Tree-Guided Group Lasso for Multi-Task Regression with Structured Sparsity

Seyoung Kim, Eric P. Xing

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

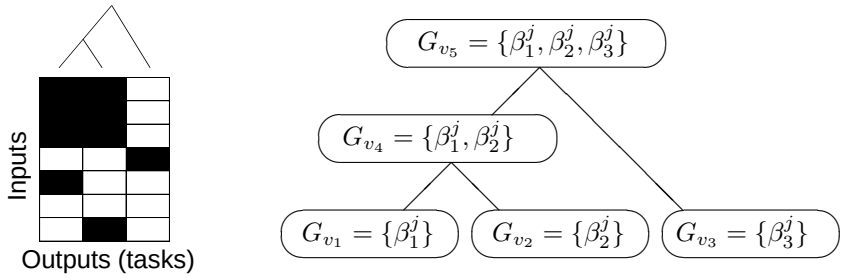
Abstract

We consider the problem of learning a sparse multi-task regression with an application to a genetic association mapping problem for discovering genetic markers that influence expression levels of multiple genes jointly. In particular, we consider the case where the structure over the outputs can be represented as a tree with leaf nodes as outputs and internal nodes as clusters of the outputs at multiple granularity, and aim to recover the common set of relevant inputs for each output cluster. Assuming that the tree structure is available as a prior knowledge, we formulate this problem as a new multi-task regularized regression called tree-guided group lasso. Our structured regularization is based on a group-lasso penalty, where the group is defined with respect to the tree structure. We describe a systematic weighting scheme for the groups in the penalty such that each output variable is penalized in a balanced manner even if the groups overlap. We present an efficient optimization method that can handle a large-scale problem as is typically the case in association mapping that involve thousands of genes as outputs and millions of genetic markers as inputs. Using simulated and yeast datasets, we demonstrate that our method shows a superior performance in terms of both prediction errors and recovery of true sparsity patterns, compared to other methods for multi-task learning.

Keywords: lasso, group lasso, structured sparsity, multi-task learning, association analysis

1 Introduction

Many real world problems in data mining and scientific discovery amount to finding a *parsimonious* and *consistent* mapping function from high dimensional input factors to a structured output



(a) (b)

Figure 1: Tree-regularization for multiple-output regression. (a) An example of a multiple-output regression when the output variables form a tree structure. (b) Groups of variables associated with each node of the tree in (a) in tree-guided group lasso.

signal. For example, in a genetic problem known as expression quantitative trait loci (eQTL) mapping, one attempts to discover an association function from a small set of causal variables known as *single nucleotide polymorphisms* (SNPs) out of a few million candidates, to a set of genes whose expression levels are interdependent in a complex manner. In computer vision, one tries to relate the high-dimensional image features to a structure labeling of objects in the image. An effective approach to this kind of problems is to formulate it as a regression problem from inputs to outputs. In the simplest case where the output is a univariate continuous or discrete response (e.g., a gene expression measure for a single gene), techniques such as lasso [10] or L_1-regularized logistic regression [6, 11] have been developed to find a sparse and consistent regression function that identifies a parsimonious subset of inputs that determine the outputs. However, when the output is a multivariate vector with an internal sparsity structure, the estimation of the regression parameters can potentially benefit from taking into account this sparsity structure in the estimation process such that the output variables that are strongly related can be mapped to the input factors in a synergistic way, which is not possible using the standard lasso.

In a univariate-output regression setting, sparse regression methods that extend lasso have been proposed to allow the recovered relevant inputs to reflect the underlying structural information among the inputs. For example, group lasso [12] assumed that the groupings of the inputs are available as a prior knowledge, and used groups of inputs instead of individual inputs as a unit of variable selection by applying an L_1 norm of the lasso penalty over groups of inputs, while using an L_2 norm for the input variables within each group. This L_1/L_2 norm for group lasso has been extended to a more general setting with various types of more complex structures on the sparsity pattern rather than a simple grouping information, where the key idea is to allow the groups to have an overlap. The hierarchical selection method [13] assumed that the input variables form a tree structure, and designed groups so that the child nodes enter the set of relevant inputs only if its parent node does. The situations with arbitrary overlapping groups have been considered as well [4, 5].

Many of these ideas related to group lasso in a univariate regression may be directly applied to the multi-task regression problems. The L_1/L_2 penalty of group lasso has been used to recover inputs that are jointly relevant to all of the outputs, or tasks, where the L_2 norm is applied to the

outputs instead of groups of inputs as in group lasso [8, 7]. Although the L_1/L_2 penalty has been shown to be effective in a joint covariate selection in multi-task learning, it assumed that all of the tasks are equally related with each other and share the same relavant inputs. However, when there is a complex pattern in the way that the tasks are related, only a subset of highly related tasks may share the same sparsity pattern in their regression coefficients. In order to address this problem of a structured sparsity recovery in a multi-task learning, extensions of group lasso with overlapping groups [13, 4, 5] could be applied. However, the overlapping groups in their regularization methods can cause an imbalance among different outputs, since the regression coefficients for an output that appears in a large number of groups are more heavily penalized than for other outputs with memberships to fewer groups. An ad hoc weighting scheme that weights each group differently in the regularization function has been introduced to correct for this imbalance.

