Gaussian Graphical models and Dependency networks

Sushmita Roy
sroy@biostat.wisc.edu
Computational Network Biology
Biostatistics & Medical Informatics 826
https://compnetbiocourse.discovery.wisc.edu

Sep 25th 2018
Plan for this section

- Overview of network inference (Sep 18\textsuperscript{th})
- Directed probabilistic graphical models
  Bayesian networks (Sep 18\textsuperscript{th}, Sep 20\textsuperscript{th})
- Gaussian graphical models (Sep 25\textsuperscript{th})
- Dependency networks (Sep 25, 27\textsuperscript{th})
- Integrating prior information for network inference (Oct 2\textsuperscript{nd}, 4\textsuperscript{th})
Goals for today

• Graphical Gaussian Models (GGMs)
• Different algorithms for learning GGMs
  – Graphical Lasso
  – Neighborhood selection
• Dependency networks
• GENIE3
• Evaluation of expression-based network inference methods
Recall the different types of probabilistic graphs

• In each graph type we can assert different conditional independencies
• Correlation networks
• Markov networks
  – Gaussian Graphical models
• Dependency networks
• Bayesian networks
Recall the univariate Gaussian distribution

Gaussian distribution

\[ f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \]

The Gaussian distribution is defined by two parameters:
Mean: \( \mu \)
Standard deviation: \( \sigma \)
A multi-variate Gaussian Distribution

• Extends the univariate distribution to higher dimensions ($p$ in our case)

$$P(x|\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)$$

• As in the univariate case, we have two parameters
  – Mean: a $p$-dimensional vector $\mu$
  – Co-variance: a $p \times p$ dimensional matrix $\Sigma$
    • Each entry of the matrix specifies the variance of co-variance between any two dimensions
A two-dimensional Gaussian distribution

- The mean $\mathbf{\mu} = [\mu_1, \mu_2]$

- The covariance matrix

$$
\Sigma = \begin{bmatrix}
\sigma_{11} & \sigma_{12} \\
\sigma_{21} & \sigma_{22}
\end{bmatrix}
$$

Probability density of a Gaussian with

$$
\mathbf{\mu} = [0, 0]
$$

$$
\Sigma = \begin{bmatrix}
0.25 & 0.3 \\
0.3 & 1
\end{bmatrix}
$$
Graphical Gaussian Models (GGMs)

• An undirected probabilistic graphical model
• Graph structure encode conditional independencies among variables
• The GGM assumes that $X$ is drawn from a $p$-variate Gaussian distribution with mean $\mu$ and co-variance $\Sigma$
• The graph structure specifies the zero pattern in the $\Sigma^{-1} = \Theta$
  – Zero entries in the inverse imply absence of an edge in the graph
Absence of edges and the zero-pattern of the precision matrix

\[
\Theta = \begin{bmatrix}
\theta_{11} & \theta_{12} & 0 & 0 & \theta_{15} \\
\theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} & 0 \\
0 & \theta_{32} & \theta_{33} & \theta_{34} & 0 \\
0 & \theta_{42} & \theta_{43} & \theta_{44} & \theta_{45} \\
\theta_{51} & 0 & 0 & \theta_{54} & \theta_{55}
\end{bmatrix}
\]

For example:

\[
X_1 \perp X_4 | X_2, X_5 \\
X_1 \perp X_3 | X_2, X_5
\]
Matrix trace and determinant properties

• Trace of a $p \times p$ square matrix $M$ is the sum of the diagonal elements

$$\text{Tr}(M) = \sum_{i}^{p} M_{ii}$$

• Trace of two matrices

$$\text{Tr}(MN) = \text{Tr}(NM)$$

• For a scalar $a$

$$\text{Tr}(a) = a$$

• Trace is additive

$$\text{Tr}(A + B) = \text{Tr}(A) + \text{Tr}(B)$$

• Determinant of inverse

$$\det(A^{-1}) = \frac{1}{\det(A)}$$
Joint probability of a sample from a GGM

• It is easier to work with the log

$$\log P(x|\mu, \Sigma) = \log \left(\frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}}\right) - \left(\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$$

$$\log P(x|\mu, \Sigma) = -\frac{1}{2} \log ((2\pi)^p |\Sigma|) - \left(\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$$

$$\propto -\frac{1}{2} \log |\Sigma| - \left(\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$$

$$= \frac{1}{2} \log |\Theta| - \left(\frac{1}{2} \text{Tr}((x - \mu)^T \Sigma^{-1}(x - \mu))\right)$$
Joint probability of a sample from a GGM (contd)

- The previous term can be re-written as

\[
= \frac{1}{2} \log |\Theta| - \left( \frac{1}{2} \text{Tr}((x - \mu)^T \Theta (x - \mu)) \right)
= \frac{1}{2} \log |\Theta| - \left( \frac{1}{2} \text{Tr}(\Theta (x - \mu)(x - \mu)^T) \right)
\]

\[
= \frac{1}{2} \log |\Theta| - \left( \frac{1}{2} \sum_{i=1}^{p} \theta_{ii} (x_i - \mu_i)^2 + \sum_{i \neq j} \theta_{ij} (x_i - \mu_i)(x_j - \mu_j) \right)
\]

