9 Multiple Sequence Alignment

This exposition is based on the following sources, which are recommended reading:

1. D. Mount: *Bioinformatics*. CSHL Press, 2004, chapter 5.
2. D. Gusfield: *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997, chapter 14.

9.1 Introduction

Two facts of biological sequence comparison:

1. High sequence similarity $\overset{\text{usually}}{\longrightarrow}$ significant functional/structural similarity
2. Evolutionary/functionally related sequences can differ significantly at the primary sequence level.[1]

9.2 Introduction

What is a multiple sequence alignment? A multiple sequence alignment is simply an alignment of more than two sequences, like this:

```
SRC_RSVP   -FPIKWTAPEAALY---GRFTIKSDVWSFGILLTELTTKGRVPYPGMVNR-EVLDQVERG
YES_AVISY  -FPIKWTAPEAALY---GRFTIKSDVWSFGILLTELVTKGRVPYPGMVNR-EVLEQVERG
ABL_MLVAB  -FPIKWTAPESLAY---NKFSIKSDVWAFGVLLWEIATYGMSPYPGIDLS-QVYELLEKD
FES_FSVGA  QVPVKWTAPEALNY---GRYSSESDVWSFGILLWETFSLGASPYPNLSNQ-QTREFVEKG
FPS_FUJSV  QIPVKWTAPEALNY---GWYSSESDVWSFGILLWEAFSLGAVPYANLSNQ-QTREAIEQG
KRAF_MSV36 TGSVLWMAPEVIRMQDDNPFSFQSDVYSYGIVLYELMA-GELPYAHINNRDQIIFMVGRG
```

(A small section of six tyrosine kinase protein sequences.)

In this example multiple sequence alignment is applied to a set of sequences that are assumed to be homologous (have a common ancestor sequence) and the goal is to detect homologous residues and place them in the same column of the multiple alignment.

However, this is by far not the only use. While multiple sequence alignment (MSA) is a straightforward generalization of pairwise sequence alignment, there are lots of new questions about scoring, the significance of scores, gap penalties, and efficient implementations.

Definition.
Assume we are given k sequences $x_1, \ldots, x_k$ over an alphabet Σ.

Let - $\notin \Sigma$ be the *gap symbol*. Let $h\colon (\Sigma \cup \{\text{-}\})^* \to \Sigma^*$ be the mapping that removes all gap symbols from a sequence over the alphabet $\Sigma \cup \{\text{-}\}$. For example, $h(\texttt{-FPIKWTAPEAALY---GRFT}) = \texttt{FPIKWTAPEAALYGRFT}$.

Then a *global alignment* of $x_1, \ldots, x_k$ consists of k sequences $x'_1, \ldots, x'_k$ over the alphabet $\Sigma \cup \{\text{-}\}$ such that

[1] basically, this says that the reverse of 1. is *not* true in general.

- $h(x'_i) = x_i$ for all i,
- $|x'_i| = |x'_j|$ for all i, j, and
- $(x'_{1,p}, \ldots, x'_{k,p}) \neq (-, \ldots, -)$ for all p.

Example.

```
 x1  =  G C T G A T A T A G C T
 x2  =  G G G T G A T T A G C T
 x3  =  G C T A T C G C
 x4  =  A G C G G A A C A C C T

x'1  =  - G C T G A T A T A G C T
x'2  =  G G G T G A T - T A G C T
x'3  =  - G C T - A T - - C G C -
x'4  =  A G C G G A - A C A C C T
```

Below we give a list of the most common uses of multiple sequence alignment.

- detecting faint similarities in sequences that are not detected by pairwise sequence comparison.
- detecting structural homologies.
- grouping proteins into families.
- computing the consensus sequence in assembly projects.
- inferring evolutionary trees.
- and more ...

