

MA3259 Lecture 5

Multiple Sequence Alignment

LX Zhang

Department of Mathematics

National University of Singapore

matzlx@nus.edu.sg

1. What is Multiple Sequence Alignment(MSA)

MSA is the alignment of three or more DNA or protein sequences.

Goal: To write each sequence along the others to bring up the similarity among the sequences.

- each letter of sequence is either placed alongside a corresponding letter in the other sequences or a space symbol.



Motivation for MSA

- MSA can reveal biologically important but weak sequence similarity that are often hidden in the sequences. So they can provide strong evidence for structural and functional inferences that are invisible in pairwise sequence alignment
- MSA can provide information about the evolution of the related sequences.
- MSA can give insight to the composition and other facts about sequence similarity between homologous sequences.

2. Scoring MSAs:

The sum of pairs (SP) scheme

- As pairwise sequence alignment, the score for a multiple sequence alignment \mathcal{A} is equal to the sum of the scores for all columns (m_j) in the alignment:

$$SP(\mathcal{A}) = \sum_j SP(m_j).$$

- Given a scoring matrix that gives the score $s(x,y)$ for aligning two letters x and y , the score $SP(m_j)$ for column m_j is calculated using the formula:

$$SP(m_j) = \sum_{k < l} s(m_j^{(k)}, m_j^{(l)})$$

where $m_j^{(k)}$ and $m_j^{(l)}$ are the k -th entry and l -th entry in the column respectively and we assume $s(-, -) = 0$.

Scoring Example: DNA sequences

Consider three sequences:

Seq1: GCTC

Seq2: AC

Seq3: GATC

We wish to use the SP method to score the following alignment \mathcal{A} of the sequences:

G	C	–	T	C
–	–	A	–	C
G	–	A	T	C

If we use the following simplified scoring matrix:

$$s(x, x) = 3 \quad (\text{match scores } 3)$$

$$s(x, y) = -2, \quad x \neq y \quad (\text{mismatch scores } -2)$$

$$s(x, -) = s(-, y) = -1 \quad (\text{indel scores } -1)$$

$$s(-, -) = 0, \quad (\text{prevent double counting of gaps})$$

m_1	m_2	m_3	m_4	m_5
G	C	-	T	C
A	-	-	-	C
G	-	A	T	C
-1	-2	-2	1	9

$$SP(\mathcal{A}) = 5$$

$$\begin{aligned} SP(m_1) &= s(G, A) + s(G, G) + s(A, G) \\ &= -2 + 3 + -2 \\ &= -1 \end{aligned}$$

$$\begin{aligned} SP(m_2) &= s(C, -) + s(C, -) + s(-, -) \\ &= -1 + -1 + 0 \\ &= -2 \end{aligned}$$

$$\begin{aligned} SP(m_3) &= s(-, -) + s(-, A) + s(-, A) \\ &= 0 + -1 + -1 = -2 \end{aligned}$$

$$\begin{aligned} SP(m_4) &= s(T, -) + s(T, T) + s(-, T) \\ &= -1 + 3 + -1 = 1 \end{aligned}$$

$$\begin{aligned} SP(m_5) &= s(C, C) + s(C, C) + s(C, C) \\ &= 3 + 3 + 3 = 9 \end{aligned}$$

Problem with the SP scheme

- It tends to overweight the effect of mutations especially when an alignment has a large number of sequences. Let match score 3 and mismatch score -2.

$$\text{SP} \begin{pmatrix} G \\ G \\ G \\ G \\ G \end{pmatrix} = \binom{5}{2} s(G, G) = 10 \times 3 = 30 \quad \text{SP} \begin{pmatrix} A \\ G \\ G \\ G \\ G \end{pmatrix} = \binom{4}{2} s(G, G) + 4s(A, G) = 10$$

In general, a match column having n G's scores $S = 3n(n-1)/2$, and a column having only letter different from others scores $S' = 3(n-1)(n-2)/2 - 2(n-1)$.

$$\frac{S - S'}{S} = \frac{3n(n-1)/2 - [3(n-1)(n-2)/2 - 2(n-1)]}{3n(n-1)/2} = \frac{10}{3n}$$

