DECOMPOSING GENE EXPRESSION INTO CELLULAR PROCESSES

E. SEGAL AND A. BATTLE AND D. KOLLER
Computer Science Department
Stanford, CA 94305-9010
E-mail: eran@cs.stanford.edu,ajbattle@stanford.edu,koller@cs.stanford.edu

Abstract

We propose a probabilistic model for cellular processes, and an algorithm for discovering them from gene expression data. A *process* is associated with a set of genes that participate in it; unlike clustering techniques, our model allows genes to participate in multiple processes. Each process may be active to a different degree in each experiment. The expression measurement for gene g in array a is a sum, over all processes in which g participates, of the activity levels of these processes in array a. We describe an iterative procedure, based on the EM algorithm, for decomposing the expression matrix into a given number of processes. We present results on Yeast gene expression data, which indicate that our approach identifies real biological processes.

1 Introduction

A living cell is a complicated system that performs multiple functions and has to respond to a variety of signals. To organize this complex web of activity, the cell tends to compartmentalize its activity into distinct *processes*, or modules. This global organization cannot be discerned by studying the properties of isolated components. Genome-wide measurements of mRNA expression level across multiple experimental conditions provide us with a global picture of the cell's activities, and provide the potential for a high-level understanding of its behavior.

Clustering techniques are the most common approaches to identifying functional groups in gene expression data. These approaches generate clusters of genes that have similar expression profiles over a range of experimental conditions [6, 4, 14]. However, these approaches group genes into mutually exclusive clusters, and are thus limited in their ability to represent the true underlying biological system: many genes are known to be multi-functional, and thus should belong to more than one functional group.

In this paper, we introduce a probabilistic framework for discovering biological processes from expression data. Each process is associated with a set of genes that participate in it; unlike clustering methods, our model allows genes to participate in multiple processes. Each process might be more active in some conditions and less active in others. Thus, our model defines for each experiment the extent to which each process is active in that experiment. In our model, the expression measurement for gene g in array a is a sum, over all processes in which g participates, of the activity levels of these processes in array a. Thus, we decompose the entire expression matrix as a sum of the expression levels of all the active processes.

Our model resembles the Plaid model proposed by Lazzeroni and Owen [11]. The Plaid model also decomposes the expression data as a sum of overlapping "layers"; each layer is associated with a set of genes and experiments that define the expression of that layer. There are several differences between our model and Plaid. Of these, the most important is that Plaid uses a greedy sequential approach to perform the decomposition, attempting, in each step, to explain as much of the unexplained expression data as possible. Once a layer has been learned, it remains unchanged and is subtracted from the expression data. In contrast, our model is trained as a unified whole, allowing the association of genes with processes to change as the process models become more refined, and the process models to change as the set of assigned genes changes. As we demonstrate in our experimental results, our approach discovers much "cleaner" processes than does Plaid: The set of genes associated with a process by our approach often contains a very high fraction of genes that are known to share a functional role. By contrast, Plaid layers are much larger and more heterogeneous, and do not correspond as neatly to a biological process.

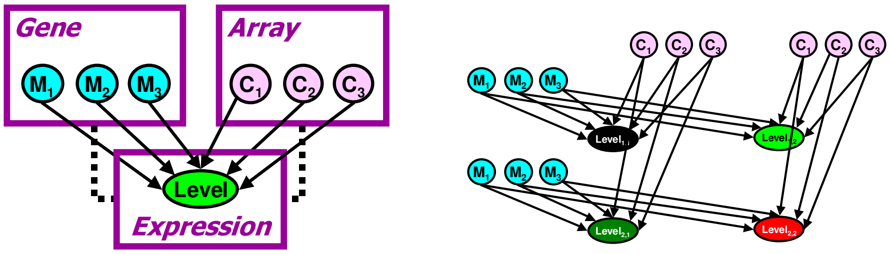
We provide an iterative algorithm for learning the model from gene expression data, based on the *expectation maximization (EM)* algorithm [5]. Once a model has been learned, we can read the processes directly from it. For each process, we read both the genes that participate in it as well as the levels of activity of each process in all experiments. We describe encouraging results on real data, providing evidence that our approach identifies real biological processes. Specifically, we show a high correlation between the gene sets constructed and known biological processes. We also show significant DNA binding sites in the promoter regions of the genes in the process. Finally, we show cases where our learned activity levels for processes had extremely high correlations with the expression levels of known regulators of those processes; importantly, these regulators were *not* part of the input data given to our program, indicating that our program reconstructed the levels of activity of these processes.