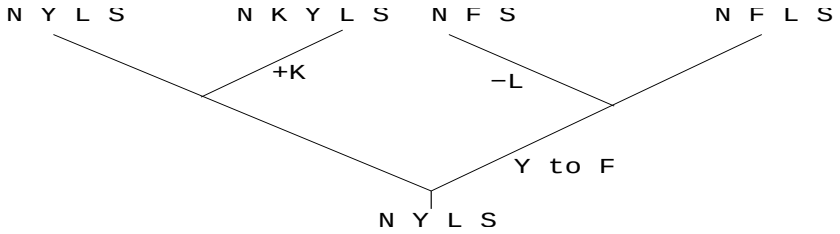
We now give some more details about the different uses.

9.3 MSA and evolutionary trees

One main application of multiple sequence alignment lies in phylogenetic analysis. Given an MSA:

$$
\begin{array}{lccccc}
a_1 = & \mathtt{N} & \mathtt{-} & \mathtt{F} & \mathtt{L} & \mathtt{S} \\
a_2 = & \mathtt{N} & \mathtt{-} & \mathtt{F} & \mathtt{-} & \mathtt{S} \\
a_3 = & \mathtt{N} & \mathtt{K} & \mathtt{Y} & \mathtt{L} & \mathtt{S} \\
a_4 = & \mathtt{N} & \mathtt{-} & \mathtt{Y} & \mathtt{L} & \mathtt{S}
\end{array}
$$

We would like to reconstruct the evolutionary tree that gave rise to these sequences, e.g.:



Before we can apply algorithms for phylogenetic tree reconstruction we need to find out how the positions of the sequences correspond to each other.

9.4 Protein families

Assume we have established a family $s_1, s_2, \ldots, s_r$ of homologous protein sequences. Does a new sequence s_0 belong to the family?

One method of answering this question would be to align s_0 to each of $s_1, \ldots, s_r$ in turn. If one of these alignments produces a high score, then we may decide that s_0 belongs to the family.

However, perhaps s_0 does not align particularly well to any one specific family member, but does well in a multiple alignment, due to common motifs etc.

9.5 Sequence Assembly

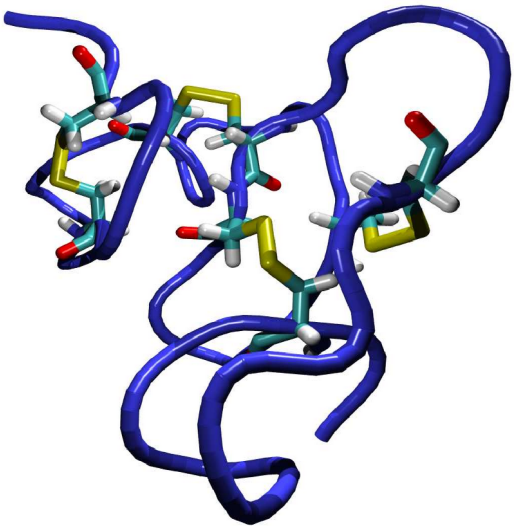
Assume we are given a layout of several genomic reads in a sequencing project that were produced using the shotgun sequencing method. These fragments will be highly similar, and hence easy to align. Nevertheless we want to do this with great speed and accuracy:

```
f1  ACCACAACCCTGCATGGGGCAT-ATTTGGCCTAGCT
f2                  AGGGCCTTATATG-GCTAGCT-CGTTCCCGGGCATGGC
f3              GCATGGGGCATTATCTGGCCTAGCT--GAT
f4                                      CCGTTCCCGG-CTTGGCAACG
============================================================
cns ACCACAACCCTGCATGGGGCATTATCTGGCCTAGCT-CGTTCCCGGGCATGGCAACG
```

