6-2010

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

Seyoung Kim
*Carnegie Mellon University*

Eric P. Xing
*Carnegie Mellon University*, epxing@cs.cmu.edu



Published In
Proceedings of the 27th International Conference on Machine Learning.

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

**Seyoung Kim** SSSYKIM@CS.CMU.EDU
**Eric P. Xing** EPXING@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA

## Abstract

We consider the problem of learning a sparse multi-task regression, where the structure in the outputs can be represented as a tree with leaf nodes as outputs and internal nodes as clusters of the outputs at multiple granularity. Our goal is to recover the common set of relevant inputs for each output cluster. Assuming that the tree structure is available as 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 groups are 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. 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.

## 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 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 structured 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 measurement for a single gene), techniques such as lasso (Tibshirani, 1996) or $L_1$-regularized logistic regression (Ng, 2004; Wainwright et al., 2006) have been developed to identify a parsimonious subset of covariates that determine the outputs. However, in the problem of *multi-task regression*, where 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. This will allow the strongly related output variables to be mapped to the input factors in a synergistic way, which is not possible in 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 assumed that the groupings of the inputs are available as prior knowledge, and used groups of inputs instead of individual inputs as a unit of variable selection (Yuan & Lin, 2006). Group lasso achieved this 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 to encode prior knowledge on various sparsity patterns, where the key idea is to allow the groups to have an overlap. The hierarchical selection method (Zhao et al., 2008) assumed that the input variables form a tree structure, and designed groups

Appearing in *Proceedings of the 27$^{th}$ International Conference on Machine Learning*, Haifa, Israel, 2010. 

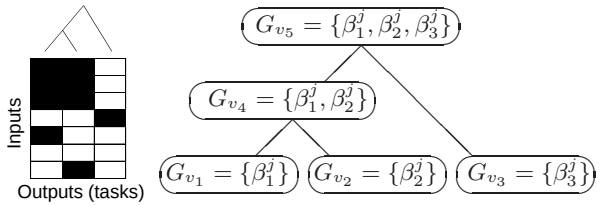



*Figure 1.* Tree regularization for multiple-output regression. (a) An example of a multiple-output regression when the correlation structure in output variables forms a tree. (b) Groups of regression coefficients associated with each node of the tree in (a) in tree-guided group lasso.

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 (Jacob et al., 2009; Jenatton et al., 2009).

Many of these ideas related to group lasso in a univariate-output regression may be directly applied to 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, by applying the $L_2$ norm to outputs instead of groups of inputs as in group lasso (Obozinski et al., 2008; 2009). Although the $L_1/L_2$ penalty has been shown to be effective in a joint covariate selection for multi-task learning, it does not assume any structure among the outputs. The extensions of group lasso with overlapping groups (Zhao et al., 2008; Jacob et al., 2009; Jenatton et al., 2009) may be directly applicable in a multi-task learning to incorporate prior knowledge on structures. However, the overlapping groups in their regularization methods can cause an imbalance among different outputs, because 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. Although a weighting scheme that weights each group differently in the regularization function has been proposed to correct for this imbalance, this ad hoc approach can still lead to inconsistent estimates (Jenatton et al., 2009).

In this paper, we consider a particular case of a sparse multi-task regression problem when the outputs can be grouped at multiple granularity. We assume that this multi-level grouping structure is encoded as a tree over the outputs, 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. As illustrated in Figure 1(a), the outputs in each cluster are likely to be influenced by a common set of inputs. In order to achieve this type of structured sparsity at multiple levels of the hierarchy among the outputs, we propose a novel regularized regression method called *tree-guided group lasso* that applies group lasso to groups of output variables defined in terms of a hierarchical clustering tree. We assume this tree is available as prior knowledge, and define groups at multiple granularity along the tree to encourage a joint covariate selection within each cluster of outputs. Our approach can handle any types of trees with an arbitrary height.