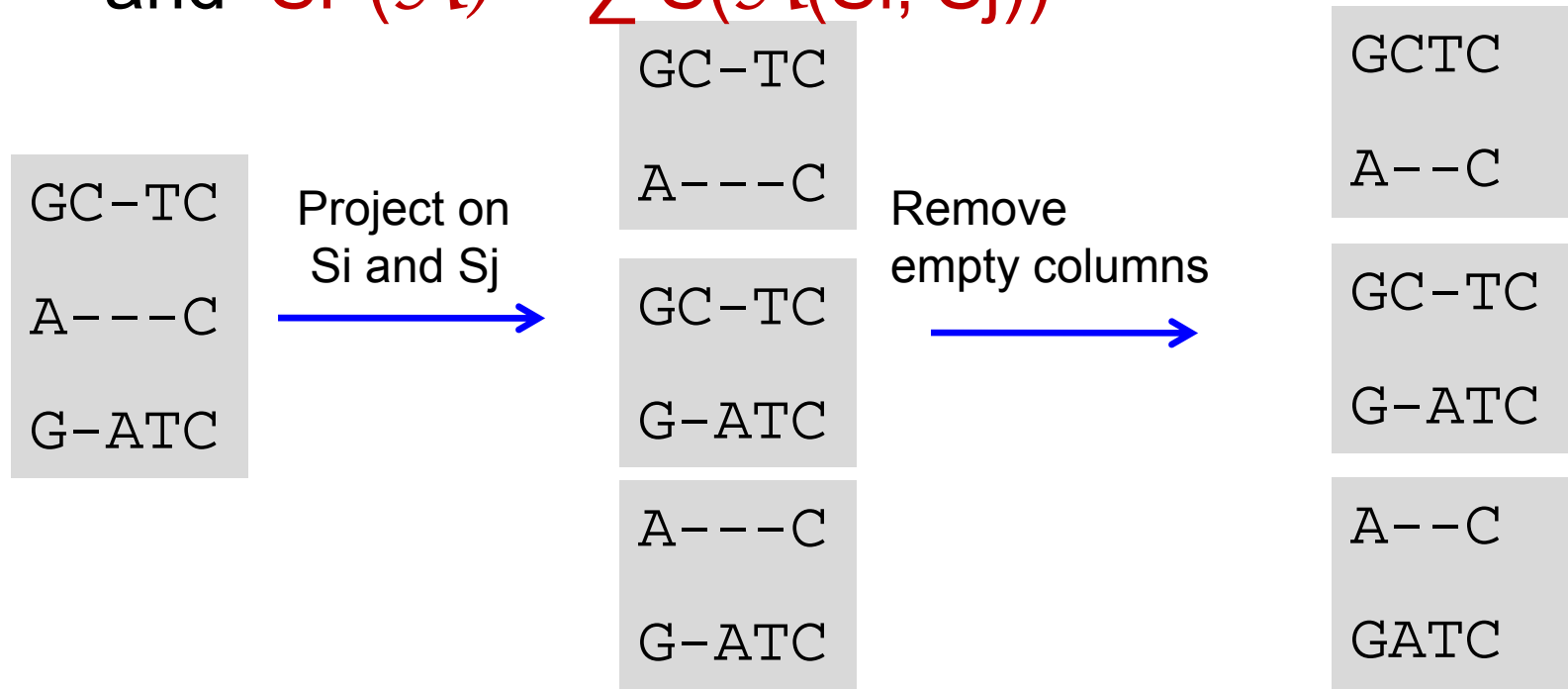
So, the relative difference in score due to a single mutation decreases as the number of sequences.

