Modeling Dependencies in Protein-DNA Binding Sites

Yoseph Barash[1], Gal Elidan[1], Nir Friedman[1]*, Tommy Kaplan[1,2]
[1]School of Computer Science & Engineering, Hebrew University, Jerusalem, 91904 Israel
[2]Dept. of Molecular Genetics & Biotechnology, The Hebrew University, Hadassah Medical School, Jerusalem, 91120 Israel
{hoan,galel,nir,tommy}@cs.huji.ac.il

ABSTRACT

The availability of whole genome sequences and high-throughput genomic assays opens the door for *in silico* analysis of transcription regulation. This includes methods for discovering and characterizing the binding sites of DNA-binding proteins, such as transcription factors. A common representation of transcription factor binding sites is a *position specific score matrix* (PSSM). This representation makes the strong assumption that binding site positions are independent of each other. In this work, we explore *Bayesian network* representations of binding sites that provide different tradeoffs between complexity (number of parameters) and the richness of dependencies between positions. We develop the formal machinery for learning such models from data and for estimating the statistical significance of putative binding sites. We then evaluate the ramifications of these richer representations in characterizing binding site motifs and predicting their genomic locations. We show that these richer representations improve over the PSSM model in both tasks.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models—*statistical, structural*; J.3 [**Computer Applications**]: Life and Medical Sciences—*biology and genetics*

General Terms

Algorithms

Keywords

Bayesian networks, DNA sequence motifs, transcription factors binding sites

1. INTRODUCTION

A key issue in modern molecular biology is understanding the mechanisms of transcriptional regulation. Many aspects of transcription regulation involve DNA-binding proteins, called *transcription factors*. These factors modulate the expression of genes by binding to specific positions in nearby genomic regions. Transcription factors bind to specific DNA subsequences that can be pinpointed by biological assays [27]. Indeed, the TRANSFAC database [41] contains hundreds of biologically validated binding sites. Such assays, however, are labor intensive and cannot identify all the binding sites of a transcription factor.

The recent availability of complete genomic sequences (including intergenic regions) motivates attempts to understand the regulatory mechanisms through *in silico* analysis. Binding site identification involves two main tasks. The first task is to predict potential binding sites of a known transcription factor on a genomic scale. Here one uses examples of biologically verified binding sites and aims to find similar sites in other intergenic regions such as gene promoter regions [2, 34]. The second task is to discover a sequence motif as well as its putative sites in a collection of relatively long intergenic sequences that are suspected of being bound by the same factor. An example of such a task is examining the promoter regions of a set of genes that have common functional annotation or are co-expressed. In this case, the discovered motif indicates a possibly unknown factor that regulates the set of genes. Many works in recent years have proposed different schemes to handle this task [3, 4, 24, 31, 39, 40].

Both tasks require us to describe a *motif* that characterizes sequences that appear at binding sites of the transcription factor. The biological literature suggests that the relevant sequences are relatively short (up to 20bp long). Moreover, although binding sites are quite conserved, they do show some variability. The sequence motif has to represent multiple allowed (or preferred) subsequences at the binding site. A commonly used representation of such motifs is a *position specific score matrix* (PSSM). A PSSM records the preference for each potential DNA nucleotide at each binding site position. This representation inherently assumes that positions within the motif are independent of each other.

It is an open question whether this strong independence assumption is reasonable. Recent results indicate that in specific cases, there might be dependence between positions (e.g., [1, 7, 9]). In this paper, we take a pragmatic approach to this issue. We aim to test whether modeling dependencies leads to better performance in the computational tasks of binding site annotation and motif discovery. If the result is positive, it suggests that there are dependencies between positions, at least for some transcription factors.

The main technical question we face is how to relax the assumption of independence. For this purpose we need to model the joint distribution of all positions. A naive representation requires a large number of parameters (exponential in the motif length). The full independence model (a PSSM) and the full dependence model (an exponentially large joint distribution) represent the two ends of a spectrum. The choice of representation leads to a tradeoff: Less expressive models cannot represent complex dependencies, but have succinct representation and can be learned robustly from few examples. More expressive models can represent complex dependencies, but involve many parameters and require a larger number of examples for learning.

In this paper we examine models that explore the middle ground of this spectrum. These models are based on the language of *Bayesian Networks* [33]. We describe procedures that learn such representa-

*Contact author: nir@cs.huji.ac.il

RECOMB'03, April 10–13, 2003, Berlin, Germany.

tions from data, and allow us to carry out the two tasks described above. We first demonstrate how our flexible set of models generalizes better than the PSSM model on biologically verified sites. We then show that the models learned are more precise in predicting putative binding sites (in the sense of achieving a better false positives vs. false negatives tradeoff) using genome-wide *S. cerevisiae* localization assays [29].

2. MODELING BINDING SITE MOTIFS

We now consider how to model a sequence motif representing the binding sites of a transcription factor. We want to represent the commonalities among different binding sites. One way of going about this is to represent a probability distribution over sequences that will assign high probability to sequences that are likely to be found at the binding site. We assume that binding sites are of length K. We want to represent a probability distribution over all 4^K possible K-mers that can appear at the binding site. Formally, we need a distribution $P(X_1, \ldots, X_K)$ where X_i is a *random variable* that represents the nucleotides in the i'th position of the K-mer. A naive representation of such a distribution simply lists the probabilities of all 4^K assignments. This representation is clearly wasteful and is also unrealistic to estimate from data. Thus, we are interested in more succinct ways of representing such distributions. We will now review the PSSM representation and several richer models for this task.

Position Specific Score Matrix (PSSM) A common way of representing a binding site is to assume that the nucleotide at one position is independent of the nucleotides at all other positions. This assumption implies that

$$P(X_1 \ldots X_K) = \prod_{i=1}^{K} P(X_i)$$

Where $P(X_i)$ is the *marginal probability* of each nucleotide X_i in the distribution and is a shorthand for probability events of the form $P(X_i = A)$. We call distributions that have this form *PSSM models*.[1]

A PSSM model requires $3K$ parameters to describe the marginal distribution of nucleotides at each position. A PSSM can capture specific preferences at each position and also different levels of specificity in positions.

Mixture of PSSMs Suppose that the transcription factor can have several "types" of binding. These might correspond to slightly different physical configurations of the protein at the binding site, each with somewhat different preferences. Thus, it can bind to any sequence that fits any one of the configurations. To model this case, we assume there is an additional, unobserved, random variable T that describes the type of the binding. We have some prior probability $P(T)$ (possibly uniform) over the type of binding, and we assume that for each type the positions are independent, just as in the PSSM case. This requires a distribution $P(X_i \mid t)$ over the nucleotide at the i'th position given the value t of T. The joint probability of the observed positions require that we sum over all possible values of T

$$P(X_1 \ldots X_K) = \sum_{t=1}^{C} \left[P(t) \prod_{i=1}^{K} P(X_i \mid t) \right]$$

That is, the distribution is a mixture of PSSM representations where the mixing probabilities are determined by $P(T)$.

[1]The term PSSM typically refers to a matrix of log-odd score $\log \frac{P(X_i)}{P_0(X_i)}$, where $P_0(X_i)$ is a background distribution. Our definition of a PSSM is equivalent when combined with $P_0(X_i)$.

