Probabilistic Discovery of Overlapping Cellular Processes and Their Regulation

Alexis Battle
ajbattle@stanfordalumni.org

Eran Segal
erans@cs.stanford.edu

Daphne Koller
koller@cs.stanford.edu

Computer Science Department
Stanford University
Stanford, CA 94305-9010

ABSTRACT

Many of the functions carried out by a living cell are regulated at the transcriptional level, to ensure that genes are expressed when they are needed. Thus, to understand biological processes, it is thus necessary to understand the cell's transcriptional network. In this paper, we propose a novel probabilistic model of gene regulation for the task of identifying overlapping biological processes and the regulatory mechanism controlling their activation. A key feature of our approach is that we allow genes to participate in multiple processes, thus providing a more biologically plausible model for the process of gene regulation. We present an algorithm to learn this model automatically from data, using only genome-wide measurements of gene expression as input. We compare our results to those obtained by other approaches, and show significant benefits can be gained by modeling both the organization of genes into overlapping cellular processes and the regulatory programs of these processes. Moreover, our method successfully grouped genes known to function together, recovered many regulatory relationships that are known in the literature, and suggested novel hypotheses regarding the regulatory role of previously uncharacterized proteins.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Biology and genetics

General Terms

Experimentation, Algorithms

Keywords

Cellular Processes, Gene Regulation, Probabilistic Relational Models

RECOMB'04, March 27–31, 2004, San Diego, California, USA.

1. Introduction

A living cell is a complex system that needs to perform various functions and adapt to changing environments. Essential to the cell's ability to respond to different circumstances is its organization into a set of *processes*, whose activity varies according to circumstance. The activation of these processes is often controlled by a variety of regulatory signals, which are themselves triggered by various aspects of the current state. Genome-wide measurements of mRNA expression level across multiple experimental conditions provide us with a global picture of the cell's activities, and open the way to a high-level understanding of its behavior.

The most common approach for analyzing gene expression is to *cluster* genes whose expression profile is similar over a range of experimental conditions [7, 5]. As the clusters correspond to sets of genes that respond similarly in different circumstances, their member genes are likely to share a common function. However, these approaches group genes into mutually exclusive clusters, whereas the real biological system is much more intricate, with many genes playing multiple roles in different circumstances. To address this, other approaches [29, 31, 15, 19, 1, 26] have been proposed for discovering *processes*, or overlapping groups of genes, thereby providing a more realistic model for cellular activity.

However, finding the genes that participate in each process provides only a partial view of the biological system. In many cases, we are also interested in discovering the regulatory mechanism that controls the activation of each process and its response to different circumstances. This type of analysis is typically done as a post-processing step, by searching, for example, for common *cis*-regulatory motifs in the promoter region of the genes in the process [32], or (more recently) for correlation with protein-DNA binding data [20]. However, such approaches are limited both by the amount of noise in these data, and, more importantly, by the fact that many regulatory relationships are *context-specific*, occurring only under certain conditions, a phenomenon which does not manifest in these data.

Recent work [24, 28] shows that regulatory relationships can be learned directly from gene expression data, in a way that accounts for the context-specificity of regulatory events. Moreover, as shown by Segal *et al.*[28, 27], by searching for sets of co-regulated genes and their actual regulatory program simultaneously, a much more accurate organization of genes into processes can be obtained. However, while their approach models aspects relating to regulatory mechanisms, they partition the genes into mutually exclusive processes, and thus their models are limited in the comprehensiveness of the regulatory picture they provide.

In this paper, we propose a novel model of gene regulation for discovering overlapping cellular processes and their regulatory programs — the Coregulated Overlapping Processes model, or *COPR* model (pronounced "copper"). Our approach builds on the ideas in our earlier work [26] for modeling overlapping processes, and extends the ideas of Segal *et al.* [28] for modeling regulatory programs, resulting in a unified probabilistic framework of gene regulation. The COPR model makes explicit the notion of a biological *process*, and each gene can then be a member of one or more of these processes.

Following our earlier work, we assume that a process is active to different extents in different conditions. Specifically, we assume that each process p has some activity level $a.C_p$ in each given array a. The expression level of a gene in an array a is therefore a sum of the activities of the processes with which it is associated.

For modeling regulatory relationships, we make the simplifying assumption that regulation is done at the level of processes, rather than individual genes. Thus, we assume that each process is associated with some (unknown) regulatory program, which determines its activity level $a.C_p$ in each array a. Thus, the activity of p is different in different arrays, but it is governed by the same set of rules. The process' activity level is assumed to be a function of its regulators. We use a regulatory program that captures two of the most important forms of regulation: both the combinatorial (context-specific) nature of some types of gene regulation, and the dependence of the activity level of a process on the actual abundance of the regulatory protein (as indicated by its mRNA level) in the cell.

We present an efficient algorithm for learning the COPR model automatically from data, using only the raw gene expression measurements as input. In our learning task, we must associate genes with processes, as well as learn the regulatory program of each process — both the set of regulators for each process and the program itself. This learning problem is quite challenging, as both the (discrete) assignment of genes to processes and the (continuous) activity levels of processes in arrays are all unobserved.

We test the COPR model on two expression data sets, including a large compendium of 394 yeast microarrays compiled from four different studies. We evaluate its performance relative to a variety of existing biological data sources, including known gene annotations [2], presence of motifs in the genes' promoter regions [32], and correlation with protein-DNA binding data [20]. Relative to all of these sources, we show that the COPR approach discovers groups of genes that correspond to biologically coherent processes, significantly outperforming both our earlier process model [26] and the module network framework of Segal *et al.* [28]. We also evaluate the predicted activity levels for our learned processes, and show that they are very consistent with the processes' role in the cell. Overall, our results support our basic assumptions, that overlapping processes provide a better model of cellular activity than disjoint gene clusters, and that by searching simultaneously for co-regulated genes and for their regulatory programs, we converge to models that are biologically more plausible.

2. Related Work

As described above, the COPR model addresses two main goals: placing genes into functional groups, and discovering regulatory mechanisms for those gene groups. The first goal, grouping genes into meaningful groups, is shared by approaches such as clustering (e.g., [7]). Standard clustering partitions genes into mutually exclusive groups, while COPR places genes in multiple processes simultaneously.

Another approach to relaxing the assumption of mutually exclusive clusters is "soft clustering", which assigns probabilities that each member belongs to each one group. This type of model is more reflective of uncertainty about the cluster assignment of a gene, rather than its assignment to multiple clusters simultaneously. Other approaches, such as biclustering [5] or context-specific clustering [29] allow genes to participate in different groups in different contexts, but in any given array, a gene still belongs to exactly one group. In other words, each individual expression measurement is explained by a gene's membership in precisely one cluster. In the COPR model, a gene can belong to multiple processes simultaneously, and a gene's expression value in any condition is a sum of the activity of all of the processes in which the gene participates.