Efficiency of the SP scheme

- However, it allows us to generalize different methods for pairwise sequence alignment to MSA due to the following fact:

An alignment \mathcal{A} of k sequences S_1, S_2, \dots, S_k induces $k(k-1)/2$ pairwise alignments $\mathcal{A}(S_i, S_j)$

and $SP(\mathcal{A}) = \sum S(\mathcal{A}(S_i, S_j))$



Remark

With the SP scoring scheme, the optimal multiple sequence alignment does not necessarily induce an optimal alignment for each pair of sequences.

p a t r

p a - r

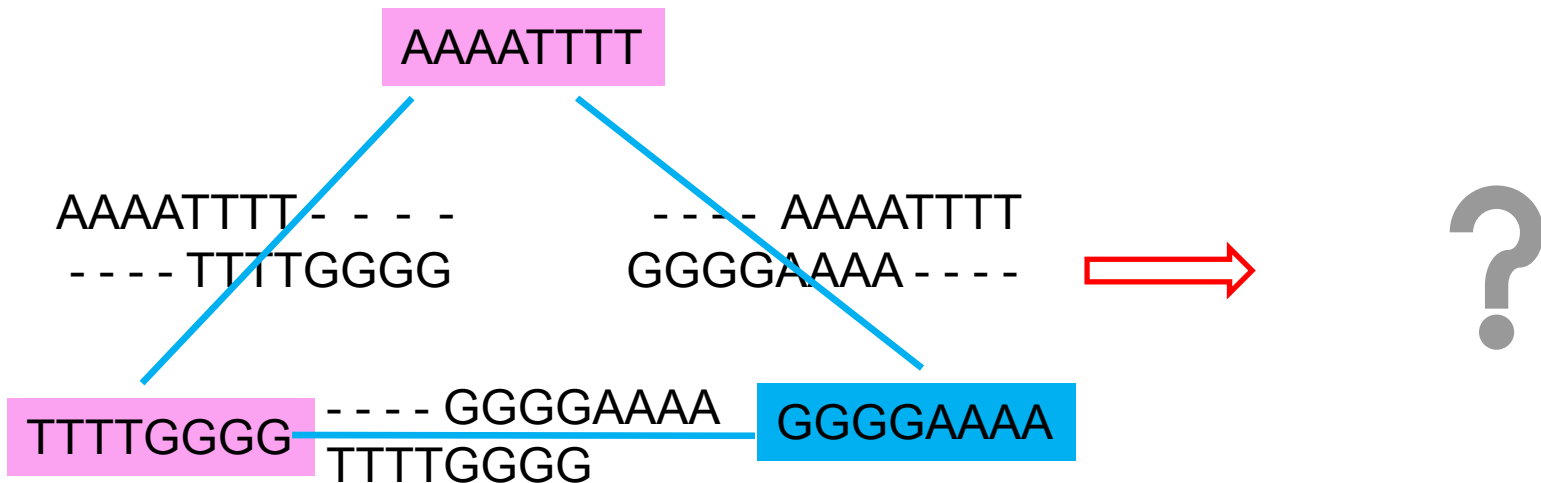
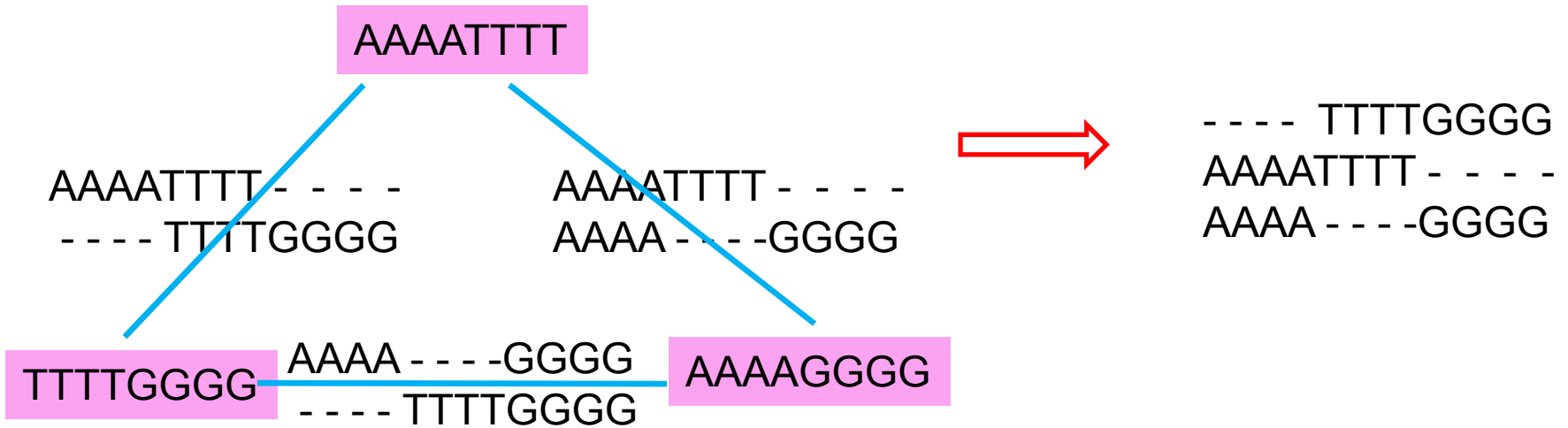
p - t r

