# A Generic Coordinate Descent Algorithm for Inverse Covariance Estimation

Junhui Wang

Department of Mathematics, Statistics,
and Computer Science
University of Illinois at Chicago

December, 2011

# Outline

- Brief review of inverse covariance estimation
- Idea of the generic coordinate descent algorithm
- Estimation of the inverse covariance matrix
- Simulation and real examples
- Extension to graph clustering
- Summary

# Inverse covariance matrix

- Assume $X = (X_1, \ldots, X_p)^T \sim N_p(0, \Sigma_0)$, the goal is to estimate the precision matrix $\Omega_0 = \Sigma_0^{-1}$.

# Inverse covariance matrix

- Assume $X = (X_1, \ldots, X_p)^T \sim N_p(0, \Sigma_0)$, the goal is to estimate the precision matrix $\Omega_0 = \Sigma_0^{-1}$.
- Connection with Gaussian graphical models (Edwards, 2000): the dependence structure among $X_j$'s can be fully determined by $\Omega_0$, since

$$(\Omega_0)_{jk} = 0 \Longleftrightarrow X_j \perp\!\!\!\perp X_k \mid \text{other variables.}$$

# Inverse covariance matrix

- Assume $X = (X_1, \ldots, X_p)^T \sim N_p(0, \Sigma_0)$, the goal is to estimate the precision matrix $\Omega_0 = \Sigma_0^{-1}$.
- Connection with Gaussian graphical models (Edwards, 2000): the dependence structure among $X_j$'s can be fully determined by $\Omega_0$, since

$$(\Omega_0)_{jk} = 0 \Longleftrightarrow X_j \perp\!\!\!\perp X_k \mid \text{other variables.}$$

- Can improve learning performance (Rothman et al., 2008).

# Existing estimation methods

Assume $X_{(1)}, \ldots, X_{(n)}$ are i.i.d. from $N_p(0, \Sigma_0)$.

- The negative log-likelihood is, after dropping some constants,

$$l(\Omega) = \text{tr}(\Omega S) - \log |\Omega|,$$

where $S = n^{-1} \sum_{i=1}^{n} \|X_{(i)} - \bar{X}\|^2$.

# Existing estimation methods

Assume $X_{(1)}, \ldots, X_{(n)}$ are i.i.d. from $N_p(0, \Sigma_0)$.

- The negative log-likelihood is, after dropping some constants,

$$l(\Omega) = \mathrm{tr}(\Omega S) - \log |\Omega|,$$

  where $S = n^{-1} \sum_{i=1}^{n} \|X_{(i)} - \bar{X}\|^2$.
- $l(\Omega)$ is strictly convex in $\Omega$ and its first derivative is

$$l'(\Omega) = S - \Omega^{-1}.$$

# Existing estimation methods

Assume $X_{(1)}, \ldots, X_{(n)}$ are i.i.d. from $N_p(0, \Sigma_0)$.

- The negative log-likelihood is, after dropping some constants,

$$l(\Omega) = \text{tr}(\Omega S) - \log |\Omega|,$$

  where $S = n^{-1} \sum_{i=1}^{n} \|X_{(i)} - \bar{X}\|^2$.
- $l(\Omega)$ is strictly convex in $\Omega$ and its first derivative is

$$l'(\Omega) = S - \Omega^{-1}.$$

- If $S$ is p.d., the mle of $\Omega$ is $S^{-1}$.

# Existing estimation methods

- $S$ may be semi p.d.; or $S^{-1}$ can be dense and suboptimal when $\Omega_0$ is assumed to be sparse.

# Existing estimation methods

- $S$ may be semi p.d.; or $S^{-1}$ can be dense and suboptimal when $\Omega_0$ is assumed to be sparse.
- Regularized log-likelihood function,

$$\min_{\Omega \text{ p.d.}} \; l_\lambda(\Omega) = \text{tr}(\Omega S) - \log|\Omega| + \lambda\|\Omega^-\|_1,$$

where $\Omega^- = \Omega - \Omega^+$ with $\Omega^+ = \text{diag}(\Omega)$, and $\|\cdot\|_1$ is componentwise $L_1$-norm.

# Existing estimation methods

- $S$ may be semi p.d.; or $S^{-1}$ can be dense and suboptimal when $\Omega_0$ is assumed to be sparse.
- Regularized log-likelihood function,

$$\min_{\Omega \text{ p.d.}} \; l_\lambda(\Omega) = \text{tr}(\Omega S) - \log|\Omega| + \lambda\|\Omega^-\|_1,$$

  where $\Omega^- = \Omega - \Omega^+$ with $\Omega^+ = \text{diag}(\Omega)$, and $\|\cdot\|_1$ is componentwise $L_1$-norm.
- Some existing estimation methods:
  - Graphical Lasso (glasso; Friedman et al., 2007)
  - Cholesky decomposition (SPICE; Rothman et al., 2008)

# A generic coordinate descent (CD) algorithm

Let $s(\Omega)$ be any strictly convex function of $\Omega$,

$$\min_{\Omega \text{ p.d.}} \; s(\Omega).$$

# A generic coordinate descent (CD) algorithm

Let $s(\Omega)$ be any strictly convex function of $\Omega$,

$$\min_{\Omega \text{ p.d.}} s(\Omega).$$

The key of the CD algorithm is to update the current $\widehat{\Omega}_t$ by one diagonal entry or two symmetric off-diagonal entries,

$$\widehat{\Omega}_{t+1} = \widehat{\Omega}_t - v_t W_t,$$

where $W_t$ is the CD direction and $v_t$ is the step size.

# The CD algorithm assuring p.d.

- Let $D_t = s'(\widehat{\Omega}_t)$, $(a, b) = \text{argmax}_{j,k} |(D_t)_{jk}|$, and then set

$$(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}, \text{ and } (W_t)_{jk} = 0 \text{ otherwise.}$$

# The CD algorithm assuring p.d.

- Let $D_t = s'(\widehat{\Omega}_t)$, $(a, b) = \text{argmax}_{j,k} |(D_t)_{jk}|$, and then set

$$(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}, \text{ and } (W_t)_{jk} = 0 \text{ otherwise.}$$

- Set $v_t$ as follows.

**Theorem**

*Given that $\widehat{\Omega}_t$ is p.d., $\widehat{\Omega}_{t+1} = \widehat{\Omega}_t - v_t W_t$ is p.d. if and only if $\det(\widehat{\Omega}_{t+1}) > 0$.*

# The CD algorithm assuring p.d.

- Let $D_t = s'(\widehat{\Omega}_t)$, $(a, b) = \text{argmax}_{j,k} |(D_t)_{jk}|$, and then set

$$(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}, \text{ and } (W_t)_{jk} = 0 \text{ otherwise.}$$

- Set $v_t$ as follows.

**Theorem**

*Given that* $\widehat{\Omega}_t$ *is p.d.,* $\widehat{\Omega}_{t+1} = \widehat{\Omega}_t - v_t W_t$ *is p.d. if and only if* $\det(\widehat{\Omega}_{t+1}) > 0$. *In addition,* $\det(\widehat{\Omega}_{t+1}) > 0$ *when*

$$v_t < v_t^* = \begin{cases} \dfrac{-(D_t)_{ab}(\widehat{\Omega}_t^{-1})_{ab} + |(D_t)_{ab}| \sqrt{(\widehat{\Omega}_t^{-1})_{aa}(\widehat{\Omega}_t^{-1})_{bb}}}{(D_t)_{ab}^2 \Delta_t}, & \text{if } a \neq b; \\ \dfrac{1}{|(D_t)_{ab}|(\widehat{\Omega}_t^{-1})_{aa}}, & \text{if } a = b, \end{cases}$$

*where* $\Delta_t = (\widehat{\Omega}_t^{-1})_{aa}(\widehat{\Omega}_t^{-1})_{bb} - (\widehat{\Omega}_t^{-1})_{ab}^2$.

# Precision matrix estimation using $l_\lambda(\Omega)$

- Set $s(\Omega) = l_\lambda(\Omega)$.
- Initialize $\widehat{\Omega}_0 = (\text{diag}(S))^{-1}$.
- $D_t = l'(\widehat{\Omega}_t) = S - \widehat{\Omega}_t^{-1} + \lambda \,\text{sign}(\widehat{\Omega}_t^-)$ and $W_t$ is defined as in the last slide.
- $v_t = \alpha \cdot \text{argmin}_{v \leq v_t^*} \, l_\lambda(\widehat{\Omega}_t - vW_t)$ with $0 < \alpha < 1$.

# Precision matrix estimation using $l_\lambda(\Omega)$

- Set $s(\Omega) = l_\lambda(\Omega)$.
- Initialize $\widehat{\Omega}_0 = (\text{diag}(S))^{-1}$.
- $D_t = l'(\widehat{\Omega}_t) = S - \widehat{\Omega}_t^{-1} + \lambda \,\text{sign}(\widehat{\Omega}_t^-)$ and $W_t$ is defined as in the last slide.
- $v_t = \alpha \cdot \text{argmin}_{v \le v_t^*} l_\lambda(\widehat{\Omega}_t - vW_t)$ with $0 < \alpha < 1$.

Some remarks:

- Requires line search for finding $v_t$;
- Needs to re-run the iteration for different $\lambda$'s;

# Precision matrix estimation using $l(\Omega)$

- Set $s(\Omega) = l(\Omega)$.
- Initialize $\widehat{\Omega}_0 = (\text{diag}(S))^{-1}$.
- $D_t = l'(\widehat{\Omega}_t) = S - \widehat{\Omega}_t^{-1}$ and $W_t$ is defined as in the last slide.
- $v_t = \alpha \cdot \text{argmin}_v \, l(\widehat{\Omega}_t - vW_t)$ with $0 < \alpha \leq 1$,