In this paper we consider a particular case of a sparse multi-task regression problem, where the outputs can be grouped at multiple granularity. We assume that this multi-level grouping structure is encoded as a tree over the outputs with an arbitrary height, where each leaf node represents an individual output variable and each internal node indicates the cluster of the output variables that correspond to the leaf nodes of the subtree rooted at the given internal node. Each internal node in the tree is associated with a weight that represents the height of the subtree, or how tightly the outputs in the cluster for that internal node are correlated. As illustrated in Figure 1(a), the outputs in each cluster are likely to be influenced by a common set of inputs, and this type of sharing of sparsity pattern is stronger among tightly correlated outputs in the cluster with a smaller height in the tree.

In order to achieve this type of structured sparsity at multiple levels of the hierarchy among the outputs, we propose a new regularized regression method called *tree-guided group lasso* that defines groups of variables based on a tree which is assumed to be available as prior knowledge. The groups are defined at multiple granularity along the tree to encourage a joint covariate selection within each cluster of outputs. We describe a weighting scheme that weights each group such that clusters of strongly correlated variables are more encouraged to share common inputs than clusters with weaker correlation. Compared to an arbitrary assignment of values for the group weights which can lead to an inconsistent estimate [5], the weights are systematically defined in terms of the heights of the internal nodes in the tree, and each output variable is penalized in a balanced manner even if the groups overlap.

Our work is primarily motivated by the genetic association mapping problem, where the goal is to identify a small number of SNPs (inputs) out of millions of SNPs that influence phenotypes (outputs) such as gene expression measurements for thousands of genes. Many previous studies have found that multiple genes often participate in the same biological pathways, and are co-expressed as a module. Furthermore, evidence has been found that these genes within a module often share a common genetic basis that causes the variations in their expression levels [14, 2]. However, most of the previous approaches were based on a single-phenotype analysis that treats the multiple phenotypes as independent of each other, and there has been a lack of statistical tools that can take advantage of this relatedness among multiple genes to identify SNPs that influence the module jointly. In this paper, we apply the hierarchical agglomerative clustering algorithm, a popular method for visualizing the clustering structure among the genes, to phenotype data, and use the

clustering tree to construct a tree regularization in our regression method. Although this clustering tree from the hierarchical agglomerative clustering has been previously used as a structural representation of genes in a regression framework, they computed averages over members of the cluster for each internal node in the tree, and used these averages as inputs, leading to a potential loss of information [3]. In our method, we use the original data with the clustering tree as a guide towards a structured sparsity. In our experiments, we demonstrate that our proposed method can be successfully applied to select SNPs correlated with multiple genes, using both simulated and yeast datasets.

We begin our discussion with a brief overview of sparse regression methods and multi-task learning in Section 2. We describe our proposed method in Section 3, and the optimization algorithm in Section 4. We present the experimental results using simulated data and yeast data in Section 5, and conclude in Section 6.

2 Background on Sparse Regression and Multi-task Learning

Let us assume a sample of N instances, each represented by a J-dimensional input vector and a K-dimensional output vector. Let $\mathbf{X}$ denote the $N \times J$ input matrix, whose column corresponds to observations for the j-th input $\mathbf{x}_j = \{x_j^1, \ldots, x_j^N\}^T$. In genetic association mapping, each element x_j^i of the input matrix takes values from $\{0, 1, 2\}$ according to the number of minor alleles at the j-th locus of the i-th individual. Let $\mathbf{Y}$ denote the $N \times K$ output matrix, whose column is a vector of observations for the k-th output $\mathbf{y}_k = \{y_k^1, \ldots, y_k^N\}^T$. For each of the K output variables, we assume a linear model:

$$\mathbf{y}_k = \mathbf{X}\boldsymbol{\beta}_k + \boldsymbol{\epsilon}_k, \quad \forall k = 1, \ldots, K, \tag{1}$$

where $\boldsymbol{\beta}_k$ is a vector of J regression coefficients $\{\beta_k^1, \ldots, \beta_k^J\}^T$ for the k-th output, and $\boldsymbol{\epsilon}_k$ is a vector of N independent error terms having mean 0 and a constant variance. We center the $\mathbf{y}_k$'s and $\mathbf{x}_j$'s such that $\sum_i y_k^i = 0$ and $\sum_i x_j^i = 0$, and consider the model without an intercept.

When J is large and the number of inputs relevant to the output is small, lasso offers an effective feature selection method for the model in Equation (1) [10]. Let $\mathbf{B} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ denote the $J \times K$ matrix of regression coefficients of all K outputs. Then, lasso obtains $\hat{\mathbf{B}}^{\text{lasso}}$ by solving the following optimization problem:

$$\hat{\mathbf{B}}^{\text{lasso}} = \text{argmin} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \sum_k |\beta_k^j|, \tag{2}$$

where λ is a tuning parameter that controls the amount of sparsity in the solution. Setting λ to a small value leads to a smaller number of non-zero regression coefficients. Clearly, the standard lasso in Equation (2) offers no mechanism to explicitly couple output variables.

