The megaprior heuristic for discovering protein sequence patterns

Timothy L. Bailey and **Michael Gribskov**
San Diego Supercomputer Center
P.O. Box 85608
San Diego, California 92186-9784
{tbailey, gribskov}@sdsc.edu

Abstract

Several computer algorithms for discovering patterns in groups of protein sequences are in use that are based on fitting the parameters of a statistical model to a group of related sequences. These include hidden Markov model (HMM) algorithms for multiple sequence alignment, and the MEME and Gibbs sampler algorithms for discovering motifs. These algorithms are sometimes prone to producing models that are incorrect because two or more patterns have been combined. The statistical model produced in this situation is a convex combination (weighted average) of two or more different models. This paper presents a solution to the problem of convex combinations in the form of a heuristic based on using extremely low variance Dirichlet mixture priors as part of the statistical model. This heuristic, which we call the megaprior heuristic, increases the strength (i.e., decreases the variance) of the prior in proportion to the size of the sequence dataset. This causes each column in the final model to strongly resemble the mean of a single component of the prior, regardless of the size of the dataset. We describe the cause of the convex combination problem, analyze it mathematically, motivate and describe the implementation of the megaprior heuristic, and show how it can effectively eliminate the problem of convex combinations in protein sequence pattern discovery.

Keywords: sequence modeling; Dirichlet priors; expectation maximization; machine learning; protein motifs; hidden Markov models; unsupervised learning; sequence alignment, multiple

Introduction

A convex combination occurs when a model combines two or more sequence patterns that should be distinct. This can occur when a sequence pattern discovery algorithm tries to fit a model that is either too short (multiple alignment algorithms) or has too few components (motif discovery algorithms). This situation arises with HMM algorithms (Krogh *et al.* 1994; Baldi *et al.* 1994; Eddy 1995) when the model contains too few main-line states; with the Gibbs sampler motif discovery algorithm (Lawrence *et al.* 1993) when the user instructs the algorithm to assume sequences contain motif occurrences that in actuality they do not; and with the MEME motif discovery algorithm (Bailey and Elkan 1995a; 1995b), when the motif model chosen by the user does not assume that there is exactly one copy of the motif in each sequence in the training set. Since reducing the number of free parameters in the model is generally desirable, many pattern discovery algorithms use heuristics to minimize the length of the sequence model. If the heuristic shortens the sequence model too much, convex combinations can occur.

We use the term convex combination because, with the type of sequence model common to profiles, motifs and HMMs, the parameters of a model that erroneously combines distinct patterns are a weighted average of the parameters of the correct models, where the weights are positive and sum to one—in other words, a convex combination. Consider protein motifs, where a motif is an approximate, fixed-width, gapless pattern that occurs in a family of sequences or repeatedly in a single sequence. The commonly used protein motif model is a residue-frequency matrix, each of whose columns describes the observed frequencies of each of the twenty amino acids at that position in the motif. A convex combination model can be visualized by imagining aligning all the occurrences of two distinct (but equal width) protein sequence motifs and calculating the residue frequencies in each column. The resulting frequency matrix is a convex combination motif model.

An example convex combination motif model produced by a motif discovery algorithm is shown in Fig. 1. The training set, shown at the top of the figure, contains five protein sequences, each of which contains one occurrence of two distinct, known motifs. The residue-frequency matrix found by the algorithm is shown at the bottom of the figure (all frequencies are multiplied by ten and rounded to one digit; zeros are replaced with ":"). The residue-frequency matrix (the model) is a convex combination of models for the two known motifs. This can be seen by examining the the positions predicted as motif occurrences by the model, shown immediately above the residue-frequency matrix. Each of these is labeled as belonging to known

Training Set

```
ICYA_MANSE   1 gdifypgycpdvkpvnD FDLSAFAGAWHEIA Klplenenqgkctiaeyky
ICYA_MANSE  51 dgkkasvynsfvsngvkeymegdleiapdakytkqgkyvmtfkfgqrvvn
ICYA_MANSE 101 lv pWVLATDYKNYAIN YNCdyhpdkkahsihawilskskvlegntkevvd
ICYA_MANSE 151 nvlktfshlidaskfisndfseaacqysttysltgpdrh

LACB_BOVIN   1 mkclllalaltcgaqalivtqtmkG LDIQKVAGTWYSLA Maasdisllda
LACB_BOVIN  51 qsaplrvyveelkptpegdleillqkwengecaqkkiiaektkipavfki
LACB_BOVIN 101 dalnenkvLVLDTDYKKYLLFCMEnsaepeqslacqclvrtpevddeale
LACB_BOVIN 151 kfdkalkalpmhirlsfnptqleeqchi

BBP_PIEBR    1 nvyhdgacpevkpvdN FDWSNYHGKWWEVA Kypnsvekygkcgwaeytpe
BBP_PIEBR   51 gksvkvsnyhvihgkeyfiegtaypvgdskigkiyhkltyggvtkenv fN
BBP_PIEBR  101  VLSTDNKNYIIG YYCkydedkkghqdfvwvlsrskvltgeaktavenyli
BBP_PIEBR  151 gspvvdsqklvysdfseaackvn

RETB_BOVIN   1 erdcrvssfrvkeN FDKARFAGTWYAMA Kkdpeglflqdnivaefsvden
RETB_BOVIN  51 ghmsatakgrvrllnnwdvcadmvgtftdtedpakfkmkywgvasflqkg
RETB_BOVIN 101 nddhWIIDTDYETFAVQYSCrl lnldgtcadsysfv fardpsgfspevqk
RETB_BOVIN 151 ivrqrqeelclarqyrliphngycdgksernil

MUP2_MOUSE   1 mkmllllclgltlvcvhaeeasstgrN FNVEKINGEWHTII Lasdkreki
MUP2_MOUSE  51 edngnfrlfleqihvlekslvlkfhtvrdeecselsmvadktekageysv
MUP2_MOUSE 101 tydgfnt fTIPKTDYDNFLMA HLInekdgetfqlmglygrepdlssdike
MUP2_MOUSE 151 rfaklceehgilreniidlsnanrclqare
```

