Joint Estimation of Graphical Models Across Multiple Classes

Tavis Abrahamsen, Ray Bai,
Syed Rahman, Andrey Skripnikov

Department of Statistics
University of Florida

April 22, 2015

Background

Suppose that observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^P$ are independent and identically distributed $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} \in \mathbb{R}^p$ and $\boldsymbol{\Sigma}$ is a positive definite $p \times p$ matrix.

Background

Suppose that observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^P$ are independent and identically distributed $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} \in \mathbb{R}^p$ and $\boldsymbol{\Sigma}$ is a positive definite $p \times p$ matrix.

The zeros in the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ correspond to pairs of features that are conditionally independent - that is, pairs of variables that are independent of each other, given all the other variables in the data set.

Background

Suppose that observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^P$ are independent and identically distributed $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} \in \mathbb{R}^p$ and $\boldsymbol{\Sigma}$ is a positive definite $p \times p$ matrix.

The zeros in the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ correspond to pairs of features that are conditionally independent - that is, pairs of variables that are independent of each other, given all the other variables in the data set.

In a Gaussian graphical model, the conditional dependence relationships are represented by a graph in which nodes represent the features and edges connect conditionally dependent pairs of features.

Background

A natural way to estimate the precision (or concentration) matrix $\boldsymbol{\Sigma}^{-1}$ is via maximum likelihood. Letting $\boldsymbol{S}$ denote the empirical covariance matrix of X, the Gaussian log likelihood takes the form (up to a constant)

$$\frac{n}{2}\log\,\det\boldsymbol{\Sigma}^{-1} - \mathsf{trace}\left(\boldsymbol{S}\,\boldsymbol{\Sigma}^{-1}\right)$$

Background

A natural way to estimate the precision (or concentration) matrix $\boldsymbol{\Sigma}^{-1}$ is via maximum likelihood. Letting $\boldsymbol{S}$ denote the empirical covariance matrix of X, the Gaussian log likelihood takes the form (up to a constant)

$$\frac{n}{2}\log\,\det\boldsymbol{\Sigma}^{-1} - \mathsf{trace}\left(\boldsymbol{S}\,\boldsymbol{\Sigma}^{-1}\right)$$

Maximizing this function with respect to $\boldsymbol{\Sigma}^{-1}$ yields the maximum likelihood estimate S^{-1}.

Background

A natural way to estimate the precision (or concentration) matrix $\boldsymbol{\Sigma}^{-1}$ is via maximum likelihood. Letting $\boldsymbol{S}$ denote the empirical covariance matrix of X, the Gaussian log likelihood takes the form (up to a constant)

$$\frac{n}{2} \log \det \boldsymbol{\Sigma}^{-1} - \mathsf{trace}\left(\boldsymbol{S}\,\boldsymbol{\Sigma}^{-1}\right)$$

Maximizing this function with respect to $\boldsymbol{\Sigma}^{-1}$ yields the maximum likelihood estimate S^{-1}.

Problems with using the MLE of $\boldsymbol{\Sigma}^{-1}$:

1. In the high dimensional setting where $p > n$ the matrix $\boldsymbol{S}$ is singular and cannot be inverted to yield an estimate of $\boldsymbol{\Sigma}^{-1}$.

2. Even when $\boldsymbol{S}^{-1}$ exists, this estimator will typically not be sparse.

Background

One method for obtaining sparse estimates of $\boldsymbol{\Sigma}^{-1}$ is to instead solve the *graphical lasso* problem

$$\max_{\boldsymbol{\Theta}} \log\, \det \boldsymbol{\Theta} - \mathsf{trace}\,(\boldsymbol{S}\,\boldsymbol{\Theta}) - \lambda ||\boldsymbol{\Theta}||_1,$$

where λ is a nonnegative tuning parameter.

Background

One method for obtaining sparse estimates of $\mathbf{\Sigma}^{-1}$ is to instead solve the *graphical lasso* problem

$$\max_{\mathbf{\Theta}} \log\, \det \mathbf{\Theta} - \mathsf{trace}\,(\boldsymbol{S}\,\mathbf{\Theta}) - \lambda ||\mathbf{\Theta}||_1,$$

where λ is a nonnegative tuning parameter.

Advantages of using the graphical lasso estimator:

1. The l_1 penalty term yields sparse estimates of $\mathbf{\Sigma^{-1}}$ when λ is "large".

2. This problem can be solved even if $p \gg n$.

Estimating Models for Heterogeneous Data

Problem: The standard formulation for estimating a Gaussian graphical model assumes that each observation is drawn from the same distribution. However, in many datasets the observations may correspond to several distinct classes.

Estimating Models for Heterogeneous Data

Problem: The standard formulation for estimating a Gaussian graphical model assumes that each observation is drawn from the same distribution. However, in many datasets the observations may correspond to several distinct classes.

Estimating a single graphical model would mask the underlying heterogeneity, while estimating separate models for each category ignores the common structure.

Estimating Models for Heterogeneous Data

Problem: The standard formulation for estimating a Gaussian graphical model assumes that each observation is drawn from the same distribution. However, in many datasets the observations may correspond to several distinct classes.

Estimating a single graphical model would mask the underlying heterogeneity, while estimating separate models for each category ignores the common structure.

We want to **jointly** estimate several graphical models corresponding to different categories. In this presentation, we will describe three algorithms that enable us to do this.

Motivation - Genetics

Graphical models are of particular interest in the analysis of gene expression data since they can provide a useful tool for visualizing the relationships among genes and generating biological hypotheses.

Motivation - Genetics

Graphical models are of particular interest in the analysis of gene expression data since they can provide a useful tool for visualizing the relationships among genes and generating biological hypotheses.

The datasets may correspond to several distinct classes. For example, suppose a cancer researcher collects gene expression measurements for a set of cancer tissue samples and a set of normal tissue samples. One might want to estimate a graphical model for the cancer samples and a graphical model for the normal samples.

Motivation - Genetics

Graphical models are of particular interest in the analysis of gene expression data since they can provide a useful tool for visualizing the relationships among genes and generating biological hypotheses.

The datasets may correspond to several distinct classes. For example, suppose a cancer researcher collects gene expression measurements for a set of cancer tissue samples and a set of normal tissue samples. One might want to estimate a graphical model for the cancer samples and a graphical model for the normal samples.

One might expect the two graphical models to be similar to each other since both are based on the same type of tissue, but also have important differences resulting from the fact that gene networks are often dysregulated in cancer.

Problem Formulation

Suppose we have a heterogeneous data set with p variables and K categories.

Problem Formulation

Suppose we have a heterogeneous data set with p variables and K categories.

The k-th category contains n_k observations $(\boldsymbol{x}_1^{(k)}, ..., \boldsymbol{x}_{n_k}^{(k)})$, where each $\boldsymbol{x}_i^{(k)} = (x_{i,1}^{(k)}, ..., x_{i,p}^{(k)})$ is a p-dimensional row vector.

Problem Formulation

Suppose we have a heterogeneous data set with p variables and K categories.

The k-th category contains n_k observations $(\boldsymbol{x}_1^{(k)}, ..., \boldsymbol{x}_{n_k}^{(k)})$, where each $\boldsymbol{x}_i^{(k)} = (x_{i,1}^{(k)}, ..., x_{i,p}^{(k)})$ is a p-dimensional row vector.

Without loss of generality, we assume the observations in the same category are centered along each variable, i.e., $\sum_{i=1}^{n_k} x_{i,j}^{(k)} = 0$ for all $1 \leq j \leq p$ and $1 \leq k \leq K$.

Problem Formulation

We further assume that $\boldsymbol{x}_1^{(k)}, ..., \boldsymbol{x}_{n_k}^{(k)}$ are independent and identically distributed, sampled from a p-variate Gaussian distribution with mean zero (without loss of generality since we center the data) and covariance matrix $\Sigma^{(k)}$.

Let $\Omega^{(k)} = (\Sigma^{(k)})^{-1} = (\omega_{j,j'}^{(k)})_{pxp}$.

Problem Formulation

We further assume that $\boldsymbol{x}_1^{(k)}, ..., \boldsymbol{x}_{n_k}^{(k)}$ are independent and identically distributed, sampled from a p-variate Gaussian distribution with mean zero (without loss of generality since we center the data) and covariance matrix $\Sigma^{(k)}$.

