}$ 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 $p_d(x)$. This estimator uses the elastic-net penalty for enforcing sparsity. We use the sparseKLIEP[1], that uses sub-gradient descent optimization as a baseline to our method.

[1] http://allmodelsarewrong.net/kliep_sparse/demo_sparse.html

Diff-CLIME[25]: This study directly learns Δ through a constrained optimization formulation.

$$\begin{aligned} &\underset{\Delta}{\operatorname{argmin}} ||\Delta||_1 \\ &\text{Subject to: } ||\widehat{\Sigma}_c \Delta \widehat{\Sigma}_d - (\widehat{\Sigma}_c - \widehat{\Sigma}_d)||_\infty \leq \lambda_n \end{aligned} \tag{S:1.4}$$

The optimization reduces to multiple linear programming problems, which in turn makes this method less scalable to large p with a computational complexity of $O(p^8)$.

S:1.2 Incorporating Additional Knowledge in GGM Estimation

While previous studies do not use available additional knowledge for differential structure estimation, a few studies have tried to incorporate edge level weights for other types of GGM estimation.

NAK [3]: For the single task sGGM, one recent study [3] (following ideas from [17]) proposed to use a weighted Neighborhood selection formulation to integrate edge-level Additional Knowledge (NAK) as: $\widehat{\beta}^j = \underset{\beta, \beta_j = 0}{\operatorname{argmin}} \frac{1}{2}||\boldsymbol{X}^j - \boldsymbol{X}\beta||_2^2 + ||\mathbf{r}_j \circ \beta||_1$. Here $\widehat{\beta}^j$ is the j-th column of a single sGGM $\widehat{\Omega}$. Specifically, $\widehat{\beta}^j_k = 0$ if and only if $\widehat{\Omega}_{k,j} = 0$. $\mathbf{r}_j$ represents a weight vector designed using available extra knowledge for estimating a brain connectivity network from samples $\boldsymbol{X}$ drawn from a single condition. The NAK formulation can be solved by a classic Lasso solver like glmnet.

JEEK[22]: Two related studies, JEEK[22] and W-SIMULE[18] incorporate edge-level extra knowledge in the joint discovery of K heterogeneous graphs. In both these studies, each sGGM corresponding to a condition i is assumed to be composed of a task specific sGGM component $\Omega_I^{(i)}$ and a shared component Ω_S across all conditions, i.e., $\Omega^{(i)} = \Omega_I^{(i)} + \Omega_S$. The minimization objective of W-SIMULE is as follows: objective:

$$\begin{aligned} &\underset{\Omega_I^{(i)}, \Omega_S}{\operatorname{argmin}} \sum_i ||W \circ \Omega_I^{(i)}||_1 + \epsilon K ||W \circ \Omega_S||_1 \\ &\text{subject to: } ||\Sigma^{(i)}(\Omega_I^{(i)} + \Omega_S) - I||_\infty \leq \lambda_n, \; i = 1, \dots, K \end{aligned} \tag{S:1.5}$$

W-SIMULE is very slow when $p > 200$ due to the expensive computation cost $O(K^4 p^5)$. In comparison, JEEK is an EE-based optimization formulation:

$$\begin{aligned} &\underset{\Omega_I^{tot}, \Omega_S^{tot}}{\operatorname{argmin}} ||W_I^{tot} \circ \Omega_I^{tot}||_1 + ||W_S^{tot} \circ \Omega_S^{tot}|| \\ &\text{subject to: } ||\frac{1}{W_I^{tot}} \circ (\Omega^{tot} - B^*(\widehat{\phi}))||_\infty \leq \lambda_n \\ &||\frac{1}{W_S^{tot}} \circ (\Omega^{tot} - B^*(\widehat{\phi}))||_\infty \leq \lambda_n \\ &\Omega^{tot} = \Omega_S^{tot} + \Omega_I^{tot} \end{aligned} \tag{S:1.6}$$

Here, $\Omega_I^{tot} = (\Omega_I^{(1)}, \Omega_I^{(2)}, \dots, \Omega_I^{(K)})$ and $\Omega_S^{tot} = (\Omega_S, \Omega_S, \dots, \Omega_S)$. The edge knowledge of the task-specific graph is represented as weight matrix $\{W^{(i)}\}$ and W_S for the shared network. JEEK differs from W-SIMULE in its constraint formulation, that in turn makes its optimization much faster and scalable than WSIMULE. In our experiments, we use JEEK as our baseline.

Drawbacks: However, none of these studies are flexible to incorporate other types of additional knowledge like node groups or cases where overlapping group and edge knowledge are available for the same target parameter. Further, these studies are limited by the assumption of sparse single condition graphs. Estimating a sparse difference graph directly is more flexible as it does not rely on this assumption.

S:1.3 Computational Complexity

We optimize KDiffNet through a proximal algorithm, while KDiffNet-E and KDiffNet-G through closed-form solutions. The resulting computational cost for KDiffNet is $O(p^3)$, broken down into the following steps:

- Estimating two covariance matrices: The computational complexity is $O(max(n_c, n_d)p^2)$.
- Backward Mapping: The element-wise soft-thresholding operation $[T_v(\cdot)]$ on the estimated covariance matrices, that costs $O(p^2)$. This is followed by matrix inversions $[T_v(\cdot)]^{-1}$ to get the proxy backward mapping, that cost $O(p^3)$.
- Optimization: For KDiffNet , each operation in the proximal algorithm is group entry wise or entry wise, the resulting computational cost is $O(p^2)$. In addition, the matrix multiplications cost $O(p^3)$.

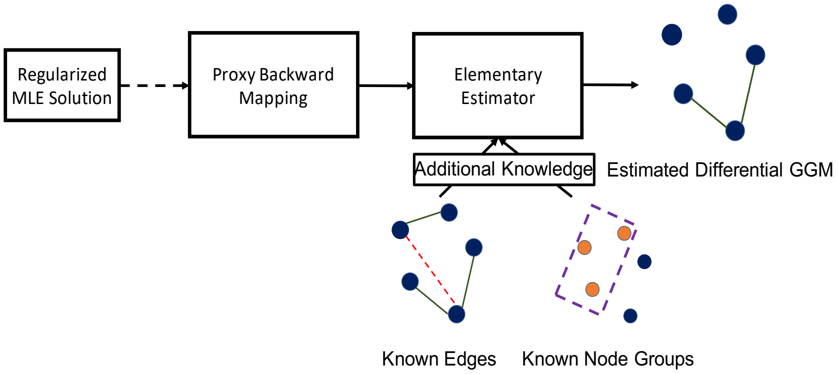

Figure S:1: Schematic Diagram of KDiffNet : integrating extra edge and node groups knowledge for directly estimating the sparse change in the dependency structures of two p-dimensional GGMs (differential GGMs)

For KDiffNet-E and KDiffNet-G versions, the solution is the element-wise soft-thresholding operation S_{λ_n}, that costs $O(p^2)$.

S:2 Optimization of KDiffNet and Its Variants

S:2.1 Optimization via Proximal Solution

We assume $\Delta_{tot} = [\Delta_e; \Delta_g]$, where ; denotes row wise concatenation. Consider operator $L_d(\Delta_{tot}) = \Delta_e$ and $L_g(\Delta_{tot}) = \Delta_g$, $L_{tot}(\Delta_{tot}) = \Delta_e + \Delta_g$.

$$\begin{aligned}
&\underset{\Delta}{\operatorname{argmin}} \, ||W_E \circ (L_e(\Delta_{tot}))||_1 + \epsilon ||L_g(\Delta_{tot})||_{\mathcal{G}_V,2} \\
\text{s.t.: } & ||(1 \oslash W_E) \circ \Big(L_{tot}(\Delta_{tot}) - \big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\big)\Big) ||_\infty \le \lambda_n \\
& ||L_{tot}(\Delta_{tot}) - \big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\big) ||^*_{\mathcal{G}_V,2} \le \epsilon \lambda_n
\end{aligned} \tag{S:2.1}$$

This can be rewritten as:

$$\begin{aligned}
&\underset{\Delta}{\operatorname{argmin}} \, F_1(\Delta_{tot_1}) + F_2(\Delta_{tot_2}) + G_1(\Delta_{tot_3}) + G_2(\Delta_{tot_4}) \\
&\Delta_{tot} = \Delta_{tot_1} = \Delta_{tot_2} = \Delta_{tot_3} = \Delta_{tot_4}
\end{aligned} \tag{S:2.2}$$

Where:

$$\begin{aligned}
&F_1(\cdot) = ||W_E \circ (L_e(\cdot))||_1 \\
&G_1(\cdot) = \mathcal{I}_{||(1 \oslash W_E) \circ \left(L_{tot}(\cdot) - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)\right)||_\infty \le \lambda_n} \\
&F_2(\cdot) = \epsilon ||L_g(\cdot)||_{\mathcal{G}_V,2} \\
&G_2(\cdot) = i_{||L_{tot}(\cdot) - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)||^*_{\mathcal{G}_V,2} \le \epsilon \lambda_n}
\end{aligned} \tag{S:2.3}$$

Here, L_e,L_g and L_{tot} can be written as Affine Mappings. By Lemma in [],

$$\begin{aligned}
L_e &= A_e \Delta_{tot} \\
A_e &= [\boldsymbol{I}_{p \times p} \quad \mathbf{0}_{p \times p}] \\
L_g &= A_g \Delta_{tot} \\
A_g &= [\mathbf{0}_{p \times p} \quad \boldsymbol{I}_{p \times p}] \\
L_{tot} &= A_{tot} \Delta_{tot} \\
A_{tot} &= [\boldsymbol{I}_{p \times p} \quad \boldsymbol{I}_{p \times p}]
\end{aligned} \tag{S:2.4}$$

if $AA^T = \beta I$, and $h(x) = g(Ax)$,

$$prox_h(x) = x - \beta A^T (Ax - prox_{\beta^{-1} g}(Ax)) \tag{S:2.5}$$

$\beta_g = 1$, $\beta_e = 1$ and $\beta_{tot} = 2$.

Solving for each proximal operator:

A. $F_1(\Delta_{tot}) = ||W_E \circ (L_e(\Delta_{tot}))||_1$
$L_e(\Delta_{tot}) = A_e\Delta_{tot} = \Delta_e$.

$$\begin{aligned} &prox_{\gamma F1}(y) = y - A_e^T(x - prox_{\gamma f}(x)) \\ &x = A_e y \end{aligned} \tag{S:2.6}$$

Here, $x_{j,k} = \Delta_{e_{j,k}}$.

$$\begin{aligned} \text{prox}_{\gamma f_1}(x) &= \text{prox}_{\gamma||W\cdot||_1}(x) \\ &= \begin{cases} x_{j,k} - \gamma w_{j,k}, x_{j,k}^{(i)} > \gamma w_{j,k} \\ 0, |x_{j,k}^{(i)}| \leq \gamma w_{j,k} \\ x_{j,k}^{(i)} + \gamma w_{j,k}, x_{j,k}^{(i)} < -\gamma w_{j,k} \end{cases} \end{aligned} \tag{S:2.7}$$

Here $j, k = 1, \ldots, p$. This is an entry-wise operator (i.e., the calculation of each entry is only related to itself). This can be written in closed form:

$$\text{prox}_{\gamma f_1}(x) = \max((x_{j,k} - \gamma w_{j,k}), 0) + \min(0, (x_{j,k} + \gamma w_{j,k})) \tag{S:2.8}$$

We replace this in Eq. (S:2.6).

B. $F_2(\Delta_{tot}) = \epsilon||L_g(\Delta_{tot})||_{\mathcal{G}_V,2}$ Here, $L_g(\Delta_{tot}) = A_g\Delta_{tot} = \Delta_g$.

$$\begin{aligned} &x = A_g y \\ &prox_{\gamma F2}(y) = y - A_g^T(x - prox_{\gamma f_2}(x)) \end{aligned} \tag{S:2.9}$$

Here, $x_{j,k} = \Delta_{g_{j,k}}$.

$$\begin{aligned} \text{prox}_{\gamma f_2}(x_g) &= \text{prox}_{\gamma||\cdot||_{\mathcal{G},2}}(x_g) \\ &= \begin{cases} x_g - \epsilon\gamma\frac{x_g}{||x_g||_2}, ||x_g||_2 > \epsilon\gamma \\ 0, ||x_g||_2 \leq \epsilon\gamma \end{cases} \end{aligned} \tag{S:2.10}$$

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

$$\begin{aligned} \text{prox}_{\gamma f_2}(x_g) &= \text{prox}_{\epsilon\gamma||\cdot||_{\mathcal{G},2}}(x_g) \\ &= x_g \max((1 - \frac{\epsilon\gamma}{||x_g||_2}), 0) \end{aligned} \tag{S:2.11}$$

We replace this is Eq. (S:2.9).