Thus, the COPR model decomposes the expression matrix as a sum of process activity matrices. This characteristic makes COPR more similar to singular value decomposition (SVD) [1] or the Plaid model of Lazzeroni and Owen [19]. These approaches also decompose the expression matrix as a sum of activities of (overlapping) *components* or *layers*, which are roughly analogous to our processes. However, our definition of processes differs significantly, in that we incorporate an explicit bias against having genes that participate in a large number of processes. Thus, in our learned COPR models, each gene tends to belong to at most two or three processes. By contrast, SVD or Plaid often produce models where genes belong to a very large number of processes. The sparser models are more biologically plausible, and contain processes that are more biologically coherent [26].

Finally, and most notably, none of these approaches include a model of gene regulation. Including gene regulation in the COPR model both allows us to discover regulatory relationships, and to use these discovered relationships to actually improve the grouping of genes. The COPR model presented in this paper is based on the regulatory framework proposed in the module network framework [28], but extends it in two significant ways. First, it allows the use of overlapping processes, whereas the module network framework assumes disjoint clusters. Second, it extends the notion of a regulatory program to allow both combinatorial and linear dependence on the abundance of regulators.

3. Probabilistic Model

The COPR model is based on the framework of Bayesian networks [23], and on the language of *probabilistic relational models* (PRMs) [16, 10, 29], which represents the domain in terms of the different biological entities that interact in it: genes, arrays, expression measurements, regulators, and biological processes. Each object is also associated with a set of attributes that are relevant to its interactions with the other entities.

Gene Expression Model. The first component of the COPR model represents the gene expression and its decomposition into the activity level of processes. This component of COPR includes a set $\mathbf{G}$ of n gene objects, $\mathbf{G} = \{g_1, \ldots, g_n\}$, a set $\mathbf{A}$ of k array objects, $\mathbf{A} = \{a_1, \ldots, a_k\}$, and a set $\mathbf{E}$ of expression objects $\mathbf{E} = \{e_{1,1}, \ldots, e_{n,k}\}$, one for each gene in each array. Each expression object e is associated with a gene object $e.\mathit{Gene} = g$, an array object $e.\mathit{Array} = a$, and a real-valued attribute $e.\mathit{Level}$ denoting the mRNA expression level of $e.\mathit{Gene} = g$ in $e.\mathit{Array} = a$.

As we discussed, a key property of the our approach is that it allows genes to participate in multiple processes. To this end, COPR follows the approach of Segal, Battle, and Koller [26] (SBK hereafter), and explicitly includes a set of j *processes*; each gene object g is associated with a set of binary *process membership* attributes $g.M_1, \ldots, g.M_j$, where $g.M_p$ denotes whether g is a member of process p. As processes may be active to various degrees in different arrays, we also define a set of continuous *activity level attributes* $a.C_1, \ldots, a.C_j$ for each array a, where $a.C_p$ corresponds to the degree to which process p is active in array a. The expression level of gene g in array a is assumed to be a sum of g's expression levels in each of the processes in which it participates, where g's expression level in process p is the activity of the process $a.C_p$. More precisely, we assume that the expression of gene g in array a is normally distributed with a mean that is equal to the sum, over processes p in which which g participates, of the activity level of p:

$$P(e.\mathit{Level} \mid g.\mathbf{M}, a.\mathbf{C}) = \mathcal{N}(\sum_{p=1}^{j} g.M_p \cdot a.C_p; \sigma_a^2). \quad (1)$$

where σ_a^2 is the variance associated with array a.

Regulatory Model. The second component of the COPR model is the regulatory model. We assume that genes in the same process are co-regulated, and therefore share the same regulatory mechanism. Thus, we define a *regulation program* for each process, which is then shared across all genes assigned to that process. To model regulation programs, we extend the framework of Segal *et al.* [28], where each regulation program is defined in terms of some set of regulators, and describes the expression of genes in the process as a function of the mRNA expression level of these regulators. We therefore assume that the model contains a set of t *candidate regulators*, where for each regulator r and each array a we have a continuous attribute $a.R_r$ that denotes the mRNA expression level of regulator R_r in array a. We define $a.\mathbf{R} = a.R_1, \ldots, a.R_t$.

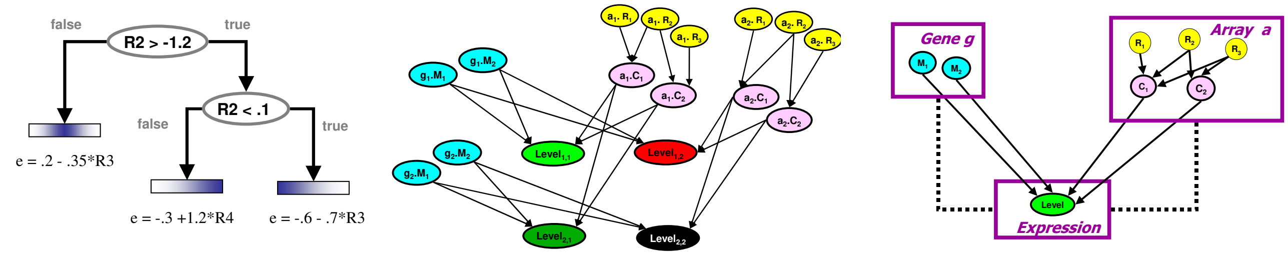
We model the activity level of process p in the different arrays as a stochastic function of the expression level of some small set of process regulators. The regulators are specified separately for each process, allowing us to represent regulation programs that are specific to different processes. Thus, for each process p, the COPR model specifies a set of d parent regulator attributes $\mathcal{R}_p = \{R_{p,1}, \ldots, R_{p,d}\}$, and the activity level of process p in array a, $a.C_p$, is then described as a stochastic function of $a.R_{p,1}, \ldots, a.R_{p,d}$. In the COPR framework, this stochastic function takes the form of a *regression tree* [4]. A regression tree allows us to model two types of dependence on regulators: linear dependencies, which are very common, and combinatorial regulation, which is crucial for any realistic model of gene regulation in higher eukaryotes.