Let $\Omega^{(k)} = (\Sigma^{(k)})^{-1} = (\omega_{j,j'}^{(k)})_{pxp}$.

Then the likelihood function for k-th category will look like this:

$$l(\Omega^{(k)}) = -\frac{n_k}{2}\log(2\pi) + \frac{n_k}{2}[\log\{\det(\Omega^{(k)})\} - \mathsf{trace}(\hat{\Sigma}^{(k)}\Omega^{(k)})],$$

where $\hat{\Sigma}^{(k)}$ is the sample covariance matrix for the k-th category.

Problem Formulation

The most direct way to deal with such heterogeneous data is to estimate K individual graphical models. We can compute a separate l_1-regularized estimator for each category k, $1 \leq k \leq K$, by solving

$$\min_{\Omega^{(k)}} \text{trace}(\hat{\Sigma}^{(k)}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\} + \lambda_k \sum_{j \neq j'} |\omega_{j,j'}^{(k)}|,$$

where the minimum is taken over symmetric positive definite matrices.

Problem Formulation

The most direct way to deal with such heterogeneous data is to estimate K individual graphical models. We can compute a separate l_1-regularized estimator for each category k, $1 \leq k \leq K$, by solving

$$\min_{\Omega^{(k)}} \text{trace}(\hat{\Sigma}^{(k)}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\} + \lambda_k \sum_{j \neq j'} |\omega_{j,j'}^{(k)}|,$$

where the minimum is taken over symmetric positive definite matrices.

This problem can be efficiently solved by existing algorithms such as $glasso$ of Freidman et al. (2008). However, we would prefer to jointly estimate several graphical models instead of estimating each one separately.

Joint Estimation

To improve estimation in cases where graphical models for different categories may share some common structure, J. Guo, E. Levina, G. Michailidis, and J. Zhu propose the following *joint estimation method* in their paper “Joint Estimation of Multiple Graphical Models” (2009).

Joint Estimation

To improve estimation in cases where graphical models for different categories may share some common structure, J. Guo, E. Levina, G. Michailidis, and J. Zhu propose the following *joint estimation method* in their paper "Joint Estimation of Multiple Graphical Models" (2009).

First, they reparameterize each $\omega_{j,j'}^{(k)}$ as

$$\omega_{j,j'}^{(k)} = \theta_{j,j'}\gamma_{j,j'}^{(k)}, 1 \le j \ne j' \le p,\ 1 \le k \le K$$

.

Joint Estimation

To improve estimation in cases where graphical models for different categories may share some common structure, J. Guo, E. Levina, G. Michailidis, and J. Zhu propose the following *joint estimation method* in their paper "Joint Estimation of Multiple Graphical Models" (2009).

First, they reparameterize each $\omega_{j,j'}^{(k)}$ as

$$\omega_{j,j'}^{(k)} = \theta_{j,j'}\gamma_{j,j'}^{(k)}, 1 \leq j \neq j' \leq p,\ 1 \leq k \leq K$$

.

For identifiability purposes, we restrict $\theta_{j,j'} > 0,\ 1 \leq j \neq j' \leq p$. To preserve symmetry, they also require $\theta_{j,j'} = \theta_{j',j}$ and $\gamma_{j,j'}^{(k)} = \gamma_{j',j}^{(k)}$, $1 \leq j \neq j' \leq p$ and $1 \leq k \leq K$.

Joint Estimation

To improve estimation in cases where graphical models for different categories may share some common structure, J. Guo, E. Levina, G. Michailidis, and J. Zhu propose the following *joint estimation method* in their paper "Joint Estimation of Multiple Graphical Models" (2009).

First, they reparameterize each $\omega_{j,j'}^{(k)}$ as

$$\omega_{j,j'}^{(k)} = \theta_{j,j'}\gamma_{j,j'}^{(k)}, 1 \leq j \neq j' \leq p,\ 1 \leq k \leq K$$

.

For identifiability purposes, we restrict $\theta_{j,j'} > 0,\ 1 \leq j \neq j' \leq p$. To preserve symmetry, they also require $\theta_{j,j'} = \theta_{j',j}$ and $\gamma_{j,j'}^{(k)} = \gamma_{j',j}^{(k)}$, $1 \leq j \neq j' \leq p$ and $1 \leq k \leq K$.

This decomposition treats $\{\omega_{j,j'}^{(1)}, ..., \omega_{j,j'}^{(K)}\}$ as a group, with the common factor $\theta_{j,j'}$ controlling the presence of the link between nodes j and j' in any of the categories, and $\gamma_{j,j'}^{(k)}$ reflects the differences between categories.

Joint Estimation

Let $\Theta = (\theta_{j,j'})_{pxp}$ and $\Gamma^{(k)} = (\gamma_{j,j'}^{(k)})_{pxp}$. To estimate this model, we propose the following penalized criterion:

$$\min_{\Theta,\{\Gamma^{(k)}\}_{k=1}^{K}} \sum_{k=1}^{K} [\text{trace}(\hat{\Sigma}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\}] + \eta_1 \sum_{j\neq j'} \theta_{j,j'} + \eta_2 \sum_{j\neq j'} \sum_{k=1}^{K} |\gamma_{j,j'}^{(k)}|, \tag{1}$$

subject to

$$\begin{aligned} &\omega_{j,j'}^{(k)} = \theta_{j,j'}\gamma_{j,j'}^{(k)},\ \theta_{j,j'} > 0,\ 1 \leq j, j' \leq p \\ &\theta_{j,j'} = \theta_{j',j},\ \gamma_{j,j'}^{(k)} = \gamma_{j',j}^{(k)}, 1 \leq j \neq j' \leq p;\ 1 \leq k \leq K \\ &\theta_{j,j'} = 1, \gamma_{j,j}^{(k)} = \omega_{j,j'}^{(k)}, 1 \leq j \leq p;\ 1 \leq k \leq K, \end{aligned}$$

where η_1 and η_2 are two tuning parameters.

Joint Estimation

- The first parameter, η_1 , controls the sparsity of the common factors $\theta_{j,j'}$ and it can effectively remove the common zero elements across $\Omega^{(1)}, ..., \Omega^{(K)}$; i.e., if $\theta_{j,j'}$ is shrunk to zero, there will be no link between nodes j and j' in any of the K graphs.

Joint Estimation

- The first parameter, η_1 , controls the sparsity of the common factors $\theta_{j,j'}$ and it can effectively remove the common zero elements across $\Omega^{(1)}, ..., \Omega^{(K)}$; i.e., if $\theta_{j,j'}$ is shrunk to zero, there will be no link between nodes j and j' in any of the K graphs.
- If $\theta_{j,j'}$ is not zero, some of the $\gamma_{j,j'}^{(k)}$s (and hence some of the $\omega_{j,j'}^{(k)}$ s) can still be set to zero by the second penalty controlled by η_2. This allows graphs belonging to different categories to have different structures.

Joint Estimation

To simplify the model, the two tuning parameters η_1 and η_2 in (1) can be replaced by a single tuning parameter, by letting $\eta = \eta_1\eta_2$. It turns out that solving (1) is equivalent to solving

$$\min_{\Theta,\{\Gamma^{(k)}\}_{k=1}^{K}} \sum_{k=1}^{K}[\text{trace}(\hat{\Sigma}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\}] + \sum_{j\neq j'}\theta_{j,j'} + \eta\sum_{j\neq j'}\sum_{k=1}^{K}|\gamma_{j,j'}^{(k)}|, \tag{2}$$

subject to
$$\begin{aligned} &\omega_{j,j'}^{(k)} = \theta_{j,j'}\gamma_{j,j'}^{(k)},\ \theta_{j,j'} > 0,\ 1 \le j, j' \le p \\ &\theta_{j,j'} = \theta_{j',j},\ \gamma_{j,j'}^{(k)} = \gamma_{j',j}^{(k)}, 1 \le j \neq j' \le p;\ 1 \le k \le K \\ &\theta_{j,j'} = 1, \gamma_{j,j}^{(k)} = \omega_{j,j'}^{(k)}, 1 \le j \le p;\ 1 \le k \le K, \end{aligned}$$

Joint Estimation

First, Guo et al. reformulate the problem (2) in a more convenient form for computational purposes.