In multi-task learning, an L_1/L_2 penalty has been used to take advantage of the relatedness of the outputs and recover the sparsity pattern shared across the related tasks. In an L_1/L_2 penalty, an L_2 norm is applied to the regression coefficients for all outputs for each input, $\boldsymbol{\beta}^j$, separately, and

these J L_2 norms are combined through an L_1 norm to encourage sparsity across input variables. The L_1/L_2-penalized multi-task regression is defined as the following optimization problem:

$$\hat{\mathbf{B}}^{L_1/L_2} = \text{argmin} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \|\boldsymbol{\beta}^j\|_2 \tag{3}$$

The L_1 part of the penalty plays the role of selecting inputs relevant to at least one task, and the L_2 part combines information across tasks. Since the L_2 penalty does not have the property of encouraging sparsity, if the j-th input is selected as relevant, all of the elements of $\boldsymbol{\beta}^j$ take non-zero values. Thus, the estimate $\hat{\mathbf{B}}^{L_1/L_2}$ is sparse only across inputs but not across outputs.

3 Tree-Guided Group Lasso for Sparse Multiple-output Regression

The L_1/L_2-penalized regression assumes that all of the outputs in the problem share the common set of relevant input variables. Although this method has been shown to be effective under this scenario [8, 7], in many real-world applications, the correlation pattern in the multiple outputs often has a complex structure such as in gene expression data with subsets of genes forming a functional module, and it is not realistic to assume that all of the tasks share the same set of relevant inputs as in the L_1/L_2-regularized regression. A subset of highly related outputs may share a common set of relevant inputs, whereas weakly related outputs are less likely to be affected by the same inputs.

We assume that the relationships among the outputs can be represented as a tree T with the set of vertices V of size $|V|$, as shown in Figure 1(a), where each of the K leaf nodes is associated with an output variable. The internal nodes of the tree represent groupings of the output variables located at the leaves of the subtree rooted at the given internal node. Each internal node near the bottom of the tree shows that the output variables of its subtree are highly correlated, whereas the internal nodes near the root represent weak correlations among the outputs in its subtree. This tree structure may be available as a prior knowledge, or can be learned from data using methods such as a hierarchical agglomerative clustering. Furthermore, we assume that each node $v \in V$ is associated with a weight w_v, representing the height of the subtree rooted at v.

Given this tree T over the outputs, we generalize the L_1/L_2 regularization in Equation (3) to a tree regularization as follows. We expand the L_2 part of the L_1/L_2 penalty into a group-lasso penalty, where the group is defined based on tree T as follows. Each node $v \in V$ of tree T is associated with a group G_v whose members consist of all of the output variables (or leaf nodes) in the subtree rooted at node v. For example, Figure 1(b) shows the groups associated with each of the nodes of the tree in Figure 1(a). Given these groups of outputs that arise from tree T, tree-guided group lasso can be written as

$$\hat{\mathbf{B}}^T = \text{argmin} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \sum_{v \in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2, \tag{4}$$

where $\boldsymbol{\beta}^j_{G_v}$ is a vector of regression coefficients $\{\beta^j_k : k \in G_v\}$. Each group of regression coefficients $\boldsymbol{\beta}^j_{G_v}$ is weighted with w_v so that the group with a large weight is penalized more.

Assuming that each internal node v of the tree T is associated with two quantities s_v and g_v that satisfy the condition $s_v + g_v = 1$, $0 \leq s_v, g_v \leq 1 > 0$, we define w_v's in Equation (4) in terms of s_v's and g_v's as we describe below. The s_v represents the weight for selecting the output variables associated with each of its child nodes separately, whereas the g_v represents the weight for selecting them jointly. We first consider a simple case with two outputs ($K = 2$) with a tree of three nodes that consist of two leaf nodes (v_1 and v_2) and one root node (v_3), and then, generalize this to an arbitrary tree. When $K = 2$, the penalty term in Equation (4) can be written as

$$\sum_j \sum_{v \in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 = \sum_j \Big[s_3 \Big(|\beta^j_1| + |\beta^j_2| \Big) + g_3 \Big(\sqrt{(\beta^j_1)^2 + (\beta^j_2)^2} \Big) \Big]. \tag{5}$$

This is equivalent to an elastic-net penalty [15], where β^j_1 and β^j_2 can be selected either jointly or separately according to the weights s_3 and g_3. When $s_3 = 0$, the penalty in Equation (5) becomes equivalent to a ridge-regression penalty, whereas setting $g_3 = 0$ in Equation (5) leads to a lasso penalty. In general, when tree T has a height one with the root node having all of the outputs as its leaf nodes, the tree-guided group-lasso penalty corresponds to an elastic-net penalty, and the s_v and g_v are weights for the L_1 and L_2 penalties, respectively. A large value of g_v indicates that the outputs are highly related, and encourages a joint input selection by heavily weighting the L_2 part of the elastic-net penalty.