9.6 Conservation of structural elements

The below figure shows the alignment of N-acetylglucosamine-binding proteins and the tertiary structure of one of them, the hevein.

```
AATAHAQRCG EQGSNMECPN NLCCSQYGYC GMGGDYCGKG ..CQNGACYT
VAATNAQTCG KQNDGMICPH NLCCSQFGYC GLGRDYCGTG ..CQSGACCS
VGLVSAQRCG SQGGGGTCPA LWCCSIWGWC GDSEPYCGRT ..CENK.CWS
AATAQAQRCG EQGSNMECPN NLCCSQYGYC GMGGDYCGKG ..CQNGACWT
AATAQAQRCG EQGSNMECPN NLCCSQYGYC GMGGDYCGKG ..CQNGACWT
......QRCG EQGSGMECPN NLCCSQYGYC GMGGDYCGKG ..CQNGACWT
SETVKSQNCG .......CAP NLCCSQFGYC GSTDAYCGTG ..CRSGPCRS
RGSAE..QCG RQAGDALCPG GLCCSSYGWC GTTVDYCGIG ..CQSQ.CDG
AGPAAAQNCG .......CQP NFCCSKFGYC GTTDAYCGDG ..CQSGPCRS
AGPAAAQNCG .......CQP NVCCSKFGYC GTTDEYCGDG ..CQSGPCRS
RGSAE..QCG QQAGDALCPG GLCCSSYGWC GTTADYCGDG ..CQSQ.CDG
RGSAE..QCG RQAGDALCPG GLCCSFYGWC GTTVDYCGDG ..CQSQ.CDG
TGVAIAEQCG RQAGGKLCPN NLCCSQWGWC GSTDEYCSPD HNCQSN.CK.
......EQCG RQAGGKLCPN NLCCSQYGWC GSSDDYCSPS KNCQSN.CK.
```



The example exhibits 8 cysteins that form 4 disulphid bridges and are an essential structural part of those proteins.

9.7 Scoring schemas

What is the quality, or the score of a multiple alignment? There are many possible alignments and the natural question is of course which one is the best under some scoring scheme that resembles the relevant biological question best.

Generally one can define similarity measures or distance measures as it is the case for pairwise alignment. In what follows we will use interchangeably *distance* measures, for which we try to minimize the *cost*, and *similarity* measures for which we try to maximize the *score*.

9.8 Projection of a multiple alignment

We will also refer to a multiple alignment $x'_1, \ldots, x'_k$ of k sequences $x_1, \ldots, x_k$ by a $k \times l$-matrix A with

$$l = |x'_1| \qquad (= |x'_2| = |x'_2| = \ldots = |x'_k|) \ ,$$

where $A[i][j]$ contains the j-th symbol of x'_i.

In what follows we need the definition of multiple alignments of subsets of the k strings.

Definition. Let A be a multiple alignment for the k strings $x_1, \ldots, x_k$ and let $I \subseteq \{1, \ldots, k\}$ be a set of indices defining a subset of the k strings. Then we define A_I as the alignment resulting from first removing all rows $i \notin I$ from A and then deleting all columns consisting entirely of blanks. We call A_I the *projection* of A to I. If I is given explicitly, we simplify notation and write, e.g., $A_{i,j,k}$ instead of $A_{\{i,j,k\}}$.

Example.

```
a1 = - G C T G A T A T A G C T
a2 = G G G T G A T - T A G C T
a3 = - G C T - A T - - C G C -
```

```
a4 = A G C G G A - A C A C C T
```

The projection $A_{2,3}$ is given by first taking the second and third row of the alignment:

```
a2 = G G G T G A T - T A G C T
a3 = - G C T - A T - - C G C -
```

and then deleting the column consisting only of blanks.

```
a2 = G G G T G A T T A G C T
a3 = - G C T - A T - C G C -
```

9.9 Cost functions

Let $c : \mathcal{A} \to \mathbb{R}$ be a function that maps each possible alignment in the set of all alignments $\mathcal{A}$ to a real number. Our goal is to find an optimal multiple alignment A^*, that is,

$$A^* = \arg\min_{A \in \mathcal{A}} c(A) \qquad \text{or} \qquad A^* = \arg\max_{A \in \mathcal{A}} c(A) \ ,$$

depending on whether we maximize similarity or minimize a distance.