Aligned Fragments

```
(1) ICYA_MANSE  18  ycpdvkpvnD FDLSAFAGAWHEIA Klplenenqg
(2) ICYA_MANSE 103  kfgqrvvnlv pWVLATDYKNYAIN YNCdyhpdkk
(1) LACB_BOVIN  26  alivtqtmkG LDIQKVAGTWYSLA Maasdislld
(1) BBP_PIEBR   17  acpevkpvdN FDWSNYHGKWWEVA Kypnsvekyg
(2) BBP_PIEBR   99  tyggvtkenv fNVLSTDNKNYIIG YYCkydedkk
(1) RETB_BOVIN  15  rvssfrvkeN FDKARFAGTWYAMA Kkdpeglflq
(-) RETB_BOVIN 123  TFAVQYSCrl lnldgtcadsysfv fardpsgfsp
(1) MUP2_MOUSE  28  aeeasstgrN FNVEKINGEWHTII Lasdkrekie
(2) MUP2_MOUSE 108  ysvtydgfnt fTIPKTDYDNFLMA HLInekdget
```

Convex Combination Model

```
A  :::12:311::2:6
C  ::::::1:::::::
D  :4:1::3:2:::::
E  :::1::::1::2::
F  7::::2::::1:1:
G  ::::1::6:::::1
H  ::::::1:::2:::
I  ::2::1:::::141
K  ::1:3:::3:::::
L  2:22:::::::11:
M  ::::::::::::2:
N  :3::1:11:3:::1
P  1::1::::::::::
Q  :::1::::::::::
R  ::::1:::::::::
S  :::21::::1:2::
T  :1:::4::2::1::
V  ::3::1::::::11
W  :11::::::51:::
Y  :::::1:2::6:::
```

Figure 1: **Illustration of the convex combination problem.** Training set sequences (lipocalins taken from Lawrence *et al.* (1993)) are shown at the top of the figure with known (uppercase) and predicted (boxed) occurrences of the two known motifs indicated. Aligned sequence fragments containing the predicted occurrences and the (abbreviated) residue-frequency matrix of the convex combination model are shown at the bottom of the figure. Sequence fragments are labeled on the left with which known motif—(1), (2) or none (-)—they contain.

motif (1) or (2) on the left of the figure. The model identifies all the instances of one of the motifs as well as three of five instances of the other. The predicted starts of the occurrences of motif 1 are all shifted one position to the right of the known starts. Similarly, the three predicted motif 2 occurrences are shifted one position to the left of the known occurrences. This is typical of convex combination models since they tend to align motif columns that have similar residue frequencies in order to maximize the information content of the model.[1]

Convex combinations are undesirable because they distort multiple alignments and lead to motif descriptions that make unrelated sequence regions appear to be related. Eliminating them will greatly enhance the utility of automated algorithms for sequence pattern discovery. This paper presents a solution to the convex combination problem in the form of a heuristic based on the use of a mixture of Dirichlet distributions prior (Brown *et al.* 1993). This type of prior contains information about the types of residue-frequency vectors (i.e., residue-frequency matrix columns) that are biologically reasonable in a protein sequence model. By using priors with extremely low variance, the search for patterns can be strongly biased toward biologically reasonable patterns and away from convex combinations, which tend to be biologically unreasonable.

The organization of this paper is as follows. First, we discuss the problem of convex combinations when searching for motifs and show mathematically why it occurs. Next we give an overview of the use of Dirichlet mixture priors in discovering protein sequence models. We then describe the megaprior heuristic and discuss its implementation. The results section demonstrates the dramatic improvement in motif models found by MEME using the heuristic as a result of the elimination of convex combination motif models. In the last section, we discuss why the megaprior heuristic works so well and opportunities for utilizing the heuristic in other algorithms.

Convex combinations

The convex combination (CC) problem is most easily understood in the context of protein sequence motifs. A motif is a recurring sequence pattern that can be modeled by a position dependent residue-frequency matrix. Such a matrix is equivalent to a gapless profile (Gribskov *et al.* 1990)—a profile with infinite gap opening and extension costs. Each column in the frequency matrix describes the distribution of residues expected at that position in occurrences of the motif. If a large number of occurrences of the motif were aligned, we would expect to observe residue-frequencies in each column of the alignment approximately equal to the values in the corresponding column of the motif residue-frequency matrix.

One objective of motif analysis is to discover motif models that identify and describe regions critical to the function or folding of proteins. This is possible because certain regions (motif occurrences) of distantly related proteins tend to be conserved precisely because they are essential to the functioning or folding of the protein. Motif discovery proceeds by looking for a fixed-length sequence pattern that is present in several sequences that otherwise share little homology. This can be done manually, as was done in creating the Prosite dictionary of sequence patterns (Bairoch 1995), or automatically, using a computer algorithm such as MEME or the Gibbs sampler.

