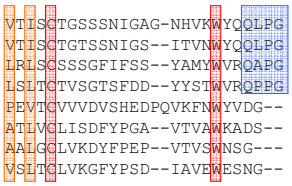
Multiple Alignment

Multiple Sequence Alignment (MSA)

- An MSA of these sequences:



Conserved residues, regions, patterns

Profile Representation of Multiple Alignment

```
          -  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         1                 1        .8
     C      .6                1        .4  1     .6 .2
     G               1 .2                    .2        .4  1
     T      .2                      1     .6              .2
     -      .2        .8                          .4 .8 .4
```

Earlier, we were aligning a **sequence against a sequence**

Can we align a **sequence against a profile?**

Can we align a **profile (i.e., alignment) against another profile (i.e., alignment)?**

Consensus String of a Multiple Alignment

```
          -  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

          C  A  G  -  C  T  A  T  C  A  C  -  G  G
Consensus
String:   C  A  G  C  T  A  T  C  A  C  G  G
```

- The *consensus string* S_M derived from multiple alignment *M* is the concatenation of the consensus characters for each column of *M*.
  - The *consensus character* for column *i* is the character that minimizes the summed distance to it from all the characters in column *i.* (i.e., if match and mismatch scores are equal for all symbols, the majority symbol is the consensus character)

Star alignment

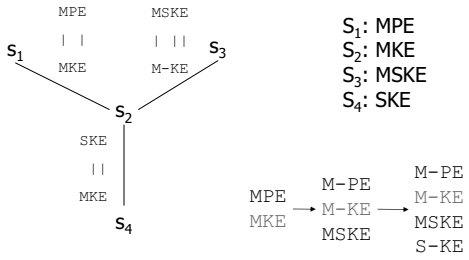
- Heuristic method for multiple sequence alignments
- Select a sequence *c* as the center of the star
- For each sequence $x_1, \ldots, x_k$ such that index $i \neq c$, perform a Needleman-Wunsch global alignment
- Aggregate alignments with the principle "once a gap, always a gap."

Choosing a center

- Try them all and pick the one which is most similar to all of the sequences
- Let $S(x_i,x_j)$ be the optimal score between sequences x_i and x_j.
- Calculate all $O(k^2)$ alignments, and choose as x_c the sequence x_i that maximizes the following

$$\sum_{j \neq i} S(x_i,x_j)$$

Star alignment example



S_1: MPE
S_2: MKE
S_3: MSKE
S_4: SKE

ClustalW

- Popular multiple alignment tool today
- 'W' stands for 'weighted' (different parts of alignment are weighted differently).
- Three-step process
  1.) Construct pairwise alignments
  2.) Build Guide Tree (by Neighbor Joining method)
  3.) Progressive Alignment guided by the tree
  - The sequences are aligned progressively according to the branching order in the guide tree

Step 1: Pairwise Alignment

- Aligns each sequence again each other giving a similarity matrix
- Similarity = exact matches / sequence length (percent identity)

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–

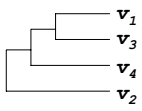
(.17 means 17 % identical)

Step 2: Guide Tree

- Create Guide Tree using the similarity matrix
  - ClustalW uses the neighbor-joining method
  - Guide tree roughly reflects evolutionary relations

Step 2: Guide Tree (cont'd)

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–



Calculate:

$v_{1,3}$ = alignment (v_1, v_3)
$v_{1,3,4}$ = alignment $((v_{1,3}), v_4)$
$v_{1,2,3,4}$ = alignment $((v_{1,3,4}), v_2)$

Step 3: Progressive Alignment

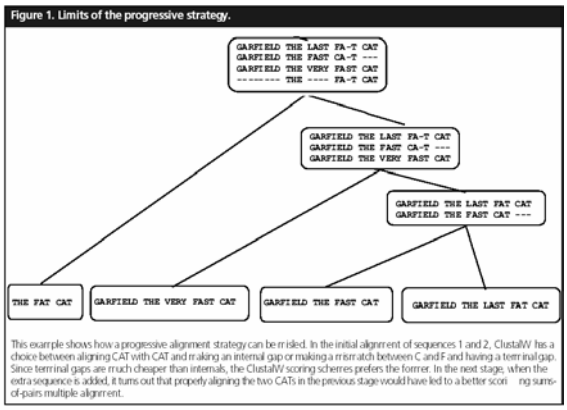
- Start by aligning the two most similar sequences
- Following the guide tree, add in the next sequences, aligning to the existing alignment
- Insert gaps as necessary

```
FOS_RAT          PEEMSVTS-LDLTGGLPEATTPESEEAFTLPLLNDPEPK-PSLEPVKNISNMELKAEPFD
FOS_MOUSE        PEEMSVAS-LDLTGGLPEASTPESEEAFTLPLLNDPEPK-PSLEPVKSISNVELKAEPFD
FOS_CHICK        SEELAAATALDLG----APSPAAAEEAFALPLMTEAPPAVPPKEPSG--SGLELKAEPFD
FOSB_MOUSE       PGPGPLAEVRDLPG-----STSAKEDGFGWLLPPPPPPP-----------------LPFQ
FOSB_HUMAN       PGPGPLAEVRDLPG-----SAPAKEDGFSWLLPPPPPPP-----------------LPFQ
                 .  . :  ** .     :.. *:.*  *  . *                    **:
```

Dots and stars show how well-conserved a column is.