In the following we describe a few common cost functions, namely

- consensus score
- profile score
- weighted sum of pairs (WSOP) score (with linear gap costs)
- phylogenetic score

9.10 Consensus alignment

Let $A[\,][j]$ be any column of a multiple alignment A. Then the letter x_j is called the j-th *consensus*-letter, if the *consensus*-error

$$\sum_{i=1}^{k} d(x_j, A[i][j])$$

is minimal. The concatenation of the consensus letters yields the *consensus*-string. Hence the goal is to find an alignment A^* that minimizes the consensus error summed over all columns.

The cost $c(A)$ is then defined as follows:

$$c(A) = \sum_{j=1}^{l} \sum_{i=1}^{k} d(x_j, A[i][j]) \qquad x_j \text{ is the } j\text{-th consensus letter}$$

Example.

Let $d(x,y) = \begin{cases} 2 & \text{for } x \neq y, \quad (x,y) \in \Sigma \times \Sigma \\ 1.5 & \text{for } x = - \text{ or } y = -, \text{ but } (x,y) \neq (-,-) \\ 0 & \text{otherwise .} \end{cases}$

```
              a1 =  -   G   C   T   G   A   T   A   T   A   A   C   T
              a2 =  G   G   G   T   G   A   T   -   T   A   G   C   T
              a3 =  A   G   C   G   G   A   -   A   C   A   C   C   T
                    ---------------------------------------------------
consensus   :       -   G   C   T   G   A   T   A   T   A   X   C   T
column value:       3   0   2   2   0   0  1.5 1.5  2   0   4   0   0 = 16
```

An *X* in the consensus string symbolizes that the consensus can be any letter (occuring in the corresponding column).

9.11 Profile alignment

Profiles are another way to describe a motif common to a family of sequences. Given a multiple alignment of a set of strings, a *profile* is obtained by recording the frequency of each character (including the blank) for each column.

Aligning a string S to a profile A accounts to computing the weighted sum of the scores of the letters of S to the columns of the profile A. The alignment algorithm is very similar to the dynamic programming for two sequences.

For a character y and a column j, let $p(y, j)$ be the frequency of character y in column j. Then the *score* of aligning a character x with column j is

$$\sum_y p(y, j)s(x, y)\,.$$

Note that here we need an entry for $s(-, -)$ in the scoring matrix.

Profiles are also often called *position specific scoring matrices* (PSSM) in the biological literature.

Example.
We align AABBC against a profile for 4 sequences using a similarity score.

```
a1 = A B C - A       profile:  c1   c2   c3   c4   c5     score:   A   B   C   -
a2 = A B A B A                A: .75       .25       .50           A  2  -1  -3  -1
a3 = A C C B -                B:      .75       .75               B -1   2  -2  -1
a4 = C B - B C                C: .25  .25  .50       .25           C -3  -2   2  -1
                              -:           .25  .25  .25           - -1  -1  -1   1

-------------------------------------------------------------
   column   |    column value calculation             |   sum
  seq  prof |    A        B        C        -         |

   A    1   =   +0.75*2           -0.25*3             =   0.75
   A    -   =   -1.0 *1                               =  -1.0
   B    2   =            +0.75*2  -0.25*2             =   1.0
   -    3   =   -0.25*1           -0.50*1  +0.25*1    =  -0.5
   B    4   =            +0.75*2           -0.25*1    =   1.25
   C    5   =   -0.5*3            +0.25*2  -0.25*1    =  -1.25
                                                       -------
                                                         0.25
```

9.12 WSOP score

One of the most common scoring functions for MSA is the *(weighted) sum of pairs*, which is defined as the (weighted) sum of the score of all pairwise projections of the MSA, that is,

$$c(A) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} w_{i,j} \cdot c(A_{i,j})$$

Each pair (i, j) can be given a different *weight* $w_{i,j}$. Note that $c(A_{i,j})$ involves another summation over the columns of $A_{i,j}$. If we assume independence of the score of the alignment columns then we can rewrite this as follows:

$$c(A) = \sum_{h=1}^{l} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} w_{i,j} \cdot s(A[i,h], A[j,h])$$

The nice thing about WSOP is that we can move the "inner" summation (running over the column index h) "outside", to the front.