The CC problem occurs when automatic motif discovery algorithms such as MEME are given an inaccurate estimate (or no estimate) of the number of occurrences of the motif that are present in each sequence. The algorithm must then balance the conciseness of the motif model (its information content) with the amount of the data that it describes (its coverage). When the megaprior heuristic, to be described in detail later, is not used, the algorithm tends to select a model that is a combination of two or more models of distinct motifs. This is because, without constraints on the number or distribution of occurrences of the motif within the sequences, a convex combination can maximize the motif discovery algorithm's objective function by explaining more of the data using fewer free parameters than would a model of a single motif.

We can show mathematically why MEME chooses CC motif models. In its least constrained mode, MEME fits a mixture model to the sequences in the training set. To do this, it slices up the sequences into all their overlapping subsequences of length W, where W is the width of the motif. Suppose the sequences contain two width-W motifs, one consisting of all "a"s, and one of all "b"s. Suppose further that the rest of the sequences were essentially uniform random noise. A hidden Markov model with three-components like that in Fig. 2 would be appropriate in this case. This model generates "random" strings with probability λ_1, all "a"s with probability λ_2, and all "b"s with probability $\lambda_3 = 1 - \lambda_1 - \lambda_2$.

Learning the parameters of a multi-component model is difficult due to local optima in the likelihood surface. To minimize this problem, MEME learns the informative components of the motif model one-at-a-time. Using the expectation maximization algorithm (EM) (Dempster *et al.* 1977), MEME repeatedly fits a two-component mixture model to the subsequences generated from the data. We would like the algorithm to converge to the model shown in Fig. 3 (or a similar one modeling the all "b" component). The informative component of the model provides a description of one of the motifs (the strings of "a"s) in the dataset

[1] The model was produced by MEME without using the megaprior heuristic. Using the megaprior heuristic, MEME produces two distinct motif models each of which correctly describes one of the known motifs in the dataset.

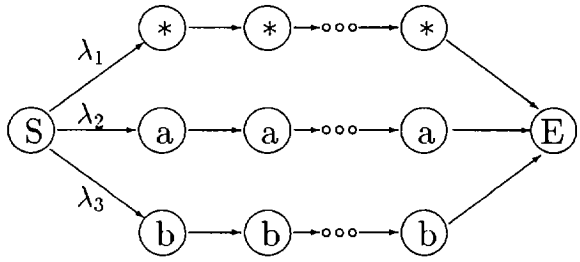

Figure 2: Hidden Markov model representation of a three-component mixture distribution which produces a mixture of strings of all "a"s, strings of all "b"s and uniformly random strings.

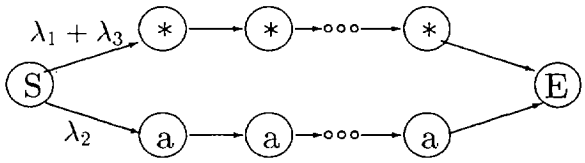

Figure 3: Desired two-component hidden Markov model that models one of the peaked components of the data in component two (lower path), and the rest of the data is modeled by the first component which is constrained to be uniformly random.

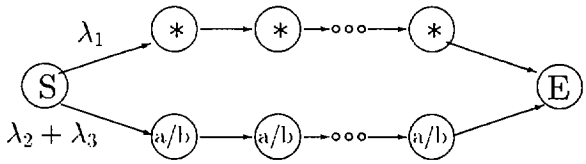

Figure 4: Two-component hidden Markov model where the second component is a convex combination of the peaked components of the data and the first component is constrained to be uniformly random. This model will tend to have higher likelihood than if the second component generated only strings of "a"s as in Fig. 3.

which the algorithm then effectively erases from the data. The algorithm then fits a new two-component model to the remaining data to discover the second motif. Unfortunately, the CC model shown in Fig. 4 will sometimes have higher likelihood than the desired model in Fig. 3. Since MEME searches for the model with the highest likelihood, CC models can be chosen.

The tendency for the CC model (Fig. 4) to have higher likelihood than the correct model (Fig. 3) increases with the size of the alphabet, m, the width of the motif, W, and the size of the dataset (number of width-W subsequences), n. The difference in the expected value of the log-likelihood on a set of sequences, X, of the two models can be shown (Bailey 1996) to approach infinity as either W, the width of the motif, or m, the size of the alphabet does,

$$\lim_{W,m\to\infty} (E[\log Pr_{ab}(X)] - E[\log Pr_a(X)]) = \infty.$$

The expectation is over samples of size n and it can also be shown (Bailey 1996) that the difference in expectation approaches infinity with increasing sample size whenever λ_2 is closer to 0.5 than λ_1 is,

$$\lim_{n\to\infty} (E[\log Pr_{ab}(X)] - E[\log Pr_a(X)]) = \infty$$

if and only if $|0.5 - \lambda_2| < |0.5 - \lambda_1|$. This means that for large alphabets and/or large motif widths, the convex combination model will have higher likelihood than the desired model. Additionally, for certain values of the ratio of number of motif occurrences to total dataset size, the problem becomes worse as the size of the dataset increases.

Dirichlet mixture priors

A basic understanding of Dirichlet mixture priors is necessary in order to understand the megaprior heuristic. Dirichlet mixture priors encode biological information in the form of "likely" residue-frequency columns. The mean of each component of a Dirichlet mixture prior is a "typical" column of a MEME motif or a match state in an HMM. It is well known that the twenty amino acids can be grouped according to similarity along several dimensions such as the size, polarity and hydrophobicity of their side-chains. As a result, it is not surprising that the columns of residue-frequency matrices can be grouped into a fairly small number of classes. Each of these classes can be described by a Dirichlet distribution and the overall distribution of columns in residue-frequency matrices can be modeled by a mixture of these Dirichlet distributions. The experiments discussed in this paper use a thirty-component Dirichlet mixture prior (our "standard" prior) estimated by Brown *et al.* (1993) from the residue-frequency vectors observed in a large number of trusted multiple alignments.

