glasso: Graphical lasso for learning sparse inverse covariance matrices

Aramayis Dallakyan

Department of Statistics
Texas A&M University

Stata Conference 2021

1Available in https://github.com/adallak/stataglasso
Acknowledgment

Part of this work was done while I was an Intern at StataCorp.

I am grateful to Yulia Marchenko and Houssein Assaad for their comments and suggestions.
Outline

1. Graphical Models
2. Graphical Lasso
3. Syntax
4. Numerical Results
A graph consists of a set of vertices (nodes) along with a set of edges joining pairs of the vertices.

Graphical model is a statistical object where each vertex represents a random variable.

The graph gives a visual way of understanding the joint distribution of the entire set of random variables.

Graphical models can be useful for either unsupervised or supervised learning.
Two popular types of graphs: **Directed Acyclic Graph** and **Undirected Graph**

DAGs or *Bayesian Networks* are graphical models in which the edges have directional arrows but no directed cycles.

The joint distribution can be factorized:

$$P(X_1, X_2, X_3) = P(X_3|X_1)P(X_1|X_2)$$

There is an intimate relationship between DAGs, Causality, and SEMs (Pearl, 2009; Peters et.al, 2017; Dallakyan and Pourahmadi, 2021).
We focus on undirected graphs, also known as **Markov random fields**.

In a Markov graph $\mathcal{G}$, the absence of an edge implies that the corresponding random variables are conditionally independent given the variables at the other vertices.

No edge joining $X_2$ and $X_3 \iff X_2 \perp\!\!\!\!\!\!\perp X_3 | \text{rest}$
We consider network where the random vector $X \sim N_p(\mathbf{0}, \Sigma)$.

A zero off-diagonal entry of the precision $\Theta = \Sigma^{-1}$ or $\theta_{j,k} = 0$ implies $X_j$ and $X_k$ are conditionally independent given all other variables.

\[
\begin{pmatrix}
\theta_{1,1} & \theta_{1,2} & \theta_{1,3} \\
\theta_{2,1} & \theta_{2,2} & 0 \\
\theta_{3,1} & 0 & \theta_{3,3}
\end{pmatrix}
\]
Precision Estimation

- The most common way to estimate the (inverse) covariance matrix is through **sample covariance** matrix

\[ S = \frac{X^t X}{n} \]

or through **Maximum Likelihood Estimator** (MLE).

The p-variate Gaussian distribution for \( X \in \mathbb{R}^p \) is given

\[ f(x) = (2\pi)^{-p/2}\det(\Sigma)^{-1/2}e^{-\frac{x^t \Sigma^{-1} x}{2}} \]

For the entire data \( X \), the likelihood function is \( L(\Theta) = f(x)^n \). Taking logarithm and after some algebra

\[ \arg \max_{\Theta} \ell(\Theta) = \log \det(\Theta) - \text{tr}(S\Omega) \]

- The MLE of \( \Sigma \) is \( S \). Unfortunately, when \( p, p/n \) is large, \( S \) performs poorly.

- Thus it is reasonable to impose structure on \( \Theta(\Sigma) \) or assume that they are sparse. That is some of \( \theta_{i,j} = 0 \).
There are two main approaches to introduce sparsity in $\Theta$.

**Regression based or Neighborhood Selection.** (Meinhausen and Buhlmann, 2006) Here the approach is based on the idea that the entries of $\theta_{ij}$ have regression interpretation.

In particular, $\theta_{ij}$ is **proportional** to the regression coefficient of variable $X_j$ in the multiple regression of variable $X_i$ on the rest.
The zeros in coefficient are forced by a column-by-column approach through penalized least square (lasso).

\[
\frac{1}{n} \| X_i - \sum_{j \neq i} \beta_{ij} X_j \|_2^2 + \lambda \sum_{i \neq j} |\beta_{ij}|.
\]

Disadvantages: Positive definiteness is not guaranteed and do not exploit the symmetry.
Graphical Lasso

- Glasso (Tibshirani et.al 2008) performs penalized MLE estimation, solving

\[
\arg \min_{\Theta \succ 0} \Theta = -\log \det(\Theta) + \text{tr}(S\Theta) + \lambda \sum_{i,j} |\theta_{ij}|
\]  

(1)

- The tuning parameter \( \lambda \) controls sparsity level; i.e., the larger \( \lambda \), the sparser is \( \Theta \).
- The optimization is convex and global minimum is achievable.
- The symmetry and positive definiteness of estimated \( \hat{\Theta} \) is guaranteed.
- Depends on the scaling of variables. Recommended to standardize the data before running Glasso.
Glasso Algorithm

- Glasso algorithm iteratively estimates $\Theta$ and its inverse $W = \Theta^{-1}$ by solving lasso regression one row and column at a time.

