1 Brief Review

In the previous lecture we continued our discussion of sequence alignment algorithms specifically we used the divide and conquer approach to come up with a space efficient approach to sequence alignment. We went on to explain how our algorithm can be modified to enable local alignments, finishing with the heuristic approach for BLAST search.

2 A Probabilistic Model

In our discussion of sequence alignment algorithms we assumed we have a given set of scoring rules for generating the alignment. Today we will discuss a way for generating this scoring rule.

2.1 Two different approaches

It is possible to create a scoring matrix by calculated selection of criteria according to any arbitrary set of biological constraints. Still we must realize that there are countless constraints to keep in mind and once we have generated this matrix according to a chosen set of criteria we hardly have any assurances as to its success in estimating the alignment score.

We would rather create the matrix in accordance with some methodology that will give some indication to its success in estimating the likelihood of an alignment. For this we us a training set of “real” alignments. Two models exist:

1. Generative method - Use a probabilistic model for translating the frequencies in the training set to a score.

2. Discriminative method - Choose a score that prefers the training alignments to alternatives.

We will focus in our discussion on the first approach.

2.2 Building the model

Let us assume we can divide our analysis of the problem into two disjoint complimentary occurrences:

1. M - The sequences are evolutionarly related \( Pr(\vec{x}, \vec{y}|M) \)

2. R - The sequences are unrelated \( Pr(\vec{x}, \vec{y}|R) \)

Let us consider the latter case first. We shall assume that the value of a given position in the sequence is independent of adjacent positions in the sequence. Also if the sequences are unrelated we can assume that \( \vec{x}, \vec{y} \) at any position i are independent of each other.
In other words for any position i, both $x_i, y_i$ are sampled independently from some background distribution $q(\cdot)$. And so the likelihood of the $\vec{x}, \vec{y}$ given that they are unrelated is:

$$Pr(\vec{x}, \vec{y}|R) = \prod_{i=1}^{n} q(x_i) \prod_{i=1}^{n} q(y_i)$$

(1)

In the first case we are assuming the two sequences are related, that is they evolved from a common ancestor. For simplicity we will continue assuming that each position $i$ in $(x_i, y_i)$ is independent of the other positions. So if we assume $x_i, y_i$ are sampled from some distribution $p(\cdot, \cdot)$ of letter pairs. The probability that any two letters $a, b$ evolved from some ancestral letter is $p(a, b)$. And so the likelihood of the $\vec{x}, \vec{y}$, given that they are related is:

$$Pr(\vec{x}, \vec{y}|M) = \prod_{i=1}^{n} p(x_i, y_i)$$

(2)

2.3 A decision problem

So now we’ve stumbled across a decision problem. Given the two sequences $\vec{x}, \vec{y}$ we have to decide whether they are sampled form R or from M. We want to construct a decision procedure $D(\vec{x}, \vec{y})$ that returns M or R. Basically we want to compare the likelihood of our data in both models. Let us examine decision rules of the form:

$$D_k(\vec{x}, \vec{y}) = \begin{cases} M & \frac{Pr(\vec{x}, \vec{y}|M)}{Pr(\vec{x}, \vec{y}|R)} \geq k \\ R & \frac{Pr(\vec{x}, \vec{y}|M)}{Pr(\vec{x}, \vec{y}|R)} < k \end{cases}$$

The following statistics lemma gives us an interesting result:

Lemma 2.2 (Neyman-Pearson) - For any $k$ and any $D_k$ of the above form and let $D'$ be another decision rule, then if $\alpha(D') \leq \alpha(D_k)$ then it follows that $\beta(D') \geq \beta(D_k)$.

We would of course favor a procedure which minimizes both types of error.

Definition 2.1 Given two decision rules $D_1$ $D_2$ we would say that $D_2$ is dominated by $D_1$ if $\alpha(D_1) < \alpha(D_2)$ and $\beta(D_1) < \beta(D_2)$. 

We can define our scoring rule matrix as follows:

$$\sigma(a, b) = \log \frac{p(a, b)}{q(a)q(b)}$$
2.4 Estimating Probabilities

So now if we could only estimate the probabilities $p(\cdot, \cdot)$ and $q(\cdot)$ from our data we would have our scoring matrix $\sigma(\cdot, \cdot)$.

Let’s discuss parameter estimation through an example:

Example 2.3 Thumbtack Throwing

When tossed a thumbtack can either land on its Head or its Tail with an unknown probability. The estimation task: Given a sequence of toss samples $x_1, x_2, \ldots, x_m$ we would like to estimate the probabilities $Pr(H) = \theta$ and $Pr(T) = 1 - \theta$.

If the thumbtack are identical and assuming they are thrown in this same way each time then it’s pretty safe to say that samples are independent and identically-distributed (i.i.d.) And so given that $Pr(H) = \theta$ then $Pr(x_1, \ldots, x_m : \theta) = \prod_{i=1}^{m} Pr_H(x_i)$

By now we must realize that the tack throwing is a binomial experiment and so it is distributed binomially.

That is $Pr(\#H = k) = \binom{m}{k} \theta^k (1 - \theta)^{m-k}$

Thus the expectation is $E[\#H] = \frac{\#H}{m}$ and we intuitively realize that we can estimate $\theta$ by $\frac{\#H}{m}$.

If we’d like to generalize to process of finding and estimator one method is the MLE method - Maximum Likelihood Estimator.

We need to find an estimator $\hat{\theta}$ that will maximize $L(\theta) \Rightarrow \hat{\theta} = \arg \max_{\theta} L(g)$.

