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 measurements of genome-wide expression patterns allows us to apply computational tools to examine gene regulation by transcription factors. 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. Such a model represents the joint distribution of transcription factor binding sites and of expression levels of a gene in a unified probabilistic 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 life data and analyze the biological insights it provides. Finally, we demonstrate the applicability of the method to other data analysis problems in gene expression data.

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 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 (Iyer et al. 1999, Spellman et al. 1998).

The combination of these two important data sources can lead to better understanding of gene regulation (Bittner et al. 1999, Brazma & Vilo 2000). 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 the above 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 (Roth et al. 1998, Tavazoie et al. 1999, Vilo et al. 2000). 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, we focus 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 biological hypothesis we described 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.

Due to the stochastic nature of biological processes and experimental artifacts, gene expression measurements are inherently noisy. In addition, identification of putative binding sites is also noisy, and can suffer from both false positive and false negative errors. All of this indicates that using a probabilistic model in which we treat both expression and pattern identification as random variables, might lead to a better understanding of the biological mechanism as well as improve gene functional characterization and transcription sites identification.

Using this probabilistic approach, 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 clusters; (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* (Friedman 1997, Friedman 1998), an efficient search method over different models. We evaluate the resulting method on synthetic data, and apply it to real-life data. Finally, we also demonstrate the applicability and generality of the method to other problems and data sources by introducing into the model data from phylogenetic profiling and clustering experiments by their gene expression profiles.

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

In this section we describe the class of probabilistic models we to learn from data. We develop the models in a sequence of steps starting from a fairly well known model for Bayesian clustering, and refining the representation to explicitly capture the structures we want to learn. We stress that at this stage we are focusing on what can be represented by the class of models, and we examine how to learn them in subsequent sections.

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 for 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:

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

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

One must also bare in mind that such models are not necessarily a representation of real biological structure but rather a mathematical model that can give us insights into the biological connections between variables. The independence assumptions we make are *conditional ones*. For example, we assume that *given* the model, the genes are independent. That is, after we know the model, observing the expression levels of a single gene does not help predict better the expression levels of another gene. Similarly, we assume that expression level of the same gene in different condition are independent *given* the cluster the gene belongs to. This assumption states that the cluster captures the "first order" description of the gene's behavior, and we treat (in the model) all other fluctuations as noise that is independent in each measurement.

We attempt to be precise and explicit about the independence assumptions we make. However, we note that most clustering approaches we know of treat (explicitly or implicitly) genes as being independent of each other, and quite often also treat different measurement of the same gene as independent observations of the cluster.

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 us to use fairly efficient learning algorithms, such as *expectation maximization* (EM) (Dempster et al. 1977).

The distribution form in Eq. (1) specifies the global *structure* of the naive Bayesian 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

$$P(X_i \mid C = k) \sim \text{Multinomial}(\{\theta_{x_i|k} : x_i \in \text{Val}(X_i)\}$$

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 $C = k$. These 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 a *Gaussian* distribution

$$P(X_i \mid C = k) \sim N(\mu_{X_i|k}, \sigma^2_{X_i|k})$$

such that

$$P(x_i \mid C = k) = \frac{1}{\sqrt{2\pi}\sigma_{X_i|k}} \exp\left\{-\frac{(x_i - \mu_{X_i|k})^2}{2\sigma^2_{X_i|k}}\right\}.$$

We use the Gaussian model in this situation for two reasons. First, as usual, the Gaussian distribution is one of the simplest continuous density models and allow efficient estimation. Second, when we use as observations the logarithm of the expression level (or logarithms of ratios of expression between a sample and a common control sample), gene expression has roughly noise characteristics. We note however, that most of the developments in this paper can be achieved with more detailed (and realistic) noise models for gene expression.

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

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

Thus, we can view the decision boundary 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 should be considered as "noise" since they 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 a 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. Thus, a combination of many irrelevant variables might lead us to overlook the relevant ones. As a consequence, the learned model discriminate clusters by the values of the irrelevant variables. Such clusters suffer from high variability (because of their "noisy" character).

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:

$$P(X_1, \ldots, X_N) = P(X_1) \sum_k P(C = k) P(X_2 \mid C = k) \cdots P(X_N \mid C = k).$$

This representation of the joint probability 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, not all binding sites of transcription factors play an active role in the conditions in which expression levels were measured. Another example, is a putative binding site (suggested by some search method or other) that does not correspond to a biological 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 $G \subseteq \{X_1, \ldots, X_N\}$ that represents the set of variables that depend on the cluster variable C. The joint distribution then takes the form of

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

We note that this class of models is essentially a special subclass of *Bayesian networks* (Pearl 1988). Similar models were considered for a somewhat different application in *supervised* learning by Langley and Sage (1994).

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

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

This formally demonstrates the intuition that variables outside of G do not influence the choice of clusters.

	$X=0$	$X=1$	$X=2$	$X=3$
$C=1$	0.1	0.1	0.5	0.3
$C=2$	0.1	0.1	0.2	0.6
$C=3$	0.7	0.2	0.05	0.05
$C=4$	0.7	0.2	0.05	0.05
$C=5$	0.7	0.2	0.05	0.05
$C=6$	0.7	0.2	0.05	0.05

(a) explicit table representation

	$X=0$	$X=1$	$X=2$	$X=3$
$C=1$	0.1	0.1	0.5	0.3
$C=2$	0.1	0.1	0.2	0.6
$C=*$	0.7	0.2	0.05	0.05

(b) default table

Figure 1: Example of two representations of the same conditional distribution $P(X \mid C)$.

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 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 $X_1 \in G$. 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 for two classes, say 1 and 2, this introduces unnecessary complexity: 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 refines 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 independences* (CSI): once, we know that $C \notin \{1, 2\}$, then X_1 is independent of C. This issue has received much attention in the probabilistic reasoning community (Boutilier et al. 1996, Chickering et al. 1997, Friedman & Goldszmidt 1998).

Here, we choose a fairly simple representation of CSI that Friedman & Goldszmidt (1998) term *default tables*. 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 $k_j \in \{1, \ldots, K\}$. Each k_j 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:

$$P(X_i \mid C = k) = \begin{cases} P(X_i \mid C = k_j) & k = k_j \in \mathcal{L}_i \\ P(X_i \mid k \notin \mathcal{L}_i) & \text{otherwise} \end{cases}$$

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 $L_i = j$ if $C = k_j \in \mathcal{L}_i$, and $L_i = l + 1$ if $C = k \notin \mathcal{L}_i$. 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, Instead of representing the probability $P(X_1 \mid C)$ as a complete table, as in Figure 1(a), we can represent it using a more succinct table with the cases 1, 2 and the default $\{3, \ldots, K\}$ as shown in Figure 1(b). This requires estimating a different probability of X_1 in each of the first two clusters, and one probability of X_1 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, since CSI is a refinement of selective Bayesian models, it suffices to specify the choice $\mathcal{L}_i$ for each variable.

Finally, we consider classifying a gene given a model. As in Eq. (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'. On the other hand We will say that X_i *distinguishes* a cluster k_j, if $k_j \in \mathcal{L}_i$ indicating a unique conditional distribution for X_i given the cluster k_j.

3 Scoring CSI Clustering

We want to *learn* CSI Clustering 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 (Friedman 1998, Friedman & Goldszmidt 1998, Heckerman 1998) 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 of some scoring function.

In this section we review the scoring functions over different choices of clustering models (including both structure and parameters). In the next section, we consider methods for finding the high-scoring clustering models. That is, we describe computational procedures for searching the vast space of possible structures efficiently.