This term is 0, when there is no contribution from the pair \(x_i, x_j\)

Trace trick: \(\text{Tr}(MN) = \text{Tr}(NM)\)
Data likelihood from a GGM

• Data likelihood of a dataset \( \mathcal{D} = \{x_1, \ldots, x_N\} \) with \( N \) different samples from a GGM is

\[
= \frac{1}{N} \sum_{j=1}^{N} \log P(x_j | \mu, \Sigma)
\]

• After some linear algebra is proportional to

\[
= \log |\Theta| - Tr(S\Theta)
\]

• where

\[
S = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T
\]

This formulation is nice because now we can think of entries of \( \Theta \) as regression weights that we need to maximize the above objective.
Learning a Graphical Gaussian Model

• Learning the structure of a GGM entails estimating which entries in the inverse of the covariance matrix are non-zero
• These correspond to the direct dependencies among two random variables
Learning a GGM

• Graphical Lasso
  – Exact approach
  – Friedman, Hastie and Tibshirani 2008
• Neighborhood selection
  – Approximate approach
  – Meinshausen and Buhlmann 2006
**Linear regression with $p$ predictors**

- Suppose we have $N$ samples of input output pairs
  $$\left\{ (x_1, y_1), \cdots, (x_N, y_N) \right\}$$
- Where $x_i = (x_{i1}, \cdots, x_{ip})$ is $p$-dimensional
- That is we have $p$ different features/predictors
- A linear regression model with $p$ features is
  $$y_i = \beta_0 + \sum_{j=1}^{p} x_{ij} \beta_j + \epsilon_i$$
  - intercept
  - Regression coefficients
- Learning the linear regression model requires us to find the parameters than minimizes prediction error
Linear regression with $p$ predictors

- Learning a regression model requires us find the regression weights that minimize the prediction error.

\[
\text{minimize}_{\beta_0,\beta_j} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right]
\]

Residual sum of squared errors (RSS)

- To find the $\beta = \{\beta_0, \beta_1, \cdots, \beta_p\}$ we would need to the RSS with respect to each parameters, set the derivative to 0 and solve

\[
\hat{\beta}_j = \frac{\sum_{i=1}^{N} (y_i - \beta_0) x_{ij}}{\sum_{i=1}^{N} x_{ij}^2}
\]

OLS estimate
Regularized regression

• The least squares solution is often not satisfactory
  – Prediction accuracy has high variance: small variations in the training set can result in very different answers
  – Interpretation is not easy: ideally, we would like to have a good predictive model, and that is interpretable

• The regularized regression framework can be generally described as follows:

\[
\text{minimize}_{\beta_0, \beta_i} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right] + \lambda f(\beta)
\]

Depending upon \( f \) we may have different types of regularized regression frameworks
Regularized regression

- $f(\beta)$ takes the form of some norm of $\beta$
- L1 norm used in LASSO regression:
  $$\sum_{j=1}^{p} |\beta_j|$$
- L2 norm used in Ridge regression:
  $$\sum_{j=1}^{p} \beta_j^2$$
Ridge regression

- The simplest type of regularized regression is called ridge regression
- This has the effect of smoothing out the regression weights

$$\text{minimize}_{\beta_0, \beta_j} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right] + \lambda \sum_{j=1}^{p} \beta_j^2$$

- It is often convenient to center the output (mean=0) and standardize the predictors (mean=0, variance =1)

$$\text{minimize}_{\beta_j} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right] + \lambda \sum_{j=1}^{p} \beta_j^2$$
LASSO regression

• The ridge regression handles the case of variance, and suitable when there are correlated predictors
• But does not give an interpretable model
• The LASSO regression model was developed to learn a sparse model

\[
\begin{align*}
&\text{minimize}_{\beta_0, \beta_j} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right] + \lambda \sum_{j=1}^{p} |\beta_j| \\
&\text{Or after standardization:} \\
&\text{minimize}_{\beta_j} \left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{p} x_{ij} \beta_j)^2 \right] + \lambda \sum_{j=1}^{p} |\beta_j|
\end{align*}
\]
Cyclic coordinate descent to learn LASSO regression weights

• To estimate the regression weights in LASSO, we cycle through each regression weight, setting it to its optimal value while keeping the others constant.