C. $G_1(\Delta_{tot}) = \mathcal{I}_{||(1 \oslash W_E)\circ\left(L_{tot}(\Delta_{tot}) - \left([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\right)\right)||_\infty \leq \lambda_n}$ Here, $L_{tot} = A_{tot}\Delta_{tot}$ and $A_{tot} = [\boldsymbol{I}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$.

$$\begin{aligned} &x = A_{tot} y \\ &prox_{\gamma G1}(y) = y - 2A_{tot}^T(x - prox_{2^{-1}\gamma g_1}(x)) \end{aligned} \tag{S:2.12}$$

$$\begin{aligned} \text{prox}_{\gamma g_1}(x) &= \text{proj}_{||1 \obslash (W_E)\circ(x-a)||_\infty \leq \lambda_n} \\ &= \begin{cases} x_{j,k}, |x_{j,k} - a_{j,k}| \leq w_{j,k}\lambda_n \\ a_{j,k} + w_{j,k}\lambda_n, x_{j,k} > a_{j,k} + w_{j,k}\lambda_n \\ a_{j,k} - w_{j,k}\lambda_n, x_{j,k} < a_{j,k} - w_{j,k}\lambda_n \end{cases} \end{aligned} \tag{S:2.13}$$

In closed form:

$$\begin{aligned} \text{prox}_{\gamma g_1}(x) &= \text{proj}_{||x-a||_\infty \leq \lambda_n} \\ &= \min(\max(x_{j,k} - a_{j,k}, -w_{j,k}\lambda_n), w_{j,k}\lambda_n) + a_{j,k} \end{aligned} \tag{S:2.14}$$

We replace this in Eq. (S:2.12).

D. $G_2(\Delta_{tot}) = \mathcal{I}_{\{||L_{tot}(\Delta_{tot}) - B^*||^*_{\mathcal{G},2} \leq \epsilon\lambda_n\}}$ Here, $L_{tot} = A_{tot}\Delta_{tot}$ and $A_{tot} = [\boldsymbol{I}_{p\times p} \quad \boldsymbol{I}_{p\times p}]$.

$$\begin{aligned} &x = A_{tot} y \\ &prox_{\gamma G2}(y) = y - 2A_{tot}^T(x - prox_{2^{-1}\gamma g_2}(x)) \end{aligned} \tag{S:2.15}$$

$$\begin{aligned} \text{prox}_{\gamma g_2}(x_g) &= \text{proj}_{||x-a||^*_{\mathcal{G},2} \leq \epsilon\lambda_n} \\ &= \begin{cases} x_g, ||x_g - a_g||_2 \leq \epsilon\lambda_n \\ \epsilon\lambda_n\frac{x_g - a_g}{||x_g - a_g||_2} + a_g, ||x_g - a_g||_2 > \epsilon\lambda_n \end{cases} \end{aligned} \tag{S:2.16}$$

Table S:1: The four proximal operators

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

This operator is group entry-wise. In closed form:

$$\begin{aligned} \text{prox}_{\gamma g_2}(x) &= \text{proj}_{||x-a||^*_{\mathcal{G},2} \le \lambda_n} \\ &= \min(\frac{\epsilon\lambda_n}{||x_g - a_g||_2}, 1)(x_g - a_g) + a_g \end{aligned} \tag{S:2.17}$$

We replace this in Eq. (S:2.15).

S:2.2 Close form solutions if incorporating Only Edge Or Only Node Group Knowledge

In cases, where we do not have superposition structures in the differential graph estimation, we can estimate the target Δ through a closed form solution, making the method scalable to larger p. In detail:

KDiffNet-E Only Edge-level Knowledge W_E: If additional knowledge is only available in the form of edge weights, the Eq. (S:2.1) reduces to :

$$\begin{aligned} &\underset{\Delta}{\text{argmin}}\, ||W_E \circ \Delta||_1 \\ &\text{s.t.: } ||(1 \oslash W_E) \circ \Big(\Delta - \Big([T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_c)]^{-1}\Big)\Big)\, ||_\infty \le \lambda_n \end{aligned} \tag{S:2.18}$$

This has a closed form solution:

$$\widehat{\Delta} = S_{\lambda_n * W_E}\Big(\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)\Big) \tag{S:2.19}$$

Here

$$[S_{\lambda_{ij} W_{E_{ij}}}(A)]_{ij} = \text{sign}(A_{ij}) \max(|A_{ij}| - \lambda_n W_{E_{i,j}}, 0) \tag{S:2.20}$$

KDiffNet-G Only Node Groups Knowledge G_V: If additional knowledge is only available in the form of groups of vertices $\mathcal{G}_V$, the Eq. (S:2.1) reduces to :

$$\begin{aligned} &\underset{\Delta}{\text{argmin}}\, ||\Delta||_{\mathcal{G}_V,2} \\ &\text{Subject to: } ||\Delta - \mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)||^*_{\mathcal{G}_V,2} \le \lambda_n \end{aligned} \tag{S:2.21}$$

Here, we assume nodes not in any group as individual groups with cardinality$= 1$. The closed form solution is given by:

$$\widehat{\Delta} = (S_{\mathcal{G}_V,\lambda_n}(\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c))) \tag{S:2.22}$$

Where $[S_{\mathcal{G},\lambda_n}(u)]_g = \max(||u_g||_2 - \lambda_n, 0)\frac{u_g}{||u_g||_2}$ and max is the element-wise max function.

Algorithm 1 shows the detailed steps of the KDiffNet estimator. Being non-iterative, the closed form solution helps KDiffNet achieve significant computational advantages.

Algorithm 1 KDiffNet-E and KDiffNet-G

```
input Two data matrices $\mathbf{X}_c$ and $\mathbf{X}_d$. The weight matrix $W_E$ OR $\mathcal{G}_\mathcal{V}$.
input Hyper-parameter: $\lambda_n$ and $v$
output $\Delta$
  Compute $[T_v(\widehat{\Sigma}_c)]^{-1}$ and $[T_v(\widehat{\Sigma}_d)]^{-1}$ from $\widehat{\Sigma}_c$ and $\widehat{\Sigma}_d$.
  Compute $\mathcal{B}^*(\widehat{\Sigma}_d, \widehat{\Sigma}_c)$.
  Compute $\widehat{\Delta}$ from Eq. (S:2.19) if $W_E$ only; else from Eq. (S:2.22) if only $\mathcal{G}_\mathcal{V}$
output $\widehat{\Delta}$
```

S:3 More Proof about kEV Norm and Its Dual Norm

S:3.1 Proof for kEV Norm is a norm

We reformulate kEV norm as

$$\mathcal{R}(\Delta) = ||W_E \circ \Delta_e||_1 + \epsilon||\Delta_g||_{\mathcal{G}_V,2} \tag{S:3.1}$$

to

$$\mathcal{R}(\Delta) = \mathcal{R}_1(\Delta) + \mathcal{R}_2(\Delta); \mathcal{R}_1(\cdot) = ||W_E \circ \cdot||_1; \mathcal{R}_2(\cdot) = \epsilon|| \cdot ||_{\mathcal{G}_{V,2}} \tag{S:3.2}$$

Theorem S:3.1. *kEV Norm is a norm if and only if $\mathcal{R}_1(\cdot)$ and $\mathcal{R}_2(\cdot)$ are norms.*

Proof. By the following Theorem S:3.3, $R_1(\cdot)$ is a norm. If $\epsilon > 0$, $R_2(\cdot)$ is a norm. Sum of two norms is a norm, hence kEV Norm is a norm. □

Lemma S:3.2. *For kEV-norm, $W_{E\,j,k} \neq 0$ equals to $W_{E\,j,k} > 0$.*

Proof. If $W_{E\,j,k} < 0$, then $|W_{E\,j,k}\Delta_{j,k}| = | - W_{E\,j,k}\Delta_{j,k}|$. Notice that $-W_{E\,j,k} > 0$. □

Theorem S:3.3. $\mathcal{R}_1(\cdot) = ||W_E \circ \cdot||_1$ *is a norm if and only if* $\forall 1 \geq j, k \leq p, W_{E\,jk} \neq 0$.

Proof. Proof. To prove the $\mathcal{R}_1(\cdot) = ||W_E \circ \cdot||_1$ is a norm, by Lemma (S:4.2) we need to prove 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, $\mathcal{R}_1(\cdot)$ is a norm function. □

□

S:3.2 kEV Norm is a decomposable norm

We show that kEV Norm is a decomposable norm within a certain subspace, with the following structural assumptions of the true parameter Δ^*:

(EV-Sparsity): The 'true' parameter of Δ^* can be decomposed into two clear structures–$\{{\Delta_e}^*$ and ${\Delta_g}^*\}$. ${\Delta_e}^*$ is exactly sparse with s non-zero entries indexed by a support set S_E and ${\Delta_g}^*$ is exactly sparse with $\sqrt{s_G}$ non-zero groups with atleast one entry non-zero indexed by a support set S_V. $S_E \bigcap S_V = \emptyset$. All other elements equal to 0 (in $(S_E \bigcup S_V)^c$).

Definition S:3.4. *(EV-subspace)*

$$\mathcal{M}(S_E \bigcup S_V) = \{\theta_j = 0 | \forall j \notin S_E \bigcup S_V\} \tag{S:3.3}$$

Theorem S:3.5. *kEV Norm is a decomposable norm with respect to $\mathcal{M}$ and $\bar{\mathcal{M}}^\perp$*

Proof. Assume $u \in \mathcal{M}$ and $v \in \bar{\mathcal{M}}^\perp$, $\mathcal{R}(u + v) = ||W_E \circ (u_e + v_e)||_1 + \epsilon||(u_g + v_g)||_{G_V,2} = ||W_E \circ u_e||_1 + ||W_E \circ v_e||_1 + \epsilon||u_g||_{G_V,2} + \epsilon||v_g||_{G_V,2} = \mathcal{R}(u) + \mathcal{R}(v)$. Therefore, kEV-norm is a decomposable norm with respect to the subspace pair $(\mathcal{M}, \bar{\mathcal{M}}^\perp)$. □

S:3.3 Proofs of Dual Norms for kEV Norm

Theorem S:3.6. *Dual Norm of kEV Norm is* $\mathcal{R}^*(u) = \max(||(1 \oslash W_E) \circ u||_\infty, \frac{1}{\epsilon}||u||^*_{\mathcal{G}_V,2})$.

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.

$$\begin{aligned}
\mathcal{R}^*(u) &= \sup_{\theta} \frac{<\theta, u>}{\mathcal{R}(\theta)} \\
&= \sup_{\theta_\alpha} \frac{\sum_\alpha <u, \theta_\alpha>}{\sum_\alpha c_\alpha \mathcal{R}_\alpha(\theta_\alpha)} \\
&= \sup_{\theta_\alpha} \frac{\sum_\alpha <u/c_\alpha, \theta_\alpha>}{\sum_\alpha \mathcal{R}_\alpha(\theta_\alpha)} \\
&\le \sup_{\theta_\alpha} \frac{\sum_\alpha \mathcal{R}^*_\alpha(u/c_\alpha)\mathcal{R}(\theta_\alpha)}{\sum_\alpha \mathcal{R}_\alpha(\theta_\alpha)} \\
&\le \max_{\alpha \in I} \mathcal{R}^*_\alpha(u)/c_\alpha.
\end{aligned} \tag{S:3.4}$$

Connecting $\mathcal{R}_1(\cdot) = ||W_E \cdot||_1$ and $\mathcal{R}_2(\cdot) = \epsilon||\cdot||_{\mathcal{G}_V}$. By the following Theorem S:3.7, $\mathcal{R}^*_1(u) = ||(1 \oslash W_E) \circ u||_\infty$. From [13], for $\mathcal{R}_2(\theta_2) = ||\Delta||_{\mathcal{G}_V,2}$, the dual norm is given by

$$||v||_{\mathcal{G},\vec{\alpha}^*} \quad = \max_{t=1,\ldots,s_\mathcal{G}} ||v||_{\alpha^*_t} \tag{S:3.5}$$

where $\frac{1}{\alpha_t} + \frac{1}{\alpha^*_t} = 1$ are dual exponents. where $s_\mathcal{G}$ denotes the number of groups. As special cases of this general duality relation, this leads to a block $(\infty, 2)$ norm as the dual.
Hence, $\mathcal{R}^*_2(u) = ||u||^*_{\mathcal{G}_V,2}$. Hence, the dual norm of kEV norm is $\mathcal{R}^*(u) = \max(||(1 \oslash W_E) \circ u||_\infty, \frac{||u||^*_{\mathcal{G}_\mathcal{V},2}}{\epsilon})$. □

Theorem S:3.7. *The dual norm of* $||W_E \circ \cdot||_1$ *is:*

$$\mathcal{R}^*_1(\cdot) = ||(1 \oslash W_E) \circ u||_\infty \tag{S:3.6}$$

For $\mathcal{R}_1(\cdot) = ||W_E \circ ||_1$, the dual norm is given by:

$$\begin{aligned}
&\sup_{||W \circ u||_1 \le 1} \quad u^T x \\
&\le \sup_{||W \circ u||_1 \le 1} \sum_{k=1}^{p} |u_k||x_k| \\
&= \sup_{||W \circ u||_1 \le 1} \sum_{k=1}^{p} \frac{|u_k||x_k||w_k|}{|w_k|} \\
&= \sup_{||W \circ u||_1 \le 1} \sum_{k=1}^{p} |w_k u_k| \left|\frac{x_k}{w_k}\right| \\
&\le \sup_{||W \circ u||_1 \le 1} \left(\sum_{k=1}^{p} |w_k u_k|\right) \max_{k=1,\ldots,p} \left|\frac{x_k}{w_k}\right| \\
&= \left|\left|\frac{x}{w}\right|\right|_\infty
\end{aligned} \tag{S:3.7}$$

S:4 Appendix: More Background of Proxy Backward mapping and Theorems of T_v Being Invertible

Essentially the MLE solution of estimating vanilla graphical model in an exponential family distribution can be expressed as a backward mapping that computes the target model parameters from certain given moments. For instance, when learning Gaussian GM with vanilla MLE, the backward mapping is $\widehat{\Sigma}^{-1}$ that estimates Ω from the sample covariance matrix (moment) $\widehat{\Sigma}$. However, this backward mapping is normally not well-defined in high-dimensional settings. In the case of GGM, when given the sample covariance $\widehat{\Sigma}$, we cannot just compute the vanilla MLE solution as $[\widehat{\Sigma}]^{-1}$ when high-dimensional since $\widehat{\Sigma}$ is rank-deficient when $p > n$. Therefore Yang et al. [24] proposed to use carefully constructed proxy backward maps for Eq. (2.4) that are both available in closed-form, and well-defined in high-dimensional settings for exponential GM models. For instance, $[T_v(\widehat{\Sigma})]^{-1}$ in Eq. (2.3) is the proxy backward mapping [24] used for GGM.