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 $\mathcal{L}_i = \emptyset$.) 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 the model prior decomposes into several independent components, as suggested by Friedman & Goldszmidt (1998))

$$P(\mathcal{M}) \propto P(K)P(G)\prod_i P(\mathcal{L}_i).$$

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 is set to $P(\mathcal{L}_i) \propto \frac{1}{K}\binom{K}{|\mathcal{L}_i|}^{-1}$. 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 choose these priors for their mathematical simplicity (which makes some of the computations below easier) and since they slightly favor simpler models.

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 $\mathcal{M}$:

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

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

$$P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) = \prod_m \sum_k \left(P(C = k \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \prod_i P(x_i[m] \mid l_i(k), \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \right) \tag{4}$$

where $l_i(k)$ is the value of L_i when $C = k$.

In this work we follow a standard approach to learning graphical models and use *decomposable priors* for a given model parameters $\vec{\theta}_{\mathcal{M}}$ that have the form

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

For multinomial X_i and for C, we use a *Dirichlet* (DeGroot 1970) prior over the parameters, and for normal X_i, we use a *normal-gamma* prior (DeGroot 1970). We review the details of both families of priors in Appendix A.

We stress that the Bayesian method is different from the *maximum likelihood method*. In the latter, one evaluates each model by the likelihood it achieves with the best parameters. That can be misleading since poor models might have specific parameters that give the data high likelihood. Bayesian approaches avoid such "over-fitting" by averaging over all possible parameterizations. This averaging regularizes the score. In fact, a general theorem (Schwarz 1978) shows that for large data sets (i.e., as $M \to \infty$)

$$\log P(D \mid \mathcal{M}) = \log P(D \mid \mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}}) - \frac{1}{2}\log M \dim(\mathcal{M}) + O(1) \tag{5}$$

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.[1] Note that this approximation is closely related to the *minimum description length* (MDL) principle (Rissanen 1978).

[1]Note that as $M \to \infty$ the maximum likelihood parameters and the MAP parameters converge to the same values.

3.2 Complete Data

We briefly discuss the evaluation of the marginal likelihood in the case where the data is complete. This setting is easier than the setting we need to deal with, however, the developments here are needed for the ones below. In the *complete data* case we 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.) 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:

$$P(D_c \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) = L_{\text{local}}(C, \mathcal{S}_C, \vec{\theta}_C) \prod_i \prod_{l \in \mathcal{L}_i} L_{\text{local}}(X_i, \mathcal{S}_{X_i|l}, \vec{\theta}_{X_i|l}) \qquad (6)$$

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 particular detail of these likelihoods and sufficient statistics are less crucial for the developments below, and so we defer them to Appendix A.

An important property of the sufficient statistics is that once we compute the statistics for the case in which $|\mathcal{L}_i| = |C|$, i.e. we have a separate conditional distribution for each cluster in a node X_i, we can easily get statistics for other local structures, as a sum over the relevant statistics for each $l \in \mathcal{L}_i$:

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

(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. 6 and the corresponding decomposition of the prior, is that the marginal likelihood term also decomposes (see (Friedman & Goldszmidt 1998, Heckerman 1998))

$$P(D_c \mid \mathcal{M}) = S_{\text{local}}(C, \mathcal{S}_C) \prod_i \prod_{l \in \text{Val}(L_i)} S_{\text{local}}(X_i, \mathcal{S}_{X_i|l}) \qquad (7)$$

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.

Note that there are 2^K possible choices of $\mathcal{L}_i$. For each such choice we compute the sufficient statistics, and evaluate the score of the model. When K is small we can exhaustively evaluate all these choices. In such a situation we are find the optimal model given the data. In most learning scenarios, however, K is large enough to make such enumeration unfeasible. Thus, instead, we construct L_i by a greedy procedure (Friedman & Goldszmidt 1998) that at each iteration finds the best k to separate from the default case, until no improvement is made to the score.

To summarize, when we have complete data the problem of learning a CSI clustering model is straightforward: We collect the sufficient statistics $\mathcal{S}_{X_i|c_k}$ 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. Instead, we simply decide what is the best L_i for each X_i, independently of the decisions made for the other variables.

3.3 Incomplete Data

We now return to the case of interest to 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 as we need to summarize over all completions of the missing data. We denote the missing part of the data as D_H. In our case, this consist of assignment to clusters for the M samples. Using this notation, we can write Eq. (3) as:

$$P(D \mid \mathcal{M}) = \int_{\theta} \sum_{D_H} P(D, D_H \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) P(\vec{\theta}_{\mathcal{M}} \mid \mathcal{M}) d\vec{\theta}_{\mathcal{M}}$$

Although $P(D, D_H \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}})$ is a product of local terms, we cannot decompose the marginal likelihood. Moreover, unlike the complete data term, we cannot learn the structure of $P(X_i \mid C)$ independently of learning the structure of other conditional probabilities. Since we do not observe the values of the cluster variables, these choices interact. As a consequence, we cannot compute the marginal likelihood in an analytical form. Instead, we need to resort to approximations. We refer the reader to Chickering & Heckerman (1997) 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 (1978)(see Eq. (5)).

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

To evaluate this score, we perform *expected maximization* (EM) iterations to find the MAP parameters (Lauritzen 1995); see also (Chickering & Heckerman 1997, Heckerman 1998). The benefit of this score is that once we find the MAP parameters, it is fairly easy to evaluate. Unfortunately, this score is only asymptotically correct, and can over-penalize models for complexity in practice.

Another possible approximation is the *Cheeseman-Stutz* (CS) score (Cheeseman & Stutz 1995); see also (Chickering & Heckerman 1997). This score approximates the marginal likelihood as:

$$\text{CS}(\mathcal{M}, \vec{\theta}_{\mathcal{M}}) = \log P(D \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) - \log P(D^*_c \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}) + \log P(D^*_c \mid \mathcal{M})$$

where D^*_c is a fictitious data set that is represented by a set of sufficient statistics. The computation of $P(D^*_c \mid \mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}})$ and $P(D^*_c \mid \mathcal{M})$ is then performed as though the data is complete. This simply amounts to evaluating Eq. (6) and Eq. (7) using the sufficient statistics for D^*_c.

The choice of D^*_c is such that its sufficient statistics will match the *expected sufficient statistics* given $\mathcal{M}$ and $\vec{\theta}_{\mathcal{M}}$. These are defined by averaging over all possible completions D_c of the data

$$E\left[\mathcal{S}_{X_i|c_k} \mid \mathcal{M}, \vec{\theta}_{\mathcal{M}}\right] = \sum_{D_c} \mathcal{S}^{D_c}_{X_i|c_k} P(D_c \mid D, \mathcal{M}, \vec{\theta}_{\mathcal{M}}) \qquad (8)$$

where D_c represents a potential *completion* of the data (i.e., assignment of cluster value to each example) and $\mathcal{S}^{D_c}_{X_i|c_k}$ is the sufficient statistics for X_i given $C = k$ evaluated on D_c. Using the linearity of expectation, this term can be efficiently computed (Chickering & Heckerman 1997, Friedman 1998). Thus, to compute D_c^*, we find the MAP parameters $\vec{\theta}_{\mathcal{M}}$, and then compute the expected sufficient statistics given $\mathcal{M}$,$\vec{\theta}_{\mathcal{M}}$. We then use these within Eq. (6) and Eq. (7) as the sufficient statistics of the fictional data set D_c^*.

4 Learning CSI Clustering

4.1 Structural EM

Once we set our prior probabilities, and decide on the type of approximation we use (either BIC or CS), we implicitly induce a score over all possible models. Our goal is to identify the model $\mathcal{M}$ that attains the highest score. Unfortunately, for a fixed K, there are $O(2^{NK})$ choices of models with K clusters and N variables, therefore we cannot exhaustively evaluate the score on all models.