Dirichlet mixture priors are used by modifying the learning algorithm (EM in the case of MEME) to maximize the posterior probability rather than the likelihood of the training sequences given the model and

the prior distribution of the parameters of the model. The effect of the prior is to increase the probability of models whose residue-frequency columns are close to the mean of some component of the prior. This effect decreases as the size of the training set increases. The effect increases as the variance of the components of the prior decrease.

An R-component Dirichlet mixture density has the form $\rho = q_1\rho_1 + \ldots + q_R\rho_R$, where $q_i > 0$ is a mixing parameter, ρ_i is a Dirichlet probability density function with parameter $\beta^{(i)} = (\beta_a^{(i)}, \ldots, \beta_z^{(i)})$, and $\mathcal{L} = a, \ldots, z$ is the sequence alphabet. For protein sequences, the ith component, ρ_i, is described by a parameter vector $\beta^{(i)}$ of length twenty. The twenty positions in the parameter vector correspond to the twenty letters in the protein alphabet.

All the components of a Dirichlet parameter vector are positive (by the definition of a Dirichlet distribution), so we can normalize it to be a probability vector (a vector whose components are non-negative and sum to one). We do this by dividing the parameter vector, $\beta^{(i)}$, by its magnitude, $b_i = \sum_{x\in\mathcal{L}} \beta_x^{(i)}$. The normalized vector, $\beta^{(i)}/b_i$, is the mean of component ρ_i of the mixture and has the same form as a column in a residue-frequency matrix. Later we shall show that the parameter b_i is inversely proportional to the variance of component ρ_i.

The presence of component ρ_i in the standard prior we use indicates that many residue-frequency vectors with values near the mean of ρ_i are observed in trusted multiple alignments. If the variance of component ρ_i is low, then its mean was the center of a dense cluster of residue-frequency vectors in the data that was used to learn the mixture prior. If its variance is high, it was the center of a more diffuse cluster of observed residue-frequency vectors in the training data. The size of the mixing parameter for component ρ_i, q_i, is indicative of the number of observed residue-frequency vectors near the mean of ρ_i. Large mixing parameters indicate that there were (relatively) many residue-frequency vectors near the mean of that component in the multiple alignments used to learn the standard prior.

The thirty components of the standard prior can be summarized as follows. Twenty of the components have means near residue-frequency vectors corresponding to a single amino acid. This reflects the fact that in many columns in multiple alignments a single amino acid predominates. The ten other components have means corresponding (roughly) to the residue-frequency distributions observed in different protein environments such as alpha helices, beta strands, interior beta strands and interior alpha helices.

Dirichlet mixture priors can be used as follows for learning sequence models. Let $\mathbf{c} = [c_a, \ldots, c_z]^T$ be the vector of observed counts of residues in a particular column of the motif or multiple alignment. The probability of component ρ_i in the Dirichlet mixture having generated the observed counts for this column is calculated using Bayes' rule,

$$Pr(\beta^{(i)}|\mathbf{c}) = \frac{q_i Pr(\mathbf{c}|\beta^{(i)})}{\sum_{j=1}^{R} q_j Pr(\mathbf{c}|\beta^{(j)})}.$$

If we define $c = \sum_{x\in\mathcal{L}} c_x$ and $b_i = \sum_{x\in\mathcal{L}} \beta_x^{(i)}$, then

$$Pr(\mathbf{c}|\beta^{(i)}) = \frac{\Gamma(c+1)\Gamma(b_i)}{\Gamma(c+b_i)} \prod_{x\in\mathcal{L}} \frac{\Gamma(c_x + b_x^{(i)})}{\Gamma(c_x+1)\Gamma(b_x^{(i)})}$$

where $\Gamma(\cdot)$ is the gamma function. We estimate a vector of pseudo-counts as a function of the observed counts as $\mathbf{d}(\mathbf{c}) = [d_a, d_b, \ldots, d_z]^T$ where

$$d_x = \sum_{i=1}^{R} Pr(\beta^{(i)}|\mathbf{c})\beta_x^{(i)},$$

for each $x \in \mathcal{L}$. The mean posterior estimate of the residue probabilities $\mathbf{p}_k$ in column k of the sequence model is then

$$\mathbf{p}^k = \frac{\mathbf{c}_k + \mathbf{d}(\mathbf{c}_k)}{|\mathbf{c}_k + \mathbf{d}(\mathbf{c}_k)|},$$

for $k = 1$ to W. This gives the Bayes estimate of the residue probabilities for column k of the sequence model.

The megaprior heuristic

The megaprior heuristic is based on biological background knowledge about what constitutes a reasonable column in a residue-frequency matrix. Since convex combinations improperly align sequence positions, their observed residue-frequency vectors will tend to be biologically unreasonable. The megaprior heuristic extends the idea of using Dirichlet mixture distributions for modeling the distribution of the columns of protein sequence models (Brown *et al.* 1993) to prevent this from occurring by severely penalizing models with biologically unreasonable columns. This is done by linearly scaling the variance of each component of the Dirichlet mixture prior so that it is sufficiently small to make the effect of the prior dominate even when the training set is large. This turns out to be extremely simple to implement as we show below. All that is needed is to multiply the parameters of each component of the prior by a factor proportional to the size of the training set.