As illustrated in Fig. 1(a), a regression tree $\mathcal{S}_p$ for a process p with parents $\mathcal{R}_p$, is a rooted binary tree with interior tree nodes and leaf tree nodes. Each interior tree node is labeled with a *test* over one of the parents, such as $a.R_{p,1} > .7$. Each leaf node specifies a distribution over $a.C_p$. For a particular array a in which we observe the expression levels of the regulators, we define $P(a.C_p \mid a.R_{p,1}, \ldots, a.R_{p,d})$ by traversing the tree from the root, and taking the appropriate branch at each interior node: the right branch if the values $a.R_{p,1}, \ldots, a.R_{p,d}$ pass the test at that node, and the left branch otherwise. We continue this traversal until we reach a leaf node ℓ. The distribution $P(a.C_p \mid a.R_{p,1}, \ldots, a.R_{p,d})$ is then specified by the distribution associated with ℓ. The leaf distribution at each leaf ℓ in is a *linear Gaussian* — a Gaussian whose mean is a linear function of a subset of *linear parents* $\mathcal{R}_\ell = \{R_{\ell,1}, \ldots, R_{\ell,f_\ell}\}$ of the parent regulators $\mathcal{R}_p$. Thus, if array a reaches leaf ℓ in the tree, it will have a Gaussian distribution $\mathcal{N}(\mu_a; \sigma_\ell^2)$, where $\mu_a = b_{\ell,0} + \sum_{i=1}^{f_\ell} b_{\ell,i} \cdot a.R_{\ell,i}$, $b_{\ell,i}$ is the weight of regulator i in the leaf distribution of leaf ℓ, and σ_ℓ^2 is the variance associated with the leaf. Thus, we represent the activity levels both by combinatorial interactions of regulators, and by a linear function of regulator expression levels. We note that this regulation model is an extension of the model of Segal *et al.*[28], which incorporated only combinatorial regulation, and did not allow linear dependence on the actual expression level of the regulators.

Model Summary. Our overall COPR PRM model puts together these two components. The probability of the expression levels $\mathbf{E}.\mathit{Level}$ given the gene process membership variables $\mathbf{G}.\mathbf{M}$ and the process activity level $\mathbf{A}.\mathbf{C}$ is as described in our gene expression model. The probability of $\mathbf{A}.\mathbf{C}$ given the regulators is as described in the regulatory model. To complete the description of COPR, we define the gene membership attributes, $g.\mathbf{M}$, to have no parents; we simply associate with each process p a prior distribution over the membership of genes in this process $P(g.M_p) = q_p$. The regulator expression levels $a.R_1, \ldots, a.R_t$ also have no parents. We assign a Gaussian distribution $\mathcal{N}(\mu_r; \sigma_r^2)$, noting that, as these regulator attributes are always observed, this distribution will not play a role when we learn the model from data.

A simple example of COPR PRM for the case of two processes and three regulators is shown Fig. 1(a). For any set of genes and arrays, the model defines a probability distribution over the gene process memberships, the array activity levels, and the expression levels of genes in the arrays. This probability distribution is defined as a Bayesian network, whose structure and parameterization is determined by the model. An example of the instantiation of the COPR model, for the case of two genes, two arrays, two processes, and three regulators, is shown in Fig. 1(b). The joint distribution defined by our instantiation can be written in terms of the objects, their attributes and the distributions specified above:

$$\begin{aligned} &P(\mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathbf{A}.\mathbf{R}, \mathbf{E}.\mathit{Level}) = \\ &\quad \prod_{i=1}^{j} \prod_{g \in \mathbf{G}} P(g.M_i) \\ &\quad \cdot \prod_{a \in \mathbf{A}} P(a.C_i \mid a.R_{i_1}, \ldots, a.R_{i_d}) P(a.R_{i_1}, \ldots, a.R_{i_d}) \\ &\quad \cdot \prod_{e \in \mathbf{E}} P(e.\mathit{Level} \mid e.\mathit{Gene}.\mathbf{M}, e.\mathit{Array}.\mathbf{C}). \quad (2) \end{aligned}$$



(a) (b) (c)

Figure 1: (a) A sample regression tree, predicting a different distribution over the activity level of a process, depending on regulator levels. (b) COPR PRM with two processes and three regulators. (c) A simple instantiation of the COPR PRM in (b), for two genes and two experiments.

A COPR model makes explicit the basic biological entities in the domain, and the relationships between them. Thus, we can easily read off biological conclusions from the model: The variables $g.\mathbf{M}$ for each gene g specify the gene's membership in different cellular processes. The regulators of each process p are specified by the sets $\mathcal{R}_p$, and the regression tree for process p tells us exactly how each regulator affects the activity level of the genes participating in process p.

4. Learning

Learning a COPR model from microarray data alone is a complex task. Microarray data provides the values for each *e.Level* variable, and the values for each regulator expression variable $a.R_{p,i}$. All of the assignments of genes to processes are hidden, as are all activity levels. In addition, we do not know which regulators $\mathcal{R}_p$ control each process p, or by what program the regulators control their target processes. The parameters q_p, specifying the probability of a gene belonging to process p, must also be learned. Thus, from expression data alone, the goal is to group genes into processes, estimate process activity levels for each experiment, and to learn the regulatory control programs governing the activity of each process.

Overall, this problem is one of learning a probabilistic model — structure as well as parameters — from partially observed data. We address this problem using a hard-assignment variant of the *structural EM (SEM)* algorithm of Friedman [9]. Like the EM algorithm of Dempster, Laird, and Rubin [6], SEM iterates over two steps: In the *E-step*, it finds a "completion" of the values to the hidden variables given a current model; in the *M-step* it re-estimates the model given our current "completion" of the data. In SEM, the model structure as well as the parameters are both learned in the M-step.

We now describe each of the main steps of our algorithm, and then describe how we put them together in a single learning algorithm.

M-step. In the M-step, we are given a complete assignment to the hidden variables, and our task is to learn a good model, structure as well as parameters. The structure $\mathcal{S}$ specifies: the structure $\mathcal{S}_p$ of the regression tree for each process p, and the set of linear parents $\mathcal{R}_\ell$ at each leaf ℓ in the tree $\mathcal{S}_p$. The parameters specify the linear coefficients and the variances in the regression tree, the variances σ_a^2 of the expression measurements for array a, and the probability q_p of gene membership to process p.

To select $\mathcal{S}$, we use a Bayesian model selection approach, which explicitly accounts for our uncertainty about the parameters using probability distributions over the values of $\boldsymbol{\theta}$. We then search for a model that has high posterior probability given the expression data $\mathcal{D}$, integrating over all possible values of $\boldsymbol{\theta}$: $P(\mathcal{S} \mid \mathcal{D}) \propto P(\mathcal{S})P(\mathcal{D} \mid \mathcal{S}) = P(\mathcal{S})\int P(\mathcal{D} \mid \mathcal{S}, \boldsymbol{\theta}_\mathcal{S})d\boldsymbol{\theta}_\mathcal{S}$. We choose to use a structure prior $P(\mathcal{S})$ that penalizes the number of regulators in each regulatory program. Specifically, our prior shrinks exponentially with the number of distinct regulators. Our Bayesian score is simply the logarithm of this posterior probability.