S:4.1 Backward mapping for an exponential-family distribution:

The solution of vanilla graphical model MLE can be expressed as a backward mapping[21] 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:

$$\mathbb{P}(X;\theta) = h(X)\text{exp}\{<\theta, \phi(\theta) > -A(\theta)\} \tag{S:4.1}$$

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:

$$\mathcal{M} = \{v | \exists p \text{ is a distribution s.t. } \mathbb{E}_p[\phi(X)] = v\} \tag{S:4.2}$$

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

$$\mathcal{A} : \textcircled{H} \to \mathcal{M} \tag{S:4.3}$$

The learning/estimation of graphical models involves the task of the reverse computing of the forward mapping, the so-called **backward mapping** [21]. We denote the interior of $\mathcal{M}$ as $\mathcal{M}^0$. **backward mapping** is defined as:

$$\mathcal{A}^* : \mathcal{M}^0 \to \textcircled{H} \tag{S:4.4}$$

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

$$\mathcal{A}^* : v(\theta) \to \theta = \nabla A^*(v(\theta)). \tag{S:4.5}$$

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

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

$$A(\theta) = \frac{1}{2}\mu^T\Sigma^{-1}\mu + \frac{1}{2}\log(|\Sigma|) \tag{S:4.6}$$

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}.

$$\begin{aligned} \theta &= (\Sigma^{-1}\mu, -\frac{1}{2}\Sigma^{-1}) = \mathcal{A}^*(v) = \nabla A^*(v) \\ &= ((\mathbb{E}_\theta[XX^T] - \mathbb{E}_\theta[X]\mathbb{E}_\theta[X]^T)^{-1}\mathbb{E}_\theta[X], \\ &\quad -\frac{1}{2}(\mathbb{E}_\theta[XX^T] - \mathbb{E}_\theta[X]\mathbb{E}_\theta[X]^T)^{-1}). \end{aligned} \tag{S:4.7}$$

By plugging in Eq. (S:4.6) into Eq. (S:4.5), 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.

S:4.2 Backward Mapping for Differential GGM

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 [12]) essentially is a distribution in exponential families:

$$\begin{aligned} r(x, \Delta) &= \frac{p_d(x)}{p_c(x)} \\ &= \frac{\sqrt{\det(\Sigma_c)}\exp\left(-\frac{1}{2}(x-\mu_d)^T\Sigma_d^{-1}(x-\mu_d)\right)}{\sqrt{\det(\Sigma_d)}\exp\left(-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)\right)} \\ &= \exp(-\frac{1}{2}(x-\mu_d)^T\Sigma_d^{-1}(x-\mu_d) \\ &+ \frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c) \\ &- \frac{1}{2}(\log(\det(\Sigma_d)) - \log(\det(\Sigma_c)))) \\ &= \exp\left(-\frac{1}{2}\Delta x^2 + \mu_\Delta x - A(\mu_\Delta, \Delta)\right) \end{aligned} \tag{S:4.8}$$

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

$$\begin{aligned} A(\mu_\Delta, \Delta) = \frac{1}{2}\mu_d^T\Sigma_d^{-1}\mu_d - \frac{1}{2}\mu_c^T\Sigma_c^{-1}\mu_c + \\ \frac{1}{2}\log(\det(\Sigma_d)) - \frac{1}{2}\log(\det(\Sigma_c)) \end{aligned} \tag{S:4.9}$$

The canonical parameter

$$\begin{aligned} \theta &= \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Sigma_d^{-1} - \Sigma_c^{-1})\right) \\ &= \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Delta)\right) \end{aligned} \tag{S:4.10}$$

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

$$\begin{aligned} &\phi([X_c, X_d]) = ([X_c, X_d], [X_c X_c^T, X_d X_d^T]) \\ &A(\theta) = \frac{1}{2}\mu_d^T\Sigma_d^{-1}\mu_d - \frac{1}{2}\mu_c^T\Sigma_c^{-1}\mu_c + \\ &\frac{1}{2}\log(\det(\Sigma_d)) - \frac{1}{2}\log(\det(\Sigma_c)) \end{aligned} \tag{S:4.11}$$

And $h(x) = 1$.

Now we can estimate this exponential distribution (θ) through vanilla MLE. By plugging Eq. (S:4.11) into Eq. (S:4.5), we get the following backward mapping via the conjugate of the log-partition function:

$$\begin{aligned} \theta &= \left(\Sigma_d^{-1}\mu_d - \Sigma_c^{-1}\mu_c, -\frac{1}{2}(\Sigma_d^{-1} - \Sigma_c^{-1})\right) \\ &= \mathcal{A}^*(v) = \nabla A^*(v) \end{aligned} \tag{S:4.12}$$

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,

$$\begin{aligned} \widehat{\theta} =& (((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1}\mathbb{E}_\theta[X_d] \\ &- (\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1}\mathbb{E}_\theta[X_c]), \\ &- \frac{1}{2}((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1} - \\ &(\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1})). \end{aligned} \tag{S:4.13}$$

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

$$\begin{aligned}&((\mathbb{E}_\theta[X_d X_d^T] - \mathbb{E}_\theta[X_d]\mathbb{E}_\theta[X_d]^T)^{-1}\\ &-(\mathbb{E}_\theta[X_c X_c^T] - \mathbb{E}_\theta[X_c]\mathbb{E}_\theta[X_c]^T)^{-1})\\ &=\widehat{\Sigma}_d^{-1} - \widehat{\Sigma}_c^{-1}\end{aligned} \tag{S:4.14}$$

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

S:4.3 Theorems of Proxy Backward Mapping T_v Being Invertible

Based on [24] for any matrix A, the element wise operator T_v is defined as:

$$[T_v(A)]_{ij} = \begin{cases} A_{ii} + v & if\ i = j \\ sign(A_{ij})(|A_{ij}| - v) & otherwise, i \neq j \end{cases}$$

Suppose we apply this operator T_v to the sample covariance matrix $\frac{X^T X}{n}$ to obtain $T_v(\frac{X^T X}{n})$. Then, $T_v(\frac{X^T X}{n})$ under high dimensional settings will be invertible with high probability, under the following conditions:

Condition-1 (Σ-Gaussian ensemble) Each row of the design matrix $X \in \mathbb{R}^{n \times p}$ is i.i.id sampled from $N(0, \Sigma)$.

Condition-2 The covariance Σ of the Σ-Gaussian ensemble is strictly diagonally dominant: for all row i, $\delta_i := \Sigma_{ii} - \Sigma_{j \neq i} \geq \delta_{min} > 0$ where δ_{min} is a large enough constant so that $||\Sigma||\infty \leq \frac{1}{\delta_{min}}$.

This assumption guarantees that the matrix $T_v(\frac{X^T X}{n})$ is invertible, and its induced ℓ_∞ norm is well bounded. Then the following theorem holds:

Theorem S:4.1. *Suppose Condition-1 and Condition-2 hold. Then for any* $v \geq 8(max_i\Sigma_{ii})\sqrt{(\frac{10\tau \log p'}{n})}$*, the matrix* $T_v(\frac{X^T X}{n})$ *is invertible with probability at least* $1 - 4/p'^{\tau-2}$ *for* $p' := max\{n, p\}$ *and any constant* $\tau > 2$.

S:4.4 Useful lemma(s) of Error Bounds of Proxy Backward Mapping T_v

Lemma S:4.2. *(Theorem 1 of [16]). Let* δ *be* $\max_{ij} ||[\frac{X^T X}{n}]_{ij} - \Sigma_{ij}|$*. Suppose that* $\nu > 2\delta$*. Then, under the conditions (C-Sparse*Σ*), and as* $\rho_v(\cdot)$ *is a soft-threshold function, we can deterministically guarantee that the spectral norm of error is bounded as follows:*

$$|||T_v(\widehat{\Sigma}) - \Sigma|||_\infty \leq 5\nu^{1-q}c_0(p) + 3\nu^{-q}c_0(p)\delta \tag{S:4.15}$$

Lemma S:4.3. *(Lemma 1 of [15]). Let* $\mathcal{A}$ *be the event that*

$$||\frac{X^T X}{n} - \Sigma||_\infty \leq 8(\max_i \Sigma_{ii})\sqrt{\frac{10\tau \log p'}{n}} \tag{S:4.16}$$

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* $1 - 4/p'^{\tau-2}$.

S:5 Theoretical Analysis of Error Bounds

S:5.1 Background: Error bounds of Elementary Estimators

KDiffNet formulations are special cases of the following generic formulation for the elementary estimator.

$$\begin{gathered}\operatorname*{argmin}_\theta \mathcal{R}(\theta)\\ \text{subject to:}\mathcal{R}^*(\theta - \widehat{\theta}_n) \leq \lambda_n\end{gathered} \tag{S:5.1}$$

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

$$\mathcal{R}^*(v) := \sup_{u \neq 0} \frac{< u, v >}{\mathcal{R}(u)} = \sup_{\mathcal{R}(u) \leq 1} < u, v > . \tag{S:5.2}$$

Following the unified framework [13], 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. (S:5.1) is **decomposable** w.r.t the subspace pair $(\mathcal{M}, \bar{\mathcal{M}}^{\perp})$.

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

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

Definition S:5.1. ***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 S:5.2. *Suppose the regularization function in Eq. (S:5.1) satisfies condition* ***(C1)****, the true parameter of Eq. (S:5.1) 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. (S:5.1) satisfies:*

$$\mathcal{R}^*(\widehat{\theta} - \theta^*) \leq 2\lambda_n \tag{S:5.3}$$

$$||\widehat{\theta} - \theta^*||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}) \tag{S:5.4}$$

$$\mathcal{R}(\widehat{\theta} - \theta^*) \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2 \tag{S:5.5}$$

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

$$\begin{aligned} \mathcal{R}^*(\widehat{\theta} - \theta^*) &= \mathcal{R}^*(\widehat{\theta} - \widehat{\theta}_n + \widehat{\theta}_n - \theta^*) \\ &\leq \mathcal{R}^*(\widehat{\theta}_n - \widehat{\theta}) + \mathcal{R}^*(\widehat{\theta}_n - \theta^*) \leq 2\lambda_n \end{aligned} \tag{S:5.6}$$

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

$$\begin{aligned} &\mathcal{R}(\theta^*) \\ &= \mathcal{R}(\theta^*) + \mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \\ &= \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \\ &\leq \mathcal{R}[\theta^* + \Pi_{\bar{\mathcal{M}}^{\perp}}(\delta) + \Pi_{\bar{\mathcal{M}}}(\delta)] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \\ &- \mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \\ &= \mathcal{R}[\theta^* + \delta] + \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] - \mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \end{aligned} \tag{S:5.7}$$

Here, the inequality holds by the triangle inequality of norm. Since Eq. (S:5.1) minimizes $\mathcal{R}(\widehat{\theta})$, we have $\mathcal{R}(\theta^* + \Delta) = \mathcal{R}(\widehat{\theta}) \leq \mathcal{R}(\theta^*)$. Combining this inequality with Eq. (S:5.7), we have:

$$\mathcal{R}[\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta)] \leq \mathcal{R}[\Pi_{\bar{\mathcal{M}}}(\delta)] \tag{S:5.8}$$

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

$$\begin{aligned} &||\Delta||_2^2 = \langle \delta, \delta \rangle \leq \mathcal{R}^*(\delta)\mathcal{R}(\delta) \leq 2\lambda_n \mathcal{R}(\delta) \\ &= 2\lambda_n[\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) + \mathcal{R}(\Pi_{\bar{\mathcal{M}}^{\perp}}(\delta))] \leq 4\lambda_n \mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 4\lambda_n \Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \end{aligned} \tag{S:5.9}$$

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. (S:5.9), we have:

$$||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 4\lambda_n \Psi(\bar{\mathcal{M}}), \tag{S:5.10}$$

and plugging it back to Eq. (S:5.9) yields the error bound Eq. (S:5.4).

