Journal of Machine Learning Research 23 (2022) 1-36 Submitted 1/21; Revised 5/22; Published 9/22

Tree-based Node Aggregation in Sparse Graphical Models

Ines Wilms I.WILMS@MAASTRICHTUNIVERSITY.NL
Department of Quantitative Economics
Maastricht University
Maastricht, The Netherlands

Jacob Bien JBIEN@USC.EDU
Department of Data Sciences and Operations
Marshall School of Business, University of Southern California
California, USA

Editor: David Wipf

Abstract

High-dimensional graphical models are often estimated using regularization that is aimed at reducing the number of edges in a network. In this work, we show how even simpler networks can be produced by aggregating the nodes of the graphical model. We develop a new convex regularized method, called the *tree-aggregated graphical lasso* or tag-lasso, that estimates graphical models that are both edge-sparse and node-aggregated. The aggregation is performed in a data-driven fashion by leveraging side information in the form of a tree that encodes node similarity and facilitates the interpretation of the resulting aggregated nodes. We provide an efficient implementation of the tag-lasso by using the locally adaptive alternating direction method of multipliers and illustrate our proposal's practical advantages in simulation and in applications in finance and biology.

Keywords: aggregation, graphical model, high-dimensionality, regularization, sparsity

1. Introduction

Graphical models are greatly useful for understanding the relationships among large numbers of variables. Yet, estimating graphical models with many more parameters than observations is challenging, which has led to an active area of research on high-dimensional inverse covariance estimation. Numerous methods attempt to curb the curse of dimensionality through regularized estimation procedures (e.g., Meinshausen and Bühlmann, 2006; Yuan and Lin, 2007; Banerjee et al., 2008; Friedman et al., 2008; Rothman et al., 2008; Peng et al., 2009; Yuan, 2010; Cai et al., 2011, 2016). Such methods aim for sparsity in the inverse covariance matrix, which corresponds to graphical models with only a small number of edges. A common method for estimating sparse graphical models is the graphical lasso (glasso) (Yuan and Lin, 2007; Banerjee et al., 2008; Rothman et al., 2008; Friedman et al., 2008), which adds an ℓ_1-penalty to the negative log-likelihood of a sample of multivariate normal random variables. While this and many other methods focus on the *edges* for dimension reduction, far fewer contributions (e.g., Tan et al., 2015; Eisenach et al., 2020; Pircalabelu and Claeskens, 2020) focus on the *nodes* as a guiding principle for dimension reduction.

Nonetheless, node dimension reduction is becoming increasingly relevant in many areas where data are being measured at finer levels of granularity. For instance, in biology, modern high-throughput sequencing technologies provide low-cost microbiome data at high resolution; in neuroscience, brain activity in hundreds of regions of interest can be measured; in finance, data at the individual company level at short time scales are routinely analyzed; and in marketing, joint purchasing data on every stock-keeping-unit (product) are recorded. The fine-grained nature of these data brings new challenges. The sheer number of fine-grained, often noisy, variables makes it difficult to detect dependencies. Moreover, there can be a mismatch between the resolution of the measurement and the resolution at which natural meaningful interpretations can be made. The purpose of an analysis may be to draw conclusions about entities at a coarser level of resolution than happened to be measured. Because of this mismatch, practitioners are sometimes forced to devise ad hoc post-processing steps involving, for example, coloring the nodes based on some classification of them into groups in an attempt to make the structure of an estimated graphical model more interpretable and the domain-specific takeaways more apparent (e.g., Millington and Niranjan, 2019).

Our solution to this problem is to incorporate the side information about the relationship between nodes directly into the estimation procedure. In our framework, this side information is encoded as a tree whose leaves correspond to the measured variables. Such tree structures are readily available in many domains (e.g., taxonomies in biology and hierarchical classifications of jobs, companies, and products in business) and is well-suited to expressing the multi-resolution structure that is present in many problems. We propose a new convex regularization procedure, called *tag-lasso*, which stands for *t*ree-*a*ggregated-*g*raphical-*lasso*. This procedure combines node (or variable) aggregation with edge-sparsity. The tree-based aggregation serves to both amplify the signal of similar, low-level variables and render a graphical model involving nodes at an appropriate level of scale to be relevant and interpretable. The edge-sparsity encourages the graphical model involving the aggregated nodes to have a sparse network structure.

Our procedure is based on a tree-based parameterization strategy that translates the node aggregation problem into a sparse modeling problem, following an approach previously introduced in the regression setting (Yan and Bien, 2021). In Figure 1 (to be discussed more thoroughly in Section 4), we see that tag-lasso is able to recover the aggregated, sparse graph structure. By doing so, it yields a more accurate estimate of the true graph, and its output is easier to interpret than the full, noisy graph obtained by the glasso.

The rest of the paper is organized as follows. Section 2 introduces the tree-based parameterization structure for nodewise aggregation in graphical models. Section 3 introduces the tag-lasso estimator, formulated as a solution to a convex optimization problem, for which we derive an efficient algorithm. Section 4 presents the results of a simulation study. Section 5 illustrates the practical advantages of the tag-lasso on financial and microbiome data sets. Section 6 concludes.

2. Node Aggregation in Penalized Graphical Models

Let $\mathbf{S}$ be the empirical covariance matrix based on n multivariate normal observations of dimension p, with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The target of estimation

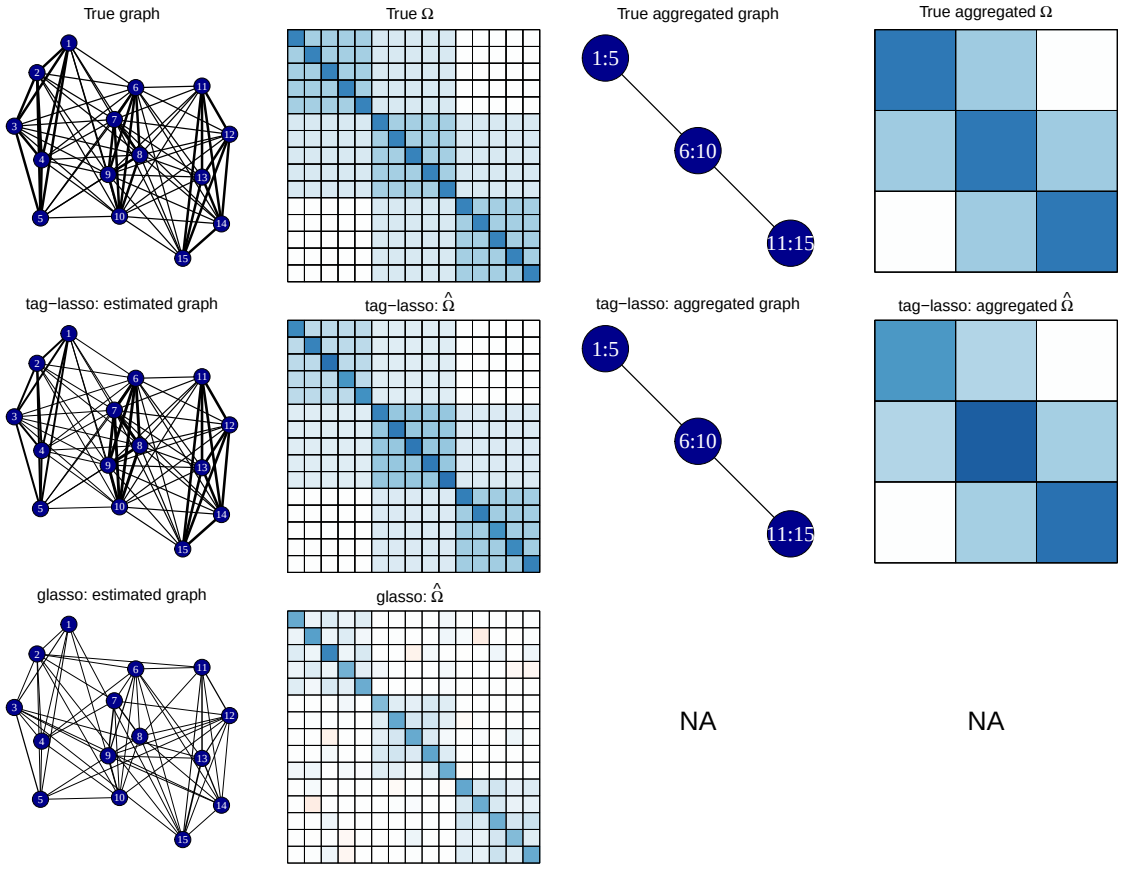


Figure 1: Top: True graph and precision matrix $\mathbf{\Omega}$ with corresponding aggregated graph and precision matrix. Middle: Estimation output of the tag-lasso. Bottom: Estimation output of the glasso.

is the precision matrix $\mathbf{\Omega} = \mathbf{\Sigma}^{-1}$, whose sparsity pattern provides the graph structure of the Gaussian graphical model, since $\Omega_{jk} = 0$ is equivalent to variables j and k being conditionally independent given all other variables. To estimate the precision matrix, it is common to use a convex penalization method of the form

$$\widehat{\mathbf{\Omega}} = \underset{\mathbf{\Omega}}{\operatorname{argmin}}\{-\operatorname{logdet}(\mathbf{\Omega}) + \operatorname{tr}(\mathbf{S}\mathbf{\Omega}) + \lambda\mathcal{P}(\mathbf{\Omega}) \;\; \text{s.t. } \mathbf{\Omega} = \mathbf{\Omega}^\top, \mathbf{\Omega} \succ 0\}, \tag{1}$$

where $\operatorname{tr}(\cdot)$ denotes the trace, $\cdot \succ 0$ denotes a positive definite matrix, $\mathcal{P}(\cdot)$ is a convex penalty function, and $\lambda > 0$ is a tuning parameter controlling the degree of penalization. Choosing the ℓ_1-norm

$$\mathcal{P}(\mathbf{\Omega}) = \|\mathbf{\Omega}^{-\text{diag}}\|_1, \tag{2}$$

where $\mathbf{\Omega}^{-\text{diag}}$ contains the unique off-diagonal elements, yields the *graphical lasso* (glasso) (Friedman et al., 2008; Yuan and Lin, 2007; Banerjee et al., 2008; Rothman et al., 2008). It encourages $\widehat{\mathbf{\Omega}}$ to be sparse, corresponding to a graphical model with few edges.

However, when $\mathbf{\Omega}$ is not sparse, demanding sparsity in $\widehat{\mathbf{\Omega}}$ may not be helpful, as we will show in Section 2.1. Such settings can arise when data are measured and analyzed at ever higher resolutions (a growing trend in many areas, see e.g. Callahan et al. 2017). A tree is a natural way to represent the different scales of data resolution, and we introduce a new choice for $\mathcal{P}$ that uses this tree to guide node aggregation, thereby allowing for a data adaptive choice of data scale for capturing dependencies. Such tree-based structures are available in many domains. For instance, companies can be aggregated according to hierarchical industry classification codes; products can be aggregated from brands towards product categories; brain voxels can be aggregated according to brain regions; microbiome data can be aggregated according to taxonomy. The resulting penalty function then encourages a more general and yet still highly interpretable structure for $\widehat{\mathbf{\Omega}}$. In the following subsection, we use a toy example to illustrate the power of such an approach.

2.1 Node Aggregation

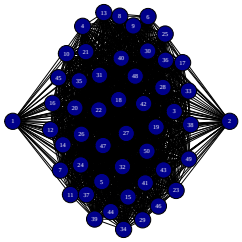
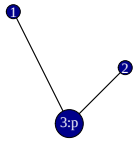
Consider a toy example with p variables

$$\begin{aligned} X_1 &= \sum_{j=3}^{p} X_j + \varepsilon_1 \\ X_2 &= \sum_{j=3}^{p} X_j + \varepsilon_2 \\ X_j &= \varepsilon_j, \text{ for } 3 \leq j \leq p, \end{aligned}$$

where $\varepsilon_1, \ldots, \varepsilon_p$ are independent standard normal random variables. By construction, it is clear that there is a very simple relationship between the variables: The first two variables both depend on the sum of the other $p-2$ variables. However, a standard graphical model on the p variables does not naturally express this simplicity. The first row of Table 1 shows the covariance and precision matrices for the full set of variables $X_1, \ldots, X_p$. The graph in the last column then visually represents the same information as the precision matrix. While this graph does convey the message that variables 1 and 2 are conditionally independent, it is extremely dense with $O(p^2)$ edges. As such, the precise structure among the remaining variables is hard to infer from the graph, an issue that only becomes worse when the number of variables p increases. Imagine if instead we could form a graphical model with only three variables: $X_1, X_2, \widetilde{X}$, where the last variable $\widetilde{X} = \sum_{j=3}^{p} X_j$ aggregates all but the first two variables. The bottom row of Table 1 results in a graphical model that matches the simplicity of the situation. The graph with aggregated nodes maintains its simplicity even when p increases.

The lack of sparsity in the p-node graphical model means that the graphical lasso will not do well; its estimation accuracy will suffer unless the sample size is extremely large. Nonetheless, a method that could perform node aggregation would be able to yield a highly-interpretable aggregated sparse graphical model since X_1 and X_2 are conditionally independent given the aggregated variable $\widetilde{X}$.

It is useful to map from the small aggregated graphical model to the original p-node graphical model. One does so by writing the precision matrix in "G-block" format (Bunea et al., 2020, although they introduce this terminology in the context of the covariance

Nodes	Covariance Matrix $\mathbf{\Sigma}$	Precision Matrix $\mathbf{\Omega}$	Graphical Model
$X_1, \ldots, X_p$	$\begin{pmatrix} p-1 & p-2 & \mathbf{1}_{p-2}^\top \\ p-2 & p-1 & \mathbf{1}_{p-2}^\top \\ \mathbf{1}_{p-2} & \mathbf{1}_{p-2} & \mathbf{I}_{p-2} \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & -\mathbf{1}_{p-2}^\top \\ 0 & 1 & -\mathbf{1}_{p-2}^\top \\ -\mathbf{1}_{p-2} & -\mathbf{1}_{p-2} & \mathbf{L} \end{pmatrix}$ with $\mathbf{L} = \mathbf{I}_{p-2} + 2 \cdot \mathbf{1}_{p-2} \cdot \mathbf{1}_{p-2}^\top$	
$X_1, X_2, \widetilde{X}$	$\begin{pmatrix} p-1 & p-2 & p-2 \\ p-2 & p-1 & p-2 \\ p-2 & p-2 & p-2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ -1 & -1 & 2+1/(p-2) \end{pmatrix}$	

Note: Let $\mathbf{1}_d$ denote a d-dimensional column vector of ones, and $\mathbf{I}_d$ be the $d \times d$ identity matrix.

Table 1: Toy example: Covariance and precision matrices with corresponding graphical model (drawn for $p = 50$) for the full (top) and aggregated (bottom) set of nodes.

matrix, not its inverse) for a given partition $G = \{G_1, ..., G_K\}$ of the nodes $\{1, \ldots, p\}$ and corresponding $p \times K$ membership matrix $\mathbf{M}$, with entries $M_{jk} = 1$ if $j \in G_k$, and $M_{jk} = 0$ otherwise. In particular, there exists a $K \times K$ symmetric matrix $\mathbf{C}$ and a $p \times p$ diagonal matrix $\mathbf{D}$ such that the precision matrix can be written as $\mathbf{\Omega} = \mathbf{MCM}^\top + \mathbf{D}$. The block-structure of $\mathbf{\Omega}$ is captured by the first part of the decomposition, the aggregated $K \times K$ precision matrix on the set of aggregated nodes can then be written as $\mathbf{\Omega}_{\text{agg}} = \mathbf{C} + \mathbf{D}_{\text{agg}}$, where $\mathbf{D}_{\text{agg}} = (\mathbf{M}^\top \mathbf{D}^{-1} \mathbf{M})^{-1}$ is diagonal. In the above example, $K = 3$, $G_1 = \{1\}$, $G_2 = \{2\}$, $G_3 = \{3, \ldots, p\}$ and $\mathbf{MCM}^\top$ has only three distinct rows/columns since the aggregated variables $j = 3, \ldots, p$ share all their entries. In the presence of node aggregation and edge sparsity, the graphical model corresponding to the aggregated precision matrix is far more parsimonious than the graphical model on the full precision matrix (see Table 1).

As motivated by this example, our main goal is to estimate the precision matrix in such a way that we can navigate from a p-dimensional problem to a K-dimensional problem whose corresponding graphical model provides a simple description of the conditional dependency structure among K aggregates of the original variables. In the following proposition, we show that this can be accomplished by looking for a precision matrix that has a G-block structure. The proof of the proposition is included in Appendix A.

