2009-04-16

XPRIME: A Method Incorporating Expert Prior Information into Motif Exploration

Rachel Lynn Poulsen
Brigham Young University - Provo

XPRIME: A METHOD INCORPORATING EXPERT PRIOR INFORMATION INTO MOTIF EXPLORATION

by

Rachel L. Poulsen

A project submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Statistics

Brigham Young University

August 2009

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a project submitted by

Rachel L. Poulsen

This project has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	W. Evan Johnson, Chair
Date	Natalie J. Blades, Member
Date	G. Bruce Schaalje, Member

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the project of Rachel L. Poulsen in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

W. Evan Johnson
Chair, Graduate Committee

Accepted for the Department

Scott D. Grimshaw
Graduate Coordinator

Accepted for the College

Thomas W. Sederberg
Associate Dean, College of Physical and Mathematical Sciences

ABSTRACT

XPRIME: A METHOD INCORPORATING EXPERT PRIOR INFORMATION INTO MOTIF EXPLORATION

Rachel L. Poulsen

Department of Statistics

Master of Science

One of the primary goals of active research in molecular biology is to better understand the process of transcription regulation. An important objective in understanding transcription is identifying transcription factors that directly regulate target genes. Identifying these transcription factors is a key step toward eliminating genetic diseases or disease susceptibilities that are encoded inside deoxyribonucleic acid (DNA). There is much uncertainty and variation associated with transcription factor binding sites, requiring these sites to be represented stochastically. Although typically each trancription factor prefers to bind to a specific DNA word, it can bind to different variations of that DNA word. In order to model these uncertainties, we use a Bayesian approach that allows the binding probabilities associated with the motif to vary. This project presents a new method for motif searching that uses expert prior information to scan DNA sequences for multiple known motif binding sites as well as new motifs. The method uses a mixture model to model the motifs of interest where each motif is represented by a Multinomial distribution, and Dirichlet prior distributions

are placed on each motif of interest. Expert prior information is given to search for known motifs and diffuse priors are used to search for new motifs. The posterior distribution of each motif is then sampled using Markov Chain Monte Carol (MCMC) techniques and Gibbs sampling.

ACKNOWLEDGEMENTS

I would first like to thank my Heavenly Father for the love and strength he has given me during the past two years. I would also like to thank Dr. W. Evan Johnson for the many hours of effort he has put into helping me complete this project as well as the many hours of counsel he has given me. I would like to thank the Huntsman Cancer Institute for the data they provided, the Johnson Lab for their input, and the Statistics Department at Brigham Young University for their support.

Most of all, I would like to thank my husband. He has given me the confidence to do anything in this world including the encouragement to pursue this degree. His patience, love, and listening ear have provided me with so much strength through this journey. I love you.

CONTENTS

TABLES

Table

FIGURES

Figure

1. INTRODUCTION

All living organisms are unique. The deoxyribonucleic acid (DNA) of each organism contains the set of genetic instructions that uniquely define it. DNA is enclosed in the nucleus of every living cell and will not leave the nucleus, which is separate from the rest of the cell. In order for DNA to pass genetic instructions outside the nucleus and communicate with the rest of the organism, ribonucleic acid (RNA) is created. RNA has the ability to travel through the nucleus and carry genetic instructions from the DNA to the rest of the cell. RNA is created and receives instructions from DNA through a process called *transcription*. Transcription is the process of the DNA sequence information being translated into RNA sequence information. DNA and RNA sequences use a complementary language that allows the genetic information to simply be transcribed, or copied, from one sequence to the other.

One of the primary goals of current research in molecular biology is to better understand the process of transcription regulation. A small subset of functioning proteins returns to the nucleus to assist in the transcription process. These proteins are also known as transcription factors (TFs), as they aid in regulating transcription. TFs regulate transcription by binding to specific subsequences of DNA and controlling the transfer of genetic information from DNA to RNA by either promoting or blocking the transfer of genetic infromation. Studying these proteins can help researchers understand transcription regulation. Ultimately, understanding TFs can help researchers to understand the origin of different genetic diseases. These proteins have the ability to block or promote the transfer of genetic instructions from the DNA that may contain diseases or disease susceptibilities such as colon cancer or heart disease. However, each protein performs multiple functions, so removing the protein that

promotes arthritis may also be removing the protein that blocks abdominal cancer. Thus, it is of great importance to understand the process of transcription regulation and each of the proteins that aid in the process.

DNA is made up of a long sequence of four nucleotides that are represented by the letters A, C, G, and T. These nucleotides are also referred to as bases. TFs will most often bind to DNA close to the transcription start site. This binding action then promotes or blocks the transcription process from occurring. It can require the right combination of TFs binding next to the transcription start site to actively transcribe the gene, or specific set of genetic instructions. Each individual TF usually prefers to bind to a small DNA word, typically five to twenty base pairs long, called a *binding motif*. A binding motif represents a recurring word pattern of a short sequence of DNA that identifies an active TF binding site in the DNA. There can be multiple variations of each binding motif. One of the most important objectives in understanding transcription is to identify target genes that are directly regulated by any given TF. Identifying a binding motif is challenging because its presence does not necessarily imply that the TF is actively binding to the DNA. Also, the binding motif and its variations for the TF of interest may not be well characterized. Because these DNA words are not fixed and the protein can bind to slightly different variations of the motif, they are stochastic, and their occurrences can be represented using a probability mass function.

Researchers will usually try to identify the highest affinity binding motif for a TF. One way of representing a binding motif is by a *motif matrix*, better known as a *position specific weight matrix* (PWM) (Hertz et al. 1990). A PWM is defined as a 4 x n matrix, where n is the length of the motif of interest, and 4 represents the four nucleotides A, C, G, and T. The p_{ij} represents the elements of a PWM where p_{ij} is the probability that the jth (column) position of the motif binds to the ith (row)

nucleotide. For example, the PWM for the TF known as ETS1 can be given by

$$\begin{array}{cc} \text{Position:} & \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{array}{c} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} & \begin{pmatrix} 0.067 & 0.333 & 0.0 & 0.0 & 1.0 & 0.533 & 0.267 & 0.067 \\ 0.933 & 0.600 & 0.0 & 0.0 & 0.0 & 0.133 & 0.067 & 0.400 \\ 0.000 & 0.000 & 1.0 & 1.0 & 0.0 & 0.000 & 0.667 & 0.000 \\ 0.000 & 0.067 & 0.0 & 0.0 & 0.0 & 0.333 & 0.000 & 0.533 \end{pmatrix} \end{array}.$$

Notice that all the columns should sum to one. (In the above example, some of the columns may not add to exactly one due to rounding.) It can be assumed that the positions or columns are independent of one another. The implications of this assumption are very useful and will be discussed in detail later.

In this project we are interested in scanning the DNA for multiple known motif binding sites as well as new motifs simultaneously. The process of identifying new motifs is better known as *de novo* motif searching. When searching for new motifs, the many repetitive elements in the DNA add noise to the motif search, making it difficult to identify which repetitive elements are true binding motifs and which are simply noise. Also, motifs themselves are highly variable, making the actual binding site unclear. A known TFs binding motif usually has many variations that are not well characterized. It is important that an approach other than simply checking for the occurrence of the known motif be used in order to account for these variations. These variations can be represented by using a PWM to represent the motif. Our proposed model allows users to search for as many known motifs and *de novo* motifs as they would like.

In order to account for the large amount of uncertainty and variation associated with binding motifs, we use a Bayesian approach. Using a Bayesian approach is advantageous in that it does not assume the p_{ij}s in a PWM are fixed. A Bayesian analysis allows for the uncertainty and variation associated with the binding motif to be measured by giving probability to a parameter that is usually fixed, like the

p_{ij}s in a PWM. In the process of calculating the posterior density, this uncertainty is integrated out, leaving a parameter with a measured variation. The posterior distribution measures the variation among the probabilities in the PWM, allowing each p_{ij} to have a probability density. A Bayesian approach also allows us to use expert prior information to influence the final model. Though this approach is biased, allowing solely the data to have weight on the final model is subject to random error. More specifically, we only have one sample of the data; we do not have every *possible* sample, which is assumed in most frequentist statistical analysis. It is very possible that our single sample is not a good representation of our population. This is especially important in DNA sequence samples, as they are so small in comparison to the large amount of DNA contained in the human genome, meaning the data alone cannot tell us everything. Allowing expert prior information from literature and experiments to influence the results gives more accurate results and a richer final model. Since the computation for the posterior distribution is complicated, our method samples from the posterior distribution using Gibbs sampling. We call our method XPRIME: Incorporating E***X***pert ***PR***ior ***I***nformation into ***M***otif ***E***xploration.

There are several publicly and commercially available literature-based databases of known TF binding motifs. TRANSFAC (Matys et al. 2003) is the most well-known commercially available database. TRANSFAC contains over 10 times the amount of information as the leading publicly available database, JASPAR (Sandelin et al. 2004). The information contained in TRANSFAC comes from the available literature as well as *in vitro* experiments. When an experiment is performed *in vitro*, it is performed in a controlled environment outside the living organism. This means it is performed in an environment in which it does not regularly function. It is possible that a TF behaves one way *in vitro* and a different way in its own environment, *in vivo*. Because the information from TRANSFAC is pulled from multiple *in vitro* experiments and literature suggestions, it cannot be considered truth. Thus, we believe the infor-

mation from TRANSFAC would better serve as the expert prior information in our model.

2. LITERATURE REVIEW

Over the past decade, many computational methods have been developed to search for and identify transcription factor binding sites (TFBS) within a list of DNA sequences. Most methods include some sort of scoring function to measure the likelihood of the existence of a TFBS. There are methods available for *de novo* motif searching as well as known motif searching. Searching for new motifs is a difficult task because binding motifs are not well characterized and are highly variable.

De novo motif searching is typically done using regular word enumeration or PWM updating. Regular word enumeration can be performed by simply scanning a DNA sequence for the DNA word motif and computing a ratio of actual count to expected count This is similar to a χ^2 goodness-of-fit test. However, other more sophisticated methods have been created using regular word enumeration.

One method available using regular word enumeration involves creating a dictionary-based sequence model (Bussemaker et al. 2000). DNA sequences can be considered another language consisting of an alphabet of four letters. The model takes any sequence of symbols, segments it into words, and probabilistically creates a dictionary of these words. In a DNA sequence, these words are simply a sequence of nucleotides of different lengths, each with an associated probability, p_α. The p_αs are normalized so that they sum to one. *De novo* motif searching involves constructing a dictionary of words from a given sequence, called S. The model begins by observing the frequency of the individual letters and then identifying the overrepresented pairs. The overrepresented pairs are then added to the dictionary. The words' associated probabilities, p_α, are found using a maximum likelihood procedure. Finally, DNA words are added to the dictionary by using the following Z-score to test for overrepresentation. In the equation below, $\boldsymbol{N_{\alpha\beta}}$=$N_{av}p_\alpha p_\beta$, where N_{av} is the average number of words in S equal

to $\frac{L}{l}$, such that L represents the total length of S, and l=$\sum_\alpha lp_\alpha$, or the average word length.

$$Z_{\alpha\beta} = \frac{\boldsymbol{N_{\alpha\beta}} - N_{av}p_\alpha p_\beta}{\sqrt{N_{av}p_\alpha p_\beta}}. \tag{2.1}$$

At each step, the p_αs are determined and the Z-scores are calculated. Pairs with a Z-score above a specified threshold are considered significant and added to the dictionary as new words. The algorithm is repeated until the Z-scores for all remaining word pairs are below the specified threshold.

Another regular word enumeration method was proposed by Sinha and Tompa (2006). They believed that the occurrences of a motif were dependent on each other and could not be assumed independent. Thus, they compared overrepresented DNA words to their probability of being randomly generated from a 3rd-order Markov chain. By definition of a Markov chain, the current state is conditional only on the previous state. A 3rd-order Markov chain allows the current state to be conditional on the previous three states, thereby allowing the occurrences of the positions in the sequence to be dependent on each other. A score is given to each possible motif. Allowing N_s to be the number of times motif s is found in sequence U, and X_s to be the number of times s is found in the random DNA sequence generated from a 3rd-order Markov chain,

$$z_s = \frac{N_s - E(X_s)}{\sigma(X_s)}. \tag{2.2}$$

The Z-score is calculated for every motif, and new motifs are then defined using a significance criterion.

