BIOINFORMATICS ORIGINAL PAPER
Vol. 22 no. 3 2006, pages 332–340
doi:10.1093/bioinformatics/bti792

*Gene expression*

# A bi-dimensional regression tree approach to the modeling of gene expression regulation

Jianhua Ruan[1] and Weixiong Zhang[1,2,*]

[1]Department of Computer Science and Engineering and [2]Department of Genetics, Washington University in St Louis, St Louis, MO 63130, USA

Received on March 16, 2005; revised on October 10, 2005; accepted on November 15, 2005
Advance Access publication November 22, 2005
Associate Editor: Chris Stoeckert

**ABSTRACT**

**Motivation:** The transcriptional regulation of a gene depends on the binding of *cis*-regulatory elements on its promoter to some transcription factors and the expression levels of the transcription factors. Most existing approaches to studying transcriptional regulation model these dependencies separately, i.e. either from promoters to gene expression or from the expression levels of transcription factors to the expression levels of genes. Little effort has been devoted to a single model for integrating both dependencies.

**Results:** We propose a novel method to model gene expression using both promoter sequences and the expression levels of putative regulators. The proposed method, called bi-dimensional regression tree (BDTree), extends a multivariate regression tree approach by applying it simultaneously to both genes and conditions of an expression matrix. The method produces hypotheses about the condition-specific binding motifs and regulators for each gene. As a side-product, the method also partitions the expression matrix into small submatrices in a way similar to bi-clustering. We propose and compare several splitting functions for building the tree. When applied to two microarray datasets of the yeast *Saccharomyces cerevisiae*, BDTree successfully identifies most motifs and regulators that are known to regulate the biological processes underlying the datasets. Comparing with an existing algorithm, BDTree provides a higher prediction accuracy in cross-validations.

**Availability:** The software is available upon request from the authors.

**Contact:** zhang@cse.wustl.edu

**Supplementary information:** http://cic.cs.wustl.edu/bdtree/

## 1 INTRODUCTION

The complex function of a living cell is controlled by regulating the expression of specific genes at several levels. One of the most important and best understood regulation mechanisms is at the transcriptional level, where the expression of a gene is mediated by the binding of transcription factors (TFs) to specific DNA sequences in the promoter region of the gene. Two basic assumptions are often made when studying transcriptional regulation: first, the expression of a gene is determined by the binding sites of specific TFs on its promoter; second, the expression of a gene is a function of the concentration of specific TFs around its promoter. Based on the two assumptions, two distinct classes of approaches have been proposed in studying transcriptional regulation.

*To whom correspondence should be addressed.

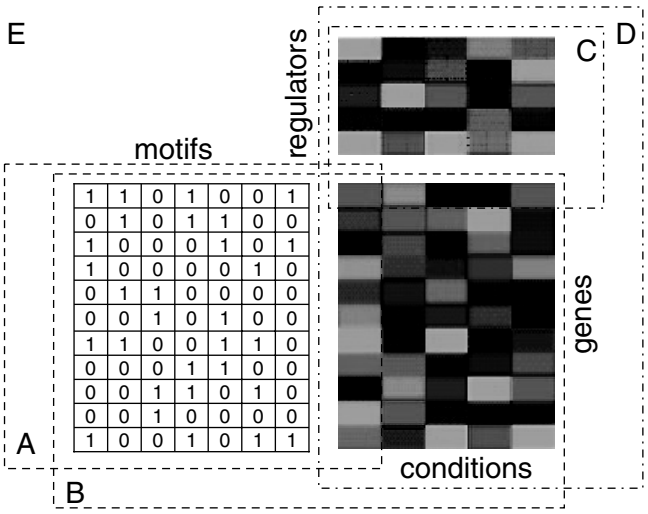


**Fig. 1.** Relationships between our method and previous methods. The bottom-right matrix represents gene expression levels. The bottom-left matrix represents motif occurences on promoter sequences. The top matrix is the expression levels of regulators. Box A, the expression levels of multiple genes under a single condition are modeled by the motifs on their promoters (e.g. Bussemaker *et al*., 2001). Box B, the expression levels of multiple genes under multiple conditions are modeled by the motifs on their promoters (Phuong *et al*., 2004). Box C, the expression levels of a single gene under multiple conditions are modeled by the expression levels of putative regulators (Soinov *et al*., 2003). Box D, the expression levels of multiple genes under multiple conditions are modeled by the expression levels of putative regulators (Segal *et al*., 2003). Box E, the expression levels of multiple genes under multiple conditions are modeled by the motifs on their promoters and the expression levels of putative regulators (Middendorf *et al*., 2004 and our method).

The first class of approaches attempted to build quantitative or qualitative models to associate gene expression levels with putative binding motifs on their promoter sequences (Fig. 1, boxes A and B). Several approaches of this type have been proposed within the classification and regression framework. Bussemaker *et al*. (2001) and others (Keles *et al*., 2002; Conlon *et al*., 2003) modeled the expression levels of genes as a linear regression of putative binding motifs and applied feature selection techniques to find the most significant motifs. We (Ruan and Zhang, 2004) and Hu *et al*. (2000) used decision trees to find motif combinations that best

 

separate two sets of genes. Beer and Tavazoie (2004) built probabilistic graphical models, e.g. Bayesian networks, to explain gene expression patterns from motifs. Phuong *et al*. (2004) applied multivariate regression trees to model the transcriptional regulation of gene expression over several time points simultaneously.

The second class of approaches has been proposed to model gene expression levels from the expression levels of other genes, i.e. TFs and other regulators (Fig. 1, boxes C and D). For example, Soinov *et al*. (2003) used a decision tree to identify possible regulators for several cell-cycle genes individually. Segal *et al*. (2003) proposed a more sophisticated procedure suitable for whole-genome analysis. The method first clusters genes according to their expression patterns, and then builds a regression tree for each cluster to represent their common regulation program. The procedure then iteratively refines the clusters and the trees.

Middendorf *et al*. (2004) recently introduced a method that combines the previous two classes of approaches. Their method models gene expression levels from both putative binding motifs on promoter sequences and the expression levels of putative regulators (Fig. 1, box E). Here we propose a method that also falls into this category. Although our method has the same schematic representation as theirs, the underlying modeling rationales are very different, which we will compare in detail in Section 4.

Our method, called bi-dimensional regression tree or BDTree for short, is an extension to the multivariate regression tree approach of Segal (1992) and Phuong *et al*. (2004). Breiman *et al*. (1984) first introduced the univariate regression tree approach to recursively partition instances into groups, where the instances in each group have similar attribute values and responses. Segal (1992) extended the method to handle multiple responses, so that the instances in each group have a similar pattern of responses across multiple conditions. The basic idea of our method, as suggested by its name, is to extend the multivariate regression tree approach to both dimensions of the expression matrix (Fig. 1). On one dimension, each gene is treated as an instance, where the attributes are the binding motifs on its promoter sequence and the responses are its expression levels across the conditions. Genes are partitioned so that those in the same subset have common binding motifs and similar expression patterns across the conditions. On the other dimension, each condition is treated as an instance, where the attributes are the expression levels of candidate regulators under the condition and the responses are the expression levels of genes under that condition. Conditions are partitioned so that the expression levels of a gene under each subset of conditions are similar.