Joint Estimation

First, Guo et al. reformulate the problem (2) in a more convenient form for computational purposes.

Let $\{\hat{\Omega}^{(k)}\}_{k=1}^{K}$ be a local minimizer of

$$\min_{\{\Omega^{(k)}\}_{k=1}^{K}} \sum_{k=1}^{K} [\text{trace}(\hat{\Sigma}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\}] + \lambda \sum_{j \neq j'} \sqrt{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|}, \quad (3)$$

where $\lambda = 2\sqrt{\eta}$. Then, there exists a local minimizer of (2), $(\hat{\Theta}, \{\hat{\Gamma}\}_{k=1}^{K})$, such that $\hat{\Omega}^{(k)} = \hat{\Theta}^{(k)}\hat{\Gamma}^{(k)}$, for all $1 \leq k \leq K$. On the other hand, if $(\hat{\Theta}, \{\hat{\Gamma}^{(k)}\}_{k=1}^{K})$ is a local minimizer of (2), then there also exists a local minimizer of (3), $\{\hat{\Omega}^{(k)}\}_{k=1}^{K}$, such that $\hat{\Omega}^{(k)} = \hat{\Theta}^{(k)}\hat{\Gamma}^{(k)}$ for all $1 \leq k \leq K$.

Joint Estimation

One can see that because of the penalty term in (3), the optimization criterion is *not* convex.

Joint Estimation

One can see that because of the penalty term in (3), the optimization criterion is *not* convex.

In order to reach convexity, an iterative optimization approach based on local linear approximation (LLA) is used (Zou and Li, 2008). If $p_\lambda(|\omega|)$ denotes the penalty term then LLA does the following:

$$p_\lambda(|\omega|) \approx p_\lambda(|\omega^0|) + p'_\lambda(|\omega^0|)(|\omega| - |\omega^0|), \text{ for } \omega \approx \omega^0.$$

Joint Estimation

One can see that because of the penalty term in (3), the optimization criterion is *not* convex.

In order to reach convexity, an iterative optimization approach based on local linear approximation (LLA) is used (Zou and Li, 2008). If $p_\lambda(|\omega|)$ denotes the penalty term then LLA does the following:

$$p_\lambda(|\omega|) \approx p_\lambda(|\omega^0|) + p'_\lambda(|\omega^0|)(|\omega| - |\omega^0|), \text{ for } \omega \approx \omega^0.$$

Specifically, letting $(\omega_{j,j'}^{(k)})^{(t)}$ denote the estimates from the previous iteration t, we approximate

$$\sqrt{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|} \sim \frac{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|}{\sqrt{\sum_{k=1}^{K} |(\omega_{j,j'}^{(k)})^{(t)}|}}.$$

Joint Estimation

One can see that because of the penalty term in (3), the optimization criterion is *not* convex.

In order to reach convexity, an iterative optimization approach based on local linear approximation (LLA) is used (Zou and Li, 2008). If $p_\lambda(|\omega|)$ denotes the penalty term then LLA does the following:

$$p_\lambda(|\omega|) \approx p_\lambda(|\omega^0|) + p'_\lambda(|\omega^0|)(|\omega| - |\omega^0|), \text{ for } \omega \approx \omega^0.$$

Specifically, letting $(\omega_{j,j'}^{(k)})^{(t)}$ denote the estimates from the previous iteration t, we approximate

$$\sqrt{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|} \sim \frac{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|}{\sqrt{\sum_{k=1}^{K} |(\omega_{j,j'}^{(k)})^{(t)}|}}.$$

LLA estimate helps alleviate the computation burden and overcome the potential local minima problem in minimizing the nonconvex function.

Joint Estimation Algorithm

Thus, at the $(t+1)$-th iteration of our joint estimation algorithm, problem (3) may be decomposed into K individual optimization problems:

$$(\Omega^{(k)})^{(t+1)} = \arg\min_{\Omega^{(k)}} \mathsf{trace}(\hat{\Sigma}^{(k)}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\} + \lambda \sum_{j \neq j'} \tau_{j,j'}^{(k)} |\omega_{j,j'}^{(k)}|, \tag{4}$$

where $\tau_{j,j'}^{(k)} = (\sum_{k=1}^{K} |(\omega_{j,j'}^{(k)})^{(t)}|)^{-1/2}$.

Joint Estimation Algorithm

Thus, at the $(t+1)$-th iteration of our joint estimation algorithm, problem (3) may be decomposed into K individual optimization problems:

$$(\Omega^{(k)})^{(t+1)} = \arg\min_{\Omega^{(k)}} \mathsf{trace}(\hat{\Sigma}^{(k)}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\} + \lambda \sum_{j \neq j'} \tau^{(k)}_{j,j'} |\omega^{(k)}_{j,j'}|, \tag{4}$$

where $\tau^{(k)}_{j,j'} = (\sum_{k=1}^{K} |(\omega^{(k)}_{j,j'})^{(t)}|)^{-1/2}$.

Note that criterion (4) is exactly the sparse precision matrix estimation problem with weighted l_1 penalty; the solution can be efficiently computed using $glasso$.

Convexity of the Criterion

- trace$(\hat{\Sigma}^{(k)}\Omega^{(k)})$ is an affine function of elements of $\Omega^{(k)}$, therefore is convex.

Convexity of the Criterion

- trace$(\hat{\Sigma}^{(k)}\Omega^{(k)})$ is an affine function of elements of $\Omega^{(k)}$, therefore is convex.
- $\log\{\det(\Omega^{(k)})\}$ is convex by the restriction to the line proof from class.

Convexity of the Criterion

- trace$(\hat{\Sigma}^{(k)}\Omega^{(k)})$ is an affine function of elements of $\Omega^{(k)}$, therefore is convex.
- $\log\{\det(\Omega^{(k)})\}$ is convex by the restriction to the line proof from class.
- $\lambda \sum_{j \neq j'} \tau_{j,j'}^{(k)} |\omega_{j,j'}^{(k)}|$ is convex as an affine function of elements of $\Omega^{(k)}$.

The sum of convex functions is convex, therefore the minimization criterion is a convex function of $\Omega^{(k)}$.

Joint Estimation Algorithm

In summary, the proposed algorithm for solving

$$\min_{\{\Omega^{(k)}\}_{k=1}^{K}} \sum_{k=1}^{K}[\mathsf{trace}(\hat{\Sigma}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\}] + \lambda\sum_{j\neq j'}\sqrt{\sum_{k=1}^{K}|\omega_{j,j'}^{(k)}|},$$

is:

Joint Estimation Algorithm

In summary, the proposed algorithm for solving

$$\min_{\{\Omega^{(k)}\}_{k=1}^{K}} \sum_{k=1}^{K} [\text{trace}(\hat{\Sigma}\Omega^{(k)}) - \log\{\det(\Omega^{(k)})\}] + \lambda \sum_{j \neq j'} \sqrt{\sum_{k=1}^{K} |\omega_{j,j'}^{(k)}|},$$

is:

(a) Initialize $\hat{\Omega}^{(k)} = (\hat{\Sigma}^{(k)} + \nu I_p)^{-1}$ for all $1 \leq k \leq K$, where I_p is the identity matrix and the constant ν is chosen to guarantee $\hat{\Sigma}^{(k)} + \nu I_p$ is positive definite;

(b) Using $glasso$, update $\hat{\Omega}^{(k)}$ by (4) for all $1 \leq k \leq K$.

(c) Repeat the previous step until convergence is achieved.

Model Selection.

The tuning parameter λ in (4) controls the sparsity of the resulting estimator.

Model Selection.

The tuning parameter λ in (4) controls the sparsity of the resulting estimator.

Guo et al. select the optimal tuning parameter by minimizing a Bayesian information criterion (BIC), which balances the goodness of fit and the model complexity. Specifically, we define BIC for the proposed joint estimation method as

$$BIC(\lambda) = \sum_{k=1}^{K} [\text{trace}(\hat{\Sigma}^{(k)}\hat{\Omega}_{\lambda}^{(k)}) - \log\{\det(\hat{\Omega}_{\lambda}^{(k)}\} + \log(n_k)df_k]$$

where $\hat{\Omega}_{\lambda}^{(1)}, ..., \hat{\Omega}_{\lambda}^{(K)}$ are the estimates from (4) with tuning parameter λ and the degrees of freedom are $df_k = \#\{(j,j') : j < j', \hat{\omega}_{j,j'}^{(k)} \neq 0\}$.