# Precision matrix estimation using $l(\Omega)$

- Set $s(\Omega) = l(\Omega)$.
- Initialize $\widehat{\Omega}_0 = (\text{diag}(S))^{-1}$.
- $D_t = l'(\widehat{\Omega}_t) = S - \widehat{\Omega}_t^{-1}$ and $W_t$ is defined as in the last slide.
- $v_t = \alpha \cdot \text{argmin}_v \, l(\widehat{\Omega}_t - vW_t)$ with $0 < \alpha \leq 1$, which has analytic solution,
  - if $a = b$, $v_t = (S_{aa}(\widehat{\Omega}_t^{-1})_{aa})^{-1}$;
  - if $a \neq b$ and $S_{ab} = 0$, $v_t = \Delta_t^{-1}$;
  - if $a \neq b$ and $S_{ab} \neq 0$,

$$v_t = \frac{-(\Delta_t + 2(\widehat{\Omega}_t^{-1})_{ab} S_{ab}) + \sqrt{\Delta_t^2 + 4S_{ab}^2 \Delta_t + 4S_{ab}^2((\widehat{\Omega}_t^{-1})_{ab})^2}}{2\Delta_t (D_t)_{ab} S_{ab}}.$$

# Some remarks

- $v_t < v_t^*$, and thus $\widehat{\Omega}_{t+1}$ is always p.d..

# Some remarks

- $v_t < v_t^*$, and thus $\widehat{\Omega}_{t+1}$ is always p.d..
- The algorithm generates a p.d. solution path of $\widehat{\Omega}$, that starts from the diagonal matrix and gradually converges to a dense matrix.

# Some remarks

- $v_t < v_t^*$, and thus $\widehat{\Omega}_{t+1}$ is always p.d..
- The algorithm generates a p.d. solution path of $\widehat{\Omega}$, that starts from the diagonal matrix and gradually converges to a dense matrix.
- The sparse $\widehat{\Omega}$ can be obtained by early stopping the iteration.
- Any model selection criterion can be used as the stopping rule, such as

$$\begin{aligned} \text{AIC}(\Omega) &= l(\Omega) + \frac{2}{n} \cdot \text{df}(\Omega), \\ \text{BIC}(\Omega) &= l(\Omega) + \frac{\log(n)}{n} \cdot \text{df}(\Omega), \end{aligned}$$

where $\text{df}(\Omega) = \#\{(j,k) : j < k, \Omega_{jk} \neq 0\}$.

# Simulation setup

Four covariance structures are considered:

- Model 1 (AR(1)): $(\Sigma_0)_{jk} = \rho^{|j-k|}$ with $\rho = 0.5$, and $\Omega_0 = (\Sigma_0)^{-1}$;
- Model 2 (AR(3)): $(\Omega_0)_{jk} = I(|j-k| = 0) + 0.5I(|j-k| = 1) + 0.2I(|j-k| = 2) + 0.1I(|j-k| = 3)$;
- Models 3 & 4 (Randomly generated matrix): $(\Omega_0)_{jk} \sim 0.5 \cdot \text{Bern}(\gamma)$ when $j \neq k$, with $\gamma = 0.1$ for Model 3 and $\gamma = 0.5$ for Model 4, and $(\Omega_0)_{jj}$'s are set so that the smallest eigenvalue of $\Omega_0$ is 0.1.

Sample size $n = 80$, and dimension $p = 25$, $50$, or $100$.

# Various performance measures

- Kullback-Leibler loss:

$$KL(\Omega_0, \widehat{\Omega}_M) = \text{tr}(\Sigma_0 \widehat{\Omega}_M) - \log |\Sigma_0 \widehat{\Omega}_M| - p;$$

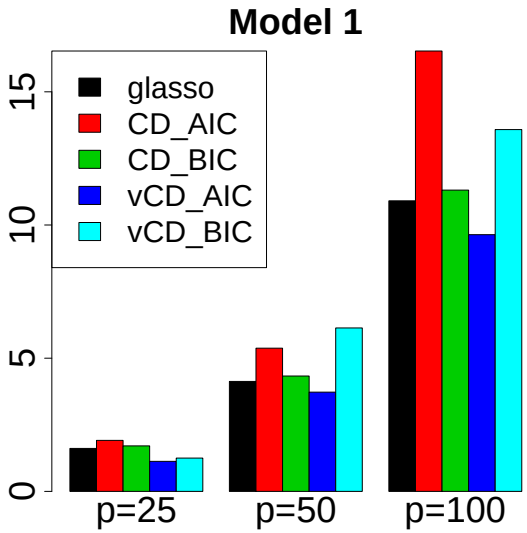
- Frobenius norm:

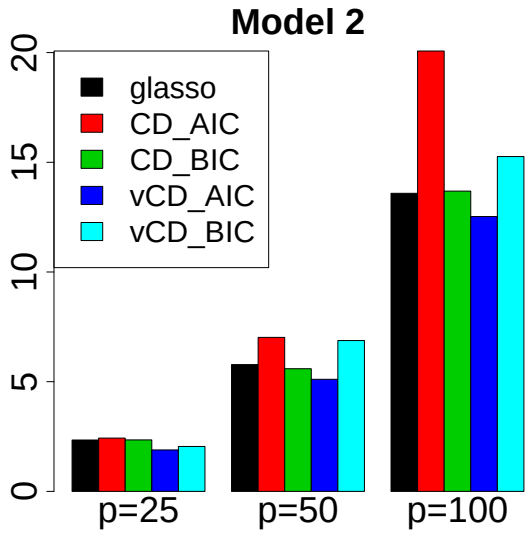
$$F(\Omega_0, \widehat{\Omega}_M) = \|\Omega_0 - \widehat{\Omega}_M\|_F;$$

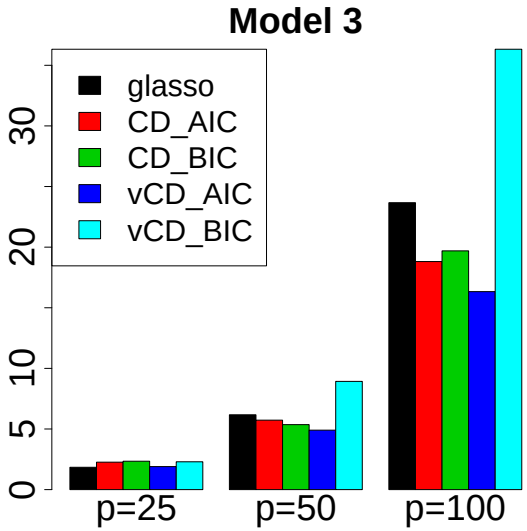
- Variable selection loss (Ravikumar et al., 2011):

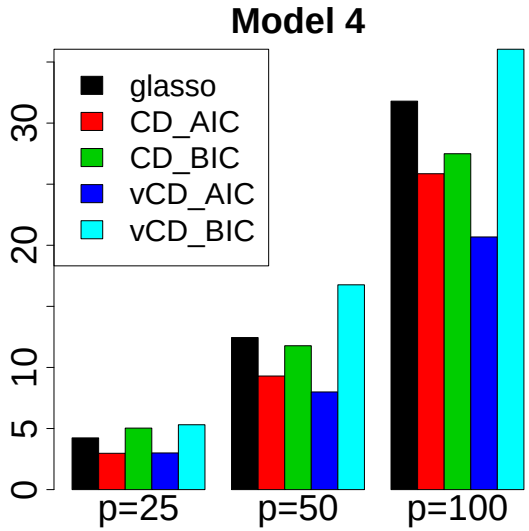
$$VS(\Omega_0, \widehat{\Omega}_M) = (p(p-1))^{-1} \sum_{j \neq k} I(\text{sign}((\Omega_0)_{jk}) \neq \text{sign}((\widehat{\Omega}_M)_{jk})).$$

