Algorithms for multiple sequence alignment

Lecture 6.1

https://phylo.cs.mcgill.ca/play.php

Motivation

- Comparing multiple strings is more than technical exercise – it is a critical cutting-edge tool for extracting important faint commonalities from a set of strings
- We can reveal critical conserved motifs, common 2D and 3D structures which give a clue to a common biological functions (HIV drug)

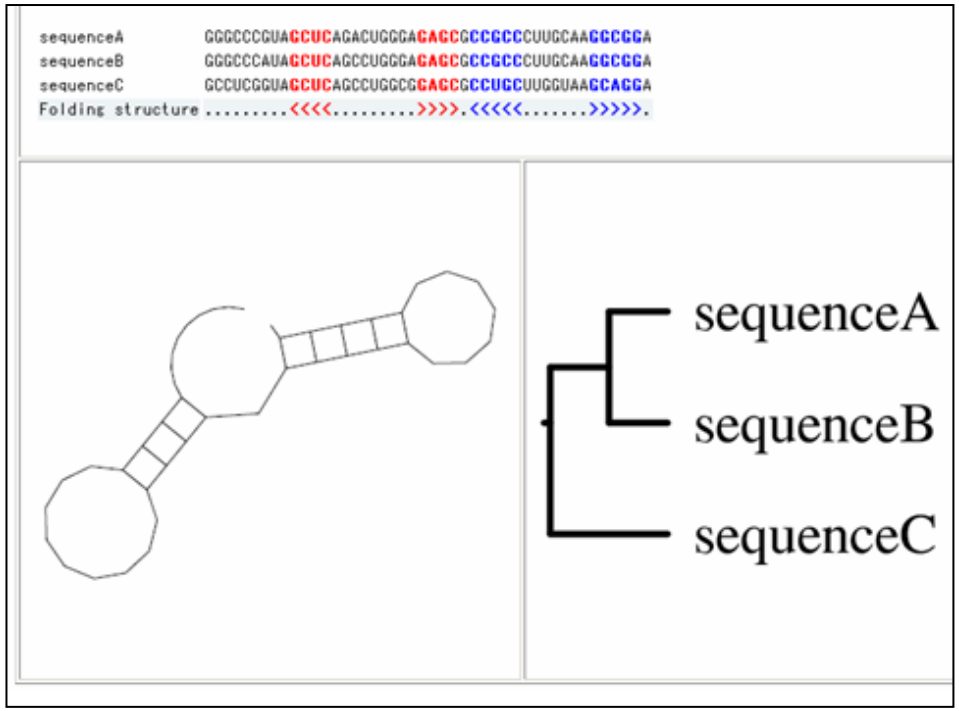
Arthur Lesk: *"One or two homologous sequences whisper. A full multiple alignment shouts out loud."*

Multiple string comparison vs. 2-string comparison

- When we are looking for sequence similar to a given sequence, performing the **pairwise** alignment, we try to discover a **new biological relationship** based on the fact that the **two sequences are similar**
- When we are performing **multiple** alignment, the input **sequences may not be similar**, but they are known to have a **similar biological function** or shape, so we are looking for the similar regions to deduce what is responsible for their common biological function

Example 1: Structure prediction

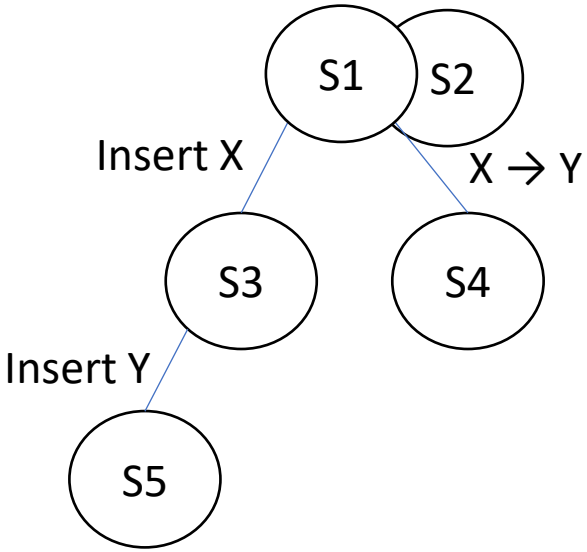
- For proteins with the similar shape or function, compute a multiple alignment and find what regions are conserved between all of them.
- These regions must play important role in defining their common 3D structure (function)