We search for a set of regression trees $\mathcal{S}_p$ that maximize this posterior probability, given a hard assignment to the hidden variables. Given a fixed assignment to $\mathbf{A}.\mathbf{C}$, the optimization task for each process p is independent, and depends only on the activity levels of p in each array, $Array.C_p$. Thus, our task for process p reduces to one of finding the regression tree that optimizes the Bayesian score, given the (completed) data $\mathbf{A}.C_p$.

To find a high scoring regression tree $\mathcal{S}_p$, we perform a greedy search over possible structures. We begin with a single root node in the tree, which would predict the activity level of process p in all conditions. Then, we consider every possible single split that could be added to the tree. A split is a test of the form $a.R_i > z$ for any regulator expression $a.R_i$ and any real value z. Adding this split to the tree would allow the model to specify two distributions, one for activity levels in experiments where $a.R_i > z$, and one for experiments where $a.R_i \leq z$. Note that, while the addition of a split can never lower a likelihood score, it can hurt the Bayesian score. We calculate the change in score for each possible split, and if any improve the score, we use the best split to modify the tree. When no more splits can improve the score, the tree structure and split cutoff values are fixed.

To score each split, we first note that the Bayesian score of a regression tree can be decomposed as a sum over terms, each reflecting the score of a single leaf node in the tree. Thus, to calculate the improvement made by a split, we need only consider the contribution to the score of the affected leaves. Scoring a leaf, unfortunately, is not trivial.

Recall that each leaf ℓ defines a linear Gaussian distributions that depends on some set of linear parents $\mathcal{R}_\ell$. Given a fixed set of linear parents, we can compute the Bayesian score of the leaf by integrating over all possible values of the linear coefficients (with a Gaussian prior), as described by Geiger *et al.* [13]. We select the set of linear parents at each leaf using a greedy search, adding linear parents which improve the Bayesian score of the leaf. We note that this computation is performed every time a new leaf is considered as part of a split operation.

The result of this computation is a model structure, specifying the regression tree structure $\mathcal{S}_p$ for every process p. To perform the E-step effectively, however, we need a single model, with a single set of parameters. Given the structure, we estimate the parameters using a simple maximum likelihood computation: The probabilities q_p are estimated as the fraction of genes assigned to process p; the variances σ_a^2 associated with (1) and σ_ℓ^2 associated with the leaf ℓ are estimated using their empirical estimates; and the linear coefficients at each leaf are estimated using standard linear regression.

E-step. We now turn to the problem of finding the most likely joint assignment to $\mathbf{G}.\mathbf{M}$ and $\mathbf{A}.\mathbf{C}$. This task is a hard one: As can be seen in Fig. 1(c), these variables are all correlated via their joint influence on the expression levels, making this inference task intractable for large models. The problem is further complicated by the fact that it involves both discrete and continuous variables, a setting where even very simple network structures are intractable [21]. Thus, we decompose the E-step into two substeps: finding the most likely assignment of genes to processes given the process activity levels, and finding the most likely assignment of process activity levels given gene assignments.

The first of these two substeps — finding the assignment of genes to processes — can be formulated as follows: given an assignment to $\mathbf{A}.\mathbf{C}$ and a model $\mathcal{M}$, to optimize:

$$\begin{aligned}
&\mathrm{argmax}_{\mathbf{G}.\mathbf{M}} P(\mathbf{G}.\mathbf{M} \mid \mathcal{M}) P(\mathbf{A}.\mathbf{R} \mid \mathcal{M}) \\
&\qquad P(\mathbf{E}.\mathit{Level} \mid \mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathcal{M}) P(\mathbf{A}.\mathbf{C} \mid \mathbf{A}.\mathbf{R}, \mathcal{M}) = \\
&\ \mathrm{argmax}_{\mathbf{G}.\mathbf{M}} P(\mathbf{G}.\mathbf{M} \mid \mathcal{M}) \\
&\qquad P(\mathbf{E}.\mathit{Level} \mid \mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathcal{M}). \qquad (3)
\end{aligned}$$

As the activation levels are given, the regulatory model is irrelevant. Thus, this problem is precisely identical to the one addressed by SBK. As they show, this optimization problem can be performed independently for each gene. However, an exact calculation requires a number of operations which is exponential in the number of processes. SBK propose an approximate algorithm, which we adopt here. Rather than considering every possible assignment to $g.\mathbf{M}$, we select a smaller subset of processes to which the gene is allowed to belong. This subset is selected by defining a continuous relaxation of our original maximization problem, which can be solved using a bounded least squares algorithm [3]. The resulting continuous solution is used to select the subset of processes to consider for each gene. We experimentally selected a threshold, and if the continuous "membership" of a gene to a process is above that threshold, that process is included in the set of candidates for that gene. In practice, a large number of the continuous memberships were actually assigned to 0. Once each gene has been assigned a candidate set of processes, the remaining discrete optimization problem is solved exactly using exhaustive enumeration.

In the second part of the E-step, we compute the most likely assignment to the activity levels $\mathbf{A}.\mathbf{C}$ with the regulatory program and gene memberships $\mathbf{G}.\mathbf{M}$ fixed. Our goal here is to maximize:

$$\begin{aligned}
&\mathrm{argmax}_{\mathbf{A}.\mathbf{C}} P(\mathbf{G}.\mathbf{M} \mid \mathcal{M}) P(\mathbf{A}.\mathbf{R} \mid \mathcal{M}) \\
&\qquad P(\mathbf{E}.\mathit{Level} \mid \mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathcal{M}) P(\mathbf{A}.\mathbf{C} \mid \mathbf{A}.\mathbf{R}, \mathcal{M}) = \\
&\ \mathrm{argmax}_{\mathbf{A}.\mathbf{C}} P(\mathbf{A}.\mathbf{C} \mid \mathbf{A}.\mathbf{R}, \mathcal{M}) \\
&\qquad P(\mathbf{E}.\mathit{Level} \mid \mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathcal{M}). \qquad (4)
\end{aligned}$$