When tree T has a height larger than one, we recursively apply the similar operation in Equation (5) starting from the root node towards the leaf nodes as follows:

$$\sum_j \sum_{v \in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 = \lambda \sum_j W_j(v_{\text{root}}), \tag{6}$$

where

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

It can be shown that the following relationship holds between w_v's and (s_v, g_v)'s.

$$w_v = \begin{cases} g_v \displaystyle\prod_{m \in \text{Ancestors}(v)} s_m & \text{if } v \text{ is an internal node} \\ \displaystyle\prod_{m \in \text{Ancestors}(v)} s_m & \text{if } v \text{ is a leaf node.} \end{cases}$$

The above weighting scheme extends the elastic-net penalty hierarchically, where the L_2 norm of the standard elastic-net penalty corresponds to the group-lasso-like L_2 norm in tree-guided group lasso. Thus, at each internal node v, a large value (small penalization) of s_v encourages a separate selection of covariates for the outputs associated with the given node v, whereas a large value for g_v encourages a joint covariate selection across the input. If s_v=1 and $g_v = 0$ for all $v \in V$,

then only separate selections are performed, and the tree-guided group lasso penalty reduces to the lasso penalty. On the other hand, if s_v=0 and $g_v = 1$ for all $v \in V$, the penalty reduces to the L_1/L_2 penalty in Equation (3) that performs only a joint covariate selection for all outputs. The unit contour surfaces of various penalties for β_1^j, β_2^j, and β_3^j with groups as defined in Figure 1 are shown in Figure 2.

Example 1. *Given the tree T in Figure 1, for the j-th input the penalty of the tree-guided group lasso in Equation (6) can be written as follows:*

$$\begin{aligned}
W_j(v_1) &= |\beta_1^j|, \quad W_j(v_2) = |\beta_2^j|, \quad W_j(v_3) = |\beta_3^j|, \\
W_j(v_4) &= g_{v_4} \cdot \|\boldsymbol{\beta}^j_{G_{v_4}}\|_2 + s_{v_4} \cdot (|W_j(v_1)| + |W_j(v_2)|) = g_{v_4} \cdot \|\boldsymbol{\beta}^j_{G_{v_4}}\|_2 + s_{v_4} \cdot (|\beta_1^j| + |\beta_2^j|) \\
W_j(v_{root}) &= W_j(v_5) = g_{v_5} \cdot \|\boldsymbol{\beta}^j_{G_{v_5}}\|_2 + s_{v_5} \cdot (|W_j(v_4)| + |W_j(v_3)|) \\
&= g_{v_5} \cdot \|\boldsymbol{\beta}^j_{G_{v_5}}\|_2 + s_{v_5} \cdot g_{v_4} \|\boldsymbol{\beta}^j_{G_{v_4}}\|_2 + s_{v_5} \cdot s_{v_4}(|\beta_1^j| + |\beta_2^j|) + s_{v_5}|\beta_3^j|.
\end{aligned}$$

Proposition 1. *For each of the k-th output, the sum of the weights w_v for all nodes $v \in V$ in T whose group G_v contains the k-th output as a member equals one. In other words, the following holds:*

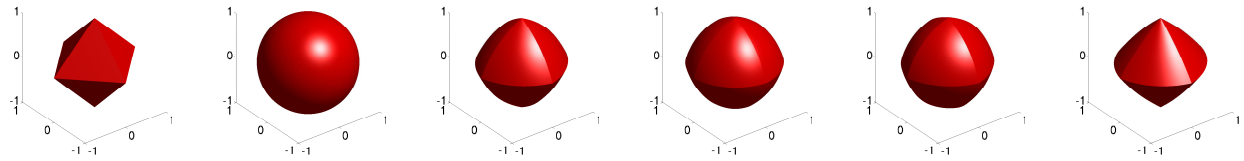
$$\sum_{v:k \in G_v} w_v = \prod_{m \in Ancestors(v_{leaf})} s_m + \sum_{l \in Ancestors(v_{leaf})} g_l \prod_{m \in Ancestors(l)} s_m = 1.$$

Proof. We assume an ordering of the nodes $\{v : k \in G_v\}$ from the leaf v_k to the root v_{root}, and represent the ordered nodes as $v_1, \ldots, v_M$. Since we have $s_v + g_v = 1$ for all $v \in V$, we have

$$\begin{aligned}
\sum_{v:k \in G_v} w_v &= \prod_{m=1}^{M} s_m + \sum_{l=1}^{M} g_l \prod_{m=l+1}^{M} s_m = s_1 \prod_{m=2}^{M} s_m + g_1 \prod_{m=2}^{M} s_m + \sum_{l=2}^{M} g_l \prod_{m=l+1}^{M} s_m \\
&= (s_1 + g_l) \cdot \prod_{m=2}^{M} s_m + \sum_{l=2}^{M} g_l \prod_{m=l+1}^{M} s_m = \prod_{m=2}^{M} s_m + \sum_{l=2}^{M} g_l \prod_{m=l+1}^{M} s_m = \ldots = 1
\end{aligned}$$

□

Proposition 1 states that even if each ouput k belongs to multiple groups associated with internal nodes $\{v : k \in G_v\}$ and appears multiple times in the overall penalty in Equation (6), the sum over weights of all of the groups that contain the given output variable is always one. Thus, the weighting scheme in Equation (6) guarantees that the regression coefficients for all of the outputs are penalized equally. In contrast, group lasso with overlapping groups proposed in [5] used an arbitrarily defined weights, which was empirically shown to lead to an inconsistent estimate. Another main difference between our method and the work in [5] is that we take advantage of groups which contain other groups along the tree structure, whereas they tried to remove such groups as redundant in [5].