In particular, we describe a novel weighting scheme that systematically weights each group in the tree-guided group-lasso penalty such that clusters of strongly correlated outputs are more encouraged to share common covariates than clusters of weakly correlated outputs. As was noted in Jenatton et al. (2009), an arbitrary assignment of values for the group weights can lead to an inconsistent estimate. Our approach is the first method for achieving a structured sparsity that offers a systematic weighting scheme and penalizes regression coefficients corresponding to each group in a balanced manner even when 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. Many previous studies have found that multiple genes in the same biological pathways are often co-expressed. Furthermore, evidence has been found that these genes within a module may share a common genetic basis for the variations in their expression levels (Zhu et al., 2008; Chen et al., 2008). Although the hierarchical agglomerative clustering algorithm has been a popular method for visualizing the clustering structure in the genes, statistical methods that can take advantage of this clustering structure to identify causal genetic variants associated with gene modules were unavailable. In our experiments, using both simulated and yeast datasets, we demonstrate that our proposed method can be successfully applied to select genetic variants affecting multiple genes.

## 2. Background on Sparse Regression and Multi-task Learning

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 $\mathbf{x}_j = (x_j^1, \ldots, x_j^N)^T$. 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) (Tibshirani, 1996). Let $\mathbf{B} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ denote the $J \times K$ matrix of regression coefficients for all $K$ outputs. Then, lasso obtains $\hat{\mathbf{B}}^{\text{lasso}}$ by solving the following optimization problem:

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

where $\lambda$ is a tuning parameter that controls the amount of sparsity in the solution. Setting $\lambda$ to a large value leads to a smaller number of non-zero regression coefficients. Clearly, the standard lasso above offers no mechanism to explicitly couple the estimates of the regression coefficients for correlated output variables.

In multi-task learning, where the goal is to select input variables that are relevant to at least one task, an $L_1/L_2$ penalty has been used to take advantage of the relatedness of the outputs. An $L_2$ norm is applied to the regression coefficients $\boldsymbol{\beta}^j$ for all outputs for each input $j$, 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:

$$\hat{\mathbf{B}}^{L_1/L_2} = \text{argmin} \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \|\boldsymbol{\beta}^j\|_2 \tag{2}$$

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, and has been shown to be effective in this scenario (Obozinski et al., 2008; 2009). However, in many real-world applications, different outputs are related in a complex manner such as in gene expression data, where subsets of genes form functional modules. In this case, 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). In this tree $T$, each of the $K$ leaf nodes is associated with an output variable, and 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 node near the root represents relatively weaker correlations among the outputs in its subtree. This tree structure may be available as prior knowledge, or can be learned from data using methods such as a hierarchical agglomerative clustering algorithm. Furthermore, we assume that each node $v \in V$ is associated with 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 (2) 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 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 node 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

$$\hat{\mathbf{B}}^{\text{Tree}} = \text{argmin} \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \sum_j \sum_{v \in V} w_v \|\boldsymbol{\beta}_{G_v}^j\|_2, \tag{3}$$

where $\boldsymbol{\beta}_{G_v}^j$ is a vector of regression coefficients $\{\beta_k^j : k \in G_v\}$. Each group of regression coefficients $\boldsymbol{\beta}_{G_v}^j$ is

weighted with $w_v$ that reflects the strength of correlation within the group.

In order to define the weights $w_v$'s, we first associate each internal node $v$ of the tree $T$ with two quantities $s_v$ and $g_v$ that satisfy the condition $s_v + g_v = 1$, and then, define $w_v$'s in Equation (3) 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 the children of node $v$ separately, and 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 consists 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 (3) can be written as

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

where the weights are given as $w_1 = s_3$, $w_2 = s_3$, and $w_3 = g_3$. This is similar to the elastic-net penalty (Zou & Hastie, 2005), where $\beta^j_1$ and $\beta^j_2$ can be selected either jointly or separately according to the weights $s_3$ and $g_3$.

Given an arbitrary tree $T$, we recursively apply the similar operation starting from the root node towards the leaf nodes as follows:

$$\sum_j \sum_{v\in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 = \lambda \sum_j W_j(v_{\text{root}}), \tag{4}$$

where

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

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

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

The above weighting scheme extends the elastic-net-like penalty hierarchically. Thus, at each internal node $v$, a high value of $s_v$ encourages a separate selection of inputs for the outputs associated with the given node $v$, whereas high values of $g_v$ encourages a joint covariate selection across the outputs. 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 (2) that performs only a joint covariate selection for all outputs. The unit contour surfaces of various penalties for $\beta^j_1$, $\beta^j_2$, and $\beta^j_3$ with groups as defined in Figure 1 are shown in Figure 2.