One of the most well-known *de novo* motif searching methods using PWM updating is known as Multiple EM for Motif Elicitation (MEME) (Bailey and Elkan 1994). The latest version of MEME takes a list of DNA sequences and outputs a series of probabilistic sequence models corresponding to the motif of interest (Bailey and Elkan 1995). MEME uses a two-component mixture model, expectation maximization, and a Bayes-optimal classifier to search for TFBSs. The two-component mixture model consists

of one model for the motif and one model for background noise. Motifs are modeled using PWMs and the background is modeled using a single discrete random variable. Once the background motif parameters are specified, they remain fixed throughout the algorithm. MEME updates the parameters of the motif using the expectation-maximization (EM) algorithm to maximize the expectation of the joint likelihood of the model. The log joint likelihood is given below. W is the width of the motif of interest, L is the length of the given sequence, and m=L-W+1 is the number of possible starting positions for a motif occurrence in each sequence. We will refer to each of the possible subsets of a DNA sequence that can be a motif of length W as an m-mer. Thus, m can also be considered the possible starting positions for each m-mer. $Z_{i,j}$ is an indicator variable, indicating if a motif occurrence starts at position j in sequence X_i. θ_0 represents the PWM of the motif of interest, θ_1 represents the background PWM, and $\lambda = Pr(Z_{ij} = 1)$.

$$logPr(X, Z|\theta, \lambda) = \sum_{i=1}^{n}\sum_{j=1}^{m}((1 - Z_{i,j})logPr(X_{i,j}|\theta_0) \tag{2.3}$$

$$+ Z_{i,j}logPr(X_{i,j}|\theta_1) \tag{2.4}$$

$$(1 - Z_{i,j})log(1 - \lambda) + (Z_{i,j})log\lambda). \tag{2.5}$$

The EM algorithm is also useful to impute values for the missing information, which is whether or not a given nucleotide belongs to the motif or to the background.

MEME can also identify multiple different motifs iteratively by fitting the mixture model to the data, probabilistically erasing the occurrences of the motif found in the prior iteration, and then repeating the process. Probabilistically erasing the occurrences of the motif found in the prior iteration of MEME allows the algorithm to find a different motif each time. The first iteration will then identify the most probable motif occurrence followed by the next most probable motif occurrence and so forth. MEME can also incorporate prior information about the letter frequency parameters in θ_0. MEME uses Dirichlet mixture priors to calculate the mean posterior

estimates in the M-step. Although MEME is widely used, the execution of MEME is very inefficient, taking weeks to run, especially with a large number of sequences.

Another method using PWM updating involves a Gibbs sampling approach without prior information to idenfity new motifs (Lawerence et al. 1993). This method is also known as the Gibbs Motif Sampler or GMS. GMS tries to locate relatively short patterns that are shared by multiple sequences. Lawrence et al. (1993) believe that the Gibbs sampler has a more robust optimization procedure because it allows for the integration of information from multiple patterns.

The GMS algorithm is employed as follows. Given a set of N sequences $S_1, \cdots, S_N$, the algorithm searches for mutually similar segments of a specified width, W, within each sequence. The algorithm models the segment pattern description of each sequence by giving a frequency to each position, represented by $q_{i,j}$. The algorithm gives background frequencies to the remaining positions, represented by p_j. The $q_{i,j}$ can also be representative of the probabilities contained in a PWM. The objective is to identify the most probable or "best" common pattern. This pattern is updated by locating the alignment that maximizes the ratio of the corresponding pattern probability to the background probability such that F in the following equation is maximized.

$$F = \sum_{i}^{W} \sum_{j}^{L} c_{i,j} log(\frac{q_{i,j}}{p_j}), \tag{2.6}$$

where $c_{i,j}$ represent the counts of nucleotide j in position i. The algorithm is initialized by choosing random starting positions within the various sequences. This ratio is then succesively sampled from the given sequences using a Gibbs sampler.

The results of GMS have been used to serve as a starting point for "Aligns Nucleic Acid Conserved Elements" (AlignACE) algorithm (Roth et al. 1998). Given a sequence of DNA, the algorithm searches for similar aligned segments of short candidate motifs. GMS is used as a starting point as Roth et al. (1998) believe it to have the most flexible and exhaustive search methodology. AlignACE then uses both

an alignment score and an occurrence score criteria to classify TFBSs. The alignment score measures the "goodness" of the sequence aligment and is given by Liu et al. (1995, equation 10). This equation integrates out all but the unobserved parameters, giving a more concise formula and a more computationally efficient algorithm for the Gibbs sampler. The occurrence score measures a ratio similar to the residue frequencies in the GMS algorithm.

BioProspector is another *de novo* method that uses PWM updating. BioProspector also uses a Gibbs sampling strategy, but evaluates the background frequencies using a 3rd-order Markov chain, allowing for the positions in the sequence to be dependent on each other (Liu et al. 2001). More specifically, the probability of generating the segment "ATGTA" from a third-order Markov background model is calculated as follows:

$$\begin{aligned} P^3_{ATCTA} =& p(A)P(T|previous\ base\ is\ A)p(G|previous\ 2\ bases\ are\ AT) \\ & p(T|previous\ 3\ bases\ are\ ATG)p(A|previous\ 3\ bases\ are\ TGT). \end{aligned}$$

Bioprospector uses the following equation to score each possible location for the motif in a given DNA sequence. The equation is estimated using a Monte Carlo method, where q_{ij}, p_j, and w are defined as in the GMS algorithm and N represents the total number of segments in which the motif can exist, similar to m in the GMS algorithm.

$$MotifScore = N * exp[\sum_{(all\ positions\ i)} \sum_{(all\ nucleotides\ j)} log\frac{q_{ij}}{p_j}]/w. \tag{2.7}$$

At each run of BioProspector, a Gibbs sampling strategy is used to initialize a PWM by first taking a random sample of the input sequence. BioProspector then samples new possible alignments using two score thresholds, a low threshold and a high threshold. These thresholds are helpful when a sequence does not contain any copies

of the motif so that a sequence with a zero probability of containing a motif cannot be considered statistically significant. All the nonoverlapping segments of the sequence with a score higher than the high threshold are automatically added to the list of new motifs. All segments with scores between the two thresholds are put into a group of segments where only one is probabilistically chosen, and all segments with a score below the low threshold are removed. Sampling only among segments with scores between the two thresholds allows the algorithm to converge more quickly. If a sequence has no segments with a score higher than the low threshold, it is considered as not containing the motif.

Perhaps the most statistically intensive method for *de novo* motif searching using PWM updating is proposed by Liu et al. (1995). Their primary goal is to identify the most probable motif and alignment pattern. Their focus is on identifying TFBSs while accounting for the variability in sequences presented by mutations that occur during evolution. These mutations can cause misalignment in the binding sites and, therefore add variation and uncertainty to the data. Assuming independence in the positions allows them to describe TFBSs using what they refer to as *product multinomial models*. The *product multinomial model* will be discussed in detail later. It can be shown that the conjugate prior for a product multinomial model is a *product Dirichlet model*. The background model is described using a lower-order Markov chain.

The positions in the PWM are assumed to be independent observations from a product multinomial model. The parameter used to describe this product multinomial model is $\boldsymbol{\theta_j} = (\theta_{1,j}, \cdots, \theta_{4,j})$. The background parameter is represented by $\boldsymbol{\theta_0}$. In order to keep track of the alignment, a new variable, a_k, is presented. a_k represents the possible starting positions for the motif in each sequence. The a_ks are similar to the starting positions for each m-mer described in MEME. Thus, a_k is a missing observation. Allowing $A = (k, a_k + j - 1)$ where $k = 1, \cdots, K$, the length of the

sequence, and $j = 1, \cdots, J$, where J is the width of the motif, the following is the complete-data likelihood of the parameters.

$$\pi(\boldsymbol{R}, A|\boldsymbol{\theta_0}, \boldsymbol{\Theta}) \propto \boldsymbol{\theta_0}^{h(\boldsymbol{R}_{A^c})} \prod_{j=1}^{J} \boldsymbol{\theta_j}^{h(\boldsymbol{R}_{A(j)})}. \tag{2.8}$$

Note that $h(R_{.j})$ represents the sufficient statistics of θ_j. This likelihood can be generalized to multiple motifs by extending it to include a θ_i for each motif of interest. Lastly, the PWM is updated using the predictive posterior distribution for A to obtain, what is referred to as the *predictive update version* of the Gibbs sampler.

Motif Discovery scan (MDscan) is another computational method that searches for *de novo* motifs (Liu et al. 2002). MDscan combines the methods of regular word enumeration and PWM updating. MDscan also incorporates chromatin immunoprecipitation array (ChIP-array) ranking information. ChIP-array information is only important for the purposes of this algorithm in terms of data collection. ChIP-array experiments provide high-resolution maps of the interactions between the TFs and the DNA. MDscan first scans the highly ChIP-array-enriched fragments, generating candidate motif patterns. These candidate motifs are then updated using Bayesian statistical scoring functions.

MDscan starts with a list of n DNA sequences from ChIP-array experiments and ranks the sequences according to how enriched they are. The top three to twenty ranked sequences are used to form a set of candidate motifs. Assuming the motif to be of width w, MDscan searches for all w-mers in the top sequences with at least m base pairs matching the candidate motif, where m is chosen by the user. The m-matches in the top sequences are used to form a PWM. The following maximum *a posteriori* scoring function is then used to evaluate a PWM.

$$\frac{x_m}{w} * \left[\sum_{i=1}^{w} \sum_{j=A}^{T} p_{ij} log(p_{ij}) - \frac{1}{x_m} \sum_{all\ \ seqments} log(p_0(s)) - log(\frac{expected\ bases}{site}) \right], \tag{2.9}$$

where x_m is the number of m-matches in the motif, p_{ij} is the frequency of nucleotide j at position i of the PWM, and $p_0(s)$ is the probability of generating the m-match

s from a background model. A 3rd-order Markov model is used to generate the background. After the scores are computed for all candidate motifs, MDscan saves the highest scoring 10–50 candidate motifs for updating in the next iteration.

Another *de novo* motif searching method uses a conditional two-component mixture model (CTCM) to update a PWM (Shim and Keles 2007). The two-component mixture model is conditional on the ChIP-array data scores. The mixture model is similar to the mixture model presented by Bailey and Elkan (1995) in MEME, which they refer to as their two-component mixture (TCM) model. The notation for CTCM is as follows. $\tilde{X}_{i,j}$ represents the W-mer beginning at position j in the sequence X_i, and all the $\tilde{X}_{i,j}$ are assumed independent of each other. $Z_{i,j}$ is an indicator variable indicating whether $\tilde{X}_{i,j}$ belongs to the PWM or not. λ' represents the proportion of the data that belongs to the motif of interest. A Markov chain is used to model the background distribution. Using this notation, the following likelihood is used to repesent the $X_{i,l}$ conditional on $Z_{i,l}$.

$$Pr(\tilde{X}_{i,l}|Z_{i,l}=1,\Phi_W)=\prod_{k=1}^{W}\prod_{j=1}^{4}p_{kj}^{I(X_{i,l+k-1}=j)}, \tag{2.10}$$

where Φ_W refer to the parameters or positions in the PWM.

Another approach to *de novo* motif searching is a method known as the Hidden Markov Model (HMM) approach (Baldi et al. 1994). The HMM approach uses a set of training sequences to iteratively modify a set of given parameters using the product of the likelihoods of the sequences. The model is first reparameterized in order to preserve normalization of the probability distributions and to never allow transition probabilities to reach an absorbing state of 0. In the HMM approach, $\boldsymbol{T}=(t_{ij})$, or the transition matrix, and $\boldsymbol{E}=(e_{i\alpha})$, or the probability emission matrix. In other words, when a system is in state i, it has probability t_{ij} of moving to state j and probability $e_{i\alpha}$ of emitting symbol α. The most likely path through the model is computed using the Viterbi algorithm. More specifically, the reparameterized parameters are updated

as follows.

$$\Delta w_{ij} = \nu(T_{ij} - t_{ij}) \tag{2.11}$$

$$\Delta v_{i\alpha} = \nu(E_{i\alpha} - e_{i\alpha}), \tag{2.12}$$

where w_{ij} and $v_{i\alpha}$ are the reparameterized parameters and ν is a learning rate. The algorithm is expected to converge to a local maximum likelihood estimator. Finally, once the data are trained in this manner, new motifs are found by aligning the optimal paths using the maximum likelihood estimator.