Assume we obtain it by scoring mismatch and indel -2, then the optimal alignment for the 2nd and 3rd sequence is

p a r

p t r

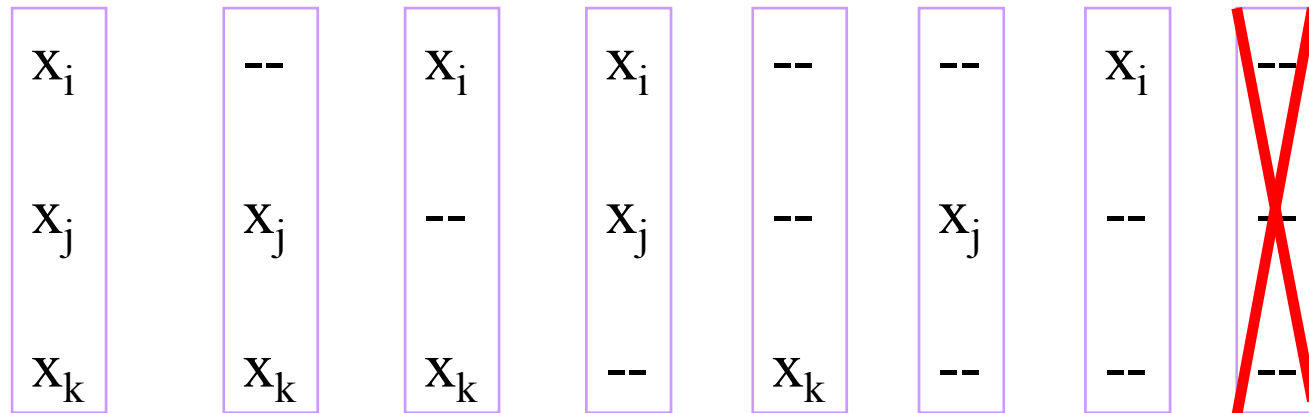
Aligning multiple sequence is hard when the optimal pairwise alignments are not compatible.



3. Dynamic Programming Method for MSA

Consider three sequences $S1=x[1,..m]$, $S2[1,..,m]$, $S3[1,..,m]$.

Then, any alignment of these prefixes has 7 possibilities in the last column:



Let $F(i, j, k)$ denote the score of optimally aligning the i -character prefix of $S1$, the j -character prefix of $S2$ and the k -character prefix of $S3$.