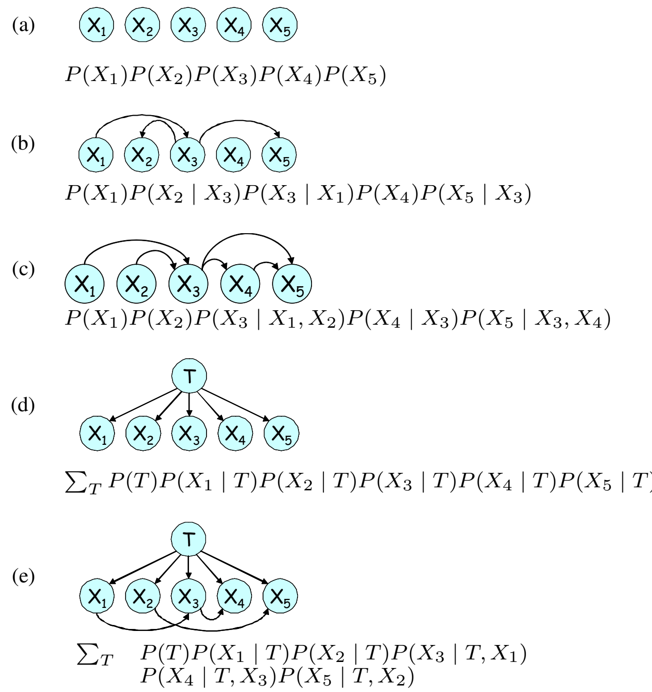


Figure 1: Examples of different Bayesian network models for a motif with 5 positions. For each model, we show an example of a Bayesian network structure, and the corresponding representation of the joint distribution. (a) PSSM, (b) tree, (c) non-tree, (d) mixture of PSSMs, and (e) mixture of trees.

A mixture model has several benefits both in terms of representation and in terms of semantic interpretation. First, the number of parameters is fairly small: $C - 1$ parameters for $P(T)$, and $3KC$ parameters for the conditional probabilities $P(X_i \mid T)$. Second, as suggested above, this model offers an important representational concept. Although the nucleotides are dependent when T is unknown, they are independent when T is observed. Thus, T plays the role of a hidden biological mechanism the renders the positions independent. Understanding the interaction between the hidden mechanism T and the nucleotides at each position can provide insights about the physical protein-DNA interactions.

Bayesian Networks Mixtures of PSSMs capture "broad" dependencies among all the positions via the T variable. An alternative approach to describe dependencies is to consider how each position depends on the others. For example, a particular nucleotide in position 1 might cause the conformation of a particular amino acid side-chain to change. This, in turn, affects the conformation of other amino acids in the binding site, and may have an effect on the preference of binding in position 3.

One representation designed to capture "local" dependencies is the language of *Bayesian Networks*. In this representation, we use a *directed acyclic graph* G to represent the dependencies. The vertices of G correspond to the random variables $X_1 \ldots X_K$ and a parameterization which describes a conditional distribution for each variable given its immediate parents in G. The corresponding joint probability distribution decomposes into the product form:

$$P(X_1 \ldots X_K) = \prod_{i=1}^{K} P(X_i \mid \mathbf{Pa}_i^G) \tag{1}$$

Where $\mathbf{Pa}_i^G$ is the (possibly empty) set of parents of X_i in G. Figure 1 (a)–(c) show few examples of Bayesian networks, and their associated form of the probability distribution.

The formal semantics of Bayesian Networks is in terms of conditional independence statements: Each position X_i is independent of it's non-descendants in G given its parents in G [33]. In general, the more edges we have in G, the more complex the dependencies between positions are. The simplest network has no edges, like the one in Figure 1(a). It is easy to see that in this case, the distribution is simply a PSSM. For the general case, the number of parameters in the networks depends on the number of edges: The conditional distribution $P(X_i \mid \mathbf{Pa}_i^G)$ requires $3 \cdot 4^{|\mathbf{Pa}_i^G|}$ parameters.

Tree Bayesian Networks One sub-class of Bayesian network that we want to single out is the class of *tree* Bayesian networks. In tree models each position has at most one parent, making G a forest. For example, the networks in Figure 1(a) and (b) are both tree networks. Tree networks also generalize first-order Markov chains (where G is $X_1 \rightarrow X_2 \rightarrow \ldots \rightarrow X_K$). They provide a flexible language for modeling dependencies, while limiting the number of parameters to be at most $3 \cdot 4K$. Another important benefit of this class of models is that there are efficient algorithms to learn the best tree structure [12, 17].

Mixture of Trees In some cases, a tree structured network might be too limited. One possible approach of enriching the representation is to combine the benefits of a tree structure with the added richness of a hidden mechanism. This leads to a natural extension, similarly to mixture of PSSMs, that is a mixture of trees . Each X_i has as parents the hidden variable T and at most one other nucleotide. The network of Figure 1(e) is an example of this model. Intuitively, the unobserved variable T enhances the ability of the tree to model additional dependencies while multiplying the number of parameters only by a factor of C. An important advantage of mixture of trees is that, similarly to trees, there exist efficient algorithms for learning the best structure [17, 32].

3. LEARNING MOTIF MODELS

3.1 The Learning Setup

Suppose we want to learn motif models from data. We assume that our input is a set of *aligned* binding sites of the transcription factor. Our task is to learn a probabilistic model that captures the common features of these sequences. This is an instance of the well studied problem of learning Bayesian networks from data . We sketch the main issues without going into details. The interested reader can find more details in [6, 16, 17, 22].

We assume we have a training dataset $\mathcal{D}$ of M aligned binding sites. We denote by $x_i[m]$ the value of X_i at the m'th example. To clarify the discussion, it is conceptually easier to think of the input to the learning problem in terms of the *empirical probability* $\hat{P}$ that measures the frequency of events in the training examples. We do so by using the distribution:

$$\hat{P}(x_1, \ldots, x_k) = \frac{1}{M} \sum_{m}^{M} \mathbf{1}\left\{x_1[m] = x_1, \ldots, x_K[m] = x_K\right\}$$

where $\mathbf{1}\{\}$ is the indicator function that has the value 1 if the condition in its argument is true and 0 otherwise. While we do not represent this distribution explicitly, it will be convenient to refer to the marginal probabilities in this distribution in the discussion below. It is easy to compute such marginal distribution from the input examples.

Parameter Learning A key component in learning is assigning parameters to the conditional distribution. In some models, such as a PSSM , this is the only part we need to learn. In other models, we also learn the structure G. However, for each structure we consider, we also need to estimate its parameters.

Our task is to find the parameters θ that maximize the average (log)-probability of each sample over the data. That is, we want to maximize the (log)-likelihood function

$$\ell(G, \theta : \mathcal{D}) = \boldsymbol{E}_{\hat{P}}[\log P(X_1, \ldots, X_K \mid G, \theta)] \qquad (2)$$

In models where we do not have a hidden variable T, learning parameters is straightforward. It turns out that the *maximum likelihood* parameters for $P(X_i \mid \mathbf{Pa}_i^G)$ are simply the matching conditional distributions in $\hat{P}$. That is, we set $P(X_i \mid \mathbf{Pa}_i^G) = \hat{P}(X_i \mid \mathbf{Pa}_i^G)$. Thus, parameter learning in this case reduces to estimating marginal probability from $\hat{P}$. Since we usually have a small number of training examples, we *smooth* the maximum likelihood estimates by using *Dirichlet priors* [23]. This amounts to adding a small number (5 in our experiments) of *pseudo instances* that are distributed according to a background distribution.

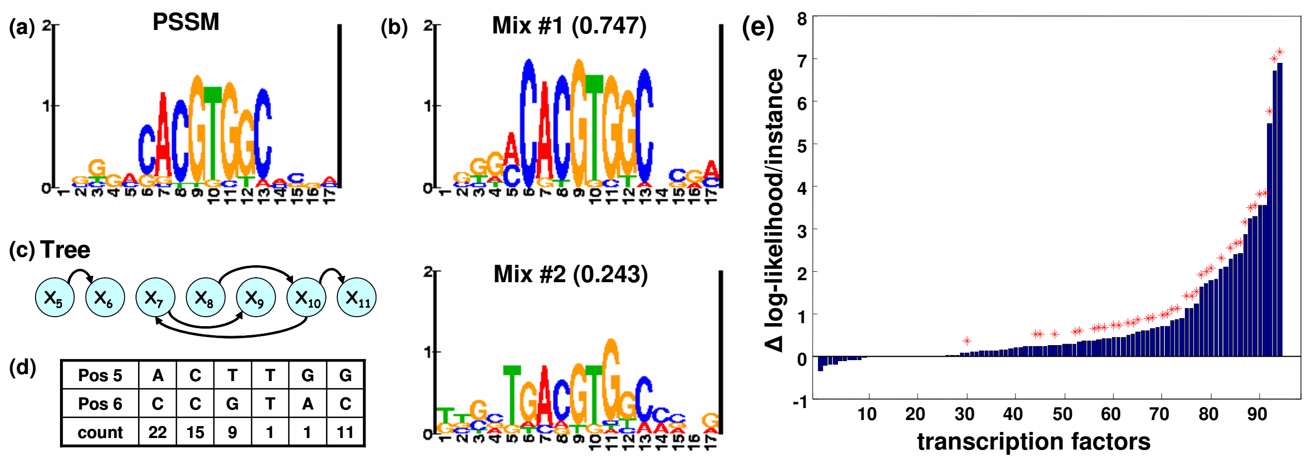
In models where we have a hidden variable T, parameter estimation is somewhat more complex. We need to perform an iterative procedure of *Expectation Maximization* [13, 28] to find a (local) maximum of the likelihood function.