# Simulation: Kullback-Leibler loss

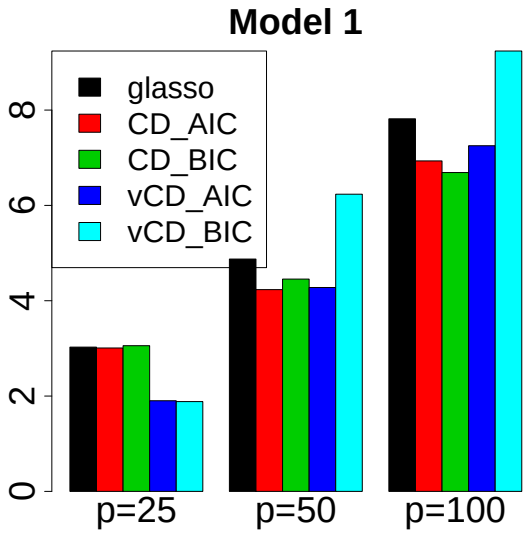


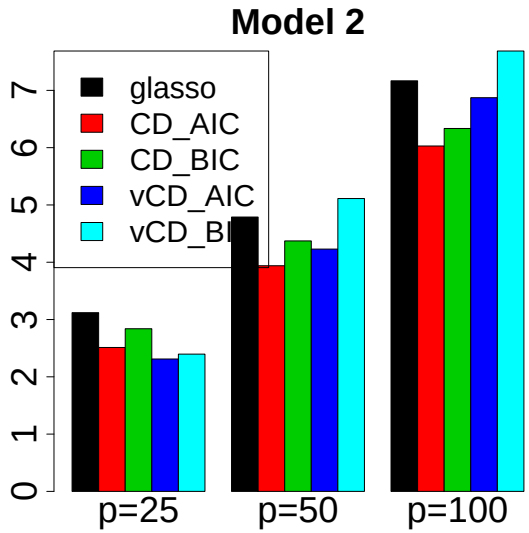



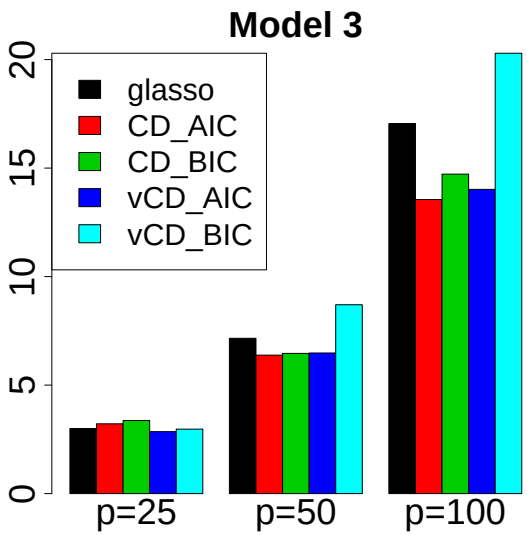



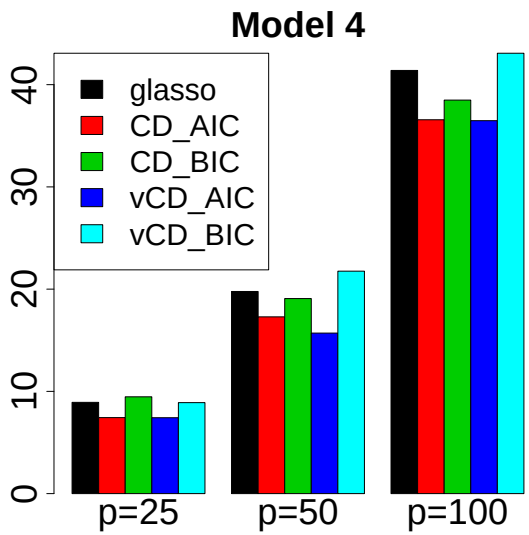



# Simulation: Frobenius norm

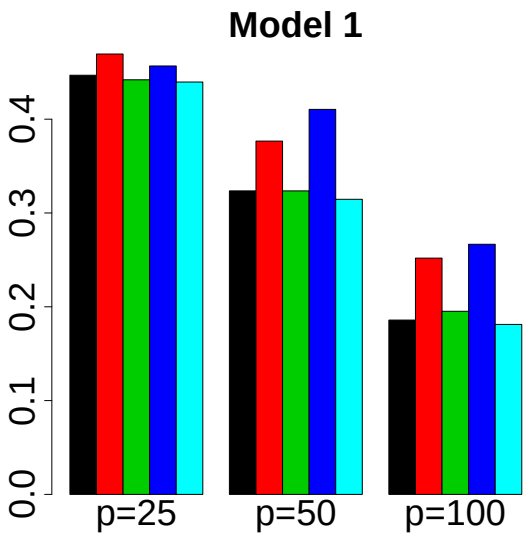


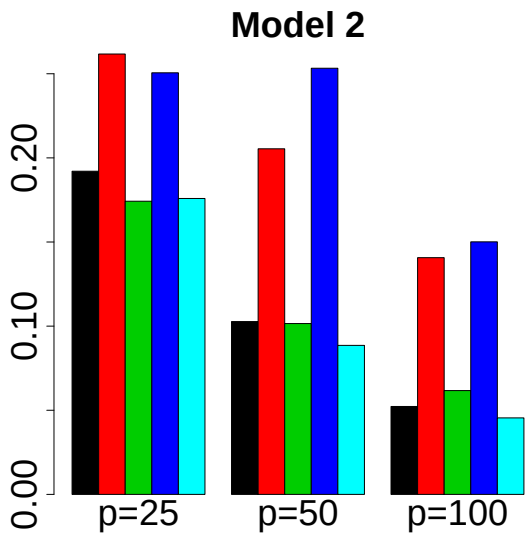



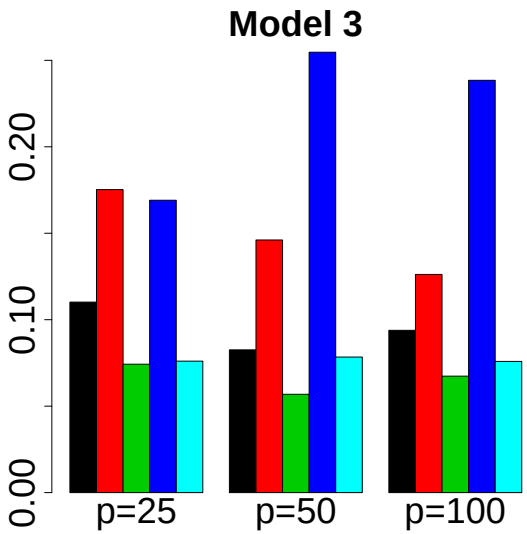



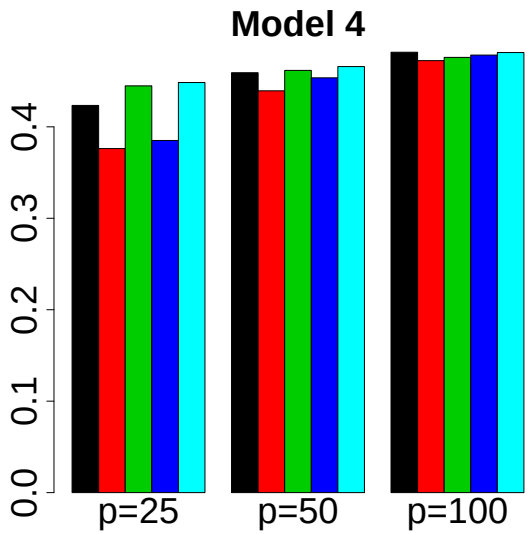



# Simulation: Variable selection loss



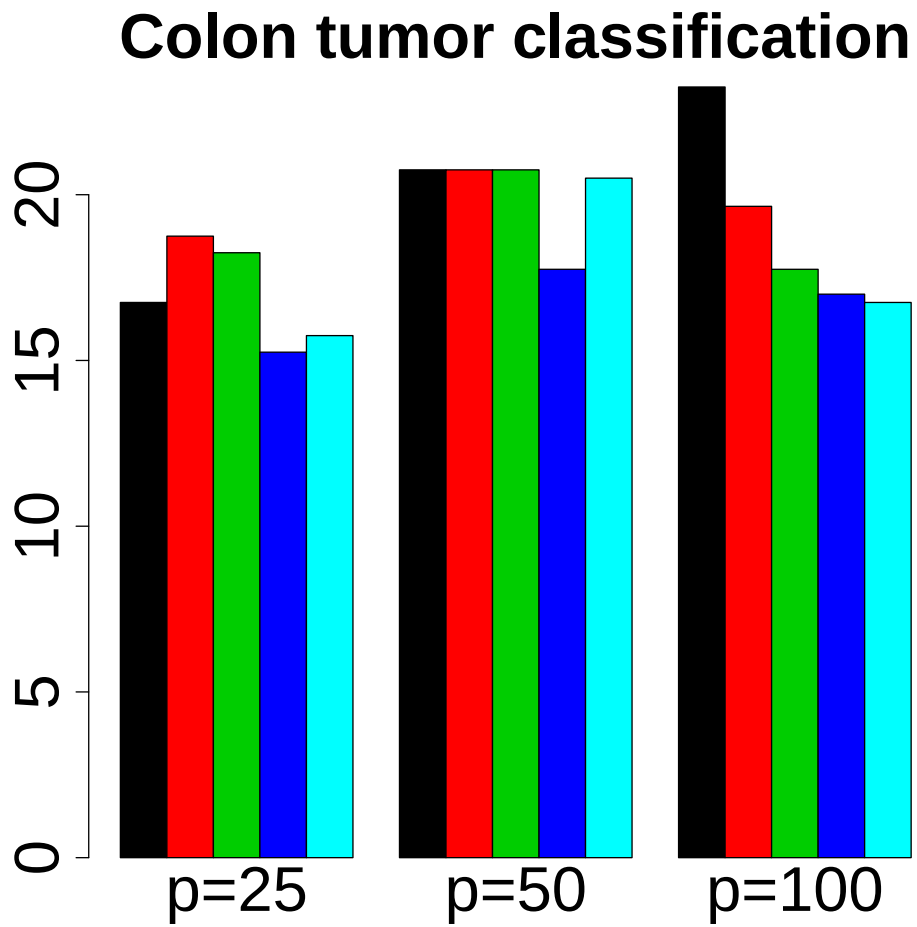



# Colon tumor classification

- 62 colon adenocarcinoma tissue samples are gathered, where 40 are tumor tissues and 22 are non-tumor tissues.
- 2,000 gene expression profiles are available for each tissue.
- 42 tissues are randomly selected for training and the remaining 20 tissues are for testing.
- $p = 25$, $50$ or $100$ most significant genes are selected for illustration.
- LDA is employed for classification, with estimated precision matrices.

# Colon tumor classification: misclassification error



# Extension to graph clustering

- Graph clustering aims to group the vertices (variables) into clusters so that the vertices in the same cluster are well connected (similar) and the vertices between clusters are not.

# Extension to graph clustering

- Graph clustering aims to group the vertices (variables) into clusters so that the vertices in the same cluster are well connected (similar) and the vertices between clusters are not.
- Assume the variables follow a p-variate Gaussian distribution, then the graph clustering problem becomes solving

$$\min_{\Omega} \ l(\Omega)$$

subject to $\Omega$ is p.d. and *block diagonal*.

# Extension to graph clustering

- Graph clustering aims to group the vertices (variables) into clusters so that the vertices in the same cluster are well connected (similar) and the vertices between clusters are not.
- Assume the variables follow a p-variate Gaussian distribution, then the graph clustering problem becomes solving

$$\min_{\Omega} \; l(\Omega)$$

  subject to $\Omega$ is p.d. and *block diagonal*.
- Each block corresponds to a cluster.

# Graph clustering via CD algorithm

At step t, suppose $\widehat{\Omega}_t = \mathrm{diag}\{K_1, \ldots, K_m\}$.

