Multiple Sequence Alignment

Lecturer: Marina Alexandersson

6 November 2002

1 Introduction

In pairwise alignment we used the likelihood ratio

$$s(a,b) = \ln \frac{\Pr(a,b \mid \text{model})}{\Pr(a,b \mid \text{random})} = \ln \frac{p_{ab}}{q_a q_b}$$

and the total score an alignment

E A A S
V F - T

would be

$$S = s(E,V) + s(A,F) + \gamma(1) + s(S,T)$$

where $s(a,b)$ came from a substitution score matrix such as PAM or BLOSUM, and γ was a gap score.

How do we align three or more sequences?

E A A S
V F - T
G B A -

A natural generalization would be

$$S = s(E,V,G) + s(A,F,B) + s(A,-,A) + s(S,T)$$

where

$$s(a,b,c) = \ln \frac{p_{abc}}{q_a q_b q_c}.$$

But how do we obtain p_{abc}? And what about p_{abcd}? not possible, so we need a method that is based on pairwise scores.

2 SP scores (Sum of Pairs)

We assume the columns to be independent and the score for each column

E
V
G

becomes the sum of all pairwise scores

$$S = s(E,V) + s(E,G) + s(V,G)$$

where s comes from PAM or BLOSUM.

Formally, if

$$\begin{aligned} m_i &= \text{residues in column } i \\ m_i^j &= \text{residue in column } i \text{and sequence } j \end{aligned}$$

then the total score is

$$S = \sum_i S(m_i) = \sum_i \sum_{j<k} s(m_i^j, m_i^k).$$

(Note: $j < k$ to not sum over $s(a, b) = s(b, a)$ twice.)

One problem with SP scores is that the different pairs in the multiple alignment might not be at the same evolutionary distance. Hence using the same scoring matrix might not be "fair" on the different sequences. With SP scores we assume that any of the sequences could be the ancestor of the others.

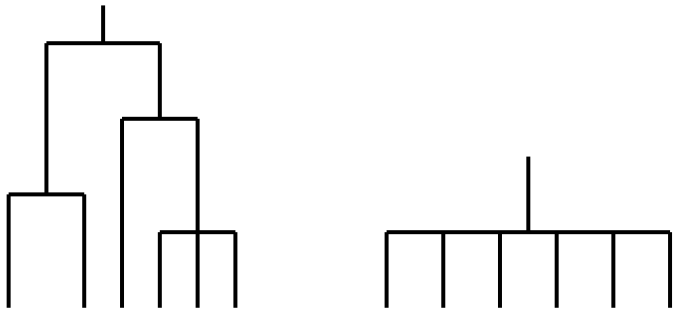


Figure 1: The left tree is an example of a real situation, with organisms at different evolutionary distances. The tree to the right is the assumption in the SP scores method, where all sequences are at equal distances to the ancestor.

Ideally we would like to model the molecular sequence evolution. Given the correct phylogenetic tree for the sequences, the probability of a multiple alignment would be the product of the probabilities of all the evolutionary events necessary to produce that alignment. This is a very complex model and not enough data to estimate the parameters.

It is possible to generalize dynamic programming as used in pairwise alignment to multiple alignments, but this will require a lot of memory and many computations and would be restricted to only small problems. In pairwise alignments we had an $n \times m$ matrix F where

$$F(i, j) = \text{score of best alignment up to } (i, j)$$

and we filled the matrix in a tabular fashion by calculating these values recursively. For three sequences we would need a cube, for four sequences a four-dimensional hypercube and so on, such that the matrix F would have as many dimensions as sequences.

3 MSA (Multiple Sequence Alignment)

Let a^{kl} be the score for an alignment between sequences k and l. Then the score for the multiple alignment is

$$S = \sum_{k<l} S(a^{kl}).$$

If $\hat{a}^{kl}$ is the best pairwise alignment then

$$S(a^{kl}) \leq S(\hat{a}^{kl}).$$

MSA uses a lower bound β^{kl} and only considers pairwise alignments of higher score

$$S(a^{kl}) \geq \beta^{kl}.$$

For each sequence pair x^k and x^l we create a subset of possible alignments such that in each coordinate combination int the alignment (i_k, i_l) the score for the best alignment going through that point is greater than β^{kl}. This is done for all pairs of sequences and then the multi-dimensional dynamic programming algorithm is performed in this subset of the hypercube.