One method for searching for a known motif is to simply scan the sequence for the occurrence of the motif. Though this approach is simple statistically, it is only effective when the motif has few variations that are well characterized (Johnson et al. 2009). A more computationally intensive known motif searching method was presented by Hertz et al. in 1990; the method is simply a database search using a scoring function. This allows for variation in the motif. The user first chooses the width of the motif of interest to search for, L. The algorithm then forms every possible list that contains exactly one L-mer from each of the N sequences. An L-mer is defined as follows. For each sequence, the algorithm builds possible PWMs by counting the number of times each base occurs at each position. Note that at each iteration, the algorithm only saves the PWM with the lowest probability of occurring by chance. This is due to the limited amount of computation space available when this method was created, allowing the algorithm to be more efficient. This probability for each motif is calculated as follows.

$$P = \sum_{i=1}^{L} \sum_{b=A}^{T} \frac{N_{bi}}{N} log_2 \frac{N_{bi}/N}{P_b}, \tag{2.13}$$

where $b = A, C, G, T$ and P_b is the genomic frequency of base b. This method can be considered a PWM updating approach for known motifs. Thus, if users know the PWM they are searching for, this method has the ability to update a PWM using collected data.

Some other interesting methods for motif searching are Motif Regressor, CisModule, and EMCMODULE. Motif Regressor combines PWM updating and motif-expression regression analysis (Conlon et al. 2003). Motif Regressor first uses MDscan to generate a large set of motif candidates. Each candidate motif is then given a motif-matching score similar to the ratio used in GMS. For each motif found by MDscan, a simple linear regression is fit such that the response is the log-expression value of the gene, and the predictors are the motif-matching scores. All motifs with significant coefficients are then placed into a multiple regression model from which stepwise selection is used to find an optimal model. The motifs that correspond to the coefficients in the final model are then considered statistically significant.

CisModule (Zhou and Wong 2004) is different from other motif searching methods in that it searches for groups of K motifs. These K motifs are considered modules, and the algorithm searches for the K motifs at one time. CisModule also uses a product multinomial model to describe binding sites and includes a two-level Bayesian hierarchical mixture model. Because the length between the K motifs needs to vary, a hierarchical model is needed to perform Bayesian inference. The hierarchical model consists of placing a Poisson prior distribution on W, the widths of the motifs of interest, which is also a parameter for the prior distribution of the PWM. CisModule then samples from the posterior distribution using Gibbs sampling. However, the addition of the prior distribution on W requires the use of the Metropolis-Hastings algorithm in order to execute Gibbs sampling techniques, making the algorithm computationally slower.

Another method for motif searching involving a cluster or module search is known as EMCMODULE (Gupta and Liu 2005). The algorithm starts with a collection of PWMs obtained from existing algorithms and databases. It then iteratively selects PWMs that are likely members of the module. It is assumed that K PWMs exist in the module of interest. In order to model the dependencies of these PWMs on

each other, a $K\,x\,K$ transition matrix is created, denoted by V. The module itself is allowed to occur anywhere in the DNA sequences with equal probability. The distances between the K PWMs, d_{ij}, are modeled using a truncated geometric distribution. The background noise is modeled by a lth-order Markov chain. Prior distributions are put on the transition matrix V, λ, the parameter in the truncated gemometric distribution, ρ, the transition probability of the background noise Markov chain, and K. The maximum *a posteriori* estimate for each of these parameters is then found using a forward-backward recursion method. More specifically, the EMCMODULE forward-backward recursion algorithm follows these three steps:

(1) Given the sequence configuration, the motif site locations are updated.

(2) Given the motif site locations, the K PWMs are updated.

(3) The sequence configuration is updated using Monte Carlo methods.

This technique is effective in identifying TF binding sites and their dependencies on each other.

Lawrence et al. (1993) suggest that frequent input of expert knowledge is needed as the quantity of sequence data available is growing. Most methods use information from TRANSFAC and JASPAR as fixed parameters when searching for motif binding sites, or as a comparison to illustrate how well a given method performs. Thus, information in TRANSFAC and JASPAR is usually considered the "truth." As discussed previously, we will use information from TRANSFAC as expert prior information in our new proposed method, XPRIME. XPRIME has the ability to search for both *de novo* motifs and known motifs at the same time. The user either gives a list of known PWMs to search for, the number of new motifs to search for, or both. The algorithm outputs posterior distributions and marginal values.

3. METHODS

3.1 Model Definition

Recall that assuming independence in the positions allows us to represent a PWM as the joint distribution of L independent multinomial distributions, where L represents the number of columns in a PWM. We will refer to this product of distributions as the *product multinomial distribution* as also described by Lawerence et al (1993) earlier. Lawerence et al (1993) suggest that the assumption of independence can be closely achieved through careful selection of the data set. The statistical methods used in this paper also work well even when this assumption is violated.

The multinomial distribution is simply an extension of the binomial distribution. The binomial distribution measures the number of successes in n independent Bernoulli trials. The parameter, p, in the binomial distribution represents the probability of success and allows for the same probability of success in each trial. The multinomial distribution allows for multiple probabilities of success, $p_1, \cdots, p_k$. In other words, as opposed to only the two outcomes of success or failure in each trial, the multinomial distribution allows for k outcomes in each trial, such that $\sum_{i=1}^{k} p_k = 1$. Consider a single column in a PWM. Recall that each column in a PWM must sum to one. Thus, a single column in a PWM can be modeled using a multinomial distribution such that k=4. Assuming independence across the columns of a PWM allows us to model the joint distribution of the positions in a PWM as a product of multinomial distributions, or as a *product multinomial distribution.*

Using a product multinomial distribution to represent a motif of length L, we can create a likelihood score for each segment of length L from a DNA sequence. These segments are also the L-mers as described in MEME. We let $\mathbf{y}$ represent a candidate DNA sequence, and y_j represent the jth nucleotide in the candidate sequence. p_{ij}

will represent the *(i,j)*th element of the PWM, and $I(y_j = i) = 1$ if $y_j = i$ and $I(y_j = i) = 0$ otherwise. The likelihood score for the motif at any given L-mer in a sequence of DNA can be represented by the likelihood for a product multinomial distribution:

$$f(\mathbf{y})_{Motif\ Score} = \prod_{j=1}^{L} \sum_{i \in A,C,G,T} p_{ij} I(y_j = i). \tag{3.1}$$

All the data used in our model were pulled from ChIP-sequence (ChIP-seq) experiments and consist of S sequences which will be denoted by $z_1, \cdots, z_S$, with corresponding lengths, $N_1, \cdots, N_s$. In general, $z_s = (y_{is}, \Delta_{1i}, \Delta_{2i}, \cdots, \Delta_{(m+1)i})$. y_{is} represents the ith L-mer in sequence s. The Δs are simply indicator variables indicating which motif y_{is} belongs to, such that $\Delta_{Mi} = 1$ if y_{is} belongs to motif M and $\Delta_{Mi} = 0$ if y_{is} does not belong to motif M, where $M = 1, 2, \cdots, m + 1$. Also notice that only one Δ can be equal to one for each y_{is}. Thus, the algorithm does not allow for y_{is} to belong to more than one motif. Finally, m is the total number of known motifs and new motifs the user is interested in searching for. One more Δ is included to represent a background noise motif.

Incorporating the data and motifs of interest allows us to represent the complete data likelihood function as follows. The parameters are represented by

$$\boldsymbol{\theta} = ([p_{jk}], r_1, r_2, \cdots, r_{m+1}), \tag{3.2}$$

where $[p_{mjk}]$ represents the PWM for motif M with positions *(j,k)* and r_M represents the proportion of data that belongs to motif M. More specifically, r_M is the ratio of the total number of L-mers that belong to motif M to the total number of L-mers that belong to something other than motif M, and $C(y_i)$ is a normalizing constant. Note that Δ_{Mi} is an unobservable part of the data. $f_m(y)$ is the motif score equation derived from the likelihood of the product multinomial distribution and represents the distribution of the mth binding motif of interest. Thus, r_M also represents the proportion of the DNA sequence that belongs to $f_M(y)$. $f_{m+1}(y)$ represents the distri-

bution of the background motif of a DNA sequence. $f_{m+1}(y)$ is fixed *a priori* and will be discussed in detail below, and $\sum_{i=1}^{m+1} r_i{=}1$. Finally, the complete data likelihood for one DNA sequence is

$$L(\boldsymbol{\theta}|\boldsymbol{z}) = \prod_{i=1}^{N} C(y_i)[r_1 f_1(y_i)]^{\Delta_{1i}}[r_2 f_2(y_i)]^{\Delta_{2i}}[r_3 f_3(y_i)]^{\Delta_{3i}} \cdots [r_{m+1} f_{m+1}]^{\Delta_{(m+1)i}}. \quad (3.3)$$

Since there is no available PWM from TRANSFAC to represent the background motif, it is randomly generated. The columns are assumed to be independent, and prior to running the algorithm, each column of the background motif is randomly drawn from a Dirichlet distribution. The parameters used to draw from the Dirichlet distribution are p_A, p_C, p_G, and p_T, such that p_A represents the proportion of the complete data set that are the nucleotide A, and so forth for p_C, p_G, and p_T. We chose to create the background matrix in this manner in effort to avoid any bias toward one or more of the nucleotides in the data. Some of our data sets are rich in one or two nucleotides. Thus, we hope to allow the background to represent the random noise of the nucleotides within our given data set. The width of the background motif is such that it is the same length as the longest motif of interest. For example, if we choose to search for two known motifs of interest, the ETS1 TF and the RUNX TF, where ETS1 is of length eight and RUNX is of length seven, the background motif would be of length eight. This is for simplicity in the computation of the algorithm and does not significantly alter our results. The background matrix is also scored using the product multinomial likelihood equation above. However, it is only scored once and not at each iteration of the algorithm. Finally, an L-mer will belong to the background motif, $f_{m+1}(y)$, when it does not belong to any other motif, $f_1(y)$, $f_2(y), \cdots$, or $f_m(y)$.

Product Dirichlet priors were chosen as prior distributions for $f_1(y), f_2(y), \cdots$, and $f_m(y)$. The Dirichlet distribution is the conjugate prior for the multinomial distribution, and it can be shown that the *product Dirichlet distribution* is conjugate for the product multinomial distribution. In Bayesian theory, the posterior distribution,

$\pi(\theta|x)$, is proportional to the likelihood of the data multiplied by the prior. More specifically,

$$\pi(\theta|x) = \frac{\pi(x|\theta)\pi(\theta)}{\int \pi(x|\theta)\pi(\theta)d\theta}. \tag{3.4}$$

The prior distribution, $\pi(\theta)$, is said to be conjugate to the likelihood function, $\pi(x|\theta)$, if the posterior distribution, $\pi(\theta|x)$, is in the same family of distributions as $\pi(\theta)$. Using a product Dirichlet distribution as the prior distribution for $f_m(y)$ results in a Dirichlet posterior distribution. This information makes drawing samples from the posterior distribution in our algorithm simpler because we can know the posterior distribution.

The Dirichlet distribution is an extension of the beta distribution. The beta distribution is such that the counts of successes in n independent Bernoulli trials is the measured parameter, as opposed to the probability of success, which is the measured parameter in the binomial distribution. Similarly, the parameters for the Dirichlet distribution are the counts of each k success from n independent trials where there can be k different outcomes each trial. Each PWM taken from TRANSFAC has a matrix analogous to the PWM that includes the expected counts of each nucleotide in each position as opposed to the probabilities. Each column still needs to sum to the same number. For example, the columns in an ETS1 PWM that was shown in the Introduction section of this paper has an analogous matrix where each of the columns add to 15 instead of one. In order to make these matrices PWMs, the user simply divides each element of the matrix by the sum of one of its columns. We allow the elements of these counted PWMs, $[p_{ij}]$, to be the parameters for the product Dirichlet priors. The algorithm can still take a PWM of probabilities as prior parameters without problems as it is just a scaled version of the counted PWMs.

Diffuse product Dirichlet priors are chosen for each new or *de novo* motif the user would like to search for. For computational simplicity, the current algorithm constrains the *de novo* motifs to be of the same width as the background motif. We

expect the algorithm to find background noise for some of the *de novo* motifs we search for as we do not know how many *de novo* motifs exist in our data set or if any exist at all. Future research will consider searching for *de novo* motifs of various lengths. One idea is to place a prior on the width of the *de novo* motifs, similar to the approach taken in CisModule (Zhou and Wong 2004), thereby allowing the widths to vary. The diffuse priors placed on the *de novo* PWMs are given an equal probability for each nucleotide at each position. Thus, the PWMs for the *de novo* motifs are simply $4 \times w$ matrices of 0.25 where w is the width of the background motif. An analogous "expected count" matrix to this PWM would be a $4 \times w$ matrix of ones.

