**Econometrics** **June 10, 2023**

# Topic 16: Graphical Network Inference

*by Sai Zhang*

**Key points**:

**Disclaimer**: *The note is built on Prof. Jinchi Lv's lectures of the course at USC, DSO 607, High-Dimensional Statistics and Big Data Problems.*

## 16.1 Motivation

Consider a classic question: For $n$ observations of dimension $p$, how can we capture the statistical relationships between the variables of interest? Consider the example of the multivariate Gaussian distribution:

**Example 16.1.1: Multivariate Gaussian Distribution**

Suppose we have $n$ observations of dimension $p$, $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. let $\mathbf{S}$ be the empirical covariance matrix. Then the probability density

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}\det(\boldsymbol{\Sigma})^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

define the **inverse covariance matrix** or **precision matrix** as $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$, then we have

$$f_{\mu,\boldsymbol{\Omega}} = \exp\left\{\boldsymbol{\mu}'\boldsymbol{\Omega}\boldsymbol{x} - \left\langle \boldsymbol{\Omega}, \frac{1}{2}\mathbf{x}\mathbf{x}'\right\rangle - \frac{p}{2}\log(2\pi) + \frac{1}{2}\log\det(\boldsymbol{\Omega}) - \frac{1}{2}\boldsymbol{\mu}'\boldsymbol{\Omega}\boldsymbol{\mu}\right\}$$

where $\langle \mathbf{A}, \mathbf{B}\rangle = \mathrm{tr}(\mathbf{A}\mathbf{B})$.

In this example, we know that **every** multivariate Gaussian distribution can be represented by a pairwise **Gaussian Markov Random Field (GMRF)**, which an **undirected graph** $G = (V, E)$

- representing the collection of variables $\mathbf{x}$ by a vertex set $\mathcal{V} = \{1, \cdots, p\}$
- encoding correlations between variables by a set of edges $\mathcal{E} = \{(i, j) \in \mathcal{V} \mid i =\neq j, \Omega_{ij} \neq 0\}$

For simplicity, we normalize $\boldsymbol{\mu} = \mathbf{0}$. If we draw $n$ i.i.d. samples $\mathbf{x}_1, \cdots, \mathbf{x}_n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, then the log-likelihood is

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Omega}) &= \frac{1}{n}\sum_{i=1}^{n}\log f(\mathbf{x}_i) = \frac{1}{2}\log\det(\boldsymbol{\Omega}) - \frac{1}{2n}\sum_{i=1}^{n}\mathbf{x}_1'\boldsymbol{\Theta}\mathbf{x}_i \\ &= \frac{1}{2}\log\det(\boldsymbol{\Omega}) - \frac{1}{2}\left\langle \Omega, \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i'\mathbf{x}_i'\right\rangle\end{aligned}$$

**What's the goal?** We want to estimate a **sparse** graph structure given $n \ll p$ i.i.d. observations. But what does sparsity means in this context? A sparse graph is **equivalent** to a sparse precision matrix: the precision matrix should have many 0s.

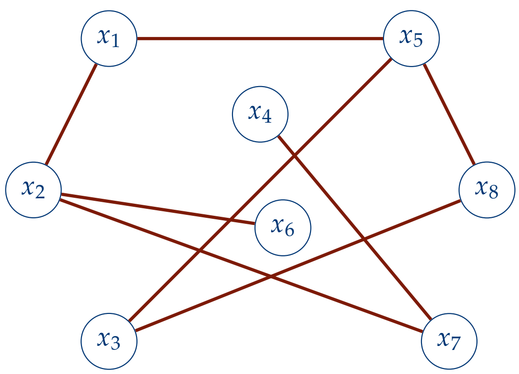
**Sparse precision matrix** for the Gaussian vector mentioned above $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, we have $\forall u, v$

$$x_u \perp x_v \mid \mathbf{x}_{\mathcal{V}\setminus\{u,v\}} \Leftrightarrow \Omega_{u,v} = 0$$

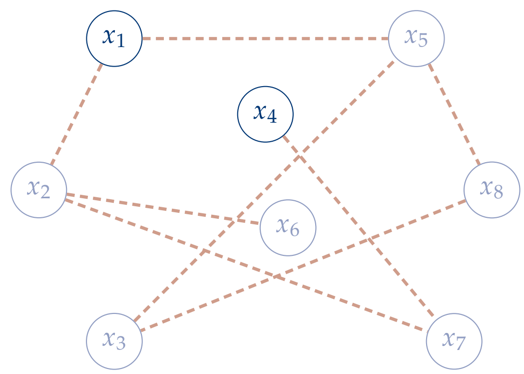
that is, sparsity of the precision matrix is equivalent to **conditional independence**[1]. Consider a graph, where $x_1$ and $x_4$ are only connected through other nodes, that is $x_1$ and $x_4$ are conditional independent, then we can have the precision matrix be something like:

$$\mathbf{\Theta} = \begin{bmatrix} * & * & 0 & 0 & * & 0 & 0 & 0 \\ * & * & 0 & 0 & 0 & * & * & 0 \\ 0 & 0 & * & 0 & * & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & * & 0 \\ * & 0 & * & 0 & * & 0 & 0 & * \\ 0 & * & 0 & 0 & 0 & * & 0 & 0 \\ 0 & * & 0 & * & 0 & 0 & * & 0 \\ 0 & 0 & * & 0 & * & 0 & 0 & * \end{bmatrix}$$

where 0 captures precisely the conditional independence.




