FITTING A MIXTURE MODEL BY EXPECTATION MAXIMIZATION TO DISCOVER MOTIFS IN BIOPOLYMERS

UCSD Technical Report CS94-351

TIMOTHY L. BAILEY TBAILEY@CS.UCSD.EDU
Department of Computer Science and Engineering,
University of California at San Diego, La Jolla, California 92093-0114
phone: (619) 534-8187
fax: (619) 534-7029

CHARLES ELKAN ELKAN@CS.UCSD.EDU
Department of Computer Science and Engineering,
University of California at San Diego, La Jolla, California 92093-0114
phone: (619) 534-8897
fax: (619) 534-7029

ABSTRACT: The algorithm described in this paper discovers one or more motifs in a collection of DNA or protein sequences by using the technique of expectation maximization to fit a two-component finite mixture model to the set of sequences. Multiple motifs are found by fitting a two-component finite mixture model to the data, probabilistically erasing the occurrences of the motif thus found, and repeating the process to find successive motifs. The algorithm requires only a set of sequences and a number specifying the width of the motifs as input. It returns a model of each motif and a threshold which together can be used as a Bayes-optimal classifier for searching for occurrences of the motif in other databases. The algorithm estimates how many times each motif occurs in the input dataset and outputs an alignment of the occurrences of the motif. The algorithm is capable of discovering several different motifs with differing numbers of occurrences in a single dataset. Motifs are discovered by treating the set of sequences as though they were created by a stochastic process which can be modelled as a mixture of two densities, one of which generated the occurrences of the motif, and the other the rest of the positions in the sequences. Expectation maximization is used to estimate the parameters of the two densities and the mixing parameter.

KEYWORDS: Unsupervised learning, expectation maximization, mixture model, consensus pattern, motif, biopolymer, binding site.

Contents

Acknowledgements: Timothy Bailey is supported by NIH Genome Analysis Pre-Doctoral Training Grant No. HG00005. Charles Elkan is supported in part by the National Science Foundation under Award No. IRI-9110813.

1 Introduction

Finding a cluster of numerous similar subsequences in a set of biopolymer sequences is evidence that the subsequences occur not by chance but because they share some biological function. For example, the shared biological function which accounts for the similarity of a subset of subsequences might be a common protein binding site or splice junction in the case of DNA sequences, or the active sites of related enzymes in the case of protein sequences. This paper describes an algorithm called MM which, given a dataset of possibly related biopolymer sequences, estimates the parameters of a probabilistic model which could have generated the dataset. The probabilistic model is a two-component finite mixture model. One component describes a set of similar subsequences of fixed width (the "motif"), while the other component describes all other positions in the sequences (the "background"). Fitting the model includes estimating the relative frequency of motif occurrences. This estimate can be used to select the threshold for a Bayes-optimal classifier for finding occurrences of the motif in other databases.

The MM algorithm is an extension of the expectation maximization technique for fitting finite mixture models developed by Aitkin and Rubin [1985]. It is related to the algorithm based on expectation maximization described by Lawrence and Reilly [1990], but it relaxes the assumption that each sequence in the dataset contains one occurrence of the motif. In other words, MM solves an unsupervised learning problem: it is intended to be useful for discovering new motifs in datasets that may or may not contain motifs, treating each subsequence of width W in the dataset as an unlabeled sample. On the other hand, the algorithm of Lawrence and Reilly [1990] treats each sequence as a labeled sample, and solves a supervised learning problem.

MM is also related to the algorithm described in [Bailey and Elkan, 1993]. Unlike that algorithm, MM estimates cluster size (number of occurrences of the motif) at the same time it is learning models of the motif and the background. This removes the need for the user of the algorithm to know in advance the number of times the motif occurs in the dataset. This makes it possible to search for motifs in datasets about which very little is known.

The MM algorithm has been implemented as an option to the MEME software for discovering multiple motifs in biopolymer sequences [Bailey and Elkan, 1993]. MM can therefore be used to discover multiple motifs in a dataset. Briefly, this is done by repeatedly applying MM to the dataset and then probabilistically erasing all occurrences of the discovered motif. Because MM estimates the number of occurrences of each motif, MEME using MM is able to find motifs with different numbers of occurrences in a single dataset. This increases the usefulness of MEME as a tool for exploring datasets that contain more than one motif.

The rest of this paper is organized as follows. Section 2 describes the finite mixture model used by MM, and Section 3 gives the analysis needed to apply

the expectation maximization idea to this type of model. Section 4 describes the implementation of MM in the context of MEME. Section 5 presents experimental results of using the MM algorithm to discover motifs in several DNA and protein datasets. Finally, Section 6 concludes the paper by discussing the strengths and limitations of the MM algorithm.

2 The finite mixture model

The MM algorithm searches for maximum likelihood estimates of the parameters of a finite mixture model which could have generated a given dataset of biopolymer sequences. We will refer to the dataset as $Y = (Y_1, Y_2, \ldots Y_N)$, where N is the number of sequences in the dataset. The sequences Y_i are assumed to be over some fixed alphabet, say $A = (a_1, a_2, \ldots, a_L)$, which is given as input to the algorithm. The mixture model used by MM does not actually model the dataset directly. Instead, the dataset is broken up conceptually into all n (overlapping) subsequences of length W which it contains. This new dataset will be referred to as $X = (X_1, X_2, \ldots, X_n)$. MM learns a finite mixture model which models the new dataset. Although this model does not, strictly speaking, model the original dataset, in practice it is a good approximation, especially when care is taken to ensure that the model does not predict that two overlapping subsequences in the new dataset both were generated by the motif. This is done by enforcing a constraint on the estimated probabilities of overlapping subsequences being motif occurrences. How this constraint is enforced is discussed later.

The model for the new dataset consists of two components which model the motif and background (non-motif) subsequences respectively. The motif model used by MM says that each position in a subsequence which is an occurrence of the motif is generated by an independent random variable describing a multinomial trial with parameter $f_i = (f_{i1}, \ldots, f_{iL})$. That is, the probability of letter a_j appearing in position i in the motif is f_{ij}. The parameters f_{ij} for $i = 1, \ldots, W$ and $j = 1, \ldots, L$ must be estimated from the data. The background model says that each position in a subsequence which is not part of a motif occurrence is generated independently, by a multinomial trial random variable with a common parameter $f_0 = (f_{01}, \ldots, f_{0L})$. In other words, MM assumes that a sequence of length W generated by the background model is a sequence of W independent samples from a single background distribution. The overall model for the new dataset which MM uses is that the motif model (with probability λ_1) or the background model (with probability $\lambda_2 = 1 - \lambda_1$) is chosen by nature and then a sequence of length W is generated according to the probability distribution governing the model chosen. In summary, the parameters for the overall model of the data assumed by MM are the mixing parameter $\lambda = (\lambda_1, \lambda_2)$, vectors of letter frequencies for the motif model $\theta_1 = (f_1, f_2, \ldots, f_W)$, and a single vector of letter frequencies for the background model $\theta_2 = f_0$.

In order to find more than one motif in a dataset, a value called an "erasing factor" is associated with each letter in the dataset. These values are all initially set to 1 to indicate that no erasing has occurred. After MM converges to a motif, the erasing factors are all adjusted downward towards 0 in proportion to the probability that each character is part of an occurrence of the motif just found. The erasing factors are used by MM in the reestimation of θ and effectively remove occurrences of motifs already found from the dataset. This method of finding multiple motifs is discussed in more detail in [Bailey and Elkan, 1993].

The motif discovered by MM can be used to search for motif occurrences in other datasets. It can be used directly to estimate the conditional probability of any subsequence of length W given the motif. It can also be used to compute the likelihood ratio for testing whether the subsequence was generated by the motif or the background components of the model. This can be done by using the estimated values of the parameter θ to create a log-odds matrix of the type referred to as a "specificity matrix" by Hertz *et al.* [1990]. The estimated value of the mixing parameter, λ, can be used to calculate the optimum threshold for using the log-odds matrix as a classifier. In addition, the log-odds matrix can be used as a profile (without gaps) [Gribskov *et al.*, 1990] of the motif in the other applications of profile analysis.

3 Expectation maximization in finite mixture models

The MM algorithm uses expectation maximization (EM) to discover motifs in datasets of sequences. The next two sections describe how EM can be applied to the problem at hand and how the likelihoods of the motifs found are calculated.

3.1 The MM algorithm

The MM algorithm does maximum likelihood estimation: its objective is to discover those values of the parameters of the overall model which maximize the likelihood of the data. To do this, the expectation maximization algorithm for finite mixture models of Aitkin and Rubin [1985] is used. This iterative procedure finds values for $\lambda = (\lambda_1, \lambda_2)$ and $\theta = (\theta_1, \theta_2)$ which (locally) maximize the likelihood of the data given the model.

A finite mixture model assumes that data

$$X = (X_1, X_2, \ldots, X_n), \text{ where } n \text{ is the number of samples}$$

arises from two or more groups with known distributional forms but different, unknown parameters. Expectation maximization (EM) can be used to find maximum likelihood estimates of the unknown parameters

$$\theta = (\theta_1, \theta_2, \ldots, \theta_g), \text{ where } g \text{ is the number of groups}$$

of the distributions, as well as the mixing parameters

$\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_g)$, where λ_i is the relative size of the ith group, and

$$\sum_{i=1}^{g} \lambda_i = 1, \;\; \lambda_i \geq 0.$$

The EM algorithm makes use of the concept of missing data. In this case, the missing data is the the knowledge of which group each sample in the data came from. The following notation is useful:

$$\begin{aligned} Z &= (Z_1, Z_2, \ldots, Z_n), \text{ where } n \text{ is the number of samples} \\ Z_i &= (Z_{i1}, Z_{i2}, \ldots, Z_{ig}), \text{ where } g \text{ is the number of groups} \\ Z_{ij} &= \begin{cases} 1 & \text{if } X_i \text{ from group } j \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The variable Z_i gives the group membership for the ith sample. In other words, if $Z_{ij} = 1$, then X_i has the distribution $p(X_i|\theta_j)$. The values of the Z_{ij} are unknown, and are treated by EM as missing information to be estimated along with the parameters θ and λ of the mixture model.