The way of partitioning genes and conditions in BDTree is analogous to bi-clustering (Cheng and Church, 2000). However, the partitioning in BDTree is supervised by some intrinsic attributes of the genes and conditions, i.e. the binding motifs on gene promoters and the expression levels of regulators under the conditions. In contrast, in bi-clustering, the partitioning is unconstrained by those attributes. As a result, the model learned by BDTree is both exploratory and predictive. It suggests a set of testable hypotheses of condition-specific binding motifs and regulators for the genes in each cluster, and can also be used to predict the expression levels of unseen genes under unseen conditions, given appropriate attributes of the genes and conditions.

The rest of the paper is organized as follows. The next section first introduces the univariate regression tree and its multivariate extension, and then describes the bi-dimensional multivariate regression tree approach. Section 3 presents some experimental results from applying the method to the yeast cell-cycle and stress response data. In Section 4 we discuss the differences and relationships between BDTree and several related methods.

## 2 ALGORITHM

### 2.1 Univariate regression trees

Here we give a brief overview of the univariate regression tree method and refer the reader to Breiman *et al*. (1984) for details. Suppose that there are $p$ attributes $X_1, X_2, \ldots, X_p$ and a response $Y$. The values of the attributes and responses are observed for $m$ instances: $\{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$, for $i = 1, \ldots, m$. Here the responses are real values. We restrict all attributes to be real values for the convenience of the discussion, although the method can handle categorical or mixed values. In the context of transcriptional regulation, each gene is an instance, the attributes are motifs and the response is the gene expression level under a single condition (Fig. 1, box A).

In the classic CART (Classification Analysis and Regression Tree) program of Breiman *et al*. (1984), a greedy search algorithm is used to construct a binary regression tree. The basic algorithm is as follows.

(1) Initially there is only the root node containing all instances.

(2) If the current node has not met the stopping criterion, examine every possible binary split of the instances within the node based on each attribute $X_i$, $i = 1, \ldots, p$, such that the attribute values for all the instances in one subset are smaller than those in the other subset.

(3) Choose the best split to maximize an objective function and create two child nodes for the current node.

(4) Repeat steps 2 and 3 for each child node.

To build a regression tree, three rules need to be specified: a splitting rule that defines the best split, a stopping rule that determines when the splitting should terminate and a third rule to prune certain branches of the tree after the tree is built. Here we only discuss the splitting rule, while the other two will be discussed after we introduce the BDTree method.

The goal of a split is to produce child nodes as homogeneous as possible with respect to the responses. A frequently used criterion is the least-square rule that aims at minimizing the sum-of-squares of responses within each node. Let $r$ denote a node of the tree and $n_r$ denote the number of instances in $r$. That is, $r$ contains a subset of the indices $\{1, \ldots, m\}$. The within-node sum-of-squares is given by

$$\mathrm{SS}(r) = \sum_{i \in r} (y_i - \bar{y})^2, \tag{1}$$

where $\bar{y} = \sum_{i \in r} y_i / n_r$. The gain of a split that partitions $r$ into two child nodes $r_1$ and $r_2$ is given by

$$G(r, r_1, r_2) = \mathrm{SS}(r) - \mathrm{SS}(r_1) - \mathrm{SS}(r_2). \tag{2}$$

The best split is determined by an attribute $X_i$ and a threshold $T$ such that $G(r,r_1,r_2)$ is maximized and that the value of $X_i$ for every instance in $r_1$ is less than $T$ while that for every instance in $r_2$ is no less than $T$. To find the best split, all possible thresholds for each attribute are tested and the split with the highest gain is chosen.

### 2.2 Multivariate regression trees

It is not uncommon to encounter domains where the responses are observed under multiple conditions, i.e. the response of an instance is also a vector: $\mathbf{y}_i = (y_{i1}, \ldots, y_{in})$, where $n$ is the number of conditions and $y_{ij}$ is the response of the $i$-th instance under the $j$-th condition. For example, in DNA microarrays, gene expression levels are typically recorded for several time points or experimental conditions.

A naive solution for this situation is to build a regression tree for each condition separately. However, it is difficult to combine multiple trees. Segal (1992) introduced a multivariate regression tree method to construct a single tree to model multiple responses simultaneously. He generalized the within-node sum-of-squares in Equation (1) as follows:

$$\mathrm{SS1}(r) = \sum_{i \epsilon r} (\mathbf{y}_i - \bar{\mathbf{y}}) \mathbf{V}^{-1} (\mathbf{y}_i - \bar{\mathbf{y}})^t, \tag{3}$$

where $\mathbf{y}_i$ is the vector of responses for the $i$-th instance, $\mathbf{V}$ is the model covariance matrix of $\mathbf{y}_i$, and $\bar{\mathbf{y}}$ is the average of $\mathbf{y}_i$ within node $r$. With SS($r$) defined, the gain function remains the same as in Equation (2) and the recursive algorithm proceeds to split the instances as in the case with a single response.

Phuong *et al*. (2004) applied the multivariate regression tree method to gene expression data by treating genes as instances, where the numbers of occurrences of motifs in promoters are the attribute values and the expression levels at different conditions are the multivariate responses (Fig. 1, box B). As noted by Segal (1992) and Phuong *et al*. (2004), the multivariate regression tree method is intermediary between classification and clustering. The responses of different instances can be written as a matrix $\mathcal{Y} = (y_{ij})$, where $i$ is the index of an instance and $j$ is the index of a condition. The multivariate regression tree partitions the matrix into submatrices, where each submatrix contains all the columns (conditions) but only some rows (genes) of the original matrix. Therefore, clustering is achieved directly when instances with similar patterns of responses are grouped together.

### 2.3 Bi-dimensional multivariate regression trees

Now consider a multivariate response situation where each condition can also be described by a set of attributes, $W_1, \ldots, W_q$, just as the instances can be described by attributes $X_1, \ldots, X_p$. In this case, the response matrix $\mathcal{Y} = (y_{ij})$ can be transposed, and the regression problem can be defined for the conditions. Each condition is now treated as an instance. The observations for the $j$-th condition can be written as $(\mathbf{w}_j, \mathbf{y}_j)$, where $\mathbf{w}_j = (w_{j1}, \ldots, w_{jq})$ and $\mathbf{y}_j = (y_{1j}, \ldots, y_{nj})$, with $n$ being the number of conditions and $q$ the number of attributes of a condition. A regression tree can then be learned to model the condition data. For clarity, we call the conditions column instances and the instances row instances, since they correspond to the columns and rows of the response matrix, respectively. Accordingly, we call $\mathrm{X}_1, \ldots, \mathrm{X}_p$ row attributes, and $W_1, \ldots, W_q$ column attributes.

In the case of gene expression analysis, the column attributes that can be used to describe each condition are the expression levels of a set of candidate regulators under that condition (Fig. 1, boxes C and D). Therefore, a regression tree built from the column instances explains the expression levels of genes under different conditions with the expression levels of selected regulators. The motivating assumption is that the expression level of a gene depends on the expression levels of its regulators.