Example 2: Molecular evolution

- Inferring evolutionary relationships between species

S1	A	-	X	-	Z
S2	A	-	X	-	Z
S3	A	-	X	X	Z
S4	A	-	Y	-	Z
S5	A	Y	X	X	Z



Multiple Strings Comparison: inexact matching

- The mutation rate between organisms is high.
- Up to some extent, the changes in DNA do not impact the functionality of the molecule, so all these similar regions we want to find are *inexact* matches

https://en.wikipedia.org/wiki/Phylo_(video_game)

https://phylo.cs.mcgill.ca/play.php

Global Multiple Sequence Alignment (MSA)

- A global multiple alignment for k>2 strings is a table with k rows
- The spaces are inserted in chosen positions of any of the aligned strings, then each string is arrayed in a separate row such that each character and space is in a unique column

S1	A	-	X	-	Z
S2	A	-	X	-	Z
S3	A	-	X	X	Z
S4	A	-	Y	-	Z
S5	A	Y	X	X	Z

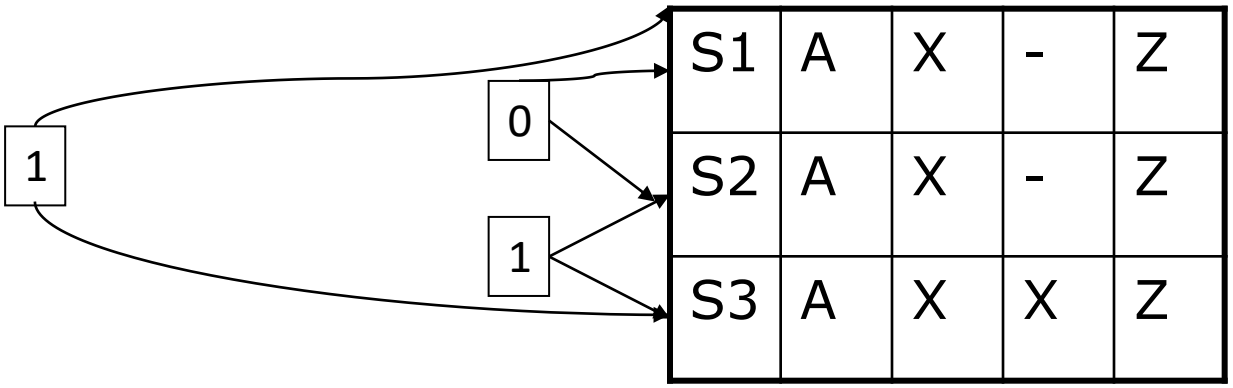
How to score MSA

- Objective score functions:
  - Sum of pairs
  - Consensus
  - Consistency with a tree

- Subjective score function:
  - have an expert to look at the alignment

The sum-of-pairs (SP) score

- The SP score is the sum of scores of pairwise global alignments for each pair of strings in the MSA
- Example: suppose the pairwise alignment scores are edit distances



Total SP-score (edit distance) is 2

The consensus score

S1	A	-	X	-	Z
S2	A	-	X	-	Z
S3	A	-	X	X	Z
S4	A	-	Y	-	Z
S5	A	Y	X	X	Z
S*	A	-	Y	-	Z
	0	1	4	2	0

Consensus string

Consensus score: 7

Consensus score (MSA, S*)=$\Sigma_{\text{all columns } j} \Sigma_{1 \le i \le k}$ score(Si[j],S*[j])