The definition of Z makes it clear that the prior probability that a particular $Z_{ij} = 1$ is just λ_j,

$$P(Z_{ij} = 1|\theta, \lambda) = \lambda_j, \;\; 1 \leq i \leq n. \tag{1}$$

For a given i, all Z_{ij} are 0 except for one so the conditional densities of the data and of the missing data can be written as

$$p(X_i|Z_i, \theta, \lambda) = \prod_{j=1}^{g} p(X_i|\theta_j)^{Z_{ij}} \quad \text{and} \tag{2}$$

$$p(Z_i|\theta, \lambda) = \prod_{j=1}^{g} \lambda_j^{Z_{ij}}. \tag{3}$$

By the definition of conditional probability and (2) and (3), we can write the joint density of a sample and its missing information as

$$\begin{aligned} p(X_i, Z_i|\theta, \lambda) &= p(X_i|Z_i, \theta, \lambda) p(Z_i|\theta, \lambda) \\ &= \prod_{j=1}^{g} [p(X_i|\theta_j)\lambda_j]^{Z_{ij}}. \end{aligned} \tag{4}$$

By the assumption of independent samples X, the joint density of the data and all the missing information is

$$p(X, Z|\theta, \lambda) = \prod_{i=1}^{n} p(X_i, Z_i|\theta, \lambda) \tag{5}$$

$$= \prod_{i=1}^{n} \prod_{j=1}^{g} [p(X_i|\theta_j)\lambda_j]^{Z_{ij}}. \tag{6}$$

The likelihood of the model parameters θ and λ given the joint distribution of the data X and the missing data Z is defined as

$$L(\theta, \lambda|X, Z) \quad = \quad p(X, Z|\theta, \lambda). \tag{7}$$

The log of the likelihood (*log likelihood*) is therefore

$$\log L(\theta, \lambda|X, Z) \quad = \quad \sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij} \log(p(X_i|\theta_j)\lambda_j). \tag{8}$$

The EM algorithm iteratively maximizes the expected *log likelihood* over the conditional distribution of the missing data Z, given (a) the observed data X, and (b) current estimates of parameters θ and λ.

The E-step of EM finds the expected value of the *log likelihood* (8) over the values of the missing data Z, given the observed data X, and the current parameter values $\theta = \theta^{(0)}$ and $\lambda = \lambda^{(0)}$. Since the expected value of a sum of random variables is the sum of their individual expectations, we have

$$\begin{aligned}
\underset{(Z|X,\theta^{(0)},\lambda^{(0)})}{\mathrm{E}}[\log L(\theta, \lambda|X, Z)] \quad &= \quad \underset{(Z|X,\theta^{(0)},\lambda^{(0)})}{\mathrm{E}}[\sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij} \log(p(X_i|\theta_j)\lambda_j)] \\
&= \quad \sum_{i=1}^{n}\sum_{j=1}^{g} \underset{(Z|X,\theta^{(0)},\lambda^{(0)})}{\mathrm{E}}[Z_{ij} \log(p(X_i|\theta_j)\lambda_j)] \\
&= \quad \sum_{i=1}^{n}\sum_{j=1}^{g} \mathrm{E}[Z_{ij}|X, \theta^{(0)}, \lambda^{(0)}] \log(p(X_i|\theta_j)\lambda_j).
\end{aligned} \tag{9}$$

The expected value of Z_{ij} can be found using the definition of expected value, Bayes' rule, and the definitions (1) and (2). Defining $Z_{ij}^{(0)} = \mathrm{E}[Z_{ij}|X, \theta^{(0)}, \lambda^{(0)}]$, we have

$$\begin{aligned}
Z_{ij}^{(0)} \quad &= \quad \mathrm{E}[Z_{ij}|X, \theta^{(0)}, \lambda^{(0)}] \\
&= \quad 1 \cdot P(Z_{ij} = 1|X_i, \theta^{(0)}, \lambda^{(0)}) + 0 \cdot P(Z_{ij} = 0|X_i, \theta^{(0)}, \lambda^{(0)}) \\
&= \quad \frac{p(X_i|Z_{ij} = 1, \theta^{(0)}, \lambda^{(0)}) \;\; P(Z_{ij} = 1|\theta^{(0)}, \lambda^{(0)})}{p(X_i|\theta^{(0)}, \lambda^{(0)})} \\
&= \quad \frac{p(X_i|\theta_j^{(0)})\lambda_j^{(0)}}{\sum_{k=1}^{g} p(X_i|\theta_k^{(0)})\lambda_k^{(0)}}, \quad i = 1, \ldots, n, \; j = 1, \ldots, g.
\end{aligned} \tag{10}$$

Substituting $Z_{ij}^{(0)}$ into equation (9) and rearranging we have

$$\begin{aligned}
E[\log L(\theta, \lambda|X, Z)] \quad &= \quad \sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij}^{(0)} \log(p(X_i|\theta_j)\lambda_j) \\
&= \quad \sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij}^{(0)} \log p(X_i|\theta_j) + \sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij}^{(0)} \log \lambda_j.
\end{aligned} \tag{11}$$

The M-step of EM maximizes (11) over θ and λ in order to find the next estimates for them, say $\theta^{(1)}$ and $\lambda^{(1)}$. The maximization over λ involves only the second term in (11):

$$\lambda_j^{(1)} = \underset{\lambda}{\text{argmax}} \sum_{i=1}^{n} \sum_{j=1}^{g} Z_{ij}^{(0)} \log \lambda_j$$

which has the solution

$$\lambda_j^{(1)} = \sum_{i=1}^{n} \frac{Z_{ij}^{(0)}}{n}, \qquad j = 1, \ldots, g. \tag{12}$$

We can maximize over θ by maximizing the first term in (11) separately over each θ_j. To solve

$$\theta_j^{(1)} = \underset{\theta_j}{\text{argmax}} \sum_{i=1}^{n} Z_{ij}^{(0)} \log p(X_i|\theta_j), \qquad j = 1, \ldots, g \tag{13}$$

we need to know the form of $p(X_i|\theta_j)$. The MM algorithm assumes that the mixture contains two classes ($g = 2$) and the distributions for class 1 (the motif) and class 2 (the background) are

$$p(X_i|\theta_1) = \prod_{j=1}^{W} \prod_{k=1}^{L} f_{jk}^{I(k,X_{ij})}, \text{ and} \tag{14}$$

$$p(X_i|\theta_2) = \prod_{j=1}^{W} \prod_{k=1}^{L} f_{0k}^{I(k,X_{ij})}, \tag{15}$$

where X_{ij} is the letter in the jth position of sample X_i, and $I(k, a)$ is an indicator function which is 1 if and only if $a = a_k$. That is,

$$I(k, a) = \begin{cases} 1 & \text{if } a = a_k \\ 0 & \text{otherwise} \end{cases}$$

For $k = 1, \ldots, L$, let

$$c_{0k} = \sum_{i=1}^{n} \sum_{j=1}^{W} Z_{i2}^{(0)} I(k, X_{ij}) \text{ and} \tag{16}$$

$$c_{jk} = \sum_{i=1}^{n} E_i Z_{i1}^{(0)} I(k, X_{ij}), \text{ for } j = 1, \ldots, W. \tag{17}$$

Then c_{0k} is the expected number of times letter a_k appears in positions generated by the background model in the data, and $c_{jk},\ j = 1, \ldots, W$ is the expected number

of times letter a_k appears at position j in occurrences of the motif in the data.[1] We reestimate θ_1 by substituting (14) into the right-hand side of (13) yielding

$$\begin{aligned}
\theta^{(1)} &= (\hat{f}_0, \hat{f}_1, \ldots, \hat{f}_W) \\
&= \operatorname*{argmax}_{\theta} \sum_{i=1}^{n} Z_{i1}^{(0)} \log \left[\prod_{j=1}^{W} \prod_{k=1}^{L} f_{jk}^{I(k,X_{ij})} \right] \\
&= \operatorname*{argmax}_{\theta} \sum_{i=1}^{n} Z_{i1}^{(0)} \sum_{j=1}^{W} \sum_{k=1}^{L} I(k, X_{ij}) \log f_{jk} \\
&= \operatorname*{argmax}_{\theta} \sum_{j=1}^{W} \sum_{k=1}^{L} \log f_{jk} \sum_{i=1}^{n} Z_{i1}^{(0)} I(k, X_{ij}) \\
&= \operatorname*{argmax}_{\theta} \sum_{j=1}^{W} \sum_{k=1}^{L} c_{jk} \log f_{jk}. \qquad (18)
\end{aligned}$$

So,

$$\hat{f}_{jk} = \frac{c_{jk}}{\sum_{k=1}^{L} c_{jk}}, \quad j = 0, \ldots, W \text{ and } k = 1, \ldots, L. \qquad (19)$$

Estimating the parameters of a multinomial random variable by maximum likelihood is subject to boundary problems. If any letter frequency $\hat{f}_{ij}$ ever becomes 0, as is prone to happen in small datasets, its value can never change. Following Brown *et al.* [1993] and Lawrence *et al.* [1993], the equations above for $\hat{f}_{ij}$ are replaced by

$$\hat{f}_{ij} = \frac{c_{ij} + \beta_j}{\sum_{k=1}^{L} c_{ik} + \beta}, \; i = 0, \ldots, W, \; j = 1, \ldots, L, \; \beta = \sum_{k=1}^{L} \beta_k. \qquad (20)$$

This turns out to be equivalent to using the Bayes estimate for the value of θ under squared-error loss (SEL) [Santner and Duffy, 1989] assuming that the prior distribution of each θ_j, $P(\theta_j)$, is the so called Dirichlet distribution with parameter $\beta' = (\beta_1, \ldots, \beta_L)$. The value of β' must be chosen by the user depending on what information is available about the distribution of θ_j for motifs and for the background. The choice of β' will be discussed in Section 4.

[1] The factor E_i in the calculation of the motif counts is the "erasing factor" for that position in the data. Erasing is mentioned in the introduction and further described in the next section. The erasing factors vary between 1 and 0 and are set to 1 initially. After each pass, they are reduced by a factor between 0 and 1 representing the probability that the position is contained in an occurrence of the motif found on that pass. The counts for the background are not scaled by the erasing factors to make the values of the *log likelihood* function comparable among passes.