Simulation Study

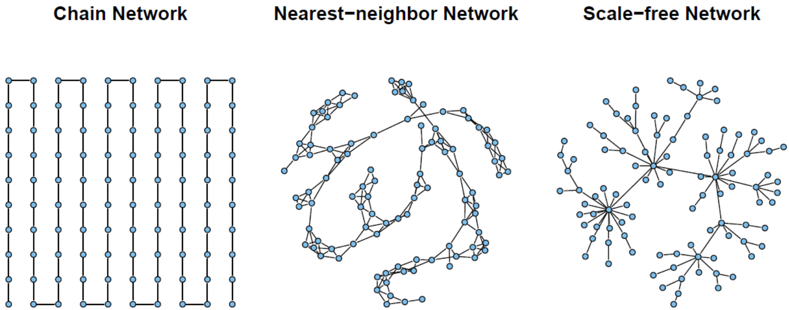
To assess the performance of the joint estimation algorithm in Guo et al., three types of simulated networks are simulated: a chain network, a 5-nearest neighbors network, and a scale-free (power law) network. In all cases, $p = 100$ features and $K = 3$. For $k = 1, ..., K$, $n_k = 100$ iid observations from a multiariate normal distribution $N(\mathbf{0}, (\mathbf{\Omega}^{(k)})^{-1})$, where $\mathbf{\Omega}^{(k)}$ is the precision matrix for the k-th category.

Simulation Study

To assess the performance of the joint estimation algorithm in Guo et al., three types of simulated networks are simulated: a chain network, a 5-nearest neighbors network, and a scale-free (power law) network. In all cases, $p = 100$ features and $K = 3$. For $k = 1, ..., K$, $n_k = 100$ iid observations from a multiariate normal distribution $N(\mathbf{0}, (\mathbf{\Omega}^{(k)})^{-1})$, where $\mathbf{\Omega}^{(k)}$ is the precision matrix for the k-th category.

A constant ρ is used to add heterogeneity to the common structure by creating additional indiivdual links (i.e. ρ is the ratio of the number of individual links to the number of common links).

Networks



Chain Network

Nearest−neighbor Network

Scale−free Network

Figure 1: The common links (present in all categories) in the three simulated networks.

Model Diagnostics

We compare the *joint estimation method* to the *separate estimation method*. To assess performance:

Model Diagnostics

We compare the *joint estimation method* to the *separate estimation method*. To assess performance:

- Plot ROC curves which plot the average proportion of correctly detected links against the average false positive range of values over a range of values of λ.

Model Diagnostics

We compare the *joint estimation method* to the *separate estimation method*. To assess performance:

- Plot ROC curves which plot the average proportion of correctly detected links against the average false positive range of values over a range of values of λ.
- Compute and compare the following metrics: average entropy loss (EL), average Frobenius loss (FL), average false positive (FP) and average false negative (FN) rates, and the average rate of mis-identified common zeros among the categories (CZ).

Simulation Results

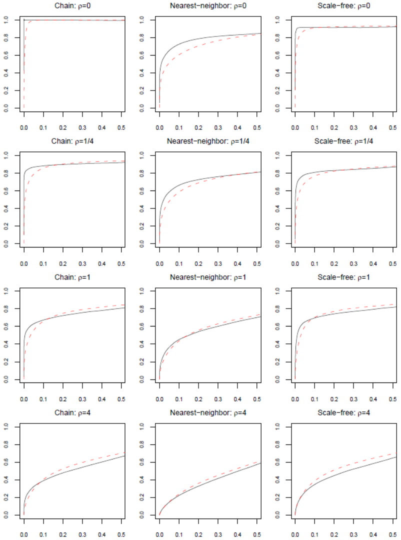


Figure 2: ROC curves. The solid line corresponds to the joint estimation method, and the dashed line to the separate estimation method.

Simulation Results

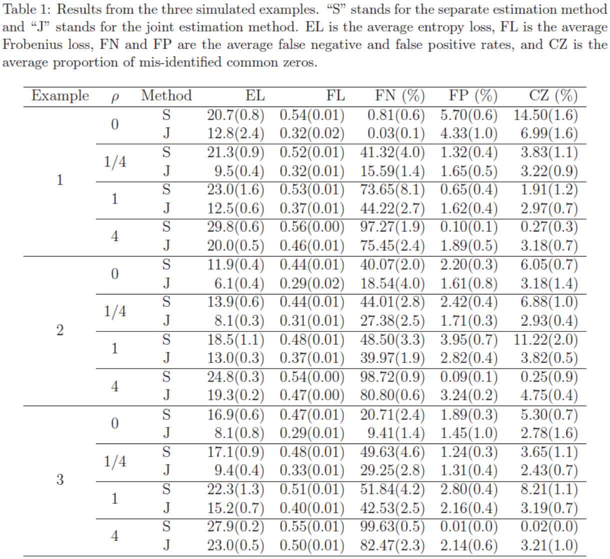


Table 1: Results from the three simulated examples. "S" stands for the separate estimation method and "J" stands for the joint estimation method. EL is the average entropy loss, FL is the average Frobenius loss, FN and FP are the average false negative and false positive rates, and CZ is the average proportion of mis-identified common zeros.

Example	ρ	Method	EL	FL	FN (%)	FP (%)	CZ (%)
1	0	S	20.7(0.8)	0.54(0.01)	0.81(0.6)	5.70(0.6)	14.50(1.6)
		J	12.8(2.4)	0.32(0.02)	0.03(0.1)	4.33(1.0)	6.99(1.6)
	1/4	S	21.3(0.9)	0.52(0.01)	41.32(4.0)	1.32(0.4)	3.83(1.1)
		J	9.5(0.4)	0.32(0.01)	15.59(1.4)	1.65(0.5)	3.22(0.9)
	1	S	23.0(1.6)	0.53(0.01)	73.65(8.1)	0.65(0.4)	1.91(1.2)
		J	12.5(0.6)	0.37(0.01)	44.22(2.7)	1.62(0.4)	2.97(0.7)
	4	S	29.8(0.6)	0.56(0.00)	97.27(1.9)	0.10(0.1)	0.27(0.3)
		J	20.0(0.5)	0.46(0.01)	75.45(2.4)	1.89(0.5)	3.18(0.7)
2	0	S	11.9(0.4)	0.44(0.01)	40.07(2.0)	2.20(0.3)	6.05(0.7)
		J	6.1(0.4)	0.29(0.02)	18.54(4.0)	1.61(0.8)	3.18(1.4)
	1/4	S	13.9(0.6)	0.44(0.01)	44.01(2.8)	2.42(0.4)	6.88(1.0)
		J	8.1(0.3)	0.31(0.01)	27.38(2.5)	1.71(0.3)	2.93(0.4)
	1	S	18.5(1.1)	0.48(0.01)	48.50(3.3)	3.95(0.7)	11.22(2.0)
		J	13.0(0.3)	0.37(0.01)	39.97(1.9)	2.82(0.4)	3.82(0.5)
	4	S	24.8(0.3)	0.54(0.00)	98.72(0.9)	0.09(0.1)	0.25(0.9)
		J	19.3(0.2)	0.47(0.00)	80.80(0.6)	3.24(0.2)	4.75(0.4)
3	0	S	16.9(0.6)	0.47(0.01)	20.71(2.4)	1.89(0.3)	5.30(0.7)
		J	8.1(0.8)	0.29(0.01)	9.41(1.4)	1.45(1.0)	2.78(1.6)
	1/4	S	17.1(0.9)	0.48(0.01)	49.63(4.6)	1.24(0.3)	3.65(1.1)
		J	9.4(0.4)	0.33(0.01)	29.25(2.8)	1.31(0.4)	2.43(0.7)
	1	S	22.3(1.3)	0.51(0.01)	51.84(4.2)	2.80(0.4)	8.21(1.1)
		J	15.2(0.7)	0.40(0.01)	42.53(2.5)	2.16(0.4)	3.19(0.7)
	4	S	27.9(0.2)	0.55(0.01)	99.63(0.5)	0.01(0.0)	0.02(0.0)
		J	23.0(0.5)	0.50(0.01)	82.47(2.3)	2.14(0.6)	3.21(1.0)

Simulation Results

As the estimated ROC curves in Figure 2 show, the joint estimation method dominates the separate estimation method when the proportion of individual links is low. When the proportion of indvidual links is *high*, the methods perform similarly.