(a) (b) (c) (d) (e) (f)

Figure 2: Unit contour surface for $\{\beta_1^j, \beta_2^j, \beta_3^j\}$ in various penalties, assuming the tree structure of output variables in Figure 1. (a) Lasso, (b) L_1/L_2, (c) tree-guided group lasso with $g_1 = 0.5$ and $g_2 = 0.5$, (d) $g_1 = 0.7$ and $g_2 = 0.7$, (e) $g_1 = 0.2$ and $g_2 = 0.7$, and (f) $g_1 = 0.7$ and $g_2 = 0.2$.

4 Parameter Estimation

In order to estimate the parameters in tree-guided group lasso, we use the alternative formulation of the problem in Equation (4) that was previously introduce for group lasso [1], given as

$$\hat{\mathbf{B}}^T = \text{argmin} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \Big(\sum_j \sum_{v \in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 \Big)^2.$$

Since the L_1/L_2 norm in the above equation is a non-smooth function, it is not trivial to optimize it directly. Using the fact that the variational formulation of a mixed norm regularization is equal to a weighted L_2 regularization [9], we re-write the above problem so that it contains only smooth functions, as follows:

$$\begin{aligned} \hat{\mathbf{B}}^T = \text{argmin} \quad & \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \sum_{v \in V} \frac{w_v^2 \|\boldsymbol{\beta}^j_{G_v}\|_2^2}{d_{j,v}} \\ & \text{subject to} \quad \sum_j \sum_v d_{j,v} = 1, \; d_{j,v} \geq 0 \quad \forall j, v, \end{aligned}$$

where we introduced additional variables $d_{j,v}$'s that need to be estimated. We solve the problem in the above equation by optimizing $\boldsymbol{\beta}_k$'s and $d_{j,v}$'s alternately over iterations until convergence. In each iteration, we first fix the values for $\boldsymbol{\beta}_k$'s and update $d_{j,v}$'s as follows:

$$d_{j,v} = \|\boldsymbol{\beta}_{j,v}\|_2 / \Big[\sum_j \sum_{v \in V} \|\boldsymbol{\beta}_{j,v}\|_2 \Big].$$

Then, we hold the values $d_{j,v}$'s as constant, and update $\boldsymbol{\beta}_k$'s as

$$\boldsymbol{\beta}_k = \Big(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{D} \Big)^{-1} \mathbf{X}^T \mathbf{y}_k,$$

where $\mathbf{D}$ is a $J \times J$ diagonal matrix with $\sum_{v \in V} w_v^2 / d_{j,v}$ in the j-th element along the diagonal. The regularization parameter λ can be selected using a cross-validation.

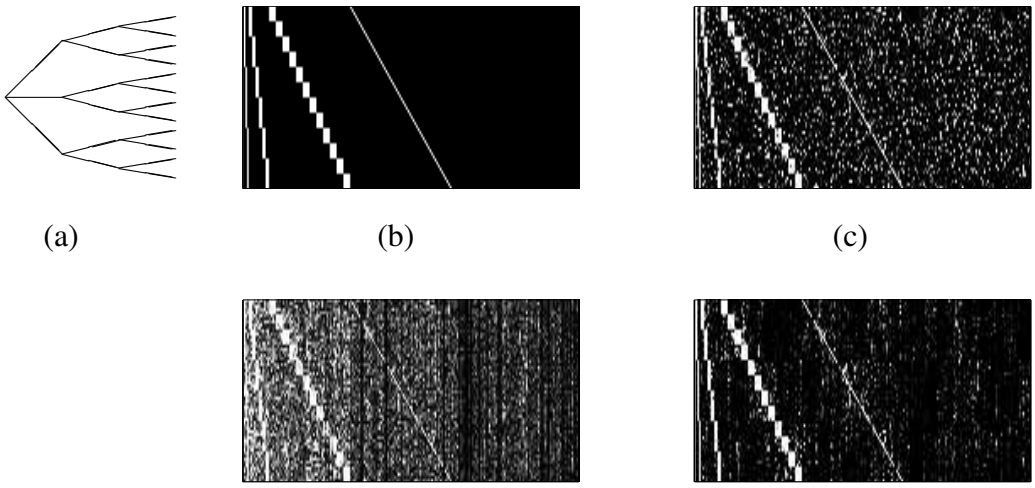


Figure 3: An example of regression coefficients estimated from a simulated dataset. (a) Tree structure of the output variables, (b) true regression coefficients, (c) lasso, (d) L_1/L_2, (e) tree-guided group lasso. The rows represent outputs, and the columns inputs.

5 Experiments

We demonstrate the performance of our method on simulated datasets and a yeast dataset of genotypes and gene expressions, and compare the performance with those from lasso and the L_1/L_2-regularized regression that do not assume any structure among outputs. We evaluate these methods based on two criteria, test error and sensitivity/specificity in detecting true relevant inputs.