3.2 Computing the expected likelihood of a model

The *log likelihood* reported in this paper is actually the expected *log likelihood* over the conditional distribution of the missing data Z,

$$E[\log L(\theta, \lambda|X, Z)] = \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log p(X_i|\theta_j) + \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log \lambda_j. \quad (21)$$

We can also compute the *log likelihood* of a particular model given just the data as

$$\begin{aligned} \log L(\theta, \lambda|X) &= \log p(X|\theta, \lambda) \\ &= \log \prod_{i=1}^{n} p(X_i|\theta, \lambda) \\ &= \log \prod_{i=1}^{n}\sum_{j=1}^{2} p(X_i|\theta_j)\lambda_j. \end{aligned} \quad (22)$$

These two values are not the same since, in general,

$$L(\theta, \lambda|X, Z) < L(\theta, \lambda|X).$$

This can be seen from the fact that

$$\begin{aligned} \log L(\theta, \lambda|X, Z) &= \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij} \log(p(X_i|\theta_j)\lambda_j) \\ &\leq \sum_{i=1}^{n}\sum_{j=1}^{2} \log(p(X_i|\theta_j)\lambda_j) \\ &\leq \sum_{i=1}^{n} \log \sum_{j=1}^{2} p(X_i|\theta_j)\lambda_j \\ &= \log L(\theta, \lambda|X). \end{aligned}$$

So, $E[\log L(\theta, \lambda|X, Z)]$ will tend to underestimate $\log L(\theta, \lambda|X)$. In practice, however, they tend to be close. At any rate, since MM maximizes (21), this is what is calculated and output by the implementation of the MM algorithm which is described in Section 4. The remainder of the current section will show that (21) for a model $(\hat{\theta}, \hat{\lambda})$ discovered by running MM to convergence can be approximated as

$$E[\log L(\hat{\theta}, \hat{\lambda}|X, Z)] \approx n[\sum_{k=1}^{L}(\hat{\lambda}_1 \sum_{j=1}^{W} \hat{f}_{jk} \log \hat{f}_{jk} + \hat{\lambda}_2 W \hat{f}_{0k} \log \hat{f}_{0k}) + \sum_{j=1}^{2} \hat{\lambda}_j \log \hat{\lambda}_j].$$

The values reported as *log likelihood* in this paper were computed using this approximation.

To make the derivation of the above approximation clearer, we separate the right-hand side of

$$E[\log L(\theta, \lambda|X, Z)] = \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log p(X_i|\theta_j) + \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log \lambda_j \quad (23)$$

into two terms

$$L_\theta(\theta, \lambda) = \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log p(X_i|\theta_j) \quad \text{and} \quad (24)$$

$$L_\lambda(\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \log \lambda_j. \quad (25)$$

We further split (24) into separate terms corresponding to the motif and background components

$$L_{\theta_1}(\theta, \lambda) = \sum_{i=1}^{n} Z_{i1}^{(0)} \log p(X_i|\theta_1) \quad \text{and} \quad (26)$$

$$L_{\theta_2}(\theta, \lambda) = \sum_{i=1}^{n} Z_{i2}^{(0)} \log p(X_i|\theta_2). \quad (27)$$

Substituting the known form for the motif component $p(X_i|\theta_1)$,

$$p(X_i|\theta_1) = \prod_{j=1}^{W}\prod_{k=1}^{L} f_{jk}^{I(k,X_{ij})} \quad (28)$$

into (26) we have

$$\begin{aligned} L_{\theta_1}(\theta, \lambda) &= \sum_{i=1}^{n} Z_{i1}^{(0)} \log p(X_i|\theta_1) \\ &= \sum_{i=1}^{n} Z_{i1}^{(0)} \log(\prod_{j=1}^{W}\prod_{k=1}^{L} f_{jk}^{I(k,X_{ij})}) \\ &= \sum_{i=1}^{n} Z_{i1}^{(0)} \sum_{j=1}^{W}\sum_{k=1}^{L} \log(f_{jk}^{I(k,X_{ij})}) \\ &= \sum_{i=1}^{n} Z_{i1}^{(0)} \sum_{j=1}^{W}\sum_{k=1}^{L} I(k, X_{ij}) \log f_{jk} \\ &= \sum_{k=1}^{L}\sum_{j=1}^{W} \log f_{jk} \sum_{i=1}^{n} Z_{i1}^{(0)} I(k, X_{ij}). \end{aligned} \quad (29)$$

Recall that the expected counts for letters in the positions of the motif are

$$c_{jk} = \sum_{i=1}^{n} E_i Z_{i1}^{(0)} I(k, X_{ij}), \quad \text{for } j = 1, \ldots, W. \quad (30)$$

Assuming for the moment the "erasing factors" E_i for $i = 1, \ldots, n$ are all equal to 1, we can substitute for Z_{ij} in (29) yielding

$$L_{\theta_1}(\theta, \lambda) \quad = \quad \sum_{k=1}^{L} \sum_{j=1}^{W} c_{jk} \log f_{jk}. \tag{31}$$

Now we recall that $\theta = (f_0, f_1, \ldots, f_W)$ is reestimated as

$$\hat{f}_{jk} \quad \approx \quad \frac{c_{jk}}{\sum_{k=1}^{L} c_{jk}} \quad \text{for } j = 0, \ldots, W \text{ and } k = 1, \ldots, L. \tag{32}$$

(The approximation is due to the fact that the actual formula for updating θ includes the pseudo-counts β_j. These tend to be small in practice, so the approximation should be close to the actual value.) We can write (31), evaluated at the model discovered by MM, $(\hat{\theta}, \hat{\lambda})$, as

$$L_{\theta_1}(\hat{\theta}, \hat{\lambda}) \quad \approx \quad \sum_{k=1}^{L} \sum_{j=1}^{W} S_j \hat{f}_{jk} \log \hat{f}_{jk}, \tag{33}$$

where

$$S_j \quad = \quad \sum_{k=1}^{L} c_{jk} \quad \text{for } j = 0, \ldots, W. \tag{34}$$

Recalling how λ is reestimated,

$$\lambda_j^{(1)} \quad = \quad \sum_{i=1}^{n} \frac{Z_{ij}^{(0)}}{n}, \qquad j = 1, 2, \tag{35}$$

we can rewrite S_j as

$$\begin{aligned} S_j \quad &= \quad \sum_{k=1}^{L} \sum_{i=1}^{n} Z_{i1}^{(0)} I(k, X_{ij}) \\ &= \quad \sum_{i=1}^{n} Z_{i1}^{(0)} \sum_{k=1}^{L} I(k, X_{ij}) \\ &= \quad \sum_{i=1}^{n} Z_{i1}^{(0)} \\ &= \quad n\lambda_1^{(1)} \\ &\approx \quad n\lambda_1^{(0)} \text{ for } j = 1, \ldots, W. \end{aligned} \tag{36}$$

(The last approximation should be very close since MM has converged.) We can now write (33) entirely in terms of $\hat{\theta} = (\theta_1^{(0)}, \theta_2^{(0)})$ and $\hat{\lambda} = (\lambda_1^{(0)}, \lambda_2^{(0)})$ as

$$L_{\theta_1}(\hat{\theta}, \hat{\lambda}) \quad \approx \quad n\hat{\lambda}_1 \sum_{k=1}^{L} \sum_{j=1}^{W} \hat{f}_{jk} \log \hat{f}_{jk}, \tag{37}$$

How the "erasing factors" used in the calculation of the counts c_{ij} in (30) are updated is discussed in Section 4. Anticipating that discussion somewhat, the assumption that all $E_i = 1$ is correct while the first motif in a dataset is being discovered using MM. As each motif is discovered, the values of the E_i corresponding to occurrences of motifs found so far are reduced towards 0, effectively erasing the motif occurrences from the dataset. This will cause the value calculated for $L_{\theta_1}(\theta, \lambda)$ be somewhat lower than the true value for the second and later motifs found in a given dataset. This effect should be very small, however, unless two very similar motifs are found. This is exactly what the erasing factors are intended to prevent from happening, so this problem is minimal in practice.

We can compute the value of $L_{\theta_2}(\hat{\theta}, \hat{\lambda})$ in a completely analogous fashion. When we substitute the known form of the background component

$$p(X_i|\theta_2) = \prod_{j=1}^{W} \prod_{k=1}^{L} f_{0k}^{I(k,X_{ij})} \tag{38}$$

into (27) we have

$$\begin{aligned} L_{\theta_2}(\theta, \lambda) &= \sum_{i=1}^{n} Z_{i2}^{(0)} \log p(X_i|\theta_2) \\ &= \sum_{i=1}^{n} Z_{i2}^{(0)} \log(\prod_{j=1}^{W} \prod_{k=1}^{L} f_{0k}^{I(k,X_{ij})}) \\ &= \sum_{i=1}^{n} Z_{i2}^{(0)} \sum_{j=1}^{W} \sum_{k=1}^{L} \log(f_{0k}^{I(k,X_{ij})}) \\ &= \sum_{i=1}^{n} Z_{i2}^{(0)} \sum_{j=1}^{W} \sum_{k=1}^{L} I(k, X_{ij}) \log f_{0k} \\ &= \sum_{k=1}^{L} \log f_{0k} \sum_{j=1}^{W} \sum_{i=1}^{n} Z_{i2}^{(0)} I(k, X_{ij}). \end{aligned} \tag{39}$$

We recall the expected counts for letters in the background component are

$$c_{0k} = \sum_{i=1}^{n} \sum_{j=1}^{W} Z_{i2}^{(0)} I(k, X_{ij}) \tag{40}$$

so, using (32) and (34) we have

$$\begin{aligned} L_{\theta_2}(\hat{\theta}, \hat{\lambda}) &= \sum_{k=1}^{L} c_{0k} \log \hat{f}_{0k} \\ &\approx \sum_{k=1}^{L} S_0 \hat{f}_{0k} \log \hat{f}_{0k}. \end{aligned} \tag{41}$$