Simulation Results

As the estimated ROC curves in Figure 2 show, the joint estimation method dominates the separate estimation method when the proportion of individual links is low. When the proportion of indvidual links is *high*, the methods perform similarly.

As Table 1 shows, the joint estimation method produces lower entropy and Frobenius losses, as well as lower false negative rates.

Simulation Results

As the estimated ROC curves in Figure 2 show, the joint estimation method dominates the separate estimation method when the proportion of individual links is low. When the proportion of indvidual links is *high*, the methods perform similarly.

As Table 1 shows, the joint estimation method produces lower entropy and Frobenius losses, as well as lower false negative rates.

When the proportion of common links is high enough, it produces lower false positive rates and performs better at identifying common zeros than the separate estimation method.

Simulation Results

As the estimated ROC curves in Figure 2 show, the joint estimation method dominates the separate estimation method when the proportion of individual links is low. When the proportion of indvidual links is *high*, the methods perform similarly.

As Table 1 shows, the joint estimation method produces lower entropy and Frobenius losses, as well as lower false negative rates.

When the proportion of common links is high enough, it produces lower false positive rates and performs better at identifying common zeros than the separate estimation method.

Conclusion: In the presence of a common structure, the joint estimation method outperforms the separate estimation method. In the absence of a common structure, their performance is fairly similar.

Data Example

Webpages from websites at computer science departments in four universities (Cornell, Texas, Washington, and Wisconsin) are manually classified into seven categories: student, faculty, staff, department, course, project, and other. The joint estimation method is applied to $n = 1396$ documents in the four largest categories and $p = 100$ terms, and links between the terms are explored. The resulting common structure is shown in Figure 3.

Data Example

Webpages from websites at computer science departments in four universities (Cornell, Texas, Washington, and Wisconsin) are manually classified into seven categories: student, faculty, staff, department, course, project, and other. The joint estimation method is applied to $n = 1396$ documents in the four largest categories and $p = 100$ terms, and links between the terms are explored. The resulting common structure is shown in Figure 3.

The model also allows us to explore the heterogeneity between different categories. For example, the graphs for the "student" and "faculty" categories have some links that appear in only one or the other (e.g. the terms "teach" and "assist" are only linked in the "student" category, while "assist-professor" and "select-public" are only linked in the "faculty" category.

Data Example

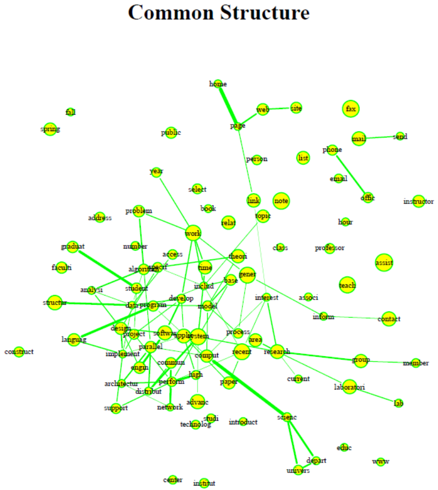


Figure 3: The common structure in the estimated graph. The nodes represent the 100 terms with highest log-entropy weights. The areas of the yellow circles are proportional to the degrees of the associated nodes in the common structure. The width of each edge is proportional to the magnitude of the associated partial correlation.

Data Example

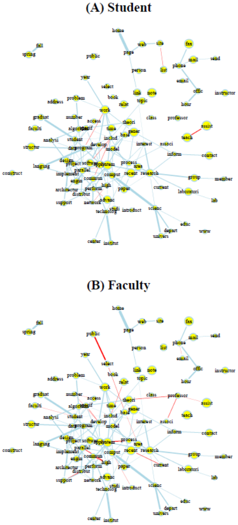


Figure 5: "Student" and "Faculty" graphs. The light blue lines are the links appearing in both categories, and the red lines are the links only appearing in one category.

The Joint Graphical Lasso

In their paper "The joint graphical lasso for inverse covariance estimation across multiple classes," P. Danaher, P. Wang, & D. Witten propose the *joint graphical lasso* as a technique for jointly estimating multiple graphical models corresponding to distinct but related conditions, such as cancer and normal tissue.

The Joint Graphical Lasso

In their paper "The joint graphical lasso for inverse covariance estimation across multiple classes," P. Danaher, P. Wang, & D. Witten propose the *joint graphical lasso* as a technique for jointly estimating multiple graphical models corresponding to distinct but related conditions, such as cancer and normal tissue.

Suppose we have data from $K \geq 2$ distinct classes. Instead of estimating $\mathbf{\Sigma}_1^{-1}, \mathbf{\Sigma}_2^{-1}, \ldots, \mathbf{\Sigma}_K^{-1}$ separately, the authors propose estimating estimating these values jointly by maximizing the following penalized log-likelihood function

$$\max_{\{\mathbf{\Theta}\}} \sum_{k=1}^{K} n_k \left[\log\,\det \mathbf{\Theta}^{(k)} - \mathsf{trace}\left(\boldsymbol{S}^{(k)}\mathbf{\Theta}^{(k)}\right)\right] - P(\{\mathbf{\Theta}\}),$$

subject to the constraint that $\mathbf{\Theta}^{(1)}, \mathbf{\Theta}^{(2)}, \ldots, \mathbf{\Theta}^{(K)}$ are positive definite, where $\{\mathbf{\Theta}\} = \{\mathbf{\Theta}^{(1)}, \mathbf{\Theta}^{(2)}, \ldots, \mathbf{\Theta}^{(K)}\}$.

The Joint Graphical Lasso

In their paper "The joint graphical lasso for inverse covariance estimation across multiple classes," P. Danaher, P. Wang, & D. Witten propose the *joint graphical lasso* as a technique for jointly estimating multiple graphical models corresponding to distinct but related conditions, such as cancer and normal tissue.

Suppose we have data from $K \geq 2$ distinct classes. Instead of estimating $\mathbf{\Sigma}_1^{-1}, \mathbf{\Sigma}_2^{-1}, \dots, \mathbf{\Sigma}_K^{-1}$ separately, the authors propose estimating estimating these values jointly by maximizing the following penalized log-likelihood function

$$\max_{\{\mathbf{\Theta}\}} \sum_{k=1}^{K} n_k \left[\log\, \det \mathbf{\Theta}^{(k)} - \mathsf{trace}\left(\boldsymbol{S}^{(k)} \mathbf{\Theta}^{(k)} \right) \right] - P(\{\mathbf{\Theta}\}),$$

subject to the constraint that $\mathbf{\Theta}^{(1)}, \mathbf{\Theta}^{(2)}, \dots, \mathbf{\Theta}^{(K)}$ are positive definite, where $\{\mathbf{\Theta}\} = \{\mathbf{\Theta}^{(1)}, \mathbf{\Theta}^{(2)}, \dots, \mathbf{\Theta}^{(K)}\}$.

$P(\{\mathbf{\Theta}\})$ denotes a convex penalty function, so the objective function is strictly concave.

Joint Graphical Lasso

Danaher et al. propose choosing a penalty function P that will encourage $\hat{\mathbf{\Theta}}^{(1)}, \hat{\mathbf{\Theta}}^{(2)}, \ldots, \hat{\mathbf{\Theta}}^{(K)}$ to share certain characteristics, such as the locations or values of the nonzero elements, in addition to providing sparse estimates of the precision matrices.

Joint Graphical Lasso

Danaher et al. propose choosing a penalty function P that will encourage $\hat{\mathbf{\Theta}}^{(1)}, \hat{\mathbf{\Theta}}^{(2)}, \ldots, \hat{\mathbf{\Theta}}^{(K)}$ to share certain characteristics, such as the locations or values of the nonzero elements, in addition to providing sparse estimates of the precision matrices.

Two penalty functions suggested by Danaher et al. are the *fused graphical lasso* and *group graphical lasso* penalties.