A prior distribution is also specified for $r_1, r_2, \cdots, r_{m+1}$. Recall that r_i represents the proportion of DNA that belongs to $f_i(y)$ for $i = 1, 2, \cdots, M$. This requires $\sum_{i=1}^{M} r_i$=1. By definition $\boldsymbol{r}$ can be represented as a multinomial distribution. Thus, a Dirichlet prior was chosen for $\mathbf{r}$. The thoughtfulness, or expert knowledge of the prior parameters is left for the user to decide. The parameters for the Dirichlet prior for $\mathbf{r}$ can be chosen by the user if desired. Thus, if the user feels the proportion of $f_1(y)$ will be particularly high in a given dataset, he or she may specify this in the prior parameters for $\mathbf{r}$. If the user does not wish to specify prior parameters for $\mathbf{r}$, the algorithm chooses the parameters such that $r_1 = r_2 = \cdots = r_{m+1}$. It is not recommended that the user perform extra computation to solve for these prior parameters as it is expected that the posterior distribution for $\mathbf{r}$ would converge quickly. This will also be shown in the Results section of this paper.

The algorithm also considers the fact that a TF can also bind to its reverse complement. Each nucleotide of DNA also has a complement. The nucleotides A and C are complements of each other and the nucleotides G and T are complements of each other. Suppose we are searching for the binding motif "GGA" in a DNA sequence. The TF that can bind to "GGA" can also bind to the reverse complement "TCC." Thus, if we are searching for the presence of the TF that binds to "GGA" in

our sequence, we must also consider the binding motif "TCC." In order to account for this, when given a PWM, our algorithm creates a *reverse complement PWM* which it will also search for. For example, suppose the algorithm was given the following PWM to search for:

$$\begin{array}{cc} \text{Position:} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} & \begin{pmatrix} 0.300 & 0.000 & 0.000 & 0.100 \\ 0.600 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.800 \\ 0.100 & 0.000 & 1.000 & 0.100 \end{pmatrix} \end{array}.$$

The algorithm would then create a reverse complement PWM by reversing the rows and the columns such that the reverse complement PWM of the above example would be

$$\begin{array}{cc} \text{Position:} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} & \begin{pmatrix} 0.100 & 1.000 & 0.000 & 0.100 \\ 0.800 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.600 \\ 0.100 & 0.000 & 0.000 & 0.300 \end{pmatrix} \end{array}.$$

The counts both the regular PWM and the reverse complement PWM that are discovered by the algorithm are then added together.

3.2 ETS and RUNX

The algorithm will be presented using an example in which we wish to update the PWMs for two known motifs and in which we wish to search for nine *de novo* motifs. We would like to search our data for the known transcription factors ETS1 and RUNX. The ETS1 TF is significant in promoting the transcription of T-cells. T-cells belong to a group of white blood cells and play a signficant role in the body's immune system. The ETS1 TF has also been found to be associated with the progression of

malignant tumors. Recently, The ETS1 TF has been found to be expressed in the presence of the skin cancer gene; however, no significant correlation between the ETS1 TF and skin cancer was found. (Torlakovic et al. 2004) Nevertheless, due to the association between the ETS1 TF and the progression of malignant tumors, the hypothesis that the ETS1 TF contributes to the progression of skin cancer cannot be ruled out.

It has also been suggested that the ETS1 TF may be associated with rheumatoid arthritis and diabetic retinopathy. More specifically, its presence may aid in the treatment of these diseases. Forough et al (2006) found that the activation of ETS1 is required for fibroblast growth factor 1 (FGF-1) mediated angiogenesis *in vivo*, suggesting that ETS1 might be a potential target for inhibitor drugs in the treatment of FGF-dependent diseases such as rheumatoid arthritis and diabetic retinopathy (Forough et al. 2006). The various roles of the ETS1 TF lends great importance to further understanding of the transcription factor's behavior.

GABP is a TF that belongs to a family of TFs called ETS. ETS1 also belongs to this family of TF. Specifically, this family is represented by the motif "GGAA." A family of transcription factors will bind to the same motif, such as "GGAA." Thus, if the "GGAA" motif is found to be significant in the regulation of a specific gene, there could be multiple transcription factors associated with the regulation of that gene. It is believed that there are not any genes that are specifically controlled by GABP. Current hypotheses suggest that the ETS1 TF will bind more frequently and be more defined in the presence of the GABP TF. This is due to the existence of the family binding motif. In order for GABP and ETS1 to co-occupy the same binding motif, the DNA word "GGAA" must be present.

It has also been hypothesized that the ETS1 TF will bind more frequently when accompained by a RUNX TF (Hollenhorst et al. 2007). The RUNX TF is also known as AML due to the association that has been found between the RUNX TF

and Acute Mioblastic Leukemia (AML). The RUNX TF has also been found to be associated with various aspects of embryonic development, specifically development associated with the nervous system (Inoue et al. 2008). Due to the recent discovery of the possible relationship between ETS1 and RUNX by Hollenhorst et al. (2007), we have decided to search our data sets for both the ETS1 TF and the RUNX TF. The implications of this relationship are that an overabundance of the RUNX TF without the ETS1 TF or with a weaker ETS1 TF may result in an individual developing immunodeficient disesases such as AML. Thus, it is critical that we can understand the ETS1 TF's binding behavior when in the presence of its family binding motif, GABP, and without GABP. It is of great interest to discover the possibility of an ETS1 specific binding site. We would like to see if the ETS1 TF will bind to sites other than the family site. This would suggest that ETS1 is directly correlated with the regulation of the T-cell gene, and that the ETS1 TF has a specific purpose separate from the family of TFs. We would also like to understand any relationship between ETS1 and RUNX.

We will use our algorithm to update the PWMs for ETS1 and RUNX TFs as well as search for *de novo* motifs in the following collections of DNA sequences.

(1) A collection of sequences where transcription is expected to occur with only the ETS1 binding motif. This data set will be referred to as ETS1 only.

(2) A collection of sequences where transcription is expected to occur only with the GABP binding motif. This data set will be referred to as GABP only.

(3) A collection of sequences where transcription is expected to occur and both the ETS1 and GABP motifs are expected to bind. This data set will be referred to as ETS1 and GABP.

The second data set will act as a control set. We can use the ETS1 PWM as prior information for the ETS family. This will allow us to identify the difference between a

family binding site and an ETS1 specific binding site. We hope to see some significant differences between the two PWMs. We would also like to see if RUNX binds more in the presence of the ETS1 specific binding motif. Finally, nine *de novo* motifs will also be searched for in each of these data sets. We chose to search for nine as it corresponds nicely to the number of processors we are using to run the algorithm. Searching for nine allows each processor to search for one motif. We would like to possibly identify TFs other than RUNX that may contribute to the ETS1 specific binding sites.

The known motifs for ETS1 and RUNX were taken from the TRANSFAC database. The PWM from TRANSFAC for the ETS1 TF has already been presented. The PWM of counts for the ETS1 TF is found by simply multiplying all of the positions in the PWM by 15. The PWM for the RUNX TF according to the TRANSFAC database can be seen below.

$$\begin{array}{cc} \text{Position:} & \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{array}{c} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} & \left(\begin{array}{ccccccc} 0.000 & 0.00 & 0.000 & 0.00 & 0.00 & 0.000 & 0.059 \\ 0.059 & 0.00 & 0.176 & 0.00 & 0.00 & 0.059 & 0.235 \\ 0.000 & 1.00 & 0.000 & 1.00 & 1.00 & 0.059 & 0.118 \\ 0.941 & 0.00 & 0.824 & 0.00 & 0.00 & 0.882 & 0.588 \end{array} \right). \end{array}$$

A more visually appealing way of representing a DNA binding motif is to use a sequence logo. A sequence logo is a graphical representation of a PWM where the relative height of each nucleotide within each position represents its frequency, p_{ij}. The relative height of the nucleotides between each position represents the significance or importance of the binding positions. Availabe to the public is a sequence logo generator known as Weblogo (Crooks et al. 2004). Weblogo allows the user to input a series of randomly generated motifs and outputs a sequence logo. We generated the following sequence logo using the TRANSFAC PWMs. The sequence logo for both

the ETS1 and RUNX TF can be seen below.

Figure 3.1: DNA binding motif for the ETS1 TF according to TRANSFAC

Figure 3.2: DNA binding motif for the RUNX TF according to TRANSFAC

Notice that some of the prior probabilities specified for positions p_{ij} are equal to one. Specifically, in the ETS1 TF sequence logo there is a probability of one associated with positions three, four, and five such that the TF is binding to "GGA" with 100% probability at theses positions. In Bayesian applications, this may seem very strict for a prior specification. However, biochemically, an ETS1 TF is defined by a binding site at "GGA." The uncertainty associated with an ETS1 TF is where it binds prior to and after it binds to "GGA." More specifically, biologists are interested in the position immediately following "GGA" as it has been shown to toggle between A and T. It is suggested that this position will bind mostly to nucleotide "A" in position 6 when the GABP TF is not present. The binding in position 6 is suggested to be weaker in the presence of the GABP TF. Our approach will allow us to see the distribution of the probabilities in this position so that we can further understand the probability associated with this position.

3.3 Gibbs Sampling Procedure

The Gibbs sampling procedure has the following steps.

(1) Start with $\theta^{(0)} = ([p_{jk}]^{(0)}, r_1^{(0)}, r_2^{(0)}, \cdots, r_M^{(0)})$.

(2) At step i, generate

$$\begin{aligned}
(p_{jk})^{(i)} &\sim [(p_{jk})|r_1^{i-1}, r_2^{i-1}, \cdots, r_M^{i-1}, \mathbf{y}] \\
r_1^{(i)} &\sim [r_1|(p_{jk})^{(i)}, r_2^{i-1}, r_3^{i-1}, \cdots, r_M^{i-1}, \mathbf{y}] \\
&\vdots \\
r_M^{(i)} &\sim [r_M|(p_{jk})^{(i)}, r_1^{i}, r_2^{i}, \cdots, r_m^{i}, \mathbf{y}].
\end{aligned}$$

(3) Iterate N^* times until there is a large enough posterior sample.

Step 1 are the initial prior parameters that were discussed earlier. Step 2 are referred to as the *complete conditionals* of the parameters. The complete conditional distributions are found by first calculating the complete posterior distribution. Once this is calculated, the complete conditional distribution for parameter θ_1 is found by solving for the distribution of θ_1 while assuming all other parameters are constant.

For our example, we allow $m = 1, 2, \cdots, 12$, where $m = 1$ represents the ETS1 motif, $m = 2$ represents the RUNX motif, and $m = 3, 4, \cdots, 12$ represent the *de novo* motifs. The complete posterior distribution for $m = 1, 2, \cdots, 12$ can be written as follows:

$$\pi(\theta|\mathbf{y}) = L(\boldsymbol{\theta}|\boldsymbol{z})\pi(f_1(y))\pi(f_2(y)) \cdots \pi(f_{12}(y))\pi(\mathbf{r}). \tag{3.5}$$

The complete conditional distribution for each parameter that has a specified prior can be seen below.

$$[f_M(y)|\Delta_{ji}] = \prod_{i=1}^{N}(f_M(y_i))^{\Delta_{Mi}} \prod_{j=1}^{L_s} \prod_{k\in(A,C,G,T)} p_{Mjk}^{\alpha_{Mij}-1} \tag{3.6}$$

$$[\mathbf{r}|\Delta_{ji}] = \prod_{i=1}^{N}[r_1 f_1(y_i)]^{\Delta_{1i}}[r_2 f_2(y_i)]^{\Delta_{2i}} \cdots [r_M f_M(y_i)]^{\Delta_{Mi}} r_1^{a_1-1} r_2^{a_2-1} \cdots r_M^{a_M-1}. \tag{3.7}$$

Notice $[f_M(y)] \propto \text{Dirichlet}(a_{Mij}*)$ where $a_{mij}* = \sum_{i=1}^{L} \Delta_{Mi} + \alpha_{Mij}$. Thus, we can write $[\mathbf{f}] \propto \text{Dir}(a_{Mij}*)$ where L is the length of the sequence of interest. α_{Mij} are the prior parameters for $f_M(y)$ which are simply the counts from the PWM. In general, α_{Mij} is the count from motif M's PWM, position j, nucleotide i.