We note that

$$\begin{aligned} S_0 &= \sum_{k=1}^{L} c_{0k} \\ &= \sum_{k=1}^{L}\sum_{i=1}^{n}\sum_{j=1}^{W} Z_{i2}^{(0)} I(k, X_{ij}) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{W} Z_{i2}^{(0)} \sum_{k=1}^{L} I(k, X_{ij}) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{W} Z_{i2}^{(0)} \\ &= W\sum_{i=1}^{n} Z_{i2}^{(0)} \\ &= Wn\lambda_2^{(1)} \\ &\approx Wn\lambda_2^{(0)} \end{aligned} \tag{42}$$

so

$$L_{\theta_2}(\hat{\theta}, \hat{\lambda}) \approx Wn\hat{\lambda}_2 \sum_{k=1}^{L} \hat{f}_{0k} \log \hat{f}_{0k}. \tag{43}$$

The value of $L_\lambda(\hat{\lambda})$ is easy to calculate from the definition of $\lambda^{(1)}$ given in (35). Once again assuming that because MM has converged $\lambda^{(0)} \approx \lambda^{(1)}$, we can write

$$\begin{aligned} L_\lambda(\hat{\lambda}) &= \sum_{i=1}^{n}\sum_{j=1}^{2} Z_{ij}^{(0)} \\ &= \sum_{j=1}^{2} \log \lambda_j^{(0)} \sum_{i=1}^{n} Z_{ij}^{(0)} \\ &= \sum_{j=1}^{2} \log \lambda_j^{(0)} n\lambda_j^{(1)} \\ &\approx \sum_{j=1}^{2} n\lambda_j^{(0)} \log \lambda_j^{(0)} \\ &\approx n\sum_{j=1}^{2} \hat{\lambda}_j \log \hat{\lambda}_j \end{aligned} \tag{44}$$

All of this algebra enables us to write the desired result

$$\begin{aligned} E[\log L(\hat{\theta}, \hat{\lambda}|X, Z)] &= L_\theta(\hat{\theta}, \hat{\lambda}) + L_\lambda(\hat{\lambda}) \\ &= L_{\theta_1}(\hat{\theta}, \hat{\lambda}) + L_{\theta_2}(\hat{\theta}, \hat{\lambda}) + L_\lambda(\hat{\lambda}) \end{aligned}$$

$$\approx \sum_{k=1}^{L}(n\hat{\lambda}_1 \sum_{j=1}^{W} \hat{f}_{jk} \log \hat{f}_{jk} + Wn\hat{\lambda}_2 \hat{f}_{0k} \log \hat{f}_{0k}) + n \sum_{j=1}^{2} \hat{\lambda}_j \log \hat{\lambda}_j$$

$$\approx n[\sum_{k=1}^{L}(\hat{\lambda}_1 \sum_{j=1}^{W} \hat{f}_{jk} \log \hat{f}_{jk} + \hat{\lambda}_2 W \hat{f}_{0k} \log \hat{f}_{0k}) + \sum_{j=1}^{2} \hat{\lambda}_j \log \hat{\lambda}_j].$$

If we had been smart and let λ_0 be the prior probability of the background component, and let λ_1 be the prior on the motif component, we could have written

$$E[\log L(\hat{\theta}, \hat{\lambda}|X, Z)] \approx n \sum_{j=0}^{1}(\hat{\lambda}_j \log \hat{\lambda}_j + \sum_{k=1}^{L} \sum_{i=1}^{W} \hat{\lambda}_j \hat{f}_{(i\cdot j)k} \log \hat{f}_{(i\cdot j)k}).$$

4 Implementation of MM

The MM algorithm is implemented as an option to the MEME+ software for discovering multiple motifs in biopolymer sequences. MEME+ is based on the MEME program described in [Bailey and Elkan, 1993]. The next three sections describe the MEME+ algorithm, its output, and how starting points for MM are selected. The last section also discusses the execution time required by MEME+.

4.1 The MEME+ algorithm

The MEME+ algorithm is shown in Figure 1. Recall that n is the number of overlapping length-W subsequences in the entire dataset and N is the number of sequences in the dataset. Motifs of width W are searched for in the outer loop of the algorithm. Within that loop, various values of $\lambda^{(0)}$ are tried. The limits on $\lambda^{(0)}$ correspond to there being as few as $\sqrt{N}$ occurrences of a motif to as many as one half of the (non-overlapping) subsequences in the dataset being motif occurrences. The sampling of $\lambda^{(0)}$ is done in a geometrically increasing series because experiments showed that starting points where $\lambda^{(0)}$ was within a factor of 2 of the correct value were usually sufficient for MM to converge (results not shown.) In the innermost loop, MEME+ uses $Q = \frac{log(1-\alpha)}{log(1-\lambda)}$ actually occurring subsequences to derive values for $\theta^{(0)}$ (see Section 4.4). The best starting point (as determined by the heuristic described in [Bailey and Elkan, 1993]) is then used with MM to discover a motif. The motif is then probabilistically "erased" (see [Bailey and Elkan, 1993]) and the outer loop repeats to find the next motif.

The implementation of the MM algorithm is straightforward. Let l_i, for $i = 1, \ldots, N$ be the lengths of the individual sequences in the dataset Y. The motif and background models are stored as an array of letter frequency vectors $\theta = f_0, \ldots, f_W$. The overlapping length-W subsequences in the dataset are numbered left-to-right and top-to-bottom from 1 to n. The $Z_{k1}^{(0)}$ for $k = 1, \ldots, n$ are stored in an array z_{ij} where $i = 1, \ldots, N$ and $j = 1, \ldots, l_i$ with z_{ij} holding the value of

procedure MEME+ (
Y (dataset of sequences)
W (width of motifs to search for)
$NPASSES$ (number of distinct motifs to search for))
for $i = 1$ **to** $NPASSES$ **do**
for $\lambda^{(0)} = \frac{\sqrt{N}}{n}$ **to** $\frac{1}{2W}$ **by** $\times 2$ **do**
for $j = 1$ **to** Q **do**
Randomly (w/o replacement) select a subsequence X_j
from dataset Y.
Derive $\theta^{(0)}$ from subsequence X_j.
Estimate goodness of ($\theta^{(0)}$, $\lambda^{(0)}$) as starting point for MM.
end
Run MM to convergence from best starting point found above.
end
Print best motif found above: $\hat{\theta}$, $\hat{\lambda}$.
"Erase" occurrences of best motif found above.
end
end

Figure 1: The MEME+ algorithm.

$Z_{k1}^{(0)}$ corresponding to the subsequence starting in column j of sequence Y_i in the dataset. MM repeatedly applies the E-step and the M-step of EM to update θ, λ and z until the change in θ (euclidean distance) falls below a user-specified threshold (by default 10^{-6}) or a user-specified maximum number of iterations (by default 1000) is reached.

The E-step updates the z array using Equation (10) and the mapping just described between Z_{i1} and z_{ij}. It sets

$$z_{ij} = \frac{\lambda p(Y_{ij}|\theta_1)}{\lambda p(Y_{ij}|\theta_1) + (1-\lambda)p(Y_{ij}|\theta_2)},$$

where Y_{ij} is the length-W subsequence starting at column j in sequence Y_i. The z_{ij} values for each sequence are then normalized to sum to at most 1 over any window of size W. This is done following the algorithm given in Bailey and Elkan [1993] to enforce the constraint that

$$\sum_{j=k}^{k+W-1} z_{ij} \leq 1, \quad \text{for } i = 1, \ldots, N \text{ and } k = 1, \ldots, l_i - W$$

This is done because otherwise there is a strong tendency for MM to converge to motif models which generate repeated strings of one or two letters like "AAAAAA" or "ATATAT". This tendency arises because the overlapping substrings in the new dataset are *not* independent. MM breaks up the actual sequences into overlapping subsequences of length W, which causes any repetitive sequence in the original dataset to give rise to many overlapping substrings in the new dataset which are highly similar to each other. To avoid this problem, an *ad hoc* normalization procedure reduces the values of z_{ij} of substrings which were adjacent in the original dataset so that, in any window of length W in a sequence, the sum of the z_{ij} values for the substrings starting in that window is less than or equal to 1.

The M-step reestimates λ and θ using Equations (12) and (20), respectively. The pseudo-counts $(\beta_1, \ldots, \beta_L)$ are set to $\beta\mu_i$, $i = 1, \ldots, L$, where β is a user specified parameter to MEME and μ_i is the average frequency of letter a_i in the dataset.

4.2 Output of MEME+

MEME+ outputs the value of the *log likelihood* function (Equation 11) and the relative entropy per column, RE/col, of each motif found. We define RE/col as

$$RE/col = \frac{1}{W} \sum_{i=1}^{L} \sum_{j=1}^{W} \hat{f}_{ij} \log \frac{\hat{f}_{ij}}{\hat{f}_{0j}}.$$

RE/col gives a measure of the "crispness" of a motif.

The output of MEME+ includes a log-odds matrix $spec$ and a threshold value t for each motif found. Together these form a Bayes-optimal classifier [Duda and

Hart, 1973] for the "zero-one" loss function. The log-odds matrix has L rows and W columns and is calculated as $spec_{ij} = \log(\hat{f}_{ij}/\hat{f}_{0j})$ for $i = 1, \ldots, W$ and $j = 1, \ldots, L$. The threshold t is set to $t = \log((1-\lambda_1)/\lambda_1)$. To use $spec$ and t as a classifier with a new dataset, each (overlapping) subsequence $x = \langle x_1, x_2, \ldots, x_n \rangle$ is given a score $s(x) = \sum_{j=1}^{W} \sum_{i=1}^{L} I(i, x_j) spec_{ij}$. It can be shown that $s(x) = \log(p(x|\theta_1)/p(x|\theta_2))$. Bayesian decision theory says to classify sample x as being an occurrence of the motif only if

$$\begin{aligned} s(x) &> \log(P(background)/P(motif)) \\ &= \log((1-\lambda_1)/\lambda_1) \\ &= t. \end{aligned}$$

The threshold for any other loss function can easily be found by scaling t. The scaled threshold should be $t + \log(r_{12} - r_{22})/(r_{21} - r_{11})$, where r_{ij} is the loss incurred for deciding class i when the correct class is j, and class 1 is the motif, class 2 the background.