5.1 Simulation Study

We simulate data using the following scenario analogous to genetic association mapping. We simulate $(\mathbf{X}, \mathbf{Y})$ with $K = 60$, $J = 200$ and $N = 150$ for the training set as follows. We first generate the inputs $\mathbf{X}$ by sampling each element in $\mathbf{X}$ from a uniform distribution over $\{0, 1, 2\}$ that corresponds to the number of mutated alleles at each genetic locus. Then, we set the values of $\mathbf{B}$ by first selecting non-zero entries and filling these entries with a pre-defined value. We assume a hierarchical structure of height four over the outputs as shown in Figure 3(a), and select the non-zero elements of $\mathbf{B}$ so that they correspond to the groupings in the sparsity structure given by this tree. Figure 3(b) shows the true non-zero elements as white pixels with outputs as rows and inputs as columns. Given the $\mathbf{X}$ and $\mathbf{B}$, we generate $\mathbf{Y}$ with noise distributed as $N(0, 1)$.

We fit lasso, the L_1/L_2-regularized regression, and our method to the simulated dataset with signal strengths of the non-zero elements of $\mathbf{B}$ set to 0.4, and show the results in Figures 3(c)-(e), respectively. Since lasso does not have any mechanism to borrow strength across different tasks, false positives of the estimated non-zero regression coefficients are distributed randomly across the matrix $\hat{\mathbf{B}}^{\text{lasso}}$ in Figure 3(c). On the other hand, the L_1/L_2 regularization method blindly combines

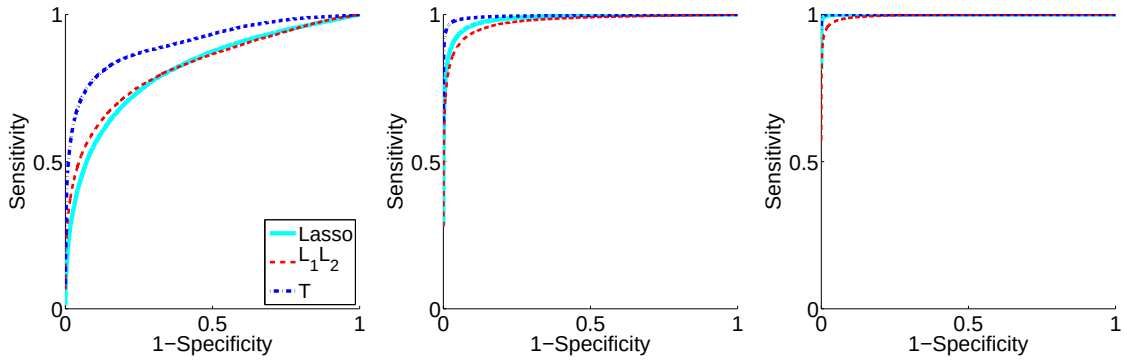


Figure 4: ROC curves for the recovery of true non-zero regression coefficients. Results are averaged over 50 simulated datasets. (a) $\beta_k^j = 0.2$, (b) $\beta_k^j = 0.4$, and (c) $\beta_k^j = 0.6$.

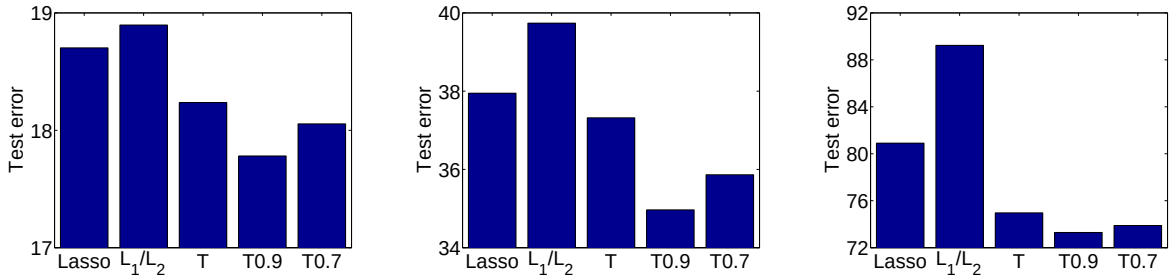


Figure 5: Prediction errors of various regression methods using simulated datasets. Results are averaged over 50 simulated datasets. (a) $\beta_k^j = 0.2$, (b) $\beta_k^j = 0.4$, and (c) $\beta_k^j = 0.6$.

information across the outputs regardless of the sparsity structure, and the L_2 penalty over the outputs does not encourage sparsity. As a result, once an input is selected as relevant for an output, it gets selected for all of the other outputs, which tends to create a vertical stripes of non-zero values as shown in Figure 3(d). When the true hierarchical structure in Figure 3(a) is available as prior knowledge, it is visually clear from Figure 3(e) that our method is able to suppress false positives of non-zero regression coefficients, and recover the true underlying sparsity structure significantly better than other methods.