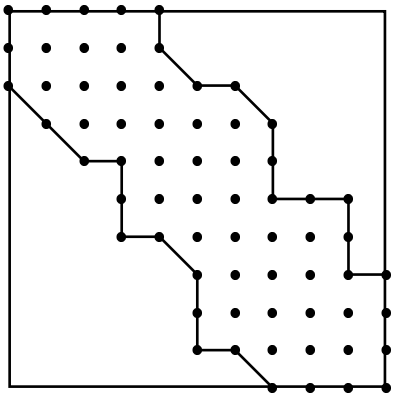


Figure 2: We include points (i_k, i_l) that are such that the best alignment going through that point has a score that is higher than the threshold β^{kl}.

4 Progressive alignment

The idea behind progressive alignments is to use dynamic programming to build a multiple alignment, starting with the most related sequences and then adding sequences or groups of sequences progressively.

The algorithms for progressive alignments differ in several ways:

1. In the way they choose alignment order.
2. In whether one sequence is aligned at a time to a growing alignment, or whether subalignments are built and then aligned to each other.
3. In how to align and score sequences to existing alignments.

The advantage with progressive alignments are that they are fast and efficient. The disadvantage is that we might not reach the optimal alignment.

4.1 The Feng-Dolittle algorithm

The Feng-Dolittle algorithm is one of the first progressive alignment algorithms. The procedure is as follows:

(i) Perform pairwise alignment of all N sequences ($= N(N-1)/2$ pairs). Convert the alignment scores to evolutionary distances using a normalized percentage of similarity.

(ii) Construct a "guide tree" from these distances (using clustering), displaying the evolutionary relationships.

(iii) Align the most related sequences in the tree using dynamic programming.

(iv) Align the sequence most closely related to the existing alignment, *or* the next most related pair to each other, *or* two subalignments.

In (iii) and (iv):

- Sequence – sequence alignment: regular pairwise alignments.
- Sequence – subalignment (group): pairwise to each sequence in group. Highest scoring alignment determines the alignment to the group.
- Group – group: all pairwise alignments between groups are performed, and the best determines the alignment.

Pairwise PAM scores and affine gap penalties are used. After each alignment gaps are replaced by a neutral character X, so that alignment to that character has no penalty score in succeeding alignments. In this way gaps are always kept. When adding a sequence to a group, new gaps may be introduced to keep the alignment consistent, but no gaps are removed. The side effect is that gaps tend to occur in the same columns in subsequent pairwise alignments.

Example

Assume that we have four sequences, S_1, S_2, S_3 and S_4, and that pairwise alignments of all pairs (6 possible pairs: (S_1S_2), (S_1S_3), (S_1S_4), (S_2S_3), (S_2S_4), (S_3S_4)) gives an evolutionary tree

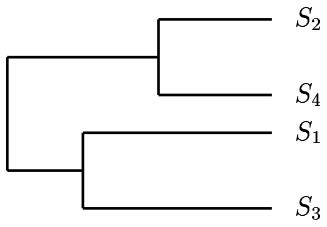
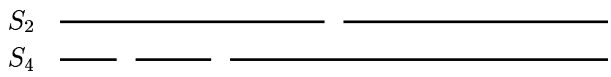
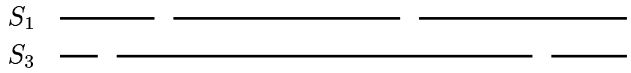


Figure 3: Evolutionary tree based on pairwise alignments between all possible pairs.

- Align S_2 and S_4.



- Align S_1 and S_3.



- Align (S_2, S_4) with (S_1, S_3).