4.3 Finding good starting points for MM

Initial values for θ_1 are selected by choosing values which are "close" to actually occurring subsequences in the dataset. A subsequence $x = (x_1, x_2, \ldots, x_W)$ in the dataset is converted to $\theta_1^{(0)} = (f_1, f_2, \ldots, f_W)$ column-by-column. In column i of subsequence x, if $x_i = a_j$, then we set $f_{ij} = m$ and $f_{ik} = (1-m)/(L-1),\ k = 1, \ldots, L,\ k \neq j$. The value for m is chosen so that the relative entropy of f_i with respect to the uniform distribution over all letters of the alphabet will be fraction γ of its maximum, where γ is a user-supplied value (default = 0.1). That is, we set

$$\sum_{k=1}^{L} f_{ik} \log \frac{f_{ik}}{p} = \frac{(L-1)(1-m)}{L-1} \log \frac{1-m}{p(L-1)} + m \log \frac{m}{p} \quad (45)$$

$$= (1-m) \log \frac{1-m}{p(L-1)} + m \log \frac{m}{p} \quad (46)$$

$$= (1-m) \log \frac{1-m}{L-1} - \log p + m \log m \quad (47)$$

equal to $-\gamma \log(p)$ where $p = 1/L$ and $-\gamma \log(p)$ is the maximum relative entropy of any distribution relative to the uniform distribution. This equation is solved numerically (binary search) for the value of m which satisfies it.

The MM algorithm is not guaranteed to find the maximum likelihood estimates of the paramters of the model, only a local maximum. Running MM from different starting points (*i.e.*, different initial values for the model parameters θ) can yield different solutions with varying likelihood values. It is usually necessary to run MM from several starting points, and pick the solution with the highest likelihood value. It is difficult to know when to stop.

In the case of biopolymer motifs, θ has very high dimension and the likelihood surface tends to be complex and have many local maxima. This makes the task of selecting starting points for MM especially important. MEME adopts the approach of deriving starting points from the actual subsequences of the dataset. The basic MEME approach is to convert each subsequence in the dataset into a starting point θ which is "close" to it. MM is then run for just one iteration on each such tentative starting point, and MM is then run to convergence from the starting point with the highest likelihood after one iteration.

Trying the starting points derived from all the subsequences in the dataset results in an algorithm whose execution time is $O((NM)^2W) = O(n^2W)$ where N is the number of sequences in the dataset, M is their average length, W is the width of the motif, and n is the number of overlapping length-W subsequences. If the dataset contains several occurrences of the motif, intuition says that it is unnecessary to try all of the subsequences as starting points. Just sampling a fraction of them will, with high probability, result in finding an actual occurrence of the motif.

To make this intuition more concrete, let S be the number of occurrences of the motif in the dataset. How many subsequences from the dataset must we (randomly) sample to insure that we sample an actual occurrence with probability $0 \leq \alpha < 1$? If we randomly select (without replacement) Q positions in the dataset (out of a possible $n = N(M - W) \approx NM$ positions), the probability that we do *not* select at least one occurrence is

$$P(\text{no occurrence chosen}) \leq (1 - S/n)^Q = (1 - \lambda)^Q$$

To make the probability that we choose at least one occurrence be $\geq \alpha$, we can choose Q so that

$$(1 - \lambda)^Q \leq 1 - \alpha$$

which happens when

$$Q \geq \frac{log(1 - \alpha)}{log(1 - \lambda)}.$$

So we sample $\min(n, Q)$ starting points from the dataset where $Q = \frac{log(1-\alpha)}{log(1-\lambda)}$, $0 \leq \alpha < 1$. For a given λ, the number of samples we need to take does not depend on the size of the dataset above some minimum dataset size such that $n \geq Q$. This gives considerable time savings for large datasets and for large values of λ. However, since MEME+ runs MM with several different values of λ (including very small ones), its execution time is still approximately $O((NM)^2W)$.

5 Experimental Results

We studied the performance of MEME+ on a number of datasets with different characteristics. The datasets are summarized in Table 1. Three of the datasets

consist of protein sequences and three consist of DNA sequences. Three contain a single known motif. One contains two known motifs, each of which occurs once in each sequence. One contains three known motifs, each of which occurs multiple times per sequence. And one contains two motifs each of which occurs in only about half of the sequences.

The protein datasets, lipocalin, hth, and farn, are described in [Lawrence *et al.*, 1993] and were used to test the Gibbs sampling algorithm described there. We reiterate briefly here. The lipocalin proteins bind small, hydrophobic ligands for a wide range of biological purposes. The dataset contains the five most divergent lipocalins with known 3D structure. The positions of the two motifs in each of the sequences in the lipocalin dataset are known from structural comparisons. The hth proteins contain occurrences of DNA-binding structures involved in gene regulation. The correct locations of occurrences of the motif are known from x-ray and nuclear magnetic resonance structures, or from substitution mutation experiments, or both. The farn dataset contains isoprenyl-protein transferases, essential components of the cytoplasmic signal transduction networks. No direct structural information is known for the proteins in the dataset, so we used the starting positions for the three motifs reported by Lawrence *et al.* [1993]. These starting positions agreed with the results of earlier sequence analysis work [Boguski *et al.*, 1992a], [Boguski *et al.*, 1992b].

The three DNA datasets, crp, lexa and crplexa, are described in [Bailey and Elkan, 1993] and were used to test MEME there. They contain DNA sequences from *E. coli*. The crp dataset contains known binding sites for CRP [Lawrence and Reilly, 1990], and a few sites which have been identified by similarity to the known motif. The lexa dataset sequences contain known binding sites for LexA [Hertz *et al.*, 1990], and some that have been identified by similarity to known sites. The crplexa dataset contains all the sequences in the crp and lexa datasets.

To analyze the the output of MEME+, we used the motifs found during a single run of MEME+ to classify the dataset from which they were learned and measured the $recall$ defined as tp/p and $precision$ defined as $tp/(tp + fp)$. Here, p is the number of known occurrences of a motif in the dataset ("positives"), tp is the number of correctly classified positives ("true positives"), and fp is the number of non-occurrences classified as occurences ("false positives"). These statistics can be used as estimates of the true precision and recall of the motif learned by MEME+ if it were used to find occurrences of the motif in a different dataset.[2]

[2]Each discovered motif was compared with each motif known to occur in the dataset. $recall$ and $precision$ are relative to the "closest" known motif where "closest" means highest $recall$. The comparison between each discovered motif and each known motif was done once for each possible shifting of the known motif a fixed number of positions, i, $|i| \leq \lfloor W/2 \rfloor$. MEME+ was thus credited with finding a motif even if the predicted occurrences were displaced a small, fixed amount.

dataset name	*type*	*number of sequences*	*sequence length (avg)*	W	*motif name*	*sites*	
						proven	*total*
lipocalin	protein	5	182	16	lipA	5	5
					lipB	5	5
hth	protein	30	239	18	hth	30	30
farn	protein	5	380	12	farnA	none	30
					farnB	none	26
					farnL	none	28
crp	DNA	18	105	20	crp	18	24
lexa	DNA	16	200	20	lexa	11	21
crplexa	DNA	34	150	20	crp	18	25
					lexa	11	21

Table 1: Overview of the contents of the datasets. Proven sites are those which have been shown to be occurrences of the motif by laboratory experiment (*i.e.*, footprinting, mutagenesis or structural analysis). Total sites include the proven sites as well as sites that have been reported in the literature but whose identification was based primarily on sequence similarity with known sites (*i.e.*, "putative" sites).

5.1 MEME+ discovers multiple motifs

MEME+ was run on each of the datasets in Table 1 for 5 passes with W set to the value in the table, sampling probability $\alpha = .99$, and prior parameter $\beta = 0.01$. All of the known motifs were found during the first pass(es) of MEME+. Table 2 summarizes the output of MEME+ on the datasets and shows the analysis of the motifs found. On the left of the table the *log likelihood* of the mixture model, the relative entropy per column of the motif model and $sites = tp + fp$ for each pass of MEME+ is shown. On the right, the name of known motif (if any) corresponding to the motif output by MEME+ how much the MEME+ motif was shifted relative to the known motif, and the recall and precision statistics are shown.

With the exception of the lipocalin dataset, MEME+ finds all the known motifs with recall and precision over 0.6 on its first passes. Many of the motifs found have much higher recall and precision, as can be seen in the table. The first motif found in the lipocalin dataset actually seems to be a combination of the two known motifs. This is why the second pass found a relatively poor version of the other known motif. Here, MEME+ is merging two known motifs into one, but is still able to find both. The farnB motif found on pass 2 with dataset farn has somewhat low recall (0.615), but pass 4 picks up the remainder of the known occurences of farnB. Here MEME+ is splitting a known motif into two sub-motifs, but the first sub-motif would probably be a good probe nevertheless. A similar splitting of a motif occurs

	Output of MEME+				*Analysis of discovered motifs*			
dataset	*pass*	*log likelihood*	RE/col	$sites$	*motif*	*shift*	$recall$	$precision$
lipocalin	1	-55013	1.740	14	lipA	0	1.000	0.357
	2	-55057	1.880	10	lipB	-2	0.400	0.200
	3	-55103	2.348	7	none			
	4	-55129	2.822	4	none			
	5	-55157	2.458	4	none			
hth	1	-496332	1.205	49	hth	0	0.933	0.571
	2	-496537	0.850	61	none			
	3	-496575	0.907	56	none			
	4	-496641	0.997	36	none			
	5	-496629	1.480	20	none			
farn	1	-92518	1.862	25	farnL	0	0.917	0.880
	2	-92585	2.070	19	farnB	-1	0.615	0.842
	3	-92569	1.461	34	farnA	0	0.733	0.647
	4	-92717	1.897	14	farnB	-1	0.192	0.357
	5	-92704	2.194	11	none			
crp	1	-60547	0.681	21	crp	1	0.792	0.905
	2	-60589	0.633	15	none			
	3	-60597	0.844	8	none			
	4	-60601	0.703	11	none			
	5	-60600	0.982	6	none			
lexa	1	-109155	0.873	26	lexa	-2	0.842	0.615
	2	-109302	0.594	26	lexa	1	0.211	0.154
	3	-109307	0.667	19	none			
	4	-109304	0.865	13	none			
	5	-109318	0.782	12	none			
crplexa	1	-169923	0.949	23	lexa	0	0.842	0.696
	2	-170048	0.600	34	crp	-1	0.667	0.471
	3	-170065	0.623	24	none			
	4	-170080	0.579	30	lexa	3	0.316	0.200
	5	-170062	0.901	11	none			

Table 2: MEME+ was run with for 5 passes on each of the datasets in described in Table 1. The left columns in this table show the *log likelihood*, RE/col and number of occurrences of the motif ("sites") discovered by MEME+ during each pass. *log likelihood* values closer to zero indicate motifs which are more statistically significant. The right columns show how the discovered motifs match the known motifs in the datasets. The name of the known motif most closely matched by the motif output by MEME+ is shown, along with any "phase shift". The $recall$ and $precision$ scores of each discovered motif when it is used as a probe for the occurrences of the known motif in the dataset are also shown. MEME+ found good representations of all the known motifs in each of the datasets and provided a good thresholds for using the motifs as a probes.

with the lexa motif.