• That is we re-write the objective as

\[
\left[ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \sum_{k \neq j} x_{ik} \beta_k - x_{ij} \beta_j)^2 \right] + \lambda \sum_{k \neq j} |\beta_k| + \lambda |\beta_j|
\]

• We derive with respect to $\beta_j$ at a time, and set it to its optimal value.
Learning the regression weights in LASSO

• Due to the absolute value in the objective function, the derivative is not defined at 0
• That is derivative of |b| at b = 0 is not defined
• To address this, we need to consider the possible scenarios of the regression weight
Learning the regression weights in Lasso

• To handle the discontinuity in the L1 norm, we consider the possible scenarios of sign of

\[
\beta_j = \begin{cases} 
\frac{1}{N} \sum_{i=1}^{N} r_i x_{ij} - \lambda, & \text{if } \beta_j > 0 \\
\frac{1}{N} \sum_{i=1}^{N} r_i x_{ij} + \lambda, & \text{if } \beta_j < 0 \\
0, & \text{otherwise}
\end{cases}
\]

• Here \( r_i = y_i - \sum_{k \neq j} x_{ik} \beta_k \)

• Notice that the regularization term controls the extent to which \( \beta_j \) is pushed to 0.
Learning a GGM

• Graphical Lasso
  – Exact approach
  – Friedman, Hastie and Tibshirani 2008

• Neighborhood selection
  – Approximate approach
  – Meinshausen and Buhlmann 2006
Graphical LASSO

- Recall the Gaussian likelihood

\[
= \log |\Theta| - Tr(S\Theta)
\]

- Deriving with respect to \( \Theta \) we get a form that allows for a LASSO-like algorithm

- The algorithm itself uses LASSO to solve a regression problem per variable.
Graphical LASSO

- Recall the Gaussian likelihood
  \[ = \log |\Theta| - Tr(S\Theta) = \log \det(\Theta) - Tr(S\Theta) \]

- Learning the GGM requires us to solve the following optimization problem
  \[ \hat{\Theta} = \arg\max_{\Theta} \log \det(\Theta) - Tr(\Theta S) \]

- But this in general is not going to work because of small sample size
  \[ \hat{\Theta} = \arg\max_{\Theta} \log \det(\Theta) - Tr(\Theta S) - \lambda \|\Theta\|_1 \]

- This is the idea behind the Graphical LASSO algorithm

Friedman, Hastie, Tibshirani 2008
Graphical LASSO algorithm

• Deriving with respect to $\Theta$ we get

$$\Theta^{-1} - S - \lambda \Psi$$

• The algorithm itself uses a blockwise coordinate descent algorithm, each time considering one row and column

$$\Theta = \begin{bmatrix} \Theta_{11} & \theta_{12} \\ \theta_{12} & \theta_{22} \end{bmatrix} \quad S = \begin{bmatrix} S_{11} & S_{12} \\ S_{12} & S_{22} \end{bmatrix}$$

Keep this fixed
Graphical LASSO contd

• Using partitioned inverse

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}^{-1} = \begin{bmatrix}
(A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\
-D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1}
\end{bmatrix}.
\]

\[
\begin{bmatrix}
\Theta_{11} & \theta_{12} \\
\theta_{12} & \theta_{22}
\end{bmatrix}^{-1} = \begin{bmatrix}
W_{11} & -W_{11} \theta_{12}/\theta_{22} \\
w_{12} & w_{22}
\end{bmatrix}
\]

• Plugging this in

\[
\Theta^{-1} = S - \lambda \Psi
\]

• For each row/column we get

\[
W_{11} \beta - s_{12} + \lambda \psi_{12} = 0,
\]

where \( \beta = -\theta_{12}/\theta_{22} \).
Graphical LASSO contd

- This specific function looks similar to the derivative of a LASSO objective

\[
\frac{1}{2N} \left( y - Z\beta \right)^T \left( y - Z\beta \right) + \lambda \|\beta\|_1
\]

Derivative

\[
\frac{1}{N} Z^T Z\beta - \frac{1}{N} Z^T y + \lambda \text{sign}(\beta) = 0
\]

\[
W_{11}\beta - s_{12} + \lambda \psi_{12} = 0,
\]

where \( \beta = -\theta_{12}/\theta_{22} \)
Graphical LASSO

• Let $W$ be the current estimate of the inverse
• Repeat for each $j^{th}$ row and column
  – Partition $W$ into the two parts,
    • $w_{12}$: associated the $j^{th}$ row and column, and
    • $W_{11}$: for the rest
  – Solve the LASSO regression problem for the $j^{th}$ to estimate $\beta$
  – Update $w_{12} = W_{11}\beta$
Neighborhood selection

• Proposed by Meinshausen and Buhlmann 2006
• Markov blanket: The immediate neighborhood of a random variable
• Key idea: Find the Markov blanket or immediate neighbor set of each random variable
Neighborhood selection

- Here also we solve a set of regression problems for each random variable \( X_s \)

\[
\frac{1}{2N} \sum_{i=1}^{N} (x_{is} - \sum_{j \neq s} x_{ij} \beta_{sj})^2 + \lambda \| \beta_s \|_1
\]

- The Markov blanket/neighborhood are those variables that have a non-zero coefficient

- Combine the neighborhood estimates using an AND or OR rule to create an undirected graph
Comparison between the two algorithms

• Neighborhood selection is fast compared to Graphical LASSO

• Neighborhood selection requires a “correction” to learn a valid structure, but this is not needed in Graphical LASSO