Lecture 14: Multiple Sequence Alignment (Gene Finding, Conserved Elements)
2 - 19 - 2015
Scribe: John Ekins

Multiple Sequence Alignment

Given N sequences x^1 , x^2, … , x^N :
Insert gaps in each of the sequences such that:
All sequences have the same length L.
Score of the global map is maximum.

Multiple alignments allow us to explore the protein sequences and other similarities between different organisms. The things that are common must be useful and thus can tell us the function and more details about the proteins and features that are conserved. The best scheme for scoring the best pairs is the sum of pairs scoring method.

Gene Structure

There is a process in which a DNA molecule is transcribed, spliced, and translated into a protein, removing introns and keeping extrons. Those are the instructions for which precise ordering of codons will make up that protein needed. Certain structures will be more prone to mutation than others.

Induced Pairwise Alignment

We take two sequences and remove gaps inserted in the same column. For example:
x: AC-GCGG-C
y: AC-GC-GAG
z: GCCGC-GAG

causes a comparison like this:

x: AC-GCGG-C	y: AC-GC-GAG	x: AC-GCGG-C
y: AC-GC-GAG	z: GCCGC-GAG	z: GCCGC-GAG

The induced pairwise alignment of 2 sequences is obtained by removing gaps and glueing the sequences together.

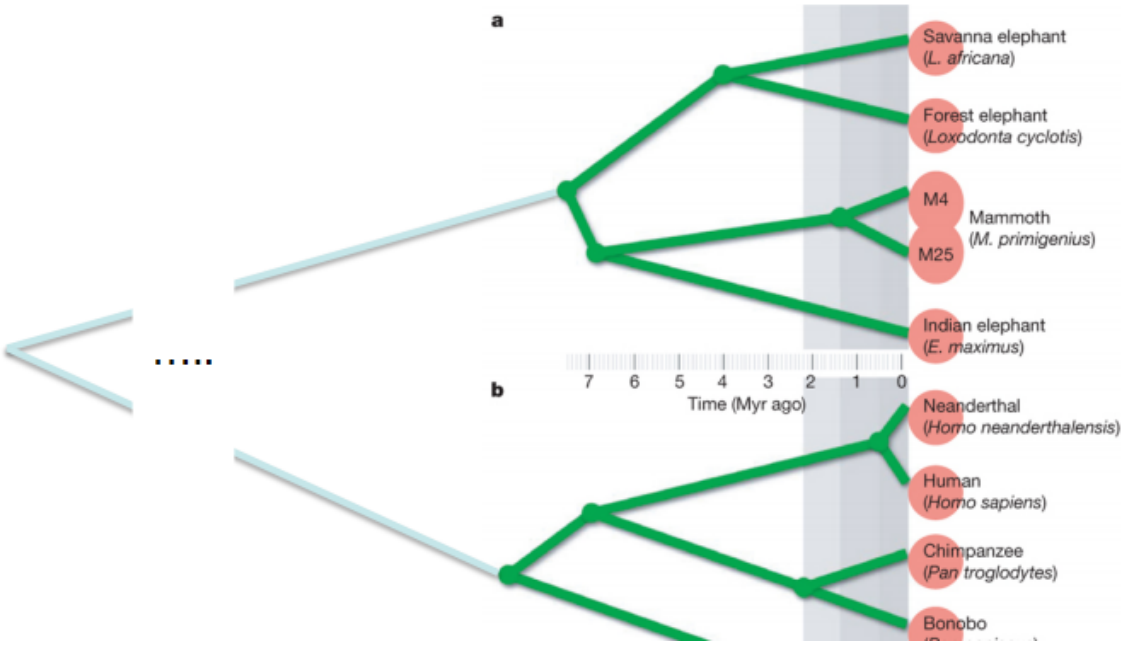
Sum of Pairs

Given n sequences which are multiply aligned, this scoring scheme sums the scores of the induced pairwise alignments between any pair of sequences.