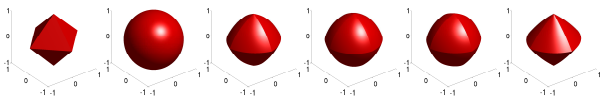



*Figure 2.* Unit contour surface for $\{\beta^j_1, \beta^j_2, \beta^j_3\}$ in various penalties, assuming the tree structure of output variables in Figure 1. (a) Lasso, (b) $L_1/L_2$, (c) tree-guided group lasso with $g_1 = 0.5$ and $g_2 = 0.5$, (d) $g_1 = 0.7$ and $g_2 = 0.7$, (e) $g_1 = 0.2$ and $g_2 = 0.7$, and (f) $g_1 = 0.7$ and $g_2 = 0.2$.

**Example 1.** *Given the tree in Figure 1, the tree-guided group-lasso penalty for the $j$th input in Equation (4) is given as follows:*

$$\begin{aligned} W_j(v_{root}) &= W_j(v_5) \\ &= g_{v_5} \cdot \|\boldsymbol{\beta}^j_{G_{v_5}}\|_2 + s_{v_5} \cdot (|W_j(v_4)| + |W_j(v_3)|) \\ &= g_{v_5} \cdot \|\boldsymbol{\beta}^j_{G_{v_5}}\|_2 \\ &\quad + s_{v_5} \cdot \Big(g_{v_4} \|\boldsymbol{\beta}^j_{G_{v_4}}\|_2 + s_{v_4}(|W_j(v_1)| + |W_j(v_2)|)\Big) \\ &\quad + s_{v_5}|\beta^j_3| \\ &= g_{v_5} \cdot \|\boldsymbol{\beta}^j_{G_{v_5}}\|_2 + s_{v_5} \cdot g_{v_4} \|\boldsymbol{\beta}^j_{G_{v_4}}\|_2 \\ &\quad + s_{v_5} \cdot s_{v_4}(|\beta^j_1| + |\beta^j_2|) + s_{v_5}|\beta^j_3|. \end{aligned}$$

**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:*

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

*Proof.* We assume an ordering of the nodes $\{v : k \in G_v\}$ along the path from the leaf $v_k$ to the root $v_{\text{root}}$, 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

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

□

Even if each output $k$ belongs to multiple groups associated with internal nodes $\{v : k \in G_v\}$ and appears multiple times in the overall penalty in Equation (4), Proposition 1 states that the sum of weights over all of the groups that contain the given output variable is always one. Thus, the weighting scheme in Equation (4) guarantees that the regression coefficients for all of the outputs are penalized equally. In contrast, group lasso with overlapping groups in Jenatton et al. (2009) used arbitrarily defined weights, which was empirically shown to lead to an inconsistent estimate. Another main difference between our method and the work in Jenatton et al. (2009) is that we take advantage of groups that contain other groups along the tree structure, whereas they tried to remove such groups as redundant in Jenatton et al. (2009).

Our proposed penalty function differs from the tree-structured penalty in Zhao et al. (2008) in that the trees are defined diffrently and contain different information. In the tree in our work, leaf nodes represent variables (or tasks) and internal nodes corresponed to clustering information. On the other hand, in Zhao et al. (2008), the variables themselves form a tree structure, where both leaf and internal nodes correspond to variables. Thus, the tree in Zhao et al. (2008) does not correspond to clustering structure but plays the role of prescribing which variables should enter the set of relevant variables first before other variables.