Structure Learning In addition to estimating parameters, we might also want to learn the dependency structure G, i.e., which edges to include. When performing structure learning we need to take into account that richer models (ones with more edges) can achieve higher likelihoods. This runs the risk of learning a model that seems good on the training data but performs badly on new instances. Thus, instead of maximizing the likelihood function, we attempt to maximize a statistical score based on Bayesian considerations [22, 23]. This score can be thought of as likelihood penalized by a term that accounts for differences in model complexity. It is designed to estimate the performance of a model on new unseen instances. For this we use the *BDeu score* of [23]. In models that do not have a hidden variable T, finding the best graph is a combinatorial optimization problem. For tree networks, this problem can be reduced to a maximum weighted forest problem and solved efficiently [12, 17]. For general Bayesian networks, this problem is intractable, and we resort to using a heuristic search.

In models where a hidden variable T is present, the situation is more complex. First, we also need to decide on the cardinality of T. Moreover, the structure score becomes intractable and we need to resort to an approximation. We use the the Cheeseman-Stutz approximation of the BDeu score [11]. To choose the cardinality of G, we evaluate the score at each cardinality. For a mixture of trees model where we also face the problem of structure learning, we can use, similarly to trees, an efficient maximum weighted forest algorithm [17, 32].

3.2 Experimental Evaluation

To evaluate the extent to which the richer models we described are beneficial in representing transcription factors binding sites, we performed the following experiment: By extracting datasets of aligned binding sites from the TRANSFAC database [41], we examined 95 transcription factors for which there were 20 or more sites. For each transcription factor, we evaluated the ability of each of the representations to describe the distribution of sequences at the binding site of that factor. To get an objective evaluation, we performed *10-fold cross validation* tests on each dataset. In such a test we learn a model (i.e., select structure and parameters) on 90% of the data and evaluate it on the remaining 10% of the instances. We repeated this learning step 10 times, so that each instance is evaluated exactly once as a test case. The performance of a representation on a dataset is summarized by the average log-probability



Pos 5	A	C	T	T	G	G
Pos 6	C	C	G	T	A	C
count	22	15	9	1	1	11

Figure 2: Comparison of dependency models and PSSM, learned on aligned binding sites from the TRANSFAC database [41]. (a)–(d) One example of evident dependency in the bindings sites for TRANSFAC `P$ABF_Q2` motifs. (a) PSSM model; (b) mixture of PSSMs model (2 components); (c) the dependency structure tree model (only middle positions are shown); (d) the occurrence table for positions 5 and 6; (e) difference of average log-likelihood per instance on test data between the best dependency model and the PSSM model (y-axis) for 95 transcription factors (x-axis). Asterisks mark statistically significant results (paired t-test with $p < 0.05$).

per instance when it is evaluated as part of the test set.

As a specific example, consider the sites of TRANSFAC identifier `P$ABF_Q2` (*Arabidopsis* ABA responsive element binding factor). The associated dataset consists of 49 binding sites, each 17bp long. A PSSM learned from this dataset (see Figure 2(a)), shows position 5 as uninformative, and position 6 as weakly informative. When we learn a mixture of PSSMs, we get a mixture between the two PSSMs shown in Figure 2(b). As we can see, these two PSSMs differ mostly in positions 5 and 6. When we learn a tree Bayesian network, we get the dependency structure shown in Figure 2(c). The strongest dependency is between position 5 and 6. We can see this dependency by examining the occurrence table for both positions, Figure 2(d). As we can see, when position 5 is 'T', position 6 is 'G' with high probability, and when position 5 is not 'T', position 6 is 'C' with high probability. In our cross validation test, the three methods achieves log-probability of -19.93, -18.70, -18.47 bits per instance for PSSMs, mixture of PSSMs, and trees, respectively. Qualitatively, we can say that by using a dependency model we were able to detect a real phenomenon in the data that was "smoothed out" by the over simplistic PSSM model.

The results of our evaluation on all 95 datasets appear in the Supplementary Information [5, Table A.1]. Figure 2(e) summarize these results by comparing the best dependency model with the PSSM model. (For a comparison of individual methods, see Supplementary Information [5, Item A.2]). We evaluated the statistical significance of these differences using a paired t-test on log-probability of particular instances. For 12 cases, the PSSM model performed better. In all 12 cases, the differences were not statistically significant. For 14 cases, there are no noticeable differences, and indeed the learned dependency models in these cases show weak correlation. For 69 cases, the dependency models were better than PSSMs. For 51 of these case, the improvement was statistically significant. When we consider individual methods, the tree networks were better in 33 cases (22 of these were significant), mixtures of 2 PSSMs were better in 59 cases (36 significant), and mixtures of 2 trees were better in 57 cases (35 significant). These results give strong support to the claim that in many cases, modeling dependencies between binding sites positions, can significantly improve generalization performance on new unseen binding sites.

4. BINDING SITES IDENTIFICATION

An important usage for the learned binding site motifs is to identifying putative binding sites in new sequences. Suppose we learned a model representing the joint distribution $P(X_1, \ldots, X_K)$ at the binding site. Given a new promoter sequence, we want to check if it contains a binding site. Naively, we can scan it and look for the most probable K-mer given our model. However, such a K-mer may be also probable in the background distribution. Thus, to minimize the number of false identifications, we want to consider only those K-mers that are probable given our model and are improbable in the background distribution. Statistically speaking, given a background distribution P_0 over the sequences, a natural score to use is the log-odds ratio of the probability of the K-mer given our model and its probability in the background distribution:

$$\text{Score}(x_1, \ldots, x_K) = \log \frac{P(x_1, \ldots, x_K)}{P_0(x_1, \ldots, x_K)} \tag{3}$$

Suppose we find a K-mer with score s. To attach statistical significance to this finding, we want to compute the *p-value* of a score s. This is the probability of finding a score as high as this in random K-mers sampled from the background distribution P_0.

There are several ways to compute p-values. A naive approach uses an empirical estimate of the p-value by sampling K-mers from P_0 and scores each one. Unfortunately, estimation of high significance levels requires vast amounts of samples. Alternatively, we can use a Gaussian approximation of the score distribution. This requires computing the mean and variance of the score distribution under P_0. For some of our models this term can be computed by a closed form analytical solution. Lamentably, in our settings this approach suffers from high inaccuracies, especially for the extreme scores region of interest (see Supplementary Information for details [5, Item C]).

To cope with this problem we use the method of *importance sampling* (e.g. [21]) to estimate the score distribution. Instead of sampling K-mers from P_0, we sample from an alternative distribution

Q. Formally, this step is justified by the following manipulation

$$\begin{aligned}
&P_0(\text{Score}(X_1,\ldots,X_K) \geq s) \\
&= \boldsymbol{E}_{P_0}\left[\mathbf{1}\left\{\text{Score}(X_1,\ldots,X_K) \geq s\right\} \cdot \frac{Q(X_1,\ldots,X_K)}{Q(X_1,\ldots,X_K)}\right] \\
&= \boldsymbol{E}_{Q}\left[\mathbf{1}\left\{\text{Score}(X_1,\ldots,X_K) \geq s\right\} \cdot \frac{P_0(X_1,\ldots,X_K)}{Q(X_1,\ldots,X_K)}\right]
\end{aligned}$$

Where $\mathbf{1}\{\}$ is the indicator function. Thus, we get an estimate of the p-value by using *weighted* samples from Q, where the weight of each sample is the ratio P_0/Q of its probability.

