BIOINFORMATICS

Vol. 1 no. 1 2001
Pages 1–9

Rich Probabilistic Models for Gene Expression

Eran Segal[1], Ben Taskar[1], Audrey Gasch[2], Nir Friedman[3] and Daphne Koller[1]

[1]Computer Science Department, Stanford University, Stanford, 94305, USA, [2]Department of Genome Sciences, Lawrence Berkeley National Labs, Berkeley, CA, 94702, USA and [3]School of Computer Science & Engineering, Hebrew University, Jerusalem, 94904, Israel

ABSTRACT

Clustering is commonly used for analyzing gene expression data. Despite their successes, clustering methods suffer from a number of limitations. First, these methods reveal similarities that exist over all of the measurements, while obscuring relationships that exist over only a subset of the data. Second, clustering methods cannot readily incorporate additional types of information, such as clinical data or known attributes of genes. To circumvent these shortcomings, we propose the use of a single coherent probabilistic model, that encompasses much of the rich structure in the genomic expression data, while incorporating additional information such as experiment type, putative binding sites, or functional information. We show how this model can be learned from the data, allowing us to discover patterns in the data and dependencies between the gene expression patterns and additional attributes. The learned model reveals *context-specific* relationships, that exist only over a subset of the experiments in the dataset. We demonstrate the power of our approach on synthetic data and on two real-world gene expression data sets for yeast. For example, we demonstrate a novel functionality that falls naturally out of our framework: predicting the "cluster" of the array resulting from a gene mutation based only on the gene's expression pattern in the context of other mutations.

Contact: eran@cs.stanford.edu

INTRODUCTION

A central goal of molecular biology is to understand the regulatory mechanisms that govern protein activity. One of the main mechanisms of regulation controls the rate of mRNA transcription of different genes. DNA microarrays provide a tool for measuring the abundance of thousands of mRNA transcripts simultaneously. This technology facilitates the characterization of every gene's expression in response to many different types of experimental conditions, generating enormous amounts of complex data, e.g., (Spellman et al., 1998; Gasch et al., 2000). A key challenge is the development of methodologies that are both statistically sound and computationally tractable for inferring biological insights from these large datasets.

The most commonly used computational method for analyzing genomic expression data is clustering, a process which identifies clusters of genes and/or array experiments that share similar expression patterns (e.g., (Alon et al., 1999; Ben-Dor et al., 1999; Eisen et al., 1998)). Genes that are similarly expressed are often coregulated and involved in the same cellular processes. Therefore, clustering suggests functional relationships between clustered genes, and helps in identifying promoter sequence elements that are shared among them (Spellman et al., 1998). Clusters of experiments can imply relationships between those experimental conditions, implying similarities in the cellular responses triggered by those conditions (Hughes et al., 2000).

Despite their successes, clustering methods suffer from a number of limitations. First, these methods reveal similarities that exist over all of the measurements, while obscuring relationships that exist over only a subset of the data. Second, although clustering identifies genes that are similar in expression, they cannot readily incorporate additional types of information, such as clinical data or experimental details. (See (Barash and Friedman, 2001; Holmes and Bruno, 2000) for some initial work on this topic.) In this paper, we propose the use of a single coherent probabilistic model, that encompasses much of the rich structure in the genomic expression data, while incorporating additional information to aid in the predictions. We show how this model can be learned from the data, allowing us to discover patterns in the data and to elucidate the interdependencies between the gene expression patterns and additional attributes.

Our approach is based on the language of *probabilistic relational models* (PRMs) (Koller and Pfeffer, 1998; Friedman et al., 1999) that extend Bayesian networks to a *relational* setting, where we have multiple interdependent objects (such as genes and arrays). PRMs overcome many of the limitations of clustering methods. They allow us to include multiple types of information to identify similar objects. For example, identifying similarities between

© Oxford University Press 2000

array experiments can be based on gene expression patterns, experimental or clinical data, the cell type or strain used in the experiment, the cellular phenotype triggered by each condition, and more. When identifying gene relationships, our approach can use gene expression data, sequence elements present in the gene promoters, functional information, and more. By incorporating all the available information into the analysis, more refined gene and experiment classifications can be achieved.

A second advantage to our method is that it presents context-specific relationships between the objects. Many gene relationships exist only over a subset of the experiments in the dataset, while similarities in the array experiments may be different over different subsets of genes. We describe learning procedures (related to that of (Barash and Friedman, 2001)) that are able to determine which attributes are informative in which context. Our procedure identifies *groupings* of measurements that correspond to a subset of both genes and experiments. Thus, unlike standard clustering methods, our approach does not produce indivisible clusters, where all of the objects in a cluster are assumed to behave the same in all contexts.

To validate our method, we present two case studies for the use of PRMs. In the first, we analyze the Yeast Stress data of Gasch et al. (2000), which characterizes the expression patterns of yeast genes under different experimental conditions. Our model identifies groupings based on similarities in gene expression, the presence of known transcription factor (TF) binding sites within the gene promoters, and functional annotation of genes. Our approach identifies expected gene clusters, that display similar gene expression patterns and are known to function in the same metabolic processes. Even more interesting is the discovery of new groupings of genes based both on expression level and on possible TF binding sites.