The typical way to handle this difficulty is by resorting to a heuristic search procedure. *Local* search procedures traverse the space of models by performing local changes (e.g., changing one of the $\mathcal{L}_i$ by adding or removing a case in the default table). The main computational cost of such a search is evaluating candidate model. Remember that since we have incomplete data, we cannot directly score a candidate models. Instead, for each candidate model we want to score, we perform another search in the parameter space (using techniques such as EM) to find the MAP parameters and then use these parameters for computing the score. Thus, the search procedure spends non-negligible computation per candidate. This severely limits the set of candidates that it can explore.

To avoid such expensive evaluations of candidates, we use the framework of *Bayesian structural EM* (Friedman 1998). In this framework, we use our current candidate to "complete" the missing values (i.e., cluster assignments). We then perform structure learning as though we have complete data, searching (efficiently) for a better model. This results in a new "best" model (with it's optimized parameters). This new model, forms the basis for the next iteration, and so on. This procedure has the benefit that structure selection is done in a situation that resembles complete data. In addition, each iteration can find a model that is quite different from the model at the beginning of the iteration. In this sense, the local moves of standard search procedure are replaced by global moves. Finally, the procedure is proven to improve the structure in each iteration.

More specifically, the Structural EM procedure consists of repeated iterations. We initialize the process with a model $\mathcal{M}^0, \vec{\theta}^0$. We discuss below the choice of this starting point. Then at the $\ell + 1$'th iteration we start with the pair $\mathcal{M}^\ell, \vec{\theta}^\ell$ of the previous iteration and construct a new pair $\mathcal{M}^{\ell+1}, \vec{\theta}^{\ell+1}$. This iteration consists of three steps.

- **E-Step:** Compute expected sufficient statistics
$$\tilde{\mathcal{S}}^\ell_{X_i|c_j} = E\left[\mathcal{S}_{X_i|c_k} \mid \mathcal{M}^\ell, \vec{\theta}^\ell\right]$$
for each $i = 1, \ldots, N$ and each $k = 1, \ldots, K$ using Eq. (8).
- **M-Step:** Learn a model $\mathcal{M}^{\ell+1}$ and parameters $\vec{\theta}^{\ell+1}$ using these expected sufficient statistics, as though they were observed in a complete data set. For each X_i choose the scoring CSI model $\mathcal{L}_i$ that maximizes the score with respect to the sufficient statistics. This is done independently for each of the variables.
- **Postprocessing-Step:** Maximize the parameters for $\mathcal{M}^{\ell+1}$ by running parametric EM. This optimization is initialized by 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 re estimating parameters, while in Structural EM we also relearn the structure. More precisely, Structural EM enables us to evaluate each possible new $\mathcal{L}_i$ based on the sufficient statistics computed with the current $\mathcal{L}_i$ instead of doing an expensive EM procedure for each such candidate.

In applying this procedure, we can use different scores in choosing models at the M-Step. This depends on the approximation we set out to use on the incomplete data. Above we discussed 2 different scores. The first one is the BIC approximation. In this case, we simply evaluate structures in the M-step using BIC on complete data (the likelihood in this case decomposes, and the complexity penalty remains the same). The second one is the CS approximation. In this case, note that CS applied to complete data is simply the Bayesian score (since $\log P(D \mid \mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}})$ and $\log P(D_c^* \mid \mathcal{M}, \hat{\vec{\theta}}_{\mathcal{M}})$ cancel out). Thus, in this case we use the exact Bayesian score with respect to the expected sufficient statistics.

These iterations are guaranteed to improve the score in the following sense. Each iteration finds a candidate that has better score (with respect to the incomplete training data) than the previous one. More precisely, if we use the BIC score (with respect to the expected sufficient statistics) in the M-step, then results of Friedman (1997) show that the BIC score of $\mathcal{M}^{\ell+1}, \vec{\theta}^{\ell+1}$ is greater than the BIC score $\mathcal{M}^{\ell}, \vec{\theta}^{\ell}$, unless the procedure converged in which case the two scores will be equal. Thus, each step improves the score we set out to maximize, and at some point the procedure will reach a (local) maxima.

When we use the CS score, the situation is more complicated. The results of Friedman (1998) show that each iteration is an approximate version of a procedure that does improve the Bayesian score on the incomplete data. In practice, most iterations do improve the CS score.

We use two different methods for initializing the structural EM procedure. In the first one, we start with the *full* model (where $|\mathcal{L}_i| = |C|$ for every variable X_i 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. The second initialization method, is by using a random model, where G (i.e. the set of variables dependent on the hidden cluster variable) is chosen at random. In both cases, we apply aggressive parametric optimization to find the initial parameters. This is done by using 100 random starting points for parametric EM, and returning the parameter vector that achieves the highest score.

4.2 Escaping Local Maxima

The structural EM procedure, as described above, can get trapped in "local" maxima. That is, it can reach sub-optimal convergence points. This can be a serious problem, since some of these convergence points are much worse than the optimal model, and thus lead to a poor clustering.

A naive way to avoid this problem is by multiple restarts. However, when the number of local maxima is large, such multiple restarts have limited utility. Instead, we want strategies for escaping local maxima that improve on the solution found by earlier iterations. We implemented two approaches for escaping local maxima.

In the first approach, we apply a directed search once the structural EM procedure converges. More specifically, assume that $\mathcal{M}^{\ell}$ is the convergence point of structural EM. Starting from this model, we apply a local search procedure that attempts to add and remove variables to the model. As explained above, such a procedure is costly since it has to separately evaluate each candidate it proposes. To avoid evaluating all moves from the current model, we apply randomized moves and evaluate each one. Once a candidate with a score higher than that of $\mathcal{M}^\ell$ is found, we restart structural EM from that point. 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.

In the second approach, we use annealing-like procedure to introduce randomness at each step of the process. This randomness needs to serve two purposes. On the one hand, a certain amount of randomness will allow the procedure to escape convergence points of structural EM. On the other hand, we want our steps to exploit the sufficient statistic computed in the E-step to choose models that build on information learned in previous iterations.

We achieve this goal by using a variant of Structural EM recently suggested by Elidan et al. (2001) and Friedman et al. (2002). The idea is simple: at each iteration of Structural EM, we perform a random *reweighting* of the training samples. More precisely, for each sample m, we sample a weight w_m^ℓ from a Gamma distribution with mean 1 and variance σ^ℓ, where σ^ℓ is an additional parameter that controls the "temperature" of the search.

In the modified E-step we compute *weighted* sufficient statistics

$$E\left[\mathcal{S}_{X_i|c_k} \mid W^\ell, \mathcal{M}, \vec{\theta}_\mathcal{M}\right] = \sum_{D_c} w_m^\ell \mathcal{S}_{X_i|c_k,D_c} P(D_c \mid D, \mathcal{M}, \vec{\theta}_\mathcal{M})$$

We then apply the M-Step with respect to these reweighted expected sufficient statistics. Additionally, we set $\sigma_{\ell+1}$ to be $\gamma \cdot \sigma_\ell$ where $\gamma < 1$ is a decay factor. The search is terminated once σ_ℓ reaches a certain predetermined threshold. In our experiments, the annealed approach dominated in performance the approach described above.

