MODULE III

Sequence alignment -local/global, pairwise sequence alignment, scoring methods. Needleman and Wunsch algorithm, global and local alignments. Multiple sequence alignment.
Scoring matrices: basic concept of a scoring matrix, Matrices for nucleic acid and proteins sequences, PAM and BLOSUM series, principles based on which these matrices are derived. Differences between distance & similarity matrix.

Sequence alignment

- Sequence alignment is the process of comparing two or more sequences by searching a series of individual characters or patterns that are in the same order in the sequences.
- Way of arranging the sequences of DNA, RNA or protein to identify regions of similarity
- Sequence alignment methods are used to find the best- matching sequences

Why align sequences

It is Useful for discovering

• Functional

• Structural and

• Evolutionary relationship

– For example

• To find whether two (or more) genes or proteins are evolutionarily related to each other

• Two proteins with similar sequences will probably be structurally or

functionally similar

Simple Alignments

- No gap

```
AATCTATA        AATCTATA        AATCTATA
AAGATA           AAGATA           AAGATA
```

FIGURE 2.2 *Three possible simple alignments between two short sequences.*

$$\sum_{i=1}^{n} \begin{cases} \text{match score; if } seq1_i = seq2_i \\ \text{mismatch score; if } seq1_i \neq seq2_i \end{cases}$$

where n is the length of the longer sequence. For example, assuming a match score of 1 and a mismatch score of 0, the scores for the three alignments shown in Figure 2.2 would be 4, 1, and 3, from left to right.

Gaps

Gaps can be introduced in the sequence to increase similarity

A trace can represent a **substitution**:

```
AKVAIL
AKIAIL
```

A trace can represent a **deletion**:

```
VCGMD
VCG-D
```

A trace can represent a **insertion**:

```
GS-K
GSGK
```

1. mutation (substitution)
2. insertion } gap
3. deletion

```
AATCTATA        AATCTATA        AATCTATA
AAG—AT-A        AA—G—ATA        AA——GATA
```

FIGURE 2.3 *Three possible gapped alignments between two short sequences.*

Gap Penalty

$$\sum_{i=1}^{n} \begin{cases} \text{gap penalty; if } seq1_i = \text{'-'} \text{ or } seq2_i = \text{'-'} \\ \text{match score; if no gaps and } seq1_i = seq2_i \\ \text{mismatch score; if no gaps and } seq1_i \neq seq2_i \end{cases}$$

For example, assuming a match score of 1, a mismatch score of 0, and a gap penalty of –1, the scores for the three gapped alignments shown in Figure 2.3 would be 1, 3, and 3, from left to right.

There are two types of alignment

- local
- global

Global Vs Local Alignment

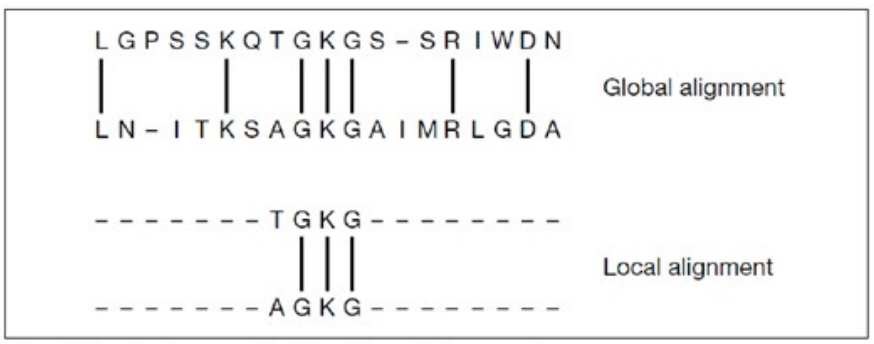


Fig.: Distinction between Global and Local alignment of two sequences

Global alignment

In global alignment, an attempt is made to align the entire sequence(end to end). If two sequences have approximately the same length and are quite similar, they are suitable for the global alignment.

```
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
   ||||||||||    |||||||  ||||||||||||| |||||||
5' ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA 3'
```

Local alignment

Local alignment concentrates on finding stretches of sequences with high level of matches.

```
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
            ||||  ||||||  |||||||||||||||
         5' TACTCACGGATGAGGTACTTTAGAGGC 3'
```

Global Sequence Alignment	Local Sequence Alignment
In global alignment, an attempt is made to align the entire sequence (end to end alignment)	Finds local regions with the highest level of similarity between the two sequences.
A global alignment contains all letters from both the query and target sequences	A local alignment aligns a substring of the query sequence to a substring of the target sequence.
If two sequences have approximately the same length and are quite similar, they are suitable for global alignment.	Any two sequences can be locally aligned as local alignment finds stretches of sequences with high level of matches without considering the alignment of rest of the sequence regions.
Suitable for aligning two closely related sequences.	Suitable for aligning more divergent sequences or distantly related sequences.
Global alignments are usually done for comparing homologous genes like comparing two genes with same function (in human vs. mouse) or comparing two proteins with similar function.	Used for finding out conserved patterns in DNA sequences or conserved domains or motifs in two proteins.
A general global alignment technique is the Needleman–Wunsch algorithm.	A general local alignment method is Smith–Waterman algorithm.
Examples of Global alignment tools: • > EMBOSS Needle • > Needleman-Wunsch Global Align Nucleotide Sequences (Specialized BLAST)	**Examples of Local alignment tools:** • > BLAST • > EMBOSS Water • > LALIGN

Needleman and Wunsch algorithm

- proposed by Needleman and Wunsch in 1970
- aimed to global alignment
- Dynamic programming algorithms find the best solution by breaking the original problem into smaller sub-problems and then solving.
- The Needleman-Wunsch algorithm is a dynamic programming algorithm for optimal sequence alignment
- In order to perform a Needleman-Wunsch alignment, a matrix is created which allows us to compare the two sequences. The score M(i,j) for every cell depends on the three cells

corresponding to either or both sequence having 1 less letter (i.e. cells M(i-1.j), M(i,j-1) and M(i-1,j-1). It is calculated as follows:

$$M(i,j) = MAX(M(i-1,j-1) + S(Ai, Bj), M(i-1, j) + gap, M(i,j-1) + gap)$$

where gap is the gap penalty and the function S returns the score/penalty for matching the two corresponding letters.

S(Ai, Bj)=score if Ai=Bj

S(Ai, Bj)=mismatch penalty if Ai not equal to Bj

Once we have computed this score for every cell, we must do a "traceback", that is to determine the actual set of operations that lead to the score. This algorithm performs alignments with a time complexity of O(mn) and a space complexity of O(mn).

Algorithm proceeds in 3 steps

1. **Draw matrix**
2. **Assign Score**
3. **Traceback and alignment**

Example:

Find the best alignment of these two sequences:

