10-708: Probabilistic Graphical Models, Spring 2015

27: Case study with popular GM III

Lecturer: Eric P. Xing *Scribes: Hyun Ah Song & Elizabeth Silver*

1 Introduction: Gene association mapping for complex diseases[1]

This class covers an example of how to use graphs to construct a *regression* model (as opposed to a probabilistic model). The tools of graphical models are used to translate complex data and complex queries into a regression method.

1.1 Biological motivation

We would like to know what genetic variations cause diseases. The naive view of this process is that one genetic variation (e.g. a Single Nucleotide Polymorphism (SNP), or copy number variation, etc.) causes one disease. However, most disease processes are much more complex. For example, flu susceptibility is influenced by many genes (as well as the environment). Furthermore, many diseases are not truly binary variables – they may consist of a cluster of traits (e.g. coronary heart disease) or occur on a gradient (e.g. there's no sharp cutoff for how much insulin resistance counts as Type II diabetes). The biological phenomenon of one gene causing multiple traits is known as "pleiotropy". Multiple genes influencing a single trait, such that one gean may mask the effect of another, is known as "epistasis".

Some SNPs make us sick – we can refer to these as the "causal SNPs". We would expect that the SNPs most closely associated with a disease are the causal SNPs. However, SNPs may be associated with each other due to linkage disequilibrium, so a SNP may have a spurious association with a disease due to its linkage with a causal SNP.

The naive view of SNP $\rightarrow$ disease implies that we can just regress the disease status onto the patients' genotypes[2] and discover the causal SNPs, perhaps using a lasso penalty to select a sparse set of SNPs. To deal with the biological complexities, we would like to generalize this method in two ways:

1. Multivariate output: instead of a binary disease status, use all the disease traits that you are trying to predict. This is an instance of *multi-task regression.*
2. Graph-structured output: use the association network among the traits to constrain the regression.

Intuitively, if a SNP is associated with one trait in a cluster of disease-related traits, it is likely to also be associated with other traits in that cluster. This intuition forms the basis of the following methods.

[1] All the materials including figures are based on the papers [1, 2, 3, 4] and slides [5].

[2] How you represent the genotype data is an interesting question. SNPs are differences of a single nucleotide at a given location; there are four possible nucleotides at each location, so each SNP can be represented as an indicator vector of length four, although not all nucleotides may be observed for each SNP. Other genetic variations, such as copy number variations, DNA duplications and deletions, can be harder to represent.

2 Structured association among traits

We use a Quantitative Trait Network (QTN) as our graph over the traits. This is simply a thresholded correlation matrix. Pairs of traits with correlation less than the threshold ρ are not adjacent, whereas traits with correlation greater than ρ are connected, and the weight on the edge is equal to their observed correlation.

3 Graph-Constrained Fused Lasso [3]

Graph-Constrained Fused Lasso leverages the Quantitative Trait Network (QTN) over the output variables to encode the structured regularization functions in regression model. There are two basic methods that are based on the Graph-Constrained Fused Lasso: G$_c$Flasso, and G$_w$Flasso.

3.1 G$_c$Flasso

G$_c$Flasso is designed based on the conventional objective function of lasso with an additional fusion penalty.

$$\hat{\mathbf{B}}^{GC} = \arg\min \sum_k (\mathbf{y}_k - \mathbf{X}\beta_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\beta_k) + \lambda \sum_k \sum_j |\beta_{jk}| + \gamma \sum_{(m,l)\in E} \sum_j |\beta_{jm} - sign(r_{ml})\beta_{jl}| \tag{1}$$

The term r_{ml} denotes the correlation coefficient between m^{th} and l^{th} features. The last term $\sum_{(m,l)\in E} \sum_j |\beta_{jm} - sign(r_{ml})\beta_{jl}|$ is the fusion penalty, which encourages the regression coefficients β_{jm} and β_{jl}) to have similar values. In other words, if the input feature, SNP j, is related to one of the outputs – say trait m – then we expect it to also be related to trait l, because traits m and l are connected in the QTN. The fusion penalty term can be thought of as flattening the regression coefficients of SNPs onto traits with high correlations, so that the effect of each SNP is equalized for the given traits. By regularization parameter γ, the influence of fusion effect is controlled. The fusion penalty is also an L$_1$ penalty, so it encourages sparsity among the *differences* of the weights among SNPS associated with the same cluster of traits.