Fused Graphical Lasso

The fused graphical lasso (FGL) penalty is given by

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{k<k'} \sum_{i,j} |\theta_{ij}^{(k)} - \theta_{ij}^{(k')}|,$$

where λ_1 and λ_2 are nonnegative tuning parameters.

Fused Graphical Lasso

The fused graphical lasso (FGL) penalty is given by

$$P(\{\boldsymbol{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{k<k'} \sum_{i,j} |\theta_{ij}^{(k)} - \theta_{ij}^{(k')}|,$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. Like the graphical lasso, FGL results in sparse estimates $\hat{\boldsymbol{\Theta}}^{(1)}, \hat{\boldsymbol{\Theta}}^{(2)}, \ldots, \hat{\boldsymbol{\Theta}}^{(K)}$ when the tuning parameter λ_1 is large.

Fused Graphical Lasso

The fused graphical lasso (FGL) penalty is given by

$$P(\{\boldsymbol{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{k<k'} \sum_{i,j} |\theta_{ij}^{(k)} - \theta_{ij}^{(k')}|,$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. Like the graphical lasso, FGL results in sparse estimates $\hat{\boldsymbol{\Theta}}^{(1)}, \hat{\boldsymbol{\Theta}}^{(2)}, \ldots, \hat{\boldsymbol{\Theta}}^{(K)}$ when the tuning parameter λ_1 is large.

2. Many of the elements $\hat{\boldsymbol{\Theta}}^{(1)}, \hat{\boldsymbol{\Theta}}^{(2)}, \ldots, \hat{\boldsymbol{\Theta}}^{(K)}$ will be identical across classes when the tuning parameter λ_2 is large.

Fused Graphical Lasso

The fused graphical lasso (FGL) penalty is given by

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{k<k'} \sum_{i,j} |\theta_{ij}^{(k)} - \theta_{ij}^{(k')}|,$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. Like the graphical lasso, FGL results in sparse estimates $\hat{\mathbf{\Theta}}^{(1)}, \hat{\mathbf{\Theta}}^{(2)}, \ldots, \hat{\mathbf{\Theta}}^{(K)}$ when the tuning parameter λ_1 is large.

2. Many of the elements $\hat{\mathbf{\Theta}}^{(1)}, \hat{\mathbf{\Theta}}^{(2)}, \ldots, \hat{\mathbf{\Theta}}^{(K)}$ will be identical across classes when the tuning parameter λ_2 is large.

3. FGL borrows information aggressively across classes, encouraging not only similar network structure but also similar edge values.

Group Graphical Lasso

The group graphical lasso (GGL) penalty function is given by

$$P(\{\boldsymbol{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{i \neq j} \sqrt{\sum_{k=1}^{K} \theta_{ij}^{(k)^2}},$$

where λ_1 and λ_2 are nonnegative tuning parameters.

Group Graphical Lasso

The group graphical lasso (GGL) penalty function is given by

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{i \neq j} \sqrt{\sum_{k=1}^{K} \theta_{ij}^{(k)^2}},$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. The group lasso penalty encourages a similar pattern of sparsity across all of the precision matrices - there will be a tendency for the zeros in the K estimated precision matrices to occur in the same places.

Group Graphical Lasso

The group graphical lasso (GGL) penalty function is given by

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{i \neq j} \sqrt{\sum_{k=1}^{K} \theta_{ij}^{(k)^2}},$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. The group lasso penalty encourages a similar pattern of sparsity across all of the precision matrices - there will be a tendency for the zeros in the K estimated precision matrices to occur in the same places.
2. When $\lambda_1 = 0$ and $\lambda_2 > 0$, each $\hat{\mathbf{\Theta}}^{(k)}$ will have an identical pattern of non-zero elements. On the other hand, the lasso penalty encourages further sparsity within each $\hat{\mathbf{\Theta}}^{(k)}$.

Group Graphical Lasso

The group graphical lasso (GGL) penalty function is given by

$$P(\{\mathbf{\Theta}\}) = \lambda_1 \sum_{k=1}^{K} \sum_{i \neq j} |\theta_{ij}^{(k)}| + \lambda_2 \sum_{i \neq j} \sqrt{\sum_{k=1}^{K} \theta_{ij}^{(k)^2}},$$

where λ_1 and λ_2 are nonnegative tuning parameters.

1. The group lasso penalty encourages a similar pattern of sparsity across all of the precision matrices - there will be a tendency for the zeros in the K estimated precision matrices to occur in the same places.
2. When $\lambda_1 = 0$ and $\lambda_2 > 0$, each $\hat{\mathbf{\Theta}}^{(k)}$ will have an identical pattern of non-zero elements. On the other hand, the lasso penalty encourages further sparsity within each $\hat{\mathbf{\Theta}}^{(k)}$.
3. GGL encourages a weaker form of similarity across the K precision matrices than FGL in that GGL only encourages a shared pattern of sparsity and not shared edge values.

Algorithm for the Joint Graphical Lasso

Danaher et al. solve the joint graphical lasso problem using an *alternating directions method of multipliers* (ADMM) algorithm.

Algorithm for the Joint Graphical Lasso

Danaher et al. solve the joint graphical lasso problem using an *alternating directions method of multipliers* (ADMM) algorithm.

The original problem can be rewritten as

$$\min_{\{\boldsymbol{\Theta}\},\{\boldsymbol{Z}\}} -\sum_{k=1}^{K} n_k \left[\log\,\det \boldsymbol{\Theta}^{(k)} - \mathsf{trace}\left(\boldsymbol{S}^{(k)}\boldsymbol{\Theta}^{(k)}\right)\right] + P(\{\boldsymbol{Z}\})$$

subject to the positive-definiteness constraint as well as the constraint that $\boldsymbol{Z}^{(k)} = \boldsymbol{\Theta}^{(k)}$, where $\{\boldsymbol{Z}\} = \{\boldsymbol{Z}^{(1)}, \boldsymbol{Z}^{(2)}, \ldots, \boldsymbol{Z}^{(k)}\}$.

Algorithm for the Joint Graphical Lasso

The scaled augmented Lagrangian for this problem is given by

$$\begin{aligned} L_\rho(\{\boldsymbol{\Theta}\}, \{\boldsymbol{Z}\}, \{\boldsymbol{U}\}) = & -\sum_{k=1}^{K} n_k \left[\log\, \det \boldsymbol{\Theta}^{(k)} - \text{trace}\left(\boldsymbol{S}^{(k)} \boldsymbol{\Theta}^{(k)}\right)\right] + P(\{\boldsymbol{Z}\}) \\ & + \frac{\rho}{2} \sum_{k=1}^{K} ||\boldsymbol{\Theta}^{(k)} - \boldsymbol{Z}^{(k)} + \boldsymbol{U}^{(k)}||_F^2, \end{aligned}$$

where $\{\boldsymbol{U}\} = \{\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \ldots, \boldsymbol{U}^{(k)}\}$.

Algorithm for the Joint Graphical Lasso

The scaled augmented Lagrangian for this problem is given by

$$L_\rho(\{\boldsymbol{\Theta}\},\{\boldsymbol{Z}\},\{\boldsymbol{U}\}) = -\sum_{k=1}^{K} n_k \left[\log\,\det \boldsymbol{\Theta}^{(k)} - \mathsf{trace}\left(\boldsymbol{S}^{(k)}\boldsymbol{\Theta}^{(k)}\right)\right] + P(\{\boldsymbol{Z}\}) \\ + \frac{\rho}{2}\sum_{k=1}^{K} ||\boldsymbol{\Theta}^{(k)} - \boldsymbol{Z}^{(k)} + \boldsymbol{U}^{(k)}||_F^2,$$

where $\{\boldsymbol{U}\} = \{\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \ldots, \boldsymbol{U}^{(k)}\}$.

The ADMM corresponding to the above problem results in iterating three simple steps.

(a) $\{\boldsymbol{\Theta}_{(i)}\} \leftarrow \mathsf{arg\,min}_{\{\boldsymbol{\Theta}\}}\{L_\rho(\{\boldsymbol{\Theta}\},\{\boldsymbol{Z}_{(i-1)}\},\{\boldsymbol{U}_{(i-1)}\})\}$

(b) $\{\boldsymbol{Z}_{(i)}\} \leftarrow \mathsf{arg\,min}_{\{\boldsymbol{Z}\}}\{L_\rho(\{\boldsymbol{\Theta}_{(i)}\},\{\boldsymbol{Z}\},\{\boldsymbol{U}_{(i-1)}\})\}$

(c) $\{\boldsymbol{U}_{(i)}\} \leftarrow \{\boldsymbol{U}_{(i-1)}\} + (\{\boldsymbol{\Theta}_{(i)}\} - \{\boldsymbol{Z}_{(i)}\})$.

