Context-Specific Bayesian Clustering for Gene Expression Data

Yoseph Barash
School of Computer Science & Engineering
Hebrew University, Jerusalem, 91904, Israel
hoan@cs.huji.ac.il

Nir Friedman
School of Computer Science & Engineering
Hebrew University, Jerusalem, 91904, Israel
nir@cs.huji.ac.il

ABSTRACT

The recent growth in genomic data and measurement of genome-wide expression patterns allows to examine gene regulation by transcription factors using computational tools. In this work, we present a class of mathematical models that help in understanding the connections between transcription factors and functional classes of genes based on genetic and genomic data. These models represent the joint distribution of transcription factor binding sites and of expression levels of a gene in a single model. Learning a combined probability model of binding sites and expression patterns enables us to improve the clustering of the genes based on the discovery of putative binding sites and to detect which binding sites and experiments best characterize a cluster. To learn such models from data, we introduce a new search method that rapidly learns a model according to a Bayesian score. We evaluate our method on synthetic data as well as on real data and analyze the biological insights it provides.

1. INTRODUCTION

A central goal of molecular biology is to understand the regulation of protein synthesis. With the advent of genome sequencing projects, we have access to DNA sequences of the *promoter* regions that contain the binding sites of *transcription factors* that regulate gene expression. In addition, the development of DNA microarrays allows researchers to measure the abundance of thousands of mRNA targets simultaneously providing a "genomic" viewpoint on gene expression. As a consequence, this technology facilitates new experimental approaches for understanding gene expression and regulation [20, 31].

The combination of these two important data sources leads to better understanding of gene regulation [1, 3]. The main biological hypothesis underlying most of these analyses is "Genes with a common functional role have similar expression patterns across different experiments. This similarity of expression patterns is due to co-regulation of genes in the same functional group by specific transcription factors." Clearly, this assumption is only a first-order approximation of biological reality. There are gene functions for which this assumption definitely does not hold, and there are co-expressed genes that are not co-regulated. Nonetheless, this assumption is useful in finding the "strong" signals in the data.

Based on this assumption, one can *cluster* genes by their expression levels, and then search for short DNA strings that appear in significant over-abundance in the promoter regions of these genes [27, 32, 33]. Such an approach can discover new binding sites in promoter regions.

Our aim here is complimentary to this approach. Instead of discovering new binding sites, our focus is on characterizing groups of genes based on their expression levels in different experiments and the presence of putative binding sites within their promoter regions. The basic biological hypothesis suggests that genes within a functional group will be similar with respect to both types of attributes. We treat expression level measurements and information on promoter binding sites in a symmetric fashion, and cluster genes based on both types of data. In doing so, our method characterizes the attributes that distinguish each cluster.

More specifically, we develop a class of clustering models that cluster genes based on random variables of two types. Random variables of the first type describe the expression level of the gene (or more precisely its mRNA transcript) in an experiment (microarray hybridization). Each experiment is denoted by a different random variable whose value is the expression level of the gene in that particular experiment. Random variables of the second type describe occurrences of putative binding sites in the promoter region of the genes. Again, each binding site is denoted by a random variable, whose value is the number of times the binding site was detected in the gene's promoter region.

Our method clusters genes with similar expression patterns and promoter regions . In addition, the learned model provides insight on the regulation of genes within each cluster. The key features of our approach are: (1) automatic detection of the number of clusters; (2) automatic detection of random variables that are irrelevant to the clustering; (3) robust clustering in the presence of many such random variables, (4) context-depended representation that describes which clusters each attribute depends on. This allows us to discover the attributes (random variables) that characterize each cluster and distinguish it from the rest. We learn these cluster models using a Bayesian approach that uses *structural EM* [10, 11], an efficient search method over different models. We evaluate the resulting method on synthetic data, and apply it to real-life data.

In Section 2 we introduce the class of probabilistic models that we call *Context-Specific Clustering* models. In Section 3 we discuss how to *score* such models based on data. In Section 4 we describe our approach for finding a high-scoring clustering model. In Section 5 we evaluate the learning procedure on synthetic and real-life data. We conclude in a discussion of related work and possible extensions in Section 6.

2. CONTEXT-SPECIFIC CLUSTERING

2.1 Naive Bayesian Clustering

Let $X_1, \ldots, X_N$ be random variables. In our main application, these random variables denote the attribute of a particular gene: the expression level of this gene in each of the experiments, and the numbers of occurrences of each binding sites in the promoter region. Suppose that we receive a dataset D that consists of M joint instances of the random variables. The m'th instance is a joint assignment $x_1[m], \ldots, x_N[m]$ to $X_1, \ldots, X_N$. In our application, instances correspond to genes: each gene is described by the values of the random variables.

In modeling such data we assume that there is an underlying joint distribution $P(X_1, \ldots, X_N)$ from which the training instances were sampled. The *estimation* task is to approximate this joint distribution based on the data set D. Such an estimate can help us understand the interactions between the variables. A typical approach to estimating such a joint distribution is to define a *probabilistic model* that defines a set of distributions that can be described in a *parametric* form, and then find the particular parameters for the model that "best fit" the data in some sense.

A simple model that is often used in data analysis is the *naive Bayes* model. In this model we assume that there is an unobserved random variable C that takes values $1, \ldots, K$, and describes which "cluster" the example belongs to. We then assume that if we know the value of C, all the observed variables become independent of each another. That is, the form of the distribution is:

$$P(X_1, \ldots, X_N) = \sum_k P(C = k) P(X_1 \mid C = k) \cdots P(X_N \mid C = k) \quad (1)$$

In other words, we estimate a *mixture* of product distributions.

This form specifies the global *structure* of the distribution. In addition, we also have to specify how to represent the conditional distributions. For this purpose we use *parametric* families. There are several of families of conditional distributions we can use for modeling $P(X_i \mid C = k)$. In this paper, we focus on two such families. If X_i is a discrete variable that takes a finite number of values (e.g., a variable that denotes number of binding sites in a promoter region), we represent the conditional probability as a *multinomial* distribution $P(X_i \mid C = k) \sim \text{Multinomial}(\{\theta_{x_i|k} : x_i \in \text{Val}(X_i)\}$. That is, for each value x_i of X_i we have a parameter $\theta_{x_i|k}$ that denotes the probability that $X_i = x_i$ when $C = k$. The parameters must be non-negative, and satisfy $\sum_{x_i} \theta_{x_i|k} = 1$, for each k. If X_i is a continuous variable (e.g., a variable that denotes the expression level of a gene in a particular experiment), we use *Gaussian* distribution $P(X_i \mid C = k) \sim N(\mu_{X_i|k}, \sigma^2_{X_i|k})$.

The naive Bayes model is attractive for several reasons. First, from estimation point of view we need to estimate relatively few parameters: the mixture coefficients $P(C = k)$, and the parameters of the conditional distributions $P(X_i \mid C = k)$. Second, the estimated model can be interpreted as modeling the data by K clusters (one for each value $k = 1, \ldots, K$), such that the distribution of different variables within each cluster are independent. Thus, dependencies between the observed variables are represented by the cluster variable. Finally, this model allows for fairly efficient learning algorithms (such as *expectation maximization* (EM) [9]).