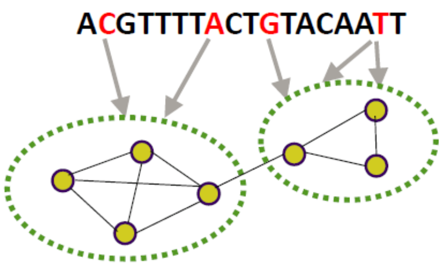
Applying the fusion penalty term in the objective function, it increases the likelihood to learn true associations among the input data, and also reduces false positives. The process of G$_c$Flasso is illustrated in Figure 1 (a).

3.2 G$_w$Flasso

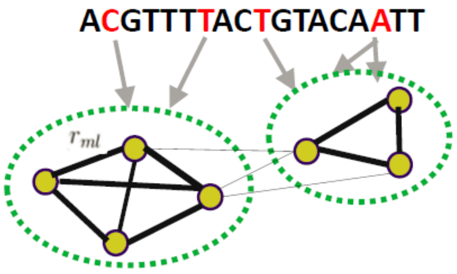
G$_w$Flasso can be considered a generalization of G$_c$Flasso. Its cost function is very similar:

$$\hat{\mathbf{B}}^{GW} = \arg\min \sum_k (\mathbf{y}_k - \mathbf{X}\beta_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\beta_k) + \lambda \sum_k \sum_j |\beta_{jk}| + \gamma \sum_{(m,l)\in E} f(r_{ml}) \sum_j |\beta_{jm} - sign(r_{ml})\beta_{jl}| \tag{2}$$

The only difference is the function $f(r_{ml})$, a function of the edge weights between trait m and trait l (where (m, l) is in the set of edges). This term weights the "fusion penalty" of G$_w$Flasso. G$_w$Flasso allows the degree of fusion to depend on the degree of association between the traits. G$_c$Flasso can be considered a limiting case of G$_w$Flasso when $f(r_{ml}) = 1$. The process of G$_w$Flasso is illustrated in Figure 1 (b).



(a)



(b)

Figure 1: Illustrations of (a) G$_c$Flasso, and (b) G$_w$Flasso.

3.3 Optimization

The optimization problems for equations 1 and 2 are convex, and can be solved using quadratic programming. However, there are two possible problems for optimization of the Graph-Constrained Fused Lasso: 1) it may not scale well, since the problem we are dealing with usually consists of thousands of traits, and 2) the objective functions include L_1 norm, which is non-smooth, and may cause difficulty in optimization.

In order to overcome these two possible problems, we use a variational technique. Additional variables can be introduced to equation 2 to transform non-smooth L_1 terms into smooth L_2 terms. The following minimization problem is equivalent to that in equation 2:

$$\underset{\beta_k, d_{jk}, d_{jml}}{\text{minimize}} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\beta_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\beta_k) + \lambda \sum_{j,k} \frac{(\beta_{jk})^2}{d_{jk}} + \gamma \sum_{(m,l)\in E} f(r_{ml})^2 \sum_j \frac{(\beta_{jm} - sign(r_{ml})\beta_{jl})^2}{d_{jml}} \tag{3}$$

$$\text{subject to} \quad \sum_{j,k} d_{j,k} = 1, \sum_{(m,l)\in E} \sum_j d_{jml} = 1, d_{jk} \geq 0 \text{ for all } j,k, d_{jml} \geq 0 \text{ for all } j, (m,l) \in E.$$

By adding new variables d_{jk} and d_{jml}, the optimization problem consists of only smooth functions, and fast coordinate-descent algorithm can be used to estimate the regression coefficients. The fast coordinate-descent algorithm iteratively updates β_k, d_{jk}, d_{jml} one at a time by fixing the rest of the variables. The update equations for the variables are given below.