2 Probabilistic Model

In this section we present our probabilistic model. Our approach is based on the language of *probabilistic relational models (PRMs)*, as described in [10, 7]. For lack of space, we do not review the general PRM framework, but focus on the details of the model, which follow the application of PRMs to gene expression in [14]. A simplified version of our model is presented in Fig. 1(a); we now describe its elements.

The PRM framework represents the domain in terms of the different biological entities that interact in it: genes, arrays, expression measurements, and biological processes. Also, each object may be associated with a set of attributes that are relevant to the interactions in the domain. Specifically, our model 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 $e.\mathit{Array} = a$.

In addition, we include a set of j *process* objects. Our model makes explicit the notion that genes participate in biological processes and that processes are active to varying degrees in arrays. Thus, for each gene object g we define a set of binary



(a) (b)

Figure 1: (a) PRM for the process model. (b) An instantiation of the PRM to a particular dataset with 2 genes, 2 arrays and 3 processes.

attributes, $g.M_1, \ldots, g.M_j$, where $g.M_p$ represents the gene's *M*embership in process p. To represent process activity levels, we associate with each array object a, a set of continuous attributes $a.C_1, \ldots, a.C_j$, where $a.C_p$ represents the a*C*tivity level of process p in array a.

Each expression measurement e associated with gene $g = e.Gene$ and array $a = e.Array$ is assumed to be a (stochastic) function of the processes in which g participates and of the activity level of those processes in the array a. More precisely, let $g.\mathbf{M}$ be the set of all g's membership variables, and $a.\mathbf{C}$ be the set of all a's activity level variables. We assume that $e.Level$ is normally distributed with mean $\mu_e = \sum_p g.M_p \cdot a.C_p$ and standard deviation σ_a. More precisely:

$$P(e.Level) = \exp\left(\frac{(e.Level - \sum_p g.M_p \cdot a.C_p)^2}{2\sigma_a^2}\right) \qquad (1)$$

where σ_a is the standard deviation of all expression measurements in array a. Thus, the expression measurement for gene g in array a can be viewed as the sum over expression components, with each component being the result of the activity in array a of some process to which g belongs.

To complete the description of our probabilistic model, we associate with each process p a prior probability $P(g.M_p) = q_p$, which is the prior probability with which any gene participates in p. We also associate with the continuous attribute $a.C_p$ a uniform distribution (over some appropriately bounded range).

Although the description of our model is compact, its instantiation to a particular data set is quite large. In a specific instantiation of the PRM model we might have 10 processes, 1000 genes and 100 arrays. Thus, we have as many as 1000×100 expression objects (if all expressions are observed), so the instantiation of our model to a particular dataset contains a large number of objects and variables that interact probabilistically. The resulting probabilistic model is a *Bayesian network* [12], where the local probability models governing the behavior of nodes of the same type (e.g., all nodes $g.M_p$ for different genes g in process p) are shared. Fig. 1(b) shows a small instantiation of such a network, for two genes, two arrays, and three processes.

Putting everything together, an instantiation of the PRM model specifies the membership $g.M_p$ for all genes $g \in \mathbf{G}$ and all processes p, the activities $a.C_p$ for all arrays $a \in \mathbf{A}$ and all processes p, and the expression level $e.Level$ for all $e \in \mathbf{E}$. The joint distribution over all possible instantiations is given by:

$$P(\mathbf{G}.\mathbf{M}, Arrays.\mathbf{C}, \mathbf{E}.Level) = \left(\prod_p \left(\prod_{g \in \mathbf{G}} P(g.M_p)\right)\left(\prod_{a \in \mathbf{A}} P(a.C_p)\right)\right)\left(\prod_{e \in \mathbf{E}} P(e.Level \mid g.\mathbf{M}, a.\mathbf{C})\right) \quad (2)$$

where $a = e.Array$ and $g = e.Gene$.

Our model has several desirable properties. First, processes are represented explicitly making it easy to "read" off processes from the model: the $g.M_p$ variables tell us which genes participate in process p and the $a.C_p$ variables tell us the activity level of each process p in each array. Second, the model allows for genes to participate in more than one process (since we could have $g.M_p = 1$ and $g.M_q = 1$ for different processes p and q), which enables us to model multi-functional genes. Finally, as we will see in the next section, the probabilistic model allows us to utilize statistical optimization techniques to learn the models from data over several processes jointly.