In the second case study, we use the Yeast Compendium data of Hughes et al. (2000), which observed the genomic expression programs triggered by specific gene mutations. The goal of these experiments is to assign hypothetical functions to uncharacterized genes, by comparing the genomic expression program triggered by their deletion to known expression programs. This data allows us to exhibit a very different capability of our approach. We learn a model based on the genomic expression programs triggered by different gene mutations. We then use our model to predict the cluster that *would be* assigned to a mutation for which we do not have the array data. This task is a novel one, that falls naturally within our framework but not within that of other approaches.

PROBABILISTIC MODELS OF GENE EXPRESSION DATA

Consider a set of measurements for a set Gene of genes across a set Array of microarrays, reporting the measured expression (or its logarithm) $m_{g,a}$ for each gene $g \in \mathsf{Gene}$ and array $a \in \mathsf{Array}$. Regularities in the expression data often correspond to important biological phenomena. *Clustering* methods are one approach for discovering such regularities, providing biological insight by identifying groups of genes and/or arrays that are similar in some sense. A *two-sided* clustering (Lazzeroni and Owen, 1999; Hofmann et al., 1999) partitions the set Gene to *gene clusters* $G_1, \ldots, G_k$, and the set Array to *array clusters* $A_1, \ldots, A_l$. This clustering "models" the data by assuming that all genes in the same cluster behave similarly, and that all arrays in the same cluster behave similarly. More precisely, the model asserts that, for gene $g \in G_i$ and array $a \in A_j$, the expression level $m_{g,a}$ is governed by a distribution specific to the combination of cluster G_i and cluster A_j. For example, this distribution might be a Gaussian with mean $\mu_{i,j}$ and variance $\sigma^2_{i,j}$. This type of clustering provides a very compact summarization of the data in terms of a $k \times l$ matrix of *groupings*, where each grouping contains the measurements corresponding to a cluster of genes and a cluster of arrays. The model explains differences of expression between groupings, and treats differences between the measurements in the same grouping as "noise." A good clustering — one which is predictive — would be one in which the variances $\sigma^2_{i,j}$ are small, implying that most of the differences between expression measurements are explained by the model, and not attributed to noise.

Two-sided clustering is a promising model. However, it is very limited in its ability to take advantage of additional available information. For genes, we might have annotations such as functional role, cellular location or the TF binding sites in a gene's promoter region. For arrays, we might have the treatment applied to the sample, the growth conditions, the strain of yeast used, etc. In the Compendium data set (Hughes et al., 2000), each array corresponds to an experiment with a mutated yeast strain, where one or more genes were knocked out; here, the attributes of the knocked out gene can provide information about the array. These attributes might be very informative about the expression level, and we want to allow models where the expression level depends on their values. However, we do not simply want to define a separate distribution for each combination of gene and array attributes: the number of resulting distributions would be enormous, and we would not have enough data to estimate their parameters. Rather, we want to consider models where only some attributes have a direct influence on the expression levels. Moreover, we want to *discover* which are the significant attributes by learning a predictive model from the data.

Probabilistic Relational Models. *Probabilistic relational models* (PRMs) (Koller and Pfeffer, 1998; Friedman et al., 1999) provide a formal framework for representing the type of dependencies we described above. A PRM provides a probabilistic model over a *relational schema*. A schema specifies the *classes* of objects that appear in our data, and the attributes of each class. In gene expression data, we typically have three classes: Gene, Array, and Exp, which corresponds to the measurement of the expression of a specific gene in a particular array. Each class X is associated with a set of *attributes* $\mathcal{A}(X)$. Attribute A of class X is denoted $X.A$. For example, if we have an annotation of genes according to several functional categories, the class Gene might have several binary attributes such as *AAM*, representing "Amino Acid Metabolism". The Exp class has the attribute *Level* that denotes the measured expression level. In clustering models, we also introduce *latent* (hidden) variables that represent the division into clusters. Thus, when modeling two-sided clustering, the class Gene would also have the attribute *GCluster*, that denotes the cluster the gene belongs to; if we have k gene clusters, the attribute *GCluster* would take on the values $1, \ldots, k$. The class Array has a corresponding attribute *ACluster*.

The schema describes the type of objects we might encounter; the set of actual objects varies from one situation to another. For example, in one case we might have a particular set of $4,000$ genes, 100 arrays, and $400,000$ measurements, in another case, we might have $16,000$ genes, 20 arrays, and $30,000$ measurements (some arrays were partial). In any such particular case, we need to specify the set of objects we deal with. A *skeleton* σ specifies the set of objects. In our example, the skeleton specifies the set of genes $\mathcal{O}^\sigma(\mathsf{Gene})$, the set of arrays $\mathcal{O}^\sigma(\mathsf{Array})$, and the set of measurements $\mathcal{O}^\sigma(\mathsf{Exps})$.