Algorithm for the Joint Graphical Lasso

The scaled augmented Lagrangian for this problem is given by

$$L_\rho(\{\boldsymbol{\Theta}\},\{\boldsymbol{Z}\},\{\boldsymbol{U}\}) = -\sum_{k=1}^{K} n_k \left[\log\,\det \boldsymbol{\Theta}^{(k)} - \mathsf{trace}\left(\boldsymbol{S}^{(k)}\boldsymbol{\Theta}^{(k)}\right)\right] + P(\{\boldsymbol{Z}\})$$
$$+\frac{\rho}{2}\sum_{k=1}^{K} ||\boldsymbol{\Theta}^{(k)} - \boldsymbol{Z}^{(k)} + \boldsymbol{U}^{(k)}||_F^2,$$

where $\{\boldsymbol{U}\} = \{\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \dots, \boldsymbol{U}^{(k)}\}$.

The ADMM corresponding to the above problem results in iterating three simple steps.

(a) $\{\boldsymbol{\Theta}_{(i)}\} \leftarrow \arg\min_{\{\boldsymbol{\Theta}\}}\{L_\rho(\{\boldsymbol{\Theta}\},\{\boldsymbol{Z}_{(i-1)}\},\{\boldsymbol{U}_{(i-1)}\})\}$

(b) $\{\boldsymbol{Z}_{(i)}\} \leftarrow \arg\min_{\{\boldsymbol{Z}\}}\{L_\rho(\{\boldsymbol{\Theta}_{(i)}\},\{\boldsymbol{Z}\},\{\boldsymbol{U}_{(i-1)}\})\}$

(c) $\{\boldsymbol{U}_{(i)}\} \leftarrow \{\boldsymbol{U}_{(i-1)}\} + (\{\boldsymbol{\Theta}_{(i)}\} - \{\boldsymbol{Z}_{(i)}\})$.

The update in step (a) preserves positive-definiteness, thus this constraint can be satisfied simply by initializing $\boldsymbol{\Theta}_{(0)}^{(k)} = \boldsymbol{I}$.

Simulation Study

The data for main simulation study consisted of generating three networks with p = 500 features belonging to ten equally sized unconnected subnetworks, each with a power law degree distribution, i.e. a scale-free network. Of the ten subnetworks, eight have the same structure and edge values in all three classes, one is identical between the first two classes and missing in the third (i.e. the corresponding features are singletons in the third network), and one is present in only the first class. The corresponding graph is shown in Figure 1.

Simulation Study

The data for main simulation study consisted of generating three networks with p = 500 features belonging to ten equally sized unconnected subnetworks, each with a power law degree distribution, i.e. a scale-free network. Of the ten subnetworks, eight have the same structure and edge values in all three classes, one is identical between the first two classes and missing in the third (i.e. the corresponding features are singletons in the third network), and one is present in only the first class. The corresponding graph is shown in Figure 1.

In addition to the 500-feature network pair, we generate a pair of networks with p = 1000 features, each of which is block diagonal with 500×500 blocks corresponding to two copies of the 500-feature networks just described.

Figure 1

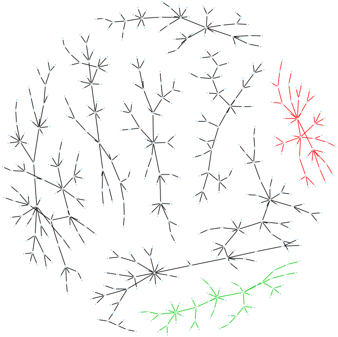


Figure: This shows the graph corresponding to the concentration matrix for the main simulation study. Black edges are common to all three classes, green edges are present only in classes 1 and 2, and red edges are present only in class 1.

Basic Result

As we will see in the next slides, when $p = 500$ and $n = 150$, FGL performs the best in the simulations conducted by Danaher et. al. although it is much slower than $glasso$ (the fastest) and the group lasso penalty case. The graphical lasso and the joint estimation method introduced by Guo et al. (2011) are included in the comparisons as well.

Basic Result

As we will see in the next slides, when $p = 500$ and $n = 150$, FGL performs the best in the simulations conducted by Danaher et. al. although it is much slower than $glasso$ (the fastest) and the group lasso penalty case. The graphical lasso and the joint estimation method introduced by Guo et al. (2011) are included in the comparisons as well.

In addition, FGL also seems to do better than GGL asymptotically when we hold p to be fixed and let n increase.

In terms of parameters:

FGL is presented in terms of λ_2 but for GGL they reparametrize the results in terms $\omega_2 = \frac{1}{\sqrt{2}}\lambda_2/(\lambda_1 + \frac{1}{\sqrt{2}}\lambda_2)$. Each line corresponds to a different value of λ_2 and ω_2.

In terms of parameters:

FGL is presented in terms of λ_2 but for GGL they reparametrize the results in terms $\omega_2 = \frac{1}{\sqrt{2}}\lambda_2/(\lambda_1 + \frac{1}{\sqrt{2}}\lambda_2)$. Each line corresponds to a different value of λ_2 and ω_2.

As λ_1 or $\omega_1 = \lambda_1 + \frac{1}{\sqrt{2}}\lambda_2$ increases, sparsity increases, or equivalently, the number of edges selected decreases.

In terms of parameters:

FGL is presented in terms of λ_2 but for GGL they reparametrize the results in terms $\omega_2 = \frac{1}{\sqrt{2}}\lambda_2/(\lambda_1 + \frac{1}{\sqrt{2}}\lambda_2)$. Each line corresponds to a different value of λ_2 and ω_2.

As λ_1 or $\omega_1 = \lambda_1 + \frac{1}{\sqrt{2}}\lambda_2$ increases, sparsity increases, or equivalently, the number of edges selected decreases.

According to the authors, approaches such as AIC, BIC, and cross-validation tend to choose models too large to be useful. In this setting, model selection should be guided by practical considerations, such as network interpretability, stability, and the desire for an edge set with a low false discovery rate.

Some measures of error

$$SSE = \sum_{k=1}^{K} \sum_{i \neq j} (\hat{\theta}_{ij}^{(k)} - (\Sigma^{(k)})_{ij}^{-1})^2$$

Some measures of error

$$SSE = \sum_{k=1}^{K} \sum_{i \neq j} (\hat{\theta}_{ij}^{(k)} - (\Sigma^{(k)})_{ij}^{-1})^2$$

Differential edges are defined as the edges that differ between classes. For FGL, it is computed as the number of pairs $k < k', i < j$ such that $\hat{\theta}_{ij}^{(k)} \neq \hat{\theta}_{ij}^{(k')}$. For GGL, the proposal of Guo et al. (2011), and the graphical lasso it is computed as the number of pairs $k < k', i < j$ such that $|\hat{\theta}_{ij}^{(k)} - \hat{\theta}_{ij}^{(k')}| > 10^{-2}$.

Some measures of error

$$SSE = \sum_{k=1}^{K} \sum_{i \neq j} (\hat{\theta}_{ij}^{(k)} - (\Sigma^{(k)})_{ij}^{-1})^2$$

Differential edges are defined as the edges that differ between classes. For FGL, it is computed as the number of pairs $k < k', i < j$ such that $\hat{\theta}_{ij}^{(k)} \neq \hat{\theta}_{ij}^{(k')}$. For GGL, the proposal of Guo et al. (2011), and the graphical lasso it is computed as the number of pairs $k < k', i < j$ such that $|\hat{\theta}_{ij}^{(k)} - \hat{\theta}_{ij}^{(k')}| > 10^{-2}$.

