glasso: Graphical lasso for learning sparse inverse covariance matrices [1]

Aramayis Dallakyan

Department of Statistics
Texas A&M University

Stata Conference 2021

[1]Available 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.

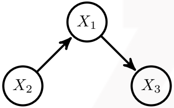
Outline

Graphical Models

- 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.

Directed Acyclic Graphs

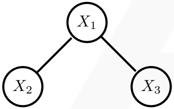
- Two popular type of graphs : **Directed Acyclic Graph** and **Undirected Graph**
- DAGs or *Bayesian Networks* are graphical models in which the edges have directional arrows but no directed cycles.



- 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).

Undirected Graphs

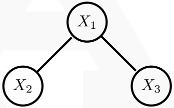
- We focus on undirected graphs, also known as **Markov random fields**.
- In a Markov graph $\mathcal{G}$, the absence of an edge implies that the corresponding random variables are conditionally independent given the variables at the other vertices.



- No edge joining X_2 and X_3 $\iff X_2 \perp\!\!\!\perp X_3 |$rest

Gaussian Undirected Graph

- We consider network where the random vector $X \sim N_p(\mathbf{0}, \boldsymbol{\Sigma})$.
- A zero off-diagonal entry of the precision $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$ or $\theta_{j,k} = 0$ implies X_j and X_k are conditionally independent given all other variables.



$$\Longleftrightarrow \quad \boldsymbol{\Theta} = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} & \theta_{1,3} \\ \theta_{2,1} & \theta_{2,2} & 0 \\ \theta_{3,1} & 0 & \theta_{3,3} \end{pmatrix}$$

Precision Estimation

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

$$\boldsymbol{S} = \frac{\mathbf{X}^t\mathbf{X}}{n}$$

or through **Maximum Likelihood Estimator** (MLE).
The p-variate Gaussian distribution for $X \in R^p$ is given

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

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

$$\arg\max_{\boldsymbol{\Theta}} \ell(\boldsymbol{\Theta}) = \log \mathsf{det}(\boldsymbol{\Theta}) - \mathsf{tr}(\boldsymbol{S}\boldsymbol{\Omega})$$

- The MLE of $\boldsymbol{\Sigma}$ is $\boldsymbol{S}$. Unfortunately, when p, p/n is large, $\boldsymbol{S}$ performs poorly.
- Thus it is reasonable to impose structure on $\boldsymbol{\Theta}(\boldsymbol{\Sigma})$ or assume that they are sparse. That is some of $\theta_{i,j} = 0$.

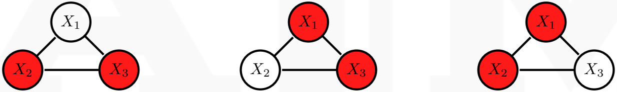
High dimensional Precision Estimation

- There are two main approaches to introduce sparsity in $\boldsymbol{\Theta}$.
- **Regression based or Neighborhood Selection.**(Meinhausen and Buhlmann, 2006) Here the approach is based on the idea that the entries of θ_{ij} have regression interpretation.
- In particular, θ_{ij} is **proportional** to the regression coefficient of variable X_j in the multiple regression of variable X_i on the **rest**.

Neighborhood Regression

- The **zeros** in coefficient are forced by a column-by column approach through penalized least square (lasso).

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



- Disandvantages: Positive definiteness is not guaranteed and do not exploit the symmetry.

Graphical Lasso

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

$$\arg\min_{\boldsymbol{\Theta} \succ 0} \boldsymbol{\Theta} = -\log \mathsf{det}(\boldsymbol{\Theta}) + \mathsf{tr}(\boldsymbol{S}\boldsymbol{\Theta}) + \lambda \sum_{i,j} |\theta_{ij}| \tag{1}$$