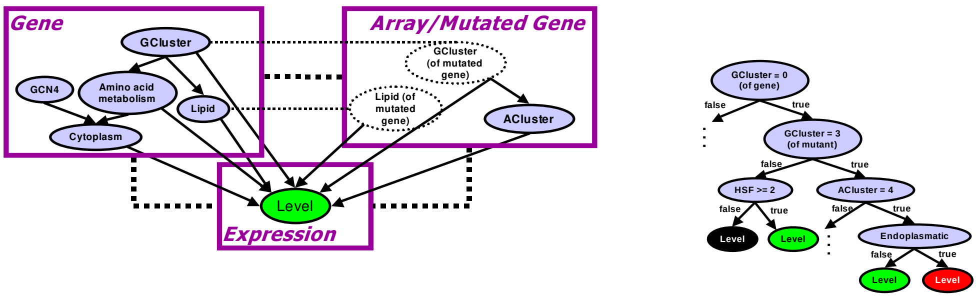
Note that the objects in our domain are related to each other. A particular measurement (e.g., M1237) would correspond to a particular gene that was measured (e.g., G12) and to a particular array (e.g., A37) in which the measurement was performed. We use *reference slots* to refer to related objects. Thus, M1237.*Of-Gene* refers to G12 and M1237.*In-Array* refers to A37. A skeleton has to specify the values of these references for each object. In our example, the skeleton specifies the value of the slots m.*Of-Gene* and m.*In-Array* (i.e., which gene is measured and in which array).

The values of the attributes of the objects are not specified in the skeleton. We treat these unknown values as *random variables*. Formally, a skeleton σ defines a set of (random) variables: one variable $x.A$ for each object x and each attribute A in the object's class. For example, if G12 is an object in $\mathcal{O}^\sigma(\mathsf{Gene})$, then we have a random variable G12.*GCluster* that denotes the cluster of the gene G12. We want to specify a single joint distribution over the values of all of these variables. However, we want this description to apply to any skeleton we might observe. Thus, we specify a "template" probabilistic model over classes of objects, which can then be instantiated for all of the objects in the class. A PRM Π consists of a qualitative dependency structure, $\mathcal{S}$, and the parameters associated with it, $\boldsymbol{\theta}_{\mathcal{S}}$. The dependency structure is defined by associating with each attribute $X.A$ a set of *parents* $\mathrm{Pa}(X.A)$. The parents of $X.A$ specify the attributes that influence it directly, i.e., the attributes whose values determine the distribution from which it is sampled. Each parent has the form of either $X.B$ or $X.R.B$ where R is a reference to a related object. For example, in a simple two-sided clustering model, the attribute Exp.*Level* might have the parents Exp.*Of-Gene*.*GCluster* and Exp.*In-Array*.*ACluster*. This model indicates that the distribution from which the value of m.*Level* is selected is different for different values of g.*GCluster* and a.*ACluster* where g and a are *the particular gene and array that are related to the particular measurement* m.

The parameters of the PRM specify the parameters of each of these distributions. Thus, for each attribute $X.A$, the parameters describe a *conditional probability distribution* (CPD), which specifies the probability of $X.A$, given any possible instantiation of values to its parents. In our simple model above, we would have a distribution over Exp.*Level* for each of the $k \times l$ assignments of values to Exp.*Of-Gene*.*GCluster* and Exp.*In-Array*.*ACluster*. As we discuss below, we have freedom to determine the form of this parameterization.

For any skeleton, a PRM induces a *Bayesian network* over all of the variables defined by a skeleton. The parents of each variable in the network are specified by the PRM dependency structure $\mathcal{S}$ and the skeleton. Each variable is associated with a conditional probability distribution, which is copied from the class-level CPD. Continuing our example, the parents of M1237.*Level* would be G12.*GCluster* and A37.*ACluster*, and its CPD would be a copy of $P(\mathsf{Exp}.\mathit{Level} \mid \mathsf{Exp}.\mathit{Of\text{-}Gene}.\mathit{GCluster}, \mathsf{Exp}.\mathit{In\text{-}Array}.\mathit{ACluster})$. The semantics of this network is defined as usual. Letting $Y_1, \ldots, Y_N$ be the set of variables, the joint distribution is defined as $P(Y_1, \ldots, Y_N) = \prod_{i=1}^N P(Y_i \mid \mathrm{Pa}(Y_i))$.

Context-Specific Models. The language of PRMs allows us to introduce gene and array attributes into the model, thereby allowing us to extend substantially the simple two-sided clustering model discussed above. More specifically, we can model the dependency of Exp.*Level* on the gene and array attributes. At the level of the PRM structure, we can model a dependence of



(a) (b)

Fig. 1. (a) PRM model for Compendium data set; (b) Part of the tree CPD in the model.

the expression level on whether the associated gene has the function Amino Acid Metabolism, by adding Exp.*Of-Gene.AAM* as a parent of Exp.*Level*. In general, we would expect the Exp.*Level* to depend on several of these attributes, e.g., biochemical functions, cellular locations, etc. If we have n of these attributes $A_1, \ldots, A_n$, and each of them can influence the expression level, the resulting model will require that we specify a CPD $P($Exp.*Level* $\mid$ Exp.*Of-Gene.GCluster*, Exp.*In-Array.ACluster*, Exp.*Of-Gene*.$A_1, \ldots,$ Exp.*Of-Gene*.$A_n)$. A naive representation of this CPD requires that we specify $2^n \cdot k \cdot l$ distributions, which is clearly unrealistic even for small values of n. Beyond the computational consequences of this explosion, this naive representation also hides important patterns that might be present in the data. For example, consider again the AAM function. We might expect that genes of this function will behave differently in arrays where this metabolism is very active (e.g., during rapid growth), or depressed (e.g., during cell arrest). In other conditions, this distinction is irrelevant. Thus, although we consider the functional category Exp.*Of-Gene.AAM* as informative about the expression values, it is relevant only when Exp.*In-Array.ACluster* has specific values. In other words, we want the distribution over Exp.*Level* to be different for the different values of Exp.*Of-Gene.AAM* only for certain values of Exp.*In-Array.ACluster*.

