1 Multiple sequence alignment

While the need for comparing 2 sequences is clear, there are circumstances in which one wants to compare more than one sequence in order to construct the alignment. For example, one could have a set of proteins thought to be related from a number of species and one would like to see what aspects of these sequences are conserved across all species. Knowledge of which domains of a protein are conserved across varying species is informative about the function of the proteins. In addition, for various purposes it is desirable to classify proteins into one of a small number of groups. Multiple sequence alignment can help us decide if a set of proteins should be considered as elements of a group of proteins. As when aligning just 2 sequences, we can discuss global and local alignment methods. We concentrate on the global alignment problem here since the local methods are just refinements of the basic idea.

1.1 Dynamic programming

One can extend the previous methods for aligning a pair of sequences using dynamic programming to allow one to align more than one sequence at a time, but the computational burden increases to the extent that this approach is not useful. To understand the nature of the problem, suppose we are trying to align 3 sequences. For aligning 2 sequences, one constructed a 2 dimensional array (i.e. a matrix) and filled in this array with the scores determined in a fashion discussed earlier. When we compare 3 sequences, we must fill in a 3 dimensional array. Not only are there larger memory requirements, but the number of paths that lead to each internal element of the table goes from 3 (or 4 for local alignment) to 7 (3 ways to have a gap in 2 of the sequences, 3 ways to have a gap in 1 of the sequences and 1 way to have no gaps). This increases the computational burden to the extent that dynamic programming is no longer a practical method for finding the optimal alignment. Clearly one can consider local and global multiple alignment as in pairwise alignment.

There have been many attempts to speed up dynamic programming methods in the context of multiple alignment. The basic idea is that we shouldn't really calculate all entries in the dynamic programming array. The decision regarding which entries should be evaluated is what distinguishes these various methods. For example, one can look at all of the pairwise alignments. These give upper bounds on how well any 2 sequences can be aligned in a multiple alignment. One can use fast heuristic methods (see below) to get lower bounds on the multiple alignment, and these bounds reduce the number of entries in the dynamic programming array that must be evaluated.

1.2 Hidden Markov models

One can also use Hidden Markov models to find a multiple alignment. To understand the basic ideas, consider the problem of aligning just 2 sequences. The hidden Markov chain has 5 types of states: align the letters from the 2 sequences, align a letter from sequence 1 to a gap, align a letter from sequence 2 to a gap, start the alignment and end the alignment. These start and stop states are necessary since the length of the aligned sequences is not known prior to conducting the alignment. (A Markov chain that stops at a random time which is known at the time the chain stops is still a Markov chain.) For such a model it makes sense to have the transition probabilities be symmetric in time, i.e. the transition probability from the state that aligns residues to the state that introduces a gap in either sequence should be equal to the probability of a transition from a gap state to the state that aligns the 2 sequences. It is also sensible to suppose the probability of going from the aligned state to either gap state is the same for both of the gap states. To allow for differing costs of opening and extending a gap, we suppose that the transition from a match state

to a gap state differs from the probability of remaining in a gap state. If the cost of opening and extending a gap is less than the cost of the worst mismatch then a one will not have a deletion adjacent to an insertion, hence one needn't be concerned about transitions between these 2 states (this assumption is made in Durbin et al.). Our observation process consists of the letters of the 2 sequences at a given location (these letters can be indels). In the case of multiple sequences, the observation process consists of the letters from all of the sequences at a given location. Hence the general methods for hidden Markov models can be used for parameter estimation and determining the optimal multiple alignment.