$x_1$ **and** $x_4$ **are *connected***




$x_1$ **and** $x_4$ **are *NOT connected*, conditionally**

Intuitively, a sparse graph is much simpler, which is why conditional independence is desired. So how to achieve sparsity? We can again use a L-1 regularization when maximizing the log-likelihood $\mathcal{L}(\mathbf{\Omega})$. Denote the sample covariance matrix as $\mathbf{S} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i'$, then the problem becomes the so-called **Graphical Lasso**

$$\max_{\mathbf{\Omega} \geq \mathbf{0}} \log\det(\mathbf{\Omega}) - \operatorname{tr}(\mathbf{S\Omega}) - \rho \, \|\mathbf{\Omega}\|_1$$

which is equivalent to

$$\min_{\mathbf{\Omega} \geq \mathbf{0}} -\log\det(\mathbf{\Omega}) + \operatorname{tr}(\mathbf{S\Omega}) + \rho \, \|\mathbf{\Omega}\|_1$$

## 16.2 Graphical Lasso

The graphical lasso method is developed by (Friedman et al., 2008). For the optimization problem

$$\min_{\mathbf{\Omega} \geq \mathbf{0}} -\log\det(\mathbf{\Omega}) + \operatorname{tr}(\mathbf{S\Omega}) + \rho \, \|\mathbf{\Omega}\|_1 \tag{16.1}$$

[1]Meanwhile, for independence: $\Sigma_{u,v} = 0 \Leftrightarrow x_u \perp x_v$

The first-order optimality condition gives

$$\mathbf{0} \in \mathbf{\Omega}^{-1} - \mathbf{S} - \lambda \mathbf{\Gamma}$$

where $\mathbf{\Gamma}$ is a matrix of component-wise signs of $\mathbf{\Omega}$

$$\mathbf{\Gamma} = \partial \|\mathbf{\Omega}\|_1 \Rightarrow \gamma_{jk} \begin{cases} = \text{sign}(\omega_{jk}), & \omega_{jk} \neq 0 \\ \in [-1, 1], & \omega_{jk} = 0 \end{cases}$$

since in a graph, we always have that, following the global stationarity conditions, $\omega_{jj} > 0$, which implies that

$$w_{ii} = s_{ii} + \lambda \qquad\qquad i = 1, \cdots, p \tag{16.2}$$

where we denote a working version of $\mathbf{\Omega}^{-1}$ as $\mathbf{W}$.

The idea is to repeatedly cycle through all columns-rows and in each step optimize only a single column-row. Consider the following partition where all matrices are partitioned into one column/row versus the rest

$$\mathbf{\Omega} = \begin{pmatrix} \mathbf{\Omega}_{11} & \boldsymbol{\omega}_{12} \\ \boldsymbol{\omega}'_{12} & \omega_{22} \end{pmatrix} \qquad \mathbf{S} = \begin{pmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}'_{12} & s_{22} \end{pmatrix} \qquad \mathbf{W} = \begin{pmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}'_{12} & w_{22} \end{pmatrix} \qquad \mathbf{\Gamma} = \begin{pmatrix} \mathbf{\Gamma}_{11} & \boldsymbol{\gamma}_{12} \\ \boldsymbol{\gamma}'_{12} & \gamma_{22} \end{pmatrix}$$

apply this partition to the optimality condition, get

$$\mathbf{\Omega}^{-1} = \mathbf{S} - \lambda \mathbf{\Gamma}$$
$$\begin{pmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}'_{12} & w_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}'_{12} & s_{22} \end{pmatrix} + \lambda \begin{pmatrix} \mathbf{\Gamma}_{11} & \boldsymbol{\gamma}_{12} \\ \boldsymbol{\gamma}'_{12} & \gamma_{22} \end{pmatrix}$$

where $\mathbf{\Omega}_{11}$ is $(p-1) \times (p-1)$, $\boldsymbol{\omega}_{12}$ is $(p-1) \times 1$, $\omega_{22}$ is a scalar.

Consider a **blockwise** step: suppose we fix all but the last row/column, then using properties of inverses of block-partitioned matrices, we have

$$\begin{aligned} \begin{pmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}'_{12} & w_{22} \end{pmatrix} &= \begin{pmatrix} \left(\mathbf{\Omega}_{11} - \frac{\boldsymbol{\omega}_{12}\boldsymbol{\omega}'_{12}}{\omega_{22}}\right)^{-1} & -\mathbf{W}_{11}\frac{\boldsymbol{\omega}_{12}}{\omega_{22}} \\ & \frac{1}{\omega_{22}} - \frac{\boldsymbol{\omega}'_{12}\mathbf{W}_{11}\boldsymbol{\omega}_{12}}{\omega_{22}^2} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{\Omega}_{11}^{-1} + \frac{\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}\boldsymbol{\omega}'_{12}\mathbf{\Omega}_{11}^{-1}}{\omega_{22} - \boldsymbol{\omega}'_{12}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}} & -\frac{\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}}{\omega_{22} - \boldsymbol{\omega}'_{12}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}} \\ & \frac{1}{\omega_{22} - \boldsymbol{\omega}'_{12}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}} \end{pmatrix} \end{aligned}$$