Proposition 1 *Suppose* $\mathbf{X} \sim N_p(\mathbf{0}, \mathbf{\Omega}^{-1})$ *with* $\mathbf{\Omega} = \mathbf{MCM}^\top + \mathbf{D}$, *where* $\mathbf{M} \in \{0,1\}^{p \times K}$ *is the membership matrix,* $\mathbf{D} \succ \mathbf{0}$, *and let* $\widetilde{\mathbf{X}} = \mathbf{M}^\top \mathbf{X} \in \mathbb{R}^K$ *be the vector of aggregated variables. Then* $\widetilde{\mathbf{X}}$ *has precision matrix* $\mathbf{\Omega}_{agg} = \mathbf{C} + \mathbf{D}_{agg}$, *where* $\mathbf{D}_{agg}$ *is a diagonal matrix, and therefore* $c_{ij} = 0$ *is equivalent to the aggregates* $\widetilde{X}_i$ *and* $\widetilde{X}_j$ *being conditionally independent given all other aggregated variables.*

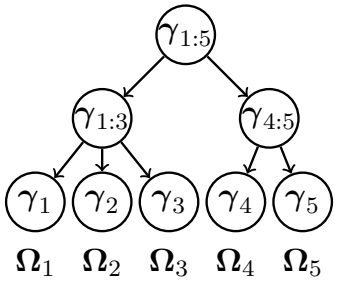


Figure 2: An example of a tree $\mathcal{T}$ encoding similarity among $p = 5$ variables.

The same matrix $\mathbf{C}$ thus enters the formula for $\mathbf{\Omega}$ as well as $\mathbf{\Omega}_{\text{agg}}$, thereby ensuring that both reflect the same conditional independence structure. While Proposition 1 thereby gives us the desired interpretation in the graphical model with K aggregated nodes, in practice, the partition G, its size K, and corresponding membership matrix $\mathbf{M}$ are, however, unknown. Rather than considering arbitrary partitions of the variables, we constrain ourselves specifically to partitions guided by a known tree. In so doing, we allow ourselves to exploit side information and help ensure that the aggregated nodes will be easily interpretable. To this end, we introduce a tree-based parameterization strategy that allows us to embed the node dimension reduction into a convex optimization framework.

2.2 Tree-Based Parameterization

Our aggregation procedure assumes that we have, as side information, a tree that represents the closeness (or similarity) of variables. We introduce here a matrix-valued extension of the tree-based parameterization developed in Yan and Bien (2021) for the regression setting. We consider a tree $\mathcal{T}$ with p leaves $\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_p$ where $\mathbf{\Omega}_j$ denotes column $1 \leq j \leq p$ of $\mathbf{\Omega}$. We restrict ourselves to partitions that can be expressed as a collection of branches of $\mathcal{T}$. Newly aggregated nodes are then formed by summing variables within branches. To this end, we assign a p-dimensional parameter vector $\boldsymbol{\gamma}_u$ to each node u in the tree $\mathcal{T}$ (see Figure 2 for an example). Writing the set of nodes in the path from the root to the j^{th} leaf (variable) as $\text{ancestor}(j) \cup \{j\}$, we express each column/row in the precision matrix as

$$\mathbf{\Omega}_j = \sum_{u \in \text{ancestor}(j) \cup \{j\}} \boldsymbol{\gamma}_u + d_j \mathbf{e}_j, \tag{3}$$

where we sum over all the $\boldsymbol{\gamma}_u$'s along this path, and $\mathbf{e}_j$ denotes the p-dimensional vector with all zeros except for its j^{th} element that is equal to one. In the remainder, we will make extensive use of the more compact notation $\mathbf{\Omega} = \mathbf{A}\mathbf{\Gamma} + \mathbf{D}$, where $\mathbf{A} \in \{0,1\}^{p \times |\mathcal{T}|}$ is a binary matrix with $A_{jk} = 1\{u_k \in \text{ancestor}(j) \cup \{j\}\} = 1\{j \in \text{descendant}(u_k) \cup \{u_k\}\}$ with $1\{\cdot\}$ denoting the indicator function, $\mathbf{\Gamma}$ is a $|\mathcal{T}| \times p$ parameter matrix collecting the $\boldsymbol{\gamma}_u$'s in its rows with $|\mathcal{T}|$ denoting the cardinality of the tree and $\mathbf{D}$ is a diagonal parameter matrix with elements $d_1, \ldots, d_p$.

By zeroing out $\boldsymbol{\gamma}_u$'s, certain nodes will be aggregated, as can be seen from the illustrative example in Figure 3. More precisely, let $\mathcal{V} = \{u : \boldsymbol{\gamma}_u \neq \mathbf{0}\}$ denote the set of non-zero rows in

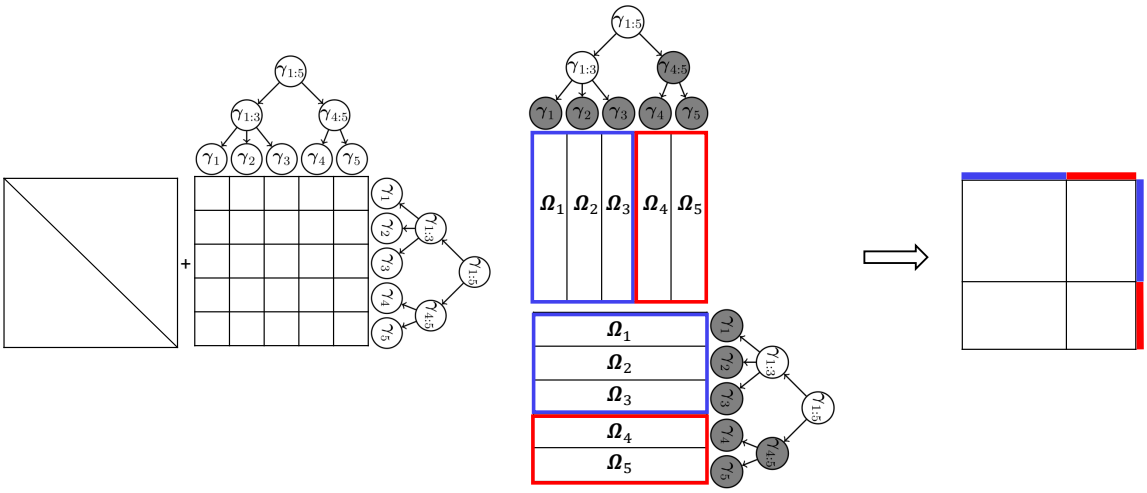


Figure 3: Left: An example of a 5×5-dimensional $\boldsymbol{\Omega}$ and a tree $\mathcal{T}$ that relates the corresponding $p = 5$ variables. We have $\boldsymbol{\Omega}_i = \boldsymbol{\gamma}_i + \boldsymbol{\gamma}_{1:3} + \boldsymbol{\gamma}_{1:5}$ for $i = 1, 2, 3$ and $\boldsymbol{\Omega}_j = \boldsymbol{\gamma}_j + \boldsymbol{\gamma}_{4:5} + \boldsymbol{\gamma}_{1:5}$ for $j = 4, 5$, by equation (3), ignoring the diagonal elements. Middle: By zeroing out the $\boldsymbol{\gamma}_i$'s in the gray nodes, we aggregate the rows/columns of $\boldsymbol{\Omega}$ into two groups indicated by the two colors: $\boldsymbol{\Omega}_1 = \boldsymbol{\Omega}_2 = \boldsymbol{\Omega}_3 = \boldsymbol{\gamma}_{1:3} + \boldsymbol{\gamma}_{1:5}$ (blue) and $\boldsymbol{\Omega}_4 = \boldsymbol{\Omega}_5 = \boldsymbol{\gamma}_{1:5}$ (red). Right: The precision matrix $\boldsymbol{\Omega}$ thus has a block-structure.

$\boldsymbol{\Gamma}$ and let $\mathbf{A}_{\mathcal{V}}$ be the sub-matrix of $\mathbf{A}$ where only the columns corresponding to the non-zero rows in $\boldsymbol{\Gamma}$ are kept. The number of blocks K in the aggregated network is then given by the number of unique rows in $\mathbf{A}_{\mathcal{V}}$. The membership matrix $\mathbf{M}$ (Section 2.1), and hence the set of aggregated nodes, can then be derived from the variables (rows) in the matrix $\mathbf{A}_{\mathcal{V}}$ that share all their row-entries. In the next section, we introduce the tag-lasso, which is based on this parameterization.

We prefer to perform aggregation according to tree-based structures, as opposed to more general directed acyclic graphs, because of the interpretability trees naturally offer. For trees, each node has only one parent and, hence, there exists a single path from the leaves (original variables) towards the root node (complete aggregation). In such cases, the intermediate levels in the tree provide a direct, natural labelling of the corresponding aggregated variables. We discuss the option to perform aggregation guided by more general graph-based structures in Section 6.

3. Tree Aggregated Graphical lasso

To achieve dimension reduction via node aggregation and edge sparsity simultaneously, we extend optimization problem (1) by incorporating the parameterization introduced above.

Our estimator, called the *tag-lasso*, is defined as

$$(\widehat{\boldsymbol{\Omega}}, \widehat{\boldsymbol{\Gamma}}, \widehat{\boldsymbol{D}}) = \underset{\boldsymbol{\Omega},\boldsymbol{\Gamma},\boldsymbol{D}}{\operatorname{argmin}}\{-\operatorname{logdet}(\boldsymbol{\Omega}) + \operatorname{tr}(\mathbf{S}\boldsymbol{\Omega}) + \lambda_1\|\boldsymbol{\Gamma}_{-r}\|_{2,1} + \lambda_2\|\boldsymbol{\Omega}^{-\text{diag}}\|_1$$
$$\text{s.t. } \boldsymbol{\Omega} = \boldsymbol{\Omega}^\top, \boldsymbol{\Omega} \succ \mathbf{0}, \boldsymbol{\gamma}_r = \gamma\mathbf{1}_p,\ \boldsymbol{\Omega} = \mathbf{A}\boldsymbol{\Gamma} + \mathbf{D},\ \mathbf{D} \text{ diag},\ D_{jj} \geq 0 \text{ for } j = 1, \ldots, p\}, \quad (4)$$

with $\|\boldsymbol{\Gamma}_{-r}\|_{2,1} = \sum_{u\in\mathcal{T}_{-r}} \|\boldsymbol{\gamma}_u\|_2$ and $\mathcal{T}_{-r}$ being the set of all nodes in $\mathcal{T}$ other than the root. This norm induces row-wise sparsity on all non-root rows of $\boldsymbol{\Gamma}$. This row-wise sparsity, in turn, induces node aggregation as explained in Section 2.2. The root is excluded from this penalty term so that in the extreme case of a large λ_1 one gets complete aggregation but not necessarily sparsity (in this extreme, all off-diagonal elements of $\widehat{\boldsymbol{\Omega}}$ are equal to the scalar γ that appears in the equality constraint involving $\boldsymbol{\gamma}_r$). While λ_1 controls the degree of node aggregation, λ_2 controls the degree of edge sparsity. When $\lambda_1 = 0$, the optimization problem in (4) reduces to the glasso.

The tag-lasso estimator imposes a block structure on the rows of $\widehat{\boldsymbol{\Omega}} - \widehat{\mathbf{D}}$ via the constraint $\boldsymbol{\Omega} = \mathbf{A}\boldsymbol{\Gamma} + \mathbf{D}$. From the sparsity pattern of $\widehat{\boldsymbol{\Gamma}}$, we obtain $\widehat{\mathcal{V}} = \{u \in \mathcal{T} : \widehat{\boldsymbol{\gamma}}_u \neq \mathbf{0}\}$, the set of non-zero rows in $\widehat{\boldsymbol{\Gamma}}$, such that we can write $\widehat{\boldsymbol{\Omega}} - \widehat{\mathbf{D}} = \mathbf{A}\widehat{\boldsymbol{\Gamma}} = \mathbf{A}_{\widehat{\mathcal{V}}}\widehat{\boldsymbol{\Gamma}}_{\widehat{\mathcal{V}}}$. Algorithm 1 in Appendix A details how the membership matrix $\widehat{\mathbf{M}}$ can be obtained from the original binary matrix $\mathbf{A}$ and the set $\widehat{\mathcal{V}}$. Due to the enforced symmetry on the precision matrix, the block structure imposed on the rows of $\widehat{\boldsymbol{\Omega}} - \widehat{\mathbf{D}}$ holds likewise for its columns. The next proposition then shows how the solution provided by the tag-lasso can be re-written in G-block format. The proof of the proposition is included in Appendix A.

Proposition 2 *Given a solution* $(\widehat{\boldsymbol{\Omega}}, \widehat{\boldsymbol{\Gamma}}, \widehat{\mathbf{D}})$ *to the tag-lasso problem, there exists a* $p \times K$ *partition matrix* $\widehat{\mathbf{M}}$ *and a symmetric* $K \times K$ *matrix* $\widehat{\mathbf{C}}$ *such that* $\widehat{\boldsymbol{\Omega}} = \widehat{\mathbf{M}}\widehat{\mathbf{C}}\widehat{\mathbf{M}}^\top + \widehat{\mathbf{D}}$.

The G-block structure imposed by the tag-lasso (in case $K < p$) implies that the tag-lasso is especially useful to apply when one believes that dimension reduction can be leveraged in terms of node-aggregation in addition to edge-sparsity.

Finally, note that optimization problem (4) fits into the general formulation of penalized graphical models given in (1) since it can be equivalently expressed as

$$\widehat{\boldsymbol{\Omega}} = \underset{\boldsymbol{\Omega}}{\operatorname{argmin}}\{-\operatorname{logdet}(\boldsymbol{\Omega}) + \operatorname{tr}(\mathbf{S}\boldsymbol{\Omega}) + \lambda_1\mathcal{P}_{\text{aggregate}}(\boldsymbol{\Omega}) + \lambda_2\mathcal{P}_{\text{sparse}}(\boldsymbol{\Omega}) \text{ s.t. } \boldsymbol{\Omega} = \boldsymbol{\Omega}^\top, \boldsymbol{\Omega} \succ \mathbf{0}\},$$

where

$$\mathcal{P}_{\text{aggregate}}(\boldsymbol{\Omega}) = \min_{\boldsymbol{\Gamma},\mathbf{D}} \{\|\boldsymbol{\Gamma}_{-r}\|_{2,1} \text{ s.t. } \boldsymbol{\gamma}_r = \gamma\mathbf{1}_p,\ \boldsymbol{\Omega} = \mathbf{A}\boldsymbol{\Gamma}+\mathbf{D},\ \mathbf{D} \text{ diag},\ D_{jj} \geq 0 \text{ for } j = 1, \ldots, p\}$$

and $\mathcal{P}_{\text{sparse}}(\boldsymbol{\Omega})$ is the ℓ_1-norm defined in (2).

The proposed tag-lasso estimator thus determines the node aggregation and edge sparsity while also producing an estimate of the precision matrix by solving one convex optimization problem. One might wonder how such a "one-stage" procedure compares to a two-stage procedure where first the level of node aggregation is determined, and secondly, the glasso is applied using the aggregated nodes determined in the first stage. In Appendix B we detail this two-stage procedure and compare its performance to the tag-lasso estimator through a simulation study. Across various simulation designs, we find that the proposed (one-stage) tag-lasso estimator provides important improvements in terms of estimation accuracy over such a two-stage benchmark as the latter struggles to retrieve the correct aggregation level.

3.1 Locally Adaptive Alternating Direction Method of Multipliers

We develop an *alternating direction method of multipliers* (ADMM) algorithm (Boyd et al., 2011), specifically tailored to solving (4). Our ADMM algorithm is based on solving this equivalent formulation of (4):

$$\min_{\substack{\boldsymbol{\Omega}^{(1)},\boldsymbol{\Omega}^{(2)},\boldsymbol{\Omega}^{(3)}\\ \boldsymbol{\Gamma}^{(1)},\boldsymbol{\Gamma}^{(2)},\boldsymbol{\Omega},\boldsymbol{\Gamma},\boldsymbol{D}}} \{-\text{logdet}(\boldsymbol{\Omega}^{(1)}) + \text{tr}(\mathbf{S}\boldsymbol{\Omega}^{(1)}) + \lambda_1\|\boldsymbol{\Gamma}^{(1)}_{-r}\|_{2,1} + \lambda_2\|\boldsymbol{\Omega}^{-\text{diag}(3)}\|_1$$

$$\text{s.t. } \boldsymbol{\Omega}^{(1)} = {\boldsymbol{\Omega}^{(1)}}^\top, \boldsymbol{\Omega}^{(1)} \succ \mathbf{0}, \boldsymbol{\gamma}^{(1)}_r = \gamma^{(1)}\mathbf{1}_p,\ \boldsymbol{\Omega}^{(2)} = \mathbf{A}\boldsymbol{\Gamma}^{(2)} + \mathbf{D},\ \mathbf{D} \text{ diag},\ D_{jj} \geq 0,\ j = 1, \ldots, p,$$
$$\boldsymbol{\Omega} = \boldsymbol{\Omega}^{(1)} = \boldsymbol{\Omega}^{(2)} = \boldsymbol{\Omega}^{(3)} \text{ and } \boldsymbol{\Gamma} = \boldsymbol{\Gamma}^{(1)} = \boldsymbol{\Gamma}^{(2)}\}. \quad (5)$$

Additional copies of $\boldsymbol{\Omega}$ and $\boldsymbol{\Gamma}$ are introduced to efficiently decouple the optimization problem.