Weighted Sum of Pairs linearly weighted score:

$$S(m) = \Sigma_{k<l} w_{kl} s(m^k, m^l)$$

We would like to weight pairs of sequences in cases where some species are densely populated and other are sparsely populated. For example, if we have a more populated horse branch and a sparsely populated human branch in the diagram shown, we may not want too much redundant information caused by the population density of horse. So we could weigh humans heavier than horses.



Profile Representation

	-	A	G	G	C	T	A	T	C	A	C	C	T	G
	T	A	G	-	C	T	A	C	C	A	-	-	-	G
	C	A	G	-	C	T	A	C	C	A	-	-	-	G
	C	A	G	-	C	T	A	T	C	A	C	-	G	G
	C	A	G	-	C	T	A	T	C	G	C	-	G	G

A	0	1	0	0	0	0	1	0	0	.8	0	0	0	0
C	.6	0	0	0	1	0	0	.4	1	0	.6	.2	0	0
G	0	0	1	.2	0	0	0	0	0	.2	0	0	.4	1
T	.2	0	0	0	0	1	0	.6	0	0	0	0	.2	0
-	.2	0	0	.8	0	0	0	0	0	0	.4	.8	.4	0

With this technique we create a profile representation of a multiple alignment and the probability that the the base pair is a certain letter in that position.

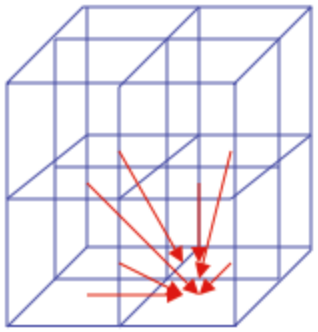
Multidimensional Dynamic Programming

Generalization of Needleman-Wunsh:

$$S(m) = \sum_i S(m_i)$$

(sum of column scores)

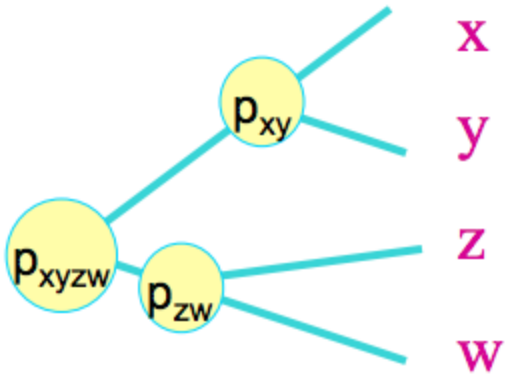
We find the optimal alignment up to a given point by maximizing over its neighbors score. This model only has a gap penalty. The graphic and formula refer to 3 sequences that with 7 neighbors for each cell.



$$
\begin{aligned}
F(i,j,k) = \max\{\ & F(i-1, j-1, k-1) + S(x_i, x_j, x_k), \\
& F(i-1, j-1, k\ \ ) + S(x_i, x_j, -), \\
& F(i-1, j\ \ , k-1) + S(x_i, -, x_k), \\
& F(i-1, j\ \ , k\ \ ) + S(x_i, -, -), \\
& F(i\ \ , j-1, k-1) + S(-, x_j, x_k), \\
& F(i\ \ , j-1, k\ \ ) + S(-, x_j, -), \\
& F(i\ \ , j\ \ , k-1) + S(-, -, x_k)\ \}
\end{aligned}
$$

One drawback is running time at $O(L^N 2^N)$ since we have N sequences of length L and every cell has 2^N-1 neighbors. If we add affine gap scoring running time becomes $O(L^N 4^N)$. This algorithm isn't used much in practice.

Progressive Alignment



We can use an evolutionary tree to compare different species and generate a pairwise alignment at each node. The weights at the node are proportional to the divergence in the corresponding edges.