## 4. Parameter Estimation

In order to estimate the regression coefficients in tree-guided group lasso, we use an alternative formulation of the problem in Equation (3) that was previously introduced for group lasso (Bach, 2008), given as

$$\hat{\mathbf{B}}^{\text{Tree}} = \text{argmin} \quad \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) + \lambda \Big( \sum_j \sum_{v\in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 \Big)^2. \quad (5)$$

Since the $L_1/L_2$ norm in the above equation is a non-smooth function, it is not trivial to optimize it directly. We make use of the fact that the variational formulation of a mixed-norm regularization is equal to a weighted $L_2$ regularization (Argyriou et al., 2008) as follows:

$$\Big( \sum_j \sum_{v\in V} w_v \|\boldsymbol{\beta}^j_{G_v}\|_2 \Big)^2 \le \sum_j \sum_{v\in V} \frac{w_v^2 \|\boldsymbol{\beta}^j_{G_v}\|_2^2}{d_{j,v}},$$

where $\sum_j \sum_v d_{j,v} = 1, d_{j,v} \ge 0, \forall j, v$, and the equality holds for

$$d_{j,v} = \frac{w_v \|\boldsymbol{\beta}_{j,v}\|_2}{\sum_j \sum_{v\in V} w_v \|\boldsymbol{\beta}_{j,v}\|_2}. \quad (6)$$

Thus, we can re-write the problem in Equation (5) so that it contains only smooth functions, as follows:

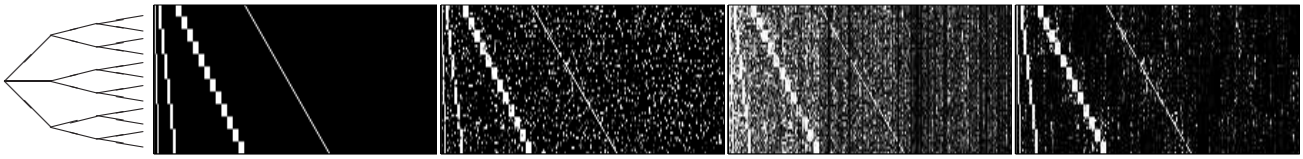
$$\begin{aligned}\hat{\mathbf{B}}^{\text{Tree}} = \text{argmin} & \sum_k (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k)^T \cdot (\mathbf{y}_k - \mathbf{X}\boldsymbol{\beta}_k) \\ & + \lambda \sum_j \sum_{v\in V} \frac{w_v^2 \|\boldsymbol{\beta}^j_{G_v}\|_2^2}{d_{j,v}} \quad (7) \\ \text{subject to} \quad & \sum_j \sum_v d_{j,v} = 1, \; d_{j,v} \ge 0, \quad \forall j, v,\end{aligned}$$

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, where the update equations for $d_{j,v}$'s are given as in Equation (6). Then, we hold $d_{j,v}$'s as constant, and optimize for $\boldsymbol{\beta}_k$'s. We differentiate the objective in Equation (7) with respect to $\boldsymbol{\beta}_k$'s, set it to zero, and solve for $\boldsymbol{\beta}_k$'s to obtain the update equation:

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

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.

Finally, the regularization parameter $\lambda$ can be selected using a cross-validation.



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

*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.

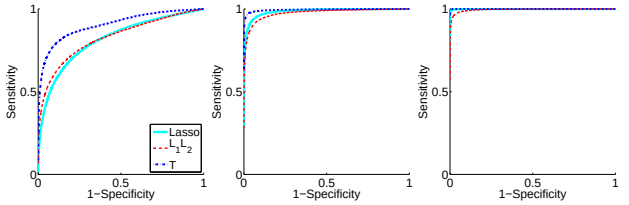
## 5. Experiments

We demonstrate the performance of our method on simulated datasets and a yeast dataset of genotypes and gene expressions, and compare the results 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 covariates.

### 5.1. Simulation Study

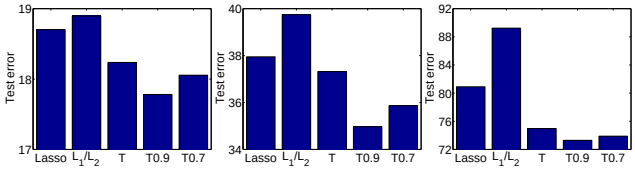
We simulate data using the following scenario analogous to genetic association mapping. We simulate $(\mathbf{X}, \mathbf{Y})$ with $K = 60$, $J = 200$ and $N = 150$ for the training set as follows. We first generate the inputs $\mathbf{X}$ by sampling each element in $\mathbf{X}$ from a uniform distribution 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, and select the non-zero elements of $\mathbf{B}$ so that they correspond to the groupings in the sparsity structure given by this tree. The tree is shown in Figure 3(a), where we only draw the top three levels to avoid clutter. Figure 3(b) shows the selected 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 $N(0, 1.0)$.