The goal of our method is to model the responses using both row attributes and column attributes, i.e. to find the row attributes and column attributes that can explain the responses. In the regression tree framework, this corresponds to recursively partitioning the response matrix horizontally according to row attributes and vertically according to column attributes. The objective is to make the submatrices in child nodes as homogeneous as possible with respect to responses.

Formally, the input to the algorithm includes a response matrix $\mathcal{Y} = (y_{ij})$, the associated row attribute matrix $\mathcal{X} = (\mathbf{x}_i)$ and column attribute matrix $\mathcal{W} = (\mathbf{w}_j)$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ is the set of attributes for the $i$-th row, $i = 1, \ldots, m$, and $\mathbf{w}_j = (w_{1j}, \ldots, w_{qj})$ is the set of attributes for the $j$-th column, $j = 1, \ldots, n$. For example, the three matrices in Figure 1 representing motif scores, gene expression levels and regulator expression levels correspond to $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{W}$, respectively. Each split divides the response matrix $\mathcal{Y}$ vertically or horizontally. As a result, each node of the regression tree contains a submatrix of $\mathcal{Y}$ and the associated row and column attributes.

To facilitate subsequent discussions, we define some symbols and notations. Let $t(r, c)$ denote a node of a tree, where $r$ contains a subset of the row indices $\{1, \ldots, m\}$ and $c$ contains a subset of the column indices $\{1, \ldots, n\}$. When there is no confusion, we abbreviate $t(r, c)$ as $t$ or $(r, c)$. Let $n_r$ and $n_c$ denote the number of rows and the number of columns of the response matrix in node $t$, respectively. Let $\mathbf{Y}^t$ denote the response matrix in $t$. Let $\bar{\mathbf{y}}^t_{r*}$ denote the vector of average responses across all columns in $t$ and $\bar{\mathbf{y}}^t_{*c}$ the vector of average responses across all rows in $t$. Furthermore let $\bar{y}^t_{i*}$ denote the $i$-th element of $\bar{\mathbf{y}}^t_{r*}$ and $\bar{y}^t_{*j}$ the $j$-th element of $\bar{\mathbf{y}}^t_{*c}$. Let $\bar{y}^t$ denote the average response of all rows and columns in $t$. The superscript $t$ is dropped when there is no confusion.

A critical issue in our algorithm is to design a measure to reflect the homogeneity of the response matrix on both dimensions. For a node $t(r, c)$, a good candidate is the sum of squared residues introduced by Cheng and Church (2000) for bi-clustering as shown in the following equation:

$$\mathrm{SS2}(r, c) = \sum_{i \in r, j \in c} (y_{ij} - \bar{y}_{i*} - \bar{y}_{*j} + \bar{y})^2, \tag{4}$$

where the superscript $t$ on variables has been dropped. Equation (4) can also be rewritten as

$$\mathrm{SS2}(r, c) = n_r n_c (s^2(\mathbf{Y}) - s^2(\bar{\mathbf{y}}_{r*}) - s^2(\bar{\mathbf{y}}_{*c})), \tag{5}$$

where $s^2(\mathbf{Y}) = \sum_{ij} (y_{ij} - \bar{y})^2 / n_r n_c$ is the sample variance of $\mathbf{Y}$, $s^2(\bar{\mathbf{y}}_{r*}) = \sum_i (\bar{y}_{i*} - \bar{y})^2 / n_r$ is the sample variance of the vector $\bar{\mathbf{y}}_{r*}$ and $s^2(\bar{\mathbf{y}}_{*c}) = \sum_j (\bar{y}_{*j} - \bar{y})^2 / n_c$ is the sample variance of the vector $\bar{\mathbf{y}}_{*c}$. It can be seen that SS2($r$,$c$) is minimal if the variance of the matrix can be explained by the variance of the row averages and the variance of the column averages. The expected value of SS2($r, c$) is a function of $n_r$ and $n_c$ as shown in the following equation:

$$E(\mathrm{SS2}(r, c)) = (n_r n_c - n_r - n_c) \delta^2(\mathbf{Y}), \tag{6}$$

where $\delta^2(\mathbf{Y})$ is the population variance of the responses. A proof of Equation (6) by the central limit theorem is provided on the Supplementary website http://cic.cs.wustl.edu/bdtree/

With SS2($r$,$c$) defined, the gain of a split can be calculated the same as in Equation (2). Note that when a row split is taken, the average responses across columns are not affected and vice versa for column splits. Therefore, when the response matrix is split horizontally, with $r_1$ and $r_2$ rows in each child node, respectively, the gain can be computed by

$$G2_r(r, r_1, r_2) = n_c(n_{r_1}s^2(\bar{\mathbf{y}}^1_{*c}) + n_{r_2}s^2(\bar{\mathbf{y}}^2_{*c}) - n_r s^2(\bar{\mathbf{y}}_{*c})), \quad (7)$$

where $\bar{\mathbf{y}}^i_{*c}$, $i = 1,2$, is the $1 \times n_c$ vector of average responses across all rows in child node $t(r_i,c)$. Similarly, when the response matrix is split vertically, with $c_1$ and $c_2$ columns in each child node, respectively, the gain can be computed by

$$G2_c(c, c_1, c_2) = n_r(n_{c_1}s^2(\bar{\mathbf{y}}^1_{r*}) + n_{c_2}s^2(\bar{\mathbf{y}}^2_{r*}) - n_c s^2(\bar{\mathbf{y}}_{r*})), \quad (8)$$

with $\bar{\mathbf{y}}^i_{r*}$ similarly defined.

Given Equations (7) and (8) for calculating gains, the algorithm tests all possible splits on rows and columns, and selects the one with the highest gain. Therefore, the algorithm automatically determines whether the split should be done on rows or columns. However, when the shape of the initial response matrix $\mathcal{Y}$ is skewed, the gain function prefers the split on the longer side. For example, if $m \gg n$ (which is normally the case in gene expression data), the gain function prefers to split on columns to produce even narrower submatrices. This is because the expected gain for a split on one dimension is proportional to the length of the other dimension (see the Supplementary website for a proof):

$$\begin{cases} E(G2_r(r, r_1, r_2)) \simeq n_c\delta^2(\mathbf{Y}); \\ E(G2_c(c, c_1, c_2)) \simeq n_r\delta^2(\mathbf{Y}). \end{cases} \quad (9)$$

To scale down this systematic bias, we define and calculate the following adjusted gains instead as follows:

$$\begin{cases} G2'_r(r, r_1, r_2) = G2_r(r, r_1, r_2)/n_c; \\ G2'_c(c, c_1, c_2) = G2_c(c, c_1, c_2)/n_r. \end{cases} \quad (10)$$