The success of this approach depends on the choice of Q. Intuitively, we want a distribution that has most of its mass in the "interesting" regions of the distribution (where the score will indeed be high). A useful proposal distribution should ensure that the sampled distribution will be precise in all regions and particularly in the region of interest. Thus, it is important to sample from distributions that span the range between the background model P_0 and our model of interest P. We do so by defining Q to be a mixture of the form $Q = \sum_{i=1}^{n} \alpha_i Q_i$, where $\sum_{i=1}^{n} \alpha_i = 1$,$Q_1 = P_0$, $Q_n = P$. Each Q_i itself represents an interpolation between P_0 and P. See [5, Item C] for further details.

When we evaluate the statistical significance of different K-mers, we must keep in mind that we are testing multiple hypotheses. If we are searching a promoter sequence of length N, we evaluate $N - K + 1$ K-mers (or twice as much if we are also searching the reverse strand). And so, even if the sequence does not contain a real binding site, we expect that the best K-mer in the sequence will receive a score of some significance. To control for this we use *Bonferroni threshold*. If we want to use significance level of 0.05, and evaluate L positions, we need to check that the best scoring K-mer has p-value less than $0.05/L$. The same effect is achieved by multiplying all p-values by the number of tested K-mers in the sequence, these are called *Bonferroni corrected* p-values.

To summarize, for each sequence suspected of regulation we calculate the discriminative score for each of its K-mers. For each K-mer we calculate its *Bonferroni corrected* significance and report it as a candidate binding site if this value is less than 0.05.

5. MOTIF DISCOVERY IN UNALIGNED PROMOTER SEQUENCES

Up to this point, we examined the question of modeling dependencies in binding sites positions, and how to use these models to identify potential binding sites. In many cases, we are only given genomic sequences suspected of co-regulation, and are asked to construct a binding site model that can "explain" this co-regulation. That is, we want to model a binding site that is common in the promoters sequences of the co-regulated promoters and is rare in other promoters. The main difficulty is that the actual binding position of the transcription factor in each sequence is unknown.

Modeling a Regulated Sequence As a preamble to learning motif models, we describe how we model regulated sequences. We use a generative approach to describe the probabilistic processes that could have generated the promoter sequences. Our model is similar to the model used by probabilistic approaches for PSSM learning (such as MEME [3]), with the important difference that we allow for a general binding site model as described in section 3.

The model assumes that each sequence $\mathbf{S} = \langle S^{(1)}, \ldots, S^{(L)} \rangle$ can be generated in two ways. It is either regulated by the transcription factor $\mathcal{T}$ and so contains a single binding site from our model $\mathcal{M}$, or it is not regulated. (The extension to deal with a multiple binding site model, as in MEME, is straightforward.) We use the random variable R to denote these two cases. The event $R = r^t$ denotes that $\mathbf{S}$ contains a binding site, and the event $R = r^f$ denotes the complementary event. The probability of generating a particular sequence given a motif model $\mathcal{M}$ is the sum over these two possible events

$$P(\mathbf{S} \mid \mathcal{M}) = P(r^f) \cdot P(\mathbf{S} \mid r^f) + P(r^t) \cdot P(\mathbf{S} \mid r^t, \mathcal{M})$$

where we use r^t and r^f as shorthand notation for the event $R = r^t$ and $R = r^f$, respectively.

If the sequence does not contain a binding site, we model it using a *background distribution*. We estimate this distribution from promoter regions of genes in the same genome. We model the background distribution by a k-order Markov chain. The probability of a sequence generated from the background probability is then

$$P(\mathbf{S} \mid r^f) = \prod_{i=1}^{L} P_0(S^{(i)} \mid \langle S^{(i-k)}, \ldots, S^{(i-1)} \rangle) \qquad (4)$$

where P_0 is a time-invariant k-order Markov model.

We now consider the case where $\mathbf{S}$ is regulated by $\mathcal{T}$. Here, we assume that the binding site is generated by the probability described by $\mathcal{M}$, and the rest of the promoter is generated from the background distribution. For now, we describe the probability in a manner that does not depend on the details of the model $\mathcal{M}$, and focus on how it effects the probability of the sequence. The basic problem in computing the probability of a regulated sequence, is that we do not know the location of the binding site. Thus, we introduce a random variable H that denotes this location, and average over all possible values H can take:

$$P(\mathbf{S} \mid r^t, \mathcal{M}) = \sum_{h=1}^{L-K+1} P(h) P(\mathbf{S} \mid r^t, h, \mathcal{M}) \qquad (5)$$

where $P(h)$ is the prior probability that $H = h$ over all possible binding positions. We take this to be a uniform distribution, although we note that prior knowledge about promoter sequence organization can be incorporated via this distribution.

For a specific value h of H, the probability of $\mathbf{S}$ is

$$\begin{aligned}
&P(\mathbf{S} \mid r^t, h, \mathcal{M}) = \\
&\quad P(\mathbf{S} \mid r^f) \frac{P(S^{(h)}, \ldots, S^{(h+K-1)} \mid \mathcal{M})}{\prod_{i=h}^{h+K-1} P_0(S^{(i)} \mid \langle S^{(i-k)}, \ldots, S^{(i-1)} \rangle)} \qquad (6)
\end{aligned}$$

The fractional term is the log-odds ratio between the probability of the K-mer $\langle S^{(h)}, \ldots, S^{(h+K-1)} \rangle$ given the motif model $\mathcal{M}$ and its probability in the background distribution P_0. As we see, a sequence is probable given r^t, if it contains a K-mer with a higher probability according to the model $\mathcal{M}$ than according to the background distribution.

Having defined a probabilistic model, we can use Bayes rule to compute both the posterior probability of regulation

$$P(r^t \mid \mathbf{S}, \mathcal{M}) = \frac{P(r^t) P(\mathbf{S} \mid r^t, \mathcal{M})}{P(\mathbf{S} \mid \mathcal{M})}$$

and the posterior probability of a specific K-mer being a binding site for the motif model $\mathcal{M}$

$$P(h \mid \mathbf{S}, r^t, \mathcal{M}) = \frac{P(h) P(\mathbf{S} \mid h, r^t, \mathcal{M})}{P(\mathbf{S} \mid r^t, \mathcal{M})}$$

Incorporating Biological Observations The above model left the choice of $P(r^t)$ as a free parameter. This parameter represents the prior probability that a sequence is regulated, *before* we see the actual sequence. During training, we want to introduce additional knowledge that mark specific sequences as regulated and others as not. A simplistic way of doing this is to assume that as part of the training data we observe R. In this case, the training data will consists of pairs $< \mathbf{S}, R >$ that contain a sequence and whether it is regulated or not. For example, if we have a cluster of co-expressed genes, we can set $R = r^t$ for promoter sequences of genes in the cluster and $R = r^f$ for all other sequences. The problem with the simplistic approach is that often we are not that confident in our training data. The cluster of co-expressed genes might contain false positive (i.e., non-regulated genes that appear in the cluster), and similarly there might be false negative genes (that are regulated but were not included in the cluster). To capture such considerations, we take a probabilistic approach, and allow the learning algorithm to view the promoter sequences of genes in the cluster as having high probability of being regulated, and all other sequences as having low probability of being regulated.

To deal with this case, we introduce a random variable O that denotes our *observation* about the gene and is dependent of the regulated status of the gene, but not on its promoter sequence. Under this assumption, the probability of a the observation and the sequence is

$$P(\mathbf{S}, O) = \sum_r P(r)P(O \mid r)P(\mathbf{S} \mid r)$$

Thus, we view O as a *noisy sensor* of the underlying biological regulation. Given O, we can readily calculate the *posterior* probability of regulation given our observation via Bayes rule.

A crucial detail lies in the choice of $P(O \mid R)$. If the observation is that of co-expressed genes or genes with similar functional annotation, we can set this distribution to reflect the fact that most regulated genes will appear in the co-regulated cluster. Similarly, we want the distribution to reflect that few non-regulated genes will appear in the cluster.