We fit lasso, the $L_1/L_2$-regularized regression, and our method to the dataset simulated 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 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 information across the outputs regardless of the sparsity structure. 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) was available as prior knowledge, it is visually clear from Figure 3(e) that our method is able to suppress false positives and recover the true underlying sparsity structure significantly better than other methods.




(a) (b) (c)

*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) $\beta_k^j = 0.6$.




(a) (b) (c)

*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) $\beta_k^j = 0.6$.

In order to systematically evaluate the performance of 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 a sparsity structure is significant.

We compare the performance of the different methods in terms of prediction errors, 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 take advantage of the structure in the 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 $g_v = 1 - h'_v$ and $s_v = h'_v$. Since the tree obtained in this manner represents a noisy realization of the true underlying tree structure, we discard the nodes for weak correlations 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 structure, and gives lower prediction errors.

### 5.2. Analysis of Yeast Data

We analyze the genotype and gene expression data of 114 yeast strains (Zhu et al., 2008) 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 clustering structure over correlated genes is not directly available as prior knowledge. Instead, 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 identify SNPs (inputs) whose variations induce significant variations in 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 groups of genes that are co-expressed.

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 running the hierarchical agglomerative 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 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 in Figure 6(c) are not sparse across genes, and tend to form vertical stripes of non-zero regression coefficients. 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.

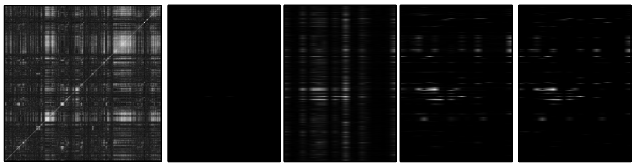



*Figure 6.* Results for the yeast dataset. (a) Correlation matrix of the gene expression data, where rows and columns are reordered after applying agglomerative hierarchical clustering. Estimated regression coefficients are shown for (b) lasso, (c) $L_1/L_2$, (d) tree-guided group lasso with $\rho =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

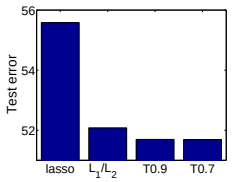
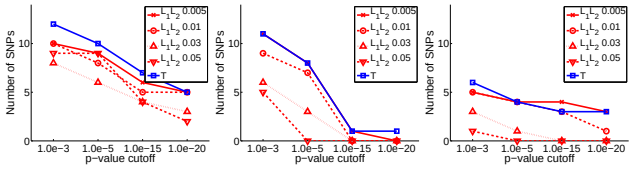



*Figure 7.* Prediction errors for the yeast dataset.

Given the estimates of $\mathbf{B}$ in Figure 6, we look for an enrichment of GO categories among the genes with non-zero regression 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 $\mathbf{B}$. Because the estimates of the $L_1/L_2$-regularized




(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.

method are not sparse across genes, 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 $\beta_k^j$ above the threshold. On the other hand, for our method, we use all of the genes with non-zero elements in $\mathbf{B}$ for each SNP.

In Figure 8, we show the number of SNPs with significant enrichments at different $p$-value cutoffs for subcategories within each of the three broad GO categories, biological processes, molecular functions, and cellular components. For example, within biological processes, SNPs were found to be enriched for GO terms such as mitocondrial translation, amino acid biosynthetic process, and carboxylic acid metabolic process. 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 tree-guided group lasso that finds a sparse estimate of regression coefficients while taking into account the structure among outputs given by a tree. We demonstrated our method using simulated and yeast datasets.

## Acknowledgements

EPX is supported by grants ONR N000140910758, NSF DBI-0640543, NSF CCF-0523757, NIH 1R01GM087694, and an Alfred P. Sloan Research Fellowship.

## References

Argyriou, A., Evgeniou, T., and Pontil, M. Convex multi-task feature learning. *Machine Learning*, 73 (3):243–272, 2008.

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

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

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

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

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

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

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

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

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

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

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

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

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