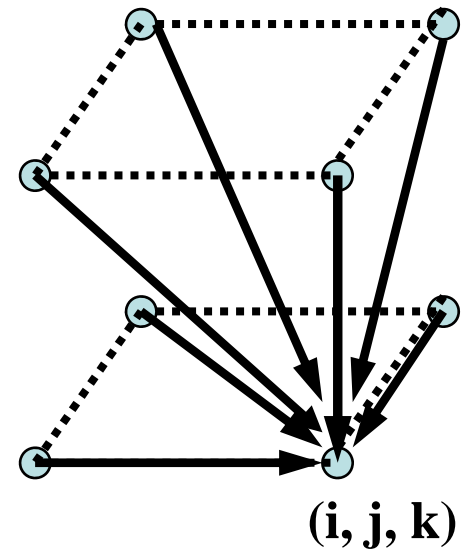
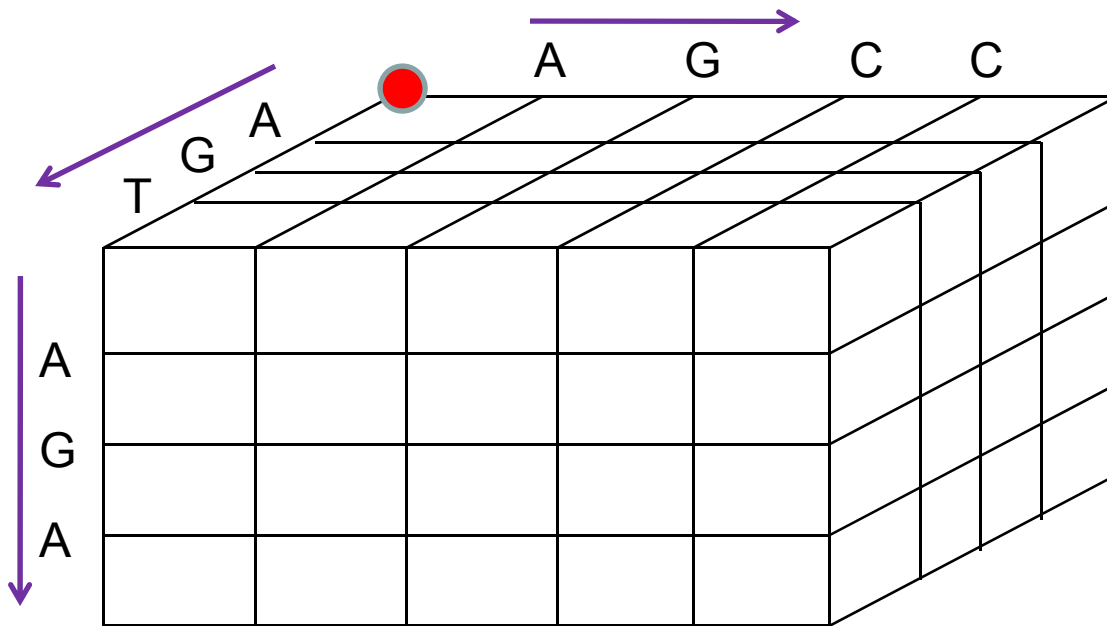
$$F(i, j, k) = \begin{cases} F(i-1, j-1, k-1) + s(x_i, y_j, z_k); \\ F(i-1, j, k) + s(-, y_j, z_k); \\ F(i, j-1, k) + s(x_i, -, z_k); \\ F(i, j, k-1) + s(x_i, y_j, -); \\ F(i-1, j-1, k) + s(x_i, y_j, -); \\ F(i-1, j, k-1) + s(x_i, -, z_k); \\ F(i, j-1, k-1) + s(-, y_j, z_k); \end{cases}$$

Basis values

$$F(0, 0, k) = \sum s(-, -, z_t),$$

$$F(i, 0, 0) = \sum s(x_t, -, -),$$

$$F(0, j, 0) = \sum s(-, y_t, -).$$



Aligning k Sequences

In general, for aligning k m-char sequences, we need a k-dimensional array of $(m+1)^k$ cells. At the last column of an alignment, there are $2^k - 1$ possibilities (1 is subtracted because all gaps are not allowed).