A problem with the homogeneity measure by SS2 is that the produced clusters are often not tight. For example, the SS2 measure of a matrix is zero if all rows (or columns) differ only by some constant values, i.e. $\mathbf{y}_i - \mathbf{y}_j = a_{ij}\mathbf{i}$, for all $i, j$, where $\mathbf{y}_i$ and $\mathbf{y}_j$ are the $i$-th and $j$-th rows of $\mathbf{Y}$, respectively, and $\mathbf{i} = (1, \ldots, 1)$. Consider a matrix

$$\begin{vmatrix} -3 & -2 & -1 \\ -1 & 0 & 1 \\ 1 & 2 & 3 \end{vmatrix},$$

where each row is a gene and the values are the log ratios of expression levels under different conditions. The cluster does not seem to provide any biological significance, despite a perfect score. To deal with this problem, we define different sum-of-squares for rows and columns as follows:

$$\begin{cases} \mathrm{SS3}_r(r, c) = \sum_{i \in r, j \in c} (y_{ij} - \bar{y}_{*j})^2 = n_r n_c(s^2(\mathbf{Y}) - s^2(\bar{\mathbf{y}}_{*c})); \\ \mathrm{SS3}_c(r, c) = \sum_{i \in r, j \in c} (y_{ij} - \bar{y}_{i*})^2 = n_r n_c(s^2(\mathbf{Y}) - s^2(\bar{\mathbf{y}}_{r*})). \end{cases} \quad (11)$$

The gain for a row or column split is defined correspondingly as shown in the following equation:

$$\begin{cases} G3_r(r, r_1, r_2) = \mathrm{SS3}_r(r, c) - \mathrm{SS3}_r(r_1, c) - \mathrm{SS3}_r(r_2, c); \\ G3_c(c, c_1, c_2) = \mathrm{SS3}_c(r, c) - \mathrm{SS3}_c(r, c_1) - \mathrm{SS3}_c(r, c_2). \end{cases} \quad (12)$$

$\mathrm{SS3}_r$ or $\mathrm{SS3}_c$ is equivalent to the sum of SS for each column or row, respectively, according to Equation (1). This measurement requires a good cluster to be coherent on at least one dimension of the matrix. Furthermore, the different gain functions defined for row and column splits enforce that a good row split must improve the homogeneity along the columns, and a good column split must improve the homogeneity along the rows. This enforcement is consistent with biological intuitions. For example, when a motif is used to separate two sets of genes, the genes within each set should have similar expression levels under the same conditions, while the expression levels under different conditions may be different.

Interestingly, the expected values of $G3$ are the same as in Equation (9). Therefore, a systematic bias between $G_r$ and $G_c$ still exists, and the adjusted gain can be defined similarly as in Equation (10).

Given the gain functions, the algorithm proceeds the same as in the case of a single response described in Section 2.1, except the step 2:

(2) If the current node has not met the stopping criterion, examine every possible binary split of the row instances or the column instances within the node, based on each row attribute $X_i$, $i = 1, \ldots, p$, or column attribute $W_j$, $j = 1, \ldots, q$, respectively, such that the attribute values for all the instances in one subset are smaller than those in the other subset.

When analyzing gene expression data, the column attributes are regulators, which may also appear in the list of genes in the rows. The algorithm does not allow a regulator to be the splitting attribute of a node that contains the regulator itself, since a gene's expression level can always be used to predict its own expression.

To prevent the tree from over-fitting the data, several parameters are implemented to control the tree size, including the minimum gain required to split a node, the minimum numbers of rows and columns within a leaf node, and the maximum number of leaf nodes. In addition, a post-pruning procedure can be performed with a separate test set, where an internal node is converted to a leaf node if by doing so the prediction accuracy on the test data does not decrease.

To predict a response, the corresponding row and column attributes are compared with the threshold values at each tree node and a branch is taken according to the result of the comparison at each step. Starting from the root node, the algorithm will always end at a terminal node $t$. The average value of the elements in $\mathbf{Y}^t$ is used as the predicted value.

## 2.4 Cross-validation and functional analysis

The prediction accuracy of BDTree is estimated by cross-validations. The procedure of cross-validation in BDTree is slightly different from that in a one-dimensional method. Given a training dataset, we denote the set of row instances as $r$ and column instances as $c$. To perform a 10-fold cross-validation, $r$ and $c$ are both randomly divided into 10 subsets of roughly equal size, denoted by $r_1, \ldots, r_{10}$ and $c_1, \ldots, c_{10}$, respectively. Every time a submatrix containing nine subsets of the rows and nine subsets of the columns, $(r \backslash r_i) \times (c \backslash c_i)$, is used for training, while three submatrices, $r_i \times (c \backslash c_i)$, $(r \backslash r_i) \times c_i$ and $r_i \times c_i$ are used for testing, for $i = 1, \ldots, 10$. The mean squared errors or the correlation coefficient between the predicted and actual values are calculated as a measure of accuracy. In addition, accuracies can be calculated for the three testing submatrices

separately, corresponding to the prediction accuracy for unseen rows, unseen columns and unseen rows plus unseen columns.

In the case of analyzing gene expression data, each leaf node of the tree contains a subset of the genes and a subset of the conditions. To determine the functional relevance of the splits, we calculate the enrichment of gene ontology (GO) terms (Harris *et al*., 2004) within each leaf node. When possible, we also group the experimental conditions into categories and calculate the enrichment of particular categories within each leaf node. The significance of enrichment is measured by an accumulative hypergeometric test, and the *p*-values are adjusted by Bonferroni corrections for multiple tests (Altman, 1991).

When a tree is built, it automatically selects a set of attributes from the row attributes and column attributes to explain the pattern of responses. Intuitively, the gain of splitting a node with a certain attribute can be used as a measure of the importance of the attribute. However, the tree only selects the attribute with the highest gain at each step, while ignoring all the others. Phuong *et al*. (2004) provide a better method to measure the importance of all attributes, based on surrogate splits. We adopt the same idea, but rank row and column attributes separately.

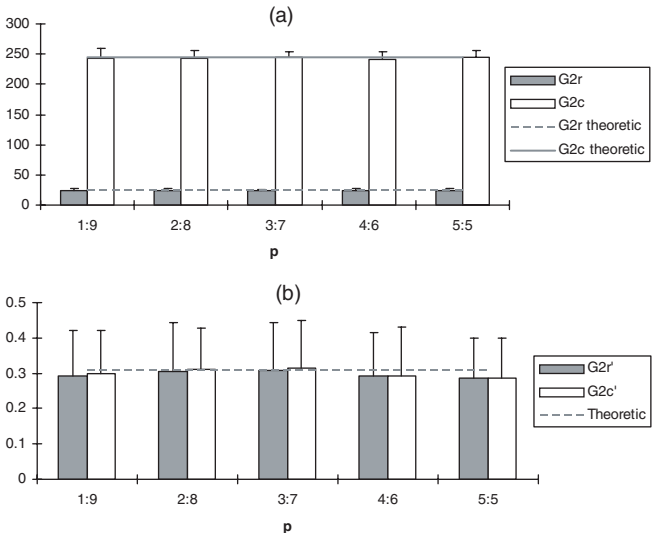


**Fig. 2.** Simulated gains. (**a**) Unadjusted gains on an i.i.d. matrix; (**b**) Adjusted gains on a real matrix. *P* is the relative sizes of the two submatrices after splitting.