$$\beta_{jk} = \left\{ \sum_i x_{ij} \Big(y_{ik} - \sum_{j' \neq j} x_{ij'} \beta_{j'k} \Big) + \gamma \Big(\sum_{(k,l)\in E} \frac{f(r_{kl})^2 sign(r_{kl})\beta_{jl}}{d_{jkl}} + \sum_{(m,k)\in E} \frac{f(r_{mk})^2 sign(r_{mk})\beta_{jm}}{d_{jmk}} \Big) \right\} \Bigg/ \left\{ \sum_i x_{ij}^2 + \frac{\lambda}{d_{jk}} + \gamma \sum_{(k,l)\in E} \frac{f(r_{kl})^2}{d_{jkl}} + \gamma \sum_{(m,k)\in E} \frac{f(r_{mk})^2}{d_{jmk}} \right\} \tag{4}$$

$$d_{jk} = \frac{|\beta_{jk}|}{\sum_{j',a} |\beta_{j'a}|'} \tag{5}$$

$$d_{jml} = \frac{f(r_{ml})|\beta_{jm} - sign(r_{ml})\beta_{jl}|}{\sum_{(a,b \in E)} \sum_{j'} f(r_{ab})|\beta_{j'a} - sign(r_{ab})\beta_{j'b}|} \tag{6}$$

4 Tree-Guided Group Lasso [4]

Tree-based methods scale much better than general graphs. Consider problems where we can cluster the output variables into a hierarchical tree structure – for example, clustering genes according to their similarity. In this case we wish to use the degree of similarity of the variables to weight the regression (see figure 2). For a pair of variables, their similarity can be represented as the height of the smallest subtree containing both variables. We can then introduce a penalty term based on the subtrees.

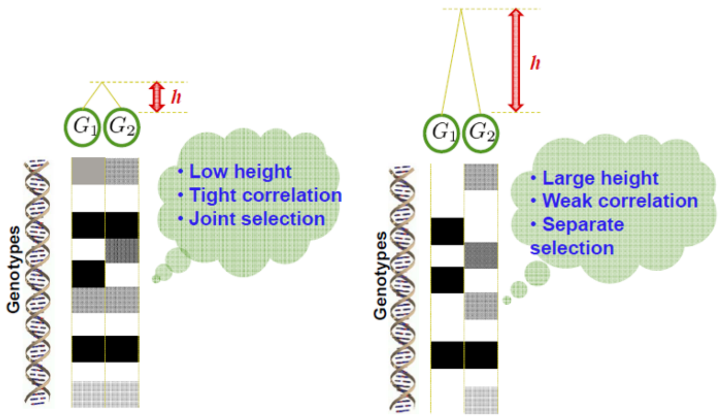


Figure 2: Illustration of two cases genes with different degrees of associations.

Each node v of the tree has a subtree below it containing leaves G_v. The subtree rooted at node v has a height, h_v. We use these terms to design the cost function:

$$\hat{\mathbf{B}}^{\text{Tree}} = \arg\min \sum_k (\mathbf{y}_k - \mathbf{X}\beta_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\beta_k) + \lambda \sum_k \sum_{v \in V} w_v ||\beta^j_{G_v}||_2 \tag{7}$$

where $\beta^j_{G_v}$ is the vector of regression coefficients for variables in G_v (i.e. the leaves of the subtree rooted at v), and the w_v terms are functions of the heights h_v.

This objective function weights the degree of coupling according to subtree height. However, pairs of nodes can appear in multiple subtrees together, in which case they will be coupled together many times with different weights. The difficulty then is to design a weighting function that is consistent: the total penalty for each individual node should be equal, regardless of how many subtrees that node appears in. Kim & Xing (2010) develop a weighting function that is consistent [4]:

To determine the w_v terms, first assign a pair of variables, s_v and g_v, to each tree node v, with the constraint that $s_v + g_v = 1$. s_v reflects the penalties on each variable $j \in G_v$ for selecting them individually, whereas g_v represents the penalty for selecting them jointly. For example, if the entire tree has height 1, and the subtree rooted at v has height h_v, one can set $s_v = h_v$ and $g_v = 1 - h_v$.