3 Learning the Model

In the previous section, we described the different components of our probabilistic model. We now consider how we learn this model from data. We assume that the only information given is the expression data itself, and the number of processes we wish to identify. We do not know which genes participate in which processes nor the levels of activity of arrays processes. Thus, all attributes $g.M_p$ and $a.C_p$ are hidden. From the perspective of the model parameters, the prior probabilities of the different processes — $P(g.M_p) = q_p$ are also unknown. We assume that the expression level model is given, as in Eq. (1), and that the distribution $P(a.C_p)$ is uniform and fixed for all a, p.

The learning problem we have here is quite complex, as it involves a large number of hidden variables. The main technique we use to address this issue is the *Expectation Maximization (EM)* algorithm [5], which allows parameter estimation with incomplete data. The EM algorithm is an iterative method. Starting from an initial setting for the parameters, it repeatedly performs two steps. In the *E-step*, it computes the distribution over the unobserved attributes $g.M_p, a.C_p$ (for all g, a, p), given the observed expression data and the current estimate of the parameters. It uses this distribution to "fill in" each missing attribute. Two variants of the EM algorithm are *soft EM*, where the completion explicitly accounts for the probability over the values of each missing attribute, and *hard EM*, which simply selects the single most likely assignments to each attribute. The *M-step* re-estimates the parameters, using the completion of the missing attributes as if it were real, using a standard maximum likelihood estimation procedure. The process then repeats, using the new parameters, until convergence. Soft EM is guaranteed to converge to a local maximum of the likelihood of the observed data $P(\mathbf{E}.Level)$; hard EM is guaranteed to converge to a local maximum of the joint likelihood of the observed data *and the completion* —$P(\mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathbf{E}.Level)$.

In applying either variant of EM to our model, we must deal with the complexities of the E-step, which require that we compute the distribution over all assignments to both $\mathbf{G}.\mathbf{M}$ and $\mathbf{A}.\mathbf{C}$. As we discussed, our model induces a complex set of interactions. In the experiments we describe below, we have 1010 genes, 173 arrays, and 10 processes. This results in a model with $11,830$ hidden variables. Moreover, the nature of the data is such that we cannot treat genes (or arrays) as independent samples. Instead, any two hidden variables are dependent on each other given the observations (see [7] for an elaboration of this point). For example, consider two genes g and h; h's assignment to processes influences our estimates of the $a.C_p$ variables, which in turn influence our membership probabilities for g. Due to these dependencies, the exact computation of the E-step is intractable for large domains.

However, our model is such that if we were given the values of $a.C_p$ for all a, p, then the assignments of the different genes to processes are rendered independent. Likewise, given a fixed assignment of genes to processes ($g.M_p$), the activity levels for each array a can be estimated from these assignments and from the expression data in a alone, without knowing the activity levels in other arrays.

This key observation suggests the use of hard assignments to the hidden attributes, rather than a soft assignment. Thus, we use a variant of hard EM. In this case, our goal in the E-step is to find the maximum probability assignment to the variables $\mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}$, given the current parameter setting. This problem is still intractable, but we can use our observation to find a very good local maximum.

Specifically, starting from an initial assignment of genes to processes (which could come from standard clustering methods), we find the most likely activity levels $\mathbf{A}.\mathbf{C}$. We then fix these activity levels, and find the most likely assignment to $\mathbf{G}.\mathbf{M}$. Each step increases the joint likelihood $P(\mathbf{G}.\mathbf{M}, \mathbf{A}.\mathbf{C}, \mathbf{E}.\mathit{Level})$ given the current parameters, and thus the process is guaranteed to converge. The resulting assignment to these variables is a fairly strong local maximum: No step that adapts only the gene memberships or the array activity levels can improve the likelihood; however, a step that adapts both gene memberships and array activities might.