then, by the partitioned optimality condition, we have[2]:

$$\mathbf{0} = -\mathbf{w}_{12} + \mathbf{s}_{12} + \lambda \boldsymbol{\gamma}_{12} = \mathbf{W}_{11}\frac{\boldsymbol{\omega}_{12}}{\omega_{22}} + \mathbf{s}_{12} + \lambda \boldsymbol{\gamma}_{12} \tag{16.3}$$

$$\mathbf{0} = \frac{\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}}{\omega_{22} - \boldsymbol{\omega}'_{12}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}} + \mathbf{s}_{12} + \lambda \boldsymbol{\gamma}_{12} = w_{22}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12} + \mathbf{s}_{12} + \lambda \boldsymbol{\gamma}_{12} \tag{16.4}$$

The graphic Lasso algorithm them solves Eq.16.3 for $\boldsymbol{\beta} = \boldsymbol{\omega}_{12}/\omega_{12}$, that is

$$\mathbf{W}_{11}\boldsymbol{\beta} + \mathbf{s}_{12} + \lambda \boldsymbol{\gamma}_{12} = \mathbf{0}$$

[2]For Eq.16.4, by Eq.16.2, we know that $w_{22} = s_{22} + \lambda$, which is fixed.

where $\gamma_{12} \in \text{sign}(\boldsymbol{\beta})$ since $\omega_{22} > 0$, which is essentially solving:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^{p-1}} \left\{ \frac{1}{2} \boldsymbol{\beta}' \mathbf{W}_{11} \boldsymbol{\beta} + \boldsymbol{\beta}' \mathbf{s}_{12} + \lambda \left\| \boldsymbol{\beta} \right\|_1 \right\}$$

and $\mathbf{W}_{11} > 0$ is assumed to be fixed.

This problem is analogous to a lasso regression problem of the **last variable** on **the rest**, but the cross-product matrix $\mathbf{S}_{11}$ is replaced by its **current estimation** $\mathbf{W}_{11}$. It is relatively easier to solve using elementwise coordinate descent, then

$$\begin{aligned}
\mathbf{w}_{12} &= -\mathbf{W}_{11} \frac{\boldsymbol{\omega}_{12}}{\omega_{22}} && \Rightarrow \hat{\mathbf{w}}_{12} = -\mathbf{W}_{11} \hat{\boldsymbol{\beta}} && \text{Step 1} \\
w_{22} &= \frac{1}{\omega_{22}} - \frac{\boldsymbol{\omega}_{12}' \mathbf{W}_{11} \boldsymbol{\omega}_{12}}{\omega_{22}^2} && \Rightarrow \frac{1}{\hat{\omega}_{22}} = w_{22} - \hat{\boldsymbol{\beta}}' \hat{\mathbf{w}}_{12} && \text{Step 2} \\
\boldsymbol{\omega}_{12} &= -\mathbf{W}_{11}^{-1} \mathbf{w}_{12} \omega_{22} && \Rightarrow \hat{\boldsymbol{\omega}}_{12} = -\mathbf{W}_{11}^{-1} \hat{\mathbf{w}}_{12} \hat{\omega}_{22} && \text{Step 3}
\end{aligned}$$

notice that after solving for $\boldsymbol{\beta}$ and updating $\mathbf{w}_{12}$ in Step 1, the graphic Lasso procedure can move onto the next block, that is, only Step 1 is used in the loop, Step 2 and 3 can be done at the end. The algorithm can be summarized as:

**Algorithm 16.2.1: Graphical Lasso algorithm**

1 Initialize $\mathbf{W} = \mathbf{S} + \lambda \mathbf{I}$
- Cycle around the columns repeatedly, performing the following steps till convergence:
  a rearrange the rows/columns so that the target column is **the last** (implicitly)
  b solve the lasso problem, starting the solution from the previous round for this column
  c update the row/column (***off-diagonal***) of the covariance using $\hat{\mathbf{w}}_{12}$
  d save $\hat{\boldsymbol{\beta}}$ for this column in the matrix $\mathbf{B}$

3 after convergence, for every row/column, compute the diagonal entries $\hat{\omega}_{jj}$, and covert the $\mathbf{B}$ matrix to $\boldsymbol{\Omega}$