A more interesting case involves ChIP localization data [29, 35, 38]. In this case the observation is a p-value that the sequence is enriched in the immnuprecipitation assay. A significant localization p-value is an indication that the sequence is bound by the assay's target transcription factor. To model the dependence of the localization p-value on the R attribute, we use the noisy sensor model of Segal *et al* [37]. This model encodes that when the p-value is small, it is most likely generated by a regulated sequence. As the p-value grows, the probability given r^t decays exponentially, and when the p-value is sufficiently large, it is most likely generated by a non-regulated sequence.

Learning We now have all the tools necessary to describe the learning procedure. The algorithm's input is a dataset $\mathcal{D}$ that consists of M promoter sequences $\mathbf{S}[1], \ldots, \mathbf{S}[M]$, and their associated observations $O[1] \ldots, O[M]$. We want to learn a motif model that maximizes the log-likelihood of the data

$$\ell(\mathcal{D} : \mathcal{M}) = \sum_{m=1}^{M} \log P(\mathbf{S}[m], O[m] \mid \mathcal{M}) \tag{7}$$

We assume the background distribution P_0 is known and fixed. Our task amounts to estimating the structure and parameters of the motif model. Unfortunately, due to the fact that both $R[m]$ and $H[m]$ are unknown, there is no simple estimation procedure for this task. Instead, we use an *Expectation Maximization (EM)* [13] approach. The EM algorithm uses the current model to "complete" the probability of the hidden values. Given such a completion, we no longer have missing values and a *maximum likelihood* model is computed analytically. We then use the new model for completing the data and so forth. The procedure iterates until it convergences to a (local) maximum. This procedure is a form of hill climbing and is guaranteed to improve the likelihood at each iteration. The *Structural Expectation Maximization (SEM)* algorithm [15] generalizes this idea when we also learn structure.

For models where the structure is fixed and we learn using maximum likelihood (PSSMs and mixtures of PSSMS), we define EM as progressing through a sequence models $\mathcal{M}^0, \mathcal{M}^1, \ldots$ such that

$$\mathcal{M}^{t+1} = \arg\max_{\mathcal{M}} Q(\mathcal{M} : \mathcal{M}^t, \mathcal{D})$$

where

$$Q(\mathcal{M} : \mathcal{M}^t, \mathcal{D}) = \sum_{m,r,h} P(r, h \mid \mathbf{S}[m], O[m], \mathcal{M}^t) \cdot \log P(\mathbf{S}[m], r, h \mid \mathcal{M}) \tag{8}$$

A useful property of $\log P(\mathbf{S}, R, H \mid \mathcal{M})$ is that it can be further decomposed into a sum of terms. Using Eq. (4)–(6) and discarding of the terms that do not involve $\mathcal{M}$, we find that maximizing $Q(\mathcal{M} : \mathcal{M}^t, \mathcal{D})$ is equivalent to maximizing

$$\sum_m \sum_h P(r^t, h \mid \mathbf{S}[m], O[m], \mathcal{M}^t) \cdot \log P(X_1 = S^{(h)}[m], \ldots, X_K = S^{(h+K-1)}[m] \mid \mathcal{M})$$

This problem is in the form of Eq. (2) but where we now consider each K-mer in the input sequences as a training sample. Essentially, each K-mer is taken into account while learning $\mathcal{M}^{t+1}$ in proportion to our *current* belief that it is a binding site. This is achieved probabilistically by weighing each K-mer by the term $P(R[m] = r^t, H[m] = h \mid \mathbf{S}[m], O[m], \mathcal{M}^t)$ which is exactly the probability of regulation given the previous model.

When we also face the problem of learning structure and use the Bayesian score to guide our learning procedure (tree networks, and mixture of trees), the details are more complex. However, the final upshot is similar [16].

To summarize, our learning procedure consists of two phases

- **E-step:** use $\mathcal{M}^t$ to compute the weight for each K-mer in the input sequences.
- **M-step:** Set $\mathcal{M}^{t+1}$ to be the model learned by the procedures of Section 3 using the weighted dataset of the E-step.

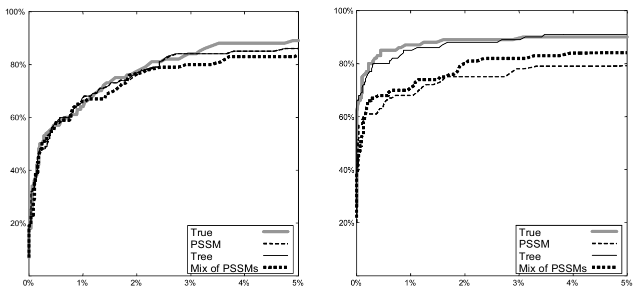
It is important to note that following the freedom we had in choosing a model in Section 3, we can consider any Bayesian network in the M-Step.

Initial Model A final issue is the choice of the initial model $\mathcal{M}^0$. The EM algorithm is typically sensitive to bad starting points and can get trapped in inferior local maximum. The choice of a reasonable (albeit not perfect) starting point is crucial to the success of the whole learning algorithm. In general, we can use any algorithm for finding regulatory motifs in DNA sequences, and use the K-mers suspected as binding sites to define the initial distribution of $\mathcal{M}^0$. As our learning favors discriminative motifs, we use a simple and efficient variant of the algorithm described by Barash *et al* [4]. This algorithm uses random projections of subsequences, as described by Buhler and Tompa [8]. Having chosen a random projection, we check whether it appears in each input sequence. Each of the projected K-mers is then scored by a hypergeometric p-value for enrichment in sequences with $P(r^t \mid O) > 0.5$. We repeat this test using several random projections, and choose the most significant projected K-mer we find. The subsequences of length K that match this projected word are used as the samples for training the initial model $\mathcal{M}^0$.

The learning method we described is similar in the general architecture to previous EM-based methods, such as MEME [3]. It differs from MEME in several aspects. First and foremost, we plug in a general Bayesian network learner that learns a model in a representation of choice (e.g., trees, mixtures of PSSMs, etc.). Second, the learning process takes into account partial observations about

Table 1: Performance evaluation on synthetic data. For each method we report the sensitivity, specificity, and statistical significant of the set of sequences predicted to contain the motif. These were compared to the known planted sequences. The left hand side of the table is for datasets created using a PSSM model (no dependencies), while the right hand side for datasets created using a tree network. Each column reports a different setting of training data parameters: Number of true positive sequences (TP) vs. false positive ones (FP) in the training data. The line with method True reports the performance of the model from which we sampled the binding sites.

	PSSM Generated						Tree Generated					
Learned Model	TP = 100; FP = 0		TP = 50; FP = 50		TP = 25; FP = 75		TP = 100; FP = 0		TP = 50; FP = 50		TP = 25; FP = 75	
PSSM	68%,56%	2.33e-95	65%,51%	1.05e-86	64%,57%	3.96e-89	68%,65%	1.07e-101	68%,65%	1.07e-101	67%,63%	7.78e-99
Tree	68%,56%	2.33e-95	65%,52%	5.15e-87	66%,56%	3.22e-92	85%,66%	6.04e-135	82%,70%	1.91e-132	80%,66%	1.47e-124
Mix of PSSMs	67%,55%	9.69e-93	61%,48%	1.55e-78	53%,46%	3.34e-65	70%,64%	2.14e-104	66%,59%	1.93e-94	67%,57%	1.68e-94
Mix of Trees	60%,48%	4.90e-77	43%,38%	2.32e-47	40%,48%	3.15e-49	72%,62%	9.39e-107	67%,54%	2.12e-92	64%,56%	2.01e-88
True	67%,54%	2.12e-92	67%,54%	2.12e-92	67%,54%	2.12e-92	86%,68%	1.03e-138	86%,68%	1.03e-138	86%,68%	1.03e-138



(a) PSSM (b) Tree

Figure 3: Evaluation of training on synthetic data. ROC curves showing ability to identify binding sites on test data. (a) Binding sites generated from a PSSM model. (b) Binding sites generated from a tree model. The x-axis shows the *false positive rate*, FP/(FP+TN), the y-axis shows the *true positive rate*, TP/(TP+FN). Each curve shows the performance for one model: True– the model that generated the data; PSSM– the learned PSSM model; Tree – the learned tree model; Mix of PSSMs– the learned mixture of PSSMs. In both graphs, the training data consisting of 100 true positive sequences of length 500bp.

each sequence. This allows us to combine sequences with different strengths of support of being regulated. Third, our method incorporates a rich background model during learning and not as a post-processing step. Finally, we use as a starting point a discriminative combinatorial search method. This search method attempts to find initial solutions that best distinguish the sequences that are believed to be regulated from these that are believed to be not regulated.