In order to systematically evaluate the performance of the different methods, we generate 50 simulated datasets, and show in Figure 4 receiver operating characteristic (ROC) curves for the recovery of the true sparsity pattern averaged over these datasets. Figures 4(a)-(c) represent results from different signal strengths in $\mathbf{B}$ of sizes 0.2, 0.4, and 0.6, respectively. Our method clearly outperforms lasso and the L_1/L_2 regularization method. Especially when the signal strength is weak in Figure 4(a), the advantage of incorporating the prior knowledge of the tree as sparsity structure is significant.

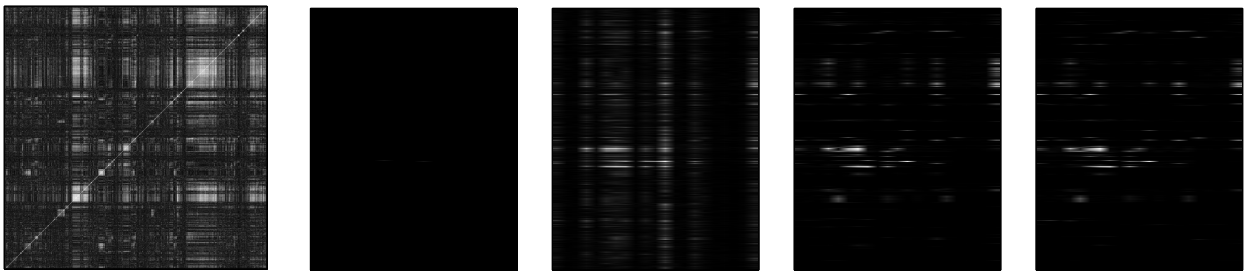
We compare the performance of the different methods in terms of prediction error, using additional 50 samples as test data, and show the results in Figures 5(a)-(c) for signal strengths of sizes 0.2, 0.4, and 0.6, respectively. We find that our method has a lower prediction error than the methods that do not incorporate the sparsity pattern across outputs.

We also consider the scenario where the true tree structure in Figure 3(a) is not known *a priori*. In this case, we learn a tree by running a hierarchical agglomerative clustering on the $K \times K$ correlation matrix of the outputs, and use this tree and the weights h_v's associated with each internal node in our method. The weight h_v of each internal node v returned by the hierarchical agglomerative clustering indicates the height of the subtree rooted at the node, or how tightly its members are correlated. After normalizing the weights (denoted as h'_v) of all of the internal nodes such that the root is at height one, we assign $g_v = h'_v$ and $s_v = 1 - h'_v$. Since the tree obtained in this manner represents a noisy realization of the true underlying tree structure, we discard the nodes for weak correlation near the root of the tree by thresholding h'_v at $\rho = 0.9$ and 0.7, and show the prediction errors in Figure 5 as T0.9 and T0.7. Even when the true tree structure is not available, our method is able to benefit from taking into account the output sparsity structure, and gives lower prediction errors.

5.2 Analysis of Yeast Data

We analyze the genotype and gene expression data of 114 yeast strains [14] using various sparse regression methods. We focus on the chromosome 3 with 21 SNPs and 3684 genes. Although it is well established that genes form clusters in terms of expression levels that correspond to functional modules, the hierarchical structure over correlated genes is not directly available as a prior knowledge, and we learn the tree structure and node weights from the gene expression data by running the hierarchical agglomerative clustering algorithm as we described in the previous section. We use only the internal nodes with heights $h'_v < 0.7$ or 0.9 in our method. The goal of the analysis is to search for SNPs (inputs) whose variation induces a significant variation in the gene expression levels (outputs) over different strains. By applying our method that incorporates information on gene modules at multiple granularity along the hierarchical clustering tree, we expect to be able to identify SNPs that influence a group of genes that are co-expressed or co-regulated.

In Figure 6(a), we show the $K \times K$ correlation matrix of the gene expressions after reordering the rows and columns according to the results of the clustering algorithm. The estimated $\mathbf{B}$ is shown for lasso, the L_1/L_2-regularized regression and our method with $\rho = 0.9$ and 0.7 in Figures 6(b)-(e), respectively, where the rows represent genes and the columns SNPs. The lasso estimates in Figure 6(b) are extremely sparse and do not reveal any interesting structure in SNP-gene relationships. We believe that the association signals are very weak as is typically the case in a genetic association study, and that lasso is unable to detect such weak signals since it does not borrow strength across genes. The estimates from the L_1/L_2 regularized regression are not sparse across genes, and tend to form vertical stripes of non-zero regression coefficients as can be seen in Figure 6(c). Our method in Figures 6(d)-(e) reveals clear groupings in the patterns of associations between genes and SNPs. Our method performs significantly better in terms of prediction errors as can be seen in Figure 7.



(a) (b) (c) (d) (e)

Figure 6: Results for the yeast dataset. (a) Correlation matrix of the gene expression data, where rows and columns are reordered after applying agglomerative hierarchical clustering. Estimated regression coefficients are shown for (b) lasso, (c) L_1/L_2, (d) tree-guided group lasso with $\rho = 0.9$, and (e) with $\rho = 0.7$. In (b)-(e), the rows represent genes (outputs), and the columns markers (inputs).