A natural representation of this type of interaction is using tree-structured CPDs, similar to decision trees (Friedman and Goldszmidt, 1998). Formally, A *CPD-tree* representation of a CPD for an attribute $X.A$ is a rooted tree; each *node* in the tree is either a *leaf* or an *interior node*. Each interior node is labeled with a test of the form $Y.B = v$, where $Y.B$ is a parent of $X.A$ and v is one of its values. Each of these nodes has two outgoing *arcs* to its children, corresponding to the outcomes of the test (true or false). For example, we might represent the CPD of Exp.*Level* using the tree shown in Figure 1(b).

Each leaf node corresponds to a unique path from the root. The nodes on the path correspond to tests, and the arcs to their outcomes. This sequence thus defines the event *induced* by the leaf — the conjunction (i.e., intersection) of the events defined by the arcs in the sequence. For example. the left-most leaf of Figure 1(b) corresponds to the event "m.*Of-Gene.GCluster* $= 0$ and m.*Of-Array.Mutant.GCluster* $\neq 3$ and m.*Of-Gene.HSF* $<$ 2". We denote by *Leaves*$(X.A)$ the set of leaves in the CPD-tree for $X.A$. If ℓ is the index of a leaf, we use the notation $L_{X.A} = \ell$ as a shorthand for the event that correspond to the leaf ℓ. Each leaf is labeled with a distribution over the values of $X.A$, representing part of its CPD — the distribution $P(X.A \mid L_{X.A} = \ell)$.

Each leaf in the CPD-tree of Exp.*Level* corresponds to a *grouping* of expression measurements that are considered to be sampled from the same distribution. Note that each such grouping is a "rectangle" in the expression matrix: a cross-product of a set of genes and a set of arrays. However, unlike the groupings defined in two-sided clustering, these groupings do not typically define a uniform grid over the expression matrix.

LEARNING THE MODELS

Our goal is to learn a PRM model from data. The input to the learning algorithm is a skeleton σ, and a (potentially partial) assignment of values to the random variables it defines. In our example, the data set will consist of: a set of expression level measurements, corresponding to some set of genes and some set of arrays, and typically a set of attributes for the genes and for the arrays. Note that the cluster variables for genes and arrays are not part of the data. The learning task can be decomposed into two parts: *parameter estimation* — estimating the parameters for a model whose structure is given, and *model selection* — choosing among the set of possible structures.

Parameter Estimation. Consider the task of estimating parameters for a model where we have fixed the dependency structure $\mathcal{S}$ that specifies the parents of each attribute, and the tree structure for each CPD. Our goal is to estimate the model parameters $\boldsymbol{\theta}_{\mathcal{S}}$: the distribution at each leaf. Most simply, we can estimate parameters by using *maximum likelihood* estimate. We define the *likelihood* of a particular set of parameters $\boldsymbol{\theta}_{\mathcal{S}}$ as the probability of the training data $\mathcal{I}$ given the model $(\mathcal{S}, \boldsymbol{\theta}_{\mathcal{S}})$. This probability is defined according to the PRM semantics, as the probability of the attribute values in $\mathcal{I}$ in the Bayesian network defined by its skeleton. The maximum likelihood parameters are the $\boldsymbol{\theta}_{\mathcal{S}}$ that maximize the likelihood.†

When the values of all attributes are fully observed, the maximum likelihood parameter estimation reduces a maximum likelihood estimation of each the separate $P(X.A \mid L_{X.A} = \ell)$ at the leaves of the different CPD-trees. The nature of this estimation task depends on the type of attribute. If the attribute is discrete valued, we estimate a multinomial distribution. If it is a continuous valued attribute, we estimate a Gaussian distribution. Both estimation tasks are standard and rely on *sufficient statistics* that summarize the data. For example, in the case of multinomial distributions, these are just the counts $C_{X.A}[v, \ell]$, specifying the number of objects $x \in \mathcal{O}^{\mathcal{I}}(X)$ for which we observe the combination $x.A = v$ and $L_{x.A} = \ell$. In the case of Gaussian distributions, these sufficient statistics are the mean and variance of the objects in which the leaf l is relevant.

Structure Learning. We now consider the task of selecting among the many possible models, where each of the possible models specifies the set of parents for each attribute, and the structure of the CPD-trees. There are two issues that need to be addressed in this setting: the *scoring function*, used to evaluate the "goodness" of different candidate structures relative to the data, and the *search algorithm* for finding a structure with a high score. We discuss each of these in turn.

We follow Friedman et al. (1999) and use Bayesian *model selection* methods to score candidate structures. The *Bayesian score* of a structure $\mathcal{S}$ is defined as the *posterior* probability of the structure given the data $\mathcal{I}$ — $P(\mathcal{S} \mid \mathcal{I}, \sigma)$. Using Bayes rule, and making a standard assumption that the different structures are equally likely *a priori*, the score reduces to $P(\mathcal{I} \mid \mathcal{S}, \sigma)$. This term evaluates the fit of the model to the data by averaging the likelihood of the data over all possible parameterizations of the model. This averaging regularizes the score and avoids overfitting the data with complex models. When the training data is fully observed, the Bayesian score has a simple analytic form (Friedman et al., 1999; Friedman and Goldszmidt, 1998; Heckerman, 1998), as a function of the sufficient statistics of that model.