Then derive the w_v recursively as follows, starting at the root:

$$\sum_j \sum_{v \in V} w_v ||\beta^j_{G_v}||_2 = \lambda \sum_j W_j(v_{\text{root}}) \tag{8}$$

where

$$W_j(v) = \begin{cases} s_v \cdot \sum_{c \in \text{Children}(v)} |W_j(c)| + g_v \cdot ||\beta^j_{G_v}||_2 & \text{if } v \text{ is an internal node,} \\ \sum m \in G_v |\beta^j_m| & \text{if } v \text{ is a leaf.} \end{cases}$$

Kim & Xing [4] show that using this scheme, the sum of the weights equal 1 for all genes, i.e. that

$$\sum_{v:k \in G_v} w_v = \prod_{m \in Ancestors(v_k)} s_m + \sum_{l \in Ancestors(v_k)} g_l \prod_{m \in Ancestors(v_l)} s_m = 1 \tag{9}$$

When $g_v = 0$ for all v, the tree-guided penalty is just the lasso penalty. When $g_v = 1$ and $s_v = 0$, then the penalty reduces to the L_1/L_2 multi-task learning penalty:

$$\hat{\mathbf{B}}^{L_1/L_2} = \arg\min \sum_k (\mathbf{y}_k - \mathbf{X}\beta_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\beta_k) + \lambda \sum_k \sum_j ||\beta^j||_2$$

which performs joint covariate selection for *all* output variables. In between, you get elastic-net regression: a linear combination of the L_1 and L_2 norms. Elastic-net regression usually applies the same linear combination of penalties to all variables. Tree-guided group lasso generalizes this so that the linear combination of penalties is customized for each output node, based on the tree structure.

5 Two-Graph Guided Multi-Task Lasso [2]

Methods introduced until now focus on incorporating knowledge of the output structure when learning regression coefficients. However, it is more natural to assume that not only outputs of genetic traits, but also the inputs of SNPs form graph structures. SNP values may be associated with each other due to common inheritance: population structures, linkage disequilibrium, etc. Two-Graph Guided Multi-Task Lasso is motivated by this notion, and considers *subnetwork-to-subnetwork* association in eQTL mapping.

5.1 Algorithm

The idea is simply to apply fusion penalty to the input structure as well as the output structure. The objective function is shown in equation 10. We let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs defined on outputs and inputs, respectively, and pen_1 and pen_2 be penalty terms for the discrepancy between the regression coefficients and the edge structures in outputs and inputs, respectively.

$$\underset{B}{\text{minimize}} \quad ||Y - XB||_F^2 + \lambda ||B||_1 + \gamma_1 \cdot pen_1(E_1, B) + \gamma_2 \cdot pen_2(E_2, B) \tag{10}$$

where

$$pen_1(E_1, B) = \sum_{e_{m,l} \in E_1} w(e_m, l) \sum_{j=1}^{J} |b_{jm} - sign(r_{ml}) b_{jl}|,$$

$$pen_2(E_2, B) = \sum_{e_{f,g} \in E_2} w(e_f, g) \sum_{k=1}^{K} |b_{fk} - sign(r_{f,g}) b_{gk}|.$$

Term $r_{m,l}$ denotes the correlation between $\mathbf{y}^m$ and $\mathbf{y}_l$, and $w(\cdot)$ denotes the weight. The process of Two-Graph Guided Multi-Task Lasso is illustrated in Figure 3.

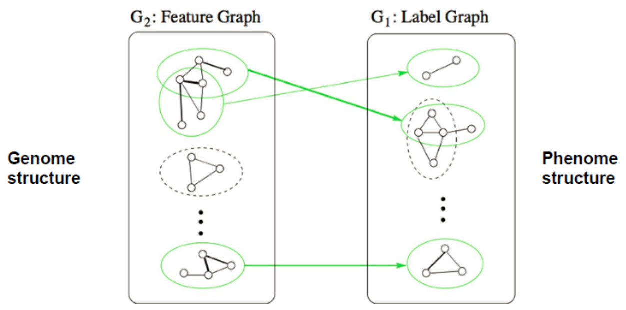


Figure 3: Illustration of Two-Graph Guided Multi-Task Lasso - finding associations between subnetworks of inputs and outputs.