At convergence, the E-step is complete, and we can use the final assignment $\mathbf{G}.\mathbf{M}^*$, $\mathbf{A}.\mathbf{C}^*$ to estimate the parameters q_p, using standard maximum likelihood estimation. More precisely, we compute the *expected* sufficient statistics:

$$N_{\mathbf{G}.M_p}[v] = \sum_{g \in \mathbf{G}} \eta(g.M_p^* = v) \tag{3}$$

where η is an indicator variable that takes value 1 when its argument holds and 0 otherwise. We then compute the probabilities q_p as:

$$q_p = \frac{N_{\mathbf{G}.M_p}[1]}{N_{\mathbf{G}.M_p}[0] + N_{\mathbf{G}.Memberp}[1]}. \tag{4}$$

The algorithm as a whole is shown in Fig. 2. As we can see, it defines two separate optimization tasks: finding the most likely memberships of genes in processes given activity levels (step 2(a)i), and finding the most likely activity levels given the memberships of genes in processes (step 2(a)ii). We discuss the implementation of these steps in the subsequent subsections.

1. Initialize $\mathbf{G}.\mathbf{M}$ using standard clustering techniques.
2. Repeat until convergence
   (a) **E-step** Repeat until convergence
      i. Find the assignment to each activity level $a.\mathbf{C}$ that maximizes $P(a.\mathbf{C} \mid \mathbf{E}.\mathit{Level}, \mathbf{G}.\mathbf{M})$.
      ii. Find the assignment to each gene membership $g.\mathbf{M}$ that maximizes $P(g.\mathbf{M} \mid \mathbf{E}.\mathit{Level}, \mathbf{A}.\mathbf{C}$.
   (b) **M-step** Estimate the parameters q_p as in Eq. (3) and Eq. (4).

Figure 2: Full Learning Algorithm

A critical part of our approach is that our algorithm does not learn the membership and activity of each process in isolation. Rather, our model is learned over all processes simultaneously, allowing information and (probabilistic) conclusions from one process to propagate and influence our conclusions about another. For instance, assume that our learning process places a gene g into process p at some step, and that this membership explains g's expression data very accurately. In this case, g will be less likely to be a member of other processes, allowing other genes assigned to the process to have a stronger influence on the activity level profile of the process.

One of our two tasks is to find the most likely activity levels given the memberships of genes in processes. Here, we assume that we are given, for each gene g, all the processes p in which it participates. Thus, we now need only to find the most likely assignment to the activity levels of arrays in processes, i.e., $\mathrm{argmax}_{\mathbf{A}.\mathbf{C}} P(\mathbf{A}.\mathbf{C} \mid \mathbf{E}.\mathit{Level}, \mathbf{G}.\mathbf{M})$. Using Bayes rule, our assumption of uniform prior over each $a.C_p$, and the model in Eq. (1), we can reformulate our maximization task as

$$\mathrm{argmax}_{\mathbf{A}.\mathbf{C}} P(\mathbf{E}.\mathit{Level} \mid \mathbf{A}.\mathbf{C}, \mathbf{G}.\mathbf{M}) =$$
$$\mathrm{argmax}_{\mathbf{A}.\mathbf{C}} \sum_{a \in \mathbf{A}} \sum_{e \in \mathbf{E}._a} \log \left(\frac{1}{\sqrt{2\pi}\sigma_a} \exp \left(\frac{(e.\mathit{Level} - \mu_e)^2}{2\sigma_a^2} \right) \right)$$

where $\mathbf{E}._a$ is the column of the expression matrix that corresponds to the array a. Simple algebraic reformulation shows that this problem is, in fact, a standard least squares problem $E \approx GC^T$, where: E is the standard $n \times k$ expression matrix; G is an $n \times j$ 0-1 matrix such that $G_{g,j}$ contains a 1 if gene g is a member in process j and C is a $k \times j$ matrix such that $C_{a,p}$ represents the activity level of array a in process p.

The matrices E and G are both fixed, and our goal is to find the matrix C that minimizes the squared-error for $E \approx GC^T$. It is well-known [9] that a least-squares solution to this system exists, and can be found effectively using standard methods.

We now turn to our second optimization problem, where we are given the activity levels of all processes in all arrays and our task is to learn the assignments of genes to processes. Thus, the attributes $g.M_p$ are hidden for all genes g and all processes p. However, with all activity levels given, assignments of genes to processes are independent across genes and we can find the most likely assignment for each gene separately.