- Let look on KKT conditions, the subdifferential for minimizing (1) is

$$W - S - \lambda \Gamma = 0,$$

(2)

where $\gamma_{ij}$ element of the subgradient matrix $\Gamma$ takes the following form: $\gamma_{ij} = \text{sign}(\theta_{ij})$ if $i, j$th element $\theta_{ij} \neq 0$, and $\gamma_{ij} \in [-1, 1]$ if $\theta_{ij} = 0$.

- The genesis of the algorithm is in exploiting the partition of $W$ and its inverse $\Theta$.

- For illustration purposes, we discuss the algorithm by focusing on the last row and column of the partitined matrices.
Glasso Algorithm

From KKT

\[
\begin{pmatrix}
W_{11} & w_{12} \\
\dot{w}_{12} & w_{22}
\end{pmatrix}
- \begin{pmatrix}
S_{11} & s_{12} \\
\dot{s}_{12} & s_{22}
\end{pmatrix}
- \lambda \begin{pmatrix}
\Gamma_{11} & \gamma_{12} \\
\dot{\gamma}_{12} & \gamma_{22}
\end{pmatrix}
= \begin{pmatrix}
0 & 0 \\
0' & 0
\end{pmatrix}
\]

\[w_{12} - s_{12} - \lambda \gamma_{12} = 0.\]
\[
\begin{bmatrix}
W_{11} & w_{12} \\
w_{12}' & w_{22}
\end{bmatrix}
\begin{bmatrix}
\Theta_{11} & \theta_{12} \\
\theta_{12}' & \theta_{22}
\end{bmatrix}
= \begin{bmatrix}
I \\
0'
\end{bmatrix}
\begin{bmatrix}
0 \\
1
\end{bmatrix}
\]

\[w_{12} = -W_{11} \frac{\theta_{12}}{\theta_{22}} = W_{11} \beta,\]  

(4)

where \( \beta = -\theta_{12}/\theta_{22} \).
After substituting (4) into (3), we obtain

\[ W_{11}\beta - s_{12} + \lambda \text{sign}(\beta) = 0, \tag{5} \]

where we used the fact that \( \beta \) and \( \theta_{12} \) have opposite signs.

After some algebra, Friedman et.al (2008) show that (5) is equivalent to lasso regression.

For each column, authors resort to pathwise coordinate descent algorithm to solve the modified lasso problem (5) by iterating for \( j = 1, 2, \ldots, p - 1, \ldots \) until convergence

\[ \hat{\beta}_j = S(s_{12,j} - \sum_{k \neq j} V_{k,j} \hat{\beta}_k, \lambda)/V_{jj}, \tag{6} \]

where \( V = W_{11} \) and \( S(x, \lambda) = \text{sign}(x)(|x| - \lambda)_+ \) is the soft-threshold operator.
Glasso Algorithm

1: input:
2: \( S, \lambda \leftarrow \) Sample covariance matrix and penalty parameter
3: top:
4: Initialize \( W = S + \lambda I \)
5: Repeat for \( j = 1, 2, \ldots, p \) until convergence
6: (a) Solve the modified lasso problem (5)
7: (b) Update \( w_{12} = W_{11} \hat{\beta} \)
8: In the final cycle solve \( \hat{\theta}_{12} = -\hat{\beta} \cdot \hat{\theta}_{22} \)
9: Output:
10: \( \Theta, W \)
In real-world applications, the value of penalty parameter $\lambda$ is unknown and, traditionally, is treated as a tuning parameter to be selected from data.

The value of $\lambda$ is directly connected to the sparsity of $\Theta$; i.e., the higher $\lambda$, the sparser is the inverse covariance matrix $\Theta$.

We discuss two popular methods for tuning parameter selection: **Cross-validation** and **eBIC**.
For $K$–fold cross-validation, we randomly split the full dataset $\mathcal{D}$ into $K$ subsets of about the same size, denoted by $\mathcal{D}^\nu$, $\nu = 1, \ldots, K$.

