Notes on Motif Finding via Gradient Decent, EM, and Gibb's Sampling

Julian E. Yarkony

***Index Terms*— Gibbs sampling, EM, motifs**

***Abstract*— This is an explanation of motif finding for intro computational biology students.**

I. Introduction

WHAT IS A SEQUENCE MOTIF? Motifs are specific patterns found in the DNA. Such patterns could be a string of 5 As followed by a C or could by an A or a T followed by 7 Gs. Patterns can be more elaborate and have all kinds of unique relationships within themselves. A motif should not be thought of as having a specific length or exact pattern (though a particular motif could have one or both of these) however it should be though of as describing a set of related sequences. In class we used motifs which can be described by a position weight matrix which is described below.

A motif of length m is represented using *Position weight matrix*

$$\theta = \begin{bmatrix} \theta_{11} & \theta_{21} & ... & \theta_{w1} \\ \theta_{12} & \theta_{22} & ... & \theta_{w2} \\ \theta_{13} & \theta_{23} & ... & \theta_{w3} \\ \theta_{14} & \theta_{24} & ... & \theta_{w4} \end{bmatrix}$$

In the above diagram each column represents the a particular spot in the sequence, each row an amino acid. so the value in row i column j is the probability amino acid i will occur in the jth spot in the sequence. Note dependencies exist between elements of the sequence once the motif's parameters are known (conditionally independent given motif parameters).

II. Purpose

what purpose does finding them serve? Motifs can be though of as common words in the language of the code of life (DNA). DNA is not a random sequence nor is the conservation of features over time in the DNA a random phenomenon. Motifs have been conserved over evolution for a reason; for they have biological significance.

Motifs are found all over the DNA; they exist in exons (the sections of DNA which describe a protein/little machine that exists inside the cell); they exist in introns which are sections of DNA which do not describe proteins but describe other stuff like protein regulation; some segments associated with a particular motif provide spots for regulatory proteins to bind to the DNA allowing them to regulate protein production.

Discovering motifs is a difficult question and a seemingly subjective question however modern artificial intelligence and statistical methods give great insight into determining what is a motif and what is not.

III. Point of note

The clustering of sequences into groups which can be represented as a motif may seem to be a rather arbitrary process. Who is to say in that case which features are so important as they must be conserved in every sequence or where there are some distinct groups inside a set weather they need their own set. These are all good questions however they do not save biologists from dealing with motifs simply because of the fuzziness of their boundaries and definition. Without the ability to generalize intelligence would be a rather shallow shell. Labels are used all over our culture and society to great effect in regards to making decisions. America for example classifies its politicians into liberals moderates and conservatives. While these definitions are not perfect (more categories can always be created liberal/moderate, conservative/ moderate and then conservative/conservative/moderate) depending on the precision needed generalizations are made for a given situation. These generalizations allow for the compaction of data and enables the citizens to make decisions. Memory, training data and computation time are limited and hence generalization and the reduction of complex data into a small number of features is essential to decision making regardless of whether humans of machines make the decision. However when making a generalization it is essential to make sure that the representation is a valid one (principle component analysis is often helpful but not always sufficient) . Experimentation, domain knowledge and mathematical theory inform that decision.

IV. General models for a motif, and some cooler ones

As stated above, the motifs that we saw in class are describe using position weight matrixes which permit descriptions of a motif without regard to interdependencies. This is a simple yet powerful model but in instances where dependence is crucial it can prove insufficient. However there exist a class of graphical models called hidden markov models. These allow for the varying of the length of a sequence generated by the motif and for dependencies between inputs at position x and position x+1 (more elaborate models allow for even more powerful and high level dependencies). Depending on the needs of the experimenter different models and model parameters are used.

V. Clustering

The fundamental idea behind the motif detection methods is to take the set of sequences and split their subsequences into background and one of a certain number of motifs k

where k is the number of motifs present. The problems that are run into are the following:
1. how many clusters or motifs should be defined
2. how should different subsets be grouped into which motif (distance metric)
3. what about points not really in either cluster
4. In the context of motifs how long and how variable should cluster lengths be.

The problems above are of significant mathematical difficulty and have many approaches to dealing with them. The study of the over fitting, the bias variance trade off, and the k-means algorithm provide a great deal of insight into this problem however due to the length of time required to go over that material I will simply refer you to the bishop book.

VI. Finding Motifs via, gradient decent:

Gradient decent is arguably the most important method in machine learning and artificial intelligence. It dominates neural networks, optimization theory, and a variety of other crucial theories, methods, and applications in machine learning and AI. However in motif finding, gradient decent is a poor match given teh nature of the problem and is not used in finding motifs in the real world. Gradient decent is a process used to determine global and local minimum (or maximum if desired) a function that has one of the following properties
1. Is not of closed form that is to say one can send in an input x and get out an output y but one can not write the function like y=mx+b
2. Has no closed form solution and has a lot of data points associated with it.
Gradient decent can be likened to a blind man walking down a hill. He starts in a given position x and checks all the spots around him then moves to the lowest of the points. He repeats this until he can go down no more. The problem is local minima. How does he know he has reached the true bottom of the hill or just fallen into a small pit? Adding noise (sometimes going in the wrong direction) and random restarts (putting the blind man at a different part of the hill in hopes that in the new position he will find a deeper bottom) help this process find the true bottom called the global minima. However this process is not guaranteed (save for infinite time) to find the true global minima.