Example.

Let $s(x,y) = \begin{cases} 3 & \text{for } x = y \\ -2 & \text{for } x \neq y, (x,y) \in \Sigma \times \Sigma \text{ (mismatch)} \\ -1 & \text{otherwise (gaps)} \end{cases}$.

All the weight factors are 1.

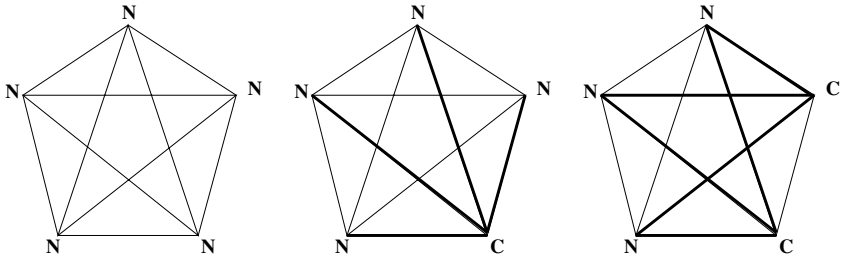
```
          a1 = -  G  C  T  G  A  T  A  T  A  A  C  T
          a2 = G  G  G  T  G  A  T  -  T  A  G  C  T
          a3 = A  G  C  G  G  A  -  A  C  A  C  C  T
               ------------------------------------
score of column: -4  9 -1 -1  9  9  1  1 -1  9 -6  9  9 = 43
```

The sum of pair score has the advantage to take all pairwise information into account, however it is easily biased by over-representation of sequences from the same family. This disadvantage can be dealt with by choosing the weights accordingly.

Multiple alignment:

		(1)		(2)		(3)	
s_1	...	N	...	N	...	N	...
s_2	...	N	...	N	...	N	...
s_3	...	N	...	N	...	N	...
s_4	...	N	...	N	...	C	...
s_5	...	N	...	C	...	C	...

Comparisons: (1) (2) (3)



	(1)	(2)	(3)
N-N pairs:	10	6	3
N-C pairs:	0	4	6
C-C pairs:	0	0	1
BLOSUM50:	70	34	22

(BLOSUM50 scores: N-N: 7, N-C: -2, C-C: 13)

An undesirable property of the WSOP cost function is the following: Consider a position i in an SP-optimal multi-alignment A^* that is *constant*, i.e., has the same residue in all sequences.

What happens when we add a new sequence? If the number of aligned sequences is small, then we would not be too surprised if the new sequence shows a different residue at the previously constant position i.

However, if the number of sequences is large, then we would expect the constant position i to remain constant, if possible.

Unfortunately, the SP score favors the opposite behavior, when scoring using a BLOSUM matrix: the more sequences there are in an MSA, the easier it is, relatively speaking, for a differing residue to be placed in an otherwise constant column.

Example:

$$\begin{array}{rccc} a_1 = & \dots & \mathtt{E} & \dots \\ \vdots & \dots & \vdots & \dots \\ a_{r-1} = & \dots & \mathtt{E} & \dots \\ a_r = & \dots & \mathtt{E} & \dots \end{array} \quad \leftrightarrow \quad \begin{array}{rccc} a_1 = & \dots & \mathtt{E} & \dots \\ \vdots & \dots & \vdots & \dots \\ a_{r-1} = & \dots & \mathtt{E} & \dots \\ a_r = & \dots & \mathtt{C} & \dots \end{array}$$

Using BLOSUM50, we obtain:

$$c(\mathtt{E}^r) = \frac{r(r-1)}{2} s(\mathtt{E},\mathtt{E}) = 6\frac{r(r-1)}{2}.$$