Furthermore, we use an extension called *locally adaptive-ADMM* (LA-ADMM, Xu et al., 2017) with adaptive penalization to improve performance. The full details of the algorithm are provided in Appendix C. The computational complexity of the algorithm in terms of the number of variables p and the size of the tree $|\mathcal{T}|$ is $\mathcal{O}(p|\mathcal{T}|^2) \times$ (number of iterations $+ 1$) since each initialization step as well as each per iteration update is at most $\mathcal{O}(p|\mathcal{T}|^2)$. Bounds on the number of iterations for ADMM and LA-ADMM can be found in Xu et al. (2017).

3.2 Selection of the Tuning Parameters

To select the tuning parameters λ_1 and λ_2, we form a 10×10 grid of (λ_1, λ_2) values and find the pair that minimizes a 5-fold cross-validated likelihood-based score,

$$\frac{1}{5}\sum_{k=1}^{5}\left\{-\text{logdet}(\widehat{\boldsymbol{\Omega}}_{-\mathcal{F}_k}) + \text{tr}(\mathbf{S}_{\mathcal{F}_k}\widehat{\boldsymbol{\Omega}}_{-\mathcal{F}_k})\right\}, \quad (6)$$

where $\widehat{\boldsymbol{\Omega}}_{-\mathcal{F}_k}$ is an estimate of the precision matrix trained while withholding the samples in the k^{th} fold and $\mathbf{S}_{\mathcal{F}_k}$ is the sample covariance matrix computed on the k^{th} fold. In particular, we take $\widehat{\boldsymbol{\Omega}}_{-\mathcal{F}_k}$ to be a re-fitted version of our estimator (e.g., Belloni and Chernozhukov, 2013). After fitting the tag-lasso, recall that we obtain $\widehat{\mathcal{V}} = \{u \in \mathcal{T} : \widehat{\boldsymbol{\gamma}}_u \neq \mathbf{0}\}$, the set of non-zero rows in $\widehat{\boldsymbol{\Gamma}}$, which suggests a particular node aggregation; and $\widehat{\mathcal{E}} = \{(i,j) : \widehat{\boldsymbol{\Omega}}_{ij} \neq \mathbf{0}\}$, the set of non-zero elements in $\widehat{\boldsymbol{\Omega}}$, which suggests a particular edge sparsity structure. We then re-estimate $\boldsymbol{\Omega}$ by maximizing the likelihood subject to these aggregation and sparsity constraints:

$$\begin{aligned}
\min_{\boldsymbol{\Omega},\boldsymbol{\Gamma}_{\widehat{\mathcal{V}}},\boldsymbol{D}} \quad & -\text{logdet}(\boldsymbol{\Omega}) + \text{tr}(\mathbf{S}\boldsymbol{\Omega}) \\
\text{subject to} \quad & \boldsymbol{\Omega} = \boldsymbol{\Omega}^\top, \boldsymbol{\Omega} \succ \mathbf{0}, \\
& \boldsymbol{\gamma}_{\widehat{\mathcal{V}},r} = \gamma\mathbf{1}_p, \\
& \boldsymbol{\Omega} = \mathbf{A}_{\widehat{\mathcal{V}}}\boldsymbol{\Gamma}_{\widehat{\mathcal{V}}} + \mathbf{D}, \mathbf{D} \text{ diag.},\ D_{jj} \geq 0 \text{ for } j = 1, \ldots, p \\
& \boldsymbol{\Omega}_{ij} = 0, \text{ for } (i,j) \notin \widehat{\mathcal{E}}.
\end{aligned} \quad (7)$$

We solve this with an LA-ADMM algorithm similar to what is described in Section 3.1 and Appendix C.

3.3 Connections to Related Work

Combined forms of dimension reduction in graphical models can be found in, amongst others, Chandrasekaran et al. (2012); Tan et al. (2015); Eisenach et al. (2020); Brownlees et al. (2020); Pircalabelu and Claeskens (2020).

Chandrasekaran et al. (2012) consider a blend of principal component analysis with graphical modeling by combining sparsity with a low-rank structure. Tan et al. (2015) and Eisenach et al. (2020) both propose two-step procedures that first cluster variables in an initial dimension reduction step and subsequently estimate a cluster-based graphical model. Brownlees et al. (2020) introduce partial correlation network models with community structures but rely on the sample covariance matrix of the observations to perform spectral clustering. Our procedure differs from these works by introducing a single convex optimization problem that simultaneously induces aggregation and edge sparsity for the precision matrix.

Our work is most closely related to Pircalabelu and Claeskens (2020) who estimate a penalized graphical model and simultaneously classify nodes into communities. However, Pircalabelu and Claeskens (2020) do not use tree-based node-aggregation. Our approach, in contrast, considers the tree $\mathcal{T}$ as an important part of the problem to help determine the extent of node aggregation, and as a consequence the number of aggregated nodes (i.e. clusters, communities or blocks) K, in a data-driven way through guidance of the tree-based structure on the nodes.

4. Simulations

We investigate the advantages of jointly exploiting node aggregation and edge sparsity in graphical models. To this end, we compare the performance of the tag-lasso to two benchmarks:

(i) *oracle:* The aggregated, sparse graphical model in (7) is estimated subject to the true aggregation and sparsity constraints. The oracle is only available for simulated data and serves as a "best case" benchmark.

(ii) *glasso*: This does not perform any aggregation (corresponding to the tag-lasso with $\lambda_1 = 0$). A sparse graph on the full set of variables is estimated. The glasso is computed using the same LA-ADMM algorithm as detailed in Appendix C. The tuning parameter is selected from a 10-dimensional grid as the value that minimizes the 5-fold cross-validation likelihood-based score in equation (6) with $\widehat{\mathbf{\Omega}}_{-\mathcal{F}_k}$ taken to be the glasso estimate.

All simulations were performed using the `simulator` package (Bien, 2016) in `R` (R Core Team, 2017). We evaluate the estimators in terms of three performance metrics: estimation accuracy, aggregation performance, and sparsity recovery. We evaluate *estimation accuracy* by averaging over many simulation runs the Kullback-Leibler (KL) distance

$$\text{KL} = -\text{logdet}(\mathbf{\Sigma}\widehat{\mathbf{\Omega}}) + \text{tr}(\mathbf{\Sigma}\widehat{\mathbf{\Omega}}) - p,$$

where $\mathbf{\Sigma} = \mathbf{\Omega}^{-1}$ is the true covariance matrix. Note that the KL distance is zero if the estimated precision matrix equals the true precision matrix.

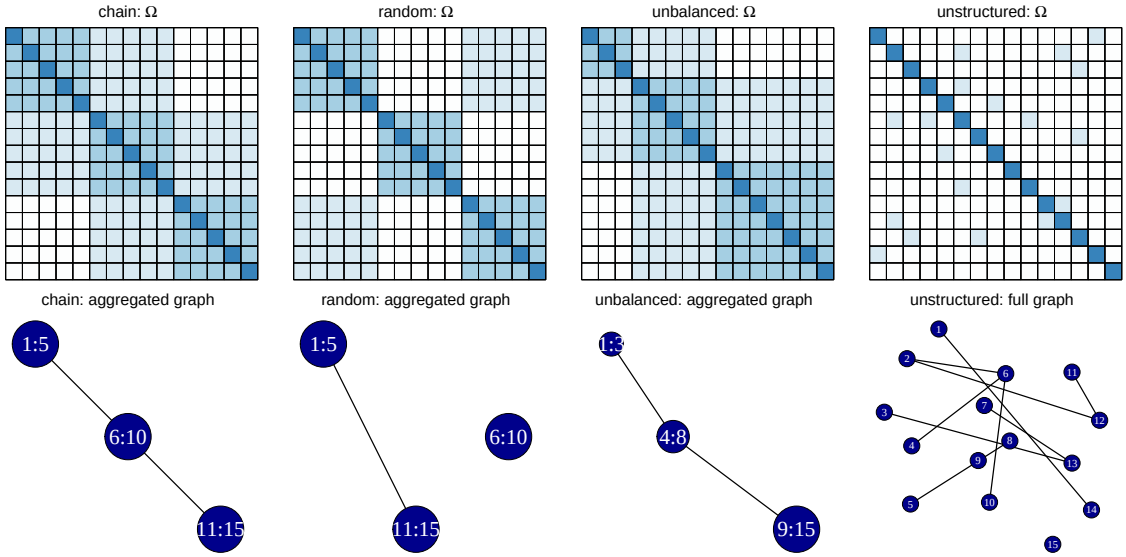


Figure 4: Four aggregation designs: chain, random, unbalanced and unstructured graphs with corresponding precision matrix (top) and graph on the set of aggregated nodes (bottom).

To evaluate *aggregation performance*, we use two measures: the Rand index (Rand, 1971) and the adjusted Rand index (Hubert and Arabie, 1985). Both indices measure the degree of similarity between the true partition on the set of nodes $1, \ldots, p$ and the estimated partition. The Rand index ranges from zero to one, where one means that both partitions are identical. The adjusted Rand index performs a re-scaling to account for the fact that random chance will cause some variables to occupy the same group.

Finally, to evaluate *sparsity recovery*, we use the false positive and false negative rates

$$\text{FPR} = \frac{\#\{(i,j) : \widehat{\Omega}_{ij} \neq 0 \text{ and } \Omega_{ij} = 0\}}{\#\{(i,j) : \Omega_{ij} = 0\}} \quad \text{and} \quad \text{FNR} \quad = \frac{\#\{(i,j) : \widehat{\Omega}_{ij} = 0 \text{ and } \Omega_{j} \neq 0\}}{\#\{(i,j) : \Omega_{ij} \neq 0\}}.$$

The FPR reports the fraction of truly zero components of the precision matrix that are estimated as nonzero. The FNR gives the fraction of truly nonzero components of the precision matrix that are estimated as zero.

4.1 Simulation Designs

Data are drawn from a multivariate normal distribution with mean zero and covariance matrix $\mathbf{\Sigma} = \mathbf{\Omega}^{-1}$. We take $p = 15$ variables and investigate the effect of increasing the number of variables in Section 4.3. We consider four different simulation designs, shown in Figure 4, each having a different combination of aggregation and sparsity structures for the precision matrix $\mathbf{\Omega}$.

Aggregation is present in the first three structures. The precision matrix has a G-block structure with $K = 3$ blocks. In Section 4.4, we investigate the effect of varying the number of blocks. In the *chain* graph, adjacent aggregated groups are connected through an edge.

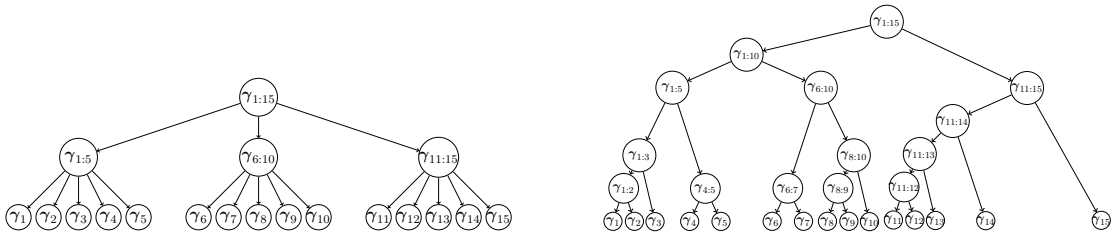


Figure 5: A simple tree used for the "tag-lasso ideal" (left) and a more realistic tree used for the "tag-lasso realistic" (right).

This structure corresponds to the motivating example of Section 1. In the *random* graph, one non-zero edge in the aggregated network is chosen at random. In the *unbalanced* graph, the clusters are of unequal size. In the *unstructured* graph, no aggregation is present.

Across all designs, we take the diagonal elements of $\mathbf{\Omega}$ to be 1, the elements within a block of aggregated variables to be 0.5, and the non-zero elements across blocks to be 0.25. We generate 100 different data sets for every simulation design and use a sample size of $n = 120$. The number of parameters ($p + p(p-1)/2 = 120$) equals the sample size.

The tag-lasso estimator relies on the existence of a tree to perform node dimension reduction. We consider two different tree structures throughout the simulation study. First, we use an "ideal" tree which contains the true aggregation structure as the sole aggregation level between the leaves and the root of the tree. As an example, the true aggregation structure for the chain graph structure is shown in the left panel of Figure 5. We form $\mathbf{A}$ corresponding to this oracle tree to obtain the *"tag-lasso ideal"* estimator.

We also consider a more realistic tree, shown in the right panel of Figure 5, following a construction similar to that of Yan and Bien (2021). The tree is formed by performing hierarchical clustering of p latent points chosen to ensure that the tree contains the true aggregation structure and that these true clusters occur across a variety of depths. In particular, we generate K cluster means $\mu_1, \ldots, \mu_K$ with $\mu_i = 1/i$. We set the number of latent points associated with each of the K means equal to the cluster sizes from Figure 4. These latent points are then drawn independently from $N(\mu_i, [0.05 \cdot \min_j(\mu_i - \mu_j)]^2)$. Finally, we form $\mathbf{A}$ corresponding to this tree to obtain the *"tag-lasso realistic"* estimator.

4.2 Results

We subsequently discuss the results on estimation accuracy, aggregation performance, and sparsity recovery.

Estimation Accuracy. Boxplots of the KL distances for the three estimators (tag-lasso ideal, tag-lasso realistic and glasso) relative to the oracle are given in Figure 6. The first three panels correspond to simulation designs with aggregation structures. In these settings, the tag-lasso estimators considerably outperform the glasso, on average by a factor five. The tag-lasso ideal method performs nearly as well as the oracle. Comparing the tag-lasso realistic method to the tag-lasso ideal method suggests a minimal price paid for using a more realistic tree.

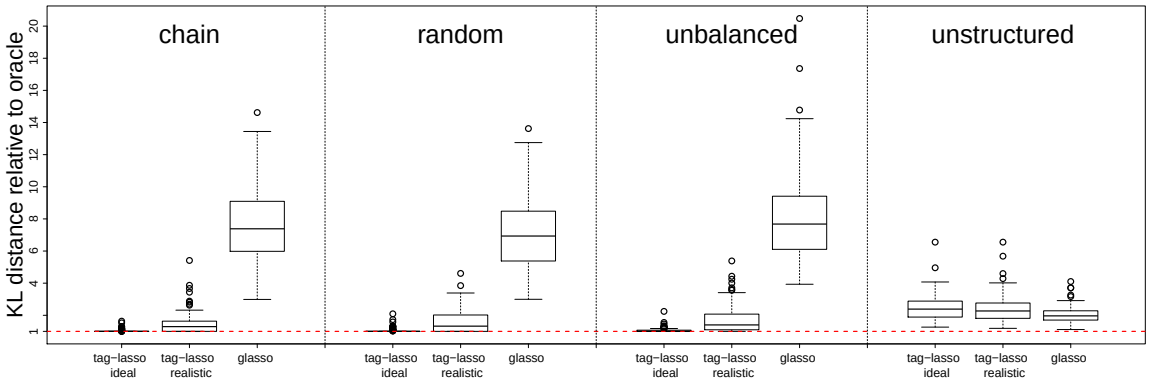


Figure 6: Estimation accuracy of the three estimators relative to the oracle.

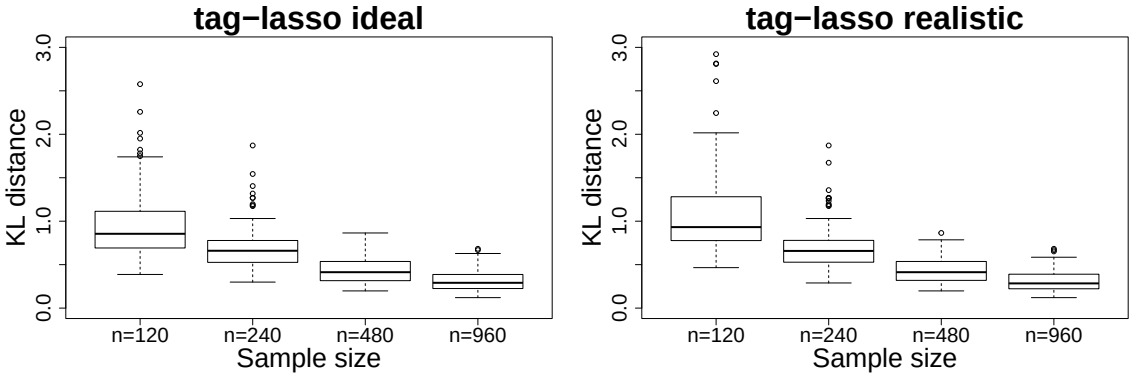


Figure 7: Estimation accuracy of the tag-lasso estimators for the chain design with fixed $p = 15$ and increasing sample size n.

The "unstructured" panel of Figure 6 shows a case in which there is sparsity but no aggregation in the true data generating model. As expected, the glasso performs best in this case: its average KL distance is around 0.41 versus 0.51 for the tag-lasso estimators. We thus observe a minimal cost to applying the tag-lasso, which encompasses the glasso as a special case when $\lambda_1 = 0$. The tag-lasso estimators can indeed reject the additional prior information provided by the tree, thereby selecting a small λ_1 value in the grid, hence a dense $\widehat{\mathbf{\Gamma}}$ and no node aggregation. In fact, the tag-lasso estimators include on average around 11 nodes in the "aggregated" graph, which is relatively close to the $p = 15$ original variables it should include, thereby explaining the small loss in estimation accuracy compared to the glasso.