Having defined a metric for evaluating different models, we need to search the space of possible models for one that has high score. As is standard in both Bayesian network and PRM learning (Heckerman, 1998; Friedman et al., 1999), we use a greedy local search procedure that maintains a "current" candidate structure and iteratively modifies it to increase the score. At each iteration, we consider a set of simple local transformations to the current structure, score all of them, and pick the one with highest score. Our operators, following Chickering et al. (1997), consider only transformations to the CPD-trees. The tree structure induces the dependency structure, as the parents of $X.A$ are simply those attributes that appear in its CPD-tree. The two operators we use are: *split* — replaces a leaf in a CPT tree by an internal node with two leafs; and *trim* — replaces the subtree at an internal node by a single leaf. To avoid local maxima associated with the greedy search procedure, we use a variant of simulated annealing: Rather than always taking the highest scoring move in each search step, we take a random step with some probability, which decays exponentially as the search progresses.

Incomplete Data. So far, we have assumed that the training data $\mathcal{I}$ specifies the values of all the attributes. In many situations, this assumption is not warranted; in particular, it is clearly false when we are learning models with *latent* variables, such as Gene.*GCluster*, that are never observed in the training data. Learning from partially observed data is substantially more difficult than the fully observable case: the likelihood function has multiple local maxima, and no general method exists for finding the global maximum.

The *Expectation Maximization (EM)* algorithm is an approach for parameter estimation with incomplete data. It is guaranteed to find a local maximum of the likelihood function. The EM algorithm is an iterative method. Starting from an initial guess for the parameters, it repeatedly performs two steps. In the E-step, it computes the distribution over the unobserved variables given the observed data and the current estimate of the parameters. It uses this distribution to "fill in" each missing attribute $x.a$ with a soft completion that takes into consideration how likely its different values are. In clustering models, this completion corresponds to a soft assignment of objects to clusters. In the M-step, it uses this completion as if it were real, and reestimates the parameters using the standard maximum likelihood estimation procedure. The process then repeats, using the new parameters, until convergence.

To fill in the missing data in the E-step, we need to run inference over the entire Bayesian network induced by the PRM over the objects in σ. In many cases, these networks are complex, and exact inference is intractable. Instead, we use *belief propagation* (Murphy and Weiss, 1999), an approximate inference algorithm which has recently been shown to be effective on a wide range of models.

† In practice, the maximum likelihood can be noisy in leaves that correspond to rare events. To reduce parameter variance, we use a Bayesian method to *smooth* the estimate.

Table 1. Reconstruction results for synthetic data

	% parents recovered	Cluster recovery Naive Bayes	PRMs
Simulated Data	84.5 ± 2.5	90.8 ± 0.42	98.4 ± 1.07
Noisy Simulated Data	56 ± 2.5	76.7 ± 1.42	88.1 ± 1.52

For learning structure with incomplete data, we use a hard-assignment variant of *structural EM* (Friedman, 1998). We fill in missing attributes with their most probable value, given the current model, and then run structure learning on the completed data. When structure learning converges, we remove the hypothesized values for the unobserved attributes, run EM to fit the parameters of the learned structure, and then select a new hard assignment for the missing attributes. This process is iterated until convergence.

CASE STUDIES

We evaluated our methods on three gene expression data sets, one synthetic and two real. The results on synthetic data demonstrate that our approach recovers structure that we know to be present in the data. The models for the real data sets illustrate the wide applicability of our approach.

Synthetic data. We generated a synthetic data set by sampling from a PRM model. To make the data realistic, we used PRM models learned from the Stress data set. These models are similar to the two-sided clustering models described above. The main difference is that we take Array.*ACluster* to be the observed experiment type (1 of 12). The Gene.*GCluster* attribute is hidden, and takes 9 values. We generated data for 1000 (imaginary) genes and 90 arrays, for a total of 90,000 measurements. Each gene was augmented with 15 function annotations and 30 TFs.

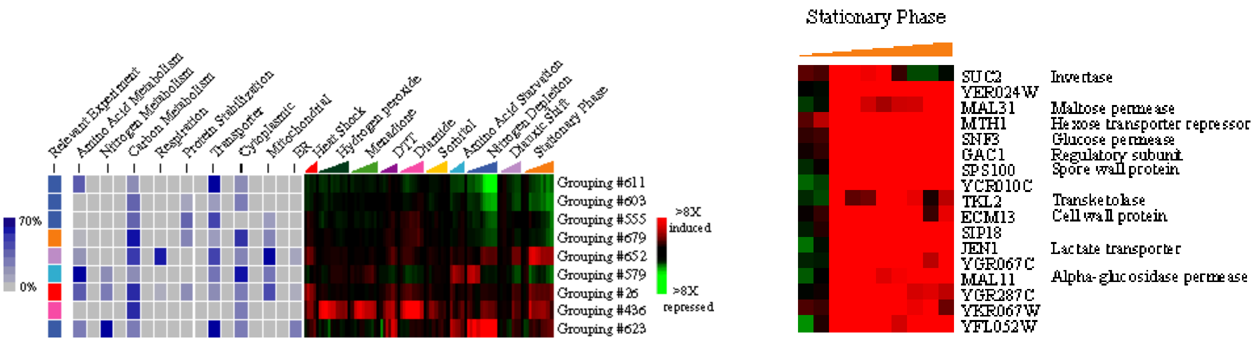
We evaluated the ability of our learning algorithm to recover the model using two metrics. To robustly estimate these, each was evaluated using 10-fold cross validation, training on 90% of the data and (where applicable) testing on the remaining 10%. The results are shown in Table 1. We first measured the extent to which the structure learned is similar to the "true" structure in the data. More specifically, we saw how many of the parents of Exp.*Level* are recovered in the learned model. Our results indicate that our algorithm recovers the "true" structure with very high accuracy. In a second test, we measured the extent to which we can recover the original gene clusters $g.GCluster$, which were hidden in the data. We learned the model on the training data, and then tried to predict the (nine-valued) cluster attribute in the test data. Our reconstruction ability for the clusters is extremely high, and much higher than we could obtain by a standard clustering algorithm using a Naive Bayes model over gene expression alone.