An alternative, yet equivalent, way to consider these hidden Markov models is to suppose there is some model sequence. This sequence is the unknown common sequence that underlies all the observed sequences. The hidden process is the path through the dynamic programming matrix for aligning each observed sequence to the model sequence. This process can be considered as a 2 dimensional Markov process. One dimension of this process is if there is an insertion or a match in the observed sequence (a binary variable) and the other dimension is the number of deletions up to the current location. It is the second component of this 2 dimensional process that possess the Markov property. Typically it is assumed that if the hidden process is in a match state, then the distribution for the observed sequence is just a point mass (i.e. a Dirac delta function) at the letter of the model state. (Where this point mass is constitutes a model parameter that must be estimated.) If the hidden process is in the insertion state then the distribution for the observed sequence is typically modeled as an observation from a single "background" distribution. (This background distribution is parameterized by a set of model parameters that must be estimated.) The extreme flexibility of hidden Markov models for multiple alignment comes from the fact that one can specify more complex models for the background distribution (e.g. the background distribution depends on location) and the distribution of the observed sequence given the hidden process is in the match state (e.g. one can allow a general probability distribution for the observations that differs depending on location in the model state).

1.3 Progressive alignment methods

There are a number of more heuristic methods for multiple alignment. Progressive alignment methods are based on the natural idea of examining all of the pairwise alignments and using these to construct a multiple alignment. There are many variations on this basic idea. What distinguishes most of these methods from the others is the manner in which one makes the choice regarding what pairwise alignments are most important for constructing the multiple alignment. The popular program `CLUSTALW` uses a progressive alignment method to construct multiple alignment. Although there are many details to this heuristic algorithm, the idea is as follows. One first calculates all pairwise alignment scores. Then one uses a cluster algorithm to determine what sequences are closest using the alignment scores. Then one aligns the 2 closest sequences to get the pairwise alignment between them. This alignment is then modified by aligning the next closest sequence to the 2 used thus far to the alignment of the original 2 sequences. One then continues by aligning the next most closely related sequence to the alignment one has obtained thus far.

1.4 Block motif methods

Another approach to multiple alignment is based on the idea that sequence similarity across species is attributable to similar function. In order for biopolymers to have similar functions, they need to have similar ligand binding regions. Hence we should look for these binding regions and align the sequences with reference to these regions, or *motifs*. These methods are inherently methods of local alignment since they search for small conserved regions in the sequences. Recently there has been a unification of these block motif methods with the hidden Markov model methods.

We will consider some of the proposed approaches in detail since much statistical research has been done in this area. Suppose there are K sequences, $R_1, \ldots, R_K$ over an alphabet of size p. Denote the collection of these sequences as R. Suppose there is a conserved sequence of length J. This sequence is called a *motif element* or more simply an element. A *motif* is the probability distribution for the elements. Originally, these methods have assumed that biopolymers are sequences of iid draws from some probability distribution, but there have been recent extensions that depart from this strong assumption. There are 2 such distributions when we seek a single motif: a background distribution describing the distribution over the letters in non-motif regions and a probability distribution over the motif region. For the motif region, we assume the distribution depends on location in the motif. Hence we can think of a motif as a set of probability distributions over the alphabet $\theta_j = (\theta_{1j}, \ldots, \theta_{pj})$ for $j = 1, \ldots, J$. We use the notation Θ to represent all of these parameters. In addition there is a background distribution $\theta_0 = (\theta_{10}, \ldots, \theta_{p0})$ In this context, an alignment is simply the location of the start of the motif in each sequence $A = \{(k, a_k) : k = 1, \ldots, K\}$, where a_k is the location of the start of the motif in the k^{th} sequence.

If we treat all of the parameters and the alignment, A, as random variables, then we can use methods of Bayesian statistics to find the posterior distribution of the alignment. Now $p(A|R) \propto p(A, R)$ thus

$$p(A|R) \propto \int p(A, R|\theta_0, \Theta) p(\theta_0, \Theta)\, d\theta_0 d\Theta.$$

If we use independent, conjugate prior distributions for the parameters θ_0 and Θ then we can perform the desired integration. Our likelihood is a product of multinomial distributions, hence the conjugate priors for each of the parameter vectors θ_i for $i = 0, \ldots, J$ is the Dirichlet distribution. Conditional on a vector of parameters β_j, the density function of this distribution is

$$p(\theta_j|\beta_j) = \frac{\Gamma(|\beta_j|)}{\Gamma(\beta_j)} \theta_j^{\beta_j - 1},$$

where we use the notation that if x and y are vectors, $\Gamma(x) = \prod_i \Gamma(x_i)$, $|x| = \sum_i x_i$, and $x^y = \prod_i x_i^{y_i}$ ($\Gamma(x)$ is the gamma function $\Gamma(x) = \int_0^\infty u^{x-1} \mathrm{e}^{-u}\, du$).