5.2 Optimization

Note that the objective function in equation 10 is not differentiable. To optimize Two-Graph Guided Multi-Task Lasso, the objective function is transformed to a series of smooth functions in forms of constrained ridge-type optimization problem. Then simple coordinate-descent algorithm can be used for optimization.

$$\begin{aligned} \underset{B, d_{j,k}, d1_{jml}, d2_{kfg}}{\text{minimize}} \quad & ||Y - XB||_F^2 + \lambda \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{b_{jk}^2}{d_{jk}} + \gamma_1 \sum_{e_{m,l} \in E_1} w^2(e_m, l) \sum_{j=1}^{J} \frac{(b_{jm} - sign(r_{ml}) b_{jl})^2}{d1_{jml}} \\ & + \gamma_2 \sum_{e_{f,g} \in E_2} w^2(e_f, g) \sum_{k=1}^{K} \frac{(b_{fk} - sign(r_{f,g}) b_{gk})^2}{d2_{kfg}}, \\ \text{subject to} \quad & \sum_{j,k} d_{j,k} = 1, \sum_{e_{m,l} \in E} d1_{jml} = 1, \sum_{e_{f,g \in E_2},k} d2_{kfg} = 1, d_{j,k}, d1_{jml}, d2_{kfg} \geq 0 \end{aligned} \tag{11}$$

6 Proximal Gradient Descent [1]

Genome association mapping usually involves dealing with high-dimensional regression models with penalties for inducing structured sparsity. The penalty terms commonly result in *nonseparability* and *nonsmoothness*, which makes the optimization difficult. Proximal Gradient Descent method simply expands and rotates the domain of penalty terms to resolve problems of *nonseparability* and *nonsmoothness* that occur in various types of penalty terms. In [1], the penalty terms of two methods, *overlapping-group-sparsity-lasso penalty* and *graph-guided-fused-lasso-penalty* are introduced as examples of applications of Proximal Gradient Descent method.

6.1 Resolving *nonseparability*

A commonly used formulation of genome association mapping is given as equation 12:

$$\arg\min_{\beta\in\mathbb{R}^J} f(\beta) = \frac{1}{2}||\mathbf{y}-\mathbf{X}\beta||_2^2 + \Omega(\beta) \tag{12}$$

For *overlapping-group-sparsity-lasso penalty* and *graph-guided-fused-lasso-penalty* methods, the penalty terms are defined as $\Omega(\beta) = \gamma\sum_{g\in\mathcal{G}} w_g||\beta_g||_2$, and $\Omega(\beta) = \gamma\sum_{e=(m,l)\in E, m<l} \tau(r_{ml})|\beta_m - sign(r_{ml})\beta_l|$, respectively, which are nonseparable in β and nonsmooth in function $\Omega(\beta)$.

By utilizing dual norm, penalty terms can be decoupled as follows. We can express $||\beta||_2$ as $||\beta||_2 = \max_{||\alpha_g||_2\leq 1}\alpha_g^T\beta_g$, where $\alpha_g \in \mathbb{R}^{|g|}$ is auxiliary variables with a closed, convex domain $Q = \{\alpha | ||\alpha_g||_2 \leq 1, \forall g \in \mathcal{G}\}$. Then the penalty term can be rewritten as below.

$$\Omega(\beta) = \max_{\alpha\in\mathcal{Q}} \alpha^T C\beta, \tag{13}$$

where

$$C_{(i,g),j} = \begin{cases} \gamma w_g, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}$$

and

$$C_{e=(m,l),j} = \begin{cases} \gamma\cdot\tau(r_{ml}), & \text{if } j = m \\ -\gamma\cdot sign(r_{ml})\tau(r_{ml}), & \text{if } j = l \\ 0, & \text{otherwise} \end{cases}$$

for *overlapping-group-lasso penalty* and *graph-guided-fused-lasso penalty* method, respectively. This reformulation of penalty terms resolves the problem of *nonseparability*, by expanding the terms into higher order.