To test the robustness of our methods, we also generated a noisy version of the same data set: within each category of data — function annotations, TFs, and expression levels — 20% of the entries were permuted among themselves. We can see that our ability to reconstruct the structure is lower, but still quite good given the number of possible parents. Our ability to reconstruct the clusters is still impressively high, whereas the simple naive Bayes clustering degraded more substantially. Thus our method is robust even to a large amount of noise in the data.

Yeast Stress data. We now consider the data set of Gasch et al. (2000), who characterized the genomic expression patterns of yeast genes in 12 different experimental conditions. We selected 954 genes that had significant changes in gene expression (eliminating the ESR genes for which clustering is trivial), and the full set of 92 arrays. We supplemented the raw gene expression data with additional attributes from two other yeast databases. For every gene, we selected 22 functional classes from the MIPS database (Mewes et al., 1999), and used them as binary attributes of genes. In addition, we introduced attributes representing the presence of binding sites for known TFs. We introduced one attribute for each of 44 TFs, and generated its value for each gene — $0, 1$, or ≥ 2 — by scanning the 1000bp upstream of the gene's ORF using the MatInspector program (Quandt et al., 1995) and counting the number of putative sites for the TF.

We used the model discussed above, with the classes Gene, Array, and Exp. The Gene class included a latent cluster variable, as well as the 66 attributes described above. The Array class included an attribute *Type* with 12 values, representing the "type" of experiment performed. We used this attribute as an observed substitute for the *ACluster* attribute.

Our algorithm learned many dependencies between the expression measurements, the type of the experiment, the latent cluster variable, the function attributes, and the TFs. Before analyzing the model, the first question of interest is whether the structure learned is indeed present in the data or perhaps our algorithms would learn dependencies even when no structure is present. To test that, we took the real data set and permuted all of it: annotations with annotations, TFs with TFs, and expression levels with expression levels (even across experiment types). We then tested three models: model 1 — a PRM trained on the original data set; model 2 — a PRM trained on the noisy data; model 3 — a PRM with no dependencies trained on the noisy data. We then evaluated the ability of the model to generalize from the training data by evaluating the log-likelihood of test data. Over 10-fold cross validation, we obtained substantial differences between the models:



(a) (b)

Fig. 2. (a) Summary of representative gene groupings in the Stress data. Each grouping corresponds to a cluster of genes in the context of a particular experiment. The right panel shows the average expression profile of the genes in the grouping in the context of all of the experiments; the experiment type is indicated by the colored triangles at the top of the figure. The particular experiment type in which the grouping arises is shown on the left. The left panel shows the functional attributes associated with each of the displayed groupings. Each box indicates the percentage of each grouping that displayed that attribute. (See http://cs.stanford.edu/∼erans/ismb01/ for additional cluster data.) (b) The expression of genes in Grouping 666 in response to stationary phase. All genes in this cluster contained two or more potential Mig1p binding sites within their promoters.

-11729 ± 272 for model 1, -14680 ± 721 for model 2, and -14923 ± 160 for model 3, indicating that our model indeed explains the data significantly better. We also examined the extent to which the dependencies added in the learning algorithm are informative, in that they cause a substantial improvement to the Bayesian score. Indeed, the learning algorithm discovered 7 annotation and 15 TF parents whose score in the model learned from real data was around twice as high as the best score of the dependencies learned from the perturbed data. As our models are much better at explaining the data, this strongly indicates that these parents correspond to dependencies that are indeed implied by the data.

Our second experiment tests whether our learning algorithm results in coherent clusters. To test that, we computed a weighted average of the variances in each of the groupings. Over the three structure-modification iterations of our algorithm, the average grouping variance decreased substantially, from 0.692 in the initial model to 0.614 in the final model. We also experimented with a novel approach to incorporating the functional annotations recovered from yeast databases: to avoid restricting the aggregation based on previous interpretation of experimental data, we use the functional annotations as a guide in the initial training of the model; we then remove the observed values of these annotations and retrain the model based only on the gene expression and TF binding site data. This process allows unannotated genes to be aggregated with characterized genes, so that we can infer hypothetical functions for those uncharacterized genes. Overall, around 20% of the functional annotations were changed in this process, mostly going from cases where the function was labeled as absent to cases where it was labeled as present. This change is quite reasonable, as the MIPS database does not distinguish between “unknown” and “known to be false”. These changes also led to a substantial improvement in the average grouping variance: from 0.692 to 0.565. Although it is not clear whether the new annotations correspond to the original meaning of the functions, it appears that they do represent a biologically predictive property.