Notice that the complete conditionals are dependent on the Δ's, which are missing. Recall that Δ_M was assumed to have a multinomial distribution. Thus, we can simply generate the Δs from a multinomial distribution. Notice also that the distribution of Δ_M depends on both $\mathbf{r}$ and $f_i(y)$. Thus, Δ_M can be successively sampled from within the Gibbs sampler. Our algorithm randomly generates values of Δ_M from a multinomial distribution at every draw from the posterior complete conditionals.

Currently the XPRIME algorithm requires that all motifs searched for be of the same length. In our example, the ETS1 TF is of length eight and the RUNX TF is of length seven. The algorithm adjusts for this problem as follows. One column of background noise was generated in the same manner in which the columns of the background motif were generated and is attached to the end of the RUNX motif. In general, the algorithm will attach columns of background noise to every smaller known motif until it is of the same length as the longest motif.

Because the complete conditional distributions above all have well-known distributions, sampling from them was computationally simple. If these complete conditional distributions were not well-known, the incorporation of the Metropolis-Hastings algorithm would be needed to employ the Gibbs sampler procedure. Without the

Metropolis-Hastings algorithm, XPRIME is more computationally efficient. XPRIME was written in the statistical program R and the code for XPRIME can be found in the appendix.

3.4 The Algorithm

The XPRIME algorithm takes the following steps,

(1) Draws Δs from a multinomial distribution with parameters $p_\Delta \propto r_M * f_M(y)$.

(2) Draws $\boldsymbol{r}$ from a Dirichlet distribution with parameters $\alpha_r = \alpha_{Mij}*$.

(3) Draws p_{Mij} from a Dirichlet distribution with parameters

$$\alpha_{p_{Mij}} = \sum_{i=1}^{L_S} \sum_{k=\{A,C,G,T\}} \Delta_{Mi} I(y_{ij} = k) + a_{p_{Mij}}.$$

(4) Repeats steps 1 through 3 N^* times

4. RESULTS

4.1 Data Sets Used

We ran the XPRIME algorithm on the data sets described in the methods section above. The first data set consists of 1,496 DNA sequences in which ETS1 specific binding sites are expected to be present. We wish to search this data set for both the ETS1 TF and the RUNX TF. We would like to update the ETS1 PWM from the TRANSFAC database to see how well-defined the ETS1 TF is without the presence of GABP TF. Because it is not believed that there are any genes specifically regulated by GABP, it is a good TF to use to represent the family of ETS TFs. Also, GABP will only bind to the ETS family binding sites of "GGAA." Thus, we can look for specific ETS1 binding sites without the presence of the family binding sites. We would also like to update the RUNX TF and see how often the RUNX TF is binding. We hypothesize that when the ETS1 TF binds without a GABP binding site, the RUNX TF will be binding more frequently and close by. We will define a close binding site by a RUNX TF that appears within the same sequence as an ETS1 TF. We also hypothesize that the ETS1 TF will be less defined, particularly in position six. We decided to search the ETS1 only data set for a less defined ETS1 motif. Specifically, we searched for positions three through six as they are defined in TRANSFAC, and allowed positions one, two, seven, and eight to have an equal probability to be any nucleotide. This allows the ETS1 specific motif to be biochemically defined to bind to "GGA" and the other positions to be diffusely defined. We also searched for nine *de novo* motifs in this data set.

Next, we ran the XPRIME algorithm on a data set that is expected to contain only GABP binding sites. This data set also consists of 1,496 DNA sequences. This data set was scanned as a type of control data set. Using the ETS1 PWM from

TRANSFAC, we could search for and identify the ETS family binding motif. We would then like to compare the family binding motif to the ETS1 specific binding motif searched for in the first data set. We hope to see some differences between the two, suggesting that ETS1 has its own specific binding sites outside its family binding sites. We would also like to examine the behavior of the RUNX TF. We would like to see if RUNX really binds more when an ETS1 specific binding site is present as compared to the presence of the ETS family binding sites. We also searched this data set for nine *de novo* motifs.

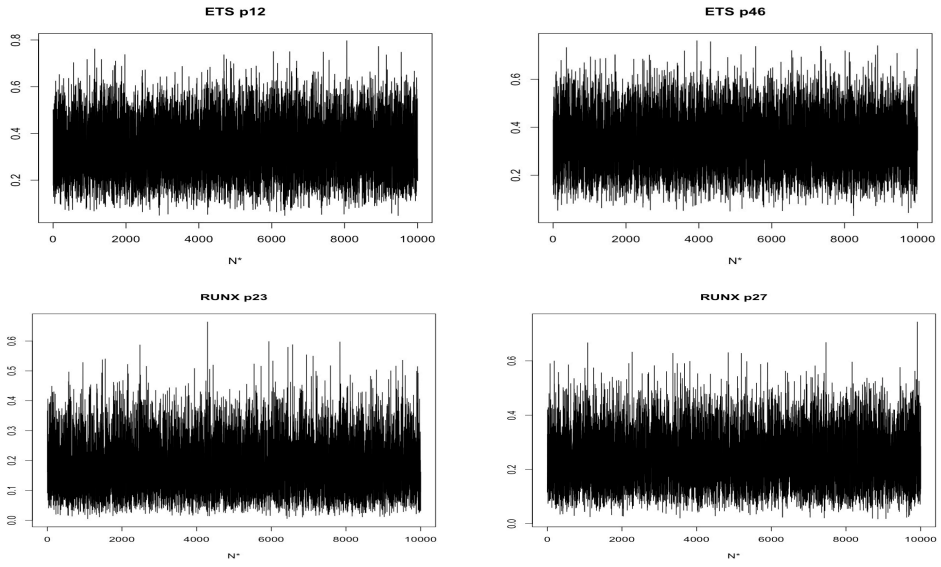
The final data set on which we ran the XPRIME algorithm consists of 1,264 DNA sequences in which both ETS1 specific binding sites and GABP binding sites are expected to be present. We would like to search this data set for the ETS1 and RUNX TF as well. We expect to see a stronger binding motif for the ETS1 TF when GABP sites are present, as ETS1 will be binding to both its specific site and the family site. In particular, we are interested in position six of the ETS1 TF. We expect to see the ETS1 TF binding more to "GGAA" in positions three through six, as compared to the ETS1 only data set. However, as compared to the GABP only data set, we expect ETS1 to bind less strongly to "A" in position six as it is also binding to its specific sites. We would also like to observe the behavior of the RUNX TF. We also searched for nine *de novo* motifs in this data set.

4.2 Convergence of Algorithm

After running all data sets on the XPRIME algorithm, we first checked if the convergence criterion of the algorithm was met. N^*=10,500 iterations were run on each data set. Note that some of the runs do not have exactly 10,500 iterations due to time constraints on the computer used. However, all of the runs with the XPRIME algorithm have at least 7,000 iterations. A burn-in period of 500 iterations was included to allow for convergence of the Gibbs sampler. Trace plots of the output

are useful in analyzing convergence of the algorithm and were used to check for convergence. It is expected in a Gibbs sampling technique that the draws from the posterior distribution will converge to the true values of the posterior distribution. Any sort of trend in a trace plot is an indication of the algorithm failing to reach convergence. Due to the large amount of parameters estimated by our algorithm, only a few trace plots demonstrating the convergence of our algorithm are presented below. The following are some of the trace plots of the posterior draws from $[p_{mij}]$ and r_m. The trace plots for select positions in the PWM for the ETS1 TF and select positions in the PWM for the RUNX TF can be seen below. Specifically, trace plots for positions p_{12} and p_{46} from the ETS1 PWM and trace plots for positions p_{23} and p_{27} from the RUNX PWM can be seen below. These positions were chosen as they are good examples of the quick convergence achieved by the XPRIME algorithm. These trace plots were taken from the ETS1 only data set.

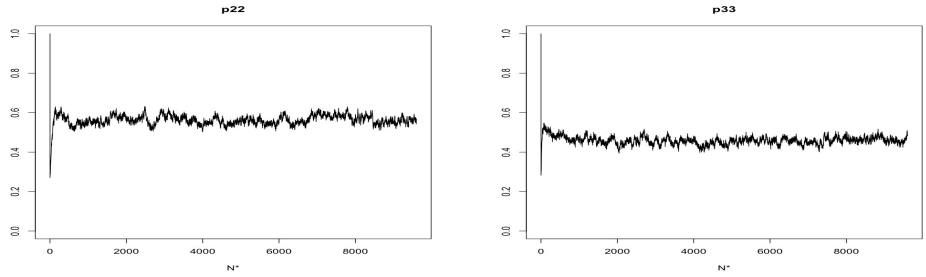
Figure 4.1: Trace plots for select positions in the ETS1 and RUNX PWMs



Notice the quick and almost immediate convergence of the algorithm. This means in the future, the algorithm can achieve similar results using fewer iterations. All other trace plots from the ETS1 PWM and the RUNX PWM have similar convergence.

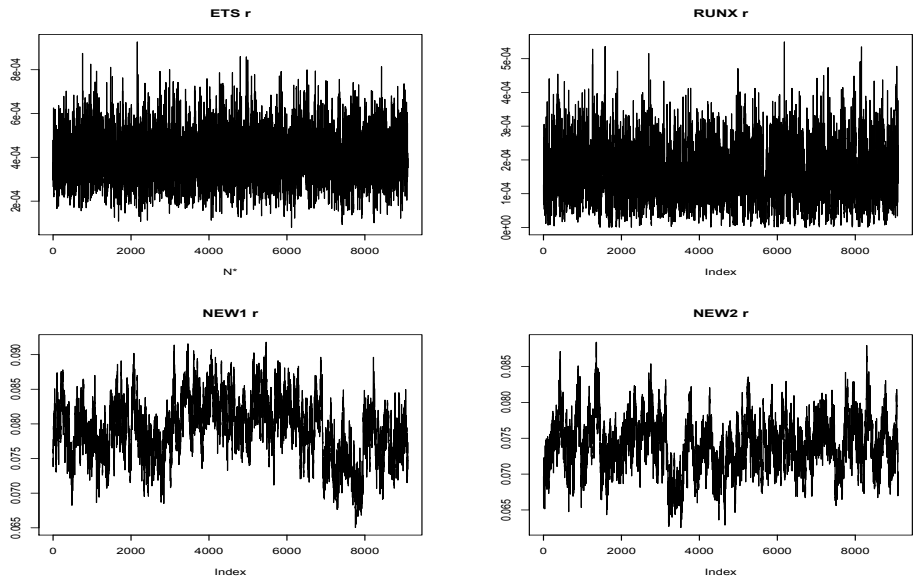
Some the trace plots for the *de novo* motifs look different and some do not achieve convergence. This is not a concern as we expect some of the *de novo* motifs to pick up background noise. If this happens, the positions will wander around searching for something concrete. Also, our interest in this particular example were the ETS1 and RUNX motifs. The *de novo* motifs were searched for as an example of how XPRIME searches for *de novo* motifs. If we believed our DNA sequences contained *de novo* motifs, we would have needed to run many more iterations in an effort to achieve convergence. Examples of trace plots for some of the *de novo* motifs can be seen below.

Figure 4.2: Trace plots for select positions in the *de novo* PWMs



We also checked the convergence of the marginal rs. Trace plots for r_1,r_2,r_3, and r_4; the proportions that correspond to the ETS1 TF; RUNX TF; and two of the *de novo* TFs can be seen below.

Figure 4.3: Trace plots for select proportions, r_m



Notice how the trace plots for the *de novo* motif proportions above wander around. Again, this is not of large concern for the *de novo* motifs as we do not expect to find a new motif for each *de novo* motif we search for. Thus, some of them will wander around without convergence because they are not finding anything.

After establishing convergence of the algorithm, we were interested in the posterior means for each p_{ij} in the ETS1 and RUNX TFs. The posterior means have been placed in a PWM similar to the prior PWM. Since the prior PWMs came from the TRANSFAC database, comparing the two can give us an idea of where some of the uncertainties in TRANSFAC may exist. In other words, we will more accurately be able to identify the uncertainty and variation among the positions in the TFs that have for so long been considered the "truth." We will also be able to view the different ways in which the ETS1 TF behaves when in the presence of GABP. Below are the posterior mean PWMs for ETS1 TF from each data set searched. Since we are not as interested in the updated RUNX TF as we are interested in the number of occurrences of the RUNX TF, the posterior mean PWMs for the RUNX TF can be

found in the appendix.