For any array a and each process p, the regulator variables are all observed, so that we know precisely which path is followed down the regression tree for p, and the values of the linear parents at the corresponding leaf. Thus, $\mathcal{M}$ and the values of the regulator variables define a Gaussian distribution $\mathcal{N}(\mu_{p,a}; \sigma_p^2)$ over the values of $a.C_p$. Given the gene process assignments, the variables $a.C_p$ for different arrays a are conditionally independent. Hence, for each a, we need to optimize:

$$\begin{aligned}
&\mathrm{argmax}_{a.\mathbf{C}} \sum_{e \in \mathbf{E}_a} \log\left(\frac{1}{\sqrt{2\pi}\sigma_a} \exp\left(\frac{(e.\mathit{Level} - \mu_e)^2}{2\sigma_a^2}\right)\right) \\
&\quad + \left(\sum_p \log\left(\frac{1}{\sqrt{2\pi}\sigma_p} \exp\left(\frac{(a.C_p - \mu_{p,a})^2}{2\sigma_p^2}\right)\right)\right) = \\
&\ \mathrm{argmin}_{a.\mathbf{C}} \frac{1}{2} \sum_{e \in \mathbf{E}_a} (e.\mathit{Level} - \mu_e)^2 \\
&\quad + \frac{\sigma_a^2}{2\sigma_p^2} \sum_p (a.C_p - \mu_{p,a})^2, \qquad (5)
\end{aligned}$$

where: $\mathbf{E}_a$ are all of the expression measurements associated with the array a, and for each $e \in \mathbf{E}_a$, μ_e is the mean of the Gaussian over the expression level, as specified in (1). Note that μ_e is a linear function of the values $a.C_p$, whose coefficients are determined by the (known values of) $g.M_p$. The form of (5) allows us to use a simple least squares computation to solve this minimization problem optimally and efficiently.

Algorithm Summary. We initialize the algorithm by using a standard expression clustering technique to select an assignment of genes to processes. The result is a model where processes are disjoint, and each gene belongs to precisely one process. We then find an initial set of activity levels using the least squares method, using a degenerate regression tree (with only a root node) as the regulatory model. The result of these two steps is an initial hard assignment to the variables $\mathbf{G}.\mathbf{M}$ and $\mathbf{A}.\mathbf{C}$.

With this initialization, the algorithm repeatedly executes an M-step and an E-step, as specified above. At each iteration, a new regulatory model is learned using the current hard assignments to $\mathbf{G}.\mathbf{M}$ and $\mathbf{A}.\mathbf{C}$. This regulatory model is then used to re-estimate new values for the activity levels $\mathbf{A}.\mathbf{C}$ and of the gene process memberships $\mathbf{G}.\mathbf{M}$.[1] The E-step and M-step are repeated until the gene process assignments stabilize. At that point, no further changes can occur, and the algorithm has converged. We then have a complete COPR model, including an assignment of genes to

[1]Note that each of these two substeps is executed only once before the next M-step; as these two substeps do not achieve a global optimum for the E-step, we could also iterate the substeps of the E-step until convergence, or for some number of steps. Our experiments suggest that this choice makes very little difference.

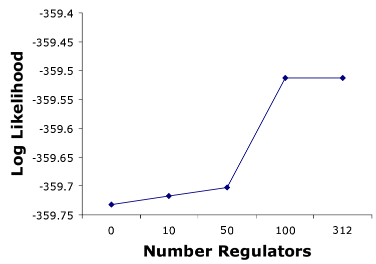


Figure 2: Variation of training set likelihood as number of regulators available to the model is increased from zero.

processes, an estimate of the activity level for each process in each array, and a learned regulatory mechanisms for each process.

5. Experimental Results

We evaluated the performance of our COPR model on yeast gene expression data. Our analyses included examining the statistical properties of COPR, evaluating our learned models with known biological attributes of genes and processes, and comparing COPR to other models on large expression data sets. In each case, we applied the learning algorithm described in Section 4, specifying a candidate set of regulators and a number of processes.

Statistical Validations. We first tested whether the COPR model, which includes both the partition of genes into overlapping processes as well as the regulatory program of each process, resulted in improved statistical behavior compared to approaches which do not consider regulatory programs, such as SBK [26]. To this end, we compared several models learned from the data set of Gasch *et al.* [11], specifying 20 processes, and using 1000 genes whose expression levels varied in the data set. For each model, we selected a different number of candidate regulators that the model could assign as parents of the activity levels of processes. We also compared these learned models to that obtained from the algorithm of SBK, which in our setting is equivalent to a model with no regulators and without a prior over the activity level variables.

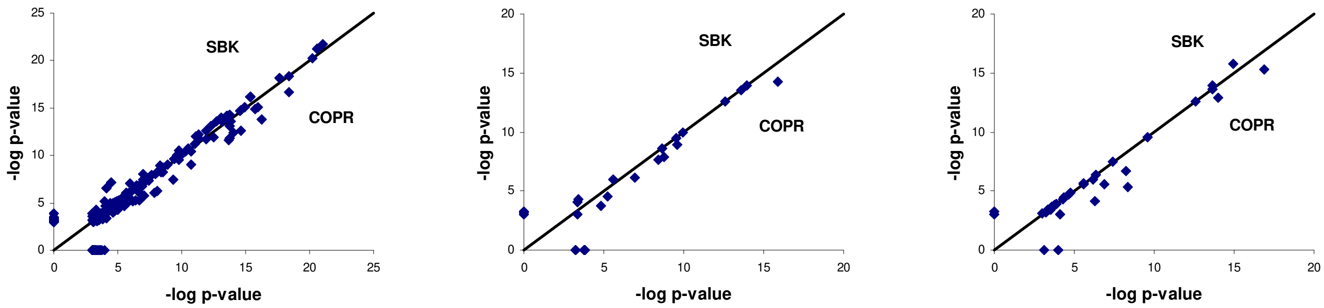
As a measure of performance, we first compared the likelihood that the different learned models assign to the expression data that was used to train the models, as this provides a measure of how well each model explains the input expression data. Specifically, we evaluated the probability $P(\mathbf{E}.Level \mid \mathbf{G.M}, \mathbf{A.C}, \mathcal{M})$, where $\mathbf{G.M}, \mathbf{A.C}$ are the values of the hidden variables learned by the algorithm, and $\mathcal{M}$ is the learned model. As shown in Fig. 2, models that use a larger number of regulators assign a higher likelihood to the input expression data. This finding is quite surprising, as the expression level is independent of the regulatory model given the learned activity levels $\mathbf{A.C}$. The SBK model can set these activity levels without constraints, attempting only to optimize this likelihood, whereas our COPR model is constrained to try to fit some regulatory program when selecting the same set of activity levels. In general, we would expect the unconstrained model to perform better, so the fact that the regulatory model provides higher likelihood indicates that the activity levels learned when we take into account regulatory models are actually more predictive of the expression levels. This behavior could result from an improved grouping of genes into processes, which would allow the process activity levels to more closely predict the actual gene expression levels.