Figure 2(a) shows a summary diagram of a representative set of groupings constructed by our model. For example, Grouping 652 consists of 73 genes that are similarly induced during the diauxic shift. A significant percentage of genes in this grouping are annotated as functioning in respiration or transport and localizing to the mitochondria, cytoplasm, and endoplasmic reticulum (ER). Inspection of the genes in this grouping confirms that many of these genes are involved in the TCA metabolic cycle, oxidative phosphorylation, and ATP synthesis (respiration), transport of sugars and amino acids, and other related functions. Thus, the attributes associated with this grouping paint a picture of the physiological response during stationary phase: when the glucose in the cells medium becomes limiting, transporters are secreted through the ER to the plasma membrane, where they import sugars and amino acids to supply the TCA cycle, which promotes respiration in the mitochondria. The algorithm also assigns 15 uncharacterized genes to this grouping, suggesting that these genes are likely to play a similar role in the cell.

The algorithm also identified groupings of genes that were related by the presence of known transcription factor binding sites in the their promoters. Most interesting is Grouping 666 identified in iteration 1, shown in Figure 2(b). This grouping is over a set of 17 genes involved in sugar metabolism that each contain two or more binding sites for the Mig1 repressor. Mig1p represses genes involved in alternative sugar metabolism when external glu-

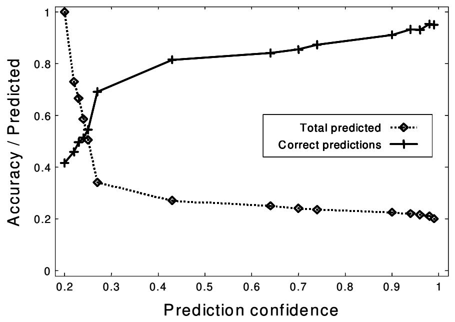


Fig. 3. Predicting the array (mutation) cluster without observing its expression data in the Compendium data.

cose levels are high, but the repressor becomes deactivated when glucose becomes limiting during the diauxic shift, leading to the increased expression of its targets. All of the genes in the grouping are substantially induced at the diauxic shift. Included in this grouping is the SUC2 gene, a well known target of Mig1p, as well as genes involved in glucose and maltose metabolism (e.g., MAL31), cell wall proteins (e.g., ECM13), and a number of genes involved in other aspects of carbon metabolism. These proteins were not previously known to be regulated by Mig1p, however the presence of the Mig1p binding site in their promoters, along with the similarities in their biochemical functions and gene expression patterns, suggests that they are also regulated by Mig1p derepression. We note that the context-sensitive nature of our groupings played an important role in identifying this cluster. Many of the genes in this grouping were also present in the much larger Grouping 652, which represented genes that were related in gene expression and functional annotation but not necessarily sharing the Mig1p promoter element. A traditional clustering algorithm that does not allow genes to participate in multiple groupings may not have been able to isolate these two clusters, and would not have revealed this new cluster of Mig1p-regulated genes.

Yeast Compendium Data. The Compendium data set (Hughes et al., 2000) is very different in nature than the Stress data. The goal of the experiments was to assign hypothetical functions to uncharacterized genes, by comparing the expression pattern triggered by deletion of these characterized genes. We selected 528 genes and 207 arrays, focusing on genes and mutations that had some functional annotations in the MIPS database.

Here we can exploit much more of the expressive power of PRMs. In this model, the Gene class has the same set of attributes as in the Stress data set above. The Array class has an attribute *ACluster*, representing a cluster of the array (mutation). Most interestingly, we introduced a reference slot — Array.*Mutation* (indicated in Figure 1(a) by the thick dashed line connecting the gene object to the array object) — which refers to the object for the mutated gene used to generate the array.

The explicit relationship between the array object and the associated mutated gene, and the dependencies that it permits, allow us to perform a task which is outside the scope of other approaches: predicting the array (mutation) cluster of an array without performing the experiment! The basic insight is that mutations that cluster together tend to induce similar effects on the genomic expression pattern when they are mutated because they are involved in similar functional processes. This insight suggests the following type of inference: For a given gene, we can infer the gene cluster to which it belongs, and then predict which mutation cluster if would fall into if it were to be mutated, based on teh observed correlations between the gene clusters and mutations clusters. We tried out this hypothesis by hiding 20 of the mutant arrays in the data, and training the model on the remaining ones. We then tried to predict the mutation cluster of the 20 hidden arrays, based only on our knowledge about the gene that was mutated. We compared this to the cluster we would have placed the array in after seeing its expression pattern. We repeated this experiment ten times, for different choices of the 20 held-out arrays. A graph of the results is displayed in Figure 3. For each prediction, the algorithm outputs a confidence measure — the probability that the unobserved array is assigned to the most probable cluster. For each such confidence level, we graph the percent of the arrays at that confidence level (or higher), and the accuracy of the prediction if we consider only those arrays at this confidence level. We can see that approximately 22% of the arrays (or 44 arrays) are predicted with 95% accuracy. Thus, there is a significant number of genes for which we can predict, with high accuracy, the mutation cluster to which they belong, without conducting the experiment of mutating them. This allows us to predict hypothetical functional information for these genes. Moreover, our approach tells us which are the arrays for which we can make a high-confidence prediction. We note that the relational nature of our approach is critical to our ability to perform this prediction; a model where we disallowed the direct dependency of the array cluster on the corresponding gene cluster did not exhibit significant predictive power.