The megaprior heuristic is implemented by multiplying each parameter of each component of the Dirichlet mixture prior by a scale factor, s, which is dependent on the sample size.[2] Consider component ρ_i of the mixture prior. Recall that the magnitude, b_i, of Dirichlet

[2]Sample size is the number of width-W subsequences present in the training set. When W is small compared to the length of the sequences, sample size is approximately equal to the total number of characters in the sequences.

distribution ρ_i with parameters $\beta^{(i)} = (\beta_a^{(i)}, \ldots, \beta_z^{(i)})$ is defined as $b_i = \sum_{x \in \mathcal{L}} \beta_x^{(i)}$. The variance of ρ_i is inversely proportional to its magnitude, b_i, since (Santner and Duffy 1989)

$$Var(\mathbf{c}) = \frac{(\beta^{(i)}/b_i)(\mathbf{I} - (\beta^{(i)}/b_i))}{b_i + 1}.$$

Thus, multiplying the parameter vector $\beta^{(i)}$ of component ρ_i of a Dirichlet mixture prior by scale factor $s > 0$ reduces the the variance of the component by a factor of approximately $1/s$. This scaling does not affect the mean of the component because the mean of a Dirichlet distribution with parameter $s\beta^{(i)}$ is $s\beta^{(i)}/sb_i = \beta^{(i)}/b_i$, the mean of ρ_i.

The actual scale factor used by the megaprior heuristic is

$$s = \frac{kn}{b},$$

where b is the sum of the magnitudes of the components of the prior, $b = \sum_{i=1}^{R} b_i$, and n is the sample size. Thus, the heuristic multiplies parameter vector $\beta^{(i)}$ of the ith Dirichlet component of the prior by kn/b, for $1 \leq i \leq R$. When k is large, this causes the posterior estimates of the parameters of columns of a model always to be extremely close to the mean of one of the components of the mixture prior. Experiments (results not shown) indicated that a good value for k is $k = 10$, although values between $k = 1$ and $k = 20$ do not change the results appreciably. The results reported in the next section use $k = 10$.

Several sequence modeling algorithms–including MEME and the HMM sequence alignment algorithms mentioned in the introduction–use mixture of Dirichlet priors because this has been shown to improve the quality of the patterns they discover (Bailey and Elkan 1995b; Eddy 1995; Baldi *et al.* 1994). Since these algorithms already use Dirichlet mixture priors, most of the algorithmic machinery needed for implementing the megaprior heuristic is already in place. In the case of these algorithms, implementing the megaprior heuristic requires no algorithmic modifications (beyond scaling the components of the prior) and the mathematics remain the same.

Results

We studied the effect of using the megaprior heuristic with the MEME (Bailey and Elkan 1995a) motif discovery algorithm. MEME takes as input a group of training sequences and outputs a series of motif models each of which describes a single motif present in the sequences. The user can specify one of three different types of motif models for MEME to use, each of which reflects different background knowledge about the arrangement of the motifs within the sequences. The OOPS model (One Occurrence Per Sequence) forces MEME to construct each motif model by choosing a single motif occurrence from each sequence. The ZOOPS model (Zero or One Occurrence Per Sequence) permits MEME to choose at most one position in each sequence when constructing a motif model. This allows for some motifs not being in all sequences in the training set and provides robustness against noise (e.g., sequences that do not belong in the training set.) The least constrained model is called the TCM model (Two Component Mixture) which allows MEME to choose as many (or few) motif occurrences in each sequence as necessary to maximize the likelihood of the motif model given the training sequences.

quantity	*mean*	(*sd*)
sequences per dataset	34	(36)
dataset size	12945	(11922)
sequence length	386	(306)
shortest sequence	256	(180)
longest sequence	841	(585)
pattern width	12.45	(5.42)

Table 1: **Overview of the 75 Prosite datasets.** Each dataset contains all protein sequences (taken from SWISS-PROT version 31) annotated in the Prosite database as true positives or false negatives for a single Prosite family. Dataset size and sequence length count the total number of amino acids in the protein sequences. The Prosite families used in the experiments are: PS00030, PS00037, PS00038, PS00043, PS00060, PS00061, PS00070, PS00075, PS00077, PS00079, PS00092, PS00095, PS00099, PS00118, PS00120, PS00133, PS00141, PS00144, PS00158, PS00180, PS00185, PS00188, PS00190, PS00194, PS00198, PS00209, PS00211, PS00215, PS00217, PS00225, PS00281, PS00283, PS00287, PS00301, PS00338, PS00339, PS00340, PS00343, PS00372, PS00399, PS00401, PS00402, PS00422, PS00435, PS00436, PS00490, PS00548, PS00589, PS00599, PS00606, PS00624, PS00626, PS00637, PS00639, PS00640, PS00643, PS00656, PS00659, PS00675, PS00676, PS00678, PS00687, PS00697, PS00700, PS00716, PS00741, PS00760, PS00761, PS00831, PS00850, PS00867, PS00869, PS00881, PS00904 and PS00933.