To perform this maximization, we maximize $P(g.\mathbf{M} \mid \mathbf{E}_g, \mathbf{A}.\mathbf{C})$ separately for each gene g, where $\mathbf{E}_g$ is the row in the expression matrix corresponding to the gene

g. More precisely, this computation can be done as follows:

$$\mathbf{v}^* = argmax_{\mathbf{v}'} P(g.\mathbf{M} = \mathbf{v}' \mid \mathbf{E}_g, \mathbf{A}.\mathbf{C}), \tag{5}$$

where

$$P(g.\mathbf{M} = \mathbf{v} \mid \mathbf{E}_g, \mathbf{A}.\mathbf{C}) = \alpha \prod_p P(g.M_p = v_p) \prod_{e \in \mathbf{E}_g} P(e.\mathit{Level} \mid g.\mathbf{M} = \mathbf{v}, \mathbf{A}.\mathbf{C}),$$

where $v_p \in \{0, 1\}$, and α is a normalization constant. The expression inside the final term in the product is simply the Gaussian model for the expression level given its parents, as in Eq. (1).

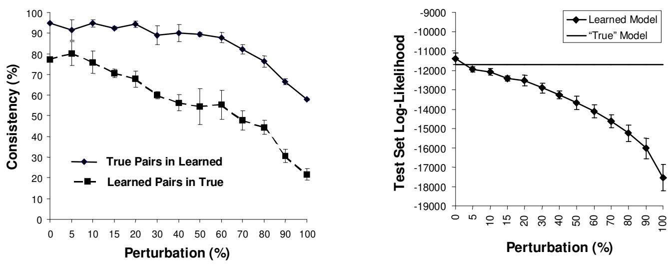
For models that include a large number of processes, we cannot perform this maximization over $g.\mathbf{M}$ exactly. The number of calculations required for each gene is exponential in the number of processes, since every possible joint assignment to $g.\mathbf{M}$ must be considered. In these cases, we use an approximation. Instead of considering every possible assignment to $g.\mathbf{M}$, we *include* only a subset $g.\mathbf{M}_I$ of processes, and *exclude* all others $g.\mathbf{M}_E$, forcing their value to 0. To select our subset, we relax the problem and allow each $g.M_p$ to be any real value between 0 and 1. We then maximize Eq. (5) subject to this relaxation. This problem reduces to a bounded least squares problem, which we can solve exactly [3]. We then select $g.\mathbf{M}_I$ from $g.\mathbf{M}$, by choosing those variables whose relaxed assignments are closest to 1 (in practice the majority of the variables are assigned to 0 in the relaxed solution). Finally, we find the most likely $\{0, 1\}$ assignment to $g.\mathbf{M}_I$ with all other variables fixed to 0 by maximizing:

$$\begin{aligned} &P(g.\mathbf{M}_I = \mathbf{v} \mid \mathbf{E}_g, \mathbf{A}.\mathbf{C}, g.\mathbf{M}_E = 0) = \\ &\quad \alpha \prod_{p \in I} P(g.M_p = v_p) \prod_{e \in \mathbf{E}_g} P(e.\mathit{Level} \mid g.\mathbf{M}_I = \mathbf{v}, g.\mathbf{M}_E = 0, \mathbf{A}.\mathbf{C}) \end{aligned}$$

4 Model Evaluation

We first evaluated our approach on synthetic data. These experiments test whether we recover structure known to be present in the data. We generated a synthetic data set by sampling from a PRM model. To make the data realistic, we used PRM models learned from real biological data [8]. Specifically, we first learned a model with 7 processes. We then sampled data for 500 genes and 173 experiments (the original data contained 173 experiments) from the model: assignments of genes to processes were sampled from the distribution our model had for the $\mathbf{G}.\mathbf{M}$ variables, and expression data was then derived by computing the expected expression levels (according to our model of expression) from the sampled assignment of genes to layers and the $a.\mathbf{C}$ means which were part of the learned model.

We then hid the true assignments of genes to processes and activity levels in arrays, as well as the original model parameters q_p, and learned a model with 7 processes from the synthetic expression data using the algorithm described in Section 3. To test the robustness of our learning algorithm to noise, we also learned models using various levels of perturbations, where a perturbation level of π corresponds to shuffling $\pi\%$ of



(a) (b)