- The tuning parameter λ controls sparsity level; i.e., the larger λ, the sparser is $\boldsymbol{\Theta}$.
- 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 $\boldsymbol{\Theta}$ and its inverse $\boldsymbol{W} = \boldsymbol{\Theta}^{-1}$ by solving **lasso regression one row and column** at a time.
- Let look on KKT conditions, the subdifferential for minimizing (1) is

$$\boldsymbol{W} - \boldsymbol{S} - \lambda\boldsymbol{\Gamma} = \mathbf{0}, \tag{2}$$

  where γ_{ij} element of the subgradient matrix $\boldsymbol{\Gamma}$ takes the following form: $\gamma_{ij} = \text{sign}(\theta_{ij})$ if i, jth 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 $\boldsymbol{W}$ and its inverse $\boldsymbol{\Theta}$.
- For illustration purposes, we discuss the algorithm by focusing on the last row and column of the partitined matrices.

Glasso Algorithm

From KKT

$$\left[\begin{array}{c|c} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \hline \boldsymbol{w}_{12}^{'} & w_{22} \end{array}\right] - \left[\begin{array}{c|c} \boldsymbol{S}_{11} & \boldsymbol{s}_{12} \\ \hline \boldsymbol{s}_{12}^{'} & s_{22} \end{array}\right] - \lambda \left[\begin{array}{c|c} \boldsymbol{\Gamma}_{11} & \boldsymbol{\gamma}_{12} \\ \hline \boldsymbol{\gamma}_{12}^{'} & \gamma_{22} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0}^{'} & 0 \end{array}\right]$$

$$\mathbf{w_{12}} - \mathbf{s_{12}} - \lambda\boldsymbol{\gamma}_{\mathbf{12}} = \mathbf{0}. \tag{3}$$

$$\left[\begin{array}{c|c} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \hline \boldsymbol{w}_{12}^{'} & w_{22} \end{array}\right] \left[\begin{array}{c|c} \boldsymbol{\Theta}_{11} & \boldsymbol{\theta}_{12} \\ \hline \boldsymbol{\theta}_{12}^{'} & \theta_{22} \end{array}\right] = \left[\begin{array}{c|c} \boldsymbol{I} & \mathbf{0} \\ \hline \mathbf{0}^{'} & 1 \end{array}\right]$$

$$\mathbf{w_{12}} = -\mathbf{W_{11}} \frac{\boldsymbol{\theta_{12}}}{\theta_{22}} = \mathbf{W_{11}} \boldsymbol{\beta}, \qquad (4)$$

where $\boldsymbol{\beta} = -\boldsymbol{\theta}_{12}/\theta_{22}$

- After substituting (4) into (3), we obtain

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

  where we used the fact that $\boldsymbol{\beta}$ and $\boldsymbol{\theta_{12}}$ 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

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

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

Glasso Algorithm

input:
$\mathbf{S}, \lambda \leftarrow$ Sample covariance matrix and penalty parameter
top:
Initialize $\mathbf{W} = \mathbf{S} + \lambda \mathbf{I}$
Repeat for $j = 1, 2, \ldots, p$ until convergence
    (a) Solve the modified lasso problem (5)
    (b) Update $\mathbf{w_{12}} = \mathbf{W_{11}} \hat{\boldsymbol{\beta}}$
In the final cycle solve $\hat{\boldsymbol{\theta}}_{12} = -\hat{\boldsymbol{\beta}} \cdot \hat{\theta}_{22}$
Output:
$\boldsymbol{\Theta}, \mathbf{W}$

Tuning Parameter Selection

- In real-world applications, the value of penalty parameter λ is unknown and, traditionally, is treated as a tuning parameter to be selected from data.
- The value of λ is directly connected to the sparsity of $\boldsymbol{\Theta}$; i.e., the higher λ, the sparser is the inverse covariance matrix $\boldsymbol{\Theta}$.
- We discuss two popular methods for tuning parameter selection: **Cross-validation** and **eBIC**.

Cross-Validation

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

$$CV(\lambda) = \frac{1}{K} \sum_{\nu=1}^{K} \Big(- d_\nu \log |\hat{\boldsymbol{\Theta}} - \nu| + \sum_{I_\nu} y_i^t \hat{\boldsymbol{\Theta}}_{-\nu} y_i \Big), \tag{7}$$

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

eBIC

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

$$\text{eBIC}_\gamma = -n \log|\boldsymbol{\Theta}| + \text{tr}(\mathbf{S}\boldsymbol{\Theta}) + E \log n + 4E\gamma \log p, \tag{8}$$

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

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

- Positive γ 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.