## 3 RESULTS

### 3.1 Datasets

To evaluate and to demonstrate the strength of our method in identifying real motifs and regulators, we tested it on two microarray sets for *Saccharomyces cerevisiae*. The cell-cycle dataset consists of the expression levels of 800 cell-cycle related genes measured under 77 different time points of cell cycles in several experiments (Spellman *et al*., 1998). The stress response dataset includes gene expression data collected under 173 different stress conditions as in Gasch *et al*. (2000). As in Middendorf *et al*. (2004), we selected 1411 genes from the stress dataset, which includes 469 highly variant genes and 1250 genes that are part of the 17 clusters identified by Gasch *et al*. (2000). We downloaded the normalized gene expression log ratios from SGD (Dwight *et al*., 2004).

We used the set of 466 candidate regulators compiled by Segal *et al*. (2003) as column attributes. These include both TFs and signaling molecules that may have transcriptional impact. We combined three types of row attributes: 356 position-specific weight matrices (PSWMs) for known motifs and putative ones derived computationally from MIPS functional categories (Pilpel *et al*., 2001), binding data of 204 TFs measured under various environmental conditions (Harbison *et al*., 2004) and 615 overrepresented *k*-mers ($5 \leq k \leq 7$) discovered by a steganalysis-based motif-finding algorithm called WordSpy developed in our lab (Wang *et al*., 2005). We used RSA tools (van Helden, 2003) to retrieve up to 500 bp of intergenic sequences upstream of each gene's start codon as its promoter and searched both strands for the appearance of motifs and *k*-mers. The program ScanACE (Roth *et al*., 1998) was used to scan each promoter, and the highest score for each motif was recorded as its attribute value. For each *k*-mer, its number of occurrences in a promoter sequence was used as its score. The list of genes and attributes is available on the Supplementary website.

### 3.2 Simulation study of gain functions

We have shown in Section 2.3 that for both SS2 and SS3, the expected gain resulted from a random split is correlated with the size of the dimension that is unsplit. Here, we use a simulation to show that this is also true in practice.

We first considered the case where gene expression data are independent and identically distributed (i.i.d.). We randomly shuffled the yeast cell-cycle gene expression matrix, which was then split into two submatrices by randomly dividing its rows or columns into two sets. The relative sizes of the two submatrices varied from 1:9 to 5:5. Figure 2a shows the average gains of 1000 random splits on rows or columns, calculated using SS2. As shown, the average gains agree with the theoretical results almost perfectly, regardless of the relative sizes of the two submatrices. The gains of column splits are much larger than those of row splits, which justifies the adjustment of gains by Equation (10). Next, we repeated the experiments on the real yeast cell-cycle expression matrix to which the i.i.d. assumption does not hold. The average gains are close to the theoretical values, although the agreement is not as well as in the i.i.d. case. As shown in Figure 2b, the adjusted gains resulted from row splits or column splits that have similar means and standard deviations. The results using SS3 or the stress response dataset are similar.

### 3.3 Model accuracy

To evaluate the performance of our method, we applied it to the yeast stress response dataset and conducted 10-fold cross-validations. We calculated the correlation coefficients between the predicted and the actual values as a measure of accuracy.

In the first set of experiments, we analyzed the effect of the choices of parameters. We have shown by simulation that the adjustment of gains is necessary to eliminate the systematic bias between row and column splits. Indeed, the model built with adjusted gains has a higher cross-validation accuracy than that with unadjusted gains (0.54 versus 0.43). We also found that the tree based on SS3 has better accuracy than that based on SS2 (0.54 versus 0.45). In addition, the prediction accuracy for unseen genes (0.56) is slightly higher than that for unseen conditions (0.52) or unseen genes plus unseen conditions (0.51).

**Table 1.** Confusion matrix

| | | Predicted by BDTree (%) | | | Predicted by Geneclass (%) | | |
|---|---|---|---|---|---|---|---|
| | | Down | Baseline | Up | Down | Baseline | Up |
| TRUE | Down | 15.2 | 7.1 | 2.9 | 16.5 | 8.9 | 1.5 |
| | Baseline | 7.0 | 34.5 | 9.2 | 9.3 | 32.4 | 6.3 |
| | Up | 2.4 | 8.4 | 13.2 | 2.8 | 9.9 | 12.0 |

We compared the accuracy of BDTree to the *k*-nearest neighbor (KNN) method. With the KNN method, the expression level of a gene at a certain condition was predicted by the average expression level of the *k*-nearest genes under the *k*-nearest conditions (best $k = 20$ in our experiment), where the distance between genes or conditions was defined by the Euclidean distance of their normalized attribute vectors. We chose KNN as a baseline classifier because it is relatively easy to implement a bi-dimensional counterpart of our algorithm. The cross-validation accuracy of BDTree (0.54) is much higher than that of the KNN method (0.37).

Second, we considered the case where BDTree was grown using row attributes only. This is equivalent, in spirit, to the method of Phuong *et al*. (2004). Since the expression matrix was only partitioned horizontally, the method was unable to predict expression levels under unseen conditions. Therefore, we conducted cross-validations only on unseen genes. The correlation coefficient obtained by this method was similar to our full model where both row and column attributes are used (0.57 versus 0.56), which means that our method did not lose any information in row attributes even though column attributes were used together.

Finally, we compared our method with the Geneclass method (Middendorf *et al*., 2004), which is similar to ours in that it can also predict gene expression levels using both regulators and binding motifs. One problem when comparing with their results, however, is that their method is only applicable to pre-discretized expression levels, while our method can be applied to real expression values. Therefore, we discretized the expression data into three levels (up, down and baseline) as in their method and obtained a confusion matrix for our predictions as shown in Table 1. It turned out that the two methods have similar prediction accuracies (63% for ours versus 61% for theirs) using discretization. On the other hand, their method uses a technique called boosting, which greatly improves prediction accuracy, but reduces interpretability of models. Besides not requiring discretization, our method also has several additional advantages that will be discussed in Section 4.

### 3.4 Biological interpretation and functional analysis

Figure 3 shows the regression tree learned from the yeast stress response data and the gene expression matrix reorganized according to the tree. The interpretation of the tree is straightforward. Each oval represents a row attribute (prefixed by 'h.') for horizontal splitting, or column attribute (prefixed by 'v.') for vertical splitting. The edge labels represent the thresholds used for splitting. Each gray box is a leaf node, where the first label is the ID of the node and the second is the average gene expression for the submatrix corresponding to the node. Note that the two subtrees have been collapsed to save space (shown as N373 and N785). Each path from the root node to a leaf node forms a rule, which represents a biological hypothesis about the logic relationships among the expression levels of target genes, presence of binding motifs and the expression levels of putative regulators. For example, node N1267 shows that the expressions of genes with both mRRPE and PAC motifs are highly repressed if the expressions of Tpk1 and Kin82 are both induced.