The value for $c(\mathtt{E}^{r-1}\mathtt{C})$ is obtained from this by subtracting $(r-1)s(\mathtt{E},\mathtt{E})$ and then adding $(r-1)s(\mathtt{E},\mathtt{C})$. So, the difference between $c(\mathtt{E}^r)$ and $c(\mathtt{E}^{r-1},\mathtt{C})$ is:

$$(r-1)s(\mathtt{E},\mathtt{E}) - (r-1)s(\mathtt{E},\mathtt{C}) = (r-1)6 - (r-1)(-3) = 9(r-1).$$

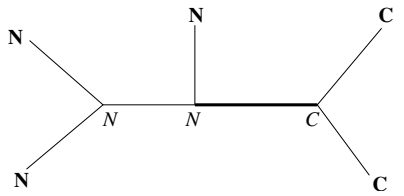
Therefore, the relative difference is

$$\frac{c(\mathtt{E}^r) - c(\mathtt{E}^{r-1}\mathtt{C})}{c(\mathtt{E}^r)} = \frac{9(r-1)}{6r(r-1)/2} = \frac{3}{r} \ ,$$

which *decreases* as the number of sequences r increases!

9.13 Scoring along a tree

Assume we have a *phylogenetic tree* T for the sequences that we want to align, i.e., a tree whose leaves are labeled by the sequences. Instead of comparing *all pairs* of residues in a column of an MSA, one may instead determine an optimal labeling of the internal nodes of the tree by symbols in a given column (in this case col. (3) from the example) and then sum over *all edges in the tree*:

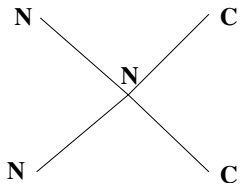


Such an optimal *most parsimonious labeling* of internal nodes can be computed in polynomial time using the *Fitch* algorithm.

Based on this tree, the scores for columns (1), (2) and (3) are: $7\times7 = 49$, $6\times7-2 = 40$ and $4\times7-2+2\times13 = 52$.

9.14 Scoring along a star

In a third alternative, one sequence is treated as the ancestor of all others in a so-called *star phylogeny*:



Based on this star phylogeny, assuming that sequence 1 is at the center of the star, the scores for columns (1), (2) and (3) are: $4 \times 7 = 28$, $3 \times 7 - 2 = 19$ and $2 \times 7 - 2 \times 2 = 10$.

9.15 Scoring Schemes

At present, there is no conclusive argument that gives any one scoring scheme more justification than the others. The sum-of-pairs score is widely used, but is problematic.

One advantage of the WSOP score (or cost) function is the ease with which gaps are modelled, since the score reduces to the summation of the pairwise scores for which the handling of gaps is well understood.

9.16 Assessing the quality of multiple alignments

Since there is no generally accepted cost function for multiple alignments it is difficult to assess their quality, especially in the case of aligning remote homologs.

There are several publically available databases which have *benchmark* alignments. The first one which is widely used is *BAliBASE* by Julie Thompson, Frédéric Plewniak and Olivier Poch (1999) Bioinformatics, 15, 87-88.
`http://www-igbmc.u-strasbg.fr/BioInfo/BAliBASE2/`

The sequences in the database are carefully aligned and expert curated. Then the *core* blocks of structural conserved regions are extracted. The score for an alignment is the percentage of those regions which it is able to find.