Once we have estimated the conditional probabilities, we can compute the probability of an example belonging to a cluster:

$$P(C = k \mid x_1, \ldots, x_N) \propto P(C = k) P(x_1 \mid C = k) \cdots P(x_N \mid C = k)$$

If the clusters are well-separated, then this conditional probability will assign each example to one cluster with high probability. However, it is possible that clusters overlap, and some examples are assigned to several clusters. If we compare the probability of two clusters, then

$$\log \frac{P(C = k \mid x_1, \ldots, x_N)}{P(C = k' \mid x_1, \ldots, x_N)} = \log \frac{P(C = k)}{P(C = k')} + \sum_i \log \frac{P(x_i \mid C = k)}{P(x_i \mid C = k')} \quad (2)$$

Thus, we can see the "decision" between any two clusters as the sum of terms that represent the contribution of each attribute to this decision. The ratio $P(x_i \mid C = k)/P(x_i \mid C = k')$ is the relative *support* that x_i gives to k versus k'.

2.2 Selective Naive Bayesian Models

The naive Bayes model gives all variables equal status. This is a potential source of problems for two reasons. First, some variables can be "noise" and have no real interactions with the other variables. Suppose that X_1 is independent from rest of the variables. By learning K conditional probability models $P(X_1 \mid C = 1), \ldots, P(X_1 \mid C = K)$, we are increasing the variability of the estimated model. Second, since we are dealing with relatively small number of training examples, if we fail to recognize that X_1 is independent of the rest, the observations of X_1 can bias our choice of clusters.

If we know that X_1 is independent from the rest, we can use the fact that $P(X_1 \mid C) = P(X_1)$ and rewrite the model in a simpler form: $P(X_1) \sum_k P(C = k) P(X_2 \mid C = k) \cdots P(X_N \mid C = k)$ This form requires less parameters and thus the estimation of these parameters is more robust. More importantly, the structure of this model explicitly captures the fact that X_1 is independent of the other variables—its distribution does not depend on the cluster variable. Note that in this model, as expected, the value of X_1 does not impact the probability of the class C.

In our biological domain, we expect to see many variables that are independent (or almost independent) of the classification. For example, binding sites of transcription factors that do not play an active role in the conditions in which expression levels were measured. Another example is a putative binding site that does not correspond to a real biologically meaningful function. Thus, learning that these sites are independent of the measured expression levels is an important aspect of the data analysis process.

Based on this discussion, we want to consider models where several of the variables do not depend on the hidden class. Formally, we can describe these dependencies by specifying a set $G \subseteq \{1, \ldots, N\}$ that represents the set of variables that depend on the cluster variable C. The form of the joint distribution is

$$P(X_1, \ldots, X_N \mid G) = \left(\prod_{i \notin G} P(X_i) \right) \sum_k \left(P(C = k) \prod_{i \in G} P(X_i \mid C = k) \right)$$

We note that this class of models is essentially a special subclass of *Bayesian networks* [24]. (Similar models were considered for somewhat different application in supervised learning by Langley and Sage [21].)

We note again, that when we compare the posterior probability of two clusters, as in (2), we only need to consider variables that are not independent of C. That is,

$$\log \frac{P(C=k \mid x_1,\ldots,x_N)}{P(C=k' \mid x_1,\ldots,x_N)} = \log \frac{P(C=k)}{P(C=k')} + \sum_{i\in G} \log \frac{P(x_i \mid C=k)}{P(x_i \mid C=k')}.$$

2.3 Context-Specific Independence

Suppose that a certain binding site, whose presence is denoted by the variable X_1, is regulating genes in two functional categories. We would then expect this site to be present with have high probability in promoter regions of genes in these two categories, and to have low probability of appearing in the promoter region of all other genes. Since X_1 is relevant to the expression level of (some) genes, it is not independent of the other variables, and so we would prefer models where $1 \in G$. In such a model, we need to specify $P(X_1 \mid C = k)$ for $K = 1,\ldots,K$. That is, for each functional category, we learn a different probability distribution over X_1. However, since X_1 is relevant only to classes 1 and 2, this introduces unnecessary distinctions: once we know that C is not one of the two "relevant" function classes (i.e., $C > 2$), we can predict $P(X_1 \mid C)$ using a single distribution.

To capture such distinctions, we need to introduce a language that refine the ideas of selective naive Bayesian models. More precisely, we want to describe additional structure within the conditional distribution $P(X_1 \mid C)$. The intuition here is that we need to specify *context-specific independencies* (CSI): once, we know that $C \not\in \{1,2\}$, then X_1 is independent of C. This issue has received much attention in the probabilistic reasoning community [2, 6, 14].

Here, we choose a fairly simple representation of CSI that is termed *default tables* in [14]. This representation is as follows. The structure of the distribution $P(X_i \mid C)$ is represented by an object $\mathcal{L}_i = \{k_1,\ldots,k_l\}$ where $k_i \in \{1,\ldots,K\}$. Each k_i represents a case that has an explicit conditional probability. All other cases are treated by a special *default* conditional probability. Formally, the conditional probability has the form:

$$P(X_i \mid C=k) = \begin{cases} P(X_i \mid C=k_i) & k = k_i \\ P(X_i \mid C \not\in \{k_1,\ldots,k_l\}) & \text{otherwise} \end{cases}$$

It will be convenient for us to think of $\mathcal{L}_i$ as defining a random variable, which we will denote L_i, with $l+1$ values. This random variable is the characteristic function of C, such that $L_i = i$ if $C = k_i$, and $L_i = l+1$ if $C \not\in \{k_1,\ldots,k_l\}$. Then, $P(X_i \mid C)$ is replaced by $P(X_i \mid L_i)$. This representation requires $l+1$ different distributions rather than K different ones. Note that each of these conditional distributions can be multinomial, Gaussian, or any other parametric family we might choose to use.

Returning to our example above. Using a default table we can represent the probability $P(X_1 \mid C)$ using the cases 1, 2 and the default $\{3,\ldots,K\}$. This requires estimating the probability of binding site in each of the first two clusters, and the probability of observing the transcription factor in the remaining clusters.

We note that in the extreme case, when $\mathcal{L}_i$ is empty, then we are rendering X_i independent of C. To see this, note that L_i has a single value in this situation, and thus $P(X_i \mid C)$ is the same for all values C. Thus, CSI is a refinement of selective Bayesian models, and it suffices to specify the choice $\mathcal{L}_i$ for each variable.

Finally, we consider classifying a gene given a model. As in (2), the decision between two clusters is a sum of terms of the form $P(x_i \mid C=k)/P(x_i \mid C=k')$. Now if both k and k' fall in the "default" category of $\mathcal{L}_i$, then they map to the same value of L_i, and thus define the same conditional probability over X_i. In such a situation, the observation x_i does not contribute to the distinction between k and k'. We will say that X_i *distinguishes* a cluster k, if the conditional probability $P(X_i \mid C=k)$ is different than the other conditional probabilities. That is $k = k_j$ for some i.

3. SCORING CSI CLUSTERINGS

We want to *learn* CSI Clusterings from data. By learning, we mean selecting the number of clusters K, the set of dependent random variables G, the corresponding local structures $\mathcal{L}_i$, and in addition, estimating the parameters of the conditional distributions in the model. We reiterate that CSI clustering is a special sub-class of *Bayesian networks* with default tables. Thus, we adopt standard learning approaches for Bayesian networks [11, 14, 16] and specialize them for this class of models. In particular, we use a Bayesian approach for learning probabilistic models. In this approach learning is posed as an optimization problem.

In this section we review the scoring functions over different choices of clustering models, and in the next section we will consider methods for finding high-scoring clustering models.

3.1 The Bayesian Score

We assume that the set of variables $X_1,\ldots,X_N$ is fixed. We define a CSI Clustering *model* to be a tuple $\mathcal{M} = \langle K, \{\mathcal{L}_i\}\rangle$, where K specifies the number of values of the latent class and $\mathcal{L}_i$ specifies the choice of *local structure* for X_i. (Recall that X_i does not depend on C if L_i has a single value.) A model $\mathcal{M}$ is *parameterized* by a vector $\vec{\theta}_{\mathcal{M}}$ of parameters. These include the mixture parameters $\vec{\theta}_k = P(C=k)$, and the parameters $\vec{\theta}_{X_i|l}$ of $P(X_i \mid L_i = l)$.