Glasso Syntax

`glasso` *varlist* [*if*] [*in*] [, lambda(#) maxiter(#) tolerance(#) diag]

options	description
lambda(#)	Penalty parameter.
maxiter(#)	Maximum number of iteration.
tolerance(#)	Maximum tolerance for convergence.
diag	Should diagonal be penalized?

CVGlasso Syntax

```
cvglasso varlist [if] [in] [, lamlist(numlist) nlam(#) maxiter(#)
    tolerance(#) nfold(#) crit(string) gamma(#) diag]
```

options	description
lamlist(*numlist*)	Grid of positive tuning parameters for penalty term. If provided, causes to disregard `nlam`.
nlam(#)	Number of generated tuning parameters for penalty term.
maxiter(#)	Maximum number of iteration.
tolerance(#)	Maximum tolerance for convergence.
crit(*string*)	Type of the criterion. Possible options are *loglik* and *eBIC*.
gamma(#)	Activated if `crit` is *eBIC*.
diag	Should diagonal be penalized?

plotglasso Syntax

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

options	description
type(*string*)	Type of the plot: graph or matrix.
newlabs(*lab1 lab2*)	Labels for the plot.
nwplot_options	Options for undirected graph plot.
	For details see (Grund and Hedstrom 2021)
nwplotmatrix_options	Options for matrix plot.
	For details see(Grund and Hedstrom 2021)