While the paper does not contain theoretical results on the consistency of the tag-lasso estimator, we do investigate this numerically. We consider the chain design with $p = 15$ and increase the sample size from $n = 120$ to 240, 480, 960. Figure 7 contains the results for both tag-lasso estimators. As expected, we see that the estimation accuracy gradually increases

Estimators	chain		random		unbalanced		unstructured	
	RI	ARI	RI	ARI	RI	ARI	RI	ARI
tag-lasso ideal	1.00 (.00)	1.00 (.01)	1.00 (.00)	1.00 (.00)	1.00 (.00)	0.99 (.01)	0.84 (.02)	NA
tag-lasso realistic	0.95 (.01)	0.88 (.01)	0.97 (.01)	0.93 (.01)	0.94 (.01)	0.85 (.02)	0.81 (.02)	NA
glasso	0.71 (.00)	0.00 (.00)	0.71 (.00)	0.00 (.00)	0.67 (.00)	0.00 (.00)	1.00 (.00)	NA

Table 2: Aggregation performance of the three estimators, as measured by the Rand index (RI) and adjusted Rand index (ARI), for the four simulation designs. Standard errors are in parentheses.

Estimators	chain		random		unbalanced		unstructured	
	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR
tag-lasso ideal	0.22 (.04)	0.00 (.00)	0.19 (.04)	0.00 (.01)	0.46 (.05)	0.00 (.00)	0.06 (.01)	0.15 (.01)
tag-lasso realistic	0.30 (.04)	0.02 (.01)	0.13 (.02)	0.09 (.01)	0.44 (.04)	0.05 (.01)	0.05 (.01)	0.14 (.01)
glasso	0.80 (.02)	0.08 (.01)	0.73 (.01)	0.09 (.01)	0.82 (.02)	0.07 (.01)	0.16 (.01)	0.04 (.01)

Table 3: Sparsity recovery of the three estimators, as measured by the false positive rate (FPR) and false negative rate (FNR), for the four simulation designs. Standard errors are in parentheses.

(equivalently the KL distance decreases) for both tag-lasso estimators as the sample size increases relative to the fixed number of variables.

Aggregation Performance. Table 2 summarizes the aggregation performance of the three estimators in terms of the Rand index (RI) and adjusted Rand index (ARI). No results on the ARI in the unstructured simulation design are reported since it cannot be computed for a partition consisting of singletons. The tag-lasso estimators perform very well. If one can rely on an oracle tree, the tag-lasso perfectly recovers the aggregation structure, as reflected in the perfect (A)RI values of the tag-lasso ideal method. Even when the tag-lasso uses a more complex tree structure, it recovers the correct aggregation structure in the vast majority of cases. The glasso returns a partition of singletons as it is unable to perform dimension reduction through aggregation, as can be seen from its zero values on the ARI.

Sparsity Recovery. Table 3 summarizes the results on sparsity recovery (FPR and FNR). The tag-lasso estimators enjoy favorable FPR and FNR, mostly excluding the irrelevant conditional dependencies (as reflected by their low FPR) and including the relevant conditional dependencies (as reflected by their low FNR). In the simulation designs with aggregation, the glasso pays a large price for not being able to reduce dimensionality through aggregation, leading it to include too many irrelevant conditional dependencies, as reflected through its large FPRs. In the unstructured design, the rates of all estimators are, overall, low.

4.3 Increasing the Number of Nodes

We investigate the sensitivity of our results to an increasing number of variables p. We focus on the chain simulation design from Section 4.1 and subsequently double p from 15 to 30, 60 and 120 while keeping the number of blocks K fixed at three. The sample size n is set proportional to the complexity of the model, as measured by $Kp+p$. Hence, the sample sizes corresponding to the increasing values of p are respectively, $n = 120, 240, 480, 960$, thereby

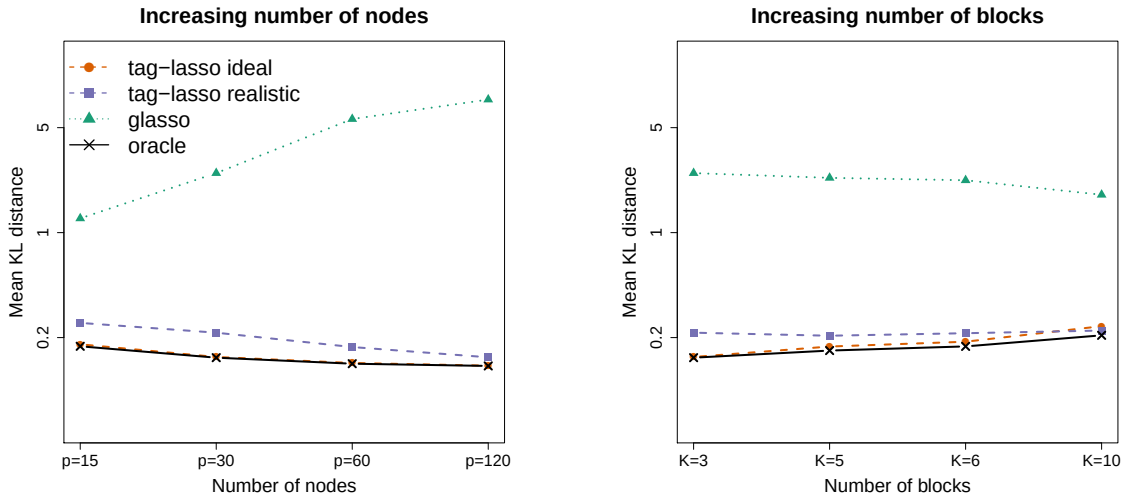
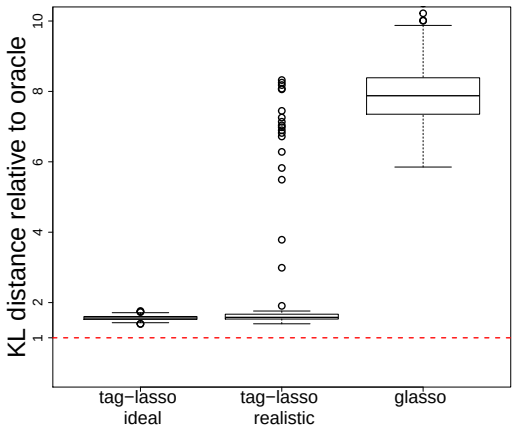


Figure 8: Estimation accuracy of the four estimators (on a log-scale) for increasing number of variables p (and fixed $K = 3$, left panel) the number of blocks K (and fixed $p = 30$, right panel).

keeping the ratio of the sample size to the complexity fixed at two. In each setting, the number of parameters to be estimated is large, equal to 120, 465, 1830, 7260, respectively; thus increasing relative to the sample size.

The left panel of Figure 8 shows the mean KL distance (on a log-scale) of the four estimators as a function of p. As the number of nodes increases, the estimation accuracy of the tag-lasso estimators and the oracle increases slightly. For fixed K and increasing p, the aggregated nodes—which can be thought of as the average of p/K random variables—may be stabler, thereby explaining why the problem at hand does not get harder when increasing p for the methods with node aggregation. By contrast, the glasso—which is unable to exploit the aggregation structure—performs worse as p increases. For $p = 120$, for instance, the tag-lasso estimators outperform the glasso by a factor 50.

Results on aggregation performance and sparsity recovery are presented in Figure 15 of Appendix D. The tag-lasso ideal method perfectly recovers the aggregation structure for all values of p. The realistic tag-lasso's aggregation performance is close to perfect and remains relatively stable as p increases. The glasso is unable to detect the aggregation structure, as expected and reflected through its zero ARIs. The tag-lasso estimators also maintain a better balance between the FPR and FNR than the glasso. While their FPRs increase as p increases, their FNRs remain close to perfect, hence all relevant conditional dependencies are recovered. The glasso, in contrast, fails to recover the majority of relevant conditional dependencies when $p = 60, 120$, thereby explaining its considerable drop in estimation accuracy.



Estimators	RI	ARI	FPR	FNR
tag-lasso ideal	1.00	1.00	0.02	0.55
tag-lasso realistic	0.89	0.84	0.58	0.28
glasso	0.67	0.00	0.05	0.93

Figure 9: High-dimensional design with $p = 150, n = 120$. Left: Estimation accuracy of the three estimators relative to the oracle. Right: Aggregation performance and sparsity recovery of the three estimators. Standard errors around the reported results are all smaller than 0.05, and are thus not reported.

4.4 Increasing the Number of Blocks

Next, we investigate the effect of increasing the number of blocks K. We take the chain simulation design from Section 4.1 and increase the number of blocks from $K = 3$ to $K = 5, 6, 10$, while keeping the number of variables fixed at $p = 30$. The right panel of Figure 8 shows the mean KL distance (on a log-scale) of the four estimators as a function of K. As one would expect, the difference between the aggregation methods and the glasso decreases as K increases. However, for all K considered, the glasso does far less well than the aggregation based methods.

Similar conclusions hold in terms of aggregation and sparsity recovery performance. Detailed results are presented in Figure 16 of Appendix D. The tag-lasso ideal method performs as well as the oracle in terms of capturing the aggregation structure; the tag-lasso realistic method performs close to perfect and its aggregation performance improves with increasing K. In terms of sparsity recovery, the tag-lasso estimators hardly miss relevant conditional dependencies and only include a small number of irrelevant conditional dependencies. The glasso's sparsity recovery performance is overall worse but does improve with increasing K.

4.5 High-dimensional $p > n$ Design

Finally, we investigate the performance of the tag-lasso in a high-dimensional design where the number of variables p exceeds the sample size n. To this end, we consider the chain design with $p = 150$ and $n = 120$. The left panel of Figure 9 presents boxplots of the KL distances of the tag-lasso estimators and glasso relative to the oracle. In this high-dimensional design, the tag-lasso estimators pay a larger price in terms of estimation accuracy compared to the oracle. The same holds for the tag-lasso realistic compared to the tag-lasso ideal. Still both tag-lasso estimators considerably outperform the glasso.

The right panel of Figure 9 summarizes the aggregation performance and sparsity recovery of the tag-lasso estimators and glasso. The aggregation performance of the tag-lasso estimators remains high. Moreover, they balance false positives and false negatives better than the glasso. While the tag-lasso ideal mainly displays a high FNR, the tag-lasso realistic also suffers from returning an overly dense graph, as can be seen from its higher FPR. Glasso, in contrast, cannot handle the many non-zero elements in the true precision matrix in combination with the small sample size. To tackle the dimensionality, it returns an overly sparse solution as can be seen from the high FNR.

5. Applications

We consider two applications: a financial (Section 5.1) and a microbiome application (Section 5.2).

5.1 Financial Application

We demonstrate our method on a financial data set containing daily realized variances of $p = 31$ stock market indices from across the world in 2019 ($n = 254$). Daily realized variances based on five minute returns are taken from the Oxford-Man Institute of Quantitative Finance (publicly available at `http://realized.oxford-man.ox.ac.uk/data/download`). Following standard practice, all realized variances are log-transformed. An overview of the stock market indices is provided in Appendix E. We encode similarity between the 31 stock market indices according to geographical region, and use the tree shown in Figure 10 to apply the tag-lasso estimator.

Since the different observations of the consecutive days are (time)-dependent, we first fit the popular and simple *heterogeneous autoregressive* (HAR) model of (Corsi, 2009) to each of the individual log-transformed realized variance series. Graphical displays of the residual series of these 31 HAR models suggest that almost all autocorrelation in the series is captured. We then apply the tag-lasso to the residual series to learn the conditional dependency structure among stock market indices.

Estimated Graphical Model. We fit the tag-lasso estimator, with 5-fold cross-validation to select tuning parameters, to the full data set, with the matrix $\mathbf{A}$ encoding the tree structure in Figure 10. The tag-lasso returns a solution with $K = 6$ aggregated blocks; the sparsity pattern of the full estimated precision matrix is shown in the top left panel of Figure 11. The coloring of the row labels and the numbering of columns convey the memberships of each variable to aggregated blocks (to avoid clutter, only the first column of each block is labeled).

Dimension reduction mainly occurs through node aggregation, as can be seen from the aggregated precision matrix in the bottom left panel of Figure 11. The resulting aggregated graphical model is rather dense with only about half of the off-diagonal entries being non-zero in the estimated aggregated precision matrix, thereby suggesting strong volatility connectedness. The solution returned by the tag-lasso estimator consists of one single-market block (block 5: Canada) and five multi-market blocks, which vary in size. The Australian, South-America, and all Asian stock markets form one aggregated block (block 6). Note that the tag-lasso has "aggregated" these merely because they have the same *non*-dependence structure (i.e. all of these markets are estimated to be conditionally inde-

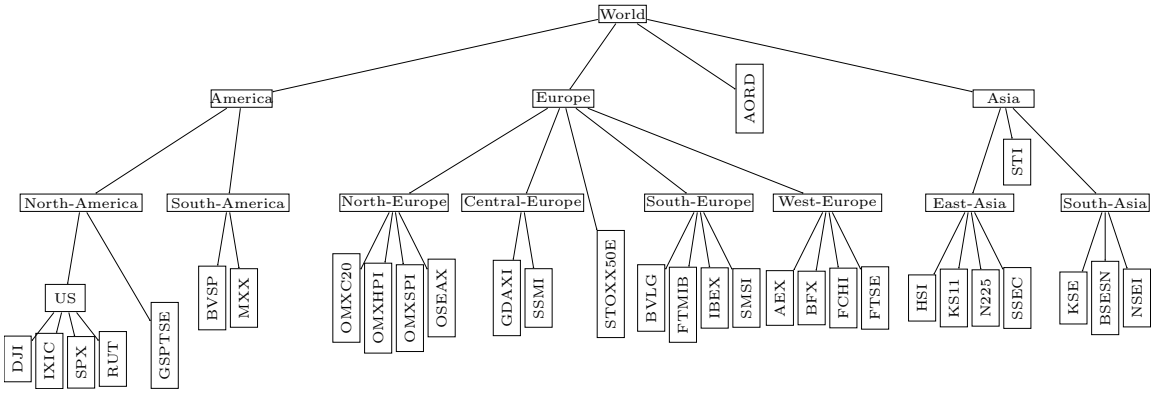


Figure 10: Geography-based tree for the stock market data, which aggregates the $p = 31$ stock market indices (leaves) over several sub-continents towards a single root. Leaves, which represent individual stock markets, are displayed horizontally.

pendent of each other and all other markets). The remaining aggregated nodes concern the US market (block 4) and three European markets, which are divided into North-Europe (block 1), Central-, South-Europe & STOXX50E (block 2), and West-Europe (block 3). In the aggregated network, the latter two and the US play a central role as they are the most strongly connected nodes: These three nodes are connected to each other, the US node is additionally connected to Canada, whereas these European nodes are additionally connected with North-Europe.

Out-of-sample Performance. We conduct an out-of-sample exercise to compare the tag-lasso estimator to the glasso estimator. We take a random sample of $n = 203$ observations (80% of the full data set) to form a "training sample" covariance matrix and use the remaining data to form a "test sample" covariance matrix $\mathbf{S}^{\text{test}}$, and repeat this procedure ten times. We fit both the tag-lasso and glasso estimator to the training covariance matrix, with 5-fold cross-validation on the training data to select tuning parameters. Next, we compute their corresponding out-of-sample errors on the test data, as in (6).

The top right panel of Figure 11 shows each of these ten test errors for both the tag-lasso (x-axis) and the glasso estimator (y-axis). The fact that in all ten replicates the points are well above the 45-degree line indicates that the tag-lasso estimator has better estimation error than the glasso. Tag-lasso has a lower test error than glasso in all ten replicates, resulting in a substantial reduction in glasso's test errors. This indicates that jointly exploiting edge and node dimension reduction is useful for precision matrix estimation in this context.