For each $\nu$, $\mathcal{D} - \mathcal{D}^\nu$ is used to estimate parameters and $\mathcal{D}^\nu$ to validate.

$$CV(\lambda) = \frac{1}{K} \sum_{\nu=1}^{K} \left( -d_\nu \log |\hat{\Theta} - \nu| + \sum_{I_\nu} y_i^t \hat{\Theta} - \nu y_i \right),$$  \hspace{1cm} (7)$$

where $\hat{\Theta} - \nu$ is the estimated precision matrix using the data set $\mathcal{D} - \mathcal{D}^\nu$, and $y_i$ is the $i$th observation of the dataset $\mathcal{D}$. 


The eBIC criterion, introduced in Foygel and Drton (2010), takes the form

$$\text{eBIC}_\gamma = -n \log |\Theta| + \text{tr}(S\Theta) + E \log n + 4E\gamma \log p,$$

where $E$ is the number of non-zero off-diagonal elements of the inverse covariance matrix $\Theta$.

The criterion is indexed by a parameter $\gamma \in [0, 1]$ and $\gamma = 0$ case is the classical BIC criterion.

Positive $\gamma$ leads to the stronger penalization of large inverse covariance matrices, and results to the model selection criterion with a good theoretical properties.

Resorting to simulation results, authors suggest $\gamma = 0.5$ as a proposed value.
```plaintext
glasso varlist[if] [in] [ , lambda(#) maxiter(#) tolerance(#) diag]
```

<table>
<thead>
<tr>
<th>options</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lambda(#)</td>
<td>Penalty parameter.</td>
</tr>
<tr>
<td>maxiter(#)</td>
<td>Maximum number of iteration.</td>
</tr>
<tr>
<td>tolerance(#)</td>
<td>Maximum tolerance for convergence.</td>
</tr>
<tr>
<td>diag</td>
<td>Should diagonal be penalized?</td>
</tr>
</tbody>
</table>
cvglasso `varlist [if] [in] [, lamlist(numlist) nlam(#) maxiter(#) tolerance(#) nfold(#) crit(string) gamma(#) diag]`

<table>
<thead>
<tr>
<th>options</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lamlist(numlist)</td>
<td>Grid of positive tuning parameters for penalty term.</td>
</tr>
<tr>
<td></td>
<td>If provided, causes to disregard <code>nlam</code>.</td>
</tr>
<tr>
<td>nlam(#)</td>
<td>Number of generated tuning parameters for penalty term.</td>
</tr>
<tr>
<td>maxiter(#)</td>
<td>Maximum number of iteration.</td>
</tr>
<tr>
<td>tolerance(#)</td>
<td>Maximum tolerance for convergence.</td>
</tr>
<tr>
<td>crit(string)</td>
<td>Type of the criterion. Possible options are <code>loglik</code> and <code>eBIC</code>.</td>
</tr>
<tr>
<td>gamma(#)</td>
<td>Activated if <code>crit</code> is <code>eBIC</code>.</td>
</tr>
<tr>
<td>diag</td>
<td>Should diagonal be penalized?</td>
</tr>
</tbody>
</table>
plotglasso Syntax

plotglasso matname [, type(string) newlabs(lab1 lab2 ...) nwplot_options
                      nwplotmatrix_options]

<table>
<thead>
<tr>
<th>options</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type(string)</td>
<td>Type of the plot: graph or matrix.</td>
</tr>
<tr>
<td>newlabs(lab1 lab2)</td>
<td>Labels for the plot.</td>
</tr>
<tr>
<td>nwplot_options</td>
<td>Options for undirected graph plot.</td>
</tr>
<tr>
<td>nwplotmatrix_options</td>
<td>Options for matrix plot.</td>
</tr>
<tr>
<td></td>
<td>For details see (Grund and Hedstrom 2021)</td>
</tr>
</tbody>
</table>
`glasso` and `cvglasso` save the following in r()

Scalar
- r(lambda)    Tuning parameter

Matrix
- r(Omega)     Inverse covariance matrix
- r(Sigma)     Covariance matrix
Simulation

- We simulate data from the Erdos-Renyi graph, where probability that there is an edge between two nodes is 0.1.

- We select sample size \( n = 50, 150 \) and dimension \( p = 100 \), covering settings where \( p < n \) and \( p > n \), respectively.