Saved Results

`glasso` and `cvglasso` save the following in `r()`

Scalar

`r(lambda)`	Tuning parameter

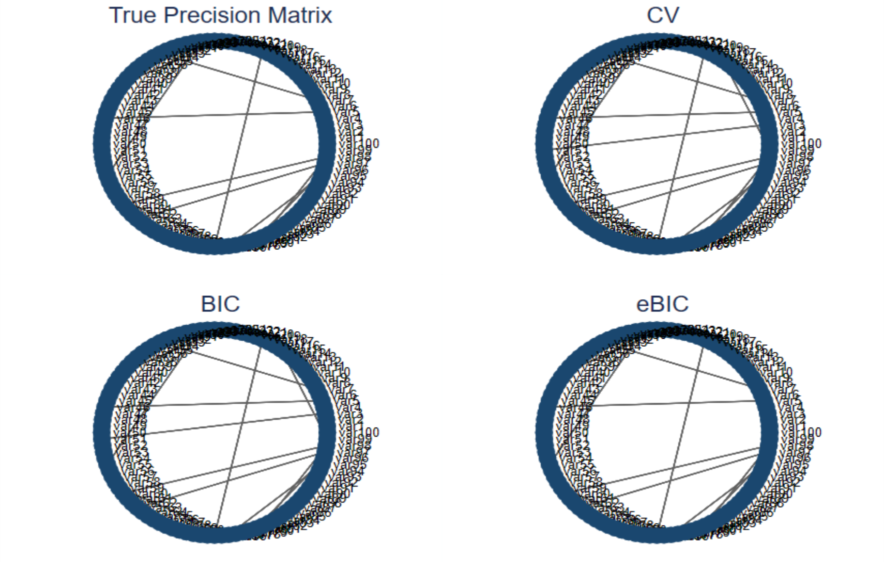
Matrix

`r(Omega)`	Inverse covariance matrix
`r(Sigma)`	Covariance matrix

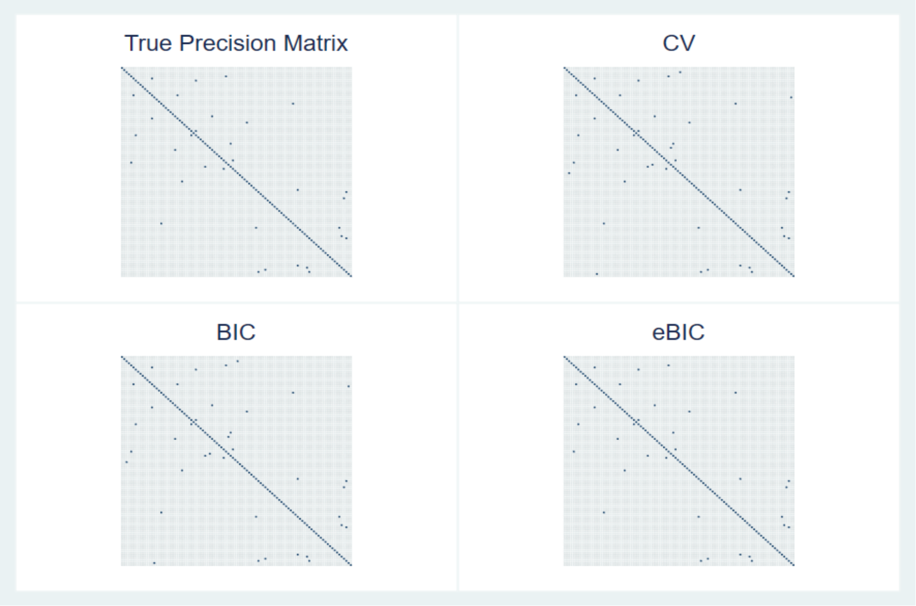
Simulation

- 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



Simulation result: Matrix



Simulation result: Metric

	CV	BIC	eBIC
TPR	0.71(0.26)	0.98(0.03)	0.99(0.02)
FPR	0.0001(0.00)	0.0001(0.00)	0.0001(0.00)
TDR	0.97(0.03)	0.83(0.10)	0.95(0.10)

Table: Averages of three metric over 20 simulated repetitions for the $n = 150$, $p = 100$ case.