However, a good fit to the training expression data does not necessarily imply that the model indeed captured true characteristics of the underlying domain. To test whether the COPR model captured meaningful properties, we evaluated the model performance on held out test data, using five-fold cross-validation over genes. Note that the values of the hidden variables — the gene process assignments — are unknown for genes in the test set. We evaluated the likelihood of the expression levels for the new genes by summing out over all possible assignments to the process memberships. Overall, the COPR model performed better than that of SBK, assigning a higher likelihood to the held out test data in four out of the five cases, and achieving only a slightly lower likelihood in the fifth. The average likelihood per gene in the test set for the COPR model was -454.88, compared to -455.76 for the model of SBK. A paired t-test on these results revealed a p-value of .04 for the improvement by the COPR model. Overall, these results indicate that the COPR model is, indeed, learning significantly better activity patterns than the model of SBK.

The experiment above demonstrates the ability of COPR to generalize to unseen genes. We also tested whether our COPR model can generalize to unseen experiments, by performing a similar five-fold cross validation scheme over arrays, using all 1000 genes. In this case, the activity levels of the test set arrays are not known. Using the expression levels of regulators in the held out arrays, our regulatory model can predict the activity levels of each process in each of these new arrays, and thus the expression levels for genes. We compared the COPR model's predictive power to a baseline model which uses no regulatory information, and simply predicts expression level to be normally distributed with mean zero. (This very naive assumption is the best we can do with no predictive information regarding activity.) COPR outperformed the baseline model ($p = 0.02$), achieving an average log-likelihood per experiment of -2687.81, as compared to -3097.42 for the baseline model. These results indicate that regulatory programs learned by the COPR model are in fact predictive of the expression levels of their target genes.

Yeast Stress Data. To obtain a more comprehensive model of biological processes and regulation, we applied our COPR method to the 173 yeast microarrays of Gasch *et al.* [12], which measured the response of yeast to various conditions of stress. We learned a COPR model over 2034 genes using 50 processes. Overall, the learned model had a reasonable partition of genes to processes, with 1384 genes predicted to participate in exactly one process, 308 in two processes, 287 in three processes, and only 40 genes in four or more processes. The learned regulation programs were quite rich, including a total of 321 regulators out of the 466 candidate regulators from which the learning algorithm was allowed to select.

To analyze the biological plausibility of the assignments of genes to processes, we tested whether the genes in each process were known to participate in the same process according to GO database of functional annotations [2]. For



(a) (b) (c)

Figure 3: Comparison of SBK model to our COPR model on yeast stress data. Each graph shows a scatter plot of the negative log p-value for the enrichment of processes for different biological properties, comparing processes in the SBK model (Y axis) and processes in our COPR model (X axis). (a) GO annotations; (b) presence of known motifs; (c) known transcription factor targets.

each process p and each GO annotation t, we used the hypergeometric distribution to assign a p-value for the enrichment of genes in process p that are assigned to annotation t according to GO. Of the 50 learned processes, 33 were highly enriched for some known biological function, with $p < 0.001$. We compared the p-value of these enrichments to the p-value enrichments obtained using the model of SBK [26]. We note that the SBK model was previously shown to produce more significantly enriched models than both standard clustering and Plaid. Our results, summarized in Fig. 3(a), show that the COPR algorithm learned processes whose genes are significantly more enriched compared to SBK [26], where 158 were significantly more enriched in the COPR model.

As the COPR model also learns the regulatory program for each process, we expect that genes assigned to the same process are also co-regulated. To test this hypothesis, we compiled a list of known binding sites from Wingender *et al.* [32], and associated each gene with the binding site that are contained within its 500bp upstream region sequence. We then tested whether genes that were assigned by the COPR model to the process were enriched for any of these known binding sites, where again we used the hypergeometric distribution to assign a p-value to each such test. Overall, 19 of the 50 processes contained genes with some shared binding site with $p < 0.001$. We also examined process enrichment for targets of 106 transcription factors, using the genome-wide protein-DNA binding data of Lee *et al.* [20]. We found that 20 of the 50 processes were enriched for targets of some transcription factor with $p < 0.001$.

A comparison to SBK, shown in Fig. 3(b) and (c), shows that genes assigned to the same process according to COPR were significantly more enriched for motifs as well as transcription factor targets compared to genes that were assigned to the same process according to SBK. Thus, we conclude that the inclusion of regulatory programs significantly improved our ability to detect truly coregulated sets of genes.

A more in-depth examination of our results revealed several cases where COPR processes were a particularly good fit to current biological knowledge. In particular, eleven processes were enriched for a motif m and an annotation t, where the transcription factor known to bind motif m has a known regulatory role in regulating process t. As one example, process 48 contained 13 genes, 10 of which were known to participate in glycolysis ($p < 9 \cdot 10^{-22}$). In addition, 10 of its member genes shared the binding site for the transcription factor GCR1, which is known to control glycolysis. From the binding data of Lee *et al.* [20], we also found that this process was enriched for targets of GCR2, which is also involved in glycolysis control. As another example, process 16 was enriched for protein folding and heat shock genes ($p < 1.8 \cdot 10^{-19}$). Importantly, genes in this process were also enriched for the binding site of the transcription factor HSF, a known regulator of protein folding and heat shock proteins. The binding data also indicated that this process was enriched for known targets of HSF, agreeing with both the GO and binding enrichments.

Next, we analyzed the quality of the actual regulatory programs that we learned for each process. In terms of their overall structure, the learned regression trees were plausible: most trees had one or two splits, no tree had more than four splits, and the majority of leaves had one linear parent. Our COPR model had multiple instances in which a regulator was used in more than one place in the regulation program. In most of these cases, the reused regulator appeared as a parent in the linear Gaussian leaf models of the regulatory program and not in the tree splits. This finding suggests that these two parts of the model correspond to different types of regulation, and that linear regulators can play a role in different combinatorial contexts; both of these conclusions are consistent with biological knowledge.

We also attempted to validate the actual regulators assigned in each regulatory program. Such validation is, by necessity, somewhat anecdotal, as the biological knowledge regarding the role of the regulators is very limited. Nevertheless, our analysis found ten processes in which the regulator had a known function in regulating the GO function that was enriched in the genes of the process.

Finally, there were many cases in which our learned model suggested novel hypotheses regarding gene regulation. For example, one of the processes we learned contained 16 genes with an unknown function. However, in the motif enrichment analysis, we found 7 motifs that were significantly enriched for the genes in this process (each occurring in at least 14 of the 16 genes). Moreover, two of the regulators assigned as part of the learned regulatory program, STE2 and GAC1, were associated with two of the enriched motifs. Combined with the coherent expression levels that we observed for these 16 genes, this result suggests a novel biological process and a putative regulatory program for controlling its activation.