5 Evaluation

5.1 Simulation Studies

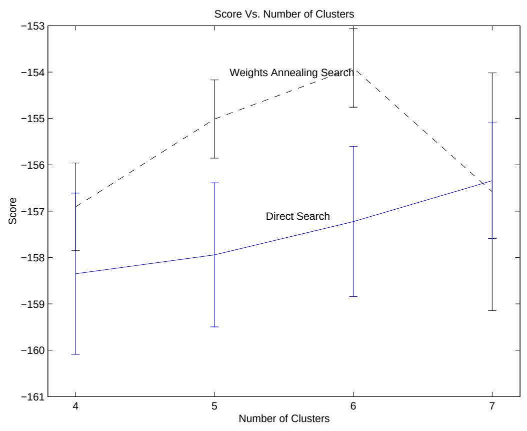
To evaluate the applicability of our clustering method, we started by performing tests on synthetic data sets. These data sets were sampled from a known clustering model (which determined the number of clusters, which variables depend on which cluster value, and the conditional probabilities). Since we know the model that originated the data, we can measure the performance of our procedure. We examined two aspects. First, how well the procedure recovers the structure of the real model (number of clusters, false positive and false negative edges in the model). Second, how well the procedure recovers the original clustering. That is, how well the model classifies a new sample (gene). The aim of these tests is to understand how the performance of the method depends on various parameters of the learning problem. We will review our techniques for evaluating the learning process results and then turn to describe the details of our artificial data set generation, followed with a summary of the results.

We first address the issue of evaluating the classification success, which can be measured in many different techniques. We use 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:

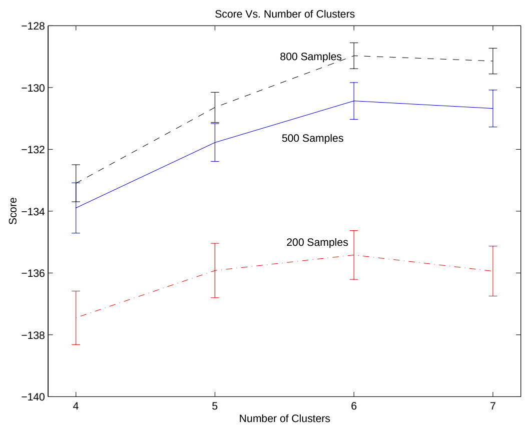
$$P(C_t, C_e) = \sum_{\mathbf{x}} P(\mathbf{x} \mid \mathcal{M}_t) P(C_t \mid \mathbf{x}, \mathcal{M}_t) P(C_e \mid \mathbf{x}, \mathcal{M}_e)$$

Table 1: Summary of results on synthetic data. The results summarize performance of the procedure on data generated from a model with 5 "true" clusters and additional background noise. We report: the number of clusters learned, logarithm of the likelihood ratio between learned model and "true model" on training data (with noisy samples) and test data (unseen samples, without noise), the information the learned clusters contain about the original clusters (see text), the fraction of edges not recovered (# false negative edges / # edges in the true models), and the fraction of false edges recovered (# false positive edges / # edges in learned model). For each figure of merit, we report the mean value and the standard deviation from results from 10 datasets (see text).

Noise	Score	N	Cluster #		Likelihood (train)		Likelihood (test)		$\frac{I(C_t,C_e)}{H(C_t)}$		$\frac{\#\text{FalseNegatives}}{\#\text{TrueEdges}}$		$\frac{\#\text{FalsePositives}}{\#\text{LearnedEdges}}$	
			Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
10%	BIC	200	6.0	0.00	8078	240.8	-3998	183.4	1.00	0.00	0.0187	0.012	0.1389	0.009
		500	6.0	0.00	20168	417.9	-1302	362.1	1.00	0.00	0.0063	0.005	0.1587	0.007
		800	6.0	0.00	32422	524.0	-673	336.2	1.00	0.00	0.0000	0.000	0.1608	0.006
	CS	200	6.4	0.49	08109	231.7	-4074	265.0	1.00	0.00	0.0083	0.008	0.1545	0.008
		500	6.6	0.49	20186	415.0	-1356	404.2	1.00	0.00	0.0042	0.005	0.1643	0.005
		800	6.6	0.49	32444	520.0	-675	328.8	1.00	0.00	0.0000	0.000	0.1666	0.007
30%	BIC	200	5.6	0.49	20255	234.9	-6304	650.7	0.94	0.07	0.0042	0.005	0.1555	0.004
		500	5.8	0.40	50738	642.4	-2847	1081.7	0.97	0.06	0.0000	0.000	0.1666	0.005
		800	5.8	0.40	81027	1415.7	-2016	1224.7	0.97	0.06	0.0000	0.000	0.1738	0.003
	CS	200	6.2	0.75	20349	209.4	-6187	413.2	0.97	0.06	0.0021	0.004	0.1626	0.006
		500	6.4	0.49	50988	487.6	-2275	126.0	1.00	0.00	0.0000	0.000	0.1695	0.007
		800	6.6	0.49	81397	691.6	-1399	88.9	1.00	0.00	0.0000	0.000	0.1738	0.003



(a)



(b)

Figure 2: Graphs comparing the scores for different cluster numbers. The x-axis denotes the number of clusters, and the y-axis denote the score per sample (logarithm of BIC score divided by number of samples). (a) Comparison of directed search and weights annealing search on training data with 30% noise and 500 training samples. (b) Comparison of weight annealing search on training data with 10% noise, with 200, 500, and 800 training samples. Each point is the average of 5 data sets, and error bars denote one standard deviations.

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*,

$$I(C_t; C_e) = \sum_{c_t, c_e} P(c_t, c_e) \log \frac{P(c_t, c_e)}{P(c_t)P(c_e)}$$

between the two clustering variables (Cover & Thomas 1991). 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 relative to the maximum possible (which is the entropy of C_t as $I(C_t, C_t) = H(C_t)$).

We now turn to the second issue of evaluating the structure learning. We measure several aspects of the learned structure. To evaluate the selected number of clusters, we record both the number of clusters in the model as well as the number of "identified" clusters in the model. These are clusters for which there is at least one training sample that is assigned to it. For the CSI structure evaluation we recorded the number of false positive and false negative edges in the implied graph. Recall that an edge corresponds to an informative attribute in the discussion above.

We generated synthetic data from a model learned from Gasch *et al* dataset we describe below. This model had 5 clusters, 93 continuous variables, and 25 discrete nodes. As described in Section 5.2, this model (as most of the ones we learned from real data) had several characteristics. The continuous attributes were mostly informative (usually about several clusters). On the other hand, most discrete attributes were uninformative and the remaining ones distinguished mostly one cluster. From this model, we sampled 5 training sets of sizes 200, 500, and 800 samples (15 training sets in total), and a test set of 1000 samples.

We expect that biological data sets to contain many samples that do not fit into clusters. Thus, we want to ensure that our procedure is robust to the presence of such "noise". To estimate this robustness, we "injected" additional noise into the training sets. This was done by adding samples, whose values were sampled uniformly from the range of values each attribute had in the "real" samples we already had at hand. These obscure the clustering in the original model. We ran our procedure on the sampled data sets that we obtained by adding 10% or 30% additional "noise" samples to the original training data.

The procedure was initialized with random starting points, and for each training data we searched for the best scoring model with the number of clusters in the range $K = 3, \ldots, 7$. We then chose the model with the best score among these. Table 1 summarizes the average performance over the 5 training sets in each parameter setting (200,500, or 800 samples with 10% or 30% "noise") using the learning procedure with two scoring methods. We briefly summarize the highlights of the results.

Search procedure: We compared the performance of the two variants of the search procedure. The directed approach applies structural EM iterations, and attempt to escape from local maxima by attempting stochastic moves and evaluating each one. The annealed approach applies structural EM iterations where in each iteration, samples are re weighted. In our experiments, we started the annealing procedure with initial temperature (variance of gamma distribution) 2, and each iteration cooled the temperature by a factor of 0.9. In particular, in Figure 2(a) we see that the annealed search procedure clearly outperforms the directed search on a particular setting. This behavior was consistently observed in all settings, and we do not report it here.

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 5/6 clusters; see Figure 2(b). Most runs added another cluster that captured the "noise" samples we added in constructing the training data, and thus, most of the runs pick 6 or slightly more clusters (see Table 1). In general, runs with the BIC score had stronger penalty for additional clusters, which resulted in choosing 6 clusters as the best scoring model more often. Runs with the CS score sometimes added more clusters. Additionally, as one might expect, the number of chosen clusters tends to grow with strong noise and with larger sample size.