Finally, Eq. (S:5.5) is straightforward from Eq. (S:5.8) and Eq. (S:5.10).

$$\begin{aligned} \mathcal{R}(\delta) &\leq 2\mathcal{R}(\Pi_{\bar{\mathcal{M}}}(\delta)) \\ &\leq 2\Psi(\bar{\mathcal{M}})||\Pi_{\bar{\mathcal{M}}}(\delta)||_2 \leq 8\lambda_n \Psi(\bar{\mathcal{M}})^2. \end{aligned} \tag{S:5.11}$$

□

S:5.2 Error Bounds of KDiffNet

Theorem S:5.2, provides the error bounds via λ_n with respect to three different metrics. In the following, we focus on one of the metrics, Frobenius Norm to evaluate the convergence rate of our KDiffNet estimator.

S:5.2.1 Error Bounds of KDiffNet through λ_n and ϵ

Theorem S:5.3. *Assuming the true parameter Δ^* satisfies the conditions **(C1)(C2)** and $\lambda_n \geq \mathcal{R}^*(\widehat{\Delta} - \Delta^*)$, then the optimal point $\widehat{\Delta}$ has the following error bounds:*

$$||\widehat{\Delta} - \Delta^*||_F \leq (4\max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}), \epsilon\sqrt{s_G})\lambda_n \tag{S:5.12}$$

Proof: KDiffNet uses $\mathcal{R}(\cdot) = ||W_E \circ \cdot||_1 + \epsilon||\cdot||_{\mathcal{G},2}$ because it is a superposition of two norms: $\mathcal{R}_1 = ||W_E \circ ||_1$ and $\mathcal{R}_2 = \epsilon||\cdot||_{\mathcal{G},2}$. Based on the results in[13], $\Psi(\bar{\mathcal{M}}_1) = \max_{i,j}(W_{E_{i,j}})\sqrt{s}$ and $\Psi(\bar{\mathcal{M}}_2) = \sqrt{s_{\mathcal{G}}}$, where s is the number of nonzero entries in Δ and $s_{\mathcal{G}}$ is the number of groups in which there exists at least one nonzero entry. Therefore, $\Psi(\bar{\mathcal{M}}) = \max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}), \epsilon\sqrt{s_G})$. Hence,Using this in Equation Eq. (S:5.4), $||\widehat{\Delta} - \Delta^*||_F \leq 4(\max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}), \epsilon\sqrt{s_G})\lambda_n$.

S:5.2.2 Proof of Corollary (2.1)-Derivation of the KDiffNet error bounds

To derive the convergence rate for KDiffNet , we introduce the following two sufficient conditions on the Σ_c and Σ_d, to show that the proxy backward mapping $\widehat{\theta}_n = B^*(\widehat{\phi}) = [T_v(\widehat{\Sigma}_d)]^{-1} - [T_v(\widehat{\Sigma}_d)]^{-1}$ is well-defined[23]:
(C-MinInf$-\Sigma$): The true Ω_c^* and Ω_d^* of Eq. (2.1) have bounded induced operator norm, i.e., $|||\Omega_c{}^*|||_\infty := \sup\limits_{w\neq 0\in\mathbb{R}^p} \frac{||\Sigma_c{}^* w||_\infty}{||w||_\infty} \leq \kappa_1$ and $|||\Omega_d{}^*|||_\infty := \sup\limits_{w\neq 0\in\mathbb{R}^p} \frac{||\Sigma_d{}^* w||_\infty}{||w||_\infty} \leq \kappa_1$.

(C-Sparse-Σ): The two true covariance matrices Σ_c^* and Σ_d^* are “approximately sparse” (following [2]). For some constant $0 \leq q < 1$ and $c_0(p)$, $\max\limits_i \sum\limits_{j=1}^p |[\Sigma_c^*]_{ij}|^q \leq c_0(p)$ and $\max\limits_i \sum\limits_{j=1}^p |[\Sigma_d^*]_{ij}|^q \leq c_0(p)$. [2]

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

We assume the true parameters Ω_c^* and Ω_d^* satisfies **C-MinInf**Σ and **C-Sparse**Σ conditions.

Using the above theorem and conditions, we have the following corollary for convergence rate of KDiffNet (Att: the following corollary is the same as the Corollary 2.1 in the main draft. We repeat it here to help readers read the manuscript more easily):

Corollary S:5.4. *In 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 $\lambda_n := \frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p}{\min(n_c,n_d)}}$ and $\min(n_c, n_d) > c\log p$, with a probability of at least*

[2] 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.

$1-2C_1\exp(-C_2 p\log(p))$, *the estimated optimal solution* $\widehat{\Delta}$ *has the following error bound:*

$$||\widehat{\Delta}-\Delta^*||_F \le \frac{16\kappa_1 a((\max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}),\epsilon\sqrt{s_G})}{\min_{i,j}(W_{E_{i,j}})\kappa_2}\sqrt{\frac{\log p}{\min(n_c,n_d)}} \tag{S:5.13}$$

where a, c, κ_1 and κ_2 are constants.

Proof. In the following proof, we first prove $||\Omega_c^*-[T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \le \lambda_{n_c}$. 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 (S:5.3) for this specific case, we only need to show that $||\Omega_c^*-[T_v(\widehat{\Sigma}_c)]^{-1}||_\infty \le \lambda_{n_c}$ for the setting of $\lambda_{n_c}=\frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}}$:

$$\begin{aligned}
&||\Omega_c^*-[T_v(\widehat{\Sigma}_c)]^{-1}||_\infty = ||[T_v(\widehat{\Sigma}_c)]^{-1}(T_v(\widehat{\Sigma}_c)\Omega_c^*-I)||_\infty\\
&\le |||[T_v(\widehat{\Sigma}_c)w]|||_\infty ||T_v(\widehat{\Sigma}_c)\Omega_c^*-I||_\infty\\
&= |||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty ||\Omega_c^*(T_v(\widehat{\Sigma}_c)-\Sigma_c^*)||_\infty\\
&\le |||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty |||\Omega_c^*|||_\infty ||T_v(\widehat{\Sigma}_c)-\Sigma_c^*||_\infty.
\end{aligned} \tag{S:5.14}$$

We first compute the upper bound of $|||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty$. By the selection v in the statement, Lemma (S:4.2) and Lemma (S:4.3) hold with probability at least $1-4/p'^{\tau-2}$. Armed with Eq. (S:4.15), we use the triangle inequality of norm and the condition (C-SparseΣ): for any w,

$$\begin{aligned}
&||T_v(\widehat{\Sigma}_c)w||_\infty = ||T_v(\widehat{\Sigma}_c)w-\Sigma w+\Sigma w||_\infty\\
&\ge ||\Sigma w||_\infty - ||(T_v(\widehat{\Sigma}_c)-\Sigma)w||_\infty\\
&\ge \kappa_2||w||_\infty - ||(T_v(\widehat{\Sigma}_c)-\Sigma)w||_\infty\\
&\ge (\kappa_2-||(T_v(\widehat{\Sigma}_c)-\Sigma)w||_\infty)||w||_\infty
\end{aligned} \tag{S:5.15}$$

Where the second inequality uses the condition (C-SparseΣ). Now, by Lemma (S:4.2) with the selection of v, we have

$$|||T_v(\widehat{\Sigma}_c)-\Sigma|||_\infty \le c_1(\frac{\log p'}{n_c})^{(1-q)/2}c_0(p) \tag{S:5.16}$$

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 $n_c > (\frac{2c_1c_0(p)}{\kappa_2})^{\frac{2}{1-q}}\log p'$ as stated, so that $|||T_v(\widehat{\Sigma}_c)-\Sigma|||_\infty \le \frac{\kappa_2}{2}$, we can conclude that $||T_v(\widehat{\Sigma}_c)w||_\infty \ge \frac{\kappa_2}{2}||w||_\infty$, which implies $|||[T_v(\widehat{\Sigma}_c)]^{-1}|||_\infty \le \frac{2}{\kappa_2}$.

The remaining term in Eq. (S:5.14) is $||T_v(\widehat{\Sigma}_c)-\Sigma_c^*||_\infty$; $||T_v(\widehat{\Sigma}_c)-\Sigma_c^*||_\infty \le ||T_v(\widehat{\Sigma}_c)-\widehat{\Sigma}_c||_\infty + ||\widehat{\Sigma}_c-\Sigma_c^*||_\infty$. By construction of $T_v(\cdot)$ in (C-Thresh) and by Lemma (S:4.3), 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,

$$||(1\oslash W_E)\circ\left(\Delta^*-\left([T_v(\widehat{\Sigma}_d)]^{-1}-[T_v(\widehat{\Sigma}_c)]^{-1}\right)\right)||_\infty \tag{S:5.17}$$

$$\le||(1\oslash W_E)\circ\left(\Omega_d-[T_v(\widehat{\Sigma}_d)]^{-1}\right)||_\infty + ||(1\oslash W_E)\circ\left(\Omega_c-[T_v(\widehat{\Sigma}_c)]^{-1}\right)||_\infty \tag{S:5.18}$$

$$\le\frac{1}{\min_{i,j}W_{i,j}}\left(\frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_c}}+\frac{4\kappa_1 a}{\kappa_2}\sqrt{\frac{\log p'}{n_d}}\right) \tag{S:5.19}$$

Because by Theorem S:5.3, we know if $\lambda_n \ge \mathcal{R}^*(\widehat{\Delta}-\Delta^*)$,

$$||\widehat{\Delta}-\Delta^*||_F \le (4\max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}),\epsilon\sqrt{s_G})\lambda_n$$

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

$$||\widehat{\Delta} - \Delta^*||_F \leq \frac{16\kappa_1 a \max(\max_{i,j}(W_{E_{i,j}})\sqrt{s}), \epsilon\sqrt{s_G})}{\min_{i,j}(W_{E_{i,j}})\kappa_2}\sqrt{\frac{\log p}{\min(n_c, n_d)}} \tag{S:5.20}$$

By combining all together, we can confirm that the selection of λ_n satisfies the requirement of Theorem (S:5.3), which completes the proof. □

S:6 More about Experiments

S:6.1 Experimental Setup

The hyper-parameters in our experiments are v, λ_n, ϵ and λ_2. In detail:

- To compute the proxy backward mapping in (S:2.1), DIFFEE, and JEEK we vary v for soft-thresholding v from the set $\{0.001i|i = 1, 2, \ldots, 1000\}$ (to make $T_v(\Sigma_c)$ and $T_v(\Sigma_d)$ invertible).
- λ_n is the hyper-parameter in our KDiffNet formulation. According to our convergence rate analysis in Section 2.6, $\lambda_n \geq C\sqrt{\frac{\log p}{\min(n_c,n_d)}}$, we choose λ_n from a range of $\{0.01 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i|i \in \{1, 2, 3, \ldots, 100\}\}$. For KDiffNet-G case, we tune over λ_n from a range of $\{0.1 \times \sqrt{\frac{\log p}{\min(n_c,n_d)}} \times i|i \in \{1, 2, 3, \ldots, 100\}\}$. We use the same range to tune λ_1 for SDRE. Tuning for NAK is done by the package itself.
- ϵ: For KDiffNet-EG experiments, we tune $\epsilon \in \{0.0001, 0.01, 1, 100\}\}$.
- λ_2 controls individual graph's sparsity in JGLFUSED. We choose $\lambda_1 = 0.0001$ (a very small value) for all experiments to ensure only the differential network is sparse.

Evaluation Metrics:

- F1-score: We use the edge-level F1-score as a measure of the performance of each method. F1 $= \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$, where Precision $= \frac{\text{TP}}{\text{TP+FP}}$ and Recall $= \frac{\text{TP}}{\text{TP+FN}}$. The better method achieves a higher F1-score. We choose the best performing λ_n using validation and report the performance on a test dataset.

- Time Cost: We use the execution time (measured in seconds or log(seconds)) for a method as a measure of its scalability. The better method uses less time[3]

S:6.2 Simulation Dataset Generation

We first use simulation to evaluate KDiffNet for improving differential structure estimation by making use of extra knowledge. We generate simulated datasets with a clear underlying differential structure between two conditions, using the following method:

Data Generation for Edge Knowledge (KE): Given a known weight matrix W_E (e.g., spatial distance matrix between p brain regions), we set $W^d = inv.logit(-W_E)$. We use the assumption that higher the value of W_{ij}, lower the probability of that edge to occur in the true precision matrix. This is motivated by the role of spatial distance in brain connectivity networks: farther regions are less likely to be connected and vice-versa. We select different levels in the matrix W^d, denoted by s, where if $W^d_{ij} > s_l$, $\Delta^d_{ij} = 0.5$, else $\Delta^d_{ij} = 0$, where $\Delta^d \in \mathbb{R}^{p\times p}$. We denote by s as the sparsity, i.e. the number of non-zero entries in Δ^d. B_I is a random graph with each edge $B_{I_{ij}} = 0.5$ with probability p. δ_c and δ_d are selected large enough to guarantee positive definiteness.

$$\Omega_d = \Delta^d + B_I + \delta_d I \tag{S:6.1}$$

$$\Omega_c = B_I + \delta_c I \tag{S:6.2}$$

$$\Delta = \Omega_d - \Omega_c \tag{S:6.3}$$

There is a clear differential structure in $\Delta = \Omega_d - \Omega_c$, controlled by Δ^d. To generate data from two conditions that follows the above differential structure, we generate two blocks of data samples following Gaussian distribution using $N(0, \Omega_c^{-1})$ and $N(0, \Omega_d^{-1})$. We only use these data samples to approximate the differential GGM to compare to the ground truth Δ.