6. EXPERIMENTAL RESULTS

At the beginning of this paper we stated two fundamental questions: whether dependencies between positions are evident in biological data, and whether learning models of position dependencies can improve *de novo* binding site discovery. We discussed the first question in Section 3.2, and now turn to the second one.

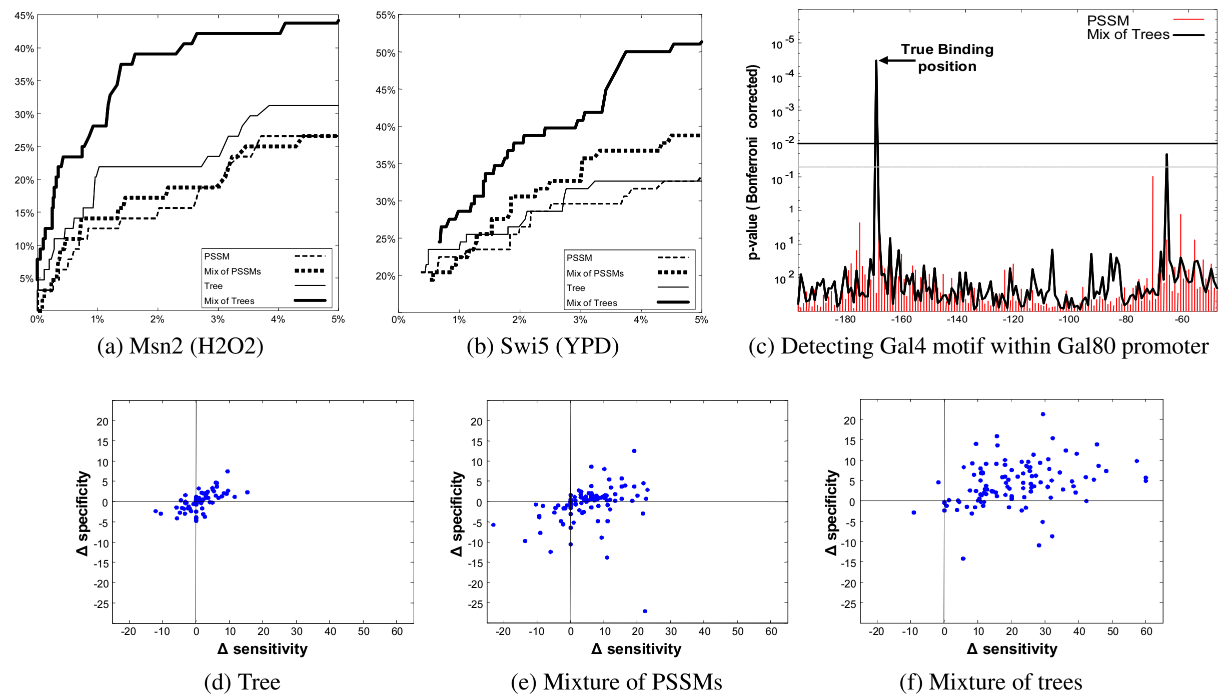
Synthetic Data We begin by evaluating our methods on synthetic data. For this task, we built several datasets, each consisting of both "positive" promoters, (i.e. sequences in which we planted binding site motifs), and "negative" ones. To simulate the underlying biological problem as accurately as possible, the motifs themselves were sampled from models trained on known binding sites of the Human LUN transcription factor from the TRANSFAC database (`V$LUN1_01`). In each setting, we created two parallel sets, one sampled from a tree network that contains position dependencies, and the other from a PSSM model. The promoter sequences were sampled from a 3-order Markov model background distribution, trained on Human promoter regions. To simulate noise, we have contaminated our datasets with another group of "false positive" promoters, where no motif was planted.

We set the observation model such that all "positive" sequences had $P(r^t \mid O) = 0.99$, while the "negative" ones had $P(r^t \mid O) = 0.01$. We tested our methods on a variety of settings, changing both the promoters lengths (from 250 to 500bp), and the composition of the "positive" promoters: from 100 true positives without false ones, down to 25 true positives with 75 false ones. The synthetic datasets are available from our site ([5, Item B.1]). After applying our methods to the synthetic training data, we tested them on unseen test data that was similarly generated. Each test promoter was assigned a Bonferroni-corrected p-value, according to its best scoring K-mer (see Section 4). We then used these scores to discriminate between putative "positive" promoters and "negative" ones.

The results on one dataset are shown in Figure 3, as ROC curves. These curves compare the *false positive rate* to the *true positive rate*, when changing the p-value threshold. It is evident that all methods perform similarly on the data generated from a PSSM, and are comparable to the *true* model that generated the data. The data generated from a tree network, shows a difference between the performances of the learned tree network to both the PSSM and the mixture of PSSMs models that are incapable of modeling the underlying dependencies and therefore perform worse. The learned tree network closely tracks the performance of the true model.

In practice, we want to retrieve sequences that contain the learned motif. We do so by selecting sequences whose Bonferroni corrected p-value is below a prespecified a threshold. In our experiments, we chose to use a threshold of 0.01. The rationale for this strict threshold is as follows. In genome-wide scans, we examine several thousands of sequences, most of which are not expected to be targets of the transcription factor in question. By setting a strict significance level, we control the number of false positives among the retrieved sequences. In Table 1 we report the quality of the selection procedure using three measures. *Sensitivity* (% of positives sequences retrieved out of all positive ones), *Specificity* (% of true positives retrieved), and the significance of the retrieved sequences, according to the hypergeometric p-value (called *specificity score* by [25]). This score is the probability of retrieving at least that many positive sequences in a random set of sequences of the same size.

As we can see, when the data does not contain false positives sequences, all models perform roughly equivalently on data generated from PSSM. On the tree generated data, we see that the PSSM does the worse, the mixture of PSSMs is slightly better, and finally the tree network is roughly equivalent to the true model. As we increase the noise ratio, the problem becomes harder. The PSSM model and to a large extent the tree model are fairly robust, even in high degrees of noise. On the other hand, the mixture models



(a) Msn2 (H2O2) (b) Swi5 (YPD) (c) Detecting Gal4 motif within Gal80 promoter

(d) Tree (e) Mixture of PSSMs (f) Mixture of trees

Figure 4: Results on ChIP localization data of Lee *et al* [29]. (a) & (b) ROC curves for two example motifs. (c) Example of detection of a motif in upstream region. For each position (x-axis) we show the Bonferroni corrected p-value (y-axis) assigned by each model. (d)–(f) Cross validation evaluation of different methods. Each point represents one localization experiment and shows the difference between the performance of the learned models and learned PSSM in sensitivity (x-axis) and specificity (y-axis).

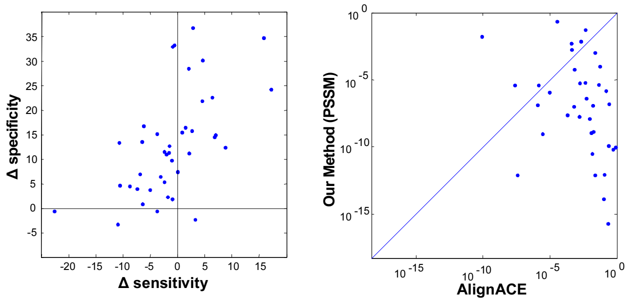
are more susceptible to noise, and their performance decays. When learning from shorter sequences of length 250bp, the same qualitative conclusions reappear (data not shown, see [5, Item B.1]).