6.2 Resolving *nonsmoothness*

Now that we have expanded and transformed our penalty terms into separable forms, we will make smooth approximations to the penalty terms. We can approximate the objective function in equation 12 as given below.

$$\arg\min_{\beta\in\mathbb{R}^J} \tilde{f}(\beta) = \frac{1}{2}||\mathbf{y}-\mathbf{X}\beta||_2^2 + f_\mu(\beta), \tag{14}$$

where we make smooth approximations to the penalty term in $f_\mu(\beta)$ by introducing additional term $\mu d(\alpha)$.

$$f_\mu(\beta) = \max_{\alpha\in\mathcal{Q}}(\alpha^T C\beta - \mu d(\alpha)). \tag{15}$$

Term μ is a positive smoothness parameter, and $d(\alpha) = \frac{1}{2}||\alpha||_2^2$ is a smoothing function. By making smooth approximations to the penalty term, the optimization problem can be carried on with less complexity. Figure 4 provides a graphical intuition for this process.

By resolving the *nonseparability* and *nonsmoothness* problems, the Proximal Gradient Descent method scales up the regression problem to larger numbers of variables, and with a faster convergence rate compared to other optimization methods such as interior-point methods.

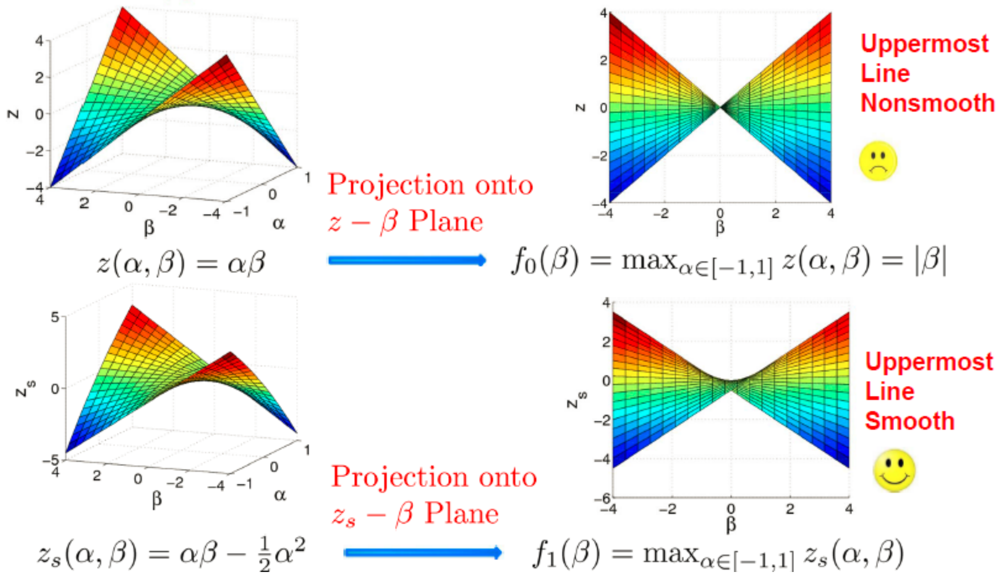


Figure 4: Graphical explanation of how proximal gradient descent method resolves nonseparability and nonsmoothness.

References

[1] Xi Chen, Qihang Lin, Seyoung Kim, Jaime G Carbonell, Eric P Xing, et al. Smoothing proximal gradient method for general structured sparse regression. *The Annals of Applied Statistics*, 6(2):719–752, 2012.

[2] Xiaohui Chen, Xinghua Shi, Xing Xu, Zhiyong Wang, Ryan Mills, Charles Lee, and Jinbo Xu. A two-graph guided multi-task lasso approach for eqtl mapping. In *International Conference on Artificial Intelligence and Statistics*, pages 208–217, 2012.

[3] Seyoung Kim and Eric P Xing. Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, 5(8):e1000587, 2009.

[4] Seyoung Kim and Eric P Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 543–550, 2010.

[5] Eric Xing. http://www.cs.cmu.edu/ epxing/class/10708-15/slides/lecture27-structsparse.pdf, 2015.