(i) Let $D_t = S - \widehat{\Omega}_t^{-1}$, then

$$(a, b) = \underset{j,k}{\mathrm{argmax}} \ \frac{\|(D_t)_{jk}\|_1}{\mathrm{dim}(K_j)\mathrm{dim}(K_k)},$$

where $(D_t)_{jk}$ is the $(j, k)$-th block of $D_t$.

# Graph clustering via CD algorithm

At step t, suppose $\widehat{\Omega}_t = \text{diag}\{K_1, \ldots, K_m\}$.

(i) Let $D_t = S - \widehat{\Omega}_t^{-1}$, then

$$(a, b) = \underset{j,k}{\text{argmax}} \ \frac{\|(D_t)_{jk}\|_1}{\dim(K_j)\dim(K_k)},$$

where $(D_t)_{jk}$ is the $(j, k)$-th block of $D_t$.

(ii) Set $(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}$, and $(W_t)_{jk} = 0$ otherwise.

# Graph clustering via CD algorithm

At step t, suppose $\widehat{\Omega}_t = \text{diag}\{K_1, \ldots, K_m\}$.

(i) Let $D_t = S - \widehat{\Omega}_t^{-1}$, then

$$(a, b) = \underset{j,k}{\text{argmax}} \ \frac{\|(D_t)_{jk}\|_1}{\text{dim}(K_j)\text{dim}(K_k)},$$

where $(D_t)_{jk}$ is the $(j, k)$-th block of $D_t$.

(ii) Set $(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}$, and $(W_t)_{jk} = 0$ otherwise.

(iii) Set $v_t = \alpha \cdot \left(\phi_{max}(K_a^{-1} D_t K_b^{-1} D_t^T)\right)^{-1/2}$ and $\phi_{max}(\cdot)$ being the largest eigenvalue, and $\widehat{\Omega}_{t+1} = \widehat{\Omega}_t - v_t W_t$.

# Graph clustering via CD algorithm

At step t, suppose $\widehat{\Omega}_t = \text{diag}\{K_1, \ldots, K_m\}$.

(i) Let $D_t = S - \widehat{\Omega}_t^{-1}$, then

$$(a, b) = \underset{j,k}{\text{argmax}} \ \frac{\|(D_t)_{jk}\|_1}{\dim(K_j)\dim(K_k)},$$

where $(D_t)_{jk}$ is the $(j, k)$-th block of $D_t$.

(ii) Set $(W_t)_{ab} = (W_t)_{ba} = (D_t)_{ab}$, and $(W_t)_{jk} = 0$ otherwise.

(iii) Set $v_t = \alpha \cdot \left(\phi_{max}(K_a^{-1} D_t K_b^{-1} D_t^T)\right)^{-1/2}$ and $\phi_{max}(\cdot)$ being the largest eigenvalue, and $\widehat{\Omega}_{t+1} = \widehat{\Omega}_t - v_t W_t$.

(iv) Reorganize $\Omega_{t+1}$ by combining the $K_a$ and $K_b$.

# Some remarks:

- The algorithm always converges and all $\widehat{\Omega}_t$'s are p.d. and block diagonal.

# Some remarks:

- The algorithm always converges and all $\widehat{\Omega}_t$'s are p.d. and block diagonal.
- Similar to agglomerative hierarchical clustering, with a number of differences.