Likelihood: As expected, training likelihood is higher than that of the true model. This occur both because the procedure "fits" better the training data, and because of the additional noisy samples in the training data. On the other hand, the learned models are always worse (as expected) on the test data. Additionally, the test data likelihood improves with number of training samples increase, even in noisy data that also has additional noise samples. As expected, models trained with noisier data are somewhat worse than models learned from cleaner data. As a general trend, the training data likelihood of models learned with the CS score are as good as or better than models learned with the BIC scores. This difference is significant mainly in the noisier data sets. The test set performance of both scores is roughly the same when learning with 10% noise (with BIC slightly better) and CS is better in 30% noise.

Structure accuracy: We measured the percentage of additional dependencies in the learned graph G when compared to the true structure (false positives) and missing ones in the learned graph G (false negatives). In general, the procedure (using both BIC and CS scores) tended to have very small ratio of false negatives which diminishes as more training samples are available. This shows the procedure is good on recognizing relevant attributes. On the other hand, the procedure had nontrivial number of false positives dependencies, about 13% - 17 % depending on the sample size, the scoring function, and the percentage of noise. In general, when using the CS score, the procedure has a slightly higher ratio of false positive. Similarly, the presence of higher noise levels, also increased the number of false positive dependencies.

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 94% and above. Runs with 500 samples had various results that depended on the level of noises. For 10% noise we got maximal information gain, while results in the noisier data set got 97% information gain. As with the likelihood of the data, the CS score had slightly better results compared to the BIC score. These results show that the learned clusters were informative about 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 evaluated our procedure on two biological data sets of budding yeast gene expression. The first data set is from Spellman et al. (1998) 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. (2000) who measured expression levels of genes in response to different environmental changes. Gasch *et al* identified a cluster of genes that have "generic" response to stress conditions. In addition, they identified

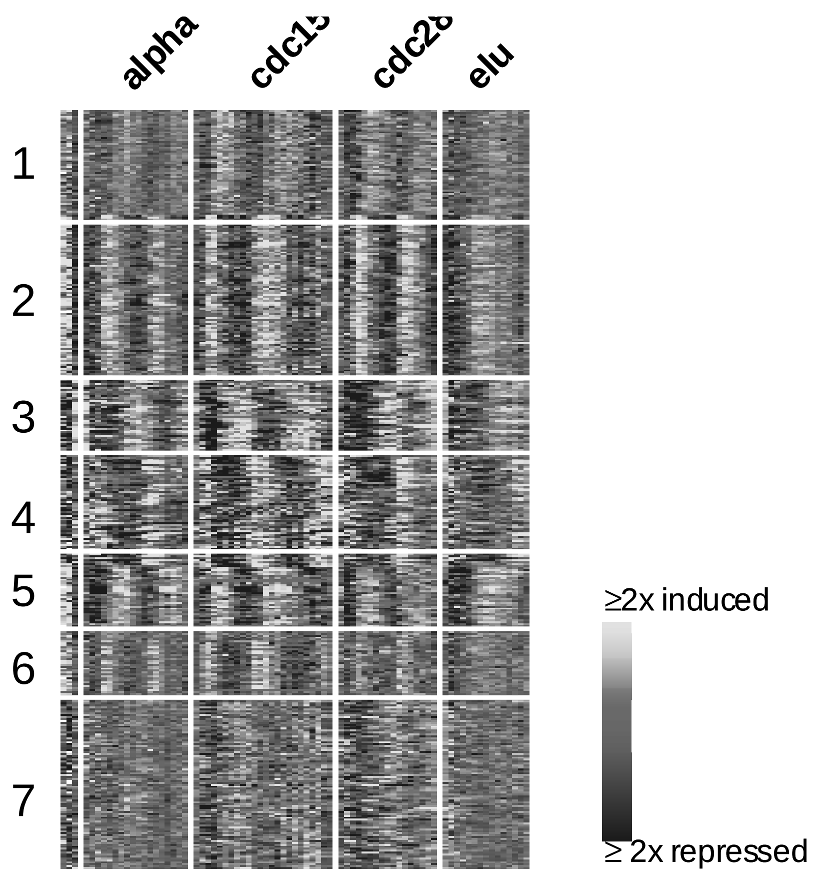


Figure 3: The clustering found for the cell-cycle data of Spellman *et al.*. Light pixels correspond to over expressed genes, and dark ones correspond to under-expressed genes. The clusters shown here, where also characterized by the existence of the following binding sites. Clusters 2 and 5: STUAP (Aspergillus Stunted protein), Cluster 3: QA1 (DNA-binding protein with repressor and activator activities, also involved in silencing at telomeres and silent mating type loci), Clusters 4 and 6: HSF (Heat shock transcription factor).

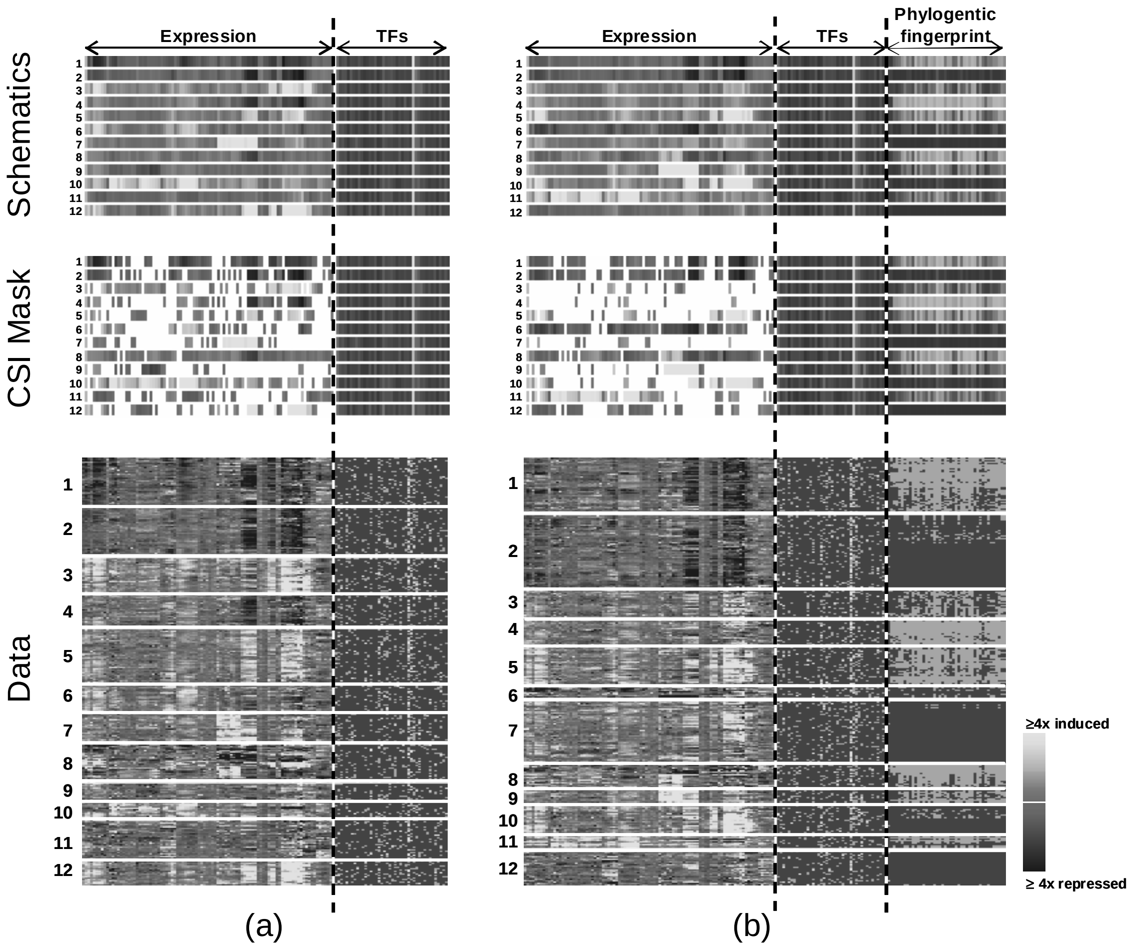


Figure 4: Representation of the clustering found in the stress data of Gasch *et al.* (a) clustering based on gene expression and TF putative binding sites. (b) clustering based also on phylogenetic profiles. The top row contains schematic representation of the clustering. The second row contains a "CSI mask" plot that hides all expression features that were considered uninformative by the model. The bottom row shows figures of all the genes, sorted by cluster identity. The following clusters were also characterized by putative binding sites: Cluster 6(a) and 8(b): GCN4 (Transcription factor of the basic leucine zipper (bZIP) family, regulates general control in response to amino acid or purine starvation) and CBF1, Cluster 2(a) HAP234, Cluster 7(b) GCN4.