In order to better interpret and understand these rules, we calculated the enrichment of GO functional categories for the genes within each leaf node. We also grouped the 173 experimental conditions into 19 categories and identified the significantly enriched categories for each leaf node. Together, the result describes the functional roles of a set of genes, their regulators and motifs, and the conditions under which they are activated or deactivated. The complete result of the analysis can be viewed interactively on the Supplementary website by clicking on the tree nodes. Overall, among the 50 leaf nodes, 45 have enriched conditions and 42 have enriched GO categories with corrected *p*-values < 0.05 (see Section 2.4).

As shown in Figure 3, the matrix is partitioned into five horizontal blocks by row attributes. The first block, corresponding to leaf nodes from N1084 to N1267, contains 245 genes whose promoters all have mRRPE motifs. It is known that mRRPE is important in regulating rRNA transcription and processing, a process that is repressed under many stress conditions (Pilpel *et al*., 2001; Gasch *et al*., 2000). Indeed, a GO analysis showed that 74 of these genes participate in rRNA processing ($p < 8e{-}65$). Furthermore, 94 genes having both mRRPE and PAC motifs (nodes N1252 and N1267) are more drastically repressed than the genes having mRRPE alone, which agrees with the fact that the two motifs work cooperatively (Sudarsanam *et al*., 2002). Comparing the GO annotation of the genes having both motifs and the genes having mRRPE only, the former are much more enriched in nucleolus (63/94 versus 32/151, $p < 8e{-}13$), and ribosome biogenesis and assembly (60/94 versus 36/151, $p < 5e{-}10$). On the other hand, the genes having only mRRPE motifs are more enriched in cytoplasm (95/151 versus 22/94, $p < 1e{-}9$), and protein biosynthesis (38/151 versus 6/94, $p < 9e{-}5$). This suggests that the genes regulated by both mRRPE and PAC have regulatory roles in ribosome biosynthesis, while the genes regulated by mRRPE alone are involved in protein synthesis. Interestingly, the genes having two or more copies of *tttctt* are downregulated (N1219), while the genes having two or more copies of *ttttct* are upregulated (N1214). Unlike other leaves in this block, N1219 is enriched in polysaccharide metabolism/glycan metabolism ($p < 0.003$). This suggests that, although *tttctt* and *ttttct* are similar to each other, they may be binding motifs of different TFs (Cliften *et al*., 2003).

The other four blocks also provide some biological insights. The second block contains 105 genes regulated by Fhl1 (leaf nodes from N1026 to N1064). Among them, 97 are structural constituents of ribosome ($p < 3e{-}144$). This is consistent with the recent results that Fhl1 (together with Ifh1) plays a central role in ribosome protein gene regulation (Wade *et al*., 2004). The third block contains leaf nodes from N579 to N1015. The main binding motif in this block, *agggg*, is the consensus sequence of the stress response element (STRE), which induces a large number of stress-responsive genes (Gasch *et al*., 2000). The most enriched GO category is the generation of precursor metabolites and energy (60/401, $p < 3e{-}26$). The genes having three or more copies of STRE motifs (node N986) show higher inductivity than those having less

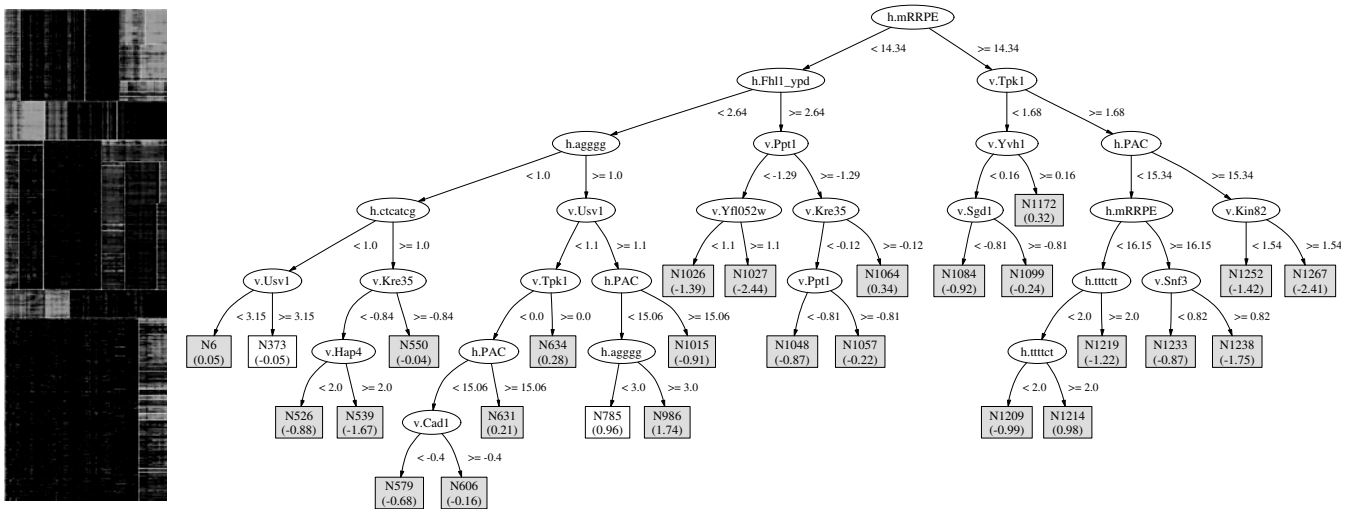


**Fig. 3.** Model built by BDTree for the yeast stress-responsive genes and the partition of the expression matrix. Node labels prefixed by 'h.' ('v.') are row (column) attributes. Row attributes whose names start with a uppercase letter followed by lowercase letters are from binding data. Row attributes with all lowercase letter names are *k*-mers from wordspy. The remaining row attributes are from the motif set of Pilpel *et al*. The bottom-left and upper-right submatrices correspond to leaf nodes N6 and 1267, respectively. This figure can be viewed with a higher resolution on the Supplementary website.

copies and are enriched in energy reserve metabolism ($p < 1e{-}9$), or specifically, trehalose metabolism, which is an important determinant of stress resistance in yeast (Winderickx *et al*., 1996). This suggests that the TF bound to STRE may have a preference for repetitive motifs. The fourth block (N526, N539 and N550) has 76 genes that are regulated by ctcactg, which is the consensus sequence of PAC. Similar to the genes in N1267, this set of genes is also enriched in ribosome biogenesis and assembly ($p < 1e{-}10$), although the degree of enrichment is lower than in N1267 (20/76 versus 60/94, $p < 1e{-}6$). Lastly, the fifth block is enriched with genes for nitrogen compound metabolism ($p < 7e{-}21$). Note that to the right of this block, the genes are separated into many small subsets, each of which is regulated by a different motif. Our GO analysis revealed that each subset of genes is responsible for some different functions. For example, there are nodes enriched with asparagine catabolism (N392 on the Supplementary website, $p < 3e{-}8$), aldehyde metabolism (N375, $p < 4e{-}5$), methionine metabolism (N442, $p < 2e{-}5$) and cell wall (N521, $p < 6e{-}7$).