To write the likelihood in a simple fashion, we introduce some notation. Define the function h to map a set of letters to a p vector that counts the frequency of each letter. We also introduce a bracket notation that takes a single starting position of an element and returns the entire element, i.e. $\{A\} = \{(k, a_k + j - 1) : k = 1, \ldots, K, j = 1, \ldots, J\}$. In adition we define the set $A(j) = \{(k, a_k + j - 1) : k = 1, \ldots, K\}$. With these definitions in addition to our rules for powers of vectors given above, we can write

$$p(A, R|\theta_0, \Theta) \propto \theta_0^{h(R_{\{A\}^c})} \prod_{j=1}^{J} \theta_j^{h(R_{A(j)})}.$$

Using this in the previous expression for $p(A|R)$ in addition to our priors, we find that

$$p(A|R) \propto \Gamma(h(R_{\{A\}^c}) + \beta_0) \prod_{j=1}^{J} \Gamma(h(R_{A(j)}) + \beta_j).$$

This result follows from the fact that the Dirichlet distribution integrates to one.

While we have an expression for the posterior distribution of interest, it is not clear what we should do with this expression in order to conduct inference. The approach of Liu et al. (1995) is to find the

conditional distribution of a_k given $A_{[-k]} = \{(j, a_j) : j \neq k\}$, then iteratively sample from these conditional distributions, or approximations to these distributions. That is, they propose to use the Gibbs sampler in order to conduct the inference. We will now compute these conditional distributions. Now

$$p(a_k|A_{[-k]}, R) = \frac{p(A|R)}{p(A_{[-k]}|R)},$$

but

$$p(A_{[-k]}|R) \propto \Gamma(h(R_{\{A_{[-k]}\}^c}) + \beta_0) \prod_{j=1}^{J} \Gamma(h(R_{A_{[-k]}(j)}) + \beta_j)$$

since we just ignore the motif element in sequence k to get this conditional. Next, note that the bracket notation implies $\{a_k\} = \{(k, a_k + j - 1) : j = 1, \ldots, J\}$, hence

$$h(R_{\{a_k\}}) + h(R_{\{A\}^c}) = h(R_{\{A_{[-k]}\}^c}),$$

so that

$$\begin{aligned}
p(a_k|A_{[-k]}, R) &\propto \frac{\Gamma(h(R_{\{A\}^c}) + \beta_0)}{\Gamma(h(R_{\{A_{[-k]}\}^c}) + \beta_0)} \prod_j \frac{\Gamma(h(R_{A(j)}) + \beta_j)}{\Gamma(h(R_{A_{[-k]}(j)}) + \beta_j)} \\
&\propto \frac{\Gamma(h(R_{\{A_{[-k]}\}^c}) - h(R_{\{a_k\}}) + \beta_0)}{\Gamma(h(R_{\{A_{[-k]}\}^c}) + \beta_0)} \prod_j \frac{\Gamma(h(R_{A(j)}) + \beta_j)}{\Gamma(h(R_{A_{[-k]}(j)}) + \beta_j)} \\
&\propto \frac{\Gamma(h(R_{\{A_{[-k]}\}^c}) - h(R_{\{a_k\}}) + \beta_0)}{\Gamma(h(R_{\{A_{[-k]}\}^c}) + \beta_0)} \prod_j \frac{\Gamma(h(R_{A_{[-k]}(j)}) + h(r_{k,a_k+j-1}) + \beta_j)}{\Gamma(h(R_{A_{[-k]}(j)}) + \beta_j)}
\end{aligned}$$