As input for the learning problem, we are given a dataset D that consists of M samples, the m'th sample specifies a joint assignment $x_1[m],\ldots,x_N[m]$ to $X_1,\ldots,X_N$. In the *Bayesian* approach, we compute the *posterior* probability of a model, given the particular data set D:

$$P(\mathcal{M} \mid D) \propto P(D \mid \mathcal{M})P(\mathcal{M})$$

The term $P(\mathcal{M})$ is the *prior* probability of the model $\mathcal{M}$, and $P(D \mid \mathcal{M})$ is the *marginal likelihood* of the data, given the model $\mathcal{M}$.

In this paper, we use a fairly simple class of priors over models, in which $P(\mathcal{M}) = P(K)P(G)\prod_i P(\mathcal{L}_i)$. We choose these for their mathematical simplicity (which makes some of the computations below easier) and since they slightly favor simpler models. We assume that $P(K) \propto \lambda^K$ is a geometric distribution with parameter λ which is fairly close to 1. The prior over G is designed to penalize dependencies. Thus $P(G) \propto \alpha^{|G|}$ for some parameter $\alpha < 1$. (Recall that $G = \{i : \mathcal{L}_i \neq \emptyset\}$.) Finally, the prior distribution over local models (following [14]) is $P(\mathcal{L}_i) = \frac{1}{K-1}\binom{K}{|\mathcal{L}_i|}^{-1}$. Thus, we set a uniform prior over the number of cases in $\mathcal{L}_i$, and then put a uniform prior over all local structures with this cardinality.

We now consider the marginal likelihood term. This term evaluates the probability of generating the data set D from the model $\mathcal{M}$. This probability requires averaging over all possible parameterizations of $\mathcal{M}$:

$$P(D \mid \mathcal{M}) = \int P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}})P(\vec{\theta}_{\mathcal{M}} \mid \mathcal{M})d\vec{\theta}_{\mathcal{M}} \qquad (3)$$

where $P(\vec{\theta}_{\mathcal{M}} \mid \mathcal{M})$ is the *prior* density over the parameters $\vec{\theta}_{\mathcal{M}}$, and $P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}})$ is the likelihood of the data which has the form

$$\prod_m \sum_k \left(P(C=k \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \prod_i P(x_i[m] \mid l_i(k), \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \right)$$

where $l_i(k)$ is the value L_i as a function of k.

In this work we follow a standard approach to learning graphical models and use *decomposable priors* that have the form

$$P(\vec{\theta}_{\mathcal{M}} \mid \mathcal{M}) = P(\theta_C) \prod_i \prod_{l \in \text{Val}(L_i)} P(\theta_{X_i|l})$$

For multinomial X_i and for C, we use a *Dirichlet* [8] prior over the parameters, and for normal X_i, we use a *normal-gamma* prior [8]. We review the details of both families of priors in Appendix A.

The marginal likelihood term evaluates the fit of the data by averaging the likelihood of the data over all possible parameterizations of the model. We stress that this is different from the *maximum likelihood method*. In that method, one evaluates each model by the likelihood it achieves with the best parameters. That score can be misleading since "bad" models can have parameters that give the data high likelihood. Bayesian approaches avoid such "overfitting" by averaging over all possible parameterizations. This averaging regularizes the score. In fact, a general theorem [28] shows that for large data sets (i.e., as $M \to \infty$), $log P(D \mid \mathcal{M})$ is equal to

$$\log P(D \mid \mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}}) - \frac{1}{2} \log M \dim(\mathcal{M}) + O(1) \quad (4)$$

where $\hat{\vec{\theta}}_{\mathcal{M}}$ are the *maximum aposteriori probability (MAP)* parameters that maximize $P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) P(\vec{\theta}_{\mathcal{M}} \mid \mathcal{M})$, and $\dim(\mathcal{M})$ is the dimensionality of the model $\mathcal{M}$ (the number of degrees of freedom in the parameterization of $\mathcal{M}$). Thus, in the limit the Bayesian score behaves like a penalized maximum likelihood score, where the penalty depends on the complexity of the model. (Note that this approximation is closely related to the *minimum description length* (MDL) principle [26].)

3.2 Complete Data

We briefly discuss the evaluation of the marginal likelihood in the case where we have *complete data*: That is, we now assume that we are learning from a data set D_c that contains M samples, each of these specifies values $x_1[m], \ldots, x_N[m], c[m]$ for $X_1, \ldots, X_N$ and C. (In this case, we also fix in advance the number of values of C.) This setting is easier than the setting we need to deal with, however, the developments here are needed for the ones below.

For such data sets, the likelihood term $P(D_c \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}})$ can be decomposed into a product of local terms:

$$P(D_c \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) = \quad (5)$$
$$L_{\text{local}}(C, \mathcal{S}_C, \vec{\theta}_C) \prod_i \prod_{l \in \text{Val}(L_i)} L_{\text{local}}(X_i, \mathcal{S}_{X_i|l}, \vec{\theta}_{X_i|l})$$

where the L_{local} terms denote the likelihood that depends on each conditional probability distribution and the associated *sufficient statistics* vectors $\mathcal{S}_C$ and $\mathcal{S}_{X_i|l}$. These statistics are cumulative functions over the training samples. These include *counts* of the number of times a certain event occurred, or sum of the values of X_i, or X_i^2 in the samples where $L_i = l$. The details of these likelihoods and sufficient statistics are less crucial for the developments below (see Appendix A for details).

An important property of the sufficient statistics is that once we compute the counts for the case in which $L_i = C$, we can easily get counts for other local structures:

$$\mathcal{S}_{X_i|l} = \sum_k P(L_i = l \mid C = k) \mathcal{S}_{X_i|k}$$

(Note that since L_i is a deterministic function of C, $P(L_i = l \mid C = c)$ is either 0 or 1.)

The important consequence of the decomposition of Eq. 5 is that the marginal likelihood term similarly decomposes (see [14, 16])

$$P(D_c \mid \mathcal{M}) = S_{\text{local}}(C, \mathcal{S}_C) \prod_i \prod_{l \in \text{Val}(L_i)} S_{\text{local}}(X_i, \mathcal{S}_{X_i|l}) \quad (6)$$

where

$$S_{\text{local}}(X_i, \mathcal{S}_{X_i|l}) = \int L_{\text{local}}(X_i, \mathcal{S}_{X_i|l}, \vec{\theta}_{X_i|l}) P(\vec{\theta}_{X_i|l} \mid \mathcal{M}) d\vec{\theta}_{X_i|l}$$

The decomposition of marginal likelihood suggests that we can easily find the best model in the case of complete data. The intuition is that the observation of C *decouples* the modeling choices for each X_i from the other variables. Formally, we can easily see that changing L_i for X_i changes only the prior associated with that L_i and the marginal likelihood term $\prod_{l \in \text{Val}(L_i)} S_{\text{local}}(X_i, \mathcal{S}_{X_i|l})$. Thus, we can optimize the choice of each L_i separately of the others. For each such choice we compute the sufficient statistics, and evaluate the change in the score of the model.[1]

To summarize, when we have complete data the problem of learning a CSI clustering model is straightforward: We collect the sufficient statistics $\mathcal{S}_{X_i|k}$ for every X_i and $k = 1, \ldots, K$, and then we can efficiently evaluate every possible model. Moreover, we can choose the one with the highest posterior without explicitly enumerating all possible models.