In actual use with datasets containing unknown motifs, the right-hand side of Table 2 would not be available to the researcher. The value of the *log likelihood* function seems to be a good indicator of when a "real" (*i.e.*, biologically significant) motif has been discovered by MEME+. This conclusion is based on the observation that the motifs found by MEME+ that correspond to known motifs tend to have much higher likelihoods than the motifs found on other passes. When one or more passes of MEME+ find motifs with likelihoods far higher than the other passes, this may be good evidence that those motifs are significant. (All the known motifs found by MEME+ seem to be significant by this criterion. The one marginal case is the crp motif in dataset crplexa.) In actual practice, therefore, it is a good idea to run MEME+ for several more passes than the number of anticipated motifs in the dataset. The *log likelihood* of the motifs found in later passes may give an idea of what the *log likelihood* of a "random" motif would be in the dataset at hand.

In no case did MEME+ find a motif with both perfect recall and precision. A natural question is whether this is due to the algorithm failing to find the motif with the highest *log likelihood* (*i.e.*, getting stuck at a local optimum), or whether the the known motif actually has lower *log likelihood* (under the assumptions of the model used by MEME+) than the motif found. Table 3 shows that the motifs discovered by MEME+ often have significantly higher *log likelihood* than the known motifs. In particular, the partial merging of the two motifs in dataset lipocalin mentioned above yields a merged motif with higher *log likelihood* than known motif lipA, and the "remainder" motif also has higher *log likelihood* than motif lipB. In only one case did the discovered motif have significantly lower *log likelihood*. This was the aforementioned farnB motif in dataset farn. Since the known farnB has much higher likelihood than the discovered one, it is clear that MEME+ got stuck at a local optimum in this case.

5.2 Sensitivity to prior

The "size" of the prior on θ_1, β', is a user-specified parameter to the MEME+ algorithm. It is therefore of interest if the choice of β' is important to the performance of MEME+. To ascertain this, MEME+ was run on the biological datasets in Table 1 with all input parameters held constant except for β'. These tests showed that MEME+ is extremely insensitive to the value chosen for β' in the range 0 to 0.001. Outside of this range several things can happen. Referring to Figures 2 and 3, several patterns can be discerned when $\beta' > .001$. When a single known motif is present in the dataset, as in the hth crp and lexa datasets, $recall$ tends to increase monotonically with β' at the same time that $precision$ decreases. This is not surprising, since large values of β' tend to "blur" θ_1 by increasing the size of the pseudo-counts added to each cell when θ_1 is updated during EM. This blurring effect will tend to make the motif more general which increases $recall$ and decreases $precision$. When two

dataset	*pass*	*motif*	*log likelihood of known motif (k)*	*log likelihood of discovered motif (d)*	*difference* $(d-k)$
lipocalin	1	lipA	-55090	-55013	77
	2	lipB	-55092	-55057	35
hth	1	hth	-496346	-496332	14
farn	1	farnL	-92525	-92518	7
	2	farnB	-92517	-92585	-68
	3	farnA	-92566	-92569	-3
crp	1	crp	-60590	-60547	43
lexa	1	lexa	-109147	-109155	-8
crplexa	1	lexa	-169918	-169923	-5
	2	crp	-170116	-170048	68

Table 3: A comparison of the *log likelihood* of known motifs and motifs found by MEME+ shows that MEME+ usually finds motifs whose statistical significance is at least as high as the known motifs. In only one case– the farnB motif in datatset farn– does MEME+ fail to discover a motif whose *log likelihood* is close to or significantly higher than that of the known motif.

or more motifs are present in the dataset, the value of $recall$ of one or all of the discovered motifs tends to decrease with β' when $\beta' > 0.001$, while $precision$ tends to decrease for all the discovered motifs. The decrease in $recall$ for some discovered motifs is probably the result of two or more known motifs getting partially merged into one as β' gets large. The lipocalin dataset was somewhat of an anomaly. For one value of β', MEME+ gets stuck in a local optimum and finds a very good version of the lipB motif. For some large values of β' the $recall$ of the lipB motif improves as well. These effects can be ascribed to the small number (5 each) of occurences of each of the known motifs in the dataset.

The conclusion is that for many datasets, very low values of β' (*i.e.*, $\beta' \approx 0.001$) are sufficient to ensure that MEME+ finds the motifs present.

5.3 Sensitivity to noise

One form of "noise" to which MEME+ may be subjected is sequences in the dataset which do not contain the motif. This is bound to occur when new datasets are being studied to see if they contain any new motifs. It is extremely desirable that MEME+ be able to find any motif which may be present even though a large number of superfluous sequences may have been included in the dataset presented to it. Related to this problem is the fact that the motif occurrences may be short compared to the length of the sequences in the dataset. The longer the sequences, the more

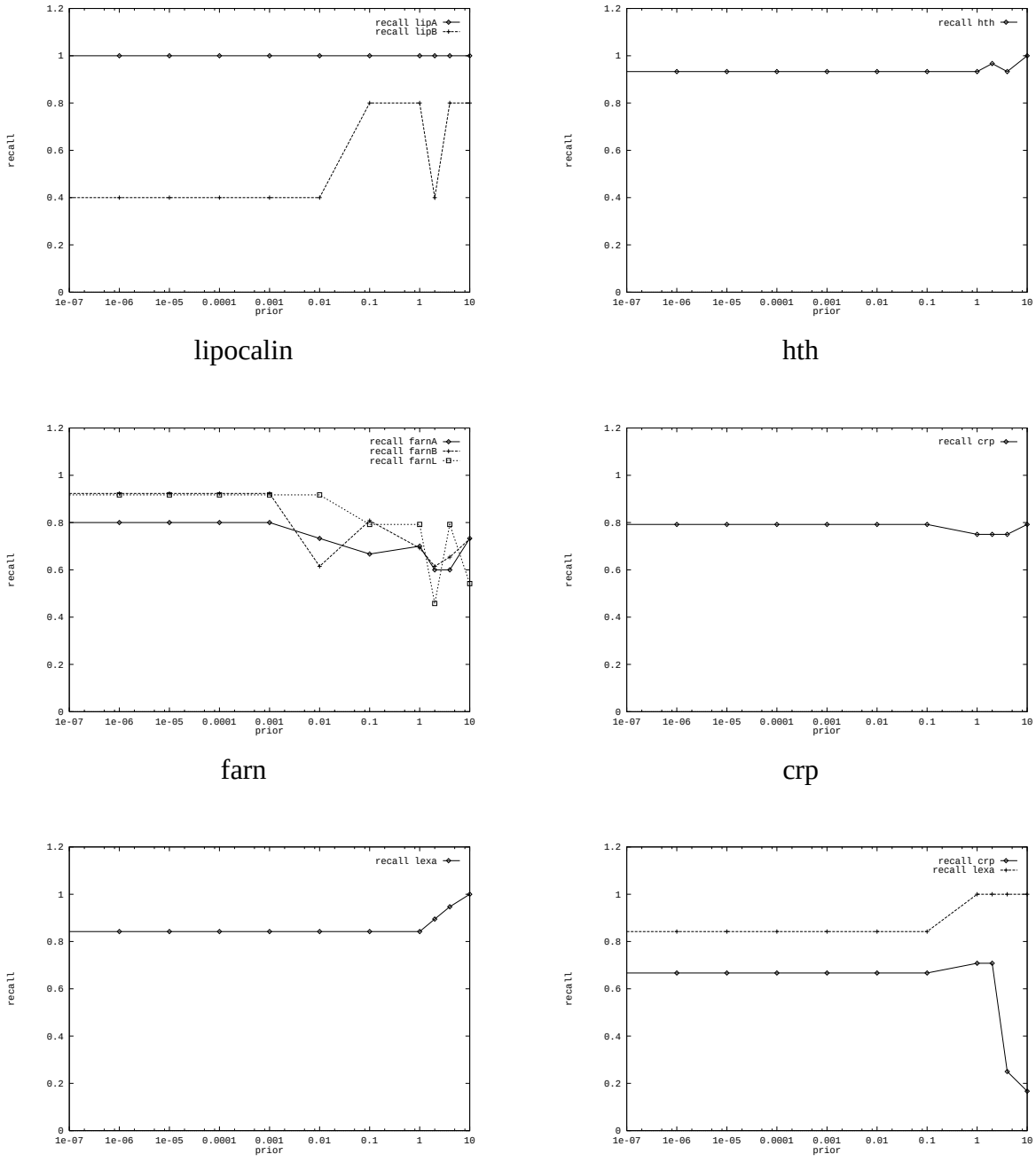
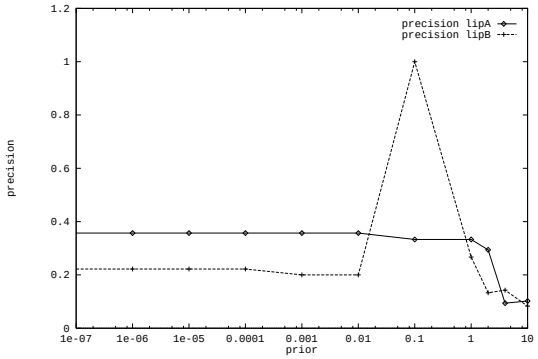
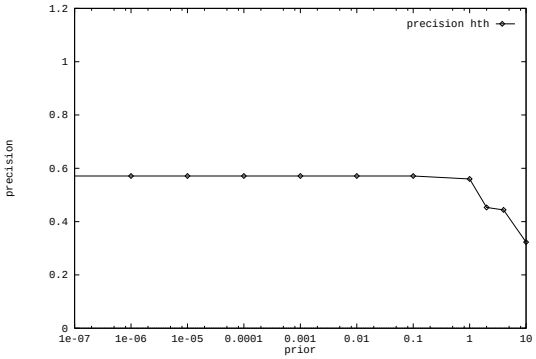


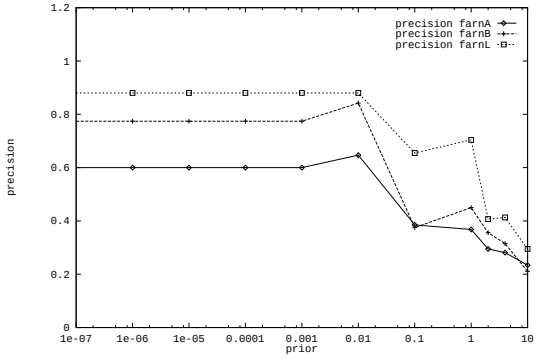
Figure 2: The value of the $recall$ statistic for motifs found by MEME+ run with different values for the size of the prior on θ_1, β', shows that $recall$ remains relatively constant for values of β' between 10^{-7} and 10^{-2}. It can be concluded that MEME+ is insensitive to the size of β' as long as it is small.