where the last line is due to the equality

$$h(R_{A(j)}) = h(R_{A_{[-k]}(j)}) + h(r_{k,a_k+j-1}).$$

In this last expression, $h(r_{k,a_k+j-1})$ is just a p vector with a single 1 and the rest of the entries zero. But if m is a vector of positive integers and v is a vector of all zeros except a 1 at a single entry, then

$$\begin{aligned}
\Gamma(m + v)/\Gamma(m) &= \prod_i \Gamma(m_i + v_i)/\Gamma(m_i) \\
&= \prod_i (\Gamma(m_i + v_i)/\Gamma(m_i))^{v_i} \\
&= \prod_i m_i^{v_i} \\
&= m^v
\end{aligned}$$

where the second to last line is by the relationship between the factorial function and the gamma function (i.e. $\Gamma(x) = (x-1)!$, a results that follows from integration by parts in the integral definition of the gamma function). Hence we arrive at the conditional

$$p(a_k|A_{[-k]}, R) \propto \frac{\Gamma(h(R_{\{A_{[-k]}\}^c}) - h(R_{\{a_k\}}) + \beta_0)}{\Gamma(h(R_{\{A_{[-k]}\}^c}) + \beta_0)} \prod_j [h(R_{A_{[-k]}(j)}) + \beta_j]^{h(r_{k,a_k+j-1})}.$$

Since this expression involves the ratio of 2 gamma functions, the authors actually use an approximation in their computations. This approximation uses the trick about ratios of gamma functions given above, which is exact if the vector v has just a single non-zero entry and is an approximation in other circumstances. Thus they approximate the ratio of gamma functions with $[h(R_{\{A_{[-k]}\}^c})^{h(R_{\{a_k\}})}]^{-1}$, and this gives the approximate posterior as

$$p(a_k|A_{[-k]},R) \propto \prod_j \left[\frac{h(R_{A_{[-k]}(j)})+\beta_j}{h(R_{\{A_{[-k]}\}^c})+\beta_0}\right]^{h(r_{k,a_k+j-1})},$$

or more simply

$$p(a_k=i|A_{[-k]},R) \propto \prod_j \left(\frac{\hat{\theta}_{j[-k]}}{\hat{\theta}_{0[-k]}}\right)^{h(r_{k,i+j-1})},$$

where $\hat{\theta}_{j[-k]}$ is the posterior mean for position j given the current alignment, and $\hat{\theta}_{0[-k]}$ is the posterior mean for the locations not involved in the alignment. Hence the algorithm proceeds by sampling from each of these distributions iteratively.

1.4.1 Extensions

There are several important extensions that one can build into the basic model. These extensions are very important from the perspective of applications, but are trivial extensions from the perspective of finding the conditional distributions, hence we do not provide much detail in regard to these computations. First, we can easily accommodate multiple motifs within a given sequence. The trick is to pretend all of the sequence data is just from one long sequence and look for motifs that don't overlap or extend over boundaries of the actual sequences. In practice one can just ignore draws from the Gibbs sampler that have these properties.

Slightly more complicated is the extension to insertions in the motifs. Here we select a number $W > J$ and look for a motif of length W. The difference is, we now introduce a set of W indicator variables that indicate if each position is part of the motif (so only J of these indicators are 1 and the rest are zero). One must provide a prior for this vector. In practice a uniform prior over the space of $\binom{W}{J}$ indicator variables that have J ones seems too diffuse, hence priors that restrict the span of the motif are typically used. These models are referred to as *fragmentation models*.

Similarly, we can treat the case where not all sequences have a motif element and some sequences have multiple motif elements. The idea is to once again treat all the sequences as one long sequence of length L^* and introduce $L^* - J + 1$ indicator variables that represent if an element starts at each possible location. In addition, we can even allow for these indicator variables to be n valued instead of just binary. In this way we can treat the problem of more than one type of motif.