The reason you want the blind man to go down the hill because the lower he goes the better position he is in (the better parameters for your model). The reason all positions cannot be probed is because there are too many of them costing too much computing power, and memory and often requires too much training data. In the context of motifs we can imagine that we are trying to maximize the likelihood of the observed data being generated by our model by modifying the parameters of our model. That is to say make the motifs that we have look like they came from the data. Gradient Decent can be thought of as a method of directed search.

Example: Lets suppose we have a set of n sequences each of length k. We know (determining the number of motifs is non-trivial but just flow with me) that there is a background model with each space independent and equally likely any amino acid and there is a SINGLE motif of length k with its members independent but each being weighted towards different amino acids.
1. Create a guess distribution for each position.
2. Determine the likelihood of your model

$$logL(\theta,\theta^0) = \sum_{i=1}^{n} log[\alpha P(S_i|\theta) + (1-\alpha)P(S_i|\theta^0)]$$

Adding logs is like multiplying the contents of those logs. see my note on logs towards the bottom. So the above statement says multiply the probabilities of each sequence being generated where the probability of a sequence being generated is: the probability the sequence was generated by the motif plus the probability it was generated by the background both multiplied by their respective probabilities .

3. If youre getting stuck at the same likelihood quit or do a random restart of your motifs parameters.
4. Increase or decrease each position in the direction of the gradient. That is to say adjust all the parameters a very small bit but choosing the direction which best improves the likelihood of the data.
θ_t: $\frac{\partial logL(\theta,\theta^0)}{\partial\theta_{ij}}|_{\theta_t}$
the last statement encodes this idea mathmatically. If the gradient is positive (updating parameters with respect to the gradient will improve the liklihood of the model) then update your model using the following formula. $\theta_{ij}^{t+1} = \theta_{ij}^{t} + \eta[\frac{\partial logL(\theta,\theta^0)}{\partial\theta_{ij}}|_{\theta_t}]$ *the important thing here is the parameter η which limits the amount of movement of the parameters. The reason the gradient is not followed for long distances is the direction of fastest decent changes. This means that the direction which takes you down the hill fastest is different on different parts of the hill and hence needs to be updated frequently. Smaller η values mean smaller step sizes*

5. Add noise. Mess up your model a little bit this can help get through local minima. This means simply adjust the parameters of your model randomly. Don't change them too much, the degree of noising depends on the data set.
6. go back to step 2.

VII. Finding Motifs via expectation maximization or the EM algorithm

One can imagine EM as an election. Everybody picks the positions that they want to take. They vote to alter the positions of both parties to a degree depending on how much they share the current platform . Then the people who are closest to the winning positions get more votes during the next election and so on. That is the core of EM EM consists of 2 steps the E step and the M step. The e step determines the likelihood that a given sequence is a member of each motif (or the background). The M step assumes the membership in each motif group and than maximizes the motifs parameters to reflect that group.

To do the E-step for a given sequence
Determine the probability that sequence Si was generated by all the motifs. Then normalize this by dividing by the sum of the probabilities. We divide because we know that the sequence was generated by one motif. Do this for all sequences. To determine the probability of a motif generating a sequence use the following formula

$$P(S_1)|\theta) = \prod_{i=1}^{L} \theta_{i,S_i}$$

this is to say multiply the likelihoods of motif m generating each each position. This assumes independence of positions (in math terms conditional independence of each position given the motif or background parameters associated with a given sequence).
Another thing to note is that the probiblity of the motif generating the sequence must be taken into account. The way to think about this is suppose that a sequence S has a 1 percent liklihood of being all A's when being generated by the background and 100 percent of being all A's when being generated by the motif. However suppose the motif is very rare and generates only .0000001 percent of the sequences. In this case it is much more likely that is S (which is all A's) was generated by the background than by the motif. Below is the mathematical description

On this step algorithm calculates posterior distribution $q(z_i)$ which is the probability a given sequence was generated by each model.

$$q(z_i)^{t+1} \sim \begin{cases} P(z_i = 1)P(S_i|\theta^t) & \text{if } z_i = 1 \\ P(z_i = 0)P(S_i|\theta_0^t) & \text{if } z_i = 0 \end{cases}$$

z refers to whether the sequence was generated by the motif or not (1 being generated by the motif.
S refers to a reference sequence
theta 1 refers to the parameters of the motif and theta 0 the parameters of the background

To do the M step for a given motif group Assume we are just working with the simple motifs from class. The probability that a given position k produces an amino acid l is set to the weighted likelihood of the data. This is like taking an average but with some sequences (the sequences which were more likely generated by the motif group being considered getting more votes). This can be written mathematically as follows. On this step we calculate new θs:

$$\theta_{ml}^{t+1} \sim \sum_{i=1}^{n} q^{t+1}(z_i) I(S_{im} = l)$$

Here the I term is a boolean(1 if true zero if false) for whether a particular amino acid in a particular position was generated and the q term is the likelihood that the sequence is part of the motif. Do this step for all motif groups though if the background is known no need to do it for the background.