ETS1 Posterior mean from the ETS1 only data set:

Position:	1	2	3	4	5	6	7	8
A	0.097	0.377	0.00	0.00	1.00	0.726	0.162	0.049
C	0.903	0.575	0.00	0.00	0.00	0.091	0.078	0.473
G	0.000	0.000	1.00	1.00	0.00	0.000	0.760	0.000
T	0.000	0.048	0.00	0.00	0.00	0.183	0.000	0.479

ETS1 (Positions 3-6) posterior mean from the ETS1 only data set:

Position:	1	2	3	4	5	6	7	8
A	0.116	0.136	0.00	0.00	1.00	0.829	0.337	0.113
C	0.218	0.114	0.00	0.00	0.00	0.079	0.167	0.243
G	0.133	0.617	1.00	1.00	0.00	0.000	0.303	0.153
T	0.533	0.132	0.00	0.00	0.00	0.092	0.193	0.491

ETS1 Posterior mean from the GABP only data set:

Position:	1	2	3	4	5	6	7	8
A	0.103	0.255	0.00	0.00	1.00	0.924	0.164	0.091
C	0.897	0.735	0.00	0.00	0.00	0.016	0.112	0.374
G	0.000	0.000	1.00	1.00	0.00	0.000	0.723	0.000
T	0.000	0.048	0.00	0.00	0.00	0.059	0.000	0.535

ETS1 Posterior mean from the GABP/ETS1 data set:

Position:	1	2	3	4	5	6	7	8
A	0.079	0.403	0.00	0.00	1.00	0.877	0.173	0.101
C	0.921	0.575	0.00	0.00	0.00	0.021	0.054	0.360
G	0.000	0.000	1.00	1.00	0.00	0.000	0.773	0.000
T	0.000	0.022	0.00	0.00	0.00	0.102	0.000	0.539

Notice how the probabilities change across position six. As expected, ETS1 TF binds more to "A" in position six in the GABP only data set. Also, the ETS1 TF binds less to "A" in position six in the ETS1 only data set. Also notice position one. The ETS1 only data set shows varied probabilities for each nucleotide. The GABP only and the GABP and ETS1 data sets show the ETS1 TF as binding mostly to "C" in this position. The binding in this position to nucleotide "C" is also a deterministic characteristic for the ETS family.

Sequence logos were then created using the above posterior mean PWMs. The sequence logo from each data set as well as the sequence logo from TRANSFAC are given below for comparison. The sequence logos for the RUNX TFs can be seen in the appendix.

Figure 4.4: TRANSFAC: ETS1 Sequence logo

Figure 4.5: ETS1 only data: ETS1 Sequence logo

Figure 4.6: GABP/ETS1 data: ETS1 Sequence logo

Figure 4.7: GABP only data: ETS1 Sequence logo

Figure 4.8: ETS1 only data: ETS1 (Positions 3-6) Sequence logo

Notice how position six in the ETS1 TF changes across each data set. Position six in the ETS1 only data set is much weaker than position six in the GABP only data set. Specifically, it binds more frequently to "T" and less frequently to nucleotide "A." Position six in the ETS1 TF also binds more frequently to "A" in the GABP and ETS1 data set than in the ETS1 only data set. This supports our hypothesis that the ETS1 TF will bind more frequently to "GGAA" when in the presence of GABP binding sites. This also supports our hypothesis that the ETS1 TF may have its own specific binding motif separate from the ETS family binding sites.

Next, notice the weaker ETS1 binding motif defined only in positions three, four, five, and six and allowing equal binding probabilities for the other positions. It is an interesting result that this weaker ETS1 binding motif binds more frequently to "A" in position six and less frequently to "T." The biological implications of this result are not fully understood. However, this may indicate that ETS1 specific binding motifs are defined past the family binding site of "GGAA." This motif will be more useful in understanding the behavior of the RUNX motif. We will compare how often the RUNX TF binds close to the ETS1 TF in the ETS1 only data set to how often the RUNX TF binds close to the weaker ETS1 TF. If we can show that the RUNX TF binds more frequently close to the weaker ETS1 TF, we may be able to infer that weaker ETS1 transcription factors contribute to immunodeficient diseases.

We are also interested in the posterior distributions of the positions in both the ETS1 TF and the RUNX TF. Specifically, we are interested in the amount of variation associated with these positions. The posterior distribution for the positions in the ETS1 TF and RUNX TF taken from the ETS1 only data set are given below. The positions of the plots correspond to the positions of the p_{ij}s in the PWMs.

Figure 4.9: ETS1 only data: Posterior distribution of the ETS1 TF

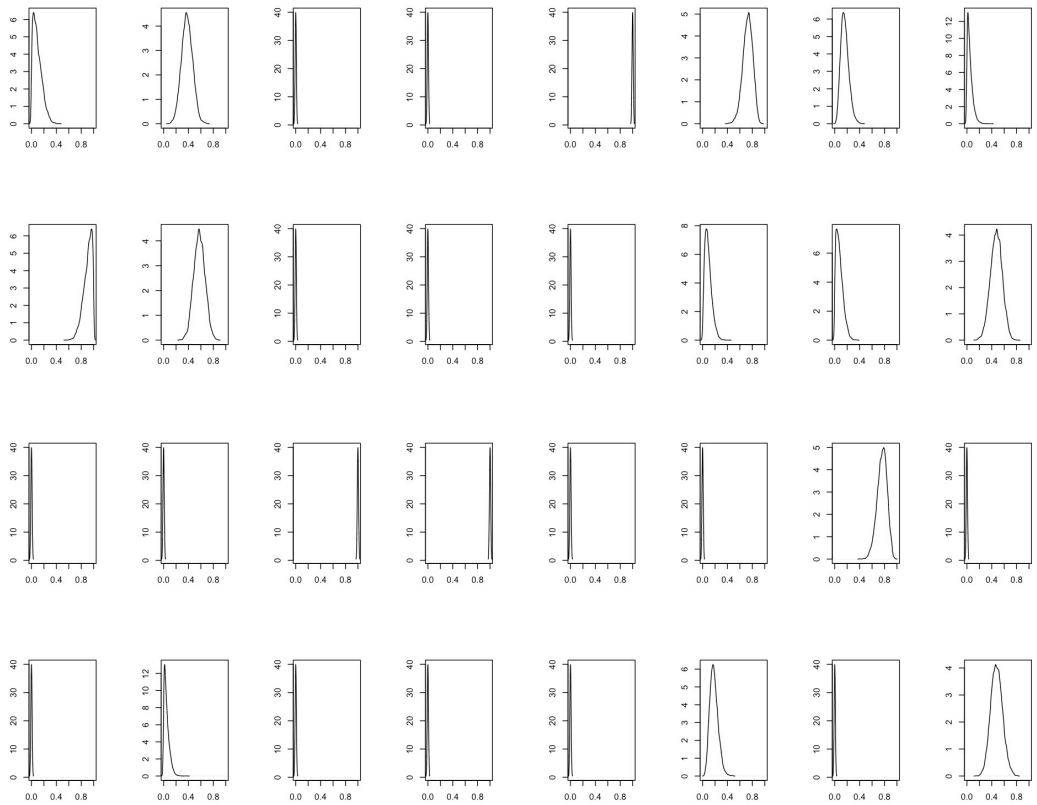
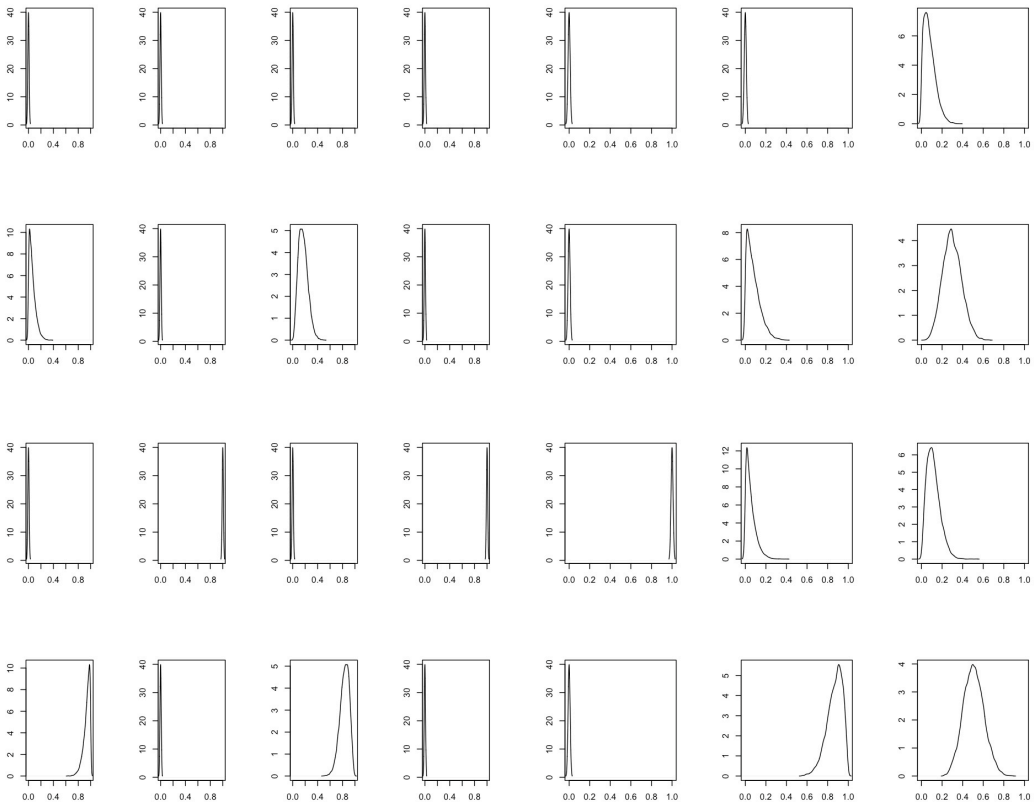


Figure 4.10: ETS1 only data: Posterior distribution of the RUNX TF



Notice the variation associated with the positions that are not biochemically defined. Many available methods for motif searching using PWM updating only calculate the posterior means or the *maximum a posterior* estimate for each position. This is done as it is believed that the positions in a PWM have little to no variation. These results suggest that the entire posterior distribution provides more information on the behavior of these binding sites than the posterior means alone, supporting the methods used in our algorithm.

In order to explore the behavior of the RUNX TF, the expected number of ETS1 and RUNX motifs were found by summing the marginal Δs. The followig table shows the expected number of ETS1 and RUNX motifs within each data set.

Table 4.1: Marginal Δs

Motif	Data Set	Expected Count
ETS1	ETS1 only	33
Weak ETS1	ETS1 only	202
ETS1	GABP only	524
ETS1	GABP/ETS1	500
RUNX	ETS1 only	23
RUNX	GABP only	10
RUNX	GABP/ETS1	59

Notice how few ETS1 TFs were found in the ETS1 only data set as compared to the weaker ETS1 TF. Also notice how many more ETS1 TFs were found in the data sets containing GABP. This supports the hypothesis that ETS1 TF may have its own specific binding motif separate from the ETS family. This may also support our hypothesis that the ETS1 TF will have a weaker binding site in the presence of RUNX. However, the RUNX TF does not bind significantly more in the ETS1 only data set. In order to explore the hypothesis that the ETS1 TF will have weaker binding sites in the presence of RUNX, the following tables have been made. First, the number of ETS1 and RUNX TFs found within the same DNA sequence were counted. An ETS1

or RUNX TF was considered to exist in a sequence if it was discovered in at least 20% of the Δ's drawn from all iterations. We counted how many sequences contained oth and ETS1 and RUNX TFs. We looked at both the weaker ETS1 PWM and the TRANSFAC ETS1 PWM. These counts were taken from the ETS1 only data set.