Process	Genes	Motif consensus	p-value
23	35	cgagccaatcaaaac	$2.37e-20$
27	76	gactatgactcacag	$3.92e-20$
12	18	tgacccaatcagatg	$4.99e-19$
25	36	cgcggcggtgaggct	$5.66e-18$
18	14	agattctagaaaacc	$1.60e-17$
16	48	aggatgggcgggagg	$2.23e-17$
2	19	cccgtcagcctgaac	$5.04e-17$
21	81	gttgtctcagcacct	$2.06e-15$
6	17	acaacacacaatcag	$5.95e-15$
0	21	tggctgactaattca	$6.48e-15$
7	17	tgagcgaagctccgc	$7.51e-15$
29	60	agctaggggtaagga	$9.91e-15$
19	16	cgcgccctaaaatcc	$3.78e-14$

(c)

Figure 3: (a) Fraction of learned pairs appearing in the true data and fraction of true pairs in the learned model for various levels of perturbations. (b) Log-Likelihood on test data achieved for learned models for various levels of perturbations. (c) Motifs learned by searching for commonalities in the upstream regions of the genes in each process.

the expression data across all genes and experiments. To gain statistical confidence, we generated five data sets for each perturbation π, and learned a model from each.

All models were evaluated by their ability to recover the "true" assignments of genes to processes (true assignments are the assignments in the sampled data) by performing a pairwise consistency test: we extracted all gene pairs appearing in the same process in our learned model, and computed the fraction of these pairs appearing in the true data. We also tested the reverse, extracting all the true pairs and computing the fraction of these pairs appearing in a learned model. The results are summarized in Fig. 3(a), indicating that our algorithm reconstructs the true structure with very high accuracy even if 30% of the data is perturbed: gene pairs assigned to the same process in the true data, are likely to appear in our learned model and vice versa. Note that in fully randomized data (100% perturbation), a high fraction of the pairs in the learned model were indeed present in the true data ($58 \pm 0.4\%$). This occurs since the randomized data contains much weaker patterns and the total number of pairs learned is small, as can be seen by the poor coverage ($21.7 \pm 2.8\%$) of true pairs in these models.

As another evaluation, we measured the ability of our learned models to predict unseen data, by computing the likelihood that each model assigns to held out data. Specifically, we randomly partitioned the data into five equally sized sets of 100 genes and learned five models from all five possible combinations of four sets. For each such model we computed the likelihood it assigned to the held out subset. We compared these results to the likelihood that the "true" model from which the data was sampled assigned to the held out test data. These experiments were also performed in the presence of varying levels of perturbations. The results are summarized in Fig. 3(b). As can be seen, the test set likelihood is comparable (and even better with very little noise) for up to 30% perturbations, dropping sharply as more noise is added.

Recall that when the number of processes is large, we resort to the approximation described in Section 3. To evaluate our approximate algorithm, we learned a 12 process model, where we could apply the exact algorithm and compare the results. In our results, 77.2% of the genes had the same assignment to processes in the approximation and exact algorithm. However, the training and test set likelihoods of both models were practically the same, implying that the errors made by the approximation had little effect.

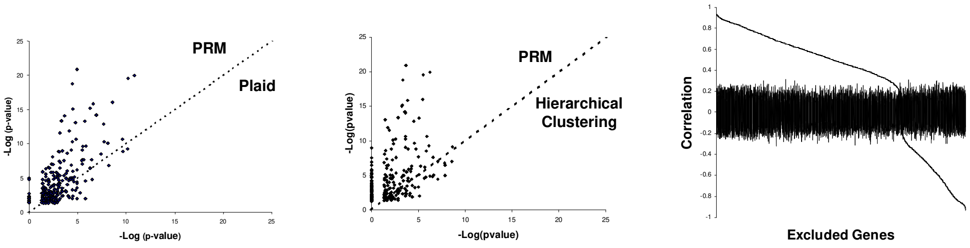
5 Biological Analysis

We now consider the data set of Gasch *et al.* [8], who characterized the genomic expression patterns of yeast genes in 15 different experimental conditions. We selected 1010 genes that had significant changes in gene expression (eliminating the ESR genes for which clustering is trivial), and the full set of 173 arrays.

We used the model discussed above, with 30 processes. Overall, our model predicted that 24 genes do not participate in any process, 552 genes participate in only one process, 257 in two, 119 in three, and 58 in four or more processes. As a comparison, we also tested a Plaid model with 30 processes, learned from the same data. (We obtained the Plaid software from `http://www-stat.stanford.edu/~owen/plaid/`.) The Plaid model assigned many more genes to layers than our model did, with 0 genes in no processes, 1 gene in one process, 4 genes in two, 10 genes in three, and 995 genes in four or more processes. According to Plaid, almost *all* genes participate in four or more processes, a situation not supported by current biological understanding. We note that the running time of our algorithm was 30 minutes on a 700MHZ Pentium 4, compared to 1 minute for running Plaid on the same machine.