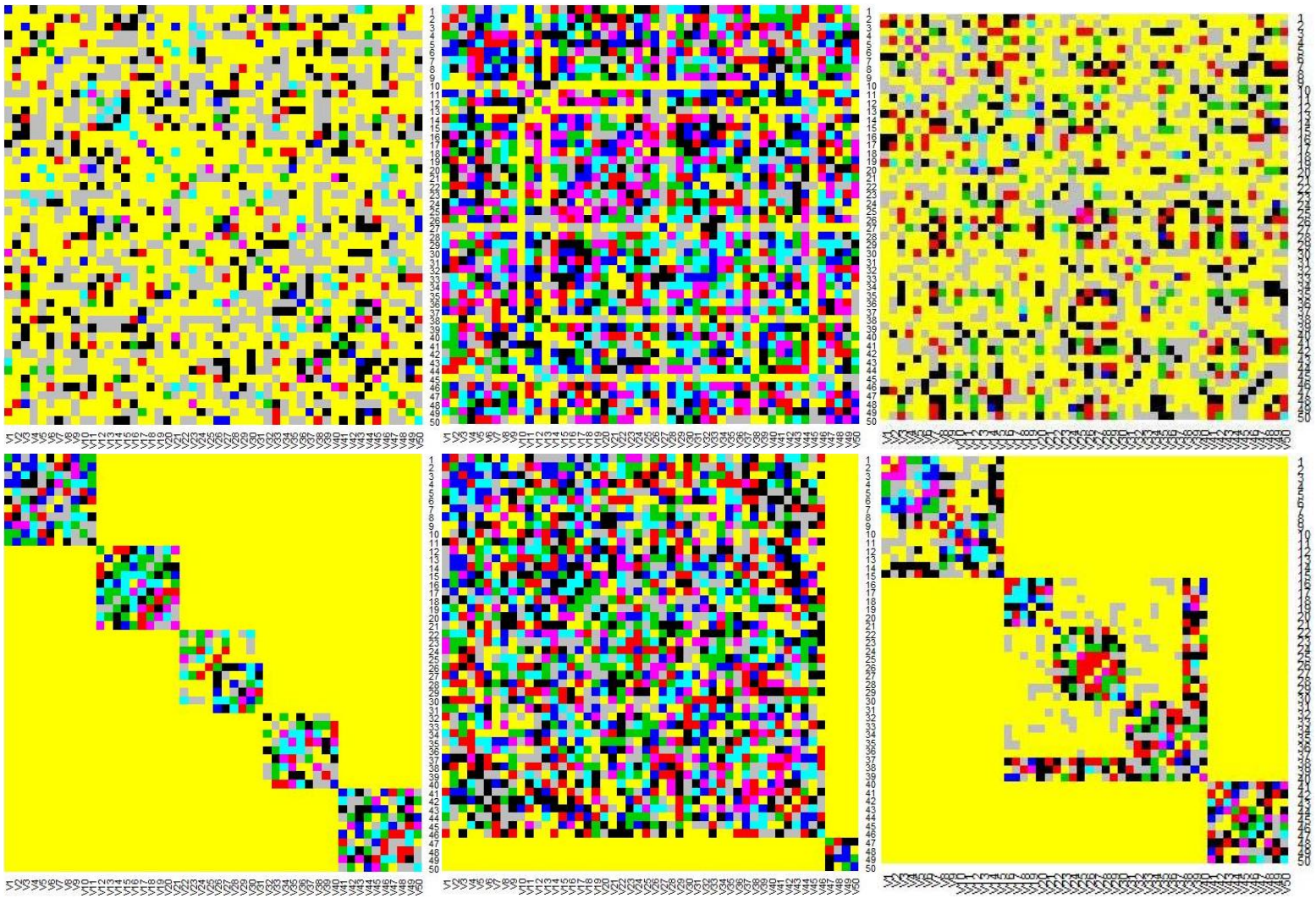
# Some remarks:

- The algorithm always converges and all $\widehat{\Omega}_t$'s are p.d. and block diagonal.
- Similar to agglomerative hierarchical clustering, with a number of differences.
- The formulation of finding $(a, b)$ is analogous to the group average, and the single linkage and the complete linkage can be defined accordingly.

# Some remarks:

- The algorithm always converges and all $\widehat{\Omega}_t$'s are p.d. and block diagonal.
- Similar to agglomerative hierarchical clustering, with a number of differences.
- The formulation of finding $(a, b)$ is analogous to the group average, and the single linkage and the complete linkage can be defined accordingly.
- The number of blocks (clusters) can be pre-specified or selected via the model selection criteria.

# Illustrative examples: heatmaps



# Illustrative examples: clustering error

Clustering error (Wang, 2010) is defined as

$$err(\hat{\psi}) = \frac{\#\Big\{\{\psi_0(x_j) = \psi_0(x_k)\}\Delta\{\hat{\psi}(x_j) = \hat{\psi}(x_k)\}\Big\}}{p(p-1)},$$

where $\psi_0$ and $\hat{\psi}$ are the true and the estimated clustering mappings, and $\Delta$ is the symmetric set difference.

# Illustrative examples: clustering error

Clustering error (Wang, 2010) is defined as

$$err(\hat{\psi}) = \frac{\#\Big\{\{\psi_0(x_j) = \psi_0(x_k)\}\Delta\{\hat{\psi}(x_j) = \hat{\psi}(x_k)\}\Big\}}{p(p-1)},$$

where $\psi_0$ and $\hat{\psi}$ are the true and the estimated clustering mappings, and $\Delta$ is the symmetric set difference.

Averaged clustering errors over 50 replications:

| | $\alpha = .2$ | $\alpha = .5$ | hclust |
|---|---|---|---|
| Model 1 | 5.10(.093) | 5.59(.109) | 19.98(2.695) |
| Model 2 | 7.00(.181) | 6.24(.141) | 26.47(6.477) |
| Model 3 | 7.84(.223) | 6.34(.127) | 22.53(4.896) |

# Some take-home message

- A generic coordinate descent algorithm is introduced to optimize any strictly convex function with respect to positive definite matrices.

# Some take-home message

- A generic coordinate descent algorithm is introduced to optimize any strictly convex function with respect to positive definite matrices.
- The algorithm is successfully applied to precision matrix estimation and graph clustering.

# Some take-home message

- A generic coordinate descent algorithm is introduced to optimize any strictly convex function with respect to positive definite matrices.
- The algorithm is successfully applied to precision matrix estimation and graph clustering.
- The algorithm can be extended to other problems such as covariance matrix estimation and metric learning.

# Some take-home message

- A generic coordinate descent algorithm is introduced to optimize any strictly convex function with respect to positive definite matrices.
- The algorithm is successfully applied to precision matrix estimation and graph clustering.
- The algorithm can be extended to other problems such as covariance matrix estimation and metric learning.

Thank you!