clusters of genes that responded to particular stress conditions, but not in a generic manner. Our data set consists of the 950 genes, selected by Segal et al. (2001), that responded to some stress conditions but are not part of the generic stress response. Both data sets are based on cDNA array technology, and the expression value of each gene is reported as the logarithm (base 2) of ratio of expression in the sample compared to the expression of the same gene in a common baseline ("control sample").

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 were generated by using the "fungi" matrices in the TRANSFAC 5.1 database (Wingender et al. 2000, Wingender et al. 2001). We used the MAST program (Bailey & Gribskov 1998) to scan the upstream regions. We used these matches to count the number of putative sites in the 1000bp 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 start by describing the parameters used in the algorithm that were reviewed in previous sections. We applied the annealed search procedure with the following parameters $\sigma_0 = 2, 4$ and $\gamma = 0.5, 0.75, 0.9, 0.95$. Best results were obtained with $\gamma = 0.9, 0.95$ with either σ_0 settings. Other variations, such as the technique for choosing the initial model structure, had no clear cut domination of one technique over the other.

We now discuss the results, they also appear (with full data file and description of the clusters) in `www.cs.huji.ac.il/compbio/ClusterCSI`.

There were several common trends in the results on both expression data sets, when used with MAST TF binding sites. First, the expression measurements were considered informative by the clustering. Most of the expression variables had impact on many 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. In any case this means these attributes had relatively small influence on the clustering results and only the ones that seem to be correlated with a clear gene expression profile of one of the clusters were chosen by the model.

To illustrate the type of clusters we found, we show in Figures 3 and 4(a) 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 each cluster's experiments. Note the schematic illustration of the "masks" that denote the expression attributes that characterize each cluster. As we can see, these capture, quite well, the experiments in which genes in the cluster deviate from the average expression.

Another clear observation is that 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.

In the Gasch *et al* data, the best scoring models had twelve clusters. In the model shown in Figure 4(a), we see two clusters of genes that are under-expressed in stress conditions (Clusters 1, and 2), seven clusters of genes that are over expressed in these conditions (Clusters 3, 5, 6, 7, 8, 10, and 12), two cluster of genes that are over expressed in some stress conditions and under-expressed in others (Cluster 4 and 9), and two cluster of genes with undetermined response to stress conditions (Cluster 8, and 11). These later clusters have high variance, while the others have relatively tight variance in most experiments.

Some of the clusters correspond to clear biological function: For example, Cluster 7 , contains genes that are over-expressed in amino-acid starvation and nitrogen depletion. Examining the MIPS (Mewes et al. 1999) functional annotation of these genes suggests that many of the genes involved amino-acid biosynthesis and in transport. Another example is Cluster 2 that contains genes that are under-expressed in late stages of nitrogen depletion, diauxic shift, and under YPD growth medium. This cluster is associated with frequent occurrences of the HAP234 binding site. This binding site (of the complex HAP2, HAP-3, and HAP-4) is associated with the control of gene expression under nonfermentative growth conditions. Many genes in this cluster are associated with mitochondrial organization and transport, respiration, and ATP transport. The association of the cluster with the HAP234 binding strengthens the hypothesis that genes with unknown function in this cluster might be related to these pathways.

We suspect that one of the reasons few clusters are associated with transcription factors binding site is the noisy prediction of these sites. To evaluate the effect of a more informative sequence motifs identification in the upstream region, we performed the following experiment. We applied our algorithm using expression values from the Gasch *et al* data set. Then, we applied the procedure of Barash *et al* (2001) to each of the clusters we identified. This procedure searches for motifs that discriminatively appear in the upstream region of genes in particular clusters and are uncommon in other genes in the genome. We then annotated each gene with the set of motifs we found, and used these annotations as additional input to a new run of our algorithm. Although we applied a fairly simple unsupervised sequence motif identification algorithm, its impact on the learning algorithm was clear. Several hundred genes have changed their hard assignment from the initial assignment made when clustering with only expression data, 26 out of 28 motifs were considered informative to the final clustering , and 2 motifs became relevant for 2 different clusters. When we ran the new hard assignments of genes to clusters in the motif finding algorithm we got a general improvement in motifs identification in clusters.

Next, in order to demonstrate the model's ability to facilitate relevant biological data from different sources, we considered adding additional attributes extracted from the COG database (Tatusov et al. 2001). This database associates each yeast gene with orthologous genes in 43 other genomes. Thus, we create for each gene a *phylogenetic pattern* that denotes whether there is an orthologous gene in each of the 43 genomes. When we include these additional features, the clusters learned changes. In general, we note that most of the phylogenetic patterns were considered informative by the model but still context specific. For example, we see pairs of clusters (e.g., Clusters 5 and 10) that are similar in terms of expression, yet have distinct phylogenetic profiles. One cluster contains genes that do not have orthologs, while the other cluster contains genes that have orthologs in many bacterial genomes.

Phylogentic patterns also allow us to gain additional insight into the functional aspects of the clusters. For example Cluster 8 contains genes that are highly over-expressed in amino-acid starvation and nitrogen depletion. It is characterized by occurrences of the binding sites of GCN4 and CBF1, and genes in it have the "typical" profile with orthologs in *C. jejuni, P. mutocide, Halobacterium sp. NRC-1, P. aeruginosa, M. tuberculosis, A. aeolicus, C. crescentus*, H. pylori J99, M. leprae, *D. radiodurans, T. volcanium,* and *T. acidophilum,* and no orthologs in *M. genitalium, B. burgdorferi, C. pneumoniae, C. trachomatis, S .pyogenes,* T. pallidum, *R. prowazekii,,*