The Kullback-Leibler Divergence (dKL) from the multivariate normal model with inverse covariance estimates $\Theta^{(1)}, ..., \Theta^{(k)}$to the multivariate normal model with the true precision matrices $\Sigma^{(1)}, ..., \Sigma^{(k)}$ is

$$\frac{1}{2} \sum_{k=1}^{K} (-\log \det(\Theta^{(k)} \Sigma^{(k)}) + \mathrm{trace}(\Theta^{(k)} \Sigma^{(k)}))$$

Figure 2 from Danaher et. al

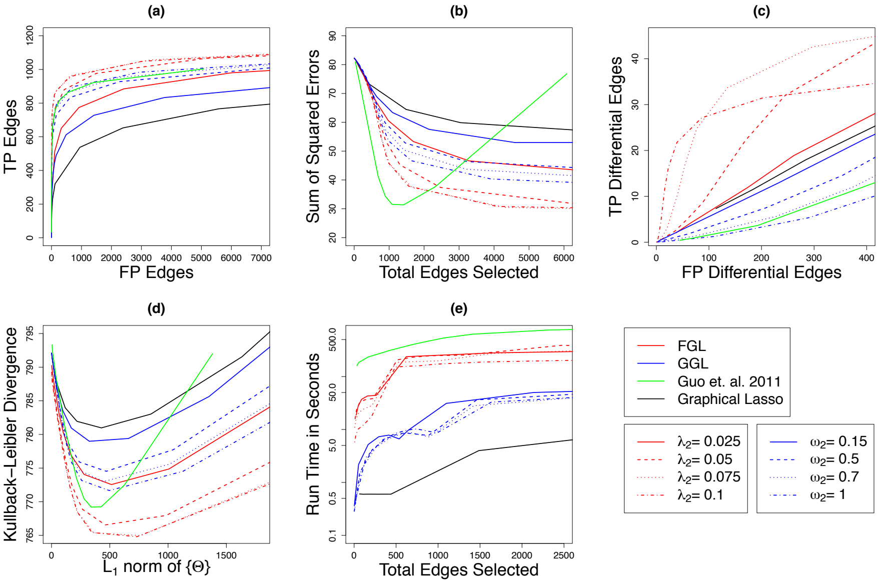


Figure 3 from Danaher et. al - Analysis of lung cancer microarray data

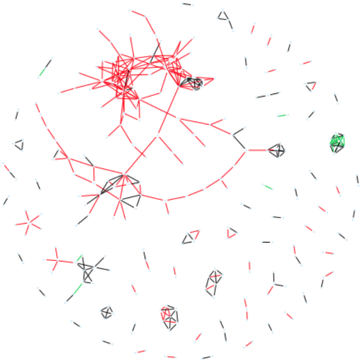


Figure: Conditional dependency networks inferred from 17,772 genes in normal and cancerous lung cells. 278 genes have nonzero edges in at least one of the two networks. Black lines denote edges common to both classes. Red and green lines denote tumor-specific and normal-specific edges, respectively. The parameters used were $\lambda_1 = 0.95$ and $\lambda_2 = 0.005$.

Table 1 from Danaher et. al

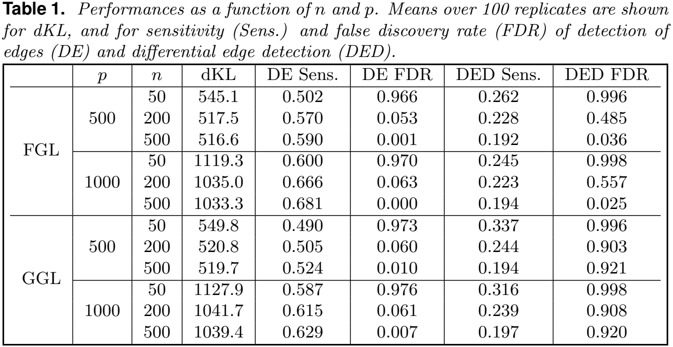


Table 1. *Performances as a function of n and p. Means over 100 replicates are shown for dKL, and for sensitivity (Sens.) and false discovery rate (FDR) of detection of edges (DE) and differential edge detection (DED).*

	p	n	dKL	DE Sens.	DE FDR	DED Sens.	DED FDR
FGL	500	50	545.1	0.502	0.966	0.262	0.996
		200	517.5	0.570	0.053	0.228	0.485
		500	516.6	0.590	0.001	0.192	0.036
	1000	50	1119.3	0.600	0.970	0.245	0.998
		200	1035.0	0.666	0.063	0.223	0.557
		500	1033.3	0.681	0.000	0.194	0.025
GGL	500	50	549.8	0.490	0.973	0.337	0.996
		200	520.8	0.505	0.060	0.244	0.903
		500	519.7	0.524	0.010	0.194	0.921
	1000	50	1127.9	0.587	0.976	0.316	0.998
		200	1041.7	0.615	0.061	0.239	0.908
		500	1039.4	0.629	0.007	0.197	0.920

Table 1 indicates that

for fixed p, as n increases, FGL seems to do much better in terms of edge detection (as expected, sensitivity increases and false discovery rate decreases as n increases) and differential edge detection (false discovery rate decreases as n increases) than GGL. Oddly enough, in all cases, DED Sens declines as n increases .

Nearest Neighbor Network (n = 100, p =25)

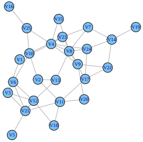
	Group	
	$\lambda_1 = 0.07, \lambda_2 = 0.05$	$\lambda_1 = 0.05, \lambda_2 = 0.25$
F	0.05642315	0.06010037
FP	0.3391197	0.02918924
FN	0.3035487	0.8517427

Table: Nearest Neighbor Network for Group Lasso Penalty

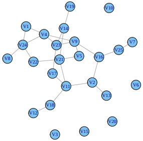
	Fused	
	$\lambda_1 = 0.05, \lambda_2 = 0.05$	$\lambda_1 = 0.2, \lambda_2 = 0.1$
F	0.06335103	0.06954624
FP	0.4269327	0.002245857
FN	0.3139659	0.9723261

Table: Nearest Neighbor Network for Fused Lasso Penalty

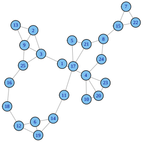
$\hat{\Omega}_1$ for Nearest Neighbor Network (n = 100, p =25)



(a) Estimated Nearest Neighbor Graph with Fused Lasso Penalty

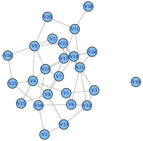


(b) Estimated Nearest Neighbor Graph with Group Lasso Penalty

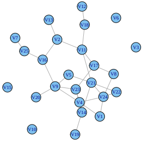


(c) True Nearest Neighbor Graph

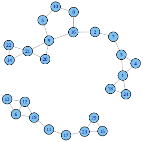
$\hat{\Omega}_2$ for Nearest Neighbor Network (n = 100, p =25)



(a) Estimated Nearest Neighbor Graph with Fused Lasso Penalty

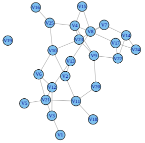


(b) Estimated Nearest Neighbor Graph with Group Lasso Penalty

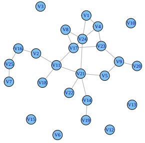


(c) True Nearest Neighbor Graph

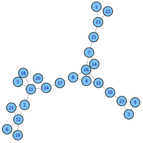
$\hat{\Omega}_3$ for Nearest Neighbor Network (n = 100, p =25)



(a) Estimated Nearest Neighbor Graph with Fused Lasso Penalty



(b) Estimated Nearest Neighbor Graph with Group Lasso Penalty



(c) True Nearest Neighbor Graph

Additional notes

In the appendix, they show that if you go down to 2 classes, FGL performs as fast as GGL while still maintaining the results best overall.

Additional notes

In the appendix, they show that if you go down to 2 classes, FGL performs as fast as GGL while still maintaining the results best overall.

We thought it was odd that the authors were much more interested in having a low false positive rate at the cost of a high false negative instead of a balance between the two.

Additional notes

In the appendix, they show that if you go down to 2 classes, FGL performs as fast as GGL while still maintaining the results best overall.

We thought it was odd that the authors were much more interested in having a low false positive rate at the cost of a high false negative instead of a balance between the two.

Danaher et al. claim that FGL and GGL are superior to existing methods such as time-varying networks (FGL and GGL do not require a natural ordering) and Guo et al. (penalty in Guo et.al lacks convexity, uses only one tuning parameter). In addition, FGL is superior to other methods when we expect edge values as well as network structure to be similar across classes.