5.2 Microbiome Application

We next turn to a data set of gut microbial amplicon data in HIV patients (Rivera-Pinto et al., 2018), where our goal is to estimate an interpretable graphical model, capturing the interplay between different taxonomic groups of the microbiome. Bien et al. (2020) recently showed that tree-based aggregation in a supervised setting leads to parsimonious predictive

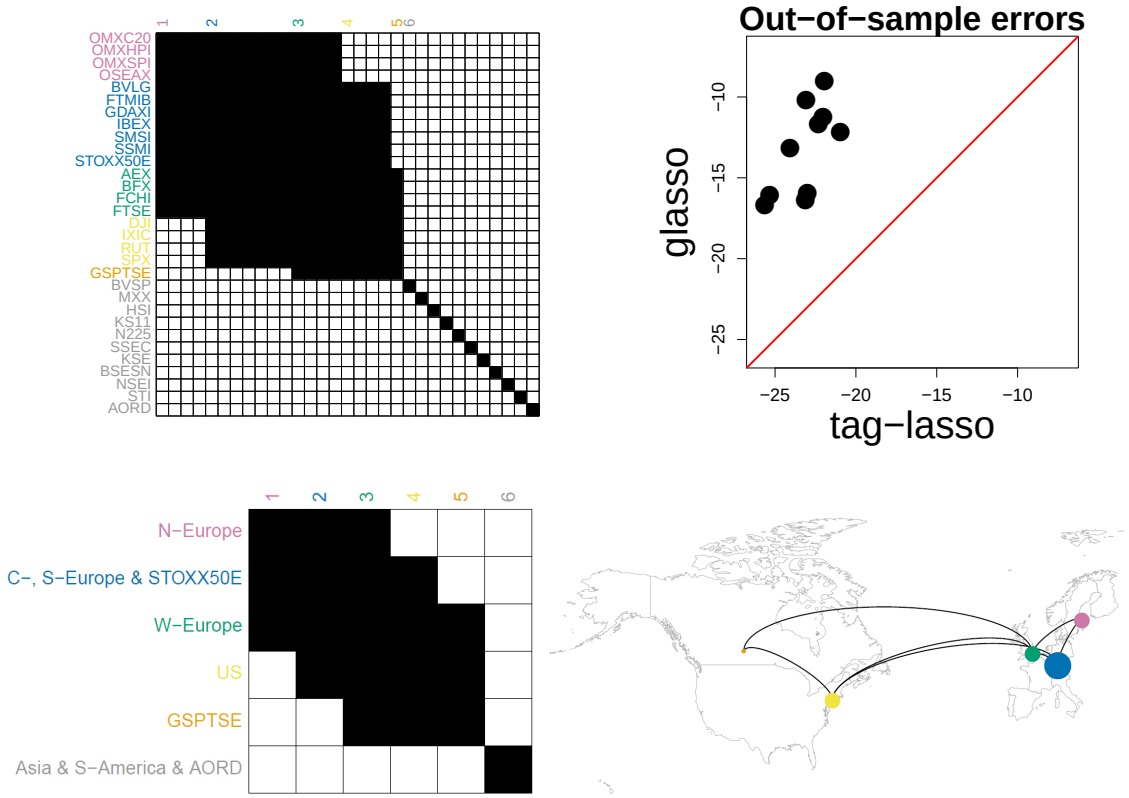


Figure 11: Stock market indices data. Top left: Sparsity pattern (non-zeros in black) of full $\hat{\mathbf{\Omega}}$ with aggregation structure conveyed through row label coloring and column numbering. Top right: Test errors across the ten replications (dots) for the tag-lasso versus glasso. Bottom: Aggregated graph for the $K = 6$ nodes obtained with the tag-lasso as an adjacency matrix (bottom left) and as a network (bottom right) with the size of each node proportional to the number of original variables it aggregates.

models. The data set has $n = 152$ HIV patients, and we apply the tag-lasso estimator to all $p = 104$ bacterial operational taxonomic units (OTUs) that have non-zero counts in over half of the samples. We use the taxonomic tree that arranges the OTUs into natural hierarchical groupings of taxa: with 17 genera, 11 families, five orders, five classes, three phyla, and one kingdom (the root node). We employ a standard data transformation from the field of compositional data analysis (see e.g., Aitchison, 1982) called the centered log-ratio (clr) transformation that is commonly used in microbiome graphical modeling (Kurtz et al., 2015; Lo and Marculescu, 2018; Kurtz et al., 2019). After transformation, Kurtz et al. (2015) apply the glasso, Lo and Marculescu (2018) incorporate phylogenetic information into glasso's optimization problem through weights within the ℓ_1-penalty, and Kurtz et al. (2019) estimate a latent graphical model which combines sparsity with a low-rank structure.

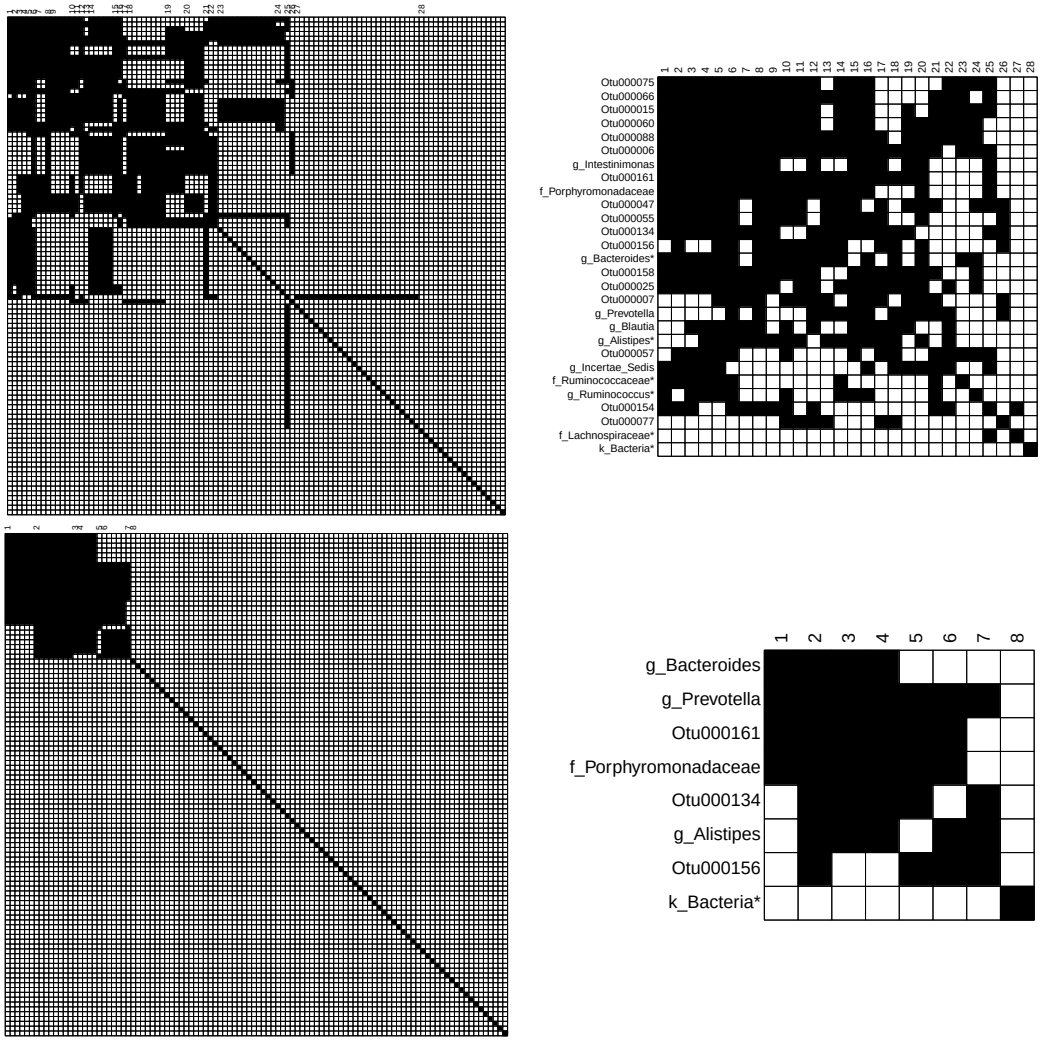


Figure 12: Microbiome data. Full precision matrix (left) and aggregated precision matrix (right) estimated by the tag-lasso with an unconstrained five-fold cross-validation (top) and with a cross-validation subject to the constraint that there are at most ten blocks (bottom).

We instead, use the tag-lasso to learn a sparse aggregated network from the clr-transformed microbiome compositions. While the clr-transform induces dependence between otherwise independent components, Proposition 1 in Cao et al. (2019) provides intuition that as long as the underlying graphical model is sparse and p is large, these induced dependencies may have minimal effect on the covariance matrix. Future work could more carefully account for the induced dependence, incorporating ideas from Cao et al. (2019) or Kurtz et al. (2019).

Estimated Graphical Model. We fit the tag-lasso to the full data set and use 5-fold cross-validation to select the tuning parameters. The tag-lasso estimator provides a sparse aggregated graphical model with $K = 28$ aggregated blocks (a substantial reduction in

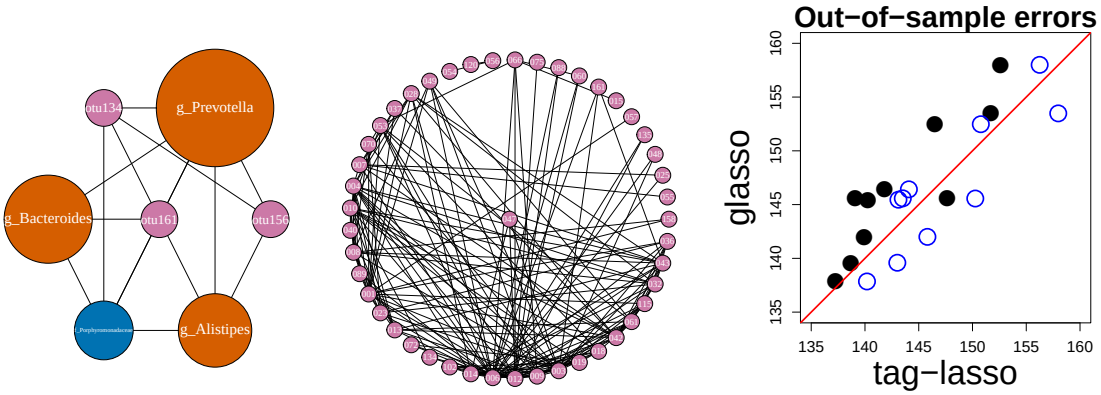


Figure 13: Microbiome data. Left: Aggregated network estimated by the constrained CV version of the tag-lasso. The colour of the nodes is based on their level of aggregation (OTU: pink, genus: orange, family: blue); their width is proportional to the number of OTUs they aggregate. Middle: Network estimated by the glasso. Right: Test errors across the ten replications for the unconstrained (solid black) and constrained (unfilled blue) CV version of the tag-lasso versus the glasso.

nodes from the original $p = 104$ OTUs). The top panel of Figure 12 shows the sparsity pattern of the $p \times p$ estimated precision matrix (top left) and of the $K \times K$ estimated aggregated precision matrix (top right). A notable feature of the tag-lasso solution is that it returns a wide range of aggregation levels: The aggregated network consists of 17 OTUs, 7 nodes aggregated to the genus level (these nodes start with "g_"), 3 to the family level (these nodes start with"f_"), and 1 node to the kingdom level (this node starts with "k_"). Some aggregated nodes, such as the "g_Blautia" node (block 19), contain all OTUs within their taxa; some other aggregated nodes, indicated with an asterisk like the "k_Bacteria*" node (block 28), have some of their OTUs missing. This latter "block" consists of 18 OTUs from across the phylogenetic tree that are estimated to be conditionally independent with all other OTUs in the data set.

While the tag-lasso determines the aggregation level in a data-driven way through cross-validation, practitioners or researchers may also sometimes wish to restrict the number of blocks K to a pre-determined level when such prior knowledge is available or if this is desirable for interpretability. As an illustration, we consider a *constrained cross-validation* scheme in which we restrict the number of blocks K to maximally ten and select the sparsity parameters with the best cross-validated error among those solutions with $K \leq 10$. The bottom panel of Figure 12 shows the sparsity pattern of the full and aggregated precision matrices estimated by this constrained version of the tag-lasso.

The resulting network consists of $K = 8$ aggregated nodes. The "k_Bacteria*" node now aggregates 78 OTUs that are estimated to be conditionally independent of each other and all others. The interactions among the remaining nodes are shown in the left panel of Figure 13, which consists of three OTUs (OTU134, OTU156, and OTU161, in pink), three genera (Prevotella, Bacteroides, and Alistipes in orange) and one family (Porphyromonadaceae in

blue). The resulting network is much simpler than the one estimated by the glasso, shown in the middle panel of Figure 13. The glasso finds 58 OTUs to be conditionally independent with all others, but the interactions among the remaining 46 OTUs are much more difficult to interpret. The glasso is limited to working at the OTU-level, which prevents it from providing insights about interactions that span different levels of the taxonomy.

Out-of-sample Performance. We conduct the same out-of-sample exercise as described in Section 5.1. The right panel of Figure 13 presents the ten test errors (black dots) for the unconstrained CV tag-lasso and glasso. In all but one case, the tag-lasso leads to a better fit than the glasso, suggesting that it is better suited for modeling the conditional dependencies among the OTUs. The unfilled blue dots show the same but for the constrained CV tag-lasso. In all ten cases, it underperforms the unconstrained CV tag-lasso (see shift to the right on the horizontal axis); however, its performance is on a par with the glasso, with test errors close to the 45 degree line. Thus, there does not appear to be a cost in out-of-sample-performance to the interpretability gains of the constrained tag-lasso over the glasso.

6. Conclusion

Detecting conditional dependencies between variables, as represented in a graphical model, forms a cornerstone of multivariate data analysis. However, graphical models, characterized by a set of nodes and edges, can quickly explode in dimensionality due to ever-increasing fine-grained levels of resolution at which data are measured. In many applications, a tree is available that organizes the measured variables into various meaningful levels of resolution. In this work, we introduce the tag-lasso, a novel estimation procedure for graphical models that curbs this curse of dimensionality through joint node and edge dimension reduction by leveraging this tree as side information. Node dimension reduction is achieved by a penalty that allows nodes to be aggregated according to the tree structure; edge dimension reduction is achieved through a standard sparsity-inducing penalty. As such, the tag-lasso generalizes the popular glasso approach to sparse graphical modelling. An `R` package called `taglasso` implements the proposed method and is available on the GitHub page (`https://github.com/ineswilms/taglasso`) of the first author.

The tree is a crucial ingredient for performing node aggregation with the tag-lasso and opens up several interesting avenues for future research. On the one hand, multiple trees could be available. For instance, we could think of aggregating stock data by sector, transaction volume, or capitalization. In such cases, it would be interesting to let the trees compete in, for instance, a cross-validation exercise from which the tree that "best" fits the data can be selected to guide the node aggregation. On the other hand, some applications lack the availability of a tree but have more general graph-based structures available that could guide the node aggregation. To this end, it would be interesting to further investigate how the machinery of graph fusion penalties (see e.g., Wang et al., 2016) could be leveraged.

Finally, we do not provide theory in the form of estimation error bounds for the tag-lasso estimator. Possible starting directions to this end can be found in Rothman et al. (2008) or Ravikumar et al. (2011). It would be interesting to investigate the effect of different tree structures on the bounds. In Ravikumar et al. (2011), for instance, the maximal node degree is a relevant quantity. For our penalty on $\mathbf{\Omega}$, which involves an internal optimization

problem over $\mathbf{\Gamma}$ and $\mathbf{D}$, it would be interesting to understand the relevant quantity that captures the interplay between the tree structure and the true node aggregation structure.

Acknowledgments

We thank the referees for their constructive comments which substantially improved the quality of the manuscript. We thank Christian Müller for useful discussions. Jacob Bien was supported in part by NSF CAREER Award DMS-1653017 and NIH Grant R01GM123993.

Appendix A. Proof of Propositions

A.1 Proof of Proposition 1

Proof First, note that $\widetilde{\mathbf{X}}$ follows a K-dimensional multivariate normal distribution with mean zero and covariance matrix $\mathbf{M}^\top(\mathbf{D}+\mathbf{M}\mathbf{C}\mathbf{M}^\top)^{-1}\mathbf{M}$. Next, we re-write this covariance matrix by two successive applications of equation (23) in Henderson and Searle (1981):

$$\begin{aligned}\mathbf{M}^\top(\mathbf{D}+\mathbf{M}\mathbf{C}\mathbf{M}^\top)^{-1}\mathbf{M} &= \mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M} - \mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M}(\mathbf{I}+\mathbf{C}\mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M})^{-1}\mathbf{C}\mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M} \\ &= \left([\mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M}]^{-1}+\mathbf{C}\right)^{-1}.\end{aligned}$$

Hence, the precision matrix of $\widetilde{\mathbf{X}}$ is given by $(\mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M})^{-1}+\mathbf{C}$. Now since $(\mathbf{M}^\top\mathbf{D}^{-1}\mathbf{M})^{-1}$ is diagonal, $c_{ij}=0 \Leftrightarrow \widetilde{X}_i \perp \widetilde{X}_j|\widetilde{\mathbf{X}}_{-\{i,j\}}$, for any $i,j=1,\ldots,K$ and with $\widetilde{\mathbf{X}}_{-\{i,j\}}$ containing all aggregated variables except for aggregate i and j. ■

A.2 Proof of Proposition 2

Proof Denote the tag-lasso solution by $(\widehat{\mathbf{\Omega}},\widehat{\mathbf{\Gamma}},\widehat{\mathbf{D}})$ and let $\widehat{\mathcal{V}}=\{u:\widehat{\gamma}_u\neq\mathbf{0}\}$ be the set of non-zero rows in $\widehat{\mathbf{\Gamma}}$. Define the partial ordering over the nodes of the tree so that $u\leq v$ means that $u\notin$ ancestor(v) and label the nodes of $\widehat{\mathcal{V}}$ so that $u_1\leq u_2\leq\cdots\leq u_{|\widehat{\mathcal{V}}|}$.