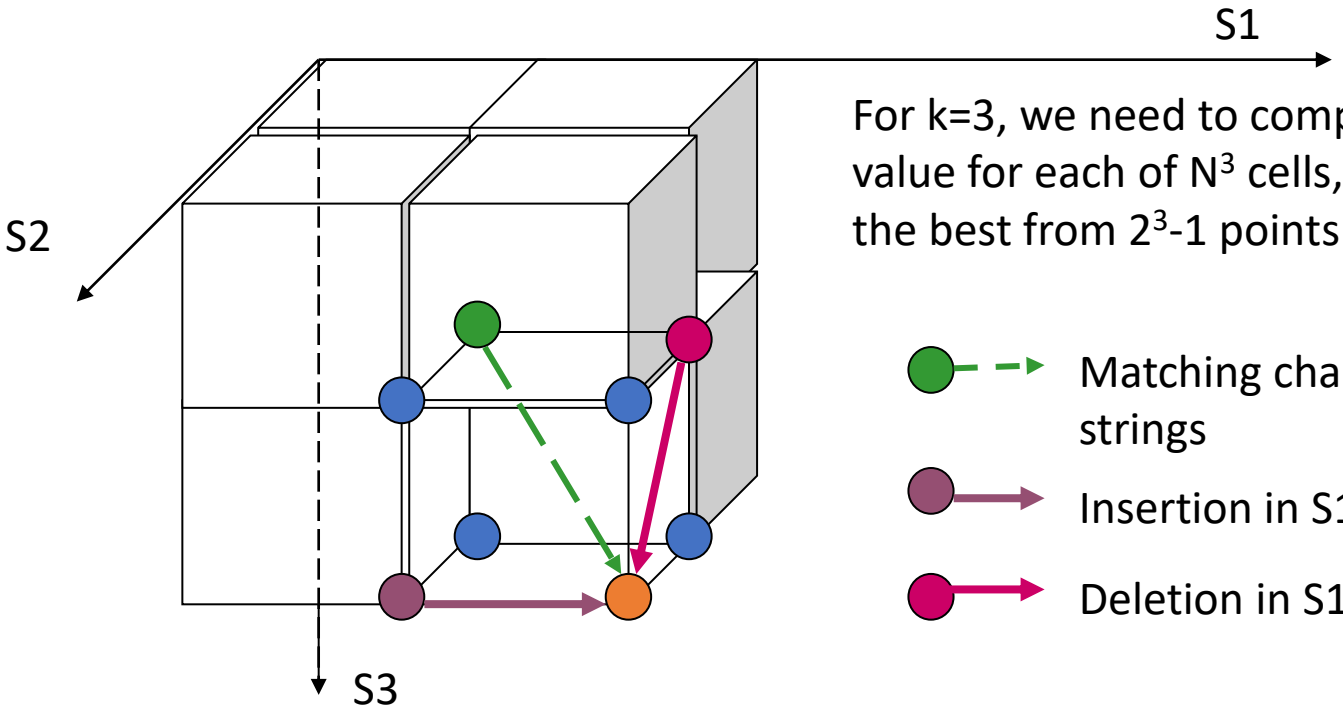
Multiple alignment problem

Given a set *S* of *k* strings and an objective scoring function, compute multiple alignment with an optimal score (minimized or maximized)

- There is no known efficient method for solving this problem for a *consensus score*, so we try to solve it for an *SP-score*

MSA with an SP-score objective function: Dynamic Programming solution

- The solution is analogous to computing an optimal path in a multi-dimensional grid, exactly as for a pairwise alignment in a 2-dimensional grid.



For k=3, we need to compute an optimal value for each of N^3 cells, each time choosing the best from 2^3-1 points

Matching characters of all 3 strings

Insertion in S1

Deletion in S1

The complexity of the DP solution

$O(N^k * 2^k) = O(N^k)$

The problem is NP-complete (See recent paper here)

Heuristic solution: Iterative alignment

- We have 5 strings:
  - S1. AXZ
  - S2. AYZ
  - S3. AXXZ
  - S4. AYXXZ
  - S5. AXZ

- Let us try to add them to an alignment *iteratively*:

Iterative alignment – align S2 to S1

S1. AXZ

S2. AYZ

S3. AXXZ

S4. AYXXZ

S5. AXZ

M (S1,S2)

S1	A	X	Z
S2	A	Y	Z

Iterative alignment – adding S3 to M(S1,S2)

S1. AXZ

S2. AYZ

S3. AXXZ

S4. AYXXZ

S5. AXZ

M (S1,S2,S3)

S1	A	X	-	Z
S2	A	Y	-	Z
S3	A	X	X	Z

Iterative alignment – adding S4 to M(S1,S2,S3)

S1. AXZ

S2. AYZ

S3. AXXZ

S4. AYXXZ

S5. AXZ

Which one is better?

How many different possibilities are for longer strings?

S1	A	-	X	-	Z
S2	A	-	Y	-	Z
S3	A	-	X	X	Z
S4	A	Y	X	X	Z

or

S1	A	X	-	-	Z
S2	A	Y	-	-	Z
S3	A	X	-	X	Z
S4	A	Y	X	X	Z

Iterative alignment – result

S1. AXZ

S2. AYZ

S3. AXXZ

S4. AYXXZ

S5. AXZ

SP score (M)=22

How good is it comparing to an optimal alignment?

How to choose the right order of sequences?