Combined Yeast Data Sets. As our COPR model allows genes to participate in multiple processes, we can apply it to large compendia of expression measurements that correspond to many different conditions, in order to study both the common and the specific regulatory relationships in these different conditions. To this end, we compiled a large compendium of 394 yeast microarrays from four different studies, including measurements of expression during the cell cycle [30], in response to various stress conditions [12, 11], and in response to different gene deletion mutations [14].

From this combined data set, we learned a COPR model over 5747 genes, using 50 processes and 464 candidate regulators. To evaluate the partition of genes to processes, we performed the same enrichment evaluation as above, testing the enrichment of genes assigned to the same process for GO annotations [2], cis-regulatory motifs [32], and for binding targets of 106 transcription factors [20]. The results are shown in Fig. 4. Overall, we found 36 processes that were enriched for at least one GO annotation with $p < .001$, 27 processes significantly enriched for DNA binding motifs, and 22 that were enriched for the targets of one of the 106 transcription factors assayed by Lee *et al.*. In total, 45 of the 50 processes had some significant enrichment from one of the three sources.

We compared our results with those of the module network framework [28], with the hypothesis that a model allowing for overlapping processes would perform better on data compiled from different studies and spanning a wide range of experimental conditions. We note that unlike our COPR model, in the module networks framework, the regulators are also included in the gene modules, so the total number of genes grouped is slightly (about 10%) larger. As shown in Fig. 4, we found that our COPR model had noticeably more significant enrichments for GO annotations, regulatory motifs, and known transcription factor targets. In particular, out of 410 GO annotations enriched in either model, 312 were more significantly enriched in the COPR model than in the module networks model. Similarly, of the 32 motifs found to be enriched for either model, 28 were more enriched in the COPR model, and of the 52 transcription factor targets enriched in either model, 40 were more enriched in the COPR model. These results support the advantages of the two significant extensions of COPR over the module network model: allowing genes to participate in multiple processes and extending the regulatory model to encompass linear regulation.

In analyzing the learned regulatory programs, we found nine processes in which at least one of the predicted regulators had a known role in regulating the function associated with the GO annotation enriched for the process. As discussed, this type of analysis is limited by our very incomplete knowledge of regulatory relationships in the biological literature.

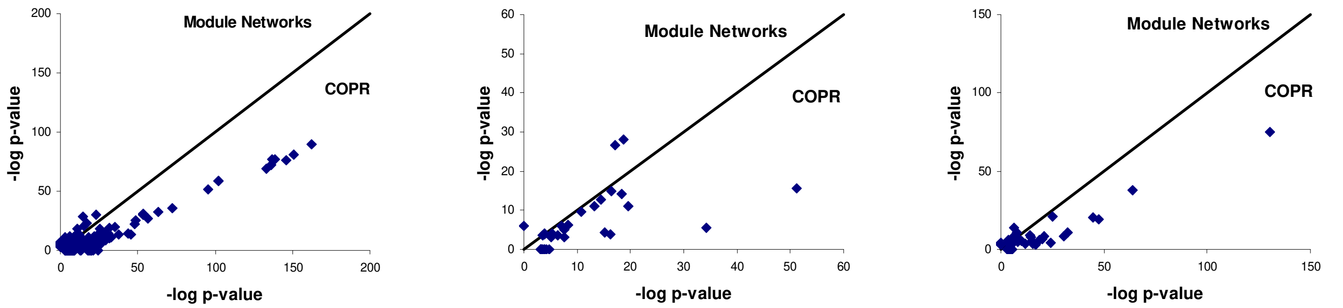
We next examined the predicted activity levels for each array. These are interesting as they provide evidence about cellular activity in different conditions; moreover, as these levels are predicted in part by the regulatory programs, they provide us with indirect evidence regarding the validity of these programs. We tested the correspondence between the activity levels and the actual conditions represented by each array. As we expect some processes to be differentially regulated in different conditions (e.g., heat shock proteins should be activated under various stress conditions), such correspondence would provide evidence that the learned activity levels correspond to true regulatory patterns.

Specifically, we compiled a list of 74 annotations for the different arrays, which specify the experimental condition that is represented by each array. For example, our list included annotations for the various cell cycle phases, as well as annotations for the different stress conditions to which the yeast was subjected (e.g., heat shock, nitrogen depletion). To measure how well our activity levels corresponded to these annotations, we performed a student t-test for each process p and each experiment annotation t, which measured whether the distribution of the activation levels was different between the arrays labeled with annotation t and those that are not labeled with it.

For all 50 processes, we found that their activation levels had a high correspondence with at least one annotation with $p < 0.025$. In 19 cases, the experiment annotation corresponded to the significant GO annotation of the process. For example, the activation levels for the (GO annotated) protein folding process had a high correspondence to the stress annotation. Protein folding is known to be activated in response to various stress conditions, such as heat and exposure to certain chemicals. A partial list of the significant associations between activation levels and experiment annotations is shown in Table 1. We also compared the experiment annotations associated with a process to transcription factors associated with it via motif or transcription factor target enrichment. Overall, we found 11 processes where the experiment annotation matched the function of the associated transcription factor.

In many cases, the genes in the process, and the activity profile associated with the process, appear to define a very coherent biological unit. For example, process 43 was enriched for the targets of three mitosis regulating transcription factors from the genome-wide protein-DNA binding data. This process was also enriched for a motif related to mitosis. The activity levels of this process were lowered significantly for the M-G1 phase annotated experiments (t-test value -3.18), and above normal for M-phase annotated experiments and experiments annotated with chromatin modification. This process has only a weak GO enrichment for the chromatin annotation, but the other evidence strongly suggests that it is associated with mitosis, perhaps with the transition from the M phase to the G1 phase in the cell cycle.

As another example, process 47 was strongly associated with GO annotations related to mating processes: “mating” ($p = 2.02x10^{-9}$), “pheromone response” ($p = 9.67x10^{-6}$), and zygote formation ($p = 1.5x10^{-5}$). Correspondingly, the activity level profile showed increased activity for experiments annotated for mating, pseudohyphal growth, and lowered activity in conditions such as salt stress, where we expect mating behavior to be repressed. The regulator RME1, which is associated with meiosis, was included in the learned program of this process. The process was strongly associated with two other transcription factors, MCM1, which regulates DNA replication, and of STE12, which is associated with pheromones, exhibiting enrichment both for the targets of these two transcription factors in the protein-DNA binding data, and for their motif in the genes’ promoter regions.



(a) (b) (c)

Figure 4: Comparison of the module network model of Segal *et al.* to our COPR model on the yeast compendium data. Each graph shows a scatter plot of the negative log p-value for the enrichment of processes for different biological properties, comparing processes in the module network model (Y axis) and processes in our COPR model (X axis). (a) GO annotations; (b) presence of known motifs; (c) known transcription factor targets.