[3] The machine that we use for experiments is an Intel Core i7 CPU with a 16 GB memory.

Data Generation for Vertex Knowledge (KG): In this case, we simulate the case of extra knowledge of nodes in known groups. Let the node group size,i.e., the number of nodes with a similar interaction pattern in the differential graph be m. We select the block diagonals of size m as groups in Δ^g. If two variables i, j are in a group g', in $\Delta^g_{ij} = 0.5$, else $\Delta^g_{ij} = 0$, where $\Delta^g \in \mathbb{R}^{p \times p}$. We denote by s_G as the number of groups in Δ^g. B_I is a random graph with each edge $B_{I_{ij}} = 0.5$ with probability p.

$$\Omega_d = \Delta^g + B_I + \delta_d I \tag{S:6.4}$$

$$\Omega_c = B_I + \delta_c I \tag{S:6.5}$$

$$\Delta = \Omega_d - \Omega_c \tag{S:6.6}$$

δ_c and δ_d are selected large enough to guarantee positive definiteness. We generate two blocks of data samples following Gaussian distribution using $N(0, \Omega_c^{-1})$ and $N(0, \Omega_d^{-1})$.

Data Generation for both Edge and Vertex Knowledge (KEG): In this case, we simulate the case of overlapping group and edge knowledge. Let the node group size,i.e., the number of nodes with a similar interaction pattern in the differential graph be m. We select the block diagonals of size m as groups in Δ^g. If two variables i, j are in a group g', in $\Delta^g_{ij} = 1/3$, else $\Delta^g_{ij} = 0$, where $\Delta^g \in \mathbb{R}^{p \times p}$.

For the edge-level knowledge component, given a known weight matrix W_E, we set $W^d = inv.logit(-W_E)$. Higher the value of $W_{E_{ij}}$, lower the value of W^d_{ij}, hence lower the probability of that edge to occur in the true precision matrix. We select different levels in the matrix W^d, denoted by s, where if $W^d_{ij} > s_l$, we set $\Delta^d_{ij} = 1/3$, else $\Delta^d_{ij} = 0$. We denote by s as the number of non-zero entries in Δ^d. B_I is a random graph with each edge $B_{I_{ij}} = 1/3$ with probability p.

$$\Omega_d = \Delta^d + \Delta^g + B_I + \delta_d I \tag{S:6.7}$$

$$\Omega_c = B_I + \delta_c I \tag{S:6.8}$$

$$\Delta = \Omega_d - \Omega_c \tag{S:6.9}$$

δ_c and δ_d are selected large enough to guarantee positive definiteness. Similar to the previous case, we generate two blocks of data samples following Gaussian distribution using $N(0, \Omega_c^{-1})$ and $N(0, \Omega_d^{-1})$. We only use these data samples to approximate the differential GGM to compare to the ground truth Δ.

S:6.3 Simulation Experiment Results

We consider three different types of known edge knowledge W_E generated from the spatial distance between different brain regions and simulate groups to represent related anatomical regions. These three are distinguished by different $p = \{116, 160, 246\}$ representing spatially related brain regions. We generate three types of datasets:Data-EG (having both edge and vertex knowledge), Data-G(with edge-level extra knowledge) and Data-V(with known node groups knowledge). We generate two blocks of data samples $\boldsymbol{X}_c$ and $\boldsymbol{X}_d$ following Gaussian distribution using $N(0, \Omega_c^{-1})$ and $N(0, \Omega_d^{-1})$. We use these data samples to estimate the differential GGM to compare to the ground truth Δ. The details of the simulation are in Section S:6.2. We vary the sparsity of the true differential graph (s) and the number of control and case samples (n_c and n_d respectively) used to estimate the differential graph. For each case of p, we vary n_c and n_d in $\{p/2, p/4, p, 2p\}$ to account for both high dimensional and low dimensional cases. The sparsity of the underlying differential graph is controlled by $s = \{0.125, 0.25, 0.375, 0.5\}$ and s_G as explained in Section S:6.2. This results in 126 different datasets representing diverse settings: different number of dimensions p, number of samples n_c and n_d, multiple levels of sparsity s and number of groups s_G of the differential graph for both KE and KEG data settings.

Edge and Vertex Knowledge (KEG): We use KDiffNet (Algorithm 1) to infer the differential structure in this case.

Figure S:2(a) shows the performance in terms of F1 Score of KDiffNet in comparison to the baselines for $p = 116$, corresponding to 116 regions of the brain. KDiffNet outperforms the best baseline in

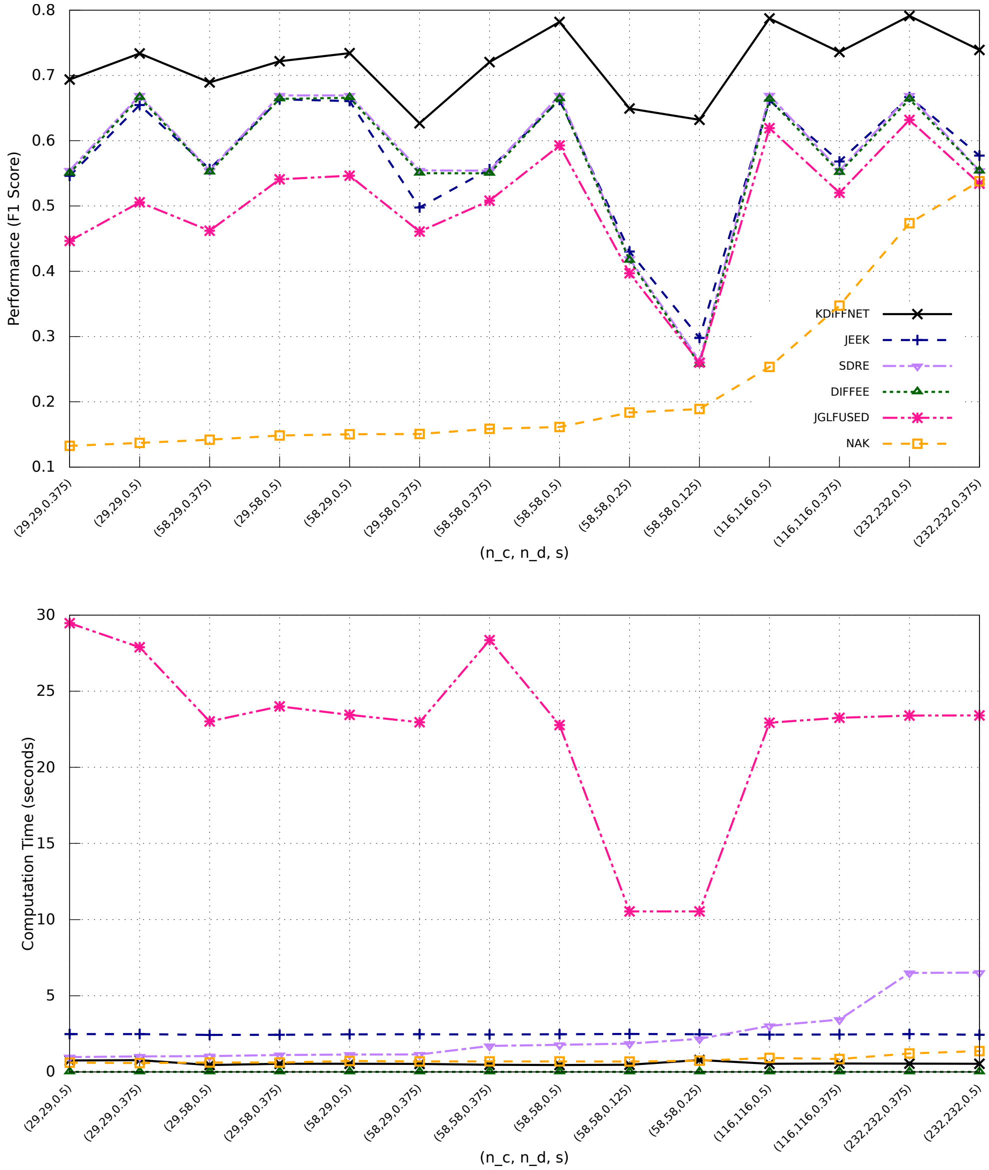

Figure S:2: KDiffNet Edge and Vertex Knowledge Simulation Results for $p = 116$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet and baseline methods.

each case by an average improvement of 414%. KDiffNet-EG does better than JEEK and NAK that can model the edge information but cannot include group information. SDRE and DIFFEE are direct estimators but perofrm poorly indicating that adding additional knowledge aids differential network estimation. JGLFUSED performs the worst on all cases. We list the detailed results in Section S:6.5.

Figure S:2(b) shows the average computation cost per λ_n of each method measured in seconds. In all settings, KDiffNet has lower computation cost than JEEK, SDRE and JGLFUSED in different cases of varying n_c and n_d, as well as with different sparsity of the differential network. KDiffNet is on average $24\times$ faster than the best performing baseline. It is slower than DIFFEE owing to DIFFEE's non-iterative closed form solution, however, DIFFEE does not have good prediction performance. Note that $B^*()$ in KDiffNet , JEEK and DIFFEE and the kernel term in SDRE are precomputed only once prior to tuning across multiple λ_n. In Figure S:3(a), we plot the test F1-score for simulated

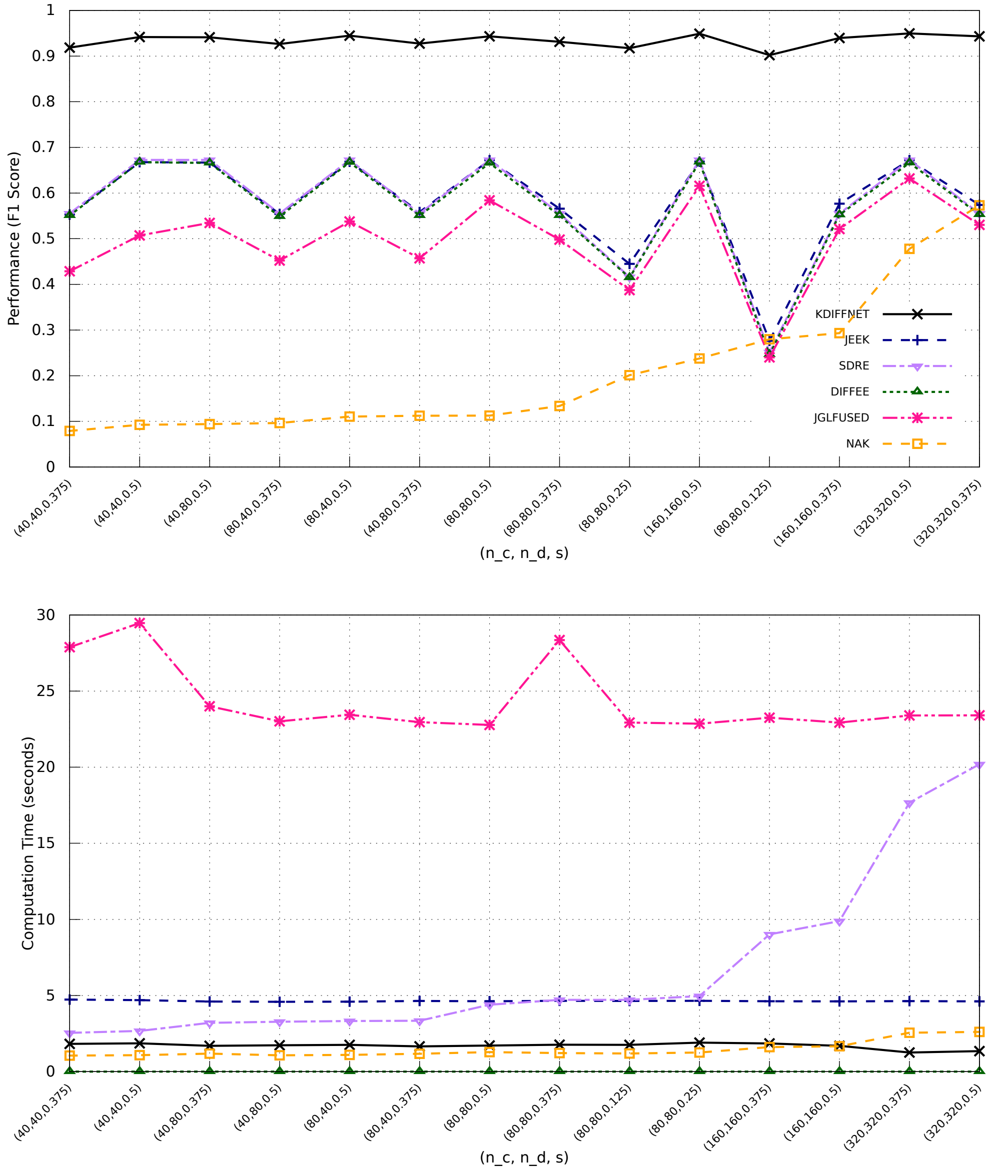

Figure S:3: KDiffNet Edge and Vertex Knowledge Simulation Results for $p = 160$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet and baseline methods.

datasets generated using W with $p = 160$, representing spatial distances between different 160 regions of the brain. This represents a larger and different set of spatial brain regions. In $p = 160$ case, KDiffNet outperforms the best baseline in each case by an average improvement of 928%. Including available additional knowledge is clearly useful as JEEK does relatively better than the other baselines. JGLFUSED performs the worst on all cases. Figure S:3(b) shows the computation cost of each method measured in seconds for each case. KDiffNet is on average $37\times$ faster than the best performing baseline.