To study the improvement in the quality of motif models found by MEME using the megaprior heuristic, we use training sets containing groups of protein sequences with known motifs. We measure how well the motif models found by MEME match the known motif occurrences in the training set by using each MEME-determined motif model as a classifier on the training set and calculating the receiver operating characteristic (*ROC*) (Swets 1988), *recall* and *precision* of the model with respect to the known motifs. To do this, we tally the number of the number of true positive (tp), false positive (fp), true negative (tn) and false negative (fn) classifications. We define $recall = tp/(tp + fn)$, which gives the fraction of the known motif occurrences that are found by the model. Likewise, $precision = tp/(tp+fp)$, gives the fraction of the predicted motif occurrences that are correct. Low values of *precision* usually correspond to models that are convex combinations; improved *precision* is an in-

	ROC		*recall*		*precision*	
	M	*S*	*M*	*S*	*M*	*S*
means	0.992	0.986	0.79	0.81	0.73	0.23
$p = 0.05$	+		−		+	
$p = 0.01$	+		−		+	

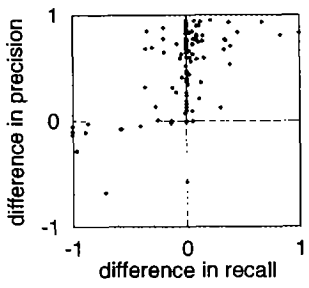

Figure 5: **Comparison of the megaprior (M) and standard prior (S) in finding protein motifs.** Data is for the best models found by MEME using the TCM model for 135 known motifs in 75 datasets. The table shows the average value and significance of the differences in *ROC*, *recall* and *precision* for models found using the two heuristics. Significance is evaluated at the 0.05 and 0.01 levels using paired t-tests. Each point in the scatter plot shows the difference (M-S) in *recall* on the x-axis and *precision* on the y-axis for models found using the two different priors.

dicator of a reduction in the CC problem.

We conducted tests using the 75 datasets (subsequently referred to as the "Prosite datasets") described in Table 1. Each dataset consists of all the sequences annotated as true positives or false negatives in a single Prosite family. Many Prosite families overlap one another, and the 75 datasets comprise a total of 135 different known motifs (where "comprise" is defined as at least five occurrences of the motif present in the dataset). MEME was run for five passes in order to find five models and the results reported are for the model with the highest *ROC* relative to a known motif in the dataset. Predicted motif occurrences are allowed to be shifted relative to the known occurrences as long as all are shifted by the same amount.

Using the megaprior heuristic greatly improves the quality of TCM models found by MEME. When the megaprior heuristic is not used, the average *precision* of learned TCM models is extremely low (0.23, Fig. 5). Using the megaprior heuristic, the average *precision* increases to 0.73. This improvement is significant at the $P = 0.01$ level in a paired t-test. The average *recall* decreases slightly from 0.81 to 0.79, but this change is not significant at even the $P = 0.05$ level.

The overall performance of the model as measured by the *ROC* statistic also improves significantly ($P = 0.01$). The scatter plot in Fig. 5 shows that most models found using the megaprior heuristic are uniformly superior (better *recall* and better *precision*) to those found with the standard prior, and virtually all have better *precision*. Each point in the plot represents one of the 135 known motifs. The highly populated upper right quadrant corresponds to the cases where the megaprior model is uniformly superior. Uniformly superior models were found using the standard prior for only 12 of the known motifs. Almost all of the points in the scatter plot lie in the upper two quadrants. This shows that the models found using the heuristic are almost always more specific (have fewer false positives) models of the known motif than when the heuristic is not used.

The improvement in the TCM models found by MEME using the megaprior heuristic is due to the elimination of convex combination models. Of the 75 training sets we tested, 45 contain five or more sequences from a second known family. More often than not, when the megaprior heuristic is not used, the TCM model found by MEME with these training sets is a convex combination of the two known motifs. Specifically, in 25 out of the 45 datasets with more than one known motif, the best model found for the primary motif had non-zero *recall* for the other known motif and therefore is a convex combination of the two known motifs. In these 25 cases, the average *recall* on the both known motifs of the CC model is 0.88, showing that these convex combination motif models indeed combine virtually all of the occurrences of both motifs. In contrast, when the megaprior heuristic is used, only one convex combination TCM model of two known motifs is found by MEME in the Prosite datasets. This shows that the megaprior heuristic essentially eliminates the convex combination problem with TCM models.

The one training set in which MEME finds a convex combination model of two known motifs even using the megaprior heuristic is instructive. The two known motifs have Prosite signatures which can be overlaid in such a way that the shorter motif essentially fits inside the longer. (Refer to the first two lines of Fig. 6.) The consensus sequence (most frequent letter in each column) for the model found by MEME in the dataset containing these two motifs is shown in the third line of Fig. 6. Where one motif is specific about a choice of amino acids, the other permits it. MEME is fooled into creating a convex combination model which describes both known motifs well because the CC model has no biologically unreasonable columns. Such situations where the megaprior heuristic is inadequate to prevent MEME from finding convex combination models appear to be rare as shown by the fact that 24 of 25 convex combinations are avoided and by the dramatic improvement in the *precision* of the TCM models using the heuristic (Fig. 5).

We have also studied a modification to the megaprior heuristic intended to make the final motif model more closely resemble the observed residue frequencies. This

	Prosite signature or MEME consensus sequence
PS00079	G-x-[FYW]-x-[LIVMFYW]-x-[CST]-x-x-x-x-x-x-x-x- G -[LM]-x-x-x-[LIVMFYW]
PS00080	H- C -H-x-x-x-H-x-x-x-[AG]-[LM]
MEME *model*	P-G-x- W -L- L -H- C -H-I-A-x-H-L-x-A- G - M