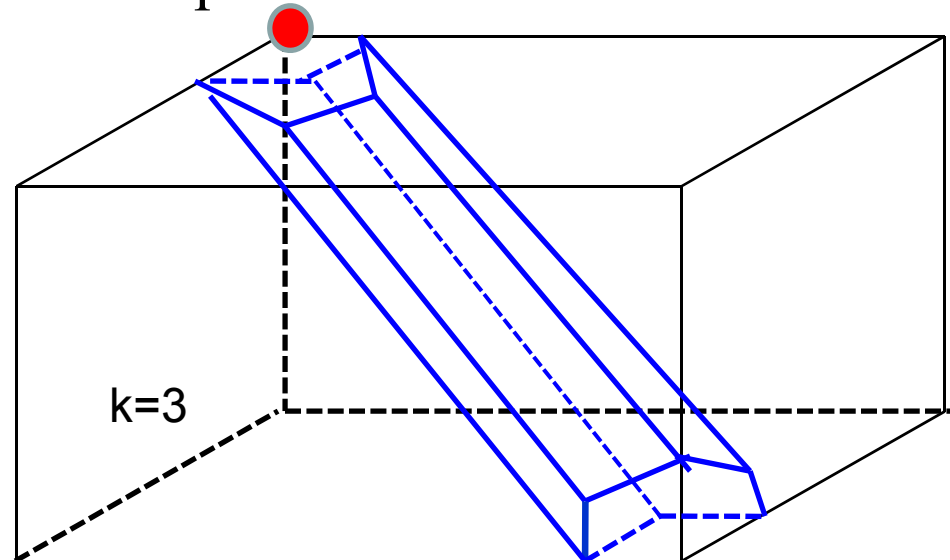
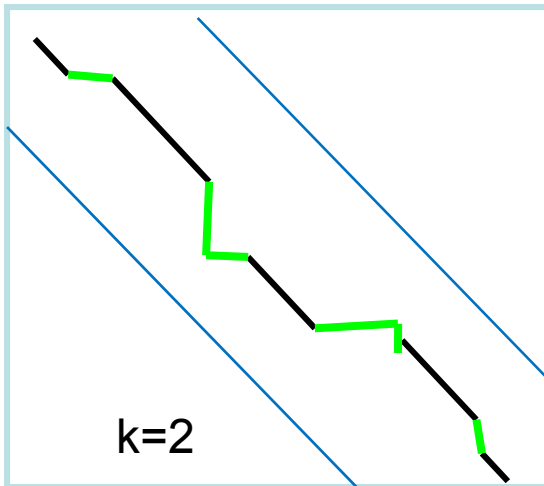
Hence, we must

- fill out all $(m+1)^k$ cells
- for each cell, consider all $2^k - 1$ possibilities.
- for each possibility, we compute the SP score using $O(k^2)$ operations.

Efficiency of the DP method

For k m -character sequences, the computation involves a matrix with $(m+1)^k$ entries; each entry is computed by 2^k arithmetic operations. Hence, the method takes about $O(k^2 2^k (m+1)^k)$ operations.

When $m > 100$, $k > 20$ or $m > 300$ and $k = 5$, aligning by this is impossible even with Carrillio-Lipman bound.



Complexity of MSA

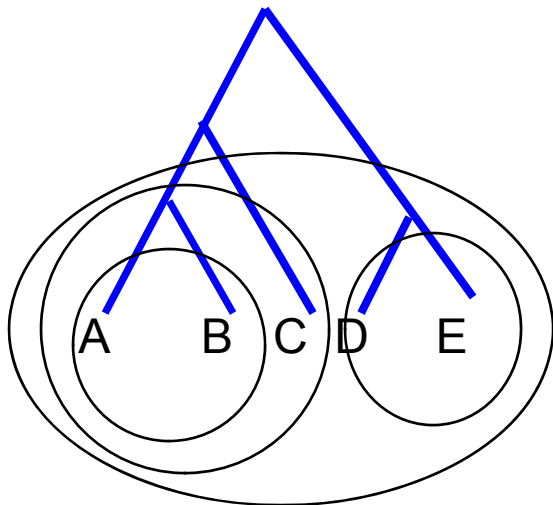
Theorem: MSA is an NP-hard problem.

- This means that **MSA** is likely not solved essentially faster than guessing and checking its solutions.
- It also seems true that **MSA** takes exponential memory space.

4. Progressive alignment paradigm

- It iteratively merges the most similar pairs, often using a clustering tree as guide.