Stop when the parameters of the motif stop changing.
Just a note: genetic algorithms (a really powerful AI technique based on biological evolution) have been shown to help EM get out of local minima. Also EM is used everywhere in AI and if you look through chapter 21 in Russel and Norvig (the green book) really carefully you too can make your own EM based algorithms. For more cool applications of EM look to Pierre Baldi's book Bioinformatics particularly chapter 7. It is really dense but really good. There is lots of math in it but if you crank through it trust me it is worth it.

VIII. Finding Motifs via Gibbs Sampling

Gibbs sampling is a very similar procedure. If you are familiar with Markov Chain Monte Carlo (not the place to gamble) then this should look familiar as the to procedures are very similar. In this Gibbs sampling algorithm sequences are selected one at a time and reassigned randomly (they are reassigned according to a probability distribution. The idea is for the samples to gradually converge to the proper answer, noise is added (via not always choosing the max likelihood estimate) to counter local minima.
Example: assume one motif and one known background model. 10 percent of sequences are members of the motif and 90 percent are background.
Step1: randomly assign sequences according to the prior probability. That is to say assign each sequence to 0 with 90 percent probability and to 1 with 10 percent probability.
Step 2: Select a sequence at random and call it Si
Step 3: determine the parameters of the motif. For each position determine the likelihood that the members take on a certain value.

To do this calculate the likelihoods that the sequences currently labeled as motif (excluding the sequence being relabeled if it is set as a member of the motif) would generate each amino acid in each position. Let n_{ij} be the number of letter j at position i. Set $\theta_{ij} = \frac{n_{ij}+\gamma}{n+4\gamma}$,

here γ is some small parameter which is called pseudo-count which prevents probabilities from being zero a condition which would make the algorithm less able to reach good answers. n - the total number of sequences with label 1, excluding S_i. We do the same estimations for θ^0

Step 4 reassign: Determine the probability Si was generated by each. THen based on these probabilities randomly reassign the sequence. This can be written mathematically as follows. Here q is the label (the t+1 indicates the new label). The probability of z term indicates likelihood of the motif (1 being motif 0 being background)

$$q(z_i)^{t+1} \sim \begin{cases} P(z_i=1)P(S_i|\theta) & \text{if } z_i = 1 \\ P(z_i=0)P(S_i|\theta_0) & \text{if } z_i = 0 \end{cases}$$

Step 5: select another sequence go to step 3 unless you have done this for a really long time, see little change in the assignment or have a better stopping criteria satisfied

IX. Pluses and minuses of the methods

Gradient Decent: Totally unsuited for this problem and is not used. Local minima prevent the motif model from reaching accurate values. A parameter called the step size must be chosen and this is a difficult parameter to determine and must be calculated for each new application and often each data set. This algorithm is also slow to converge on motif answers. In the context of motifs the gradient is also hard to calculate. This means that approximate gradients may have to be used further inhibiting convergence to a good answer.

EM: EM unlike gradient decent is fast, and guaranteed to reach a final answer and not jitter around an answer. While it is a similar to a gradient decent algorithm at a deep mathematical level it requires no step size which is a huge plus. However two problems come up (both of which can be diminished though not eliminated via the use of simulated annealing and or genetic algorithms). These two problems are local minima and sensitivity to initial parameters. Genetic algorithms and simulated annealing additions to EM help resolve these problems these by having many sets of initial parameters via random restarts and get out of local minima via noise and crossover.

Gibbs Sampling
This method is less susceptible to local minima (unlike EM which is effected by this). It is also good at incorporating information known before the clustering process has begun. Like EM it is guaranteed to reach a definite answer. However the algorithm can be much slower than EM and unlike EM it is difficult to know when to stop training the model.

X. Dealing with multiple motifs (beyond the two dimensional case)

The methods described here can be scaled to high numbers of motifs. In real world research prior determination of the number of motifs is essential. There is one algorithm, which determines the number of motifs but requires a size of the motifs. The obvious thing to do is then to use (abuse) this algorithm to determine the number of motif of each size; unfortunately this does not work, as you would have sub-motifs inside of motifs. Hence determining the number of motifs is non-trivial.

XI. Just a note all that stuff about logs

This is just a computer thing that makes stuff easier to multiply in a computer. When multiplying lots of really small numbers together on computers data is lost. This is not because computers forget but because computers do not have infinite precision. To get around that problem we use logs. How is this done?
Example: P(x)P(y)P(z)=e to the power of (log(P(x))+log(P(y))+log(P(z))) and look we added logs and hence got around the multiplication problem. Also it helps when solving differential equation and doing optimizations because it makes it easier to take derivatives.

Cheers Julian Yarkony jyarkony@uci.edu