The algorithm goes like this:

Align closest nodes first, in order of tree

In each step, align two sequences to generate a new alignment and associated profile

Heres an example problem:

Example

Profile: (A, C, G, T, -)
p_x = (0.8, 0.2, 0, 0, 0)
p_y = (0.6, 0, 0, 0, 0.4)

$s(p_x, p_y)$ = 0.8*0.6*s(A, A) + 0.2*0.6*s(C, A)
+ 0.8*0.4*s(A, -) + 0.2*0.4*s(C, -)

Result: p_{xy} = (0.7, 0.1, 0, 0, 0.2)

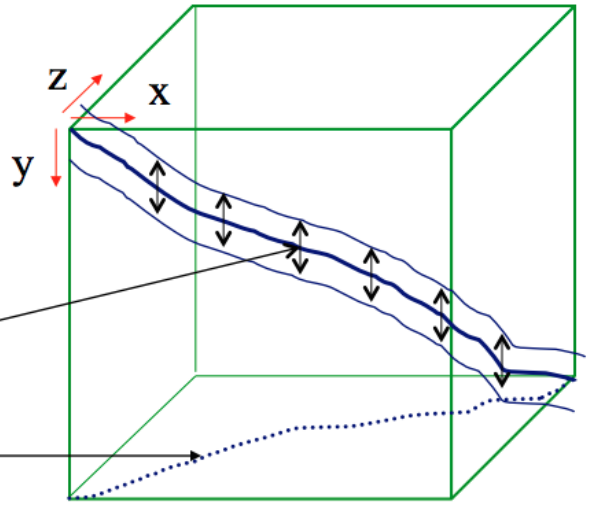
$s(p_x, -)$ = 0.8*1.0*s(A, -) + 0.2*1.0*s(C, -)

Result: p_{x-} = (0.4, 0.1, 0, 0, 0.5)