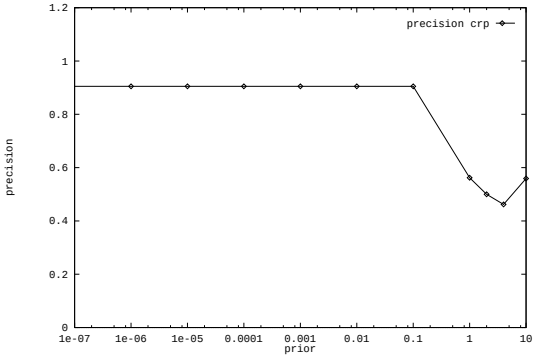
lipocalin



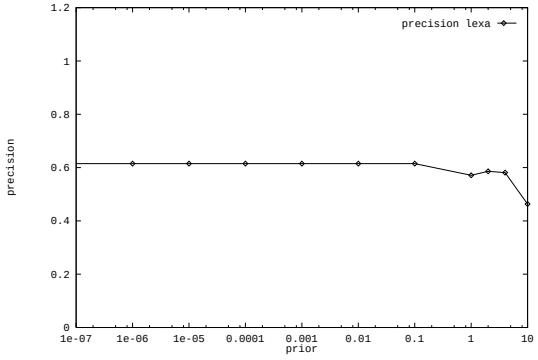
hth



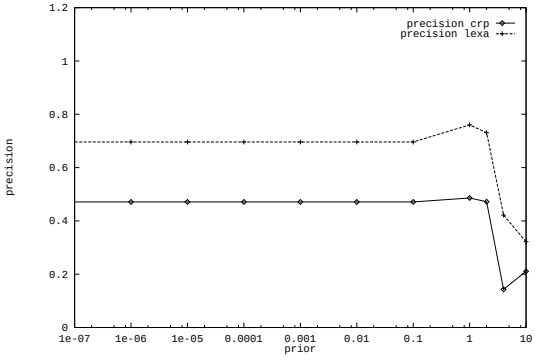
farn



crp



lexa



crplexa

Figure 3: The value of the $precision$ statistic for motifs found by MEME+ run with different values for the size of the prior on θ_1, β', shows that $precision$, like $recall$ is unaffected by β' if $beta' < .01$.

difficult it will be for MEME+ to discover the relevant motif occurrences. In this sense, all non-motif-occurrence positions in the dataset can be thought of as noise.

To study this problem further, we created additional datasets by adding pseudo-random sequences to the crp and lexa datasets. The added sequences had the same average frequencies of the four letters A, C, G and T, but were generated at random. Figure 4 shows that MEME+ had no difficulty finding the lexa motif even when 80 pseudo-random sequences of length 200 were added to the lexa dataset. This is quite impressive performance considering that the there are only 11 proven and 21 total (proven + putative) sites of length 16 in the lexa dataset. On the other hand, the 24 total sites in the crp dataset appear to be harder to find. The $recall$ and $precision$ of the discovered motif was lowered somewhat when up to 20 sequences of 105 pseudo-random bases were added. When more random sequences were added, MEME+ sometimes found the known motif, but sometimes did not, as can be seen in the figure. The explanation for this can be found in Table 4 which shows that the known crp motif has lower *log likelihood* than even the discovered motifs which had very poor $recall$ and $precision$. The lexa motifs discovered in the noisy datasets have likelihood very close to that of the known motif.

It can be concluded that MEME+ can tolerate an enormous amount of noise for some motifs, but weaker motifs require less noise in the dataset to ensure that MEME+ will be successful.

5.4 Sensitivity to subsampling

The user selectable parameter to MEME+ which determines the number of starting points tried with MM, α, affects the speed and accuracy of MEME+. A series of runs of MEME+ on the six datasets described earlier was made with varying values of α. The value of β was 0.01 in all cases.

The effect of α on the speed of MEME+ can be seen in Figure 5. Reducing α to .5 from 0.99 results in about an eight-fold speedup of MEME+ with relatively small decrease in the accuracy of the motifs found. The accuracy of MEME+ over a wide range of values of α can be seen from the high values for $recall$ and $precision$ in Table 5. The motifs were usually found on the first few passes of MEME+.

6 Discussion

The MM algorithm and its implementation in MEME+ have several important advantages over previously reported algorithms that perform similar tasks. This section first explains these advantages, and then discusses several limitations of MM and MEME+ the lifting of which would increase their usefulness for exploring collections of DNA and protein sequences.

The Gibbs sampling algorithm of Lawrence *et al.* [1993] is the most successful existing general algorithm for discovering motifs in sets of biosequences. MM

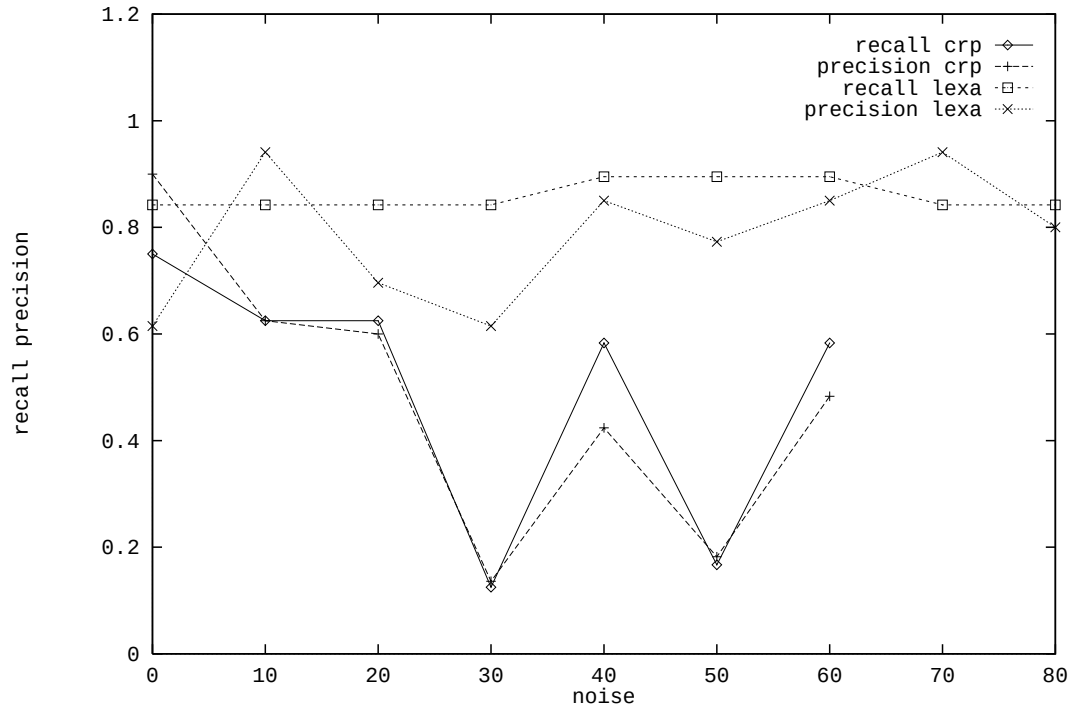


Figure 4: The value of the $recall$ and $precision$ statistics for the motifs found by MEME+ when it was run on the crp and lexa datasets with different numbers of random sequences added is plotted. The value of "noise" is the number of random sequences that were added to the crp and lexa datasets. Added sequences had the same length and average base frequencies as the sequences in the dataset to which they were added. MEME+ finds the lexa motif even when 85% of the dataset consists of random sequences. The ability of MEME+ to discover the crp motif degraded significantly when more than half the dataset was random sequences, but still found the motif occasionally with up to 77% of the dataset was random sequences. These results show the ability of MEME+ to discover motifs in datasets containing many sequences where the motif does not occur.

	Output of MEME+		*Analysis of discovered motifs*				
dataset name	*pass*	*log likelihood of discovered motif* (d)	*motif name*	$recall$	$precision$	*log likelihood of known motif* (k)	*difference* $(d-k)$
crp0	1	-60547	crp	0.750	0.900	-60590	43
crp10	1	-94216	crp	0.625	0.625	-94248	32
crp20	1	-128025	crp	0.625	0.600	-128081	56
crp30	4	-162354	crp	0.125	0.136	-162389	35
crp40	2	-196324	crp	0.583	0.424	-196377	53
crp50	4	-230075	crp	0.167	0.182	-230128	53
crp60	2	-264067	crp	0.583	0.483	-264133	66
lex0	1	-109155	lexa	0.842	0.615	-109147	-8
lex10	1	-180905	lexa	0.842	0.941	-180896	-9
lex20	1	-253117	lexa	0.842	0.696	-253109	-8
lex30	1	-325043	lexa	0.842	0.615	-325036	-7
lex40	1	-396893	lexa	0.895	0.850	-396875	-18
lex50	1	-468921	lexa	0.895	0.773	-468907	-14
lex60	1	-540897	lexa	0.895	0.850	-540881	-16
lex70	1	-612605	lexa	0.842	0.941	-612602	-3
lex80	1	-684284	lexa	0.842	0.800	-684280	-4

Table 4: The results of MEME+ on datasets created by adding random sequences to the lexa and crp datasets shows why the crp motif is hard to find. MEME+ consistently finds motifs with higher *log likelihood* than the known crp motif. The $recall$ and $precision$ statistics for these runs of MEME+ are shown in Figure 4.