Below you can see one of the reference alignments. The crucial core blocks are printed in capital letters:

```
hmgl_trybr   1 kkdsnaPKRAMTSFMFFSS....dfrskhsdlsi.vemsKAAGAAWKEL
hmgt_mouse   1 .....kPKRPRSAYNIYVSesfqeakddsaqgkl.....KLVNEAWKNL
hmgb_chite   1 ...adkPKRPLSAYMLWLNsaresikrenpdfkv.tevaKKGGELWRGL
hmgl_wheat   1 ..dpnkPKRAPSAFFVFMGefreefkqknpknksvaavgKAAGERWKSL

hmgl_trybr  45 gpeeRKVYEEMAEKDKERYKREM.........
hmgt_mouse  40 speeKQAYIQLAKDDRIRYDNEMksweeqmae
hmgb_chite  46 ..kdKSEWEAKAATAKQNYIRALqeyerngg.
hmgl_wheat  48 seseKAPYVAKANKLKGEYNKAIaaynkgesa
```

The following table lists the results of some alignment programs evaluated with the BaliBase evaluation scheme. In parenthesis is the percentage of the highest score any program reached. For example, for the 1aboA alignment, COSA is the best (0.758) of all programs and has a score of *100%*. TCOFFEE reaches (0.579) which is *76%*.

Data	COSA	TCOFFEE	PRRP	CLUSTALW	DIALIGN
BaliBase Reference 1 short V1					
1aboA	**0.758 *(100)***	0.579 *(76)*	0.327 *(43)*	0.755 *(100)*	0.645 *(85)*
1idy	0.723 *(94)*	0.196 *(25)*	**0.769 *(100)***	0.608 *(79)*	0.050 *(7)*
1r69	**0.673 *(100)***	0.567 *(84)*	0.520 *(77)*	0.640 *(95)*	0.333 *(49)*
1tvxA	0.219 *(81)*	**0.272 *(100)***	0.219 *(81)*	0.123 *(45)*	0.088 *(32)*
1ubi	0.500 *(66)*	0.447 *(59)*	0.500 *(66)*	**0.760 *(100)***	0.180 *(24)*
1wit	**0.992 *(100)***	0.925 *(93)*	**0.992 *(100)***	0.892 *(90)*	0.800 *(81)*
2trx	0.788 *(97)*	**0.814 *(100)***	0.447 *(55)*	0.795 *(98)*	0.792 *(97)*
avg.	**0.665 *(100)***	0.543 *(82)*	0.539 *(81)*	0.653 *(98)*	0.413 *(62)*

9.17 Methods for MSA

Most optimization problems using the above cost functions are *NP*-hard. Hence exact solutions cannot be expected for more than 7-15 sequences and one has to resort to heuristic approaches. Most multiple alignment methods can be classified as follows:

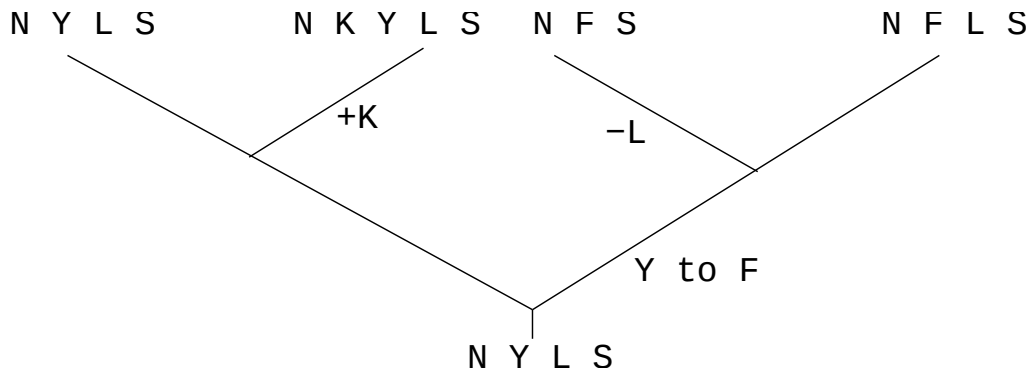
Iterative algorithms (Realigner,PRRP, SAGA)
Progressive algorithms (ClustalW, Muscle)
Consistency based algorithms (TCoffee, ProbCons)
Motif searching algorithms (Dialign, Blocks, eMotif)
Probabilistic methods (HMMs, Gibbs-Sampling)
Divide-and-Conquer algorithms (DCA,OMA)
Exact algorithms (MSA,COSA,GSA)

We give now a short overview of the central ideas.