3.3 Incomplete Data

We now return to the case that interests us, where we do not observe the class labels. Such a learning problem is said to have *incomplete data*. In this learning scenario, the evaluation of the marginal likelihood Eq. (3) is problematic and cannot be carried out in analytical form, and thus we need resort to approximations. See [5] for an overview of methods for approximating the marginal likelihood.

In this paper we use two such approximations to the logarithm of the marginal likelihood. The first is the *Bayesian Information Criterion* (BIC) approximation of Schwarz [28] (see Eq. (4)).

$$\text{BIC}(\mathcal{M}, \vec{\theta}_{\mathcal{M}}) = \log P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) - \frac{1}{2} \log M \dim(\mathcal{M})$$

To evaluate this score, we perform *expected maximization* (EM) iterations to find the MAP parameters [22] (see also [5, 16]). The benefit of this score is that once we found the MAP parameters, it is fairly easy to evaluate. Unfortunately, this score is only asymptotically correct, and can over-penalize models in practice.

A more accurate approximation is the *Cheeseman-Stutz* (CS) score [5, 4]. This score approximates the marginal likelihood as:

$$\begin{aligned} &\text{CS}(\mathcal{M}, \vec{\theta}_{\mathcal{M}}) = \\ &\quad \log P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) - \log P(D_c^* \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) + \\ &\quad \log P(D_c^* \mid \mathcal{M}) \end{aligned}$$

where D_c^* is a fictitious data set that is represented by a set of sufficient statistics. The computation of $P(D_c^* \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}})$ and $P(D_c^* \mid \mathcal{M})$ is then performed as though the data is complete. This simply amounts to evaluating Eq. (5) and Eq. (6) using the sufficient statistics for D_c^*.

The choice of D_c^* is such that its sufficient statistics will be the *expected sufficient statistics* given $\mathcal{M}$ and $\vec{\theta}_{\mathcal{M}}$. These are defined

[1]In theory, the choice of L_i requires to consider $O(2^K)$ alternatives, which is efficient when K is a small constant. In practice, we construct L_i by a greedy procedure that at each iteration finds the best k to separate from the default case, until no improvement is made to the score [14].

by averaging over all possible completions D_c of the data

$$E\left[\mathcal{S}_{X_i|c} \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}\right] = \sum_{D_c} \mathcal{S}^{D_c}_{X_i|c} P(D_c \mid D, \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \quad (7)$$

where D_c represents a potential *completion* of the data (i.e., assignment of cluster value to each example) and $\mathcal{S}^{D_c}_{X_i|c}$ is the sufficient statistics for X_i given c evaluated on D_c. Using the linearity of expectation, this term can be efficiently computed (see [5, 11]). Thus, to compute D^*_c, we find the MAP parameters $\hat{\vec{\theta}}_{\mathcal{M}}$, and then compute the expected sufficient statistics given $\mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}}$. We then use these within Eq. (5) and Eq. (6) as the sufficient statistics of the fictional data set D^*_c.

4. LEARNING CSI CLUSTERINGS

Our goal is to find a model $\mathcal{M}$ that attains the highest posterior probability. Unfortunately, there are $O(2^{N+K})$ choices of models with K clusters. Thus, we cannot exhaustively evaluate all the models. Instead we need to resort to some form of heuristic search. The main computational cost of such a search is evaluating candidate models. For each model, this procedure performs an EM optimization to find the MAP parameters $\hat{\vec{\theta}}_{\mathcal{M}}$, and then uses these parameters for computing the score (using either BIC or CS).

To avoid such expensive evaluations of candidates we use the framework of *Bayesian structural EM* [11]. This framework allows us to search for models *within* the EM loop. This algorithm is iterative. At the beginning of of the j'th iteration we have a model and parameters $\mathcal{M}^j, \vec{\theta}^j$. We then perform the following steps:

- **E-Step:** Compute expected sufficient statistics given $\mathcal{M}^j, \vec{\theta}^j$ as in Eq. (7).
- **M-Step:** Learn a model $\mathcal{M}^{j+1}$ as though the expected sufficient statistics were observed in a complete data set. For each X_i choose the CSI model $\mathcal{L}_i$, independently of the other variables. Then, estimate parameters $\vec{\theta}^{j+1}$ for $\mathcal{M}^{j+1}$ to be the MAP parameters given the expected sufficient statistics.

These iterations are reminiscent of the standard EM algorithm. The main difference is that in the standard approach the M-Step involves reestimating parameters, while in Structural EM we also relearn the structure. We can use two scores in choosing models at the M-Step. Either the Bayesian approach described in Section 3.2 or the BIC score.

These iterations are guaranteed to improve the score in the following sense. If we use the BIC score in the M-step, then results of [10] show that the BIC score of $\mathcal{M}^{j+1}, \vec{\theta}^{j+1}$ is not less than that of $\mathcal{M}^j, \vec{\theta}^j$. If we use the Bayesian score n the M-step, then results of [11] show that we are performing an approximate version of a procedure that does improve the Bayesian score in each iteration. In practice, most iterations do improve the CS score (which is often a reasonable approximation of the true Bayesian score).

Since the score is bounded from above, the structural EM iterations will converge. The convergence point can be a local maximum and depends on the initial starting point (see [10, 11]). In order to find a high scoring model, we examine several ways of initializing the algorithm and restarting it after it converged.

We use two different methods for initializing the procedure. The first is the *full* model (where $L_i = C$ for every node). This model is the most expressive in the class we consider, and thus allows the starting point to capture any type of "trend" in the data. We then let the following iterations to zoom in on this trend by adopting a more specific structure. The second initialization mode, is a using a random model, where each L_i is chosen at random. This allows us to use multiple starting points that can be quite different from each other, and thus lead the search to different maxima.

To "escape" local maximum we perform the following steps. Once a local maximum is achieved, we start a random first-ascent hill-climb search procedure over clustering topologies. More precisely, at each iteration the procedure chooses a variable i at random, and reverses its dependency status. More precisely, if X_i depends on C, we make it independent, and if X_i does not depend on C, we make it dependent (without any CSI structure). We then evaluate the new structure using standard EM and compute its score using the parameters found by EM. If the random modification improves the score, then we use the new structure as a starting point for a new structural EM run. If after a fixed amount of random trials no improvement was found, the procedure terminates the search and returns the best model found so far.

We note that the procedure we describe in this section is a specialization of the Structural EM framework for learning Bayesian networks [10, 11]. However, we focus on a specific class of networks (i.e., CSI clusterings). The restriction to this class of model allows us to perform an efficient E-step by exploiting the small number of statistics that are needed in order to evaluate CSI clusterings. In contrast, the procedures of [10] need to consider a much larger number of sufficient statistics, and compute them on demand during the M-step. In addition, the restricted class of networks allows us to perform an efficient M-step. In contrast, the procedures of [10] perform heuristic search.

5. EVALUATION

5.1 Simulation Studies

To evaluate the performance of our clustering method, we performed tests on synthetic data sets. These data sets were sampled from a known clustering model (which determined the number of clusters, which variables depend on which cluster value, and the conditional probabilities). Since we know the model that originated the data, we can measure how closely our procedure recovers this model based on the training data. We measure 2 aspects: First, how well does the model recover the real structure (number of clusters, false positive and false negative edges in the model). Second, is the "biological bottom line". How well does the model classify a given sample (gene). The aim of these tests is to understand how the performance of the method depends on various parameters of the learning problem.