To evaluate whether our assignments are biologically plausible, we checked whether the genes associated with each process showed any enrichment for known annotations. To do so, we used the GO [1] and KEGG [2] databases which assign genes to a diverse set of functional categories and biological pathways, respectively. For each process and each annotation, we counted the number of genes from the process with that annotation, and compared that to the total number of genes in our dataset with that annotation. If a process we learned indeed corresponds to known biological processes, then we expect the learned process to contain a high fraction of the genes with the corresponding annotation. For each combination of process p and annotation α, we can use the hyper-geometric distribution and assign a statistical significance (p-value) measure, corresponding to the probability that a randomly selected group of genes of the same size have similar enrichment for α. We performed this evaluation for our processes, the layers found by the Plaid model, and clusters from a standard clustering procedure.

The web supplement to this paper (`http://cs.stanford.edu/~eran/psb03`) lists the 30 processes, along with all annotations that were significant with a p-value of $1e-3$ or lower in either our model or in Plaid, where we removed some repetitive annotations from GO. Overall, we discovered highly significant processes relating to a variety of cellular functions. These included oxidative phosphorylation, various transport processes, protein folding, glycolysis, lipid metabolism, amino acid metabolism, carbohydrate metabolism, protein membrane targeting, ribosomal biogenesis, and cell cycle control. Some of the stronger active processes we identified were also present as Plaid layers, but Plaid layers typically included many extraneous genes, rendering the patterns less clear. For example, neutral lipid metabolism appears as a process of 17



(a) (b) (c)

Figure 4: (a) Scatter plot of the negative log p-value of different GO and KEGG annotations for layers in Plaid on the one hand (X axis) and processes in our framework on the other (Y axis). Each point corresponds to one annotation. (b) Scatter plot of the negative log p-value of different GO and KEGG annotations for clusters from Pearson clustering on the one hand (X axis) and processes in our framework on the other (Y axis). (c) Correlation between all genes not included in the analysis and the learned activity levels. For each gene, the best correlation (or anti-correlation) is plotted as well as the best correlation achieved for that gene after permuting its expression measurements. The genes are sorted by best correlations.

genes with a p-value of $1.26e-21$ in our model, while in Plaid it appeared in a layer of 317 genes with a p-value of $1.22e-5$. Also, protein folding appeared as a process of 14 genes with a p-value of $1.46e-16$ in our model, while the corresponding Plaid layer had 254 genes with a p-value of $2.65e-7$. Fig. 4(a) shows a scatter plot comparing the p-value for the GO and KEGG annotations that came up. We can see that in most cases (122 of 135 cases for pvalue of $1e-4$ or lower), the p-value achieved by our approach was always better and often much better than that achieved by Plaid. We performed a similar comparison to a standard hierarchical clustering algorithm [6], where we cut the hierarchy at 30 clusters to allow for a comparison to our model. The results are shown in Fig. 4(b), where again the majority of annotations appeared with greater significance in our model.

If genes assigned to the same process indeed participate together in a biological process, then the cell must have some regulatory mechanism by which it can coordinate their activity. One such mechanism is a shared DNA binding site (or multiple sites) recognized by a transcription factor (or several). To test whether genes that we associated with a process share DNA binding sites, we extracted the promoter regions of all genes (500bp upstream of translation start site) and applied a discriminative motif finder [13], searching for motifs of length 15. The result of the search is a standard position specific scoring matrix (PSSM) which can then be used to compute which genes have the binding site defined by the PSSM and which do not. From this we can derive a statistical significance measure assessing the uniqueness of the binding site to the set of genes associated with the process relative to the entire genes in the dataset. The consensus sequence of the best PSSMs learned along with the statistical significance of the PSSM to the process are summarized in Fig. 3(c). Overall, we were able to find unique binding sites in the set of genes in each process (see web supplement for full list), often with striking significance, consistent with significant annotations identified for each process using GO or KEGG. For example, we found a highly unique DNA binding site (p-value $2.37e-20$), occurring in the promoter region of 28 of the

35 genes in the oxidative phosphorylation pathway (process 23), compared to 102 appearances in the remaining 976 genes in the dataset, suggesting possible regulation of the pathway by this binding site. Indeed, the core element of this binding site is $ccaat$, which is the known target for the transcription factor regulator of oxidative phosphorylation, HAP4.