9.18 Progressive methods

Progressive alignments start by aligning the most similar sequences first in the hope that the fewest errors are made. Then, progressively, more and more sequences are aligned to the already existing alignment.

This is typically done with the help of a (heuristic) phylogenetic tree.



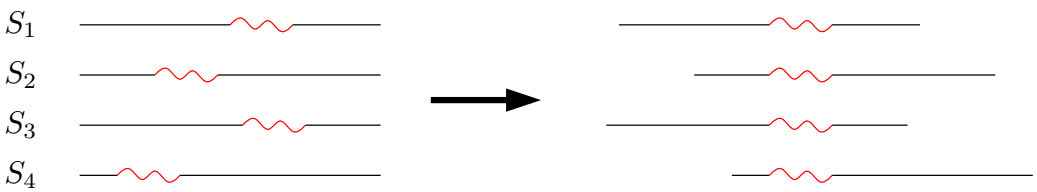
9.19 Iterative methods

The major problem with progressive methods is their sensitivity to a bad initial alignment. Iterative methods attempt to avoid this by repeatedly aligning subgroups of the sequences and then by aligning these subgroups into a larger alignment.

Since they run certain heuristics several times they are normally somewhat slower than progressive alignments. the most prominent programs here are PRRP (Gotoh) and SAGA (Notredame and Higgins).

9.20 Motif based methods

These methods try to find local motifs and use those as anchors.



The most prominent algorithms here are programs from the Dialign family and the BlockAligner.

9.21 Probabilistic methods

The most prominent of these methods are Profile HMMs and the Gibbs sampler. These methods try to maximize the likelihood in a probabilistic model of multiple alignment.

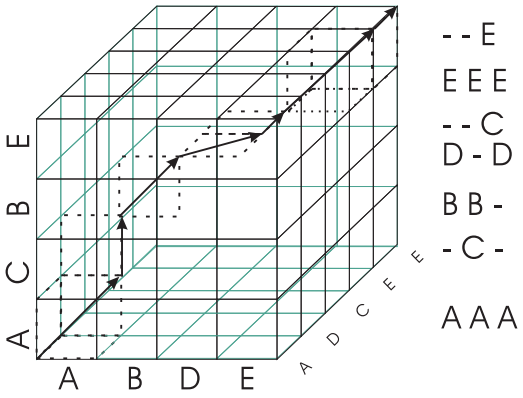
9.22 Divide-and-conquer methods

The idea is straightforward: cut the k sequences and solve the resulting subproblems recursively. The solutions can be combined trivially. If the problem size is small enough an exact algorithm can be employed.

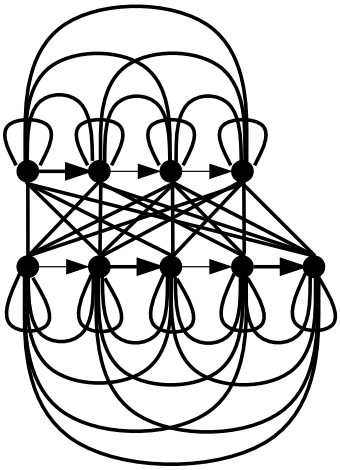
On the one hand it is clear that optimal cut positions exist, on the other hand it is clear that it is NP-hard to find them. Trivial ideas for determining the cut positions are not very succesful. Stoye showed how to compute relatively good cut positions.

9.23 Exact methods

Exact methods are either based on the natural extension of the dynamic programming algorithm for two sequences (e. g. (Lipman, Gupta, Altschul, Kececioglu),(Reinert,Lermen)).



Alternatively, exact algorithms are based on a graph-theoretic model that even allows arbitrary gap costs (Kececioglu, Reinert et al, Reinert, Althaus et al.).



```
C--GT-U
-AGGTC-
```