Consider the following algorithm to obtain the partition matrix $\widehat{\mathbf{M}}$. We work our way through the nodes in $\widehat{\mathcal{V}}$ in a bottom-up fashion (according to the tree). In a Gram-Schmidt-like procedure, for each node, we subtract out the contributions of its descendants. This approach achieves orthogonality while preserving the binary matrix structure. Note that the approach may result in one or more zero columns in the partition matrix, which are removed at the end before returning the final partition matrix $\widehat{\mathbf{M}}$.

Algorithm 1 Compute partition matrix from tag-lasso solution

Input: $\mathbf{A}, \widehat{\mathcal{V}}$

for $\ell = 1, \dots |\widehat{\mathcal{V}}|$ **do**

$$\mathbf{M}_{u_\ell} \leftarrow \mathbf{A}_{u_\ell} - \sum_{u' \in \text{descendant}(u_l) \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'} \tag{8}$$

end for

Form the $p \times K$ matrix $\widehat{\mathbf{M}}$ (with $K \leq |\widehat{\mathcal{V}}|$) by taking as columns all $\mathbf{M}_{u_\ell}$ such that $\mathbf{M}_{u_\ell} \neq \mathbf{0}$

Output: $\widehat{\mathbf{M}}$

To prove that $\widehat{\mathbf{M}}$ is a partition matrix, we need to show that it is a binary matrix with $\widehat{\mathbf{M}}^\top \widehat{\mathbf{M}}$ a diagonal matrix (i.e., orthogonal columns). Let $\mathcal{T}_u$ denote the subtree rooted at $u \in \mathcal{T}$. For any $u \in \widehat{\mathcal{V}}$, equation (8) implies that

$$\mathbf{A}_u = \mathbf{M}_u + \sum_{u' \in \text{descendant}(u) \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'} = \sum_{u' \in \mathcal{T}_u \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'}. \tag{9}$$

By equation (9), we can then re-write equation (8) as

$$\begin{aligned} \mathbf{M}_a &= \mathbf{A}_a - \sum_{u \in \text{children}(a)} \sum_{u' \in \mathcal{T}_u \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'} \\ &= \mathbf{A}_a - \sum_{u \in \text{children}(a)} \mathbf{A}_u. \end{aligned} \tag{10}$$

Now $\text{supp}(\mathbf{A}_u) \cap \text{supp}(\mathbf{A}_v) = \emptyset$ for $u, v \in \text{children}(a)$ and $u \neq v$, and with supp() denoting the support. Using that $\text{supp}(\mathbf{A}_u) \subseteq \text{supp}(\mathbf{A}_a)$ for $u \in \text{children}(a)$, it follows that $\mathbf{M}_a \in \{0,1\}^p$ and $\text{supp}(\mathbf{M}_a) \subseteq \text{supp}(\mathbf{A}_a)$.

Now, consider $a, b \in \widehat{\mathcal{V}}, a \neq b$ with the following three cases:

1. If $b \in \text{descendant}(a)$, then

$$\begin{aligned} \mathbf{M}_b^\top \mathbf{M}_a &= \mathbf{M}_b^\top \left(\mathbf{A}_a - \sum_{u' \in \text{descendant}(a) \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'} \right) \\ &= \mathbf{M}_b^\top \mathbf{A}_b - \mathbf{M}_b^\top \sum_{u' \in \text{descendant}(a) \cap \widehat{\mathcal{V}}} \mathbf{M}_{u'} \\ &= \mathbf{M}_b^\top \mathbf{A}_b^\top - \mathbf{M}_b^\top \sum_{u' \in \mathcal{T}_b \cap \widehat{\mathcal{V}}} \mathbf{M}'_u, \\ &= \mathbf{M}_b^\top \mathbf{A}_b^\top - \mathbf{M}_b^\top \mathbf{A}_b^\top \\ &= 0. \end{aligned} \tag{11}$$

The second line follows from $\text{supp}(\mathbf{M}_b) \subseteq \text{supp}(\mathbf{A}_b) \subseteq \text{supp}(\mathbf{A}_a)$; the third line follows since $\text{supp}(\mathbf{M}_b) \subseteq \text{supp}(\mathbf{A}_b)$ and $\text{descendant}(a) \cap \mathcal{T}_b = \mathcal{T}_b$ thereby recalling that $b \in \text{descendant}(a)$; and the fourth line follows from equation (9).

2. If $a \in \text{descendant}(b)$, then $\mathbf{M}_b^\top \mathbf{M}_a = 0$ follows from case 1.

3. If a and b are in disjoint branches, then $\text{supp}(\mathbf{A}_a) \cap \text{supp}(\mathbf{A}_b) = \emptyset$ so $\mathbf{M}_a^\top \mathbf{M}_b = 0$.

We have thus constructed a partition matrix $\widehat{\mathbf{M}}$, namely $\widehat{\mathbf{M}} \in \{0,1\}^{p \times K}$ with $\widehat{\mathbf{M}}^\top \widehat{\mathbf{M}}$ diagonal. By construction $\widehat{\mathbf{M}}_u$ is in the column space of $\mathbf{A}_{\widehat{\mathcal{V}}}$, denoted as $\widehat{\mathbf{M}}_u \in \mathcal{L}_{\text{col}}(\mathbf{A}_{\widehat{\mathcal{V}}})$, for each $u \in \widehat{\mathcal{V}}$. Hence, $\mathcal{L}_{\text{col}}(\widehat{\mathbf{M}}) \subseteq \mathcal{L}_{\text{col}}(\mathbf{A}_{\widehat{\mathcal{V}}})$.

It then follows that

$$\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}} = \mathbf{A}\widehat{\mathbf{\Gamma}} = \mathbf{A}_{\widehat{\mathcal{V}}}\widehat{\mathbf{\Gamma}}_{\widehat{\mathcal{V}}} \in \mathcal{L}_{\text{col}}(\widehat{\mathbf{M}}).$$

By symmetry of $\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}}$, $\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}}$ is also in the row space of $\widehat{\mathbf{M}}$. We can therefore write

$$\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}} = \widehat{\mathbf{M}}\widehat{\mathbf{M}}^+(\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}})(\widehat{\mathbf{M}}^+)^\top \widehat{\mathbf{M}}^\top,$$

where $\widehat{\mathbf{M}}^+ = (\widehat{\mathbf{M}}^\top \widehat{\mathbf{M}})^{-1}\widehat{\mathbf{M}}^\top$ is the projection matrix onto the column space of $\widehat{\mathbf{M}}$. Taking $\widehat{\mathbf{C}} = \widehat{\mathbf{M}}^+(\widehat{\mathbf{\Omega}} - \widehat{\mathbf{D}})(\widehat{\mathbf{M}}^+)^\top$ then establishes the result. ■

Appendix B. Tag-lasso Estimator Compared to a Two-stage Benchmark

We compare the performance of the tag-lasso estimator to a two-stage benchmark. For the benchmark, we first apply the tag-lasso estimator with $\lambda_2 = 0$ to solely determine the level of node aggregation. Secondly, we apply the graphical lasso with a hard aggregation constraint (similar to the one used in equation (7)), as provided by the outcome of the first stage. We use the simulation designs detailed in Section 4.1 to compare the estimation accuracy of this two-stage estimator to the tag-lasso and regular glasso. For the tag-lasso and two-stage estimators both the ideal and realistic tree structure are used.

Figure 14 presents the results on KL distance for the five estimators across the four simulation designs. The tag-lasso estimators provide, overall, a considerable improvement in terms of estimation accuracy over their two-stage benchmarks. For the simulation designs where node aggregation is present (chain, random, unbalanced), the two-stage estimators do still outperform the glasso on average.

Appendix C. Details of the LA-ADMM Algorithm

The augmented Lagrangian of (5) is given by

$$\begin{aligned}
& -\text{logdet}(\mathbf{\Omega}^{(1)}) + \text{tr}(\mathbf{S}\mathbf{\Omega}^{(1)}) + 1_\infty\{\mathbf{\Omega}^{(1)} = \mathbf{\Omega}^{(1)^\top}, \mathbf{\Omega}^{(1)} \succ \mathbf{0}\} + \langle \mathbf{U}^{(1)}, \mathbf{\Omega}^{(1)} - \mathbf{\Omega}\rangle + \frac{\rho}{2}\|\mathbf{\Omega}^{(1)} - \mathbf{\Omega}\|_F^2 \\
+ \quad & \lambda_1 \|\mathbf{\Gamma}_{-r}^{(1)}\|_{2,1} + 1_\infty\{\boldsymbol{\gamma}_r^{(1)} = \gamma^{(1)}\mathbf{1}_p\} + \langle \mathbf{U}^{(4)}, \mathbf{\Gamma}^{(1)} - \mathbf{\Gamma}\rangle + \frac{\rho}{2}\|\mathbf{\Gamma}^{(1)} - \mathbf{\Gamma}\|_F^2 \\
+ \quad & 1_\infty\{\mathbf{\Omega}^{(2)} = \mathbf{A}\mathbf{\Gamma}^{(2)} + \mathbf{D}, \mathbf{D} \text{ diag.}, D_{jj} \geq 0\} + \langle \mathbf{U}^{(2)}, \mathbf{\Omega}^{(2)} - \mathbf{\Omega}\rangle + \frac{\rho}{2}\|\mathbf{\Omega}^{(2)} - \mathbf{\Omega}\|_F^2 \\
+ \quad & \langle \mathbf{U}^{(5)}, \mathbf{\Gamma}^{(2)} - \mathbf{\Gamma}\rangle + \frac{\rho}{2}\|\mathbf{\Gamma}^{(2)} - \mathbf{\Gamma}\|_F^2 \\
+ \quad & \lambda_2 \|\mathbf{\Omega}^{-\text{diag}(3)}\|_1 + \langle \mathbf{U}^{(3)}, \mathbf{\Omega}^{(3)} - \mathbf{\Omega}\rangle + \frac{\rho}{2}\|\mathbf{\Omega}^{(2)} - \mathbf{\Omega}\|_F^2, \qquad (12)
\end{aligned}$$

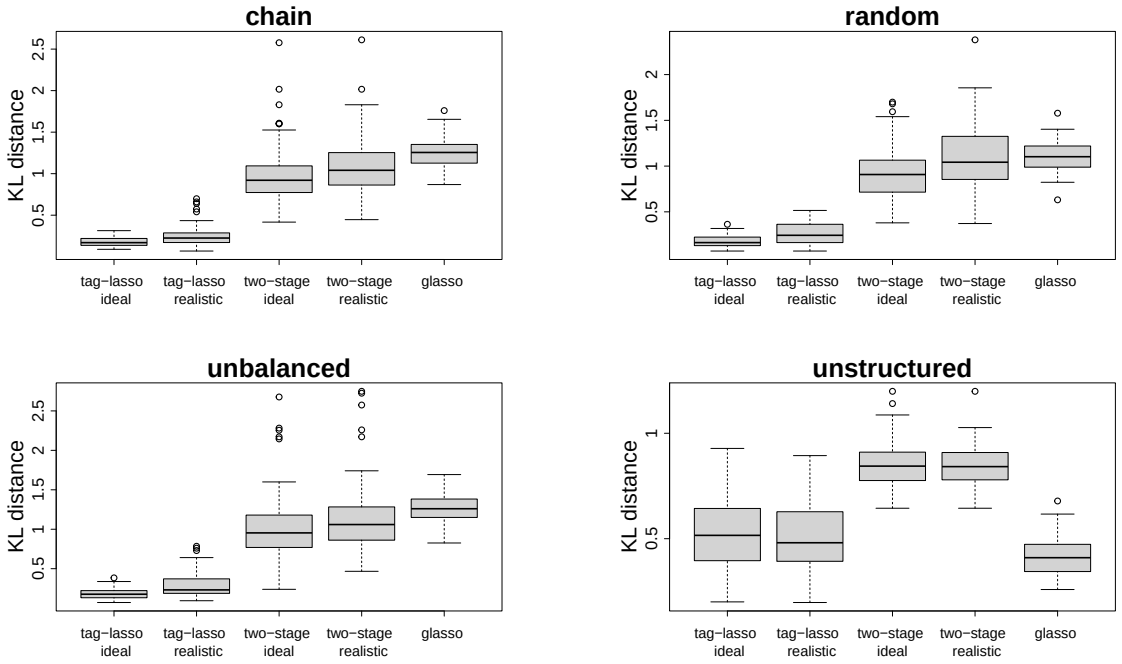


Figure 14: Estimation accuracy of the tag-lasso estimators compared to the two-stage estimators and glasso.

where $\mathbf{U}^{(i)}$ (for $i = 1, \ldots, 5$) are the dual variables, and ρ is a penalty parameter. Note that equation (12) is of the same form as Equation (3.1) in Boyd et al. (2011) and thus involves iterating three basic steps: (i) minimization with respect to $(\mathbf{\Omega}^{(1)}, \mathbf{\Omega}^{(2)}, \mathbf{\Omega}^{(3)}, \mathbf{\Gamma}^{(1)}, \mathbf{\Gamma}^{(2)}, \mathbf{D})$, (ii) minimization with respect to $(\mathbf{\Omega}, \mathbf{\Gamma})$, and (iii) update of $(\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(5)})$.

Step (i) decouples into four independent problems, whose solutions are worked out in Sections C.1-C.4. Step (ii) involves the minimization of a differentiable function of $\mathbf{\Omega}$ and $\mathbf{\Gamma}$ and boils down to the calculation of simple averages, as shown in Section C.5. Step (iii)'s update of the dual variables is provided in C.6.

Algorithms 2-3 then provide an overview of the LA-ADMM algorithm to solve problem (5). We use the LA-ADMM algorithm with $\rho_1 = 0.01$, $\texttt{T}_{\text{stages}} = 10$, $\texttt{maxit} = 100$.

C.1 Solving for $\Omega^{(1)}$

Minimizing the augmented Lagrangian with respect to $\mathbf{\Omega}^{(1)}$ gives $\widehat{\mathbf{\Omega}}^{(1)}_{k+1}$, the solution to the optimization problem

$$\begin{aligned}
&\min_{\mathbf{\Omega}^{(1)}} \{-\text{logdet}(\mathbf{\Omega}^{(1)}) + \text{tr}(\mathbf{S}\mathbf{\Omega}^{(1)}) + \langle \mathbf{U}^{(1)}, \mathbf{\Omega}^{(1)} - \mathbf{\Omega} \rangle + \frac{\rho}{2} \|\mathbf{\Omega}^{(1)} - \mathbf{\Omega}\|_F^2 \ \text{ s.t. } \ \mathbf{\Omega}^{(1)} = {\mathbf{\Omega}^{(1)}}^\top, \mathbf{\Omega}^{(1)} \succ \mathbf{0}\} \\
&= \min_{\mathbf{\Omega}^{(1)}} \{-\text{logdet}(\mathbf{\Omega}^{(1)}) + \text{tr}(\mathbf{S}\mathbf{\Omega}^{(1)}) + \frac{\rho}{2} \|\mathbf{\Omega}^{(1)} - (\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(1)}_k/\rho)\|_F^2 \ \text{ s.t. } \ \mathbf{\Omega}^{(1)} = {\mathbf{\Omega}^{(1)}}^\top, \mathbf{\Omega}^{(1)} \succ \mathbf{0}\}.
\end{aligned}$$

The solution should satisfy the first order optimality condition

$$\rho\widehat{\mathbf{\Omega}}^{(1)}_{k+1} - \left(\widehat{\mathbf{\Omega}}^{(1)}_{k+1}\right)^{-1} = \rho\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(1)}_k - \mathbf{S}. \tag{13}$$

Algorithm 2 LA-ADMM

Input: $\mathbf{S}, \mathbf{A}, p, |\mathcal{T}|, \lambda_1, \lambda_2, \rho_1, \texttt{maxit}, \texttt{T}_{\text{stages}}$

Initialization: Set

$$\widehat{\mathbf{\Omega}}_0 \leftarrow \mathbf{0}; \widehat{\mathbf{\Gamma}}_0 \leftarrow \mathbf{0}$$
$$t \leftarrow 0$$

for $t \leq \texttt{T}_{\text{stages}}$ **do**

$$t \leftarrow t+1$$
$$(\widehat{\mathbf{\Omega}}_t, \widehat{\mathbf{\Gamma}}_t, \widehat{\mathbf{D}}_t) \leftarrow \text{ADMM}(\mathbf{S}, \mathbf{A}, p, |\mathcal{T}|, \lambda_1, \lambda_2, \rho_t, \texttt{maxit}, \widehat{\mathbf{\Omega}}_{t-1}, \widehat{\mathbf{\Gamma}}_{t-1})$$
$$\rho_{t+1} \leftarrow 2\rho_t$$

end for

Output: $\widehat{\mathbf{\Omega}}_{\texttt{T}_{\text{stages}}}$, $\widehat{\mathbf{\Gamma}}_{\texttt{T}_{\text{stages}}}$, $\widehat{\mathbf{D}}_{\texttt{T}_{\text{stages}}}$