To measure the classification success, we examined the following criterion. A clustering model $\mathcal{M}$ defines a conditional probability distribution over clusters given a sample. Let $\mathcal{M}_t$ denote the true model, and let $\mathcal{M}_e$ denote the estimated model. Both define conditional distributions over clusters. We want to compare these two conditional distributions. We will denote the clusters of the true model as C_t and the clusters of the estimated model as C_e. Then, we can define a joint distribution over these two clusterings:

$$P(C_t, C_e) = \sum_{\mathbf{x}} P(\mathbf{x} \mid \mathcal{M}_t) P(C_t \mid \mathbf{x}, \mathcal{M}_t) P(C_e \mid \mathbf{x}, \mathcal{M}_e)$$

where the sum is over all possible joint assignments to $\mathbf{X}$. In practice we cannot sum over all these joint assignments, and thus we estimate this distribution by sampling from $P(\mathbf{X} \mid \mathcal{M}_t)$. Once we have the joint distribution we can compute the *mutual information* , $I(C_t, C_e)$ between the two clustering variables [7]. This term denotes the number of bits one clustering carries about the other. In the table below we report the *information ratio* $I(C_t, C_e)/H(C_t)$, which measures how much information C_e provides about C_t compared to the maximum possible (which is the entropy of C_t).

The model we sampled from has 5 clusters, 50 continuous variables, 100 discrete node. We simulated the biological domain and assumed that most of the discrete variables were relevant for a few clusters (if any) and the that the continuous ones were relevant to many clusters. From this model we sampled training sets of size 200, 500, and 800 (each a superset of the proceeding one) and independent test set of size 2000.

To estimate the robustness of our procedure, we injected additional noise into the training set. This was done by adding genes, whose values were sampled from a fairly diffuse background probability. These obscure the clustering in the original model. We expect that biological databases will contain genes that do not fall in to "nice" functional categories. We ran our procedure on the sampled data sets that we obtained by adding 10% or 30% additional "noise" genes to the original training data.

The procedure was initialized with random starting points, and for each training data and cluster number it was run 3 times. We performed this procedure for number clusters in the range 3 to 7 and choose the model with the best score among these. Due to space considerations, we briefly summarize the highlights of the results.

Cluster number: In all runs, models learned with fewer clusters than the original model were sharply penalized. On the other hand, models learned with additional clusters got scores that were close to the score received when learning with 5 clusters. Often, we noticed that a model with 6 or 7 clusters had "degenerate" clusters that none of the "true" samples (i.e., ones that were generated from the original models) were classified to. As expected, this phenomena was stronger in the runs with 30% noise. In general BIC had stronger penalty for additional clusters, and so more often choose 5 clusters as the best scoring model

Structure accuracy: We measured the percentage of additional variables in G (false positives) and missing ones in G (false negatives). In general, the procedure (using both BIC and CS scores) tended to have very small number of false negatives (and this only in the smaller sample sizes). On the other hand, both scores had false positives, about 20% for CS, and 10%–20% for BIC.

Mutual Information Ratio: In this category all the runs with 800 training samples achieved the maximal information gain. Runs with 200 samples achieved information gain of about 90%. Runs with 500 samples had varied results that depended on the level of noises. For 10% noise we got maximal information gain, while results in the noisier data set got 95% information gain. These results show that the learned clusters were very informative of the original clusters.

Clearly, these simulations only explore a small part of the space of possible parameters. However, they show that on a model that has statistical characteristics similar to real-life datasets, our procedure can perform in a robust manner and discover clusterings that are close to the original one, even in the presence of noise.

5.2 Biological Data

We also evaluated our procedure on two biological data sets of budding yeast gene expression. The first data set is from Spellman *et al* [31] who measured expression levels of genes during different cell-cycle stages. We examined the expression of the ≈ 800 genes that Spellman *et al* identify as cell-cycle related in 77 experiments. The second data set is from Gasch *et al* [15] who measured expression levels of genes in response to different environmental changes. In this data set, we examined the genes Gasch *et al* identify in their analysis [15, Figures 3, 4, 7], and genes whose expression changed by more than a 2-fold in at least 30 experiments. This list contains 1271 genes that were were measured in 92 experiments.

Name	Description
ABF1	ARS-binding factor involved in the activation of DNA-replication and transcriptional regulation of various genes
GCN4	Transcription factor of the basic leucine zipper (bZIP) family, regulates general control in response to amino acid or purine starvation
MCM	Yeast Cell cycle and metabolic regulatorYeast Cell cycle and metabolic regulator
MIG1	Zinc-finger transcriptional repressor involved in glucose-repression
PHO4	Basic helix-loop-helix (bHLH) transcription factor required for expression of phosphate pathway, inactivated by hyperphosphorylation by Pho80p-Pho85p cyclin-dependent protein kinase
RAP1	DNA-binding protein with repressor and activator activities, also involved in silencing at telomeres and silent mating type loci
STRE1	Stress Response Element MSN2/MSN4.
STUAP	Aspergillus Stunted protein

Table 1: Description of binding sites mentioned in the clusters of Figure 1. The descriptions are based on TRANSFAC database entries and YPD descriptions.

In addition to the expression levels from these two data sets, we recorded for each gene the number of putative binding sites in the 1000bp upstream of the ORF (these sequences were retrieved from SCPD [29]). These were generated by two methods. The first is based on consensus sequences found in the SCPD [29] database. The second method is based on the "fungi" matrixes in the TRANSFAC database [17]. We used the MatInspector program [25] (run with the default parameters) to scan the upstream regions. We used these matches to count (a) the number of putative sites in the 1000bp upstream region, and (b) the number putative sites in four 250bp sub-regions of the upstream region. This generated discrete valued random variables (with values $0, 1, 2, > 2$) that correspond to each putative site (either a whole promoter region or a sub-region).

We ran our procedure with different initialization points and different number of clusters. In our experiments, a useful initialization point was k-means clustering of only the expression measurements, using the XCluster program [30].

In general, most of the clusterings we learned had the following characteristics. First, the expression measurements were considered informative by the clustering. Most of the expression variables had impact on most of the clusters. Second, most binding site measurements were considered non-informative. The learning procedure decided for most of these that they have no effect on any of the clusters. Those that were considered relevant, usually had only 1 or 2 distinct contexts in their local structure. This can be potentially due to the fact that some of these factors were truly irrelevant, or to a large number of errors made by the binding site prediction programs that mask the informative signal in these putative sites. The results were similar when we used binding sites SCPD consensus sites and TRANSFAC matrix matches.

To illustrate the type of clusterings we find, we show in Figure 1 two of the clusterings we learned. These describe qualitative "cluster profiles" that helps see which experiments distinguish each cluster, and the general trend of expression at these experiments.

As we can see, the clusters learned from the Cell-Cycle data all show periodic behavior. This can be expected since the 800 genes are all correlated with the cell-cycle. However, the clusters differ in their phase. Such clusters profiles are characteristic of many of the models we found for the Cell-Cycle data. Each of the clusters depends on somewhat different transcription factors: including CGN4 (controls synthesis of nucleotides and amino acids), ARS, MCM, RAP1 and RHO4, SCB (all related to cell cycle control). Note that these sites are located in different positions in the different clusters.