ChIP Localization Data To evaluate our methods on real life data, we used a dataset of genome-wide Chromatin Immunoprecipitation (ChIP) localization measurements for 106 yeast transcription factors in 146 experiments [29, 35, 38]. This assay measures the binding affinities of a target transcription factor to promoter regions *in vivo*. The experimental protocol [35] assigns a p-value to each promoter sequence. A sequence with p-value less than 0.001 is considered to be bound by the factor. The stringent threshold of 0.001 is aimed to reduce the false positive identifications in a genome wide screening [29]. This assay provides valuable information about the binding specificity of transcription factors. However, it is important to keep in mind that it does not pinpoint the exact binding location. Based on this, we can test whether our prediction for gene regulation events match the biological predictions of the localization experiment. We stress that our predictions are *based solely on sites identifications* of *de novo* learned motifs.

We focused on 109 experiments for which there where at least 10 genes with localization p-value $\leq$ 0.001 and at least 50 genes with p-value $\leq$ 0.01. The aim of our procedure was to get an objective evaluation of the ability of the learned motif to detect the sequences that the transcription factor binds to. Thus, it is crucial to test performance on sequences that were not seen during training. To achieve this, we performed a 5-fold cross validation test. In each of the 5 runs we used 80% of the yeast genes as training data, and tested the learned motif on the remaining 20% of the genes. Finally, each sequence was scored by a Bonferroni corrected p-value, and was declared to contain an occurrence of the motif if this p-value was smaller than 0.01.

To evaluate the success of the different methods, we tested them against the set of genes that Lee *et al* [29] consider to be bound by the transcription factor (i.e., those with localization p-value $\leq$ 0.001). We note that this set is conservative by nature, and so we expect that it does not contain all the truly bound sequences. However, it provides a good objective test data. In the following discussion we treat these sequences as "true". Figures 4(a) and (b) show ROC curves for two examples: Msn2 (H2O2) and Swi5 (YPD). As we can see in these two examples, models that capture dependencies are clearly superior to the PSSM model. The differences in performances are due to the increased expressiveness of the richer models. Figure 4(c) illustrates a scan of the promoter region in search of a statistically significant putative binding site. Shown is the promoter region of Gal80 with models learned from localization assay of Gal4 (Galactose). As we can see, the mixture of trees model assigns a significant Bonferroni corrected p-value at the true binding position, while the PSSM model does not.

Next, we evaluated the accuracy of the learned motifs in a genome-wide scan for binding sites. For each method, we compared the putative set of regulated sequences against the original set of Lee *et al*, by using the sensitivity and specificity measures. The results are available in the Supplementary Information [5, Table B.2]. A summary of these results appears in Figure 4(d)–(f) that show the differences in sensitivity and specificity between each of our methods and PSSM models. Points in the upper right quadrant represent experiments where the richer model performed better in terms of both sensitivity and specificity, points in the bottom left quadrant are ones where the PSSM model performed better, while points in the other two quadrants are ones where the two methods achieve different tradeoffs between sensitivity and specificity. As we can see, in most experiments all three methods perform better than PSSM models. The tree network shows a modest improvement in 32 experiments, and a slight decrease in 15. The mixture of PSSMs does



(a) (b)

Figure 5: Comparison of PSSMs learned by our method to the ones learned by AlignACE on the gene clusters of [25]. (a) Difference between the PSSMs learned by method and by AlignACE in sensitivity (x-axis) and specificity (y-axis). (b) Comparison of the hypergeometric p-value of PSSMs learned by AlignACE (x-axis) and the ones by our method (y-axis).

somewhat better (55 better, 19 worse), and the mixture of trees is significantly better in virtually all experiments (87 better, 2 worse).

To evaluate the learned models with respect to what is known about the underlying biological context, we compared the PSSMs we learned with known yeast transcription factor binding sites from TRANSFAC [41] (see [5, Table B.2]). In 23 experiments we found a TRANSFAC PSSM for the tested transcription factor. Out of these, in 15 experiments the PSSM we learn matches the known one.

Clusters of Yeast Genes Another rich collection of datasets of genes were collected by the Church lab [25, 39]. These clusters of genes are based on functional annotations, co-expression, and known targets of transcription factors. They were originally analyzed using AlignACE [36]. This analysis included multiple runs of AlignACE, followed by filtering based on the quality of the motifs found. The best PSSMs were reported for each cluster.

To gauge the quality of our baseline method, we compared the PSSMs learned by our procedures to the ones learned and reported by [25]. For this task we used the whole training data (as done by AlignACE), and examined the two learned motifs for each group by comparing their sensitivity, specificity and their hypergeometric p-value. As we can see in Figure 5 (a), Our method had improved the PSSM's performance in 14 cases and reduced it in 3. The main observation is the different tradeoff between sensitivity and specificity of the two learning techniques. However, in most examples (32 out of 41) our PSSM obtained a more significant hypergeometric p-value, as shown in Figure 5 (b). These results show that our PSSM learning procedure is comparable to AlignACE in terms of both motif quality and significance.

Next, we evaluated the different methods on this dataset. For each group having more than 50 genes, we repeated our procedures as above. Due to the noisier nature of the data, we have set the regulation prior $P(r^t \mid O) = 0.75$ for genes inside a cluster, and 0.01 elsewhere. Since the methods differ in their expressiveness and the number of parameters in their representations, a comparison on the same set of sequences used for learning can be misleading. Indeed, when comparing the performance of the different models on the training data, the richer models, and particularly mixtures of PSSMs and trees, seem much better (see Supplementary Information [5, Table B.3]). To get a more realistic assessment, we once again used a 5-fold cross validation protocol as described above. In Figure 6 we see a summary of these results (see Supplementary Information [5, Tables B.4 and B.5] for details). As we can see, the tree networks perform similarly to PSSMs, while both mixture models perform poorly in most cases. To understand this phenomenon, we examined the PSSM results. In many clusters both the sensitivity and the specificity were small ($< 15\%$). This suggests that these clusters contain many false positives as well as false negatives genes, making the problem harder. Recall that our synthetic results show that mixture models are not as robust to the presence of noise as the simpler models. Our suspicion is that they tend to overfit spurious signals from the false positives sequences, and are therefore less suitable for such a domain. We discuss possible solutions for this in the next section.

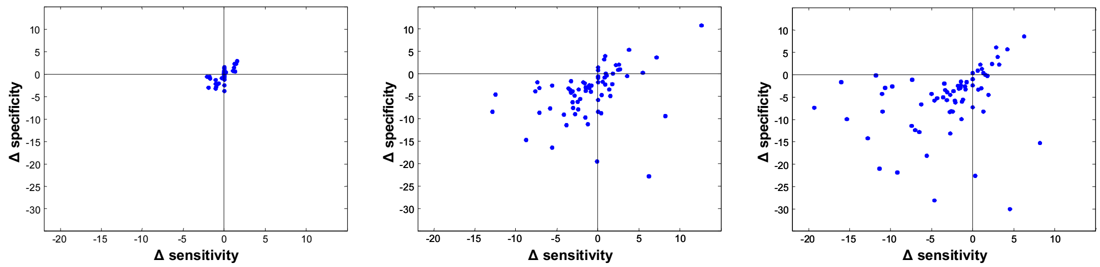
7. DISCUSSION

In this paper we expanded the probabilistic representation of DNA motifs using the language of Bayesian network. Our framework allows any model spanning the range from the position independent PSSM to the full dependency model. We described methods to learn these models from limited data and showed that several types of dependency models (trees, mixtures of PSSMs and mixture of Trees) generalize better than PSSM on unseen real life data. We presented methods for discovering putative binding sites given any Bayesian network model and described an effective approach for evaluating the statistical significance of candidate sites. Finally, we showed how to perform *de novo* discovery of motifs in unaligned genomic sequences suspected of co-regulation. In a thorough empirical evaluation, we compared the effectiveness of dependency models in discovering statistically significant transcription factors on real life clusters.