## 16.3 What Is GLasso Solving?

Again, consider the optimization problem

$$\min_{\boldsymbol{\Omega} \geq \mathbf{0}} - \log \det(\boldsymbol{\Omega}) + \text{tr}(\mathbf{S}\boldsymbol{\Omega}) + \rho \left\| \boldsymbol{\Omega} \right\|_1$$

and its stationarity condition

$$\mathbf{0} = \boldsymbol{\Omega}^{-1} - \mathbf{S} - \lambda \boldsymbol{\Gamma}$$

rewrite the stationarity condition

$$\mathbf{0} = \boldsymbol{\Omega}^{-1} - \mathbf{S} - \lambda \boldsymbol{\Gamma} = \boldsymbol{\Omega} - (\mathbf{S} + \lambda \boldsymbol{\Gamma})^{-1}$$

since $\boldsymbol{\Gamma} = \text{sign}(\boldsymbol{\Omega})$, write $\tilde{\boldsymbol{\Gamma}} = \lambda \boldsymbol{\Gamma}$, we have $\left\| \tilde{\boldsymbol{\Gamma}} \right\|_\infty \leq \lambda$. Denote the element-wise absolute value matrix of $\boldsymbol{\Omega}$ as abs($\boldsymbol{\Omega}$), then let $\tilde{\boldsymbol{\Gamma}} = \lambda \boldsymbol{\Gamma}$, $\mathbf{P} = \text{abs}(\boldsymbol{\Omega})$, we have

$$\begin{aligned}
\mathbf{0} &= \boldsymbol{\Omega} - (\mathbf{S} + \lambda \boldsymbol{\Gamma})^{-1} \\
&= \mathbf{P} \circ \text{sign}(\tilde{\boldsymbol{\Gamma}}) - (\mathbf{S} + \tilde{\boldsymbol{\Gamma}})^{-1}
\end{aligned}$$

and mechanically, we also have

$$\mathbf{P} \circ \left(\text{abs}(\tilde{\mathbf{\Gamma}}) - \lambda \mathbf{1}_p \mathbf{1}_p'\right) = \mathbf{0}$$
$$\left\|\tilde{\mathbf{\Gamma}}\right\|_\infty \leq \lambda$$

together, these are just hte KKT optimality condition for the following box-constrained SDP

$$\max_{\tilde{\mathbf{\Gamma}}:\left\|\tilde{\mathbf{\Gamma}}\right\|_\infty \leq \lambda} g(\tilde{\mathbf{\Gamma}}) := \log\det(\mathbf{S} + \tilde{\mathbf{\Gamma}}) + p \tag{16.5}$$

with the transformation $\mathbf{S} + \tilde{\mathbf{\Gamma}} = \mathbf{\Omega}^{-1}$. Essentially, this is the dual problem of the initial optimization problem, both of them are solved by the GLasso algorithm.

**Issues of GLasso method** :

- the non-monotonic behavior of Glasso in minimizing $f(\mathbf{\Omega})$
  - $\boldsymbol{\theta}_{12}$ is entangled in $\mathbf{W}_{11}$, which is ***incorrectly*** treated as a constant
  - after updating $\boldsymbol{\theta}_{12}$, the entire (working) covariance matrix $\mathbf{W}$ changes, but Glasso algorithm only updates $\mathbf{w}_{12}$ and $\mathbf{w}_{12}'$
- high dimesnionality problems
  - not computationally efficient when $p$ is ultra-large
  - $\mathbf{\Sigma}^{-1}$ doesn't exist when $p > n$
  - method is not scalable

Next, we address these issues by introducing some modifications.

## 16.4 Graphical Lasso: Modifications

### 16.4.1 Primal GLasso

Consider the optimality condition in Eq.16.4:

$$\mathbf{0} = \frac{\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}}{\omega_{22} - \boldsymbol{\omega}_{12}'\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12}} + \mathbf{s}_{12} + \lambda\boldsymbol{\gamma}_{12} = w_{22}\mathbf{\Omega}_{11}^{-1}\boldsymbol{\omega}_{12} + \mathbf{s}_{12} + \lambda\boldsymbol{\gamma}_{12}$$

Here, the dependence of the covariance submatrix $\mathbf{W}_{11}$ on $\mathbf{\Omega}_{12}$ is **explicit**. Let $\boldsymbol{\alpha} = \boldsymbol{\omega}_{12} w_{22}$ with fixed $w_{22} \geq 0$[3], then this optimality condition is essentially solving

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{p-1}} \left\{\frac{1}{2}\boldsymbol{\alpha}'\mathbf{\Omega}_{11}^{-1}\boldsymbol{\alpha} + \boldsymbol{\alpha}'\mathbf{s}_{12} + \lambda\left\|\boldsymbol{\alpha}\right\|_1\right\}$$

the minimizer of this problem $\hat{\boldsymbol{\alpha}}$ can then be used to derive the estimation for $\boldsymbol{\omega}_{12}$:

$$\hat{\boldsymbol{\omega}}_{12} = \frac{\hat{\boldsymbol{\alpha}}}{w_{22}}$$

, then we can update $\omega_{22}$ as before via

$$\hat{\omega}_{22} = \frac{1}{w_{22}} + \hat{\boldsymbol{\omega}}_{12}'\mathbf{\Theta}_{11}^{-1}\hat{\boldsymbol{\omega}}_{12}$$