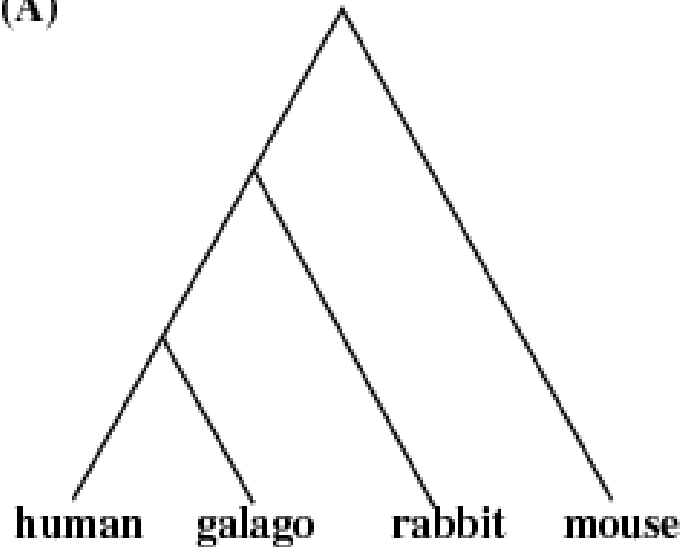
Initially, two closely related sequences are aligned; the resulting alignment is fixed and is used as a building block for the multiple alignment. Then, a third sequence is chosen and aligned to the first alignment, and so on. This process is iterated until all sequences have been aligned.



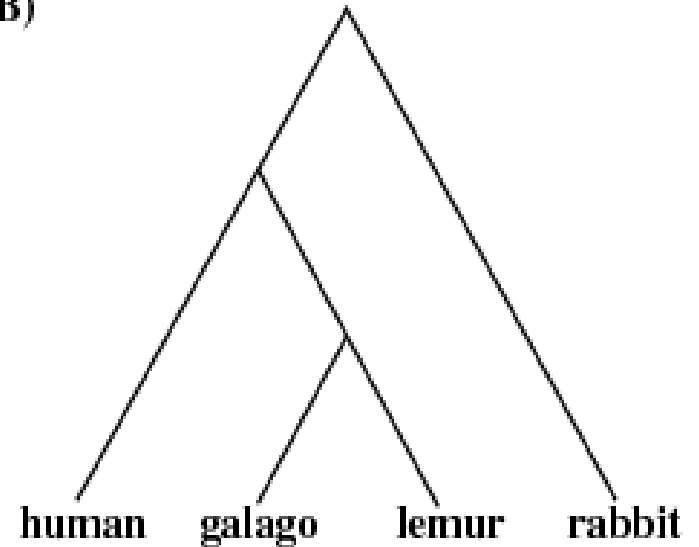
The time for progressive alignment in most cases is roughly the order of the time for computing all pairwise alignment, i.e., $O(k^2n^2)$.

Guiding Tree Examples

(A)



(B)



This heuristic approach is fast. It works well for every close sequences, but does not guarantee an optimal alignment. The problem with such an approach is that it may be misled by some spuriously strong pairwise alignment. If the very first two sequences picked for building multiple alignment are aligned in a way that is incompatible with the optimal multiple alignment, the error in this initial pairwise alignment will propagate all the way through to the whole multiple alignment.

Merge a sequence with a MSA

To add a sequence S into an existing MSA, we pairwise align S with each sequence in the MSA. The resulting pairwise alignment with the highest score is selected as the reference to merge S into the MSA together with the “**once a gap, always a gap**” principle.

S: c g a a a t c

S_1 : a g - a t -
 S_2 : - g a a t c

Step 1: Aligning S and S_1 , we obtain

S_1 : a g a - - t -
 S: c g a a a t c

Step 2: Use S_2 as reference to merge

Aligning S and S_2 , we obtain

S_1 : a g - - a t -
 S_2 : - g - a a t c
 S: c g a a a t c

S_2 : - g - a a t c
 S: c g a a a t c

The alignment with S_2 is better

Once a gap is inserted, it will never be removed

Merge two alignments

To merge two alignments, we align each pair of sequences from two alignments and select one with the highest score as reference to merge the alignments.