This means that the eigenvectors of $\widehat{\mathbf{\Omega}}^{(1)}_{k+1}$ are the same as the eigenvectors of $\rho\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(1)}_k - \mathbf{S}$ and that the eigenvalues of $\widehat{\mathbf{\Omega}}^{(1)}_{k+1}$ are a simple function of the eigenvalues of $\rho\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(1)}_k - \mathbf{S}$. Consider the orthogonal eigenvalue decomposition of right hand side:

$$\rho\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(1)}_k - \mathbf{S} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top,$$

where $\mathbf{\Lambda} = \mathbf{diag}(\delta_1, \ldots, \delta_p)$ and $\mathbf{Q}\mathbf{Q}^\top = \mathbf{Q}^\top\mathbf{Q} = \mathbf{I}$. Multiply (13) by $\mathbf{Q}^\top$ on the left and $\mathbf{Q}$ on the right

$$\rho\bar{\mathbf{\Omega}}^{(1)}_{k+1} - (\bar{\mathbf{\Omega}}^{(1)}_{k+1})^{-1} = \mathbf{\Lambda}, \text{ with } \bar{\mathbf{\Omega}}^{(1)}_{k+1} = \mathbf{Q}^\top\widehat{\mathbf{\Omega}}^{(1)}_{k+1}\mathbf{Q}.$$

Then

$$\widehat{\mathbf{\Omega}}^{(1)}_{k+1} = \mathbf{Q}\bar{\mathbf{\Omega}}^{(1)}_{k+1}\mathbf{Q}^\top, \text{ with } \bar{\Omega}^{(1)}_{k+1,jj} = \frac{\delta_j + \sqrt{\delta_j^2 + 4\rho}}{2\rho}. \tag{14}$$

C.2 Solving for $\Gamma^{(1)}$

Minimizing the augmented Lagrangian with respect to $\mathbf{\Gamma}^{(1)}$ gives

$$\widehat{\mathbf{\Gamma}}^{(1)}_{k+1} := \underset{\mathbf{\Gamma}^{(1)}}{\text{argmin}}\{\frac{\rho}{2}\|\mathbf{\Gamma}^{(1)} - (\widehat{\mathbf{\Gamma}}_k - \widehat{\mathbf{U}}^{(4)}_k/\rho)\|_F^2 + \lambda_1\|\mathbf{\Gamma}^{(1)}_{-r}\|_{2,1} \text{ s.t. } \boldsymbol{\gamma}^{(1)}_r = \gamma^{(1)}\mathbf{1}_p\}.$$

The solution is groupwise soft-thresholding:

$$\widehat{\mathbf{\Gamma}}^{(1)}_{k+1,j} = \begin{cases} S_G(\widehat{\mathbf{\Gamma}}_{k,j} - \widehat{\mathbf{U}}^{(4)}_{k,j}/\rho, \lambda_1/\rho), & \text{if } j = 1, \ldots, |\mathcal{T}|\backslash\{r\} \\ \widehat{\gamma}_k\mathbf{1}_p, & \text{if } j = r. \end{cases} \tag{15}$$

with the group-wise soft-thresholding operator $S_G(\boldsymbol{\gamma}, \lambda) = \max(1 - \lambda/\|\boldsymbol{\gamma}\|_2, 0)\boldsymbol{\gamma}$ applied to $\boldsymbol{\gamma} \in \mathbb{R}^p$, and $\widehat{\gamma}_k$ is equal to the average of the p-dimensional vector $\widehat{\mathbf{\Gamma}}_{k,r} - \widehat{\mathbf{U}}^{(4)}_{k,r}/\rho$. Note that

Algorithm 3 ADMM

Input: $\mathbf{S}, \mathbf{A}, p, |\mathcal{T}|, \lambda_1, \lambda_2, \rho, \texttt{maxit}, \mathbf{\Omega}_0, \mathbf{\Gamma}_0$.

Initialization: Set

$\widehat{\mathbf{\Omega}}_0^{(i)} \leftarrow \widehat{\mathbf{U}}_0^{(i)} \leftarrow \mathbf{\Omega}_0 \quad$ for $i = 1, \ldots, 3$

$\widehat{\mathbf{\Gamma}}_0^{(j)} \leftarrow \widehat{\mathbf{U}}_0^{(j+3)} \leftarrow \mathbf{\Gamma}_0$ for $j = 1, \ldots, 2$

$k \leftarrow 0$

$\widetilde{\mathbf{A}} \leftarrow \begin{pmatrix} \mathbf{A} \\ \mathbf{I}_{|\mathcal{T}|} \end{pmatrix}$

$\widetilde{\mathbf{A}}^+ \leftarrow (\widetilde{\mathbf{A}}^\top \widetilde{\mathbf{A}})^{-1} \widetilde{\mathbf{A}}^\top$

$\bar{\mathbf{A}} \leftarrow \mathbf{I}_{p+|\mathcal{T}|} - \widetilde{\mathbf{A}} \widetilde{\mathbf{A}}^+$

$\mathbf{C} \leftarrow (\mathbf{I}_p : \mathbf{0}_{p \times |\mathcal{T}|})^\top - \widetilde{\mathbf{A}} \widetilde{\mathbf{A}}^+_{1:p}$ with $\widetilde{\mathbf{A}}^+_{1:p}$ the first p columns of $\widetilde{\mathbf{A}}^+$

$\bar{\mathbf{C}} \leftarrow \text{diag}(\mathbf{C}^\top \mathbf{C})^{-1}$

for $k \leq \texttt{maxit}$ **do**

$k \leftarrow k + 1$

$\widehat{\mathbf{\Omega}}_k^{(1)} \leftarrow \mathbf{Q} \bar{\mathbf{\Omega}}_{k-1} \mathbf{Q}^\top$, see equation (14).

$\widehat{\Omega}_{k,ij}^{(3)} \leftarrow S(\widehat{\Omega}_{k-1,ij} - \widehat{U}_{k-1,ij}^{(3)}/\rho,\ \lambda_2/\rho), \forall\ i, j = 1, \ldots, p$, see equation (19).

$\widehat{\mathbf{\Gamma}}_{k,j}^{(1)} \leftarrow S_G(\widehat{\mathbf{\Gamma}}_{k-1,j} - \widehat{\mathbf{U}}_{k-1,j}^{(4)}/\rho,\ \lambda_1/\rho), \forall j = 1, \ldots, |\mathcal{T}| \backslash \{r\}$, see equation (15).

$\widehat{\mathbf{\Gamma}}_{k,r}^{(1)} \leftarrow \widehat{\gamma}_{k-1} \mathbf{1}_p$, see equation (15).

$\text{diag}(\widehat{\mathbf{D}}_k) \leftarrow \bar{\mathbf{C}} \text{diag}((\bar{\mathbf{A}} \widetilde{\mathbf{M}})^\top \mathbf{C})_+$, see equation (18).

$\widehat{\mathbf{\Gamma}}_k^{(2)} \leftarrow \widetilde{\mathbf{A}}^+ (\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}}_k)$, see equation (17).

$\widehat{\mathbf{\Omega}}_k^{(2)} = \mathbf{A} \widehat{\mathbf{\Gamma}}_k^{(2)} + \widehat{\mathbf{D}}_k$, see equation (16)

$\widehat{\mathbf{\Omega}}_k \leftarrow (\widehat{\mathbf{\Omega}}_k^{(1)} + \widehat{\mathbf{\Omega}}_k^{(2)} + \widehat{\mathbf{\Omega}}_k^{(3)})/3$

$\widehat{\mathbf{\Gamma}}_k \leftarrow (\widehat{\mathbf{\Gamma}}_k^{(1)} + \widehat{\mathbf{\Gamma}}_k^{(2)})/2$

$\widehat{\mathbf{U}}_k^{(i)} \leftarrow \widehat{\mathbf{U}}_{k-1}^{(i)} + \rho \left(\widehat{\mathbf{\Omega}}_k^{(i)} - \widehat{\mathbf{\Omega}}_k \right),$ for $i = 1, \ldots, 3$

$\widehat{\mathbf{U}}_k^{(j+3)} \leftarrow \widehat{\mathbf{U}}_{k-1}^{(j+3)} + \rho \left(\widehat{\mathbf{\Gamma}}_k^{(j)} - \widehat{\mathbf{\Gamma}}_k \right),$ for $j = 1, \ldots, 2$

end for

Output: $\widehat{\mathbf{\Omega}}_{\texttt{maxit}},\ \widehat{\mathbf{\Gamma}}_{\texttt{maxit}},\ \widehat{\mathbf{D}}_{\texttt{maxit}}$

in this Appendix we use the capitalized $\mathbf{\Gamma}_j$ notation to index the j^{th} row of the matrix $\mathbf{\Gamma}$ whereas we use lowercase $\boldsymbol{\gamma}_u$ when indexing a node u based on the tree structure in Section 2 of the main paper.

C.3 Solving for $\Omega^{(2)}, \Gamma^{(2)}, D$

Minimizing the augmented Lagrangian with respect to $\mathbf{\Omega}^{(2)}, \mathbf{\Gamma}^{(2)}, \mathbf{D}$ gives

$$(\widehat{\mathbf{\Omega}}^{(2)}_{k+1}, \widehat{\mathbf{\Gamma}}^{(2)}_{k+1}, \widehat{\boldsymbol{D}}_{k+1}) := \underset{\mathbf{\Omega}^{(2)}, \mathbf{\Gamma}^{(2)}, \mathbf{D}}{\operatorname{argmin}} \{\frac{\rho}{2}\|\mathbf{\Omega}^{(2)} - (\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(2)}_k/\rho)\|_F^2 + \frac{\rho}{2}\|\mathbf{\Gamma}^{(2)} - (\widehat{\mathbf{\Gamma}}_k - \widehat{\mathbf{U}}^{(5)}_k/\rho)\|_F^2$$
$$\text{s.t. } \mathbf{\Omega}^{(2)} = \mathbf{A}\mathbf{\Gamma}^{(2)} + \mathbf{D},\ \mathbf{D} \text{ diagonal},\ D_{jj} \geq 0 \text{ for } j = 1, \ldots, p\}.$$

The solution

$$\widehat{\mathbf{\Omega}}^{(2)}_{k+1} = \mathbf{A}\widehat{\mathbf{\Gamma}}^{(2)}_{k+1} + \widehat{\mathbf{D}}_{k+1} \tag{16}$$

is immediate and we are left with

$$(\widehat{\mathbf{\Gamma}}^{(2)}_{k+1}, \widehat{\boldsymbol{D}}_{k+1}) := \underset{\mathbf{\Gamma}^{(2)}, \mathbf{D}}{\operatorname{argmin}}\{\frac{1}{2}\|\widetilde{\mathbf{A}}\mathbf{\Gamma}^{(2)} + \widetilde{\mathbf{D}} - \widetilde{\mathbf{M}}\|_F^2 \text{ s.t. } \mathbf{D} \text{ diagonal},\ D_{jj} \geq 0 \text{ for } j = 1, \ldots, p\}$$

where we have substituted $\mathbf{\Omega}^{(2)} = \mathbf{A}\mathbf{\Gamma}^{(2)} + \mathbf{D}$ and we denote

$$\widetilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A} \\ \mathbf{I}_{|\mathcal{T}|} \end{pmatrix} \in \mathbb{R}^{(p+|\mathcal{T}|)\times|\mathcal{T}|},\ \widetilde{\mathbf{D}} = \begin{pmatrix} \mathbf{D} \\ \mathbf{0}_{|\mathcal{T}|\times p} \end{pmatrix} \in \mathbb{R}^{(p+|\mathcal{T}|)\times p},\ \text{and } \widetilde{\mathbf{M}} = \begin{pmatrix} \widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(2)}_k/\rho \\ \widehat{\mathbf{\Gamma}}_k - \widehat{\mathbf{U}}^{(5)}_k/\rho \end{pmatrix} \in \mathbb{R}^{(p+|\mathcal{T}|)\times p}.$$

The solution

$$\begin{aligned} \widehat{\mathbf{\Gamma}}^{(2)}_{k+1} &= (\widetilde{\mathbf{A}}^\top\widetilde{\mathbf{A}})^{-1}\widetilde{\mathbf{A}}^\top(\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}}_{k+1}) \\ &= (\mathbf{A}^\top\mathbf{A} + \mathbf{I}_{|\mathcal{T}|})^{-1}(\mathbf{A}^\top : \mathbf{I}_{|\mathcal{T}|})(\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}}_{k+1}) \end{aligned} \tag{17}$$

is immediate and we are left with

$$\begin{aligned} \widehat{\boldsymbol{D}}_{k+1} &:= \underset{\mathbf{D}}{\operatorname{argmin}}\{\frac{1}{2}\|(\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}}) - \widetilde{\mathbf{A}}(\widetilde{\mathbf{A}}^\top\widetilde{\mathbf{A}})^{-1}\widetilde{\mathbf{A}}^\top(\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}})\|_F^2 \text{ s.t. } \mathbf{D} \text{ diag.},\ D_{jj} \geq 0, j = 1, \ldots, p, \} \\ &= \underset{\mathbf{D}}{\operatorname{argmin}}\{\frac{1}{2}\|(\mathbf{I}_{p+|\mathcal{T}|} - \widetilde{\mathbf{A}}(\widetilde{\mathbf{A}}^\top\widetilde{\mathbf{A}})^{-1}\widetilde{\mathbf{A}}^\top)(\widetilde{\mathbf{M}} - \widetilde{\mathbf{D}})\|_F^2 \text{ s.t. } \mathbf{D} \text{ diag.},\ D_{jj} \geq 0,\ j = 1, \ldots, p, \} \\ &= \underset{\mathbf{D}}{\operatorname{argmin}}\{\frac{1}{2}\|\mathbf{B} - \mathbf{C}\mathbf{D}\|_F^2 \text{ s.t. } \mathbf{D} \text{ diag.},\ D_{jj} \geq 0, j = 1, \ldots, p, \}, \end{aligned}$$

with $\mathbf{B} = (\mathbf{I}_{p+|\mathcal{T}|} - \widetilde{\mathbf{A}}(\widetilde{\mathbf{A}}^\top\widetilde{\mathbf{A}})^{-1}\widetilde{\mathbf{A}}^\top)\widetilde{\mathbf{M}} \in \mathbb{R}^{(p+|\mathcal{T}|)\times p},\ \mathbf{C} = (\mathbf{I}_p : \mathbf{0}_{p\times|\mathcal{T}|})^\top - \widetilde{\mathbf{A}}(\widetilde{\mathbf{A}}^\top\widetilde{\mathbf{A}})^{-1}\mathbf{A}^\top \in \mathbb{R}^{(p+|\mathcal{T}|)\times p}$. The solution is

$$\operatorname{diag}(\widehat{\mathbf{D}}_{k+1}) = \operatorname{diag}(\mathbf{C}^\top\mathbf{C})^{-1}\operatorname{diag}(\mathbf{B}^\top\mathbf{C})_+. \tag{18}$$

C.4 Solving for $\Omega^{(3)}$

Minimizing the augmented Lagrangian with respect to $\mathbf{\Omega}^{(3)}$ gives

$$\begin{aligned}
\widehat{\mathbf{\Omega}}^{(3)}_{k+1} &:= \underset{\mathbf{\Omega}^{(3)}}{\operatorname{argmin}}\{\langle \mathbf{U}^{(3)}, \mathbf{\Omega}^{(3)} - \mathbf{\Omega}\rangle + \frac{\rho}{2}\|\mathbf{\Omega}^{(3)} - \mathbf{\Omega}\|_F^2 + \lambda_2\|\mathbf{\Omega}^{-\text{diag}(3)}\|_1\} \\
&= \underset{\mathbf{\Omega}^{(3)}}{\operatorname{argmin}}\{\frac{\rho}{2}\|\mathbf{\Omega}^{(3)} - (\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(3)}_k/\rho)\|_F^2 + \frac{1}{2\rho}\|\mathbf{U}^{(3)}\|_F^2 + \lambda_2\|\mathbf{\Omega}^{-\text{diag}(3)}\|_1\} \\
&= \underset{\mathbf{\Omega}^{(3)}}{\operatorname{argmin}}\{\frac{\rho}{2}\|\mathbf{\Omega}^{(3)} - (\widehat{\mathbf{\Omega}}_k - \widehat{\mathbf{U}}^{(3)}_k/\rho)\|_F^2 + \lambda_2\|\mathbf{\Omega}^{-\text{diag}(3)}\|_1\}.
\end{aligned}$$

The solution is simply elementwise soft-thresholding:

$$\widehat{\Omega}^{(3)}_{k+1,ij} = \begin{cases} S(\widehat{\Omega}_{k,ij} - \widehat{U}^{(3)}_{k,ij}/\rho, \lambda_2/\rho), & \text{if } i \neq j \\ \widehat{\Omega}_{k,ij} - \widehat{U}^{(3)}_{k,ij}/\rho, & \text{if } i = j, \end{cases} \tag{19}$$

with the soft-threshold operator $S(\omega, \lambda) = \text{sign}(\omega)\max(|\omega| - \lambda, 0)$ applied to $\omega \in \mathbb{R}$.