On the other hand, although we found that most regulators selected by the tree have been reported as important in regulating stress-responsive genes (e.g. Usv1, Ppt1, Tpk1, Kin82), the exact biological roles of putative regulators are hard to determine. One reason is that TFs are often post-transcriptional modified or translocated before it can bind to a promoter. Therefore, the mRNA levels of a TF may not indicate its activity. In general, since we have included signaling molecules as candidate regulators, our method may select a kinase that activates a TF instead of the TF itself. Nevertheless, we found literature support for many of the rules. For example, the tree shows that the expression of Tpk1, a subunit of cAMP-dependent protein kinase (PKA), is negatively associated with the expression of mRRPE targets. It has been reported that cAMP signaling pathway regulates the downregulation of ribosome biogenesis (Schawalder *et al*., 2004). Furthermore, it is known that the RAS/cAMP pathway negatively regulates cellular physiology characteristic of stationary phase (Schawalder *et al*., 2004). This agrees with the results that node N1267 is enriched with GO annotation ribosome biogenesis and assembly, and the most significant conditions are 'stationary phase'. Another computational study by Segal *et al*. (2003) suggested a role for Tpk1 and Sgd1 in the regulatory program for rRNA processing and ribosome biogenesis, which is similar to our results. Msn2, the TF that binds to STRE, needs to be translocated from the cytoplasm to nucleus under stress conditions, and it has been reported that PKA is involved in the translocation (Jacquet *et al*., 2003), which is consistent with our analysis (node N634). Our results suggest that Usv1, which has been identified as a top regulator for many stress responses (Segal *et al*., 2003), may play a role in this process as well.

We also calculated the importance measure of each attribute using surrogate splits (see Section 2.4). Table 2 shows the top 20 row attributes and top 20 column attributes respectively. The complete list is available on the Supplementary website.

Among the top 20 row attributes, PAC (as well as *gctcatc* and *ctcatcg*), mRRPE and Fhl1 are known to be related to stress response as discussed above. Motifs *agggg*, *ccctt*, *ggggc* and *aggggc* are variants of the extended STRE motif (*a*/*caggggc*/*ggg*) or its reverse complement (Harbison *et al*., 2004). Motifs *tccct* and *tccctt* are the binding sites of Gis1, a transcriptional factor involved in the expression of genes during nutrient limitation (Oshiro *et al*., 2003). Rap1 and Snf1 are known to control the expression of ribosomal protein genes during various stress responses (Gasch *et al*., 2000; Dwight *et al*., 2004). Both Gat3 and Yap5 have functions in stress responses and co-bind with Msn4 (Banerjee and Zhang, 2003). The binding data of Fhl1 and Rap1 measured under different conditions are all top-ranked row-attributes, which means that their binding is probably condition invariant. On the other hand, STRE-like motifs rather than Msn2 are among the top row attributes, which suggests that the binding specificity of Msn2 varies significantly under different conditions.

**Table 2.** Top row and column attributes identified by BDTree

| | |
|---|---|
| Top row attributes | PAC, mRRPE, Fhl1_rapa, Fhl1_ypd, agggg, Fhl1_sm, Fhl1_h2o2hi, ccctt, Rap1_sm, gctcatc, Rap1_ypd, Gat3_ypd, Rap1, ggggc, Sfp1_sm, ctcatcg, aggggc, tccct, Yap5_ypd, tccctt |
| Top column attributes | Usv1, Tpk1, Xbp1, Kns1, Sip2, Kin82, Yjl103c, Mtl1, Ppz2, Yak1, Gis1, Pde1, Rim11, Gpa2, Tpk2, Tos8, Nrg1, Gat2 |

Row attributes starting with uppercase letter followed by lowercase letters are from binding data of Harbison *et al*. Row attributes with all uppercase letters are from the motif set of Pilpel *et al*. rapa, nutrient deprived; sm, amino acid starvation; ypd, normal growth condition.

Many of the top 20 column attributes, such as Usv1, Tpk1, Xbp1, Gis1, Kin82, Gac1, Rim11, Gpa2, Yjl103c and Tos8, have evidence to be involved in regulating various stress responses in SGD database (Dwight *et al*., 2004) or other computational analyses (Middendorf *et al*., 2004; Segal *et al*., 2003). Interestingly, we find that only a few of the identified top regulators are TFs (XBP1, GIS1, TOS8, NRG1, GAT2), while the majority are protein kinases or hydrolases, which suggests that post-transcriptional regulation plays an important role in stress responses (Grigull *et al*., 2004).

We also learned a model of the yeast cell-cycle data. The complete tree and detailed analysis is on the Supplementary website. Our method identified almost all known TFs regulating the yeast cell-cycle genes and their binding motifs (see Supplementary website).

## 4 DISCUSSION

In this research, we have developed a novel method, the bi-dimensional regression tree (BDTree), for modeling transcriptional regulation from large-scale gene expression data. BDTree is a significant extension of previous works. First, the tree-based approach does not assume linear additivity of regulatory elements or any distribution of the underlying dataset. Second, by considering gene expression under multiple conditions simultaneously, the method can tolerate more noises than using individual arrays. More importantly, by taking into account both the expression of putative regulators and the occurrence of putative binding motifs, BDTree is able to identify condition-specific regulatory elements and regulators for each gene. We have successfully applied BDTree to the yeast cell-cycle and stress response data, and identified many biologically significant binding motifs and regulators.

Two existing methods are similar to our approach in that they also attempt to model the large-scale gene expression data under a large number of conditions. The module networks approach (Segal *et al*., 2003) clusters genes according to their expression patterns and builds a regression tree for each cluster. However, in their method, binding motifs are not considered when clustering genes. As a result, genes having similar expression patterns are assigned the same set of regulators, regardless of the difference of their promoters. Furthermore, the clustering of genes in their method is based on the expression levels across all conditions. Therefore, their method is unlikely to identify condition-specific regulation.

The Geneclass method (Middendorf *et al*., 2004) is the most similar to ours. Indeed, Geneclass and BDTree have the same schematic representation (Fig. 1 box E), i.e. modeling gene expression levels from putative binding motifs and TF expression levels. In addition, both methods build tree-based models (decision trees in Geneclass and regression trees in BDTree). Despite these similarities, the underlying modeling rationales are very different. BDTree is a novel extension to a multivariate regression tree approach, while Geneclass transforms the modeling problem into a traditional decision tree learning problem. This difference leads to several significant consequences.

First, in Geneclass, gene expression levels have to be discretized into three categories: up, down and intermediate, but only the up and down categories are used for training. These choices are arbitrary and may cause a significant amount of information to be lost. BDTree, on the other hand, accepts real-valued data and uses all data points. Second, because Geneclass treats expression levels as univariate variables, the differences between genes and conditions are disregarded. As a result, Geneclass attempts to find submatrices that have constant expression levels along both dimensions, which may not be biologically meaningful. In contrast, BDTree optimizes the homogeneity on one dimension of the expression matrix in each split, while allowing heterogeneity on the other dimension. Third, Geneclass forces each split to couple a row attribute and a column attribute. Therefore, a total of $mn$ attribute pairs need to be considered for each split, where $m$ and $n$ are the numbers of row and column attributes, respectively. BDTree only needs to consider $m+n$ row and column attributes individually for each split and is thus more scalable. By coupling row and column attributes, Geneclass may have the advantage of directly suggesting associations between regulators and binding motifs. However, it is not always advisable to relate regulators to binding motifs. For example, the binding motifs of a regulator or a regulator itself may not be present in the list of candidates. It is also possible that the regulators are post-transcriptionally regulated; therefore its expression levels do not correlate with the expression levels of its targets. BDTree is more flexible since it does not force an explicit pairing of regulators and binding motifs.