Figure 6: **A convex combination model not eliminated by the megaprior heuristic.**

model	*ROC*		*recall*		*precision*		*relative width*	
ZOOPS_DMIX	0.994	(0.025)	0.838	(0.325)	0.780	(0.301)	1.221	(0.673)
ZOOPS_MEGA	0.992	(0.032)	0.778	(0.354)	0.785	(0.338)	1.061	(0.602)
ZOOPS_MEGA'	0.992	(0.030)	0.781	(0.356)	0.785	(0.338)	1.061	(0.605)
TCM_DMIX	0.986	(0.027)	0.811	(0.301)	0.228	(0.233)	0.826	(0.418)
TCM_MEGA	0.992	(0.028)	0.789	(0.353)	0.733	(0.358)	0.912	(0.505)
TCM_MEGA'	0.992	(0.027)	0.801	(0.344)	0.714	(0.351)	0.912	(0.505)

Table 2: **Average (standard deviation) performance of best motif models found by** MEME **in the 75 Prosite datasets.** All 135 known motifs found in the datasets are considered. Data is for TCM (two-component mixture) and ZOOPS (zero-or-one-occurrence-per-sequence) models using the standard prior (DMIX), the megaprior heuristic (MEGA), or the modified megaprior heuristic (MEGA').

is done by replacing the megaprior with the standard prior for the last iteration of EM. Because the standard prior is quite weak, this causes the columns of the final residue-frequency matrix to approach the observed residue frequencies of the motif in the training set.

Table 2 summarizes the results of using the standard prior (DMIX), megaprior heuristic (MEGA) and modified megaprior heuristic (MEGA') with TCM and ZOOPS models on the 75 Prosite datasets. The modified megaprior heuristic improves the *recall* of TCM models found at the cost of degrading their *precision*. The *recall* improvement is significant at the $P = 0.05$ level and the degradation in *precision* at the $P = 0.01$ level. There is no significant change in the *ROC*. Whether to use the modified megaprior heuristic or the unmodified heuristic with TCM models thus depends on the relative importance placed on *recall* versus *precision*. Since *precision* is generally more important, MEME now uses the unmodified heuristic as the default with TCM models.

Both the modified and unmodified megaprior heuristics lower the *ROC* and *recall* of ZOOPS models on the 75 Prosite training sets while raising their *precision* slightly (Table 2). This should not be interpreted as proof that heuristics are of no use with ZOOPS models. The datasets heavily favor the standard prior because most of the known motifs present in each dataset are present in every sequence. In such situations, MEME is not likely to find a ZOOPS model that is a convex combination. A ZOOPS model constrains the algorithm to pick at most one occurrence of the motif per sequence. When every sequence contains a valid occurrence of the known motif, choosing the valid occurrence will tend to maximize the likelihood of the model. Only when a sizable fraction of the sequences do not contain a motif occurrence would we expect a ZOOPS convex combination model to have higher likelihood than the correct ZOOPS model.

datasets					
ZOOPS I		ZOOPS II		ZOOPS II	
PS00188	15	PS00606	17	PS00659	40
PS00867	20	PS00012	40	PS00448	11
PS00866	20				
totals	27		48		45

Table 3: **Datasets for testing the usefulness of the megaprior heuristics with** ZOOPS **models.** Each dataset consists of all the sequences of two or three Prosite families where many of the sequences contain both (or all three) motifs. Each column shows the names and numbers of sequences in the Prosite families in a dataset. The total number of (unique) sequences in each dataset is shown at the bottom of its column.

We expect situations where many of the sequences in the training set do not contain occurrences of all motifs to be common. This will happen, for example, when some of the sequences are descended from a common ancestor that experienced a deletion event that removed the occurrences of some motifs. Sequences unrelated to the majority of the training set sequences might also be unintentionally included in the training set and should be ignored by the motif discovery algorithm.

To determine if either the megaprior or modified megaprior heuristic improves ZOOPS models found by MEME in such situations, we created three new datasets (subsequently referred to as the "ZOOPS datasets") of naturally overlapping Prosite families. Each dataset (see Table 3) consists of all the sequences in two or three Prosite families where several of the sequences contain the known motif for both (or all three) families. A ZOOPS model is appropriate for finding

model	*ROC*		*recall*		*precision*	
ZOOPS_DMIX	1.000	(0.00)	0.93	(0.08)	0.77	(0.06)
ZOOPS_MEGA	1.000	(0.00)	0.90	(0.06)	0.95	(0.03)
ZOOPS_MEGA'	1.000	(0.00)	0.93	(0.07)	0.97	(0.02)

Table 4: **Average (standard deviation) performance of best motif models found by MEME in the three ZOOPS datasets.** Results are for the two or three known motifs in each dataset. Data is for ZOOPS (zero-or-one-occurrence-per-sequence) model using the standard prior (DMIX), the megaprior heuristic (MEGA), or the modified megaprior heuristic (MEGA').

motifs in such datasets because each sequence contains zero or one occurrence of each motif. Since no motif is present in all of the sequences, convex combinations should be possible.

The performance of the ZOOPS motif models found by MEME in the three ZOOPS datasets is shown in Table 4. The low *precision* (0.77) using the standard prior indicates that convex combination models are being found. Using the megaprior heuristic dramatically increases the *precision* (to 0.95) at the cost of a slight decrease in *recall*. As hoped, the modified megaprior heuristic improves the motifs further, increasing the *precision* to 0.97. The *precision* of ZOOPS models using the modified megaprior heuristic is thus higher than with the standard prior or unmodified heuristic. The *recall* is the same as with the standard prior and higher than using the unmodified heuristic. The large improvement in ZOOPS models seen here (Table 4) using the modified megaprior heuristic coupled with the very moderate reduction in *recall* relative to the standard prior seen in the previous test (Table 2) leads us to conclude that the modified megaprior heuristic is clearly advantageous with ZOOPS models. MEME now uses the modified megaprior heuristic by default with ZOOPS models.