[3] $w_{22} = 1/(\omega_2 2 - \boldsymbol{\omega}_{12}'\mathbf{\Omega}_{11}^{-1}\omega_{12})$

with $w_{22} = s_{22} + \lambda$. Another problem is how to obtain $\mathbf{\Omega}_{11}^{-1}$: as the iterations proceed, maintain $\mathbf{W} = \mathbf{\Omega}^{-1}$, and $\mathbf{\Omega}_{11}^{-1}$ can be derived from

$$\mathbf{\Omega}_{11}^{-1} = \mathbf{W}_{11} - \frac{\mathbf{w}_{12}\mathbf{w}_{12}'}{w_{22}}$$

once $\boldsymbol{\omega}_{12}$ is updated, the ***entire*** working covariance matrix $\mathbf{W}$ is updated using $\mathbf{\Omega}_{11}^{-1}$. This procedure, the so-called primal graphical lasso, can be represented in the following algorithm:

**Algorithm 16.4.1: P-GLasso Algorithm**

1 Initialize $\mathbf{W} = \text{diag}(\mathbf{S}) + \lambda\mathbf{I}$ and $\mathbf{\Omega} = \mathbf{W}^{-1}$

2 Cycle around the columns repeatedly, performing the following steps till convergence:

  a rearrange the rows/columns so that the target column is the last (implicitly)

  b compute $\mathbf{\Omega}_{11}^{-1}$ using $\mathbf{\Omega}_{11}^{-1} = \mathbf{W}_{11} - \frac{\mathbf{w}_{12}\mathbf{w}_{12}'}{w_{22}}$

  c solve $\min_{\boldsymbol{\alpha}\in\mathbb{R}^{p-1}} \left\{\frac{1}{2}\boldsymbol{\alpha}'\mathbf{\Omega}_{11}^{-1}\boldsymbol{\alpha} + \boldsymbol{\alpha}'\mathbf{s}_{12} + \lambda \|\boldsymbol{\alpha}\|_1\right\}$ for $\boldsymbol{\alpha}$, using as warm starts the solution from the previous round of row/column updates. Then update $\hat{\boldsymbol{\omega}}_{12} = \hat{\boldsymbol{\alpha}}/w_{22}$ and $\hat{\omega}_{22}$

  d update $\mathbf{\Omega}$ and $\mathbf{W}$, ensuring that $\mathbf{\Omega}\mathbf{W} = \mathbf{I}_p$

3 output the solution: precision matrix $\mathbf{\Omega}$ and its exact inverse, covariance matrix $\mathbf{W}$

### 16.4.2 Innovated Scalable Efficient Estimation

Now, we try to tackle the high-dimesnionality issues: $\mathbf{\Sigma}^{-1}$ does **not** exist when $p > n$. Again,

$$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$$

consider a linear transformation where $\tilde{\mathbf{x}} = \mathbf{\Omega}\mathbf{x}$ ($\mathbf{\Omega}$ is still the precision matrix $\mathbf{\Sigma}^{-1}$), and

$$\text{cov}(\tilde{\mathbf{x}}) = \mathbf{\Omega}\text{cov}(\mathbf{x})\mathbf{\Omega} = \mathbf{\Omega}\mathbf{\Sigma}\mathbf{\Omega} = \mathbf{\Omega}$$

But $\mathbf{\Omega}$ is unknown and to be estimated. To get around this, we **break** the long vector $\tilde{\mathbf{x}}$ into small sub-vectors and then estimate each one.

**Notation** for subsets $A, B \subset \{1, \cdots, p\}$, let $\mathbf{x}_A$ denote a sub-vector of $\mathbf{x}$ formed by its components with indices in $A$, and the sub-(precision)-matrix is $\mathbf{\Omega}_{A,B} = (\omega_{jk})_{j\in A, k\in B}$, $\mathbf{\Omega}_A := \mathbf{\Omega}_{A,A}$ for simplicity. Then define

$$\tilde{\mathbf{x}}_A = \mathbf{\Omega}_A\boldsymbol{\eta}_A$$

where $\boldsymbol{\eta}_A = \mathbf{x}_A + \mathbf{\Omega}_A^{-1}\mathbf{\Omega}_{A,A^C}\mathbf{x}_{A^C}$, and $A^C := \{1, \cdots, p\} \setminus A$. With this definition, we have the following proposition:

**Proposition 16.4.2: Conditional Distribution of Sub-vectors**

Conditional distribution $\mathbf{x}_A \mid \mathbf{x}_B \sim \mathcal{N}\left(\boldsymbol{\mu}_{A|B}, \mathbf{\Sigma}_{A|B}\right)$, where

$$\boldsymbol{\mu}_{A|B} = \boldsymbol{\mu}_A + \mathbf{\Sigma}_{A,B}\mathbf{\Sigma}_B^{-1}(\mathbf{x}_B - \boldsymbol{\mu}_B)$$
$$\mathbf{\Sigma}_{A|B} = \mathbf{\Sigma}_A - \mathbf{\Sigma}_{A,B}\mathbf{\Sigma}_B^{-1}\mathbf{\Sigma}_{B,A}$$

and when $\mathbf{x}_B = \mathbf{x}_{A^C}$, we have

$$\mathbf{x}_A \mid \mathbf{x}_{A^C} \sim \mathcal{N}\left(-\boldsymbol{\Omega}_A^{-1}\boldsymbol{\Omega}_{A,A^C}\mathbf{x}_{A^C}, \boldsymbol{\Omega}_A^{-1}\right)$$