#	Size	Expression Pattern	Binding Sites
Cell Cycle			
0	112	`-+--+---+ +++---- ++++--++++++ ----+++  - -- +---+++ ++---++++-++-- -- +  +`	
1	125	`  --  +++  --  +++ --- +   -  - ++ ++---   +  -++++ --   ++---- -  ++++- +--`	STUAP[4]
2	83	`--++   - -++++ ---  ++----+++++-----  +++++ -   -- -+++ -   +++ - --- +++++++`	MCM1[4]
3	122	`--+++++- ----+++ +-- +++------+++ +  ---+++++++++-  ---++ + -- +++ ----- --+++`	
4	67	`   ---  +++ - -  ++ + -  ++   --   ++  +  --    ++++   --++ +   --- +++++++ -`	STUAP[4]
5	119	`++ ---++++ --- ++  -  +++  ---  ++++ ---   ++ +++++ - - ++---- -+  +++ --`	STUAP[4]
6	155	`--++++- --+ ++- -- -++  --+ +++- -- -+  +++   +-- --+++ -  -++++++  - - -  -`	
		`AABBCCCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDDDDDDDDDDDDEEEEE EEEEEEEEEEEEFFFFFFFFFFFFF`	
Environmental Response			
0	117	`-------- +--- ------+- -- ---------++----------+ --------- ------+- --------------    -++++`	RAP1[3]
5	142	`--------- --- --    +--- --- +---------------- ---- -----+-  - -+---   + ------    - -+++++`	ABF[4]
3	247	`--- -----+------  - ---- ---  ---------------------------+---- -+ --     -----------  --++`	ABF[4]
7	141	`--- ---- +--------   - -- ---+---------------- ---- ----- -  - -+  ---- -------   -   - +++`	ABF[4]
4	58	`+++++++++  +++++++++++++++++++   ++++++++++ +++   + +  ++ -++++++  + +++ ++++++++++++  -   -`	
1	240	`+++++++++  +++++++++-++++++++    ++++++++++ +++ + + +  ++ -++++++  + +++ +++++  ++++++ +   -`	MIG1[4]
6	178	`++++++++--+++++++++++++ ++ ++ +++++++++++++++--++ ++++++++++++++  ----++--+++++++++----++++`	STRE[4] GCN4[3] PHO4[3]
2	147	`+++++++++ ++++++++++ ++ ++ ++ +++++++++++++++++++++++++++++-+++++  ++++++++++++  ++++++++----`	STRE[3,4]
		`AAAABBBBBBCCCCCCCCCCDDDDDDDDDEEEEEEFFFFFFFFFGGG GGGGHHHHHIIIIIIIIIJJJJJJKKKKKKKKKKKLLLLLMMMMM`	

Figure 1: Schematic representation of two of the clusterings we found for the Cell-Cycle data of Spellman *et al* and Environmental Response data of Gasch *et al*. For each cluster, the table shows the number of genes that were mapped to the cluster, a qualitative account of the expression patterns, and the binding sites that were recognized. The expression patterns describe three cases: " " when the measurement does not distinguish the cluster, "+"/"-" when the measure does distinguish the cluster and the average expression is above or below, respectively, of the default mean. The last column shows the binding sites that distinguish the clusters. The index in the brackets shows which sub-region of the promoter region was recognized to have higher than expected number of putative sites. The sub-regions are 1: -1000 to -751, 2: -750 to -501, 3: -500 to -251, and 4: -250 to -1. The last row in each group shows the treatment applied in each experiment. For the Cell-Cycle data, these are: A - CLN3, B - CLB2, C - Alpha, D - CDC15, E - CDC28, F - ELU (see Spellman *et al* for details). For the environmental response data, these are A - 37C Heat Shock, B - Variable Temperature Shock, C - Hydrogen Peroxide, D - Menadione, E - DDT, F - Diamide, G - Sorbitol osmotic shock, H - Amino-acid starvation, I - Nitrogen Depletion, J - Diauxic shift, K - Stationary phase, L - Continuous carbon sources, M - Continuous temperature (see Gasch *et al* for more details).

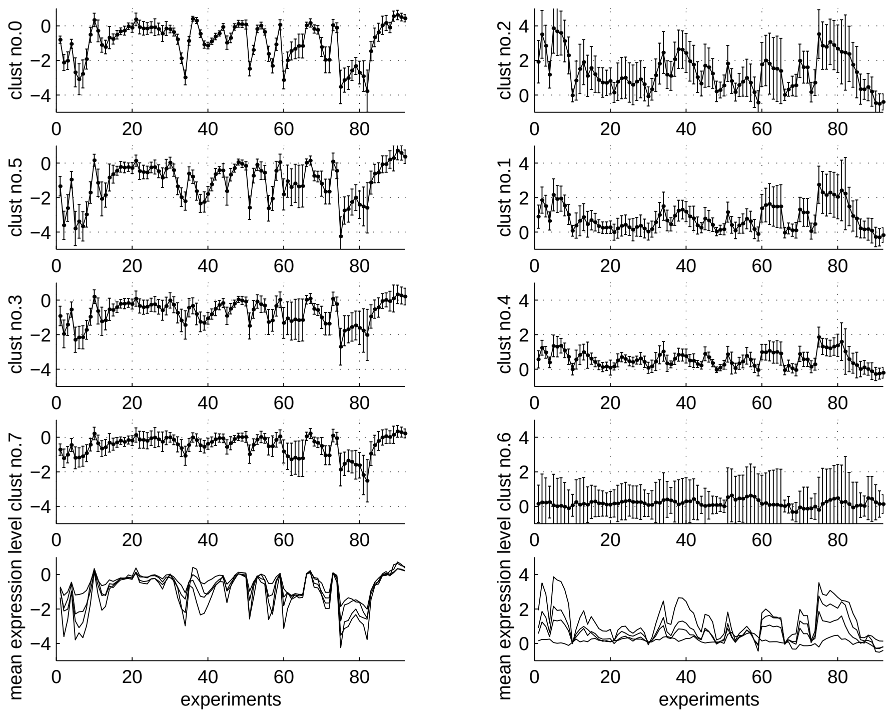

Figure 2: Detailed description of the eight clusters of the Environmental Stress data set described in Figure 1. The graphs on the left show clusters of genes that are down-regulated in response to stress, and on the right clusters of genes that are up-regulated in response to stress. Each graph shows the mean value of the expression level of genes in the clusters for each experiment, and the 1-standard deviation confidence interval around it. The bottom graphs compares the means of the clusters.

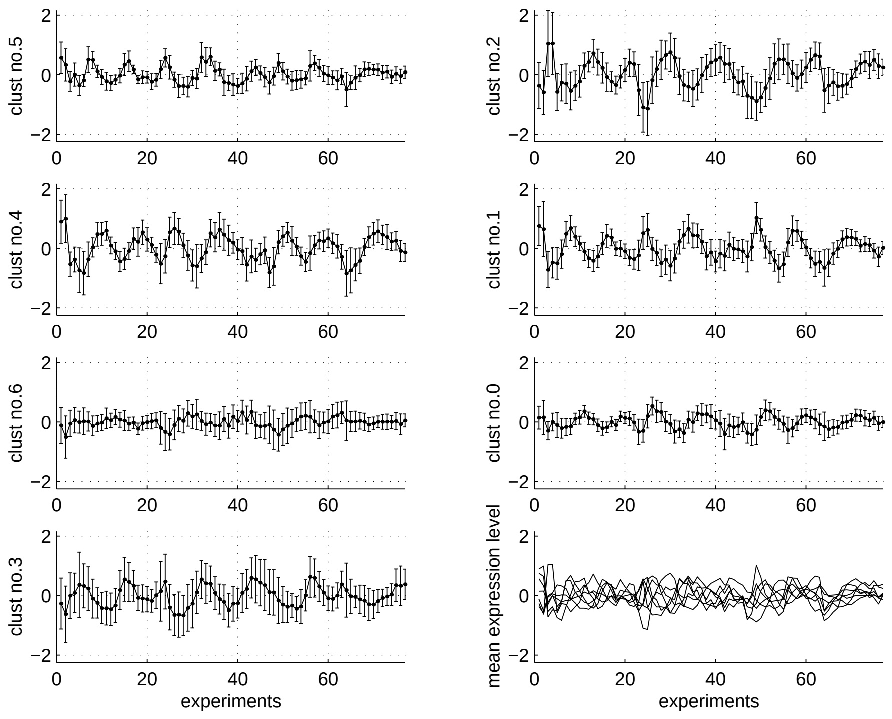

Figure 3: Detailed description of the seven clusters of the Cell-Cycle data set described in Figure 1.