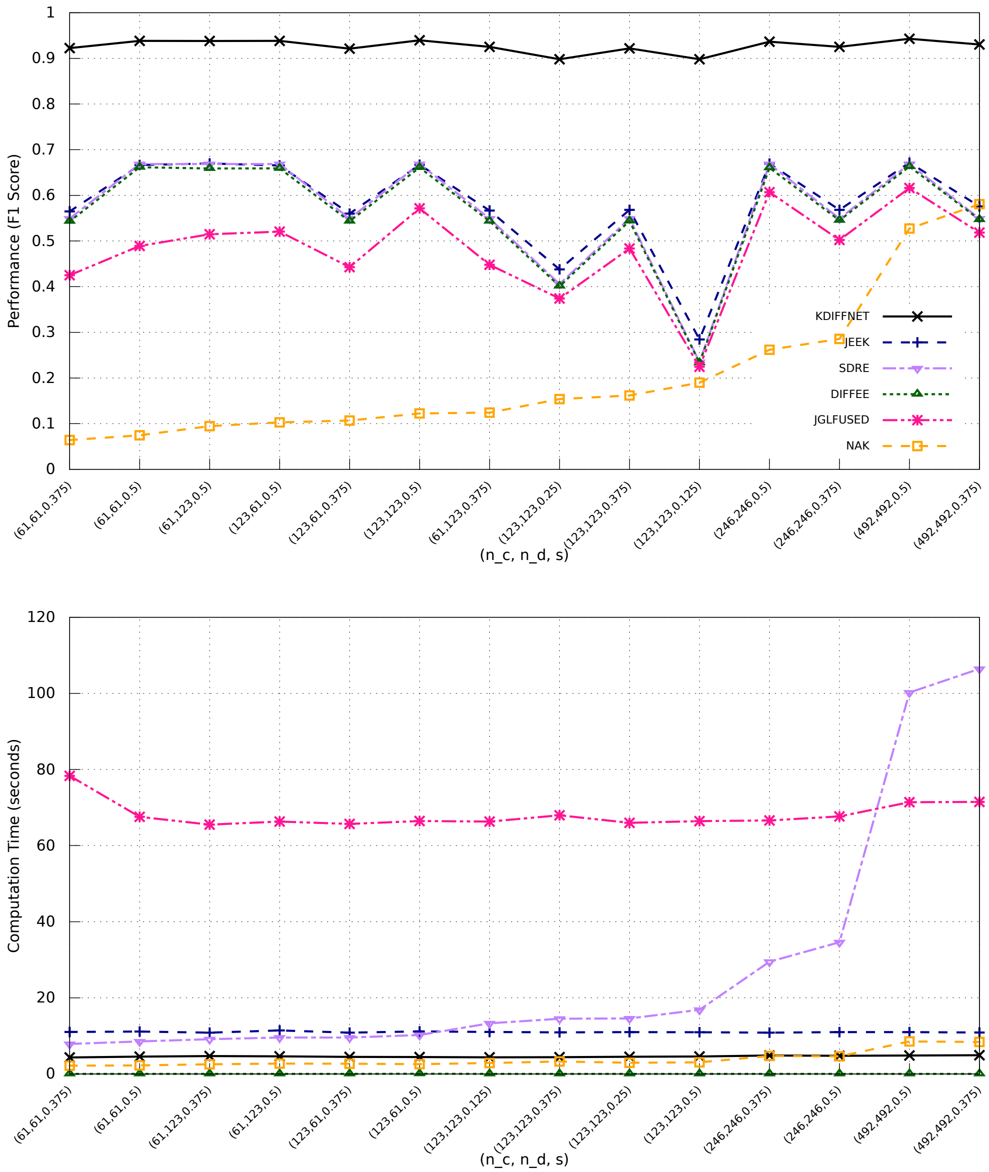

Figure S:4: KDiffNet Edge and Vertex Knowledge Simulation Results for $p = 246$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet and baseline methods

In Figure S:4(a), we plot the test F1-score for simulated datasets generated using a larger W_E with $p = 246$, representing spatial distances between different 246 regions of the brain. This represents a larger and different set of spatial brain regions. In this case, KDiffNet outperforms the best baseline in each case by an average improvement of 1400% relative to the best performing baseline. In this case as well, including available additional knowledge is clearly useful as JEEK does relatively better than the other baselines, which do not incorporate available additional knowledge. JGLFUSED again performs the worst on all cases. Figure S:4(b) shows the computation cost of each method measured in seconds for each case. In all cases, KDiffNet has the least computation cost in different settings of the data generation. KDiffNet is on average $20\times$ faster than the best performing baseline. For detailed results, see Section S:6.5.

We cannot compare Diff-CLIME as it takes more than 2 days to finish $p = 246$ case.

Edge Knowledge (KE): Given known W_E, we use KDiffNet-E to infer the differential structure in this case.

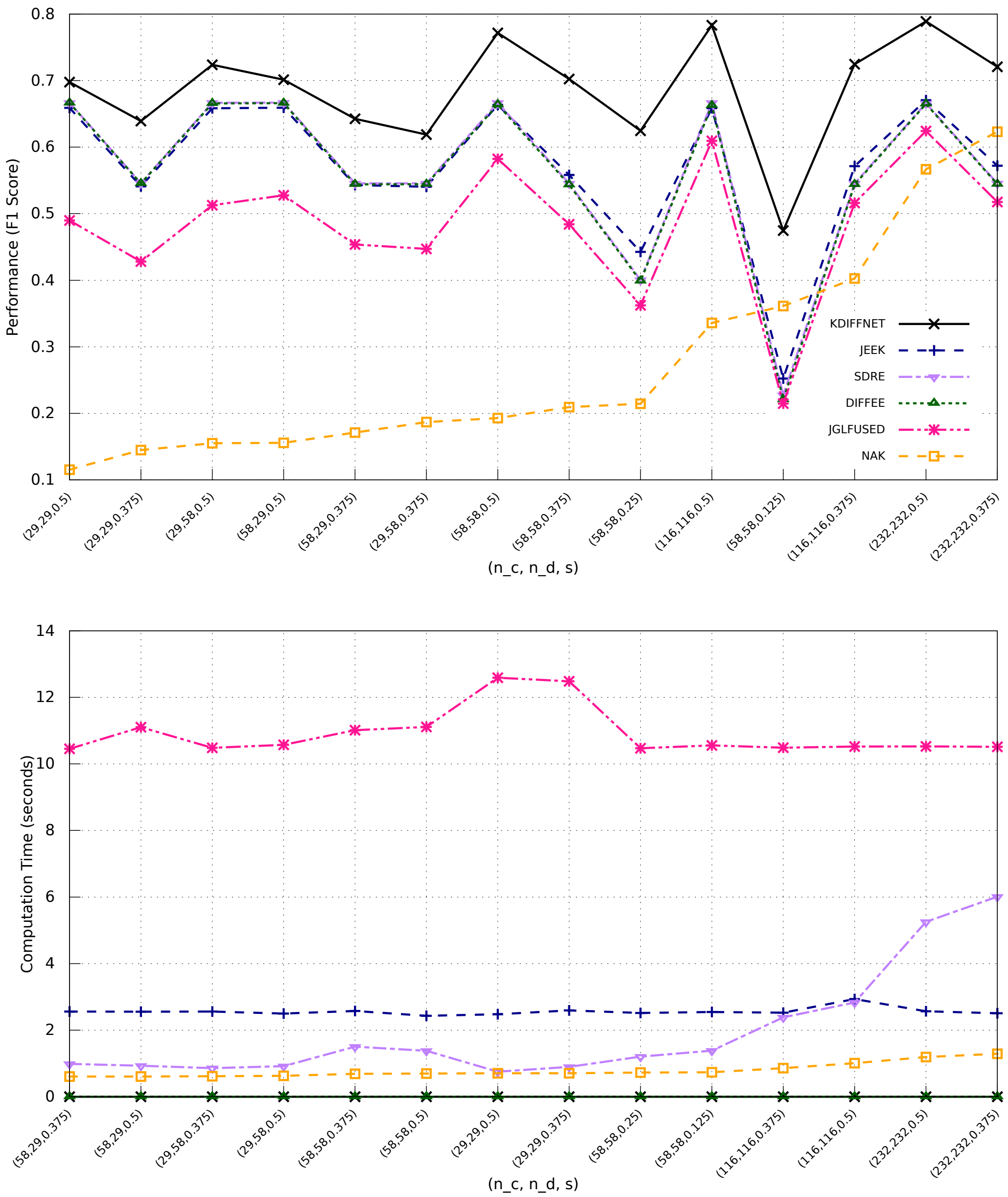

Figure S:5: KDiffNet-E Simulation Results for $p = 116$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet-E and baseline methods.

Figure S:5(a) shows the performance in terms of F1-Score of KDiffNet-E in comparison to the baselines for $p = 116$, corresponding to 116 spatial regions of the brain. In $p = 116$ case, KDiffNet-

E outperforms the best baseline in each case by an average improvement of 23%. While JEEK, DIFFEE and SDRE perform similar to each other, JGLFUSED performs the worst on all cases.

Figure S:5(b) shows the computation cost of each method measured in seconds for each case. In all cases, KDiffNet-E has the least computation cost in different cases of varying n_c and n_d, as well as with different sparsity of the differential network. For $p = 116$, KDiffNet-E , owing to an entry wise parallelizable closed form solution, is on average $2356\times$ faster than the best performing baseline. In Figure S:6(a), we plot the test F1-score for simulated datasets generated using W with

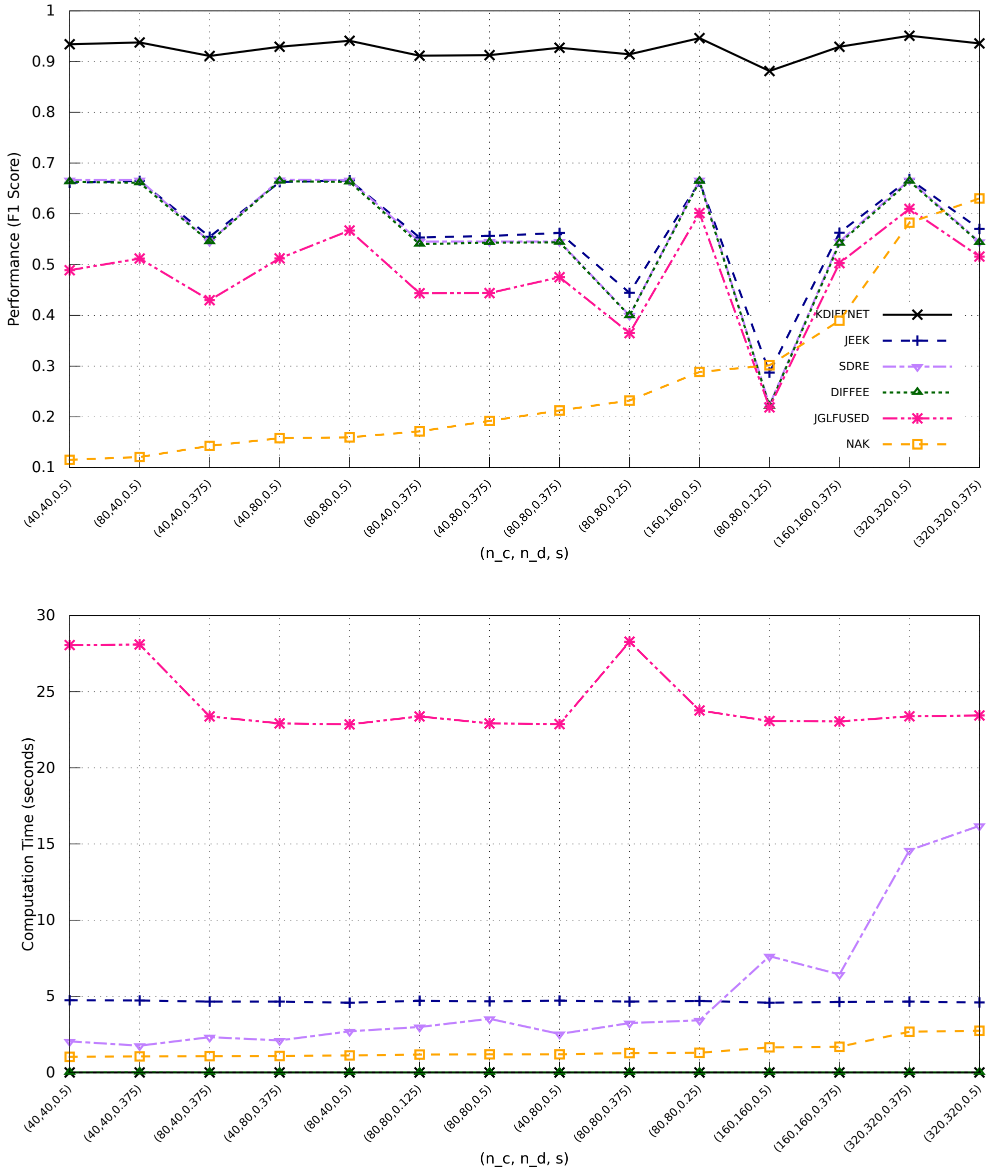

Figure S:6: KDiffNet-E Simulation Results for $p = 160$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet-E and baseline methods.