Problems with progressive alignments

- Depend on pairwise alignments
- If sequences are very distantly related, much higher likelihood of errors
- Care must be made in choosing scoring matrices and penalties



Multiple Alignments: Scoring

- Entropy score
- Sum of pairs (SP-Score)

Entropy of an Alignment: Example

column entropy:

$-(p_A \log p_A + p_C \log p_C + p_G \log p_G + p_T \log p_T)$

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

- Column 1 = -[1*log(1) + 0*log0 + 0*log0 +0*log0]
  = 0
- Column 2 = -[(1/4)*log(1/4) + (3/4)*log(3/4) + 0*log0 + 0*log0]
  = -[(1/4)*(-2) + (3/4)*(-.415)] = +0.811
- Column 3 = -[(1/4)*log(1/4)+(1/4)*log(1/4)+(1/4)*log(1/4) +(1/4)*log(1/4)]
  = 4* -[(1/4)*(-2)] = +2.0
- Alignment Entropy = 0 + 0.811 + 2.0 = +2.811

Sum of Pairs Score(SP-Score)

- Consider pairwise alignment of sequences
  a_i and a_j
  imposed by a multiple alignment of k sequences
- Denote the score of this suboptimal (not necessarily optimal) pairwise alignment as
  $s^*(a_i, a_j)$
- Sum up the pairwise scores for a multiple alignment:
  $s(a_1,\ldots,a_k) = \Sigma_{i,j}\, s^*(a_i, a_j)$

Computing SP-Score

Aligning 4 sequences: 6 pairwise alignments

Given a_1,a_2,a_3,a_4:

$$s(a_1 \ldots a_4) = \Sigma s^*(a_i,a_j) = s^*(a_1,a_2) + s^*(a_1,a_3) + s^*(a_1,a_4) + s^*(a_2,a_3) + s^*(a_2,a_4) + s^*(a_3,a_4)$$

Example

- Compute Sum of Pairs Score of the following multiple alignment with match = 3, mismatch = -1, S(X,-) = -1, S(-,-) = 0

```
X: G T A C G
Y: T G C C G
Z: C G G C C
W: C G G A C
  -2 6-2 6 2
```

Sum of pairs = -2+6-2+6+2 = 10

Review of previous weeks

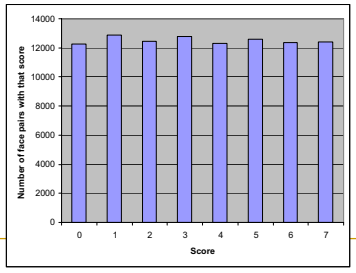
- Pairwise sequence alignment
  - Scoring matrices
    - PAM, BLOSUM, log-odds score
  - Dynamic programming
    - Needleman-Wunsch (Global)
    - Semi-global (no end-gap penalties)
    - Smith-Waterman (Local)

Review of previous weeks

- Statistical significance of alignments
  - p-value, E-value, z-score?
  - Computing significance using random samples.
  - Example:
    - Suppose we have a face matching algorithm and we assign scores based on match of complexion, eye-color, hair-color, use of glasses, use of face jewellery, existence of mustache, beard.

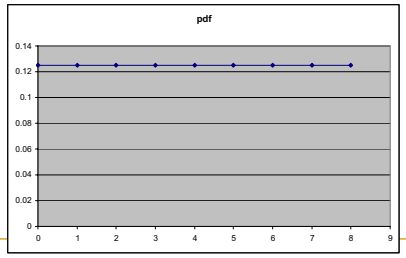
Statistical significance

- We match 100K pairs of faces randomly and here's the (hypothetical) score distribution:



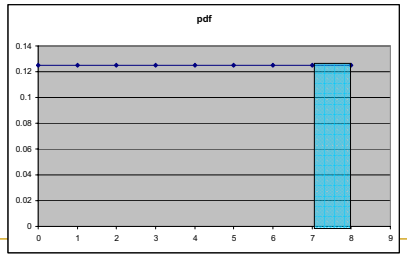
Statistical significance

- What is the p-value of a match with score 7? score 1?



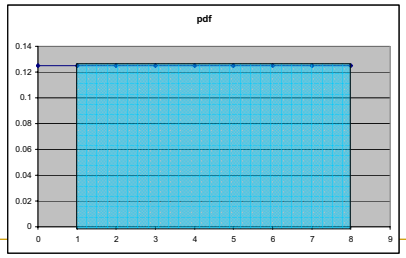
Statistical significance

- p-value(7) = 0.125



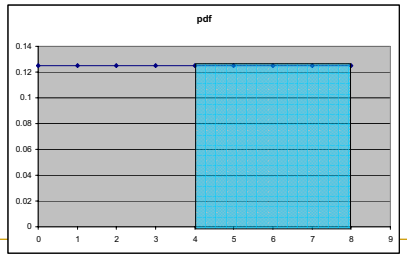
Statistical significance

- p-value(1) = 0.875



Statistical significance

- p-value(4) = 0.5



Suffix Trees and Suffix Arrays

- Construction of suffix trees and suffix arrays
- Pattern search in suffix trees/arrays
- Other applications
  - E.g.,
    - Finding the most occurring pattern of length 2 in a string (Solution: count the # of leaves below the nodes that have string depth>=2)