1.5 The propagation model

The primary drawback of the block motif methods is that one must specify the size of the blocks and the number of motifs. In contrast, the hidden Markov model based approaches do not require the user to specify this information, but the generality of hidden Markov models for multiple alignment implies they are often not very sensitive due to the large number of parameters that must be estimated. Hence there is a desire for methods that can are sensitive yet flexible, requiring little user input. The propagation model of Liu, Neuwald and Lawrence (1999) seeks to fulfill these requirements. This method has been implemented in the software called PROBE.

Consider the basic set up of the block motif methods. Suppose there are L elements in each sequence, and use A to now denote the start locations of all of these elements in each sequence. Let w_l denote the width of the l^{th} motif. If $A_{.l}$ is the starting positions of the l^{th} motif in all sequences, then supposing there are no deletions in the motifs, we can write the likelihood for R as

$$L(R|A,\theta_0,\Theta) \propto \theta_0^{h(R)} \prod_{l=1}^{L} \prod_{j=1}^{w_l} (\theta_j^l/\theta_0)^{h(R_{A_{.l}+j-1})}.$$

As before, but now conditional on A, we can integrate θ_0 and Θ out of the posterior to obtain a likelihood for the sequence data conditional on the alignment

$$L(R|A) \propto \Gamma(h(R_{\{A\}^c}) + \beta_0) \prod_{l=1}^{L} \prod_{j=1}^{w_l} \Gamma(h(R_{A_{.l}+j-1}) + \beta_j^l).$$

The novel aspect of these models that allows for a connection to the hidden Markov models is that we now use a Markovian prior for the distribution of the motif start sites within each sequence:

$$p(A_{k.}) \propto \prod_{l=1}^{L} q_l(a_{k,l}, a_{k,l+1}),$$

where $A_{k.}$ is the portion of the alignment relevant to sequence k. We suppose the alignments are independent across the different sequences $p(A) = \prod_k p(A_{k.})$. If $w_l = 1$ for all l then this model reduces to the usual hidden Markov model. For the function $q_l(x,y)$ we require $q_l(x,y) = 0$ for $|x-y| < w_l$ for motifs to not overlap. While any form of q that satisfies this criterion is acceptable, Liu et al. (1999) recommend using $q = 1$ when it is not zero.

The computation of the posterior distribution mimics the calculations for the block motif model. Once again, we compute the conditional distributions and use the Gibbs sampler to simulate from the posterior distribution. We find

$$p(A_{k.} = (i_1,\ldots,i_L)|R, A_{[-k]}) \propto \prod_{l=1}^{L} q_l(i_l, i_{l+1}) \prod_{j=1}^{w_l} \left(\frac{\hat{\theta}_j^l}{\hat{\theta}_0}\right)^{h(r_{k,i_l+j-1})},$$

where $\hat{\theta}$ are posterior means as before. A complication that now arises is that we need the normalizing constant for this conditional posterior in order to sample. The authors provide a recursive algorithm in order to compute this normalizing constant.

As in the usual block motif model, we can consider the extension of the fragmentation model and allow for deletions as before. Also as in the block motif model, one must specify the block sizes, W, and number of motifs, L. The authors propose to use *Bayes factors* $p(R|W,L)$ to select these values, but since one can not compute this probability in any simple way (one could use MCMC) they propose to use an approximate method based on $p(R|\hat{A})p(\hat{A}))$, where $\hat{A}$ is the posterior mode of the alignment.

References

Liu, J., Neuwald, A., Lawrence, C. (1995), "Bayesian models for multiple local sequence alignment and Gibbs sampling strategies", *Journal of the American Statistical Association*, 90, 1156–1170.

Liu, J., Neuwald, A., Lawrence, C. (1999), "Markovian structures in Biological sequence alignments", *Journal of the American Statistical Association*, 94, 1–15.