Process	Experiment Annotation	t-test	GO annotation	p-value
47	mating	4.455	mating	2.202×10^{-9}
46	hyperosmolarity	12.083	plasma membrane	2.65×10^{-17}
45	protein modification	27.160	26S proteasome	2.187×10^{-54}
38	mitochondrion	3.527	mitochondrial inner membrane	3.11×10^{-22}
36	heat	4.183	heat shock protein	8.76×10^{-05}
26	peroxisome organization	9.293	peroxisomal matrix	5.94×10^{-09}
19	stationary phase	3.651	S phase of mitotic cell cycle	1.91×10^{-20}
17	chaperone activity	5.262	chaperone	7.06×10^{-20}

Table 1: A sample of activity level correlations with experiment annotations, paired with relevant GO annotations.

This process has strong support in the biological literature, which suggests that MCM1 and STE12 interact in the regulation of mating [22], and even suggests a mating pathway in which MCM1 and our predicted regulator RME1 are both involved [8, 18, 17].

6. Conclusions and Future Work

This paper proposes the COPR model — an approach for discovering overlapping cellular processes, their activity level in different conditions, and their regulatory programs from gene expression data. We have shown that by guiding the model to be consistent with regulatory programs, we obtain a better explanation of the data, and processes that are more biologically coherent. We have also shown several cases where the predicted regulatory programs are consistent with current biological knowledge. In many cases, our COPR model's predictions are remarkably coherent: A process associated with a certain cellular function is often predicted to be active in precisely the conditions where that function plays a role. Such coherent results involving uncharacterized genes or regulators can suggest novel biological hypotheses that can be tested in the lab.

There are several possible extensions to our work. First, our use of a unified probabilistic framework easily allows the modular integration of additional data sources. For example, rather than using the protein-DNA binding data of Lee *et al.* [20] solely for validating our result, we can directly integrate it into our model as a noisy sensor for regulation (as in [25]). Another direction is the application of our framework to Affymetrix microarray technology, which measures the absolute expression level of a gene rather than its level relative to some control. Our approach, constrained to include only non-negative activity levels, might provide a good decomposition of the observed expression levels into overlapping processes. Finally, it would be valuable to apply the COPR framework to the analysis of cellular processes and regulation in human microarray data, where we expect the transcriptional programs to be quite complex due to tissue-specific gene activation, so that genes are likely to play multiple roles.

Acknowledgements

This work was supported by NSF grant ACI-0082554 under the ITR Program. Eran Segal was also supported by a Stanford Graduate Fellowship.

7. REFERENCES

[1] O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing. In *Proceedings of the National Academy of Sciences 97 (18), pp. 10101-10106*, 2000.

[2] M. et al. Ashburner. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[3] A. Bjorck. *Numerical methods for least squares problems.* Siam, 1996.

[4] L. Breiman, J.H. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wardsworth International Group, 1984.

[5] Y. Cheng and G.M. Church. Biclustering of expression data. In *ISMB'00*, 2000.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B 39:1–39, 1977.

[7] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95(25):14863–8, 1998.

[8] L. Frenz, A. Johnson, and L. Johnston. Rme1, which controls cln2 expression in saccharomyces cerevisiae, is a nuclear protein that is cell cycle regulated. *Molecular Genetics and Genomics*, 266(3):374, 2001.

[9] N. Friedman. The Bayesian structural EM algorithm. In *Proc. UAI*, 1998.

[10] N. Friedman, I. Nachman, and D. Peér. Learning of Bayesian network structure from massive datasets: The "sparse candidate" algorithm. Submitted, 1999.

[11] A. P. Gasch, M. Huang, S. Metzner, S. J. Elledge, D. Botstein, and P. O. Brown. Genomic expression responses to dna damaging agents and the regulatory role of the yeast atr homolog meclp. *Molecular Biology Cell*, 12, 2001.

[12] A.P. Gasch, P.T. Spellman, C.M. Kao, O.Carmel-Harel, M.B. Eisen, G.Storz, D.Botstein, and P.O. Brown. Genomic expression program in the response of yeast cells to environmental changes. *Mol. Bio. Cell*, 11:4241–4257, 2000.

[13] D. Geiger and D. Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *Annals of Stastics*, 30(5):1412–40, 2002.

[14] T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–26, 2000.

[15] J Ihmels, G Friedlander, S Bergmann, O Sarig, Y Ziv, and N Barkai. Revealing modular organization in the yeast transcriptional network. *Nat Genet.*, 31:370–7, 2002.

[16] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proc. AAAI*, 1998.

[17] T. Kunoh, Y. Kaneko, and S. Harashima. Positive regulation of transcription of homeoprotein-encoding yhp1 by the two-component regulator sln1 in saccharomyces cerevisiae. *Biochemical and Biophysical Research Communications*, 278(2):344–348, 2000.

[18] T. Kunoh, Y. Kaneko, and S. Harashima. Yhp1 encodes a new homeoprotein that binds to the ime1 promoter in saccharomyces cerevisiae. *Yeast*, 16(5):439–449, 2000.

[19] L. Lazzeroni and A. Owen. Plaid models for gene expression data. Technical report, Stanford, 1999.

[20] T.I. Lee and *et al.*. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298:824–7, 2002.

[21] U. Lerner and R. Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence*, 2001.

[22] J. Mead, A. R. Bruning, M. K. Gill, A. M. Steinera, T. B. Acton, and A. K. Vershon. Interactions of the mcm1 mads box protein with cofactors that regulate mating in yeast. *Mol. Cell. Biol.*, 22:4607–4621, 2002.

[23] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[24] D. Pe'er, A. Regev, G. Elidan, and N. Friedman. Inferring subnetworks from perturbed expression profiles. In *ISMB'01*, 2001.

[25] E. Segal, Y. Barash, I. Simon, N. Friedman, and D. Koller. From sequence to expression: A probabilistic framework. In *Proc. RECOMB*, 2002.

[26] E. Segal, A. Battle, , and D. Koller. Decomposing gene expression into cellular components. In *Proc. PSB*, 2003.

[27] E. Segal, D. Pe'er, A. Regev, D. Koller, and N. Friedman. Learning module networks. In *Proc. UAI*, 2003.

[28] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics*, 2003.

[29] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller. Rich probabilistic models for gene expression. *Bioinformatics*, 17(Suppl 1):S243–52, 2001.

[30] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell*, 9(12):3273–97, 1998.

[31] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. In *Proc. ISMB*, 2002. Bioinformatics 18, Supplement 1.

[32] E. Wingender and *et al.*. The TRANSFAC system on gene expression regulation. *Nucleic Acids Research*, 29:281–283, 2001.