S1: a t t g c c a t t - -
S2: a t c - c a a t t t t
 S_3 : a t g g c c a t t
 S_4 : a t c t t c - t t

Step 1: We compute the optimal alignment for S1 and S3, S1 and S4, S2 and S3, and S2 and S4. The resulting highest score alignment is

S_1 : a t t g c c a t t
 S_3 : a t g g c c a t t

Step 2: Merge two given alignments

S2: a t c - c a a t t t t
S1: a t t g c c a t t - -
S3: a t g g c c a t t - -
S4: a t c t t c - t t - -

CLUSTAL Program

INPUT: Sequences S_1, S_2, \dots, S_k .

- (1) Compute all the $\binom{k}{2}$ pairwise alignment scores and convert them into distances.
- (2) Construct a guide tree from pairwise distances using the Neighbor-Joining method.
- (3) Gradually build up the multiple sequence alignment following the order in the guide tree T .

- CLUSTAL: Treat sequences equally.
- CLUSTALW: User can assign weights to sequences and set other program parameters.
- CLUSTALX: provide interface to CLUSTAL.

CLUSTAL Step 1.1

- Use the dynamic programming method to compute pairwise alignments amongst the sequences.
- Convert the pairwise alignment into distances between all pairs of sequences. A simple method is as follows
 1. Count the number $N1$ of match positions
 2. Count the number $N2$ of mismatch positions
 3. Distance = $N2/(N1 + N2)$.

Example:

seq(i): N K L – O N

seq(j): -- M L N O N

There are 1 mismatch and 3 matches and so $D(i, j) = 1/4 = 0.25$

CLUSTAL Step 1.2

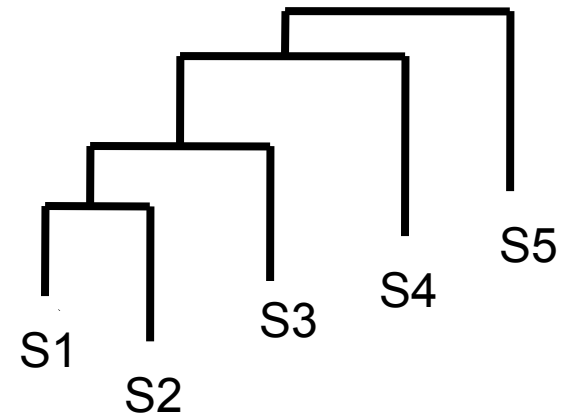
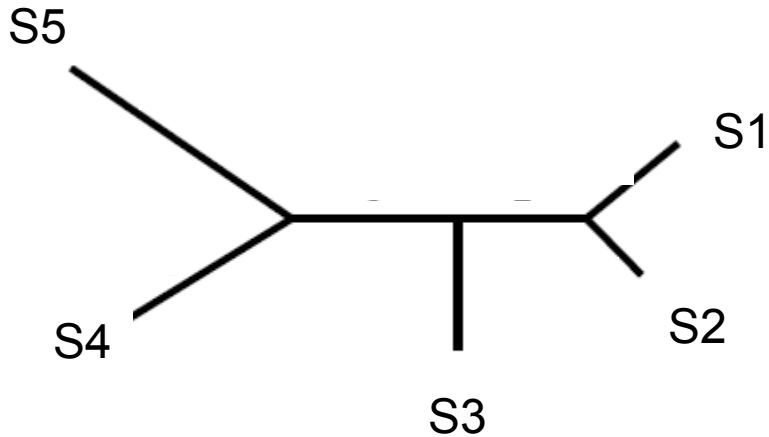
- After the distances are computed for all pairs of sequences, we put them into a distance matrix.

- Example:

	S1	S2	S3	S4	S5
S1					
S2	.14				
S3	.59	.61			
S4	.60	.60	.13		
S5	.69	.69	.72	.71	

CLUSTAL Step 2

- Construct a phylogeny tree with the neighboring-joining method



Length of each branch indicates the degree of sequence divergence

Problems with CLUSTAL

- (a) Optimal alignment may not be found.
- (b) The guide tree is derived from pairwise distances and less reliable.
- (c) When all the sequences are highly divergent (say less than 25-30% identity between any pair of sequences), this progressive approach becomes less reliable.

5. Summary

- The SP scoring scheme
- Generalization of dynamic programming to MSA
- CLUSTALW