Our method is general and can be turned into several previous methods easily, taking each of them as a special case. For example, when the minimum number of column instances is set to a sufficiently large number, BDTree is equivalent to that in Phuong *et al*. (2004). When vertical splits are restricted to occur only after horizontal splits are completed, BDTree performs similarly as in the method of Segal *et al*. (2003). BDTree can also be applied to domains other than computational biology, such as clinical studies. For example, the multivariate responses may be a time-series observation of drug efficacies on patients, for which our method can be used to identify the time-dependent impact of different factors.

There are several directions that this method may be extended. One problem with the current implementation of BDTree is that the splitting of genes or conditions is strictly based on attribute values. Such a hard split may be undesirable considering that there are inevitable noises in attribute values: motif representations may be inaccurate, expression levels of regulators are unreliable and

normalization may introduce additional noises. Furthermore, the regression tree learning algorithm is essentially greedy, never re-considering a split that has been made. To circumvent these problems, some fuzzy rules may be applied to find soft splitting. Look-ahead strategies may be used to find globally better top-level splits. Furthermore, some iterative strategies may be adopted to improve motif representations. Another problem with our approach is that it can only model simultaneous changes between regulators and targets, i.e. the expression levels of regulators have to be correlated or anti-correlated with those of the target genes. To allow shifted or reversely shifted correlations to be identified, the method may include the expression levels of regulators at previous time points as additional column attributes.

## ACKNOWLEDGEMENTS

This research was supported in part by NSF grants IIS-0196057 and EIA-0113618 under the ITR program, and a grant from Monsanto Corporation. The authors thank the anonymous reviewers for very helpful comments on an early version of this paper.

*Conflict of Interest:* none declared.

## REFERENCES

Altman,D. (1991) *Practical Statistics for Medical Research*. Chapman & Hall/CRC, New York.

Banerjee,N. and Zhang,M. (2003) Identifying cooperativity among transcription factors controlling the cell cycle in yeast. *Nucleic Acids Res.*, **31**, 7024–7031.

Beer,M. and Tavazoie,S. (2004) Predicting gene expression from sequence. *Cell*, **117**, 185–198.

Breiman,L., Friedman,J. and Stone,R.O.C. (1984) *Classification and Regression Trees*. Wadsworth Int. Group, Belmont, CA.

Bussemaker,H. *et al.* (2001) Regulatory element detection using correlation with expression. *Nat. Genet.*, **27**, 167–171.

Cheng,Y. and Church,G. (2000) Biclustering of expression data. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, **8**, 93–103.

Clifften,P. *et al.* (2003) Finding functional features in *Saccharomyces* genomes by phylogenetic footprinting. *Science*, **301**, 71–76.

Conlon,E. *et al.* (2003) Integrating regulatory motif discovery and genome-wide expression analysis. *Proc. Natl Acad. Sci. USA*, **100**, 3339–3344.

Dwight,S. *et al.* (2004) *Saccharomyces* genome database: underlying principles and organisation. *Brief Bioinform.*, **5**, 9–22.

Gasch,A. *et al.* (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, **11**, 4241–4257.

Grigull,J. *et al.* (2004) Genome-wide analysis of mRNA stability using transcription inhibitors and microarrays reveals posttranscriptional control of ribosome biogenesis factors. *Mol. Cell. Biol.*, **24**, 5534–5547.

Harbison,C. *et al.* (2004) Transcriptional regulatory code of a eukaryotic genome. *Nature*, **431**, 99–104.

Harris,M. *et al.* (2004) The gene ontology (GO) database and informatics resource. *Nucleic Acids Res.*, **32** (Database issue), D258–D261.

Hu,Y. *et al.* (2000) Combinatorial motif analysis and hypothesis generation on a genomic scale. *Bioinformatics*, **16**, 222–232.

Jacquet,M. *et al.* (2003) Oscillatory nucleocytoplasmic shuttling of the general stress response transcriptional activators Msn2 and Msn4 in *Saccharomyces cerevisiae*. *J. Cell Biol.*, **161**, 497–505.

Keles,S. *et al.* (2002) Identification of regulatory elements using a feature selection method. *Bioinformatics*, **18,** 1167–1175.

Middendorf,M. *et al.* (2004) Predicting genetic regulatory response using classification. *Bioinformatics*, **20** (Suppl. 1), I232–I240.

Oshiro,J. *et al.* (2003) Regulation of the yeast DPP1-encoded diacylglycerol pyrophosphate phosphatase by transcription factor Gis1p. *J. Biol. Chem.*, **278**, 31495–31503.

Phuong,T. *et al.* (2004) Regression trees for regulatory element identification. *Bioinformatics*, **20**, 750–757.

Pilpel,Y. *et al.* (2001) Identifying regulatory networks by combinatorial analysis of promoter elements. *Nat. Genet.*, **29**, 153–159.

Roth,F. *et al.* (1998) Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat. Biotechnol.*, **16**, 939–945.

Ruan,J. and Zhang,W. (2004) Discovering transcriptional regulatory rules from gene expression and TF-DNA binding data by decision tree learning. *Technical Report 43*, Department of Computer Science and Engineering, Washington University in St Louis.

Schawalder,S. *et al.* (2004) Growth-regulated recruitment of the essential yeast ribosomal protein gene activator Ifh1. *Nature*, **432**, 1058–1061.

Segal,E. *et al.* (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet.*, **34**, 166–176.

Segal,M. (1992) Tree-structured methods for longitudinal data. *J. Am. Stat. Assoc.*, **87**, 407–418.

Soinov,L. *et al.* (2003) Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biol.*, **4**, R6.

Spellman,P. *et al.* (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Sudarsanam,P. *et al.* (2002) Genome-wide co-occurrence of promoter elements reveals a *cis*-regulatory cassette of rRNA transcription motifs in *Saccharomyces cerevisiae*. *Genome Res.*, **12** (11), 1723–1731.

van Helden,J. (2003) Regulatory sequence analysis tools. *Nucleic Acids Res.*, **31**, 3593–3596.

Wade,J. *et al.* (2004) The transcription factor Ifh1 is a key regulator of yeast ribosomal protein genes. *Nature*, **432**, 1054–1058.

Wang,G. *et al.* (2005) WordSpy: identifying transcription factor binding motifs by building a dictionary and learning a grammar. *Nucleic Acids Res.*, **33** (Web Server issue), W412–W416.

Winderickx,J. *et al.* (1996) Regulation of genes encoding subunits of the trehalose synthase complex in *Saccharomyces cerevisiae*: novel variations of STRE-mediated transcription control? *Mol. Gen. Genet.*, **252**, 470–482.