- Each simulation setting is run over 20 repetitions and each dataset were standardized before implementing Glasso algorithm.
Simulation result: Undirected Graph

True Precision Matrix

CV

BIC

eBIC
Simulation result: Matrix

True Precision Matrix

CV

BIC

eBIC
Simulation result: Metric

<table>
<thead>
<tr>
<th></th>
<th>CV</th>
<th>BIC</th>
<th>eBIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPR</td>
<td>0.71(0.26)</td>
<td>0.98(0.03)</td>
<td>0.99(0.02)</td>
</tr>
<tr>
<td>FPR</td>
<td>0.0001(0.00)</td>
<td>0.0001(0.00)</td>
<td>0.0001(0.00)</td>
</tr>
<tr>
<td>TDR</td>
<td>0.97(0.03)</td>
<td>0.83(0.10)</td>
<td>0.95(0.10)</td>
</tr>
</tbody>
</table>

**Table:** Averages of three metric over 20 simulated repetitions for the $n = 150$, $p = 100$ case.
The flow-cytometry dataset, borrowed from Hastie et al. (2009), contains measures of 11 proteins on 7466 cells.

### Table 2: Summary of flow-cytometry data

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAF</td>
<td>7,466</td>
<td>6.09e-06</td>
<td>247.5281</td>
<td>-123.0719</td>
<td>4489.928</td>
</tr>
<tr>
<td>MEK</td>
<td>7,466</td>
<td>-0.0000317</td>
<td>377.0562</td>
<td>-144.381</td>
<td>6959.619</td>
</tr>
<tr>
<td>PLCG</td>
<td>7,466</td>
<td>3.35e-06</td>
<td>173.8598</td>
<td>-53.85364</td>
<td>6153.146</td>
</tr>
<tr>
<td>PIP2</td>
<td>7,466</td>
<td>0.0000198</td>
<td>299.3475</td>
<td>-150.1207</td>
<td>8906.88</td>
</tr>
<tr>
<td>PIP3</td>
<td>7,466</td>
<td>1.29e-06</td>
<td>43.04816</td>
<td>-26.03496</td>
<td>1247.965</td>
</tr>
<tr>
<td>ERK</td>
<td>7,466</td>
<td>2.16e-06</td>
<td>45.82672</td>
<td>-25.63119</td>
<td>2544.369</td>
</tr>
<tr>
<td>AKT</td>
<td>7,466</td>
<td>5.19e-06</td>
<td>137.7662</td>
<td>-80.16721</td>
<td>3473.833</td>
</tr>
<tr>
<td>PKA</td>
<td>7,466</td>
<td>-0.0000444</td>
<td>644.4593</td>
<td>-624.7586</td>
<td>8270.241</td>
</tr>
<tr>
<td>PKC</td>
<td>7,466</td>
<td>-3.46e-06</td>
<td>92.87002</td>
<td>-29.34166</td>
<td>1580.658</td>
</tr>
<tr>
<td>P38</td>
<td>7,466</td>
<td>-8.18e-06</td>
<td>494.7688</td>
<td>-134.0145</td>
<td>7363.985</td>
</tr>
<tr>
<td>JNK</td>
<td>7,466</td>
<td>-2.78e-06</td>
<td>215.6606</td>
<td>-72.2675</td>
<td>4666.732</td>
</tr>
</tbody>
</table>
Flow-cytometry Data

CV, $\lambda = .01$

eBIC, $\lambda = .01$
Data is borrowed from Demirer et al. (2018), where authors estimate the global bank network connectedness.

Original data contains 96 banks from 29 developed and emerging economies (countries) from September 12, 2003, to February 7, 2014.

For illustration purposes, we select only economies where the number of banks in each economy is greater than 4, total of 54 banks.

To visualize the result, we exploit a multidimensional scaling algorithm (Hastie et al. 2009) to calculate proximities between variables.
*Colors in the figure indicate the corresponding country of the bank.*
Possible Future Feature

- Graphical Lasso for the discrete data (Loh and Wainwright, 2012)
- Joint Graphical Lasso (Danaher et.al., 2014)
- Time series Graphical Lasso (Dallakyan et.al., 2021, Jung et.al., 2015)
- Time Varying Graphical Lasso (Hallac et.al, 2017)
Flow-cytometry Data

The greatest value of a picture is when it forces us to notice what we never expected to see.

— John Tukey —