DISCUSSION

We have provided a method for analyzing gene expression data based on probabilistic graphical models. Our models are very richly structured, allowing us to integrate multiple types of data. In a sense, they provide a midpoint on the spectrum between two extremes: fine-grained Bayesian network models of gene expression pathways (Friedman et al., 2000; Hartemink et al., 2001; Pe'er et al., 2001), and the more standard coarse-grained clustering approaches.

Unlike standard clustering, our approach can identify genes that are similar over multiple types of data, including functional attributes and transcription factors, providing more refined groupings than those derived from gene expression alone. However, our algorithm is flexible in its use of functional annotations, allowing the functional annotations to be modified to better predict the data. This flexibility allows uncharacterized genes that lack annotations to be associated with genes of known function, thereby suggesting details about their biochemical function and cellular role. Finally, as our algorithm presents groupings in terms that directly relate to function attributes, it provides a summary of the physiological response of the cell, and suggests how genes of different biochemical function or localization can act together to serve the same cellular role. Unlike traditional clustering methods, our approach generates context-specific groupings, in which genes can be present in more than one grouping, thereby revealing multiple gene relationships. As we have seen, this capability can identify groupings among genes that play multiple roles.

The expressive power of our framework opens the door to many exciting directions. For example, we can include potential promoter sequences as objects, and not merely as fully observed attributes. This will allow us not only to identify genes that share a given promoter sequence, but also perhaps to identify new regulatory sequences. Our approach also allows us to incorporate very rich data into the model, including phenotypical information (e.g., about the clinical attributes of patients), tissue type, and more. We plan to explore the capabilities of our framework on richer data sets involving this type of information, with the goal of automatically correlating phenotype data, sequence data, and gene expression data.

Acknowledgements

We thank D. Pe'er for useful discussions. E. Segal was supported by a Stanford Graduate Fellowship (SGF). A. Gasch works at M. Eisen's lab and was supported by training grant HL07279-23. N. Friedman was supported by Israel Science Foundation grant 244/99, Israeli Ministry of Science grant 2008-1-99, and an Alon Fellowship. D. Koller was supported by NSF grant ACI-0082554.

REFERENCES

Alon, U., N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS 96*(12), 6745–50.

Barash, Y. and N. Friedman (2001). Context-specific Bayesian clustering for gene expression data. In *RECOMB'01*.

Ben-Dor, A., R. Shamir, and Z. Yakhini (1999). Clustering gene expression patterns. *J. Comp. Bio. 6*(3-4), 281–97.

Chickering, D. M., D. Heckerman, and C. Meek (1997). A Bayesian approach to learning Bayesian networks with local structure. In *UAI'97*, pp. 80–89.

Eisen, M., P. Spellman, P. Brown, and D. Botstein (1998). Cluster analysis and display of genome-wide expression patterns. *PNAS 95*, 14863–14868.

Friedman, N. (1998). The Bayesian structural EM algorithm. In *UAI'98*.

Friedman, N., L. Getoor, D. Koller, and A. Pfeffer (1999). Learning probabilistic relational models. In *IJCAI'99*.

Friedman, N. and M. Goldszmidt (1998). Learning Bayesian networks with local structure. In *Learning in Graphical Models*, pp. 421–460. Kluwer.

Friedman, N., M. Linial, I. Nachman, and D. Pe'er (2000). Using Bayesian networks to analyze expression data. *J. Comp. Bio. 7*, 601–620.

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

Hartemink, A., D. Gifford, T. S. Jaakkola, and R. Young (2001). Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *Pac. Symp. Biocomp. 6*.

Heckerman, D. (1998). A tutorial on learning with Bayesian networks. In *Learning in Graphical Models*. Kluwer.

Hofmann, T., J. Puzicha, and M. Jordan (1999). Learning from dyadic data. In *NIPS'99*.

Holmes, I. and W. Bruno (2000). Finding regulatory elements using joint likelihoods for sequence and expression profile data. In *ISMB'00*.

Hughes, T. R., 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 (2000). Functional discovery via a compendium of expression profiles. *Cell 102*(1), 109–26.

Koller, D. and A. Pfeffer (1998). Probabilistic frame-based systems. In *AAAI'98*, pp. 580–587.

Lazzeroni, L. and A. Owen (1999). Plaid models for gene expression data. Tech. rep., Stanford.

Mewes, H., K. Heumann, A. Kaps, K. Mayer, F. Pfeiffer, S. Stocker, and D. Frishman (1999). MIPS: a database for protein sequences and complete genomes. *Nuc. Acids Res. 27*, 44:48.

Murphy, K. and Y. Weiss (1999). Loopy belief propagation for approximate inference: An empirical study. In *UAI'99*.

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

Quandt, K., K. Frech, H. Karas, E. Wingender, and T. Werner (1995). MatInd and MatInspector—new fast and versatile tools for detection of consensus matches in nucleotide sequence data. *Nuc. Acids Res. 23*, 4878–4884.

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