This might indicate that the position of the site affects its role.

In the Environmental Response data, the best scoring models had eight clusters. In the model shown in Figure 1, we see four clusters of genes that were underexpressed in stress conditions, three clusters of genes that are overexpressed in these conditions, and one cluster of genes that were mostly unaffected by the stress conditions. This later cluster has high variance, while the others have relatively tight variance in most experiments. Some of the clusters correspond to clear biological function: For example, Cluster #0 , consists mostly (103/117) of genes that are associated with protein synthesis. Cluster #3 also contains protein synthesis genes (46/247) and RNA processing and transcription genes (24/247).

We note that the clusters describe different "levels" of response. This might be a consequence of the k-means clustering. Although the learned clustering is quite different than the initial k-means clustering (more than 50% of the genes have changed clusters in the final clustering), it might be possible that additional exploration of the model space will find other "profiles" of clusters.

6. DISCUSSION

In this paper we examined the problem of clustering genes based on a combination of genomic and genetic data. We present an approach that allows to combine gene expression data and putative binding site (from various sources), and *identify and characterize* clusters of genes. Our approach is novel in its ability to deal with many attributes that are irrelevant to the clustering, and its ability to tailor each cluster to the attributes that it depends on. Due to the nature of the underlying biological problem, we believe that our approach is more suitable than standard clustering methods that treat all variables on equal footing.

The preliminary experimental results on both synthetic and real data, suggest that this approach is robust to noise and recovers groups of genes with coherent behavior. This claim, however, needs to be substantiated by additional experiments and careful analysis of the clusters found. As described above, because of local maxima, the results are affected by the initialization point. We are working on developing more suitable search methods that will escape local maxima at low cost (without resorting to large-scale stochastic simulations). Another issue, is the noisy output of the consensus and weight matrixes putative sites searches. This noise obscures subtle signals in the data, which might be learnable with better binding sites recognition tools.

Our approach is similar to that of Holmes and Bruno [18] in that both approaches use unified probabilistic models for gene expression and binding sites. However, there are several distinct differences. Holmes and Bruno focus on the problems of finding new putative sites, as such their model combines a naive Bayes model over expression attributes and an HMM-like model of DNA sequences (as in [19, 23]). This provides more detailed models of binding sites, and can discover novel ones. However, such models assume one binding site per cluster, and cannot detect multiple regulatory sites, nor detect that some experiments are irrelevant to the definition of a specific cluster. This difference is reflected in the choice of statistical methods: Holmes and Bruno's method searches for maximum likelihood parameters in a fixed parametric model, while our approach performs Bayesian model selection.

There are several ways of extending our approach. The first one is to extend the models of binding sites to finer grain models, such as PSSM/HMM like models of binding sites. This involves replacing a random variable by a sub-model that includes additional parameters that need to be learned. The language of Bayesian networks provides tools for such complex models and the foundations for learning with them. Yet, there are algorithmic and modeling issues that need to be addressed. Such a combined approach can amplify our understandings as to the relevance of new binding site(s) to the clustering of genes.

Another important aspect of our approach is that it provides a comprehensive model of expression and the binding sites that regulate it. As such it attempts to deal with all attributes that affect the gene expression pattern at once, and does not consider each binding site without regard to its context. A possible extension is to learn direct interactions between attributes (e.g., if binding site A appears in a certain region, then the probability that it will appear in other regions will decrease). We can do so by learning a Bayesian network that models these dependencies. An attractive class of such Bayesian networks are the *tree-augmented Naive Bayes* models that were introduced in the context of supervised learning [12].

Finally, the models we learn here generalize over genes. That is, they proposed a uniform model for genes. In contrast, experiments receive individual treatment. Ideally, we would like to generalize both over genes and over experiments. This requires a model that describes 2-sided clustering: clusters of genes, and clusters of experiments. To learn clusterings that also takes into account genetic information, we need more expressive language. A potential modeling language for this problem can be found in [13].

Acknowledgements

The authors thank Jill Bejerano, Yonatan Bilu, Gal Elidan, Michal Linial, Dana Peer, Eran Segal, and the anonymous reviewers for comments on previous drafts of the papers. This research was supported in part by ISF grant 244/99 and Israeli Minstry of Science grant 2008-1-99. Nir Friedman was supported by Alon Fellowship.

7. REFERENCES

[1] M. Bittner, P. Meltzer, and J. Trent. Data analysis and integration: of steps and arrows. *Nature Genetics*, 22:213–215, 1999.

[2] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proc. 12'th Conf. Uncertainty in Artificial Intelligence*, pp. 115–123. 1996.

[3] A. Brazma and J. Vilo. Gene expression data analysis. *FEBS Lett*, 480(1):17–24, 2000.

[4] P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In *Advances in Knowledge Discovery and Data Mining*, pages 153–180. 1995.

[5] D. M. Chickering and D. Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29:181–212, 1997.

[6] D. M. Chickering, D. Heckerman, and C. Meek. A Bayesian approach to learning Bayesian networks with local structure. In *Proc. 13'th Conf. Uncertainty in Artificial Intelligence*, pp. 80–89, 1997.

[7] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

[8] M. H. DeGroot. *Probability and Statistics*. 1989.

[9] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood form incomplete data via the EM algorithm. *J. Royal Statistical Society B*, 39:1–38, 1977.

[10] N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proc. 14'th Inter. Conf. on Machine Learning*, pp. 125–133. 1997.

[11] N. Friedman. The Bayesian structural EM algorithm. In *Proc. 14'th Conf. Uncertainty in Artificial Intelligence*, pages 129–138. 1998.

[12] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[13] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proc. 16'th Inter. Joint Conf. Artificial Intelligence*, 1999.

[14] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In *Learning in Graphical Models*. 1998.

[15] 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.

[16] D. Heckerman. A tutorial on learning with Bayesian networks. In *Learning in Graphical Models*. 1998.

[17] T. Heinemeyer, X. Chen, H. Karas, A.E. Kel, O.V. Kel, I. Liebich, T. Meinhardt, I. Reuter, F. Schacherer, and E. Wingender. Expanding the TRANSFAC database towards an expert system of regulatory molecular mechanisms. *NAR*, 27:318–322, 1999.

[18] I. Holmes and W. Bruno. Finding regulatory elements using joint likelihoods for sequence and expression profile data. In *ISMB*. 2000.

[19] J. D. Huges, P. E. Estep, S. Tavazoie, and G. M. Church. Computational identification of cis-regulatury elements associated with groups of functional related genes in *saccharomyces cerevisiae*. *J. Molecular Biology*, 296:1205–1214, 2000.

[20] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C.F. Lee, J.M. Trent, L.M. Staudt, J. Hudson, M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P.O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.

[21] P. Langley and S. Sage. Induction of selective Bayesian classifiers. In *Proc. 10'th Conf. Uncertainty in Artificial Intelligence*, pp. 399–406. 1994.

[22] S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201, 1995.

[23] C. E. Lawrence, S. F. Altschul, M. S. Boguski, Liu J. S., R. F. Neuwald, and J. C. Wooton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 1993.

[24] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. 1988.

[25] K. Quandt, K. Frech, H. Karas, E. Wingender, and T. Werner. Matind and matinspector - new fast and versatile tools for detection of consensus matches in nucleotide sequence data. *NAR*, 23:4878–4884, 1995.