Prop.16.4.2 gives a multivariate linear regression model:

$$\mathbf{x}_A = \mathbf{C}_A'\mathbf{x}_{A^C} + \boldsymbol{\eta}_A$$

where $\mathbf{C}_A = -\boldsymbol{\Omega}_{A^C,A}\boldsymbol{\Omega}_A^{-1}$ is the coefficient matrix, and $\boldsymbol{\eta}_A$ is model errors with Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Omega}_A^{-1})$. Then we can have the following algorithm to solve this problem:

**Algorithm 16.4.3: ISEE Algorithm**

1 Let $(A_l)_{l=1}^L$ be a partition of index set $\{1, \cdots, p\}$, s.t. $\bigcup_{l=1}^L A_l = \{1, \cdots, p\}$
2 estimate $\boldsymbol{\eta}_{A_l}$ and then obtain estimated $\tilde{\mathbf{x}}_{A_l}$
3 stack all estimated sub-vectors $\tilde{\mathbf{x}}_{A_l}$ together to obtain $\tilde{\mathbf{x}}$

ISEE algorithm breaks large-scale precision estimation into **small-scale linear regression problems**, each of which is computationally efficient and effective.

**Estimation** for the $n \times p$ data matrix $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)'$, we construct the linear transformation

$$\tilde{\mathbf{X}} = (\tilde{\mathbf{x}}_1, \cdots, \tilde{\mathbf{x}}_n) = \mathbf{X}\boldsymbol{\Omega}$$

then the multivariate linear regression model by matrix notation is

$$\mathbf{X}_A = \mathbf{X}_{A^C}\mathbf{C}_A + \mathbf{E}_A$$

and the corresponding sub-matrix $\tilde{\mathbf{X}}_A$ can be written as

$$\tilde{\mathbf{X}}_A = (\mathbf{X}\boldsymbol{\Omega})_A = \mathbf{X}_A\boldsymbol{\Omega}_A + \mathbf{X}_{A^C}\boldsymbol{\Omega}_{A^C,A} = \left(\mathbf{X}_A + \mathbf{X}_{A^C}\boldsymbol{\Omega}_{A^C,A}\boldsymbol{\Omega}_A^{-1}\right)\boldsymbol{\Omega}_A = \mathbf{E}_A\boldsymbol{\Omega}_A$$

Sparsity is achieved bia scaled Lasso: for each node $j$ in index set $A$,

$$\mathbf{X}_j = \mathbf{A}^{\mathbf{C}}\boldsymbol{\beta}_j + \mathbf{E}_j$$

where $\boldsymbol{\beta}_j$ is the column of $\mathbf{C}_A$ corresponds to node $j$. The estimation is then done in the following steps:

- run the PLS

$$\left(\hat{\boldsymbol{\beta}}_j, \hat{\theta}_j^{1/2}\right) = \arg\min_{\boldsymbol{\beta}\in\mathbf{R}^{p-|A|},\sigma\geq 0}\left\{\frac{\left\|\mathbf{X}_j - \mathbf{X}_{A^C}\boldsymbol{\beta}\right\|_2^2}{2n\sigma} + \frac{\sigma}{2} + \lambda\left\|\boldsymbol{\beta}_*\right\|_1\right\}$$

  where $\boldsymbol{\beta}_*$ is component-wise product of $\boldsymbol{\beta}$ and $\left(\frac{1}{\sqrt{n}}\|\mathbf{X}_k\|_2\right)_{k\in A^C}$, and the penalizing factor is $\lambda = C\left(\frac{2\log p}{n}\right)$.
- after obtaining $\hat{\boldsymbol{\beta}}_j$, get the estimation of the (partitioned) precision matrix $\hat{\boldsymbol{\Omega}}_A$

$$\hat{\Omega}_A = \left(\frac{1}{n}\hat{\mathbf{E}}_A'\hat{\mathbf{E}}_A\right)^{-1}$$

  where $\hat{\mathbf{E}}_j = \mathbf{X}_j - \mathbf{X}_{A^C}\hat{\boldsymbol{\beta}}_j$, $\hat{\mathbf{E}}_A = \left(\hat{\mathbf{E}}_j\right)_{j\in A}$

- for the whole partition $(A_l)_{l=1}^{L}$, we have $\hat{\mathbf{X}} = (\hat{\mathbf{X}}_{A_l})_{1\le l\le L}$ where $\hat{\mathbf{X}}_{A_l} = \hat{\mathbf{E}}_{A_l}\hat{\mathbf{\Omega}}_{A_l}$, then the initital estiation of the whole precision matrix is $\hat{\Omega}_{ISEE,ini} = \frac{1}{n}\hat{\mathbf{X}}'\hat{\mathbf{X}}$
- next, introduce a threshold $\tau \ge 0$, define

$$\hat{\mathbf{\Omega}}_{ISEE,g} = T_\tau\left(\hat{\mathbf{\Omega}}_{ISEE,ini}\right)$$

where $T_\tau(\mathbf{B}) = \left(b_{jk}\mathbf{1}_{|b_{jk}|\ge\tau}\right)$ for matrix $\mathbf{B} = \left(b_{jk}\right)$, then estimate the structure $\mathbf{E}$ as $\hat{\mathbf{E}}_{ISEE} = \operatorname{supp}\left(\hat{\mathbf{\Omega}}_{ISEE,g}\right)$. One can then use the cross validation method to choose the optimal threshold:

- randomly split sample of $n$ rows into 2 samples of $n_1$ and $n_2$, repeat $N_1$ times. Denote $\hat{\Omega}^{1,\nu}_{ISEE,ini}, \hat{\Omega}^{2,\nu}_{ISEE,ini}$ the corresponding covariance matrices
- choose $\tau$ by minimizing

$$\mathcal{R}(\tau) = \frac{1}{N_1}\sum_{\nu=1}^{N_1}\left\|T_\tau\left(\hat{\Omega}^{1,\nu}_{ISEE,ini}\right) - \hat{\Omega}^{1,\nu}_{ISEE,ini}\right\|^2$$

## 16.5 Heterogeneous Graphical Networks

Next, we introduce heterogeneity into this problem: the samples are now drawn from $k$ different subpopulations

$$\mathbf{X}^{(t)} = \left(\mathbf{X}_1^{(t)}, \cdots, \mathbf{X}_p^{(t)}\right) \sim \mathcal{N}\left(\mathbf{0}, \left(\mathbf{\Omega}^{(t)}\right)^{-1}\right) \qquad t = 1, \cdots, k$$

Some simple observations on this problem are

- For each node $1 \le j \le p$ in graph $1 \le t \le k$

$$\mathbf{X}_j^{(t)} \mid \mathbf{X}_{-j}^{(t)} \sim \mathcal{N}\left(\mathbf{X}_{-j}^{(t)\prime}\mathbf{C}_j^{(t)}, \frac{1}{\omega_{j,j}^{(t)}}\right)$$

with $\mathbf{C}_j^{(t)} = -\frac{\mathbf{\Omega}_{-j,j}^{(t)}}{\omega_{j,j}^{(t)}}$ and $\mathbf{\Omega}^{(t)} = \left(\omega_{a,b}^{(t)}\right)_{p\times p}$

- For each pair of nodes $(a, b)$,

$$\operatorname{Cov}\left(\boldsymbol{\epsilon}_a^{(t)}, \boldsymbol{\epsilon}_b^{(t)}\right) = \frac{\omega_{a,b}^{(t)}}{\omega_{a,a}^{(t)}\omega_{b,b}^{(t)}}$$

with $\boldsymbol{\epsilon}^{(t)} = \mathbf{X}_j^{(t)} - \mathbf{X}_{-j}^{(t)\prime}\mathbf{C}_j^{(t)} \sim \mathcal{N}\left(\mathbf{0}, \frac{1}{\omega_{j,j}^{(t)}}\right)$ **independent** across $t$

- Under null hypothesis on joint link strength vector

$$H_{0,ab} : \omega_{a,b}^{0} = \left(\omega_{a,b}^{(1)}, \cdots, \omega_{a,b}^{(k)}\right)' = \mathbf{0}$$

distributions of $\mathbf{X}_j^{(t)} \mid \mathbf{X}_{-j}^{(t)}$ across $t$ share **similar sparsity structure** on coefficient vectors $\mathbf{C}_j^{(t)}$

Based on these observations, we can turn the problem of multiple-network estimation into a problem of **high-dimensional multi-response linear regression with heterogeneity**.

## Nodewise Heterogeneous Multi-Response Regression

For each node $1 \le j \le p$,

$$\begin{pmatrix} \mathbf{X}_j^{(1)} \\ \mathbf{X}_j^{(2)} \\ \vdots \\ \mathbf{X}_j^{(k)} \end{pmatrix} = \begin{pmatrix} \mathbf{X}_{-j}^{(1)} & & & \\ & \mathbf{X}_{-j}^{(2)} & & \\ & & \ddots & \\ & & & \mathbf{X}_{-j}^{(k)} \end{pmatrix} \begin{pmatrix} \mathbf{C}_j^{(1)} \\ \mathbf{C}_j^{(2)} \\ \vdots \\ \mathbf{C}_j^{(k)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_j^{(1)} \\ \boldsymbol{\epsilon}_j^{(2)} \\ \vdots \\ \boldsymbol{\epsilon}_j^{(k)} \end{pmatrix}$$

The noises are **Heterogeneous** since $\omega_{j,j}^{(t)}$ generally varies over $1 \le t \le k$. To tackle this problem with a tuning-free estimation procedure, follow this roadmap:

- construct estimators based on residuals from the heterogeneous multi-response regressions
  - Initial estimation $\hat{\mathbf{C}}_j^0 = \left(\hat{\mathbf{C}}_j^{(1)\prime}, \cdots, \hat{\mathbf{C}}_j^{(k)\prime}\right)'$ for the $(p-1)k$ dimensional vector $\mathbf{C}_j^0 = \left(\mathbf{C}_j^{(1)\prime}, \cdots, \mathbf{C}_j^{(k)\prime}\right)'$ with the tuning-free heterogeneous group square-root Lasso (HGSL)

    $$\hat{\mathbf{C}}_1^0 = \arg\min_{\boldsymbol{\beta}^0 \in \mathbb{R}^{(p-1)k}} \left\{ \sum_{t=1}^{k} \sqrt{Q_t\left(\boldsymbol{\beta}^{(t)}\right)} + \lambda \sum_{l=2}^{p} \left\| \sqrt{D_{1(l)}} \boldsymbol{\beta}_{(l)}^0 \right\| \right\}$$

    where $Q_t\left(\boldsymbol{\beta}^{(t)}\right)$ is the quadratic loss over class $t$.
  - Get the residuals $\hat{\mathbf{E}}_{i,j}^{(t)} = \mathbf{X}_{i,j}^{(t)} - \mathbf{X}_{i,-j}^{(t)\prime} \hat{\mathbf{C}}_j^{(t)}$ with $1 \le i \le n^{(t)}, 1 \le j \le p$, and $n^{(t)}$ the sample size for graph $t$.
  - then get
    * an estimator for $\hat{\omega}_{j,j}^{(t)} = \frac{n^{(t)}}{\sum_{i=1}^{n^{(t)}} \hat{\mathbf{E}}_{i,j}^{(t)\prime} \hat{\mathbf{E}}_{i,j}^{(t)\prime}}$
    * a bias-corrected statistic for estimating $-\mathrm{Cov}\left(\boldsymbol{\epsilon}_a^{(t)}, \boldsymbol{\epsilon}_b^{(t)}\right)$

      $$T_{n,k,a,b}^{(t)} = \frac{1}{n^{(t)}} \left[ \sum_{i=1}^{n^{(t)}} \hat{\mathbf{E}}_{i,a}^{(t)} \hat{\mathbf{E}}_{i,b}^{(t)} + \sum_{i=1}^{n^{(t)}} \left(\hat{\mathbf{E}}_{i,a}^{(t)}\right)^2 \hat{\mathbf{C}}_{b,a}^{(t)} + \sum_{i=1}^{n^{(t)}} \left(\hat{\mathbf{E}}_{i,b}^{(t)}\right)^2 \hat{\mathbf{C}}_{a,b}^{(t)} \right]$$

      where bias can occur if $a \neq b$, and the statistic $T_{n,k,a,b}^{(t)}$ asymptotically close to statistic

      $$J_{n,k,a,b}^{(t)} = \left[1 - \omega_{a,a}^{(t)} \left(\hat{\omega}_{a,a}^{(t)}\right)^{-1} - \omega_{b,b}^{(t)} \left(\hat{\omega}_{b,b}^{(t)}\right)^{-1}\right] \frac{\omega_{a,b}^{(t)}}{\omega_{a,a}^{(t)} \omega_{b,b}^{(t)}}$$

      which in turn asymptotically close to negative covariance $-\mathrm{Cov}\left(\boldsymbol{\epsilon}_a^{(t)}, \boldsymbol{\epsilon}_b^{(t)}\right)$.
- rely on 2 tests (chi-based and linear-functional-based) for common sparsity pattern with optimality
  - Chi-based test: test null hypothesis $H_{0,ab} : \omega_{a,b}^0 = \left(\omega_{a,b}^{(1)}, \cdots, \omega_{a,b}^{(k)}\right)' = \mathbf{0}$ for joint link strength vector, then the Chi-based test statistic is defined as

    $$U_{n,k,a,b}^2 = \sum_{t=1}^{k} n^{(t)} \hat{\omega}_{b,b}^{(t)} \hat{\omega}_{a,a}^{(t)} \left(T_{n,k,a,b}^{(t)}\right)^2$$

    for testing null $H_{0,ab}$ against alternative hypothesis involving $l_2$ norm $\left\|\omega_{a,b}^0\right\|_2$. The null distribution of the test statistic is asymptotically close to chi distribution, and the test is powerful when signal strength

$\left\|\omega^0_{a,b}\right\|$ is larger than some testable region boundary, and it is a general testing procedure without any additional information assumed on the $k$ networks.

- Linear functional-based test: with some extra knowledge, we can have a more powerful testing procedure. If sign relationship of some target edge across $k$ graphs is represented by a unique sign vector $\xi = (\xi_1, \cdots, \xi_k)' \in \{1, -1\}^k$, then we have the null as $H_{0,ab} : \omega^0_{a,b} = \mathbf{0}$ equivalent to $\left\|\omega^0_{a,b}\right\|_1 = \left|\sum_{t=1}^k \xi_t \omega^{(t)}_{a,b}\right| = 0$, then the **linear functional-based** test statistic is defined as

$$V_{n,k,a,b}(\xi) = \sum_{t=1}^{k} \xi_t \sqrt{n^{(t)} \hat{\omega_{a,a}}^{(t)} \hat{\omega_{b,b}}^{(t)}} T^{(t)}_{n,k,a,b}$$

- develop the scaled iterative thresholding HGSL algorithm with provable convergence for the scalability of Tuning-free heterogeneous inference (THI)

## References

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