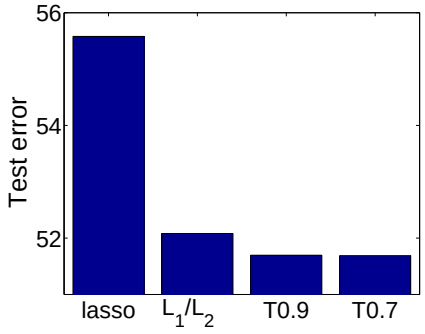
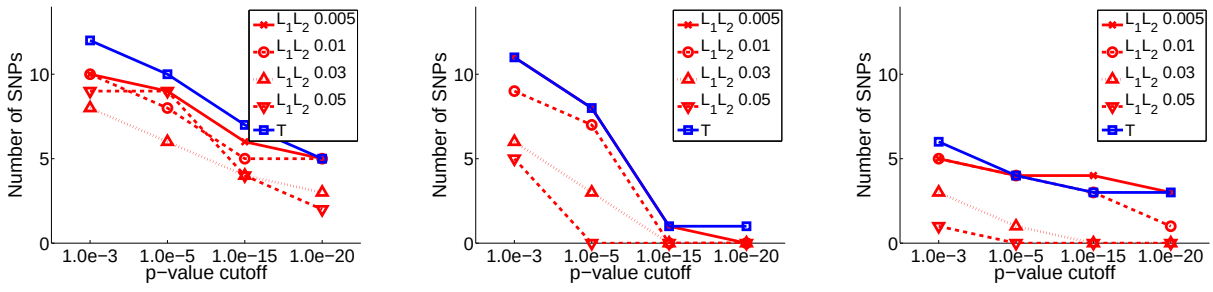


Figure 7: Prediction error for the yeast dataset.



(a) (b) (c)

Figure 8: Enrichment of GO category in estimated regression coefficients for the yeast dataset. (a) Biological process, (b) molecular function, and (c) cellular component.

Given the estimates of $\mathbf{B}$ in Figure 6, we look for an enrichment of GO categories among the genes with non-zero estimated coefficients for each SNP. A group of genes that form a module often participate in the same pathways, leading to an enrichment of a GO category among the members of the module. Since we are interested in identifying SNPs influencing gene modules and our method reflects this joint association through the hierarchical clustering tree, we hypothesize that our method would reveal a more significant GO enrichment in the estimated non-zero elements in $\mathbf{B}$. In order to search for a GO enrichment in the results for our method, we use all of the genes with non-zero elements in $\mathbf{B}$ for each SNP. On the other hand, the estimates of the L_1/L_2 regularized method are not sparse across genes. Thus, we threshold the absolute values of the estimated $\mathbf{B}$ at 0.005, 0.01, 0.03, and 0.05, and search for GO enrichment only for those genes with β_k^j above the threshold.

We perform this analysis for each of the three broad GO categories, biological processes, molecular functions, and cellular components, and plot the number of SNPs with significant GO enrichments at different p-value cutoffs in Figure 8. Regardless of the thresholds for selecting significant associations in the L_1/L_2 estimates, our method generally finds more significant enrichment.

6 Conclusions

In this paper, we considered a feature selection problem in a multiple-output regression setting when the groupings of the outputs can be defined hierarchically using a tree. We proposed a tree-guided group lasso that finds a sparse estimate of regression coefficients while taking into account the joint sparsity structure across outputs given by a tree. We demonstrated the performance of our method using simulated and yeast datasets.

References

[1] F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

[2] Y. Chen, J. Zhu, P.K. Lum, X. Yang, S. Pinto, D.J. MacNeil, C. Zhang, J. Lamb, S. Edwards, S.K. Sieberts, et al. Variations in DNA elucidate molecular networks that cause disease. *Nature*, 452(27):429–35, 2008.

[3] T. Hastie, R. Tibshirani, Botstein D., and P. Brown. Supervised harvesting of gene expression trees. *Genome Biology*, 2(1):research0003.1–0003.12, 2001.

[4] L. Jacob, G. Obozinski, and J. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.

[5] R. Jenatton, J. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, INRIA, 2009.

[6] A. Ng. Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[7] G. Obozinski, B. Taskar, and M. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Journal of Statistics and Computing*, 2009.

[8] G. Obozinski, M.J. Wainwright, and M.J. Jordan. High-dimensional union support recovery in multivariate regression. In *Advances in Neural Information Processing Systems 21*, 2008.

[9] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[10] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[11] M. J. Wainwright, P. Ravikumar, and J. Lafferty. High-dimensional graphical model selection using l_1-regularized logistic regression. In *Advances in Neural Information Processing Systems 18*, 2006.

[12] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of Royal Statistical Society, Series B*, 68(1):49–67, 2006.

[13] P. Zhao, G. Rocha, and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. Technical Report 703, Department of Statistics, University of California, Berkeley, 2008.

[14] J. Zhu, B. Zhang, E.N. Smith, B. Drees, R.B. Brem, L. Kruglyak, R.E. Bumgarner, and E.E. Schadt. Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. *Nature Genetics*, 40:854–61, 2008.

[15] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of Royal Statistical Society, Series B*, 67(2):301–320, 2005.