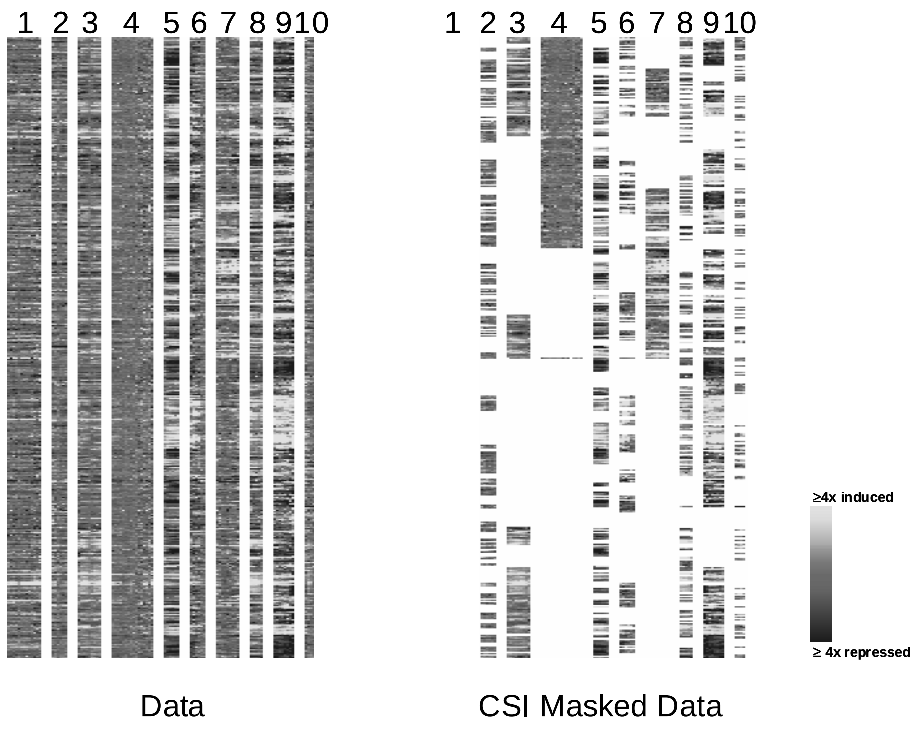


Figure 5: Clustering of arrays in the stress data set of Gasch *et al.* The left figure shows the data rearranged according to the clustering. The right figure shows only the positions that are informative the learned models (note that cluster 1 is totally masked in this model).

U. urealyticum, and *Buchnera sp. APS*. This cluster description suggests that this group of genes have common phylogenetic origins as well as common function and regulation.

As we noted in the introduction, our method can be used for other clustering tasks. As an example, we clustered the 92 samples in the stress data. In this clustering, we reversed the roles of conditions and genes. Now we consider each condition as an (independent) sample, and each gene as a (continuous) attribute of the sample. The result of the clustering are groups of samples, for each cluster we have the list of informative genes. Not surprisingly, this clustering recovered quite well the groups of samples with the same treatments. Table 2 shows the composition of each cluster in a run with 10 clusters in terms of the original treatments. Each of the following treatments were recovered in a separate cluster: ddt, diamide, YP, and steady state. In addition, the nitrogen depletion time course was split into two clusters. The earlier samples (30 minutes to 4 hours) appeared in a cluster with the amino acid starvation samples, while the later samples (8 hours to 5 days) were clustered separately. This is consistent with clusters we learned over genes, that showed that some genes had distinct behavior in later parts of the nitrogen depletion time course. Similar phenomena occurs with H2O2 samples. Earlier samples (10 minutes - 50 minutes) are clusters with Menadion samples. Later H2O2 samples (60 minutes to 80 minutes, and also 40 minutes) were clustered with sorbitol samples. Finally, both heat shock time courses (fixed temperature, and variable temperature) were mostly clustered in one cluster, although some of the heat shock samples

Table 2: Confusion matrix comparing the learned clusters of experiments in the Gasch *et al* data set (in a one run of the procedure) to the original division of experiments according to treatment.

Condition	Cluster									
	1	2	3	4	5	6	7	8	9	10
Heatshock				1						
Heatshock (variable)			1	1				5		
H2O2	4			5						
Menadione	9									
DDT										4
diamide			8							
sorbitol				7						
AA starvation							5			
Nitrogen starvation					6		4			
Diauxic shift						1				
YPD				2					8	
YP						5				
Steady state		6								

appear in other clusters.

These results demonstrate that, as can be expected, different treatments have a clear signature at the mRNA level that can be easily picked by our algorithm. More important, these results demonstrate another possibly important application of the CSI clustering algorithm. As we cluster here experiments over gene as attributes this procedure not only cluster similar experiments but also automatically extracts for each such experiments cluster the genes that are differently expressed in it.

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 learns from both gene expression data and putative binding sites (from various sources). This approach identifies and characterizes clusters of genes. Our approach is novel in its ability to handle data in which many attributes 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 experimental results on both synthetic and real data, suggest that this approach is robust to noise and recovers groups of genes with coherent behavior.

Throughout most of the paper we have focused one a particular application—clustering genes based on expression levels and transcription factors binding sites. However, it is clear that the general techniques we develop here are applicable for many other forms of data analysis where one expect to find many irrelevant and "partially relevant" attributes.

Our approach is similar to that of Holmes and Bruno (2000) 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 (Huges et al. 2000, Lawrence et al. 1993)). 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 (Friedman et al. 1997).

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. Recently, Segal et al. (2001) develop a method in this spirit that is based on the language of *Probabilistic Relational Models* (Friedman et al. 1999).

Acknowledgments

The authors thank Gill Bejerano, Yonatan Bilu, Gal Elidan, Tommy Kaplan, Michal Linial, Dana Pe'er, Eran Segal, and Zohar Yakhini 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.

References

Bailey, T. L. & Gribskov, M. (1998), 'Combining evidence using p-values: application to sequence homology searches', *Bioinformatics* **14**, 48–54.

Barash, Y., Bejerano, G. & Friedman, N. (2001), A simple hyper-geometric approach for discovering putative transcription factor binding sites, *in* O. Gascuel & B. M. E. Moret, eds, 'Algorithms in Bioinformatics: Proc. First International Workshop', number 2149 *in* 'LNCS', pp. 278–293.

Bittner, M., Meltzer, P. & Trent, J. (1999), ‘Data analysis and integration: of steps and arrows’, *Nature Genetics* **22**, 213–215.

Boutilier, C., Friedman, N., Goldszmidt, M. & Koller, D. (1996), Context-specific independence in Bayesian networks, *in* E. Horvitz & F. Jensen, eds, ‘Proc. Twelfth Conference on Uncertainty in Artificial Intelligence (UAI ’96)’, Morgan Kaufmann, San Francisco, pp. 115–123.

Brazma, A. & Vilo, J. (2000), ‘Gene expression data analysis’, *FEBS Lett* **480**(1), 17–24.

Cheeseman, P. & Stutz, J. (1995), Bayesian classification (AutoClass): Theory and results, *in* F. U., P.-S. G., S. P. & U. R., eds, ‘Advances in Knowledge Discovery and Data Mining’, AAAI Press, Menlo Park, CA, pp. 153–180.

Chickering, D. M. & Heckerman, D. (1997), ‘Efficient approximations for the marginal likelihood of bayesian networks with hidden variables’, *Machine Learning* **29**, 181–212.

Chickering, D. M., Heckerman, D. & Meek, C. (1997), A Bayesian approach to learning Bayesian networks with local structure, *in* D. Geiger & P. Shanoy, eds, ‘Proc. Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI ’97)’, Morgan Kaufmann, San Francisco, pp. 80–89.

Cover, T. M. & Thomas, J. A. (1991), *Elements of Information Theory*, John Wiley & Sons, New York.

DeGroot, M. H. (1970), *Optimal Statistical Decisions*, McGraw-Hill, New York.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977), ‘Maximum likelihood from incomplete data via the EM algorithm’, *Journal of the Royal Statistical Society* **B 39**, 1–39.

Elidan, G., Ninio, M., Lotner, N. & Friedman, N. (2001), Perturbed search strategies for escaping local maxima when learning graphical models. submitted.

Friedman, N. (1997), Learning belief networks in the presence of missing values and hidden variables, *in* D. Fisher, ed., ‘Proceedings of the Fourteenth International Conference on Machine Learning’, Morgan Kaufmann, San Francisco, pp. 125–133.

Friedman, N. (1998), The Bayesian structural EM algorithm, *in* G. F. Cooper & S. Moral, eds, ‘Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI ’98)’, Morgan Kaufmann, San Francisco.

Friedman, N., Geiger, D. & Goldszmidt, M. (1997), ‘Bayesian network classifiers’, *Machine Learning* **29**, 131–163.