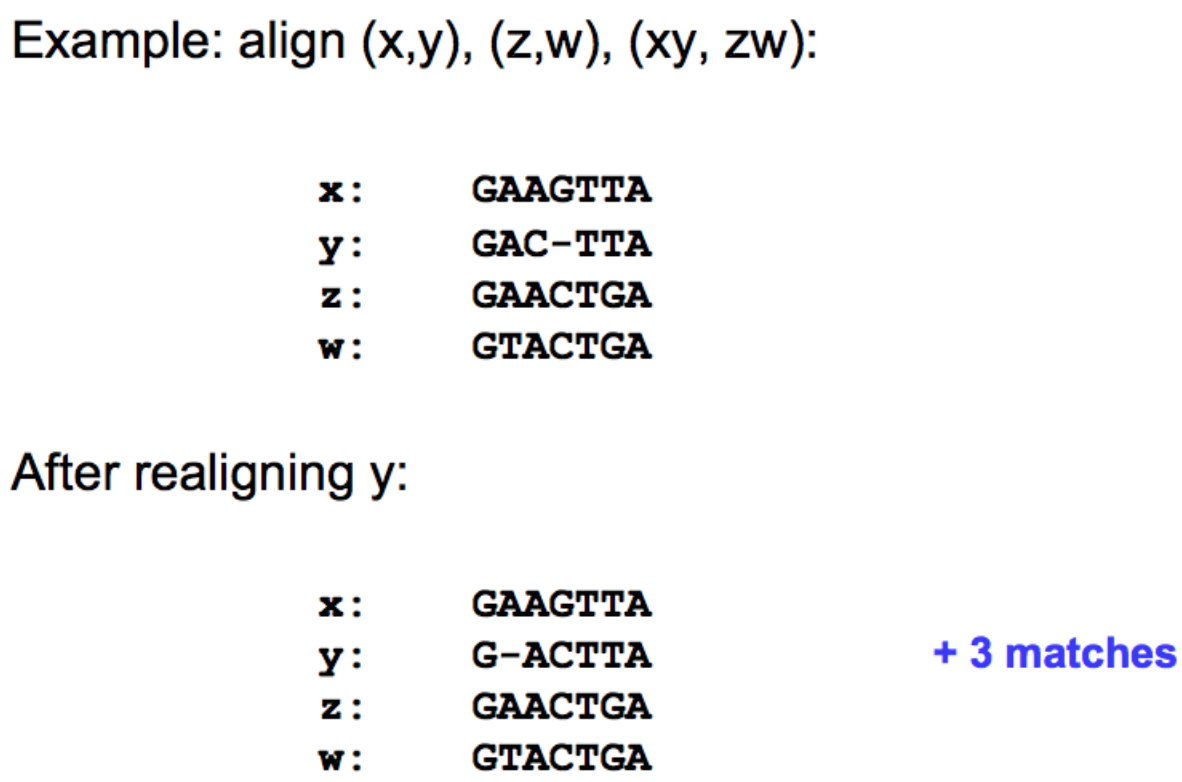
Iterative Refinement

Algorithm (Barton-Stenberg):

1. For j = 1 to N,
   Remove x^j, and realign to $x^1 \ldots x^{j-1}x^{j+1}\ldots x^N$
2. Repeat 4 until convergence



The Barton-Stenberg algorithm iteratively removes a sequence and realigns it to the rest of the sequences. We reduce time complexity by allowing the sequence to vary in an arbitrary band around sequence. Example alignment:



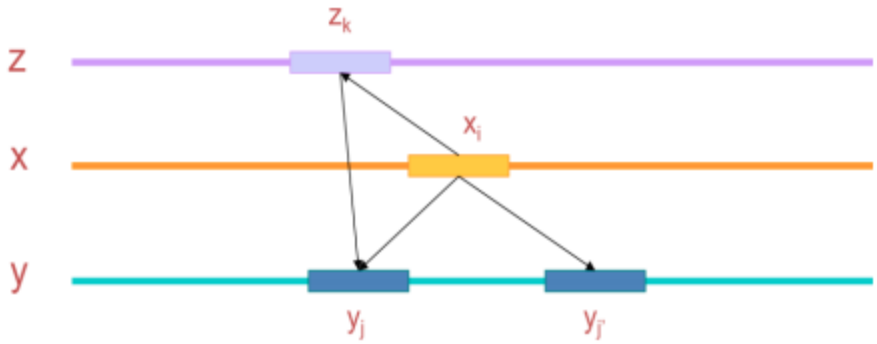
Here is an example where iterative alignment doesn't solve the multiple alignment problem because removing any single y_i doesn't change the output.

Example not handled well:

```
x:    GAAGTTA
y1:   GAC-TTA
y2:   GAC-TTA
y3:   GAC-TTA

z:    GAACTGA
w:    GTACTGA
```

Consistency



Considering triplets for Consistency technique

The consistency technique allows us to find even more accurate alignments by avoiding making bad choices in pairwise alignment between two sequences with the help of information from other multiple sequences.

Basic method for applying consistency

- Compute all pairs of alignments xy, xz, yz, …
- When aligning x, y during progressive alignment,
  - For each (x_i, y_j), let $s(x_i, y_j) = function\_of(x_i, y_j, a_{xz}, a_{yz})$
  - Align x and y with DP using the modified s(.,.) function

Multiple Alignment Programs

Genome Resources

Annotation and alignment genome browser at UCSC
http://genome.ucsc.edu/cgi-bin/hgGateway

Specialized VISTA alignment browser at
LBNL
http://pipeline.lbl.gov/cgi-bin/gateway2

ABC—Nice Stanford tool for browsing alignments
http://encode.stanford.edu/~asimenos/ABC/

Protein Multiple Aligners

CLUSTALW – most widely used
http://www.ebi.ac.uk/clustalw/

MUSCLE – most scalable
http://phylogenomics.berkeley.edu/cgi-bin/muscle/input_muscle.py

PROBCONS – most accurate
http://probcons.stanford.edu/