Table 4.2: Number of ETS1 and RUNX in the same DNA sequence

		ETS1	
		Yes	No
RUNX	Yes	0	42
	No	57	1393

Table 4.3: Number of weak ETS1 and RUNX in the same DNA sequence

		Weak ETS1	
		Yes	No
RUNX	Yes	9	32
	No	294	1146

The numbers in the above tables represent the number of sequences with which each combination of ETS1 and RUNX were found. In the ETS1 only data set, no sequences contained both the TRANSFAC ETS1 PWM and the RUNX PWM. Only nine of the 43 RUNX TF binding sites also contained a weak ETS1 TF binding site in the same DNA sequence. A hypothesis test comparing these two tables resulted in no statistically significant difference. This does not support our hypothesis that the RUNX TF will bind more frequently and close by a weak ETS1 TF. In order to further explore this result, future research will look at data sets in which the DNA sequences were drawn from proximal regions and distal regions from the transcription start site. This will allow us to see if ETS1 and RUNX exist in the same sequence when they bind farther away from the transciption start site versus closer to the transcription start site.

The search for *de novo* motifs did not result in any well-defined new motifs, but did reveal some interesting facts about our algorithm. Some of the *de novo* motifs would find the same smaller DNA word. In other words, one *de novo* motif would find a prominent "CTC" in positions two through four while another *de novo* motif would find a prominent "CTC" in positions five through seven. This is why the MEME algorithm probabilistically erases new motifs it finds at each iteration. MEME would find a prominent "CTC" in one iteration and then probabilistically erase the "CTC" motif so as to allow the next iteration to search for a different new motif. Future research will explore this further, as to keep our new motifs from finding the same patterns. Also, many of our *de novo* motifs discovered random background noise. Future research will focus on fixing this problem possibly by adding more different background motifs to search for. Due to these problems, it was difficult to conclude the discovery of a new binding motif in the presence of the ETS1, RUNX, or GABP binding sites.

5. CONCLUSIONS

We created an algorithm that successfully searches for *de novo* motifs using PWM updating. Our method is superior to other methods as we allow for the incorporation of expert prior information, specifically from the TRANSFAC database. Our method also allows the user to simultaneously update known motifs and search for new motifs. Computationally, XPRIME takes close to 72 hours to run on 1,500 sequences run in parallel over a node with a dual quad-core with 10,000 iterations. The variance associated with the posterior distributions of the PWMs gives evidence of the need for an algorithm that draws from the entire posterior distribution, as opposed to simply the posterior means. Thus, we have found that the entire posterior distribution may provide valuable information that the posterior means alone cannot.

Using the XPRIME algorithm to search update two known motifs, ETS1 and RUNX, as well as search for nine new motifs, we have found evidence that the ETS1 TF may have its own specific binding site separate from its family binding site. We have also found that the presence of a weaker ETS1 binding site does not necessarily result in the binding of the RUNX TF. We found that the ETS1 TF will bind more frequently to "GGAA" in the presence of GABP binding sites, suggesting that ETS1 binds frequently to its family motif as well as its own specific motif. Thus, in the presence of GABP binding sites, ETS1 will have a stronger, more well defined motif.

The search for *de novo* motifs did not result in any interesting new motifs, but did result in interesting information about the XPRIME algorithm. Future research will focus on a way to allow the XPRIME algorithm to search for multiple *de novo* motifs. In an effort to prevent *de novo* motifs from discovering background noise, future research will focus on including more backgrond motifs to search for, including a possible background motif generated from an nth-order Markov chain.

BIBLIOGRAPHY

[Bailey and Elkan 1994] Bailey, T. L. and Elkan, C. (1994), "Fitting a mixture model by Expectation Maximization to discover motifs in biopolymers," *UCSD Technical Report*, CS94–351.

[Bailey and Elkan 1995] Bailey, T. L., and Elkan, C. (1995), "The value of prior knowledge in discovering motifs with MEME," *Proc. Int. Conf. Intell. Syst. Mol. Biol.*

[Baldi et al. 1994] Baldi, P., Chauvin, Y., McClure, M., and Hunkapiller, T. (1994), "Hidden Markov Models of Biological Primary Sequence Information," *Proceedings of the National Academy of Science of USA*, 91, 1059–1063.

[Bussemaker et al. 2000] Bussemaker, H. J., Li, H., and Siggia, E. D. (2000) "Regulatory Element Detction using a Probabilistic Segmentation Model," *Proc Int Conf Intell Syst Mol Biol*, 8, 67–74.

[Conlon et al. 2003] Conlon, E.M., Liu, X.S., Lieb, J.D., and Liu, J.S. (2003), "Integrating regulatory motif discovery and genome-wide expression analysis," *Proc. Natl. Acad. Sci. USA*, 100, 3339–3344

[Crooks et al. 2004] Crooks, G.E., Hon, G., Chandonia, J.M., and Brenner, S.E. (2004), "Weblogo: a sequence logo generator," *Genome Research*, 14, 1188–1190.

[Forough et al. 2006] Forough, R., Weylie, B., Collins, C., Parker, J.L., Zhu, J., Barhoumi, R., and Watson, D.K. (2006), "Transcription Factor Ets-1 Regulates Fibroblast Growth Factor-1-Mediated Angiogenesis in vivo: Role of Ets-1 in the Regulation of the P13K/AKT/MMP-1 Pathway," *J Vasc Res*, 43, 327–337.

[Gupta and Liu 2005] Gupta, M. and Liu, J. S. (2005), "De novo cis-regulatory module elicitation for eukaryotic genomes," *PNAS*, 7079–7084.

[Hertz et al. 1990] Hertz, G. Z., Hartzell, G. W. and Stormo, G. D. (1990), "Identification of consensus patterns in unaligned DNA sequences known to be functionally related," *Comput. Appl. Biosci.*, 6, 81–92.

[Hong et al. 2005] Hong, P., Liu, X. S., Zhou, Q., Lu, X., Liu, J. S. and Wong, W. H. (2005), "A boosting approach for motif modeling using ChIP-chip data," *Bioinfromatics*, 21, 2636–2643.

[Hollenhorst et al. 2007] Hollenhorst, P. C., Shah, A. A., Hopkins, C., and Graves, B. J. "Genome-wide analyses reveal properties of redundant and specific promoter occupancy within the ETS gene family," *Genes and Development*, 21, 1882–1894.

[Inoue et al. 2008] Inoue, K., Shiga, T., and Ito, Y. (2008), "Runx transcription factors in nueronal development" *Neural Development*, 3, 20.

[Johnson et al. 2009] Johnson, W. E., Liu, J.S., and Liu, X.S. (2008), "Bioinformatics: Analysis of ChIP-chip data on genome tiling microarrays," *New Developments in Biostatistics and Bioinformatics*, 1.

[Lawerence et al. 1993] Lawerence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., and Wootton, J. C. (1993), "Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment," *Science*, 262, 208–213.

[Liu et al. 1995] Liu, J. S. Neuwald, A. F., and Lawrence, C. E. (1995), "Bayesian models for multiple local sequence alignment and Gibbs Sampling strategies" *JASA*, 90, 1156–1170

[Liu et al. 2001] Liu, X., Britlag, D. L., and Liu, J. S. (2001) "BioProspector: Discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes," *Pacific Symposium on Biocomputing*, 6, 127–138.

[Liu et al. 2002] Liu, X. S., Brutlag, D. L., and Liu, J. S. (2002), "An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments," *Nature Biotech*, 20, 835–839.

[Matys et al. 2003] Matys, V., Fricke, E., Geffers, R., Gossling, E., Haubrock, M., Hehl, R., Hornischer, K., Karas, D., Kel, A.E., Kel-Margoulis, O.V., Kloos, D.U., Land, S., Lewicki-Potapov, B., Michael, H., Munch, R., Reuter, I., Rotert, S., Saxel, H., Scheer, M., Thiel, S., and Wingender, E. (2003), "Transfac: transcriptional regulation, from patterns to profiles," *Nucleic Acids Res*, 31, 374–378.

[Roth et al. 1998] Roth, F. P., Hughes, J. D., Estep, P. W., and Church, G. M. (1998), "Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation," *Nature Biotech*, 16, 939–945.

[Sandelin et al. 2004] Sandelin, A., Alkema, W., Engstrom, P., Wasserman, W.W., and Lenhard, B. (2004), "Jaspar: an open-access database for eukaryotic transcription factor binding profiles," *Nucleic Acids Res*, 32, D91–D94.

[Shim and Keles 2007] Shim, H. and Keles, S. (2007), "Integrating quantitative information from chip-chip eperiments into motif finding" *Biostatistics*, 9, 51–65.

[Sinha and Tompa 2006] Sinha, S. and Tompa, M. (2006), "A statistical method for finding transcription factor binding sites," *Proc. Int. Cont. Intell. Syst. Mol. Biol.*, 8, 344–354.

[Torlakovic et al. 2004] Torlakovic, E.E., Bilalovic, N., Nesland, J.M., Torlakovic, G., Florenes, V.A. (2004), "Ets-1 transcriptionf factor is widely expressed in benign and malignant melonocytes and its expression has no significant association with prognosis," *Modern Pathology*, 17, 1400–1406.

[Wasserman et al. 2000] Wasserman, W.W., Palumbo, M., Thompson, W., Fickett,

J.W., and Lawrence, C.E. (2000), Human-mouse genome comparisons to locate regulatory sites, *Nat. Genet.*, 26, 225–228.

[Zhou and Wong 2004] Zhou, Q. and Wong, W. H. (2004) "CisModule: De novo discovery of cis-regulatory modules by hierarchical mixture modeling," *PNAS*, 101, 12114–12119.

A. APPENDIX

A.1 RUNX TF Posterior Mean PWMs

RUNX Posterior mean from the ETS1 only data set:

Position:	1	2	3	4	5	6	7
A	0.000	0.00	0.000	0.00	0.00	0.000	0.079
C	0.062	0.00	0.167	0.00	0.00	0.079	0.298
G	0.000	1.00	0.000	1.00	1.00	0.053	0.116
T	0.938	0.00	0.833	0.00	0.00	0.868	0.507

RUNX Posterior mean from the GABP only data set:

Position:	1	2	3	4	5	6	7
A	0.000	0.00	0.000	0.00	0.00	0.000	0.061
C	0.045	0.00	0.147	0.00	0.00	0.061	0.289
G	0.000	1.00	0.000	1.00	1.00	0.053	0.132
T	0.955	0.00	0.853	0.00	0.00	0.886	0.518

RUNX Posterior mean from the GABP/ETS1 data set:

Position:	1	2	3	4	5	6	7
A	0.000	0.00	0.000	0.00	0.00	0.000	0.053
C	0.104	0.00	0.057	0.00	0.00	0.146	0.321
G	0.000	1.00	0.000	1.00	1.00	0.166	0.173
T	0.896	0.00	0.943	0.00	0.00	0.688	0.453

Figure A.1: RUNX Sequence logo from TRANSFAC

Figure A.2: RUNX Sequence logo from ETS1 only data

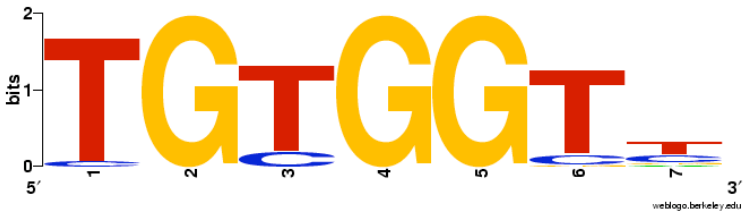


Figure A.3: RUNX Sequence logo from GABP only data

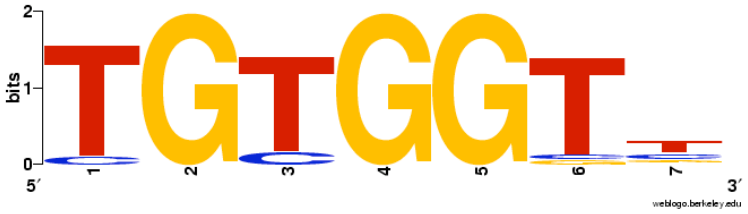
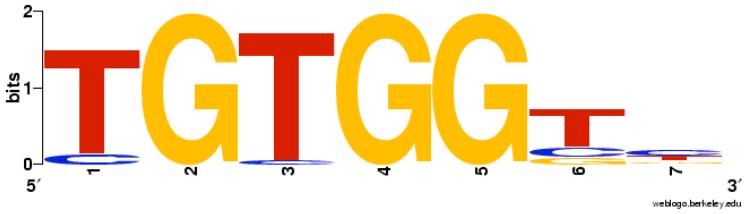