	Analysis of discovered motifs				*Output of* MEME+			
prob	*motif*	*shift*	$recall$	$precision$	*pass*	*log likelihood*	RE/col	$sites$
0.5	lipA	0	0.600	0.429	3	-55113	2.249	6
	lipB	-4	0.400	0.125	1	-55027	1.497	15
0.8	lipA	-5	1.000	0.357	1	-55037	1.631	13
	lipB	-2	0.600	0.214	1	-55037	1.631	13
0.9	lipA	0	1.000	0.357	1	-55013	1.740	13
	lipB	-3	0.600	0.429	2	-55084	2.284	6
0.5	hth	3	0.867	0.531	1	-496390	1.127	48
0.8	hth	0	0.900	0.614	1	-496322	1.308	43
0.9	hth	0	0.900	0.614	1	-496322	1.308	43
0.5	farnA	-1	0.633	0.388	2	-92473	1.275	48
	farnL	-6	0.792	0.576	1	-92551	1.542	29
	farnB	4	0.423	0.846	4	-92687	2.066	12
0.8	farnA	-1	0.233	0.269	4	-92695	1.328	23
	farnL	-4	0.083	0.250	5	-92759	2.032	8
	farnB	-1	0.731	0.380	1	-92501	1.255	46
0.9	farnA	2	0.133	0.111	3	-92683	1.170	30
	farnL	0	0.500	0.800	4	-92652	2.040	14
	farnB	0	0.769	0.690	2	-92578	1.667	24
0.5	crp	2	0.667	0.727	1	-60554	0.660	19
0.8	crp	2	0.750	0.783	1	-60549	0.660	20
0.9	crp	2	0.750	0.783	1	-60549	0.660	20
0.5	lexa	0	0.947	0.947	1	-109166	1.109	17
0.8	lexa	0	0.947	0.947	1	-109166	1.109	17
0.9	lexa	-2	0.842	0.615	1	-109155	0.873	25
0.5	crp	-1	0.542	0.464	4	-170059	0.608	27
	lexa	-2	0.842	0.762	1	-169923	1.000	20
0.8	crp	1	0.583	0.538	2	-170043	0.691	21
	lexa	-1	0.842	0.762	1	-169922	1.002	20
0.9	crp	3	0.333	0.267	2	-170035	0.632	29
	lexa	0	0.947	0.818	1	-169932	1.003	20

Table 5: The best match of each discovered motif with a known motif when MEME+ is run with different values of α on datafile crplexa shows a general tendency for $recall$ and $precision$ to improve with increasing α. This is to be expected since running MEME+ with larger values α causes it to try more subsequences in the dataset as starting points for MM, thus increasing the chances of finding a good starting point. In most cases, however, MEME+ finds the known motifs well even when α is 0.5. Running MEME+ with smaller values of α requires less CPU time as can be seen in Figure 5.

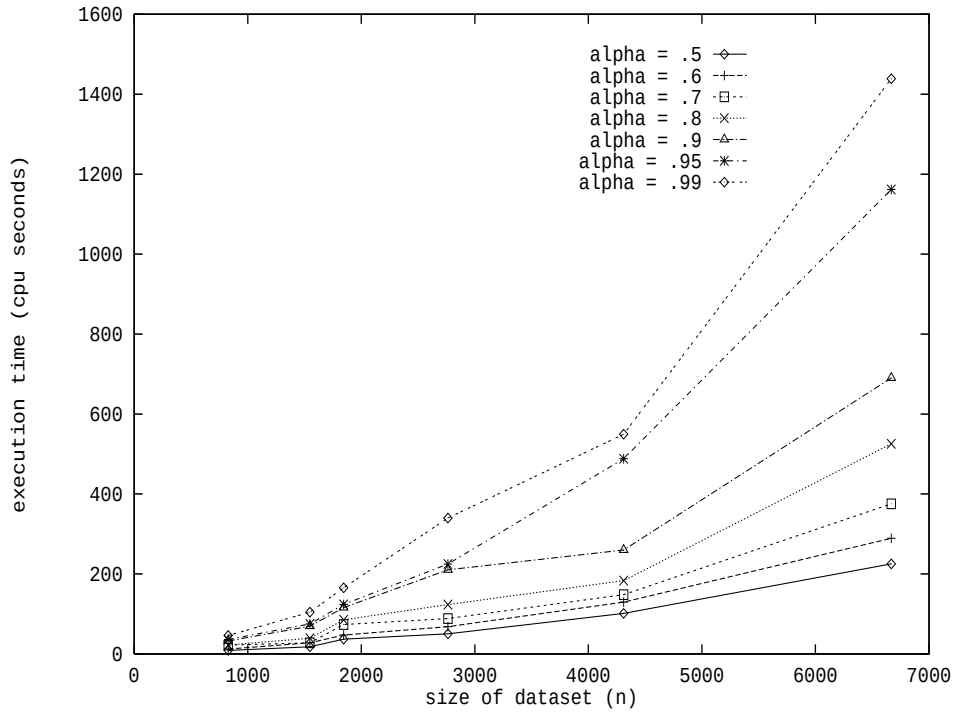


Figure 5: Speed of execution of MEME+ with different values of α is plotted. MEME+ was run with each of the datasets in Table 1, whose sizes ranged from about 1000 to about 7000 letters. The increase in speed gotten by reducing the value of α can be seen, as well as the generally quadratic dependence of execution time on the size of the dataset.

has two major advantages over this algorithm. First, MM does not require input sequences to be classified in advance by a biologist as definitely containing the motif that is being searched for. Instead, MM estimates from the data how many times a motif appears. This capability is quite robust: experiments show that even when only 20% of the sequences in a dataset contain a motif, the motif can often still be characterized well (see Section 5.4). Second, MM uses a formal probabilistic model of the entire input dataset, and systematically selects values for the parameters of this model that maximize the likelihood of the model. The MM model allows us to compare in a principled way the motif characterizations discovered by MEME+ and characterizations obtained by other methods. In most cases, the characterizations discovered by MEME have higher likelihood.

As pointed out by Lawrence *et al.* [1993] and by others, the fundamental practical difficulty in discovering motifs is the existence of many local optima in the search space of alternative motif models. The MM algorithm, like all expectation maximization applications, is a gradient descent method that cannot escape from

a local optimum. The MEME+ implementation of MM uses several heuristics to overcome the problem of local optima. These heuristics are all variations on a common theme, and should be useful in other applications also. The theme is to search the space of possible starting points for gradient descent *systematically*. By contrast, Gibbs sampling algorithms interleave gradient search steps with random jumps in the search space. These algorithms can spend an unpredictable number of iterations on a "plateau" before converging, whereas MM always converges in a predictable, relatively small number of iterations.

The focus of our current research is to overcome two significant limitations of MM and MEME+. The first of these is that all motifs found are constrained to have the same width, which is a parameter specified by the user. The main obstacle to estimating motif width endogenously is that likelihood values are not comparable for models that assume different motif widths.

The second limitation is that the number of different motifs present in a dataset is not estimated by the algorithm. We plan to overcome this limitation by generalizing from a two component mixture model to models with multiple components. A deep difficulty with multiple component models is that the induced search space is of even higher dimensionality than with two components, and local optima are even more pesky. Our current intention is to use the results of MEME+ as starting points for fitting models with multiple components. Doing this should have the additional benefit of allowing similar motifs discovered in different passes of MEME+ to be merged if overall likelihood is thereby increased.

References

[Aitkin and Rubin, 1985] Murray Aitkin and D. B. Rubin. Estimation and hypothesis testing in finite mixture models. *Journal of the Royal Statistical Society, B*, 47(1):67–75, 1985.

[Bailey and Elkan, 1993] Timothy L. Bailey and Charles Elkan. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. Technical Report CS93-302, Department of Computer Science, University of California, San Diego, August 1993.

[Boguski *et al.*, 1992a] M. S. Boguski, R. C. Hardison, S. Schwartz, and W. Miller. Analysis of conserved domains and sequence motifs in cellular regulatory proteins and locus control regions using new software tools for multiple alignment and visualization. *New Biologist*, 4(3):247–260, 1992.

[Boguski *et al.*, 1992b] M. S. Boguski, A. W. Murray, and S. Powers. Novel repetitive sequence motifs in the alpha and beta subunits of prenyl-protein transferases and homology of the alpha subunit to the MAD2 gene product of yeast. *New Biologist*, 4(4):408–411, 1992.

[Brown *et al.*, 1993] Michael Brown, Richard Hughey, Anders Krogh, I. Saira Mian, Kimmen Sjolander, and David Haussler. Using Dirichlet mixture priors to derive hidden Markov models for protein families. In *Intelligent Systems for Molecular Biology*, pages 47–55. AAAI Press, 1993.

[Duda and Hart, 1973] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., 1973.

[Gribskov *et al.*, 1990] Michael Gribskov, Roland Luthy, and David Eisenberg. Profile analysis. *Methods in Enzymology*, 183:146–159, 1990.

[Hertz *et al.*, 1990] Gerald Z. Hertz, George W. Hartzell, III, and Gary D. Stormo. Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Computer Applications in Biosciences*, 6(2):81–92, 1990.

[Lawrence and Reilly, 1990] Charles E. Lawrence and Andrew A. Reilly. An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *PROTEINS: Structure Function and Genetics*, 7:41–51, 1990.

[Lawrence *et al.*, 1993] Charles E. Lawrence, Stephen F. Altschul, Mark S. Boguski, Jun S. Liu, Andrew F. Neuwald, and John C. Wooton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 1993.

[Santner and Duffy, 1989] T. J. Santner and D. E. Duffy. *The Statistical Analysis of Discrete Data*. Springer Verlag, 1989.