[26] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[27] F.P. Roth, P.W. Hughes, J.D. Estep, and G.M. Church. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat. Biotechnol.*, 16:939–945, 1998.

[28] G. Schwarz. Estimating the dimension of a model. *Ann. Stat.*, 6:461–464, 1978.

[29] SCPD:the promotor database of *saccharomyces cerevisiae*. http://cgsigma.cshl.org/jian/.

[30] G. Sherlock. XCluster clustering software, 2000. http://genome-www.stanford.edu/ sherlock/cluster.html.

[31] 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.

[32] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nat Genet*, 22(3):281–5, 1999.

[33] J. Vilo, A. Brazma, I. Jonassen, A. Robinson, and E. Ukkonen. Mining for putative regulatory elements in the yeast genome using gene expression data. In *ISMB*. 2000.

APPENDIX

A. SUFFICIENT STATISTICS AND CONJUGATE PRIORS

In this appendix we review the details of Dirichlet and Normal Gamma priors. This is mostly text book material and can be found, for example, in [8].

A.1 Dirichlet Priors

Let X be a random variable that can take K possible values that without loss of generality are named $\{1, \ldots K\}$. A parameterization for X is a vector $\vec{\theta}_X = \langle \theta_1, \ldots, \theta_K \rangle$ such that $\theta_i \geq 0$ and $\sum_i \theta_i = 1$. Suppose, we are given a training set of N independent draws $x_1, \ldots, x_N$ of X from an unknown multinomial distribution P^*. The *likelihood* of the observations for given parameters is:

$$P(x_1, \ldots, x_N \mid \vec{\theta}_X) = \prod_i \theta_i^{N_i}$$

where N_i is the number of occurrences of the symbol i in the sequence $x_1, \ldots, x_N$. The vector $\mathcal{S} = \langle N_1, \ldots, N_K \rangle$ is the *sufficient statistics* of the sequence $x_1, \ldots, x_N$. If two sequences are such that they have the same counts, then their likelihood is the same.

The *multinomial estimation* problem is to find a good approximation for P^*. This problem can be stated as the problem of predicting the outcome x_{N+1} given $x_1, \ldots, x_N$. Given a prior distribution over the possible multinomial distributions, the Bayesian estimate is:

$$\begin{aligned} & P(x_{N+1} \mid x_1, \ldots, x_N) = \\ & \qquad \int P(x_{N+1} \mid \vec{\theta}) P(\vec{\theta} \mid x_1, \ldots, x_N) d\vec{\theta} \end{aligned} \qquad (8)$$

The posterior probability of $\vec{\theta}$ can be rewritten using Bayes law as:

$$P(\vec{\theta} \mid x_1, \ldots, x_N) \quad \propto \quad P(\vec{\theta}) \prod_i \theta_i^{N_i} \qquad (9)$$

The family of *Dirichlet* distributions is *conjugate* to the multinomial distribution. That is, if the prior distribution is from this family, so is the posterior. A Dirichlet prior for X is specified by *hyperparameters* $\alpha_1, \ldots, \alpha_K$, and has the form:

$$P(\vec{\theta}) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta_i^{\alpha_i - 1}$$

for $\sum_i \theta_i = 1$ and $\theta_i \geq 0$ for all i, where $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$ is the *gamma* function. Given a Dirichlet prior, the initial prediction for each value of X is $P(X_1 = i) = \int \theta_i P(\vec{\theta}) d\vec{\theta} = \alpha_i / \sum_j \alpha_j$. It is easy to see that, if the prior is a Dirichlet prior with hyperparameters $\alpha_1, \ldots, \alpha_K$, then the posterior is a Dirichlet with hyperparameters $\alpha_1 + N_1, \ldots, \alpha_K + N_K$. Thus, we get that the prediction for X^{N+1} is

$$P(X_{N+1} = i \mid x_1, \ldots, x_N) = \frac{(\alpha_i + N_i)}{\sum_j (\alpha_j + N_j)}$$

We can think of the hyperparameters α_i as the number of "imaginary" examples in which we saw outcome i. Thus, the ratio between hyperparameters corresponds to our initial assessment of the relative probability of the corresponding outcomes. The total weight of the hyperparameters represent our confidence (or entrenchment) in the prior knowledge. As we can see, if this weight is large, our estimates for the parameters tend to be further off from the empirical frequencies observed in the training data.

Finally, to score models we also need to compute the marginal probability:

$$\begin{aligned} P(x_1, \ldots, x_N) &= \int P(x_1, \ldots, x_N \mid \vec{\theta}) P(\vec{\theta}) d\vec{\theta} \\ &= \frac{\Gamma(\sum_i \alpha_i)}{\Gamma(\sum_i \alpha_i + N_i)} \prod_i \frac{\Gamma(\alpha_i + N_i)}{\Gamma(\alpha_i)} \end{aligned}$$

A.2 Normal-Gamma Priors

Let X be a continuous variable with a Gaussian distribution. The parameterization of $P(X)$ is usually specified by the mean μ and the variance σ^2. In our treatment, we will use the *precision* $\tau = \frac{1}{\sigma^2}$ instead of the variance. As we shall see it is more natural to use this parameterization in Bayesian reasoning. The likelihood function in this case is

$$P(x_1, \ldots, x_N \mid \mu, \tau) = \prod_i \sqrt{\frac{\tau}{2\pi}} \exp\left\{ -\frac{1}{2}\tau(x_i - \mu_i)^2 \right\}$$

It turns out that the sufficient statistics for this likelihood function are N (the number of samples), $T_1 = \sum_i x_i$, and $T_2 = \sum_i x_i^2$. Then we can write

$$\begin{aligned} & P(x_1, \ldots, x_N \mid \mu, \tau) = \\ & \qquad \exp\left\{ N\frac{1}{2}(\log \tau - \tau\mu^2) + T_1 \frac{1}{2}\tau\mu - T_2 \frac{1}{2}\tau \right\} \end{aligned}$$

The *normal-gamma* distribution is a conjugate prior for this likelihood function. This prior over μ, τ is defined as

$$P(\mu, \tau) \propto \tau^{\frac{1}{2}} e^{-\frac{1}{2}\lambda\tau(\mu - \mu_0)^2} \tau^{\alpha - 1} e^{\beta\tau}$$

where μ_0, λ, α, and β are the hyper-parameters. This prior has a gamma distribution over τ and a normal distribution for $P(\mu \mid \tau)$ with mean μ_0 and precision $\lambda\tau$.

The posterior distribution, after we have seen $x_1, \ldots, x_N$ with sufficient statistics $\langle N, T_1, T_2 \rangle$ has hyper-parameters μ_0', λ', α', β', where $\mu_0' = \frac{\lambda\mu_0 + NT_1}{\lambda + N}$, $\lambda' = \lambda + N$, $\alpha' = \alpha + \frac{1}{2}N$, and $\beta' = \beta + \frac{1}{2}N(T_2 - T_1^2) + \frac{N\lambda(T_1 - \mu_0)^2}{2(\lambda + N)}$. Finally, the marginal likelihood is

$$\begin{aligned} P(x_1, \ldots, x_N) &= \int P(x_1, \ldots, x_N \mid \vec{\theta}) P(\vec{\theta}) d\vec{\theta} \\ &= (2\pi)^{-\frac{1}{2}N} \left(\frac{\lambda}{\lambda + N} \right)^{\frac{1}{2}} \frac{\Gamma(\alpha + \frac{1}{2}N)}{\Gamma(\alpha)} \beta^\alpha \beta'^{-(\alpha + \frac{1}{2}N)} \end{aligned}$$

where β' is defined as above.