S1	A	X	-	-	Z
S2	A	Y	-	-	Z
S3	A	X	-	X	Z
S4	A	Y	X	X	Z
S5	A	-	X	-	Z

	S2	S3	S4	S5
S1	1	1	3	3
S2		2	2	3
S3			2	3
S4				2

SP-score (M): 22

An approximation algorithm for MSA with an SP-score objective function: SP-star

- Practical methods use *heuristics* to find sub-optimal SP alignment. Little is usually known about how much a produced alignment deviates from the optimal SP alignment.
- A bounded-error *approximation algorithm* is an algorithm which finds a sub-optimal solution, but which allows to evaluate the difference between the computed solution and the optimal solution

SP-star algorithm for MSA

- For this algorithm, the scoring distance must have the following properties:

Property 1. D(S1, S1)=0 **identity**

Property 2. D(S1, S3) <= D(S1, S2) + D (S2, S3)

triangle inequality for strings

(the cost of transforming S1 into S3 is no more than transforming S1 into S2 and then transforming S2 into S3)

Property 3. D(S1, S2)= D(S2, S1) **symmetry**

Edit Distance has these properties

Edit Distance: alternative definition

For each character or gap x in S1 and z in S2:

d(x,z)= 0 if x=z

1 if x!=z

Definition 1. Distance $D(S1,S2)=\sum_{i \text{ from } 1 \text{ to } L}[d(S1_{[i]}, S2_{[j]})]$

Definition 2. Edit distance

ED(S1, S2)=min { D(S1, S2)}

Center Star tree: definitions

Definition 1. Given a set S of k strings, define a center string Sc $\in$ S as a string that minimizes $\sum_{Sj \in S}$ EDistance(Sc, Sj):

$\forall i$ $\quad \sum_{j \text{ from } 1 \text{ to } k}$EDistance(Si, Sj) >= $\sum_{j \text{ from } 1 \text{ to } k}$EDistance(Sc, Sj)

Definition 2. Center start tree - a tree of k nodes with Sc as a center and adjacent nodes – the remaining (k-1) strings of the set.

Produce an alignment Mstar by optimally aligning each string to a center string.

SP-tree algorithm –1/2

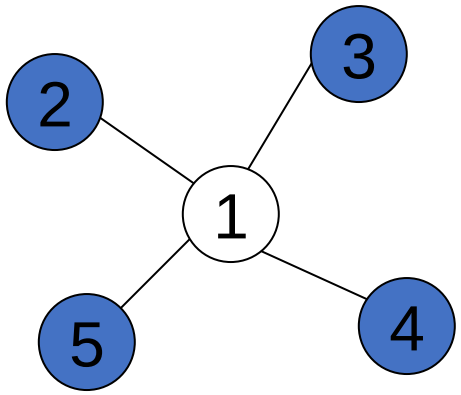
S1. AXZ

S2. AYZ

S3. AXXZ

S4. AYXXZ

S5. AXZ



Distances from each Si to all other strings

	S1	S2	S3	S4	S5	
S1	0	1	1	2	0	4
S2	1	0	2	2	1	6
S3	1	2	0	1	1	5
S4	2	2	1	0	2	7
S5	0	1	1	2	0	4

Computed in time $O(K^2N^2)$

We chose S1 to be a center string Sc

SP-start algorithm – 2/2

S1. AXZ
S2. AYZ
S3. AXXZ
S4. AYXXZ
S5. AXZ

- Align each sequence to Sc according to an edit distance between Sc and every other string

S1	A	-	X	-	Z
S2	A	-	Y	-	Z
S3	A	-	X	X	Z
S4	A	Y	X	X	Z
S5	A	-	X	-	Z

	S2	S3	S4	S5
S1	1	1	2	0
S2		2	3	1
S3			2	1
S4				2

SP score (Mc)=15

Theorem 1.