Basically what we do to find an estimator is:

1. build a likelihood function $L$.
2. find a maximum for the likelihood function.

For example in the tack problem we could define the likelihood function:

$$L(\theta|D) = \theta^m (1 - \theta)^m$$

Where $m_H, m_T$ is defined $m_T = \sum_i 1_{x_i=H}, m_T = \sum_i 1_{x_i=T}$

We would also like to point out that $m_H, m_T$ are sufficient statistics.

Definition 2.4 $S(D)$ is a sufficient statistic if for any $DD'$ if $S(D) = S(D')$ then $L(\theta|D) = L(\theta|D')$.

Basically all this means is that the transformation $S$ keeps all the information necessary to compute the likelihood.

So now we need to find the maximum of our likelihood function:

We find $L'(\theta) = 0$ and if possible $L''(\theta) < 0$.

Often it is also useful to look at log likelihood function which we denote by $\ell(\theta)$.

For example in the above problem thumbtack problem:

$$\ell(\theta) = \log L(\theta) = M_H \log(\theta) + m_T \log(1 - \theta)$$

$$\ell' = \frac{m_H}{\theta} - \frac{m_T}{1 - \theta} = 0 \Rightarrow \theta = \frac{m_H}{m_H + m_T}$$

2.5 Back to the problem at hand

So we still need to estimate our letter probabilities in our sequence.
2.5.1 Finding $q(\cdot)$

So now we have $1, \ldots, k$ possibly different probabilities for each single letter we shall mark them $\vec{\theta} = (\theta_1, \ldots, \theta_k)$

So if we maintain our previous assumptions about independence between positions and sequences that our likelihood function should be:

$$L(\vec{\theta}) = \prod_{j=1}^{m} p(x_j) = \prod_{k} \theta_k^{N_k}$$

Where $N_k = \sum_j 1_{x_j = k}$ and $\forall k (0 \leq \theta_k \leq 1)$ and $\sum_k \theta_k = 1$

So $\ell(\vec{\theta}) = \sum_k m_k \log(\theta_k)$

We can solve this problem for the general case:

$$\arg\max_{\vec{\theta}} f(\vec{\theta}) s.t.c(\vec{\theta}) = 0$$

$$J(\vec{\theta}, \lambda) = f(\vec{\theta}) - \lambda c(\vec{\theta}), \forall i \frac{\partial J}{\partial \theta_i} = 0 \Leftrightarrow \frac{\partial f}{\partial \theta_i} = \lambda \frac{\partial c}{\partial \theta_i}$$

$$\frac{\partial J}{\partial \lambda} = 0 \Leftrightarrow c(\vec{\theta}) = 0$$

Now let's solve again for our problem:

$$J = \sum_k m_k \log(\theta_k) - \lambda(\sum_k m_k - 1)$$

$$\frac{\partial J}{\partial \theta_k} = \frac{m_k}{\theta_k} - 1 \Rightarrow \frac{m_k}{\theta_k} - 1 = 0 \Rightarrow \theta_k = \frac{m_k}{\lambda}$$

$$0 = \sum_k \theta_k - 1 = \sum_k \frac{m_k}{\lambda} - 1 \Rightarrow \lambda = \sum_k m_k \Rightarrow \theta_k = \frac{m_k}{\sum_k m_k} = \frac{m_k}{m}$$

2.5.2 Finding $p(\cdot, \cdot)$

We will examine two similar approaches.

2.5.3 PAM - Percent Accepted Mutations

Very similar sequences can easily be aligned using naive means (I believe this was originally done by hand) and the differences between these sequences may be studied.

We shall define 99% of identity as 1 unit of time (u.o.t). That is 1 change in every 100 nucleotides. It should be noted that 1 u.o.t. does not correspond to any given time period.

The information about the individual kinds of mutations and about the relative mutability of the amino acids or nucleotides can be combined into one distance-dependent “mutation probability matrix”. The elements of this matrix give the probability that the amino acid or nucleotide in one column will be replaced by the amino acid or nucleotide in some row after a given evolutionary interval we defined a 1 u.o.t. For example, a matrix with an evolutionary distance of 0 PAMs would have ones on the main diagonal and zeros elsewhere. A matrix with an evolutionary distance of 1 PAM would have numbers close to one on the main diagonal and small numbers off the main diagonal.

PAM1: $P_1(x_i, y_i)$ = The probability $x,y$ diverged from a single ancestor 1 u.o.t apart.

In order to calculate the next PAM matrix:

$$P_2(x, y) = \sum_y P(x, y, z) = \sum_y P_1(x, y) P_1(z|y) = p(x) \sum_y P(y|x) P(z|y)$$

It is common to use PAM-250 matrices. At this evolutionary distance (250 substitutions per hundred residues) only one amino acid in five remains unchanged and the percent divergence has increased to roughly 80
2.5.4 BLOSUM - BLOck SUbstitution Matrix

This approach is the more recent one of the two and it is based on the idea that a certain percent of conservation between two sequences defines the evolutionary distance between them. For a certain value of evolutionary distance, the calculation of $p_{m}(a, b)$ is based on the same principle as the ones discussed above, with the exception that this time, we only count events that might occur during a defined evolutionary time range i.e.: we only sum over aligned sequences that have the corresponding percent of similarity between them. For example, a BLOSUM62 matrix is calculated from protein blocks such that if two sequences are more than 62% identical, then the contribution of these sequences is weighted to sum to one.