Figure A.4: RUNX Sequence logo from GABP/ETS1 data



A.2 The Algorithm

```
library(snow) #Package for parallel processing
library(Rlab)
library(stats)
library(MCMCpack)
c1=makeCluster(8) # # of processors

#########################
###Needed functions######
#########################

###1. READING IN THE FASTA FILE#####
readfile<-function(dna)
{
n<-length(dna)
sequence<-NULL
title<-NULL
for(i in 1:n)
{
if(strsplit(dna[i],'hg')[[1]][1]=='>')
{
sequence<-c(sequence,'')
title<-c(title,dna[i])
}
else{
     sequence[length(sequence)]=
     paste(sequence[length(sequence)],
     dna[i],sep='')
    }
    }
seq=unlist(parLapply(c1,sequence,strsplit,NULL),recursive=FALSE)
return(seq)
```

```
}

dna<-readLines("ets1chipseq.seq",n=-1) #Read File
#Read File
seq<-readfile(dna)

###4. MAKING THE SEQUENCE A MATRIX#####
seqMat=function(DNAseq)
{
tmp=list("A"=c(1,0,0,0),"C"=c(0,1,0,0),
         "G"=c(0,0,1,0),"T"=c(0,0,0,1),
         "N"=c(1/4,1/4,1/4,1/4))
    matrix(unlist(tmp[(DNAseq)]),nrow=4)
}


###5. FIXING RANDOM DIRICHLET AND RANDOM MULT. FUNCTIONS###
rdir<-function(alpha,n){rdirichlet(n,alpha)}
rmult<-function(p,n,s){rmultinom(n,s,p)}

###6. SCORING FUNCTION######
scoring=function(seq, PWM)
{
  scores=NULL
  for (i in 1:(ncol(seq)-ncol(PWM)))
  {
     scores=c(scores, prod(diag(t(PWM)%*%seq[,i:(i+ncol(PWM))])))
  }
  scores
 }

#########################
####INITIAL VALUES#########
```

```
############################


###1. Initial known PWMs of interest###
etsPWM<-matrix(c(
1,5,0,0,15,8,4,1,
14,9,0,0,0,2,1,6,
0,0,15,15,0,0,10,0,
0,1,0,0,0,5,0,8), nrow=4,byrow=T)

runxPWM<-matrix(c(
0,0,0,0,0,0,1,
1,0,3,0,0,1,4,
0,17,0,17,17,1,2,
16,0,14,0,0,15,10),nrow=4,byrow=T)

#Reverse compliment PWMs
nr<-4
ncets<-ncol(etsPWM)
ncrunx<-ncol(runxPWM)
ets_rcPWM<-etsPWM[nr:1,ncets:1]
runx_rcPWM<-runxPWM[nr:1,ncrunx:1]

PWMs<-list(etsPWM,runxPWM,ets_rcPWM,runx_rcPWM)
#Known PWMs of interest must be in list

###2. File to use and it's initial simMats####
### Deleting sequences that are too short###
simMat=parLapply(c1,seq,seqMat) #Original sequence
short=which(unlist(lapply(seq, length))<=ncol(etsPWM))
if(length(short)>0)
{
   cat("Deleted", length(short), "sequences because they are too short.\n")
```

```
   simMat=simMat[-short]
}


###3. Generating initial background and scores####
w<-max(as.numeric(lapply(PWMs,ncol))) #Allowing background to
have the width of the longest PWM of interest.
pA<-sum(unlist(seq)=="A")/length(unlist(seq))
pC<-sum(unlist(seq)=="C")/length(unlist(seq))
pG<-sum(unlist(seq)=="G")/length(unlist(seq))
pT<-sum(unlist(seq)=="T")/length(unlist(seq))
backPWM<-t(rdir(c(pA,pC,pG,pT),w))

scoresBACK<-parLapply(c1,simMat,scoring,backPWM)
#scoresBACK_crev<-parLapply(c1,simMat_crev,scoring,backPWM)

###4. Making PWMs the same length--tacking on extra background##
for(i in 1:length(PWMs))
{
extra_back<-t(rdir(c(pA,pC,pG,pT),1))
while(ncol(PWMs[[i]])<w)
{
      PWMs[[i]]<-cbind(PWMs[[i]],
                       (extra_back*sum(PWMs[[i]][,1])))
}
}

###5. Putting PWMs into probabilities for scoring function###
p.PWMs<-list()
for(i in 1:length(PWMs))
{
p.PWMs[[i]]<-PWMs[[i]]/sum(PWMs[[i]][,1])
}
```

```
###6. Generating Priors for de novo motifs###
nnew=2
len=matrix(w,nrow=nnew,ncol=1)
#if(nnew>0)
#{
new_prior<-list()
for(i in 1:nnew)
{
nprior=NULL
for(j in 1:len[i])
{
nprior=cbind(nprior,c(1,1,1,1))
}
new_prior[[i]]=nprior
}
#}
new_prior_rc<-list()
for(i in 1:length(new_prior))
{
new_prior_rc[[i]]<-new_prior[[i]][4:1,w:1]
}
new_prior<-c(new_prior,new_prior_rc)

p.PWMsnew<-list()
for(i in 1:length(new_prior))
{
p.PWMsnew[[i]]<-new_prior[[i]]/sum(new_prior[[i]][,1])
}

###MCMC Settings####
M=10000
```

```
burn<-500

#r_alpha
r_alpha=matrix(1,nrow=length(new_prior)+length(PWMs)+1,ncol=1)
#Assuming one occurrence per motif

r=matrix(1/(length(new_prior)+length(PWMs)+1),
nrow=length(new_prior)+length(PWMs)+1,ncol=1)
#Assuming Equal probabilities

N<-sum(length(unlist(seq)))-(length(seq)*w-1)
nseq<-as.matrix(unlist(lapply(seq,length)))
if(length(short)>0){
nseq<-nseq[-short]
}
den.r<-length(unlist(seq))+sum(r_alpha)

known_PWM_final<-array(0,c(4,w,M+burn,length(PWMs)))
new_PWM_final<-array(0,c(4,w,M+burn,nnew))

tmpKnow=array(0,c(4,w,M+burn,length(PWMs)))
for(i in 1:length(PWMs))
{
tmpKnow[,,1,i]<-PWMs[[i]]
}
tmpNew=array(0,c(4,w,M+burn,length(new_prior)))
for(i in 1:length(new_prior))
{
tmpNew[,,1,i]<-new_prior[[i]]
}

mdelta<-NULL
mr<-NULL
```

```
for(i in 2:(M+burn))
{
#First calculate scores for motifs of interest
Kscores<-list()
for(j in 1:length(PWMs))
{
Kscores[[j]]<-parLapply(c1,simMat,scoring,p.PWMs[[j]])
}
#Next calculate scores for new motifs
    Nscores<-list()
    for(j in 1:length(new_prior))
    {
Nscores[[j]]<-parLapply(c1,simMat,scoring,p.PWMsnew[[j]])
}
Allscores<-c(Kscores,Nscores)
p<-NULL
for(j in 1:length(Allscores))
{
p<-cbind(p,r[j]*unlist(Allscores[[j]]))
#Probability matrix for the random multinomial
    }
    p<-cbind(p,r[length(Allscores)+1]*unlist(scoresBACK))
    z<-t(apply(p,1,rmult,1,1)) #probs don't need to be normalized
    rprobs<-NULL
    for(j in 1:length(r))
    {
     rprobs<-cbind(rprobs,(sum(z[,j])+r_alpha[j]))
    }
    r=rdirichlet(1,rprobs)

    P<-list()
    for(j in 1:(length(PWMs)+length(new_prior)))
    {
```

```
 P[[j]]=matrix(0,4,w)
}
index=0
for(m in 1:length(simMat))
{
 delta<-z[(1:(nseq[m]-w))+index,]
index<-nseq[m]-w+index
   for(k in 1:(ncol(delta)-1))
   {
    for(j in 1:w)
    {
    P[[k]][,j]<-apply(matrix(delta[,k],nrow=4,
                    ncol=length(delta[,k]),byrow=TRUE)*simMat[[m]]
                    [,(1:nrow(delta))+w-1],1,sum)
 }
 }
}

Kalphas<-NULL
for(j in 1:length(PWMs))
{
Kalphas[[j]]<-P[[j]]+PWMs[[j]]
}
Nalphas<-NULL
for(j in 1:length(new_prior))
{
Nalphas[[j]]<-P[[j+length(PWMs)]]+new_prior[[j]]
}
Kalphas2<-list()
for(j in 1:(length(Kalphas)/2))
{
 Kalphas2[[j]]<-Kalphas[[j]]+Kalphas[[j+(length(Kalphas)/2)]][4:1,w:1]
}
```

```
    Nalphas2<-list()
    for(j in 1:(length(Nalphas)/2))
    {
     Nalphas2[[j]]<-Nalphas[[j]]+Nalphas[[j+(length(Nalphas)/2)]][4:1,w:1]
    }

for(j in 1:(length(PWMs)/2))
{
p.PWMs[[j]]<-apply(Kalphas2[[j]],2,rdir,1)
                #add ets_alpha+ets_rc_alphas[4:1,L:1]
}
for(k in 1:nnew)
{
p.PWMsnew[[k]]<-apply(Nalphas2[[k]],2,rdir,1)
        }

if(i==burn)
{
mdelta<-delta
}
if(i>burn)
{
mdelta<-mdelta+delta
mr<-cbind(mr,t(r))
}

for(j in 1:length(PWMs))
{
tmpKnow[,,i,j]<-p.PWMs[[j]]
}

for(k in 1:nnew)
{
```

```
tmpNew[,,i,k]<-p.PWMsnew[[k]]
}
if(i%%10==0){cat(i,"\n")}
if(i%%100==0){save.image("ets_long.RData")}
}
#return(list(tmpNew,tmpKnow,mdelta))
#}
#results<-Xprime("ets1_regions.seq",nnew=2,PWMs=PWMs,niter=1000)
#save.image("firsttry.RData")

#########################
####Achieving Results####
#########################

load("/Users/Rachel/ets_only_evan.RData")

##Trace plots###
par(mfrow=c(2,4))
for(i in 3:4){
for(j in 5:7){
plot(tmpKnow[i,j,1:9600,2],type="l",ylab="",xlab="N*",ylim=c(0,1))
}
}

plot(mr[1,],type="l",main="ETS r",xlab="N*",ylab="")
plot(mr[3,],type="l",main="RUNX r",xlba="N*",ylab="")
plot(mr[5,],type="l",main="NEW1 r",xlba="N*",ylab="")
plot(mr[7,],type="l",main="NEW2 r",xlba="N*",ylab="")

#Posterior Densities#
par(mfrow=c(2,4))
for(i in 3:4){
for(j in 5:8){
```

```
plot(density(tmpKnow[i,j,(500:9600),1],bw=0.01),type="l",ylab="",
xlim=c(0,1),xlab="",main="")
}
}

par(mfrow=c(2,4))
for(i in 1:2){
for(j in 1:4){
plot(tmpNew[i,j,1:9600,10],type="l",ylab="",xlab="N*",ylim=c(0,1))
}
}

#Some examples
plot(tmpNew[2,2,1:9600,1],type="l",ylab="",xlab="N*",ylim=c(0,1),main="p22")
plot(tmpNew[3,3,1:9600,1],type="l",ylab="",xlab="N*",ylim=c(0,1), main="p33")


#Posterior means and variances
ets_mean<-matrix(0,nrow=4,ncol=8,byrow=TRUE)
for(i in 1:4){
for(j in 1:8){
ets_mean[i,j]<-mean(tmpKnow[i,j,(500:9600),1])
}
}

ets_var<-matrix(0,nrow=4,ncol=8,byrow=TRUE)
for(i in 1:4){
for(j in 1:8){
ets_var[i,j]<-var(tmpKnow[i,j,(burn:9600),1])
}
}

runx_mean<-matrix(0,nrow=4,ncol=7,byrow=TRUE)
```

```
for(i in 1:4){
for(j in 1:7){
runx_mean[i,j]<-mean(tmpKnow[i,j,(500:9600),2])
}
}

new1_mean<-matrix(0,nrow=4,ncol=8,byrow=TRUE)
for(i in 1:4){
for(j in 1:8){
new1_mean[i,j]<-mean(tmpNew[i,j,(500:9600),1])
}
}

#Obtaining samples to create sequence logos
new_sample=function(pwm_row,ncols=1){sample(c("A","C","G","T"),
1*ncols,
prob=pwm_row, replace=TRUE)}
for (i in 1:100){
cat(apply(new1_mean,2,new_sample),'\n',sep='')
}
```