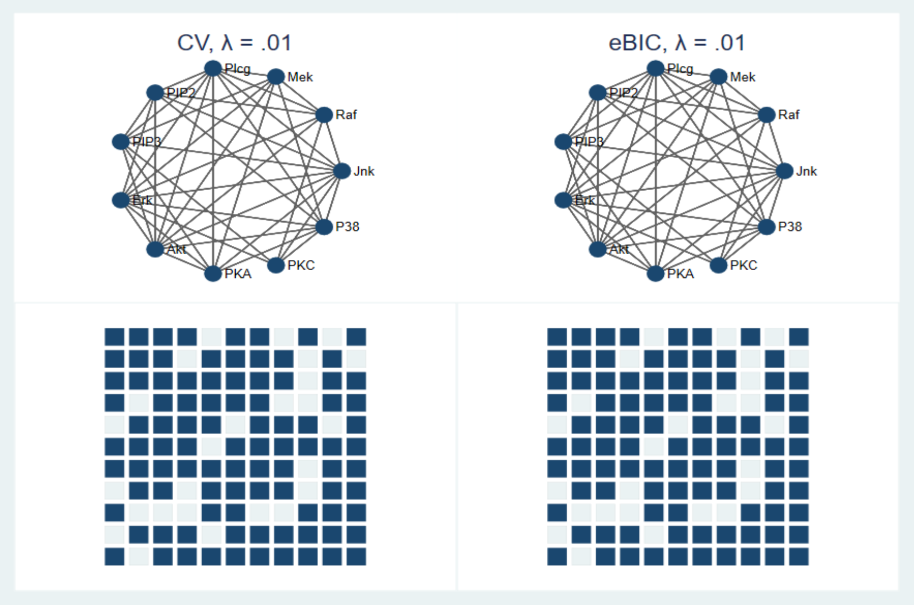
Flow-cytometry Data

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

Variable	Obs	Mean	Std. Dev.	Min	Max
Raf	7,466	6.09e-06	247.5281	-123.0719	4489.928
Mek	7,466	-.0000317	377.0562	-144.381	6959.619
Plcg	7,466	3.35e-06	173.8598	-53.85364	6153.146
PIP2	7,466	.0000198	299.3475	-150.1207	8906.88
PIP3	7,466	1.29e-06	43.04816	-26.03496	1247.965
Erk	7,466	2.16e-06	45.82672	-25.63119	2544.369
Akt	7,466	5.19e-06	137.7662	-80.16721	3473.833
PKA	7,466	-.0000444	644.4593	-624.7586	8270.241
PKC	7,466	-3.46e-06	92.87002	-29.34166	1580.658
P38	7,466	-8.18e-06	494.7688	-134.0145	7363.985
Jnk	7,466	-2.78e-06	215.6606	-72.2675	4666.732

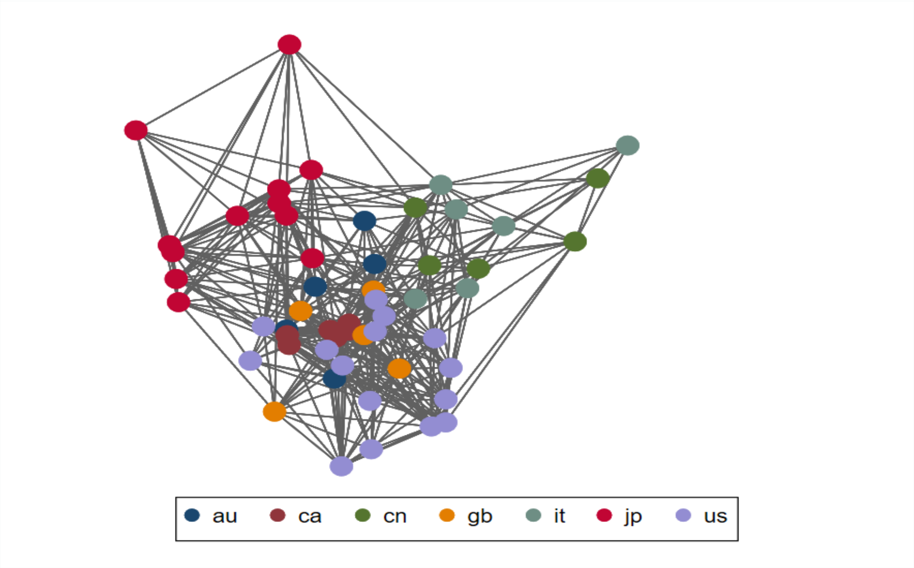
Flow-cytometry Data



Stock Return Volatility Data

- 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.

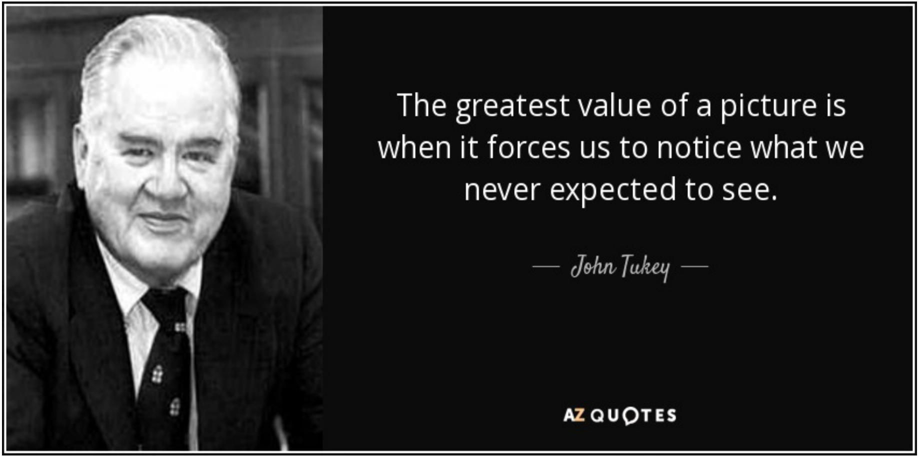
Stock Return Volatility Data



*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)


The greatest value of a picture is when it forces us to notice what we never expected to see.
— John Tukey —
AZ QUOTES