$p = 160$, representing spatial distances between different 160 regions of the brain. This represents a larger and different set of spatial brain regions. In $p = 160$ case, KDiffNet-E outperforms the best baseline in each case by an average improvement of 67.5%. Including available additional knowledge is clearly useful as JEEK does relatively better than the other baselines, which do not incorporate available additional knowledge. JGLFUSED performs the worst on all cases. Figure S:6(b) shows the computation cost of each method measured in seconds for each case. In all cases, KDiffNet-E has the least computation cost in different cases of varying n_c and n_d, as well as with different sparsity of the differential network. KDiffNet-E is on average $3300\times$ faster than the best performing baseline. In Figure S:7(a), we plot the test F1-score for simulated datasets generated using a larger W with

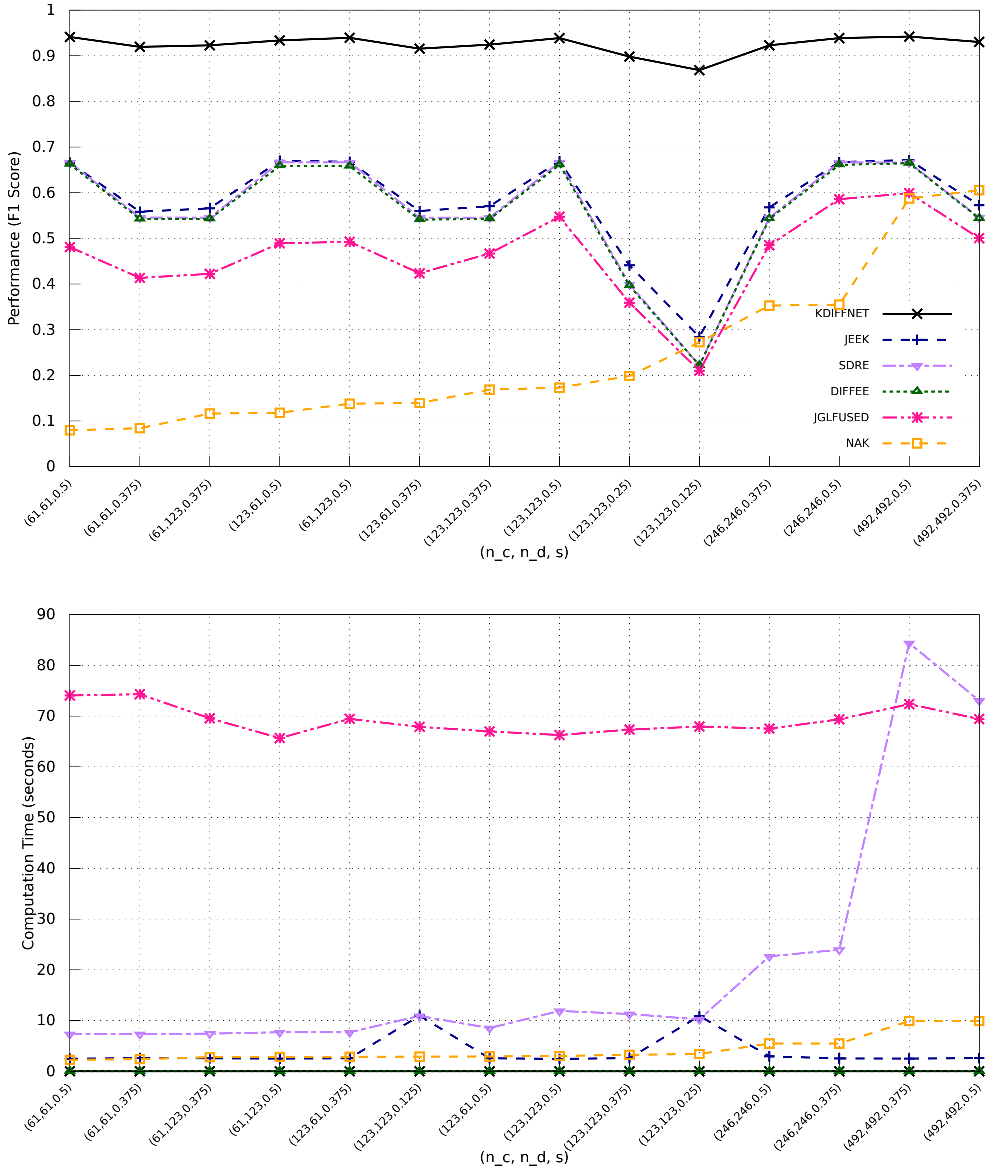

Figure S:7: KDiffNet-E Simulation Results for $p = 246$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet-E and baseline methods.

$p = 246$, representing spatial distances between different 246 regions of the brain. This represents a larger and different set of spatial brain regions. In this case, KDiffNet-E outperforms the best baseline in each case by an average improvement of 66.4% relative to the best performing baseline. Including available additional knowledge is clearly useful as JEEK does relatively better than the other baselines, which do not incorporate available additional knowledge. JGLFUSED performs the worst on all cases. Figure S:7(b) shows the computation cost of each method measured in seconds for each case. In all cases, KDiffNet-E has the least computation cost in different cases of varying

n_c and n_d, as well as with different sparsity of the differential network. KDiffNet-E is on average 3966× faster than the best performing baseline.

Node Group Knowledge : We use KDiffNet-G to estimate the differential network with the known groups as extra knowledge. We vary the number of groups s_G and the number of samples n_c and n_d for each case of $p = \{116, 160, 246\}$. Figure S:8 shows the F1-Score of KDiffNet-G and the baselines for $p = 116$. KDiffNet-G clearly has a large advantage when extra node group knowledge is available. The baselines cannot model such available knowledge.

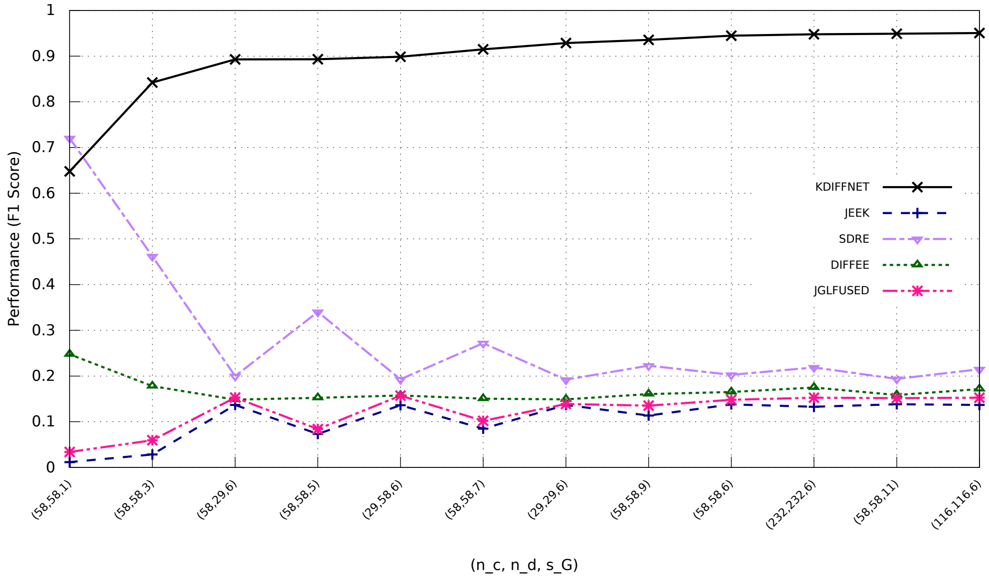

Figure S:8: KDiffNet-G Simulation Results for $p = 246$ for different settings of n_c, n_d and s: (a) The test F1-score and (b) The average computation time (measured in seconds) per λ_n for KDiffNet-E and baseline methods.

Varying proportion of known edges: We generate W_E matrices with $p = 150$ using Erdos Renyi Graph [9]. We use the generated graph as prior edge knowledge W_E. Additionally, we simulate 15 groups of size 10 as explained in Section S:6.2. We simulate Ω_c and Ω_d as explained in Section S:6.2. Figre S:9 presents the performance of KDiffNet-EG , KDiffNet-E and DIFFEE with varying proportion of known edges.

KDiffNet-EG has a higher F1-score than KDiffNet-E as it can additionally incorporate known group information. As expected, with increase in the proportion of known edges, F1-Score improves for both KDiffNet-EG and KDiffNet-E . In contrast DIFFEE cannot make use of additional information and the F1-Score remains the same.

Scalability in p: To evaluate the scalability of KDiffNet and baselines to large p, we also generate larger W_E matrices with $p = 2000$ using Erdos Renyi Graph [9], similar to the aforementioned design. Using the generated graph as prior edge knowledge W_E, we design Ω_c and Ω_d as explained in Section S:6.2. For the case of both edge and vertex knowledge, we fix the number of groups to 100 of size 10. We evaluate the scalability of KDiffNet-EG and baselines measured in terms of computation cost per λ_n.

Figure S:11 shows the computation time cost per λ_n for all methods. Clearly, KDiffNet takes the least time, for large p as well.

Choice of λ_n: For KDiffNet , we show the performance of all the methods as a function of choice of λ_n. Figure S:10 shows the True Positive Rate(TPR) and False Positive Rate(FPR) measured by varying λ_n for $p = 116$, $s = 0.5$ and $n_c = n_d = p/2$ under the Data-EG setting. Clearly, KDiffNet-EG achieves the highest Area under Curve (AUC) than all other baseline methods. KDiffNet-EG also

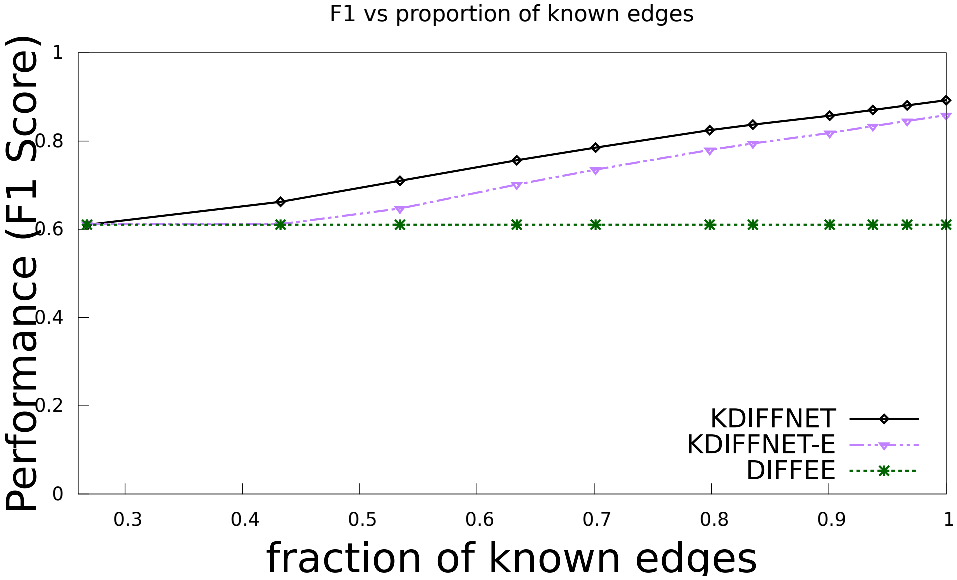

Figure S:9: F1-Score of KDiffNet-EG ,KDiffNet-E and DIFFEE with varying proportion of known edges.

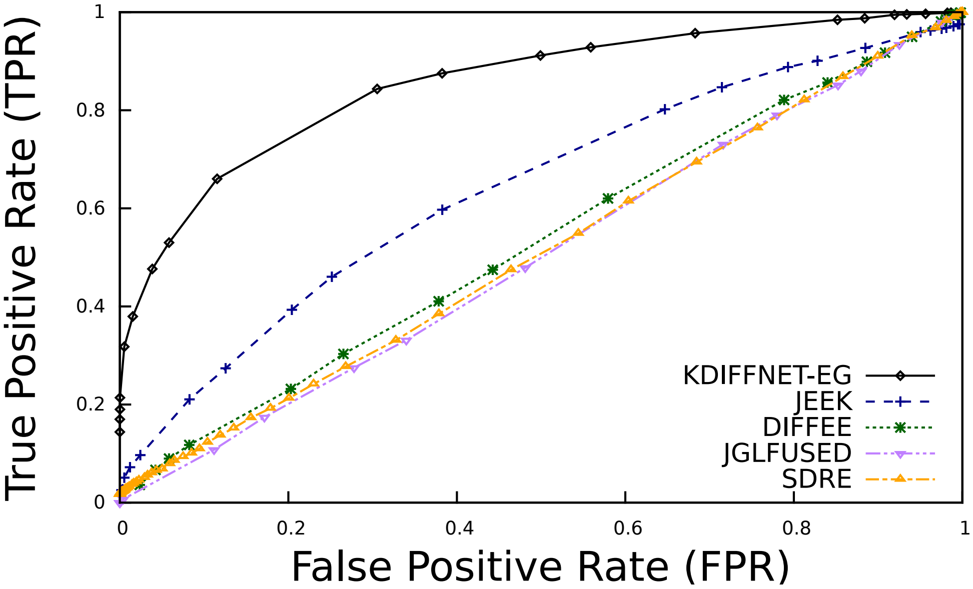

Figure S:10: Area Under Curve (AUC) Curves for KDiffNet and baselines at different hyperparameter values λ.