We are not the first to model dependencies between positions in biological sequence motifs. Agarwal and Bafna [1], suggested the tree network model, and discussed algorithms for learning it. In a related problem of modeling splice junctions, recent works examined k-order Markov models [30] and tree Bayesian networks [10]. These works learned models from aligned binding sites and used them to detect splice junctions in new sequences. Finally, Bayesian networks were used to model dependencies between positions in protein motifs that were aligned according to 3-D structure [26]. This domain introduces another layer of complications due to the large alphabet size of amino acids. Our work is, to the best of our knowledge, the first one that presents a general framework for learning *any* Bayesian network motif model in *de novo* discovery of transcription factors. As we show in Section 6, this ability can lead to dramatic improvements in the learned motifs.

This work can be extended in several directions. First, the advantage of being able to model a motif using any Bayesian network suggests further exploration of different types of models as well as general unrestricted models. This can include representational extensions that are geared toward the complexity vs. expressiveness issue such as *context specific* dependency models [6, 18]. Second, as our framework made no particular assumption on the type of binding sites, it can be readily adapted to discover other sequence motifs such as those of splicing and histone remodeling factors. Third, an important challenge is to integrate our method with additional data. As noted in Section 5, prior information about binding site location can be used. A more interesting challenge is to combine our method with other types of information such as gene expression [6, 37]. The biggest question posed by our results is how to automatically select between different dependency models (including the PSSM model). A natural measure for predicting performance of probabilistic models on test data is the *Bayesian score* [11, 22]. Unfortunately, in our experiments the *Bayesian score* was not successful in selecting one of the best models.

Finally, our analysis here focused mainly on the statistical significance of the results. However, these results also have interest-



(a) Tree (b) Mixture of PSSMs (c) Mixture of trees

Figure 6: Evaluation of different methods on the clusters of the Church Lab [25, 39]. Each point represents a cluster, and shows the difference between the cross-validated performance of the learned models to the learned PSSM in sensitivity (x-axis) and specificity (y-axis).

ing implications about protein-DNA interactions. The challenge is how to relate these dependencies to protein structure and function. For this purpose, we need to be able to estimate our confidence in the discovered dependencies (e.g., using bootstrap [14, 19] or Bayesian methods [20]) and relate these dependencies with three dimensional conformations of Protein-DNA complexes.

Acknowledgments

We thank Doug Brutlag, Hillel Fleischer, Hanah Margalit, Tomer Naveh, Dana Pe'er, Itamar Simon, and Ilan Wapinski for useful discussions relating to this work. This work was supported in part by the Israel Science Foundation (ISF), and the Israeli Ministry of Science. Y. Barash was supported by an Eshkol fellowship. G. Elidan and T. Kaplan were supported by Horowitz fellowships. N. Friedman was supported by an Alon fellowship and the Harry & Abe Sherman Senior Lectureship in Computer Science.

8. REFERENCES

[1] P Agarwal and V Bafna. Detecting non-adjacent correlations within signals in DNA. In *RECOMB'98*. 1998.

[2] TL Bailey and M Gribskov. Combining evidence using p-values: application to sequence homology searches. *Bioinformatics*, **14**:48–54, 1998.

[3] TL Bailey and C Elkan. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In *ISMB'94*. 1994.

[4] Y Barash, G Bejerano, and N Friedman. A simple hyper-geometric approach for discovering putative transcription factor binding sites. In *WABI'01*. 2001.

[5] Y Barash, G Elidan, N Friedman, and T Kaplan. Supplementary information for "modeling dependencies in protein-DNA binding sites". http://www.cs.huji.ac.il/labs/compbio/TFBN/

[6] Y Barash and N Friedman. Context-specific Bayesian clustering for gene expression data. *J. Comp. Bio.*, **9**:169–191, 2002.

[7] PV Benos, AS Lapedes, DS Fields, and GD Stormo. SAMIE: statistical algorithm for modeling interaction energies. In *PSB'01*. 2001.

[8] J Buhler and M Tompa. Finding motifs using random projections. In *RECOMB'01*. 2001.

[9] ML Bulyk, PL Johnson, and GM Church. Nucleotides of transcription factor binding sites exert interdependent effects on the binding affinities of transcription factors. *Nuc. Acids Res.*, **30**:1255–61, 2002.

[10] D Cai, A Delcher, B Kao, and S Kasif. Modeling splice sites with Bayes networks. *Bioinformatics*, **16**:152–158, 2000.

[11] DM Chickering and D Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Mach. Learn.*, **29**:181–212, 1997.

[12] CK Chow and CN Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Info. Theory*, **14**:462–467, 1968.

[13] AP Dempster, NM Laird, and DB Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc*, B **39**:1–39, 1977.

[14] B Efron and RJ Tibshirani. *An Introduction to the Bootstrap*. 1993.

[15] N Friedman. Learning belief networks in the presence of missing values and hidden variables. In *ICML'97*, 1997.

[16] N Friedman. The Bayesian structural EM algorithm. In *UAI'98*. 1998.

[17] N Friedman, D Geiger, and M Goldszmidt. Bayesian network classifiers. *Mach. Learn.*, **29**:131–163, 1997.

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

[19] N Friedman, M Goldszmidt, and A Wyner. Data analysis with Bayesian networks: A bootstrap approach. In *UAI'99*. 1999.

[20] N Friedman and D Koller. Being Bayesian about Bayesian network structure: A Bayesian approach to structure discovery in Bayesian networks. *Mach. Learn.*, **50**:95-126, 2003.

[21] A Gelman, JB Carlin, HS Stern, and DB Rubin. *Bayesian Data Analysis*, 1995.

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

[23] D Heckerman, D Geiger, and DM Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Mach. Learn.*, **20**:197–243, 1995.

[24] GZ Hertz and GD Stormo. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, **15**:563–77, 1999.

[25] JD Hughes, PE Estep, S Tavazoie, and GM Church. Computational identification of *cis*-regulatury elements associated with groups of functional related genes in *saccharomyces cerevisiae*. *J. Mol. Bio.*, **296**:1205–1214, 2000.

[26] TM Klingler and DL Brutlag. Discovering structural correlations in α-Helices. *Prot. Sci.*, **3**:1847–1857, 1994.

[27] DS Latchman. *Eukaryotic Transcription Factors*. 1999.

[28] SL Lauritzen. The EM algorithm for graphical association models with missing data. *Comp. Stat. and Data Analysis*, **19**:191–201, 1995.

[29] TI Lee *et al.* Transcriptional regulatory networks in *saccharomyces cerevisiae*. *Science*, **298**:799–804, 2002.

[30] LP Lim and CB Burge. A computational analysis of sequence features involved in recognition of short introns. *PNAS*, **98**:11193–11198, 2001.

[31] X Liu, DL Brutlag, and JS Liu. Bioprospector: discovering conserved dna motifs in upstream regulatory regions of co-expressed genes. In *PSB'01*. 2001.

[32] M Meila and MI Jordan. Estimating dependency structure as a hidden variable. In *NIPS 10*, 1998.

[33] J Pearl. *Probabilistic Reasoning in Intelligent Systems*, 1998.

[34] 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. *Nuc. Acids Res.*, **23**:4878–4884, 1995.

[35] B Ren *et al.* Genome-wide location and function of dna binding proteins. *Science*, **290**:2306–9, 2000.

[36] FP Roth, PW Hughes, JD Estep, and GM Church. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat. Biotech.*, **16**:939–945, 1998.

[37] E Segal, Y Barash, I Simon, N Friedman, and D Koller. From promoter sequence to expression: A probabilistic framework. In *RECOMB'02*. 2002.

[38] I Simon *et al.* Serial regulation of transcriptional regulators in the yeast cell cycle. *Cell*, **106**:697–708, 2001.

[39] S Tavazoie, JD Hughes, MJ Campbell, RJ Cho, and GM Church. Systematic determination of genetic network architecture. *Nat. Genet.*, **22**:281–5, 1999.

[40] 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'00*. 2000.

[41] E Wingender it et al. The TRANSFAC system on gene expression regulation. *Nuc. Acids Res.*, **29**:281–283, 2001.