In addition to the assignments of genes to processes, our approach attempts to reconstruct the activity levels of each process p in each array a, as captured by the posterior mean of $a.C_p$. For each process, we can thus construct a vector $\mathbf{A}.C_p$ of the activity levels of p across all arrays $a \in \mathbf{A}$. We examined these activity levels, and found that they were biologically plausible for their respective processes. For instance, the process associated with protein folding (process 18) had high activity levels during heat shock and exposure to diamide, and low activity levels during amino acid and nitrogen depletion, reflecting accurately the biological function of the process.

Also, genes associated with process p should have high correlations between $\mathbf{A}.C_p$ and their average expression across all experiments. Indeed, a large number of genes were either highly correlated (286 genes with correlation 0.8 or above) or highly anti-correlated (13 genes with correlation -0.8 or below).

Much more exciting is to measure the correlation between the $\mathbf{A}.C_p$ vectors for all processes p and the 5147 genes that were *not* included in our analysis. Due to the way in which we selected the 1010 genes for our analysis, the genes included are likely to contain only a fraction of the genes associated with each process. If our model learned activity levels that indeed correspond to activity levels of real processes, then we expect to see high correlations between some of the left out genes and our learned activity levels. Indeed, there were many such genes: 614 of correlation above 0.8 and 252 of correlation below -0.8. To test whether this phenomenon could have happened by chance, we permuted the vector of expression measurements for each gene and recomputed the correlations. The results are summarized in Fig. 4(c), demonstrating that it is highly unlikely that our computed correlations could have resulted by chance, as the most significant correlation achieved for any of the 5147 permuted genes was -0.32. Surprisingly, the distributions of the correlation measurements were identical between the genes included in the analysis and those not included (data not shown).

Interestingly, there were several cases where the learned process activity levels had high correlation to the expression of known regulators (e.g., transcription factors) not included in the analysis. The web supplement lists all regulators with high correlation (or anti-correlation) to any process. Overall, we had 37 unique regulators with correlation above 0.7, of which 10 had correlation above 0.8, and 8 unique regulators with correlation below -0.7, of which 5 had correlation below -0.8. For 12 of the 30 processes, we learned activity levels that had extremely high correlation with known regulators. When information about the regulator was available in the literature, we could verify that the regulator that was highly correlated to a process, indeed was known to regulate the genes associated with that process. For example, CLB2, a G2/M phase specific cyclin, had correlation 0.88 with process 12, which in turn has significant cell cycle related annotations. Even when information was not available to verify our proposed regulation relationships, the regulators were known to be related to glucose starvation, cell wall stress, cell growth, cyclic AMP, ribosome synthesis, nitrogen starvation and mating, all processes known to be affected by the conditions in the Gasch [8] dataset.

6 Conclusions

In this paper, we have presented a probabilistic framework for extracting biological processes from gene expression data. Unlike most clustering methods, our approach does not attempt to associate each gene with a single process. Rather, it attempts to explain each gene's expression level as a sum of of the activity levels of the processes to which is belongs. For each process, we learn a set of genes that are associated with the process, and the extent to which the process is active in each array.

We compared our approach to the Plaid model of [11], that uses a related decomposition, and showed that our approach extracts processes that are more clearly identified with biological functions. In general, we showed that our approach provides a coherent global picture of biological processes.

An important advantage of our approach is that it is part of a general probabilistic framework for biological processes, as described in [14, 13]. Thus, it provides a mechanism by which we can integrate heterogeneous data sources, such as array annotations, clinical outcomes, or promoter region sequences, into a single coherent framework. Thus, for example, following [13], we could try to directly identify processes based not only on the expression data, but also on the existence of shared motifs in the promoter region. We intend to explore this extension and others in future work.

References

[1] Go: Gene ontology. In *http://www.geneontology.org*.

[2] Kegg: Kyoto encyclopedia of genes and genomes. In *http://www.genome.ad.jp/kegg*.

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

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

[5] 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.

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

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

[8] 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.

[9] T. Kailath, A.H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall Information and Systems Sciences Series, 2000.

[10] D. Koller and A. Pfeffer. Probabilistic frame-based systems.

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

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

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

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