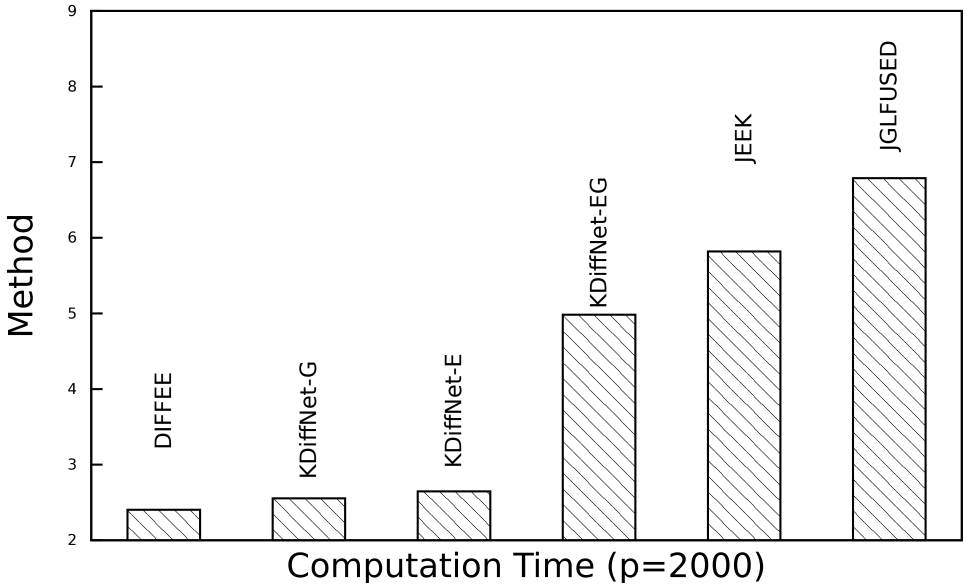

Figure S:11: Scalability of KDiffNet : Computation Cost (computation time per λ) as a function of p.

outperforms JEEK and NAK that take into account edge knowledge but cannot model the known group knowledge.

S:6.4 More Experiment: Brain Connectivity Estimation from Real-World fMRI

ABIDE Dataset: This data is from the Autism Brain Imaging Data Exchange (ABIDE) [7], a publicly available resting-state fMRI dataset. The ABIDE data aims to understand human brain connectivity and how it reflects neural disorders [19]. The data is retrieved from the Preprocessed Connectomes Project [4], where preprocessing is performed using the Configurable Pipeline for the Analysis of Connectomes (CPAC) [5] 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 [8] are examined. We also include two types of available node groups : one with 40 unique groups of regions belonging to the same functional network and another with 6 node groups about nodes belonging to the same broader anatomical region of the brain.

Cross-validation: Classification is performed using the 3-fold cross-validation suggested by the literature [14][20]. We tune over λ_n and pick the best λ_n using cross validation. The subjects are randomly partitioned into three equal sets: a training set, a validation set, and a test set. Each estimator produces $\widehat{\Omega}_c - \widehat{\Omega}_d$ using the training set. Then, these differential networks are used as inputs to Quadratic discriminant analysis (QDA), which is tuned via cross-validation on the validation set. Finally, accuracy is calculated by running QDA on the test set. This classification process aims to assess the ability of an estimator to learn the differential patterns of the connectome structures.

S:6.5 Detailed Simulation Results

Table S:2,Table S:3 and Table S:4 present a summary of results for KDiffNet-EG , KDiffNet-E and KDiffNet-G in terms of F1-Score, respectively. We report the average F1-Score(along with standard deviation across the same setting of n_c and n_d) across all simulation settings for each p. Table S:5,Table S:6 and Table S:7 present a summary of computation time for KDiffNet-EG , KDiffNet-E and KDiffNet-G , respectively. We report the average computation time per λ_n across all simulation settings for each p.

References

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

Method	Data-EG		
	W1	W2	W3
KDiffNet-EG	**0.717±0.05**	**0.927±0.01**	**0.934±0.07**
JEEK	0.572±0.10	0.581±0.094	0.58±0.09
SDRE	0.573±0.11	0.568±0.11	0.574±0.11
DIFFEE	0.57±0.11	0.56 ±0.11	0.57 ±0.11
JGLFUSED	0.502±0.08	0.481±0.08	0.495±0.08
NAK	0.226±0.08	0.204±0.07	0.207 ±0.08

Table S:2: Mean Performance (and standard deviation) of KDiffNet-EG and baselines for multiple data settings.

Method	Data-E		
	W1	W2	W3
KDiffNet-E	**0.687±(0.083)**	**0.924±(0.017)**	**0.926±(0.018)**
JEEK	0.577±(0.09)	0.581±(0.094)	0.577±(0.093)
SDRE	0.5643(±0.109)	0.564(±0.108)	0.5611(±0.109)
DIFFEE	0.562(±0.109)	0.5599(±0.109)	0.564(±0.109)
JGLFUSED	0.484(±0.085)	0.463(±0.080)	0.478(±0.081)
NAK	0.274(±0.079)	0.242(±0.071)	0.2641(±0.083)

Table S:3: Mean Performance (and standard deviation) KDiffNet-E and baselines for multiple data settings.

Method	Data-G		
	W1	W2	W3
KDiffNet-G	**0.896±(0.068)**	**0.891±(0.0615)**	**0.889±(0.034)**
JEEK	*	*	*
SDRE	0.286(±0.101)	0.318(±0.1015)	0.296(±0.064)
DIFFEE	0.1678(±0.020)	0.1355(±0.023)	0.1576(±0.147)
JGLFUSED	0.122(±0.026)	0.055±0.01	0.090(±0.039)
NAK	*	*	*

Table S:4: Mean Performance (and standard deviation) of KDiffNet-G and baselines for multiple data settings.

Method	Data-EG		
	W1	W2	W3
KDiffNet-EG	**0.562±0.077**	4.591±0.080	1.701±0.083
JEEK	2.460 ±0.020	10.998±0.1148	4.636±0.023
SDRE	2.384±0.144	27.487±2.5892	6.706 ±0.748
DIFFEE	**0.0005 ±0.0002**	0.004±0.0024	0.001 ±0.0002
JGLFUSED	22.571±3.0053	68.128±1.6624	24.336±1.450
NAK	0.780 ±0.0576	3.800±0.1518	1.432±0.043

Table S:5: Mean (and standard deviation) Computation Time (measured in seconds) per λ_n of KDiffNet-EG and baselines for multiple data settings.

Method	Data-E		
	W1	W2	W3
KDiffNet-E	**0.0009±0.0003**	0.0011±0.0005	0.0013±0.0003
JEEK	2.560±0.1099	3.761±1.1835	4.661 ±0.1099
SDRE	1.950±0.264	21.024±3.2085	5.105±0.2638
DIFFEE	**0.0005 ±0.0002**	0.004±0.0004	0.001 ± 0.0002
JGLFUSED	10.921±0.3389	69.160±2.0849	24.252±0.3389
NAK	0.790±0.061	4.235±0.1586	1.444±0.0609

Table S:6: Mean (and standard deviation) Computation Time (measured in seconds) per λ_n of KDiffNet-E and baselines for multiple data settings.

Method	Data-G		
	W1	W2	W3
KDiffNet-G	**0.0014±0.0002**	0.0056±0.0016	0.0022±0.0006
JEEK	*	*	*
SDRE	1.685±0.248	11.764±1.23	4.264±0.566
DIFFEE	**0.0009±0.0001**	0.0035±0.001	0.0015±0.0002
JGLFUSED	22.855±0.593	144.47±59.68	45.322 ±4.312
NAK	*	*	*

Table S:7: Mean (and standard deviation) Computation Time (measured in seconds) per λ_n of KDiffNet-G and baselines for multiple data settings.

n_c	n_d	sparsity	Relative F1 Score	Relative Speed up
80	80	0.125	3.07	2941.13
40	40	0.375	1.64	3376.86
80	40	0.375	1.65	3322.57
40	80	0.375	1.64	4646.60
80	80	0.25	2.06	4696.00
80	80	0.375	1.65	2907.00
160	160	0.375	1.65	2893.00
40	80	0.5	1.39	2102.92
40	40	0.5	1.40	1460.43
320	320	0.375	1.64	2584.67
80	40	0.5	1.41	2698.00
80	80	0.5	1.41	2935.75
160	160	0.5	1.42	6359.17
320	320	0.5	1.42	3285.71

Table S:8: KDiffNet-E : $p = 160$ Relative Performance and speed up with respect to the best performing baseline.

n_c	n_d	sparsity	Relative F1 Score	Relative Speed up
123	123	0.125	3.06	6094.67
123	123	0.26	2.04	3896.57
123	61	0.375	1.63	6393.00
61	61	0.375	1.65	1852.00
61	123	0.375	1.63	2557.60
246	246	0.375	1.62	3155.50
123	123	0.375	1.62	1610.25
492	492	0.375	1.62	2506.80
123	61	0.5	1.39	6390.50
123	123	0.5	1.40	2025.83
246	246	0.5	1.41	7335.00
61	123	0.5	1.41	2557.60
61	61	0.5	1.41	7333.40
492	492	0.5	1.40	1834.43

Table S:9: KDiffNet-E : $p = 246$ Relative Performance and speed up with respect to the best performing baseline.

[2] P. J. Bickel and E. Levina. Covariance regularization by thresholding. *The Annals of Statistics*, pages 2577–2604, 2008.

[3] Y. Bu and J. Lederer. Integrating additional knowledge into estimation of graphical models.

[4] C. Craddock. Preprocessed connectomes project: open sharing of preprocessed neuroimaging data and derivatives. In *61st Annual Meeting*. AACAP, 2014.

[5] C. Craddock, S. Sikka, B. Cheung, R. Khanuja, S. 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.

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

[7] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. 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.

[8] N. U. Dosenbach, B. Nardos, A. L. Cohen, D. A. Fair, J. D. Power, J. A. Church, S. M. Nelson, G. S. Wig, A. C. Vogel, C. N. Lessov-Schlaggar, et al. Prediction of individual brain maturity using fmri. *Science*, 329(5997):1358–1361, 2010.

[9] P. ERDdS and A. R&wi. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[10] F. Fazayeli and A. Banerjee. Generalized direct change estimation in ising model structure. In *International Conference on Machine Learning*, pages 2281–2290, 2016.

[11] T. Ideker and N. J. Krogan. Differential network biology. *Molecular systems biology*, 8(1):565, 2012.

[12] S. Liu, J. A. Quinn, M. U. Gutmann, T. Suzuki, and M. Sugiyama. Direct learning of sparse changes in markov networks by density ratio estimation. *Neural computation*, 26(6):1169–1197, 2014.

[13] S. Negahban, B. Yu, M. J. Wainwright, and P. 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.

[14] R. A. Poldrack, P. C. Fletcher, R. N. Henson, K. J. Worsley, M. Brett, and T. E. Nichols. Guidelines for reporting an fmri study. *Neuroimage*, 40(2):409–414, 2008.

[15] P. Ravikumar, M. J. Wainwright, G. Raskutti, B. Yu, et al. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

[16] A. J. Rothman, E. Levina, and J. Zhu. Generalized thresholding of large covariance matrices. *Journal of the American Statistical Association*, 104(485):177–186, 2009.

[17] T. Shimamura, S. Imoto, R. Yamaguchi, and S. Miyano. Weighted lasso in graphical gaussian modeling for large gene network estimation based on microarray data. 19:142–153.

[18] C. Singh, B. Wang, and Y. Qi. A constrained, weighted-l1 minimization approach for joint discovery of heterogeneous neural connectivity graphs. *arXiv preprint arXiv:1709.04090*, 2017.

[19] D. C. Van Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, K. Ugurbil, W.-M. H. Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[20] G. Varoquaux, A. Gramfort, J.-B. Poline, and B. Thirion. Brain covariance selection: better individual functional connectivity models using population prior. In *Advances in neural information processing systems*, pages 2334–2342, 2010.

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

[22] B. Wang, A. Sekhon, and Y. Qi. A fast and scalable joint estimator for integrating additional knowledge in learning multiple related sparse gaussian graphical models. *arXiv preprint arXiv:1806.00548*, 2018.

[23] B. Wang, A. Sekhon, and Y. Qi. Fast and scalable learning of sparse changes in high-dimensional gaussian graphical model structure. In *Proceedings of The 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018. [PS].

[24] E. Yang, A. C. Lozano, and P. K. Ravikumar. Elementary estimators for graphical models. In *Advances in Neural Information Processing Systems*, pages 2159–2167, 2014.

[25] S. D. Zhao, T. T. Cai, and H. Li. Direct estimation of differential networks. *Biometrika*, 101(2):253–268, 2014.