C.5 Update Variables Ω and Γ

Minimizing the augmented Lagrangian with respect to variables $\mathbf{\Omega}$ and $\mathbf{\Gamma}$ gives

$$\widehat{\mathbf{\Omega}}_{k+1} := \underset{\mathbf{\Omega}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{3} \|\widehat{\mathbf{\Omega}}^{(i)}_{k+1} - (\mathbf{\Omega} - \widehat{\mathbf{U}}^{(i)}_k/\rho)\|_F^2 \right\} = \bar{\mathbf{\Omega}}_{k+1} + \frac{1}{\rho}\bar{\mathbf{U}}^{\Omega}_k \tag{20}$$

$$\widehat{\mathbf{\Gamma}}_{k+1} := \underset{\mathbf{\Gamma}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{2} \|\widehat{\mathbf{\Gamma}}^{(i)}_{k+1} - (\mathbf{\Gamma} - \widehat{\mathbf{U}}^{(i+3)}_k/\rho)\|_F^2 \right\} = \bar{\mathbf{\Gamma}}_{k+1} + \frac{1}{\rho}\bar{\mathbf{U}}^{\Gamma}_k, \tag{21}$$

where $\bar{\mathbf{\Omega}}_k := \dfrac{\widehat{\mathbf{\Omega}}^{(1)}_k + \widehat{\mathbf{\Omega}}^{(2)}_k + \widehat{\mathbf{\Omega}}^{(3)}_k}{3}, \bar{\mathbf{U}}^{\Omega}_k := \dfrac{\widehat{\mathbf{U}}^{(1)}_k + \widehat{\mathbf{U}}^{(2)}_k + \widehat{\mathbf{U}}^{(3)}_k}{3}, \bar{\mathbf{\Gamma}}_k := \dfrac{\widehat{\mathbf{\Gamma}}^{(1)}_k + \widehat{\mathbf{\Gamma}}^{(2)}_k}{2}, \bar{\mathbf{U}}^{\Gamma}_k := \dfrac{\widehat{\mathbf{U}}^{(4)}_k + \widehat{\mathbf{U}}^{(5)}_k}{2}$.

C.6 Update Dual Variables

The updates of the dual variables are given by

$$\begin{aligned}
\widehat{\mathbf{U}}^{(i)}_{k+1} &:= \widehat{\mathbf{U}}^{(i)}_k + \rho\left(\widehat{\mathbf{\Omega}}^{(i)}_{k+1} - \widehat{\mathbf{\Omega}}_{k+1}\right), \text{ for } i = 1, \ldots, 3 \\
\widehat{\mathbf{U}}^{(j+3)}_{k+1} &:= \widehat{\mathbf{U}}^{(j+3)}_k + \rho\left(\widehat{\mathbf{\Gamma}}^{(j)}_{k+1} - \widehat{\mathbf{\Gamma}}_{k+1}\right), \text{ for } j = 1, \ldots, 2.
\end{aligned}$$

Similarly, averaging the first three updates and the latter two gives

$$\bar{\mathbf{U}}^{\Omega}_{k+1} := \bar{\mathbf{U}}^{\Omega}_k + \rho\left(\bar{\mathbf{\Omega}}_{k+1} - \widehat{\mathbf{\Omega}}_{k+1}\right), \text{ for } i = 1, \ldots, 3 \tag{22}$$

$$\bar{\mathbf{U}}^{\Gamma}_{k+1} := \bar{\mathbf{U}}^{\Gamma}_k + \rho\left(\bar{\mathbf{\Gamma}}_{k+1} - \widehat{\mathbf{\Gamma}}_{k+1}\right), \text{ for } j = 1, \ldots, 2. \tag{23}$$

Substituting (20) and (21) into (22) and (23) yields that $\bar{\mathbf{U}}^{\Omega}_{k+1} = \bar{\mathbf{U}}^{\Gamma}_{k+1} = \mathbf{0}$ after the first iteration.

Appendix D. Additional Simulation Results

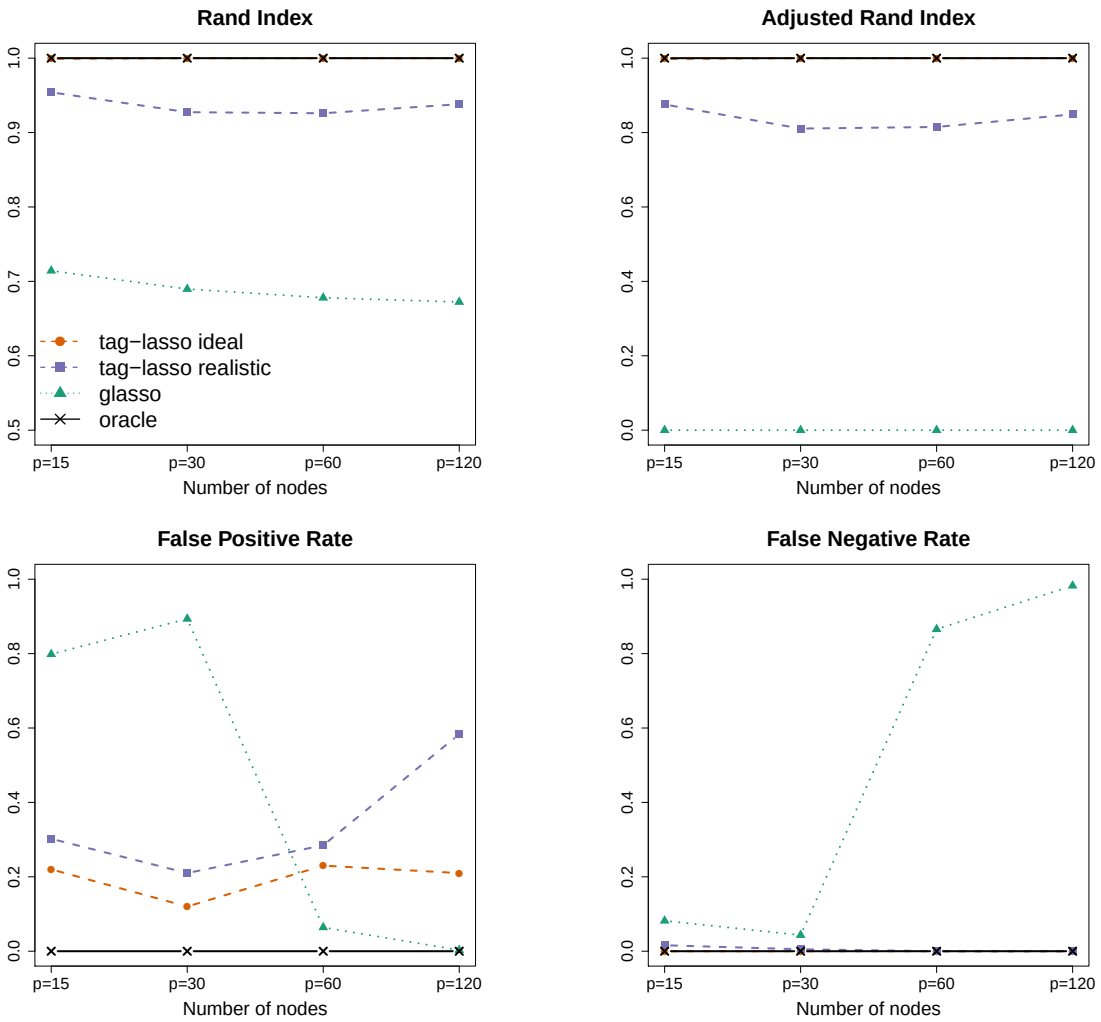


Figure 15: Simulation results for increasing number of nodes p. Top: Aggregation performance (RI: left; ARI: right); Bottom: Sparsity recovery (FPR: left; FNR: right) of the four estimators

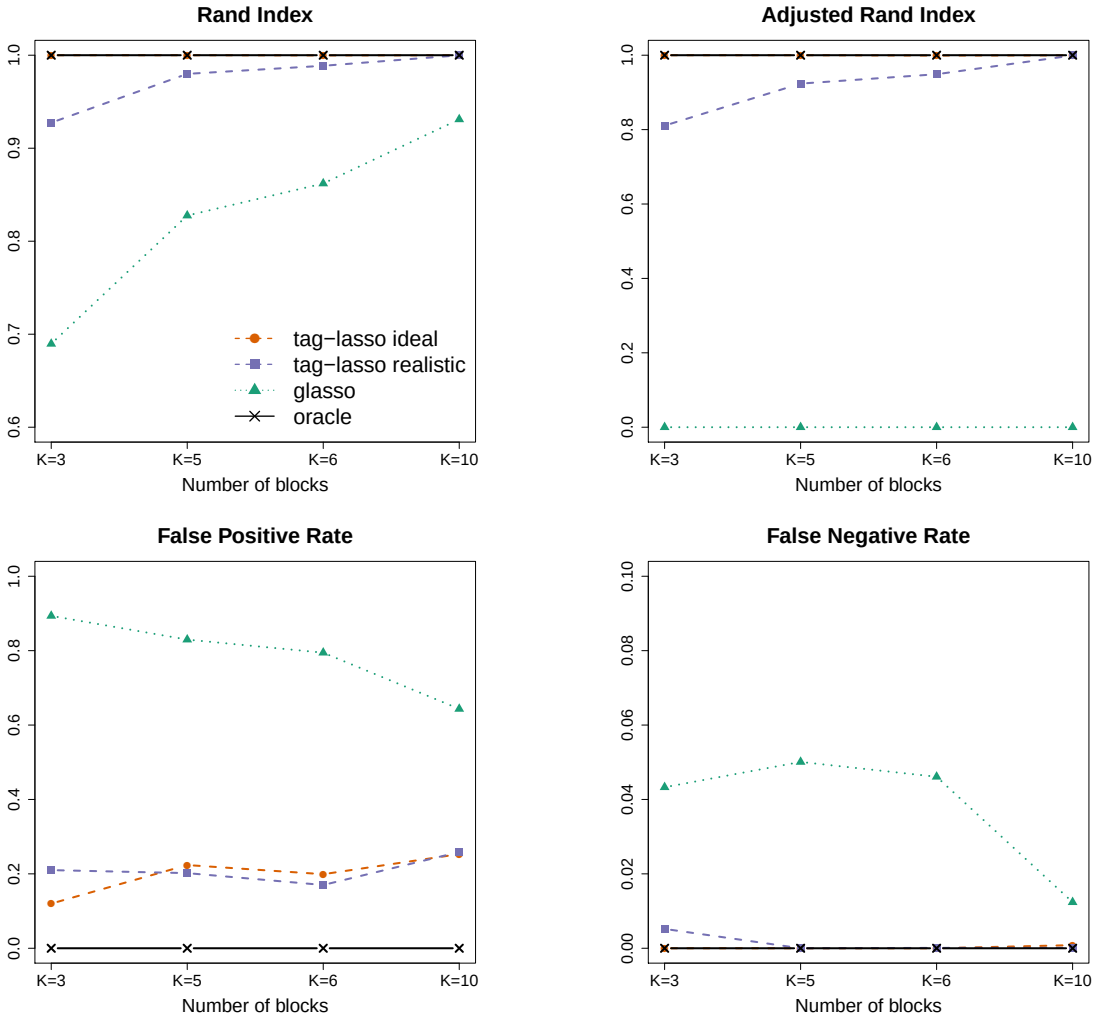


Figure 16: Simulation results for increasing number of blocks K. Top: Aggregation performance (RI: left; ARI: right); Bottom: Sparsity recovery (FPR: left; FNR: right) of the four estimators

Appendix E. Financial Application: Data Description

Abbreviation	Description	Location
DJI	Dow Jones Industrial Average	US
IXIC	Nasdaq 100	US
SPX	S&P 500 Index	US
RUT	Russel 2000	US
GSPTSE	S&P/TSX Composite index	Canada
BVSP	BVSP BOVESPA Index	Brazil
MXX	IPC Mexico	Mexico
OMXC20	OMX Copenhagen 20 Index	Denmark
OMXHPI	OMX Helsinki All Share Index	Finland
OMXSPI	OMX Stockholm All Share Index	Sweden
OSEAX	Oslo Exchange All-share Index	Norway
GDAXI	Deutscher Aktienindex	Germany
SSMI	Swiss Stock Market Index	Switzerland
BVLG	Portuguese Stock Index	Portugal
FTMIB	Financial Times Stock Exchange Milano Indice di Borsa	Italy
IBEX	Iberia Index 35	Spain
SMSI	General Madrid Index	Spain
AEX	Amsterdam Exchange Index	Netherlands
BFX	Bell 20 Index	Belgium
FCHI	Cotation Assistée en Continue 40	France
FTSE	Financial Times Stock Exchange 100	UK
STOXX50E	EURO STOXX 50	Europe
HSI	HANG SENG Index	Hong Kong
KS11	Korea Composite Stock Price Index (KOSPI)	South Korea
N225	Nikkei 225	Japan
SSEC	Shanghai Composite Index	China
STI	Straits Times Index	Singapore
KSE	Karachi SE 100 Index	Pakistan
BSESN	S&P Bombay Stock Exchange Sensitive Index	India
NSEI	NIFTY 50	India
AORD	All Ordinaries Index	Australia

Table 4: Financial Application: Data Description, as taken from `https://realized.oxford-man.ox.ac.uk/data/assets`.

References

J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 44(2):139–160, 1982.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine*

Learning Research, 9(Mar):485–516, 2008.

A. Belloni and V. Chernozhukov. Least squares after model selection in high-dimensional sparse models. *Bernoulli*, 19(2):521–547, 2013.

J. Bien. The simulator: an engine to streamline simulations. *arXiv preprint arXiv:1607.00021*, 2016.

J. Bien, X. Yan, L. Simpson, and C. L. Müller. Tree-aggregated predictive modeling of microbiome data. *bioRxiv*, 2020.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011.

C. Brownlees, G. S. Gumundsson, and G. Lugosi. Community detection in partial correlation network models. *Journal of Business & Economic Statistics*, (just-accepted):1–33, 2020.

F. Bunea, C. Giraud, X. Luo, M. Royer, and N. Verzelen. Model assisted variable clustering: minimax-optimal recovery and algorithms. *The Annals of Statistics*, 48(1):111–137, 2020.

T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

T. Cai, W. Liu, and H. Zhou. Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. *The Annals of Statistics*, 44(2):455–488, 2016.

B. J. Callahan, P. J. McMurdie, and S. P. Holmes. Exact sequence variants should replace operational taxonomic units in marker-gene data analysis. *The ISME journal*, 11(12): 2639–2643, 2017.

Y. Cao, W. Lin, and H. Li. Large covariance estimation for compositional data via composition-adjusted thresholding. *Journal of the American Statistical Association*, 114 (526):759–772, 2019.

V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. *The Annals of Statistics*, 40(4):1935–1967, 2012.

F. Corsi. A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196, 2009.

C. Eisenach, F. Bunea, Y. Ning, and C. Dinicu. High-dimensional inference for cluster-based graphical models. *Journal of Machine Learning Research*, 21(53):1–55, 2020.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

H. V. Henderson and S. R. Searle. On deriving the inverse of a sum of matrices. *Siam Review*, 23(1):53–60, 1981.

L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

Z. D. Kurtz, C. L. Müller, E. R. Miraldi, D. R. Littman, M. J. Blaser, and R. A. Bonneau. Sparse and compositionally robust inference of microbial ecological networks. *PLoS Comput Biol*, 11(5):e1004226, 2015.

Z. D. Kurtz, R. Bonneau, and C. L. Müller. Disentangling microbial associations from hidden environmental and technical factors via latent graphical models. *bioRxiv*, 2019.

C. Lo and R. Marculescu. Pglasso: Microbial community detection through phylogenetic graphical lasso. *arXiv preprint arXiv:1807.08039*, 2018.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of statistics*, 34(3):1436–1462, 2006.

T. Millington and M. Niranjan. Quantifying influence in financial markets via partial correlation network inference. In *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 306–311. IEEE, 2019.

J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

E. Pircalabelu and G. Claeskens. Community-based group graphical lasso. *Journal of Machine Learning Research*, 21(64):1–32, 2020.

R Core Team. *R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.*, 2017. URL `https://www.R-project.org/`.

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

P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

J. Rivera-Pinto, J. J. Egozcue, V. Pawlowsky-Glahn, R. Paredes, M. Noguera-Julian, and M. L. Calle. Balances: a new perspective for microbiome analysis. *mSystems*, 3(4):1–12, 2018. doi: 10.1128/mSystems.00053-18. URL `https://msystems.asm.org/content/3/4/e00053-18`.

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

K. M. Tan, D. Witten, and A. Shojaie. The cluster graphical lasso for improved estimation of Gaussian graphical models. *Computational statistics & data analysis*, 85:23–36, 2015.

Y.X. Wang, J. Sharpnack, A. J. Smola, and R. J. Tibshirani. Trend filtering on graphs. *Journal of Machine Learning Research*, 17(105):1–41, 2016.

Y. Xu, M. Liu, Q. Lin, and T. Yang. Admm without a fixed penalty parameter: Faster convergence with new adaptive penalization. In *Advances in Neural Information Processing Systems*, pages 1267–1277, 2017.

X. Yan and J. Bien. Rare feature selection in high dimensions. *Journal of the American Statistical Association*, 116(534):887–900, 2021.

M. Yuan. High dimensional inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, 11:2261–2286, 2010.

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