Friedman, N., Getoor, L., Koller, D. & Pfeffer, A. (1999), Learning probabilistic relational models, *in* ‘Proc. Sixteenth International Joint Conference on Artificial Intelligence (IJCAI ’99)’, Morgan Kaufmann, San Francisco.

Friedman, N. & Goldszmidt, M. (1998), Learning Bayesian networks with local structure, *in* M. I. Jordan, ed., ‘Learning in Graphical Models’, Kluwer, Dordrecht, Netherlands, pp. 421–460. An earlier version appeared in *Proc. 12th Conf. Uncertainty in Artificial Intelligence*, 1996.

Friedman, N., Ninio, M., Peer, I. & Pupko, T. (2002), 'A structural EM algorithm for phylogentic inference', *J. Comp. Bio.* . To appear. A preliminary version of this paper appeared in *Fifth Annual International Conference on Computational Molecular Biology*, 2001.

Gasch, A. P., Spellman, P. T., Kao, C. M., Carmel-Harel, O., Eisen, M. B., Storz, G., Botstein, D. & Brown, P. O. (2000), 'Genomic expression program in the response of yeast cells to environmental changes', *Mol. Bio. Cell* **11**, 4241–4257.

Heckerman, D. (1998), A tutorial on learning with Bayesian networks, *in* M. I. Jordan, ed., 'Learning in Graphical Models', Kluwer, Dordrecht, Netherlands.

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

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

Iyer, V., Eisen, M., Ross, D., Schuler, G., Moore, T., Lee, J., Trent, J., Staudt, L., Hudson, J., Boguski, M., Lashkari, D., Shalon, D., Botstein, D. & Brown, P. (1999), 'The transcriptional program in the response of human fibroblasts to serum', *Science* **283**, 83–87.

Langley, P. & Sage, S. (1994), Induction of selective Bayesian classifiers, *in* R. López de Mantarás & D. Poole, eds, 'Proc. Tenth Conference on Uncertainty in Artificial Intelligence (UAI '94)', Morgan Kaufmann, San Francisco, pp. 399–406.

Lauritzen, S. L. (1995), 'The EM algorithm for graphical association models with missing data', *Computational Statistics and Data Analysis* **19**, 191–201.

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

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

Pearl, J. (1988), *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann, San Francisco, Calif.

Rissanen, J. (1978), 'Modeling by shortest data description', *Automatica* **14**, 465–471.

Roth, F., Hughes, J.D. Estep, P. & Church, G. (1998), 'Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation', *Nat. Biotechnol.* **16**, 939–945.

Schwarz, G. (1978), 'Estimating the dimension of a model', *Annals of Statistics* **6**, 461–464.

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

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D. & Futcher, B. (1998), ‘Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization’, *Mol. Biol. Cell* **9**(12), 3273–97.

Tatusov, R., Natale, D., Garkavtsev, I., Tatusova, T., Shankavaram, U., Rao, B., Kiryutin, B., Galperin, M., Fedorova, N. & Koonin, E. (2001), ‘The COG database: new developments in phylogenetic classification of proteins from complete genomes’, *Nuc. Acids Res.* **29**, 22–28.

Tavazoie, S., Hughes, J. D., Campbell, M. J., Cho, R. J. & Church, G. M. (1999), ‘Systematic determination of genetic network architecture’, *Nat Genet* **22**(3), 281–5. Comment in: Nat Genet 1999 Jul;22(3):213-5.

Vilo, J., Brazma, A., Jonassen, I., Robinson, A. & Ukkonen, E. (2000), Mining for putative regulatory elements in the yeast genome using gene expression data, *in* ‘ISMB’00’.

Wingender, E., Chen, X., E., F., Geffers, R., Hehl, R., Liebich, I., Krull, M., Matys, V., Michael, H., Ohnhauser, R., Pruss, M., Schacherer, F., Thiele, S. & Urbach, S. (2001), ‘The TRANSFAC system on gene expression regulation’, *Nuc. Acids Res.* **29**, 281–283.

Wingender, E., Chen, X., Hehl, R., Karas, H., Liebich, I., Matys, V., Meinhardt, T., Pruss, M., Reuter, I. & Schacherer, F. (2000), ‘TRANSFAC: an integrated system for gene expression regulation’, *Nuc. Acids Res.* **28**, 316–319.

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 (DeGroot 1970).

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 M independent draws $x[1], \ldots, x[M]$ of X from an unknown multinomial distribution P^*. The *likelihood* of the observations for given parameters is:

$$P(x[1], \ldots, x[M] \mid \vec{\theta}_X) = \prod_i \theta_i^{M_i}$$

where M_i is the number of occurrences of the symbol i in the sequence $x[1], \ldots, x[M]$. The vector $\mathcal{S} = \langle M_1, \ldots, M_K \rangle$ is the *sufficient statistics* of the sequence $x[1], \ldots, x[M]$. 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 $x[M+1]$ given $x[1], \ldots, x[M]$. Given a prior distribution over the possible multinomial distributions, the Bayesian estimate is:

$$P(x[M+1] \mid x[1], \ldots, x[M]) = \int P(x[M+1] \mid \vec{\theta}) P(\vec{\theta} \mid x[1], \ldots, x[M]) d\vec{\theta} \qquad (9)$$

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

$$P(\vec{\theta} \mid x[1], \ldots, x[M]) \propto P(\vec{\theta}) \prod_i \theta_i^{M_i} \qquad (10)$$

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 $\alpha_1 + M_1, \ldots, \alpha_K + M_K$. Thus, we get that the prediction for X^{M+1} is

$$P(X_{M+1} = i \mid x[1], \ldots, x[M]) = \frac{(\alpha_i + M_i)}{\sum_j (\alpha_j + M_j)}$$

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:

$$P(x[1], \ldots, x[M]) = \int P(x[1], \ldots, x[M] \mid \vec{\theta}) P(\vec{\theta}) d\vec{\theta} = \frac{\Gamma(\sum_i \alpha_i)}{\Gamma(\sum_i \alpha_i + M_i)} \prod_i \frac{\Gamma(\alpha_i + M_i)}{\Gamma(\alpha_i)}$$

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

$$P(x[1], \ldots, x[M] \mid \mu, \tau) = \prod_i \sqrt{\frac{\tau}{2\pi}} \exp\left\{ -\frac{1}{2}\tau(x[i] - \mu_i)^2 \right\}$$

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

$$P(x[1], \ldots, x[M] \mid \mu, \tau) = \exp\left\{ M\frac{1}{2}(\log\tau - \tau\mu^2) + T_1 \frac{1}{2}\tau\mu - T_2 \frac{1}{2}\tau - M\frac{1}{2}\log(2\pi) \right\}$$

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 $x[1], \ldots, x[M]$ with sufficient statistics $\langle M, T_1, T_2 \rangle$ has hyper-parameters $\mu_0', \lambda', \alpha', \beta'$, where $\mu_0' = \frac{\lambda\mu_0 + MT_1}{\lambda + M}$, $\lambda' = \lambda + M$, $\alpha' = \alpha + \frac{1}{2}M$, and $\beta' = \beta + \frac{1}{2}M(T_2 - T_1^2) + \frac{M\lambda(T_1 - \mu_0)^2}{2(\lambda + M)}$. Finally, the marginal likelihood is

$$P(x[1], \ldots, x[M]) = \int P(x[1], \ldots, x[M] \mid \vec{\theta}) P(\vec{\theta}) d\vec{\theta} = (2\pi)^{-\frac{1}{2}M} \left(\frac{\lambda}{\lambda'}\right)^{\frac{1}{2}} \frac{\Gamma(\alpha')}{\Gamma(\alpha)} \beta^{\alpha} \beta'^{-\beta}$$

where α', β', and λ' are defined as above.