SP score(Mc)/SP score (M*)<2
Proof (1/3)

For simplicity, let's consider values in all cells of the pairwise distance table. They are directly proportional to SP-score

(1). SP score (Mc)= $\sum_{i=1}^{k} \sum_{j=1}^{k}$ ED(Si, Sj)

(2). ED(Si, Sj) <=ED(Si, Sc)+ED(Sc,Sj)
(triangle inequality)

(3). $\forall$i ED(Si, Sc)=ED (Sc, Si) (symmetry)

(4). From (1) & (2) =>
SP score (Mc)<= $\sum_{i=1}^{k} \sum_{j=1}^{k}$ [ED(Si, Sc)+ED(Sc,Sj)]=
= $\sum_{i=1}^{k} \sum_{j=1}^{k}$ ED(Si, Sc)+ $\sum_{i=1}^{k} \sum_{j=1}^{k}$ ED(Sc, Sj) =
=k $\sum_{j=1}^{k}$ ED(Si, Sc)+ k $\sum_{j=1}^{k}$ ED(Sc, Sj)}=
=2*k $\sum_{j=1}^{k}$ ED(Si, Sc)

	S1	S2	S3	S4	S5
S1	0	1	1	2	0
S2	1	0	2	3	1
S3	1	2	0	2	1
S4	2	3	2	0	2
S5	0	1	1	2	0

Distance table for central star algorithm: total score Mc

SPScore (Mc)<= 2k $\sum_{i=1}^{k}$ ED(Si, Sc) (I)

Theorem 1.

SP score(Mc)/SP score (M*)<2 Proof(2/3)

(5) SP score (M*)= $\sum_{i=1}^{k} \sum_{j=1}^{k} D^*(Si, Sj)$

(6) $\forall$ i $\sum_{j=1}^{k} D(Si, Sj) >= \sum_{j=1}^{k} ED(Sc, Sj)$ (from the choice of Sc to minimize this sum)

(7). From (5) and (6) =>
SP score (M*)>= k* $\sum_{j=1}^{k} ED(Sc, Sj)$

and

1/ SP score (M*) <= k* $\sum_{j=1}^{k} ED(Sc, Sj)$

This is total distance table for optimal (minimal) scores between each pair – the alignment is unknown. Let's call this unknown optimal alignment M*

	S1	S2	S3	S4	S5
S1	0	1	1	2	0
S2	1	0	2	2	1
S3	1	2	0	2	1
S4	2	2	2	0	2
S5	0	1	1	2	0

1/ SP score (M*) <= $\sum_{j=1}^{k} ED(Sc, Sj)$ (II)

Theorem 1.

SP score(Mc)/SP score (M*)<2
Proof (3/3)

(8). From (I) and (II) =>

SP score(Mc)/SP score (M*)<=2 ■

For simplicity, we proved an upper bound which is not tight.

It can be shown that the tighter upper bound is 2(k-1)/k = 2 – 2/k.

Thus, the upper bound for k=3 is 4/3=1.33, for k=4 the upper bound is 1.5 and for k=6 (a problem size considered to be too large for efficient DP solution with strings of length 200) the bound is still only 1.67

How to use this approximation for a better exact solution

- An approximate solution for the SP alignment can be used in order to cut off the number of DP table cells to be computed
- If we estimated the total SP-score to be not more than D, we can consider only the cells in the tunnel with radius not more than D around the main diagonal of the multi-dimensional DP table

MSA implementation: The Carrillo-Lipman algorithm

- The around-the-main diagonal idea is used in the MSA algorithm and its implementation
- It is able to optimally align (on a large server)
  - 20 Phospholipase A2 sequences (approximately 130 residues),
  - 14 Cytochrome C sequences (approximately 110 residues),
  - 6 Aspartal proteases (approximately 350 residues),
  - 8 Lipid binding proteins (approximately 480 residues) on our supercomputers.

All of these problems **approached the limits** of the problems that can be solved optimally by the MSA program, which can compute an optimal multiple alignment for not more than 7 strings of length approximately 200 each

- There is no practical scalable solution to this problem

The meaning of MSA scores in terms of relationships between sequences

- In the SP-score based alignment we try to minimize the total number of edit operations between each pair – but that does no mean that each sequence was transformed into each other sequence by a series of these edit operations

- In the consensus-score based alignment we try to align all sequences to their common ancestor –consensus sequence. The problem is that we cannot find this consensus ancestor by efficient computation

Multiple alignment consistent with a tree

- We optimize distance between more closely related sequences, as follows from the phylogenetic tree for these sequences
- Given an evolutionary phylogenetic tree with a distinct string labeling each leaf, a phylogenetic alignment is an assignment of one string to each internal node
- Each edge represents some mutational history (a series of edit operations), which transformed the ancestor string into its children
- The score of a phylogenetic alignment is the sum of scores of its edges
- Consensus is a phylogenetic alignment to a star-tree
- The problem of constructing a phylogenetic alignment with a minimal total score is NP-complete, and also – the tree topology should be known in advance