Discussion

The megaprior heuristic–using a Dirichlet mixture prior with variance inversely proportional to the size of the training set–greatly improves the quality of protein motifs found by the MEME algorithm in its most powerful mode in which no assumptions are made about the number or arrangement of motif occurrences in the training set. The modified heuristic–relaxing the heuristic for the last iteration of EM–improves the quality of MEME motifs when each sequence in the training set is assumed to have exactly zero or one occurrence of each motif and some of the sequences do lack motif occurrences. This later case is probably the most commonly occurring situation in practice since most protein families contain a number of motifs but not all members contain all motifs. Furthermore, training sets are bound to occasionally contain erroneous (non-family member) sequences that do not contain any motif occurrences in common with the other sequences in the training set.

The megaprior heuristic works by removing most models that are convex combinations of motifs from the search space of the learning algorithm. It effectively reduces the search space to models where each column of a model is the mean of one of the components of the Dirichlet mixture prior. To see this, consider searching for motif models of width W. Such models have $20W$ real-valued parameters (W length-20 residue-frequency vectors), so the search space is uncountable. Using a Dirichlet mixture prior with low variance reduces the size of the search space by making model columns that are not close to one of components of the prior have low likelihood. In the limit, if the variance of each component of the prior were zero, the only models with non-zero likelihood would be those where each column is exactly equal to the mean of one of the components of the prior. Thus, the search space would be reduced to a countable number of possible models. Scaling the strength of the prior with the size of the dataset insures that this search space reduction occurs even for large datasets.

The search space of motif models using the megaprior heuristic, though countable, is still extremely large–30^W in the case of a 30-component prior. It is therefore still advantageous to use a continuous algorithm such as MEME to search it rather than a discrete algorithm. Incorporating the megaprior heuristic into MEME was extremely straightforward and allows a single algorithm to be used for searching for protein and DNA motifs in a wide variety of situations. In particular, using the new heuristics, protein motif discovery by MEME is now extremely robust in the three situations most likely to occur:

- each sequence in the training set is known to contain a motif instance;
- most sequences contain a motif instance;
- nothing is known about the motif instance arrangements.

The megaprior heuristic is currently only applicable to protein datasets. It would be possible to develop a Dirichlet mixture prior for DNA as well, but experience and the analysis in the section on convex combinations shows that this is probably not necessary. The severity of the CC problem is much greater for protein datasets than for DNA, because the CC problem increases with the size of the alphabet. For this reason, convex combinations are less of a problem with DNA sequence

models.

The success of the megaprior heuristic in this application (sequence model discovery) depends on the fact that most of the columns of correct protein sequence models are close to the mean of some component of the thirty-component Dirichlet mixture prior we use. Were this not the case, it would be impossible to discover good models for motifs that contain many columns of observed frequencies far from any component of the prior. Using the modified heuristic, unusual motif columns that do not match any of the components of the prior are recovered when the standard prior is applied in the last step of EM. We have studied only one Dirichlet mixture prior. It is possible that better priors exist for use with the megaprior heuristic–this is a topic for further research.

The megaprior heuristic should improve the sequence patterns discovered by other algorithms prone to convex combinations. Applying the megaprior heuristic to HMM multiple sequence alignment algorithms is trivial for algorithms that already use Dirichlet mixture priors (e.g., that of Eddy (1995)), since it is only necessary to multiply each component by kn/b. Using the heuristic with other types of HMM algorithms will first involve modifying them to utilize a Dirichlet mixture prior. There is every reason to believe that, based on the results with MEME, this will improve the quality of alignments. Utilizing the heuristic with the Gibbs sampler may be problematic since the input to the algorithm includes the number of motifs and the number of occurrences of each motif. Avoiding the convex combination problem with the sampler requires initially getting these numbers correct.

A website for MEME exists at URL

`http://www.sdsc.edu/MEME`

through which groups of sequences can be submitted and results are returned by email. The source code for MEME is also available through the website.

Acknowledgements:

This work was supported by the National Biomedical Computation Resource, an NIH/NCRR funded research resource (P41 RR-08605), and the NSF through cooperative agreement ASC-02827.

References

Timothy L. Bailey and Charles Elkan. Unsupervised learning of multiple motifs in biopolymers using EM. *Machine Learning*, 21(1-2):51–80, October 1995.

Timothy L. Bailey and Charles Elkan. The value of prior knowledge in discovering motifs with MEME. In *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 21–29. AAAI Press, 1995.

Timothy L. Bailey. Separating mixtures using megapriors. Technical Report CS96-471, `http://www.sdsc.edu/~/tbailey/papers.html`, Department of Computer Science, University of California, San Diego, January 1996.

Amos Bairoch. The PROSITE database, its status in 1995. *Nucleic Acids Research*, 24(1):189–196, 1995.

P. Baldi, Y. Chauvin, T. Hunkapiller, and M. A. Mcclure. Hidden Markov models of biological primary sequence information. *Proc. Natl. Acad. Sci. USA*, 91(3):1059–1063, 1994.

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.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

Sean R. Eddy. Multiple alignment using hidden Markov models. In *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 114–120. AAAI Press, 1995.

Michael Gribskov, Roland Lüthy, and David Eisenberg. Profile analysis. *Methods in Enzymology*, 183:146–159, 1990.

A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, 1994.

Charles E. Lawrence, Stephen F. Altschul, Mark S. Boguski, Jun S. Liu, Andrew F. Neuwald, and John C. Wootton. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, 1993.

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

John A. Swets. Measuring the accuracy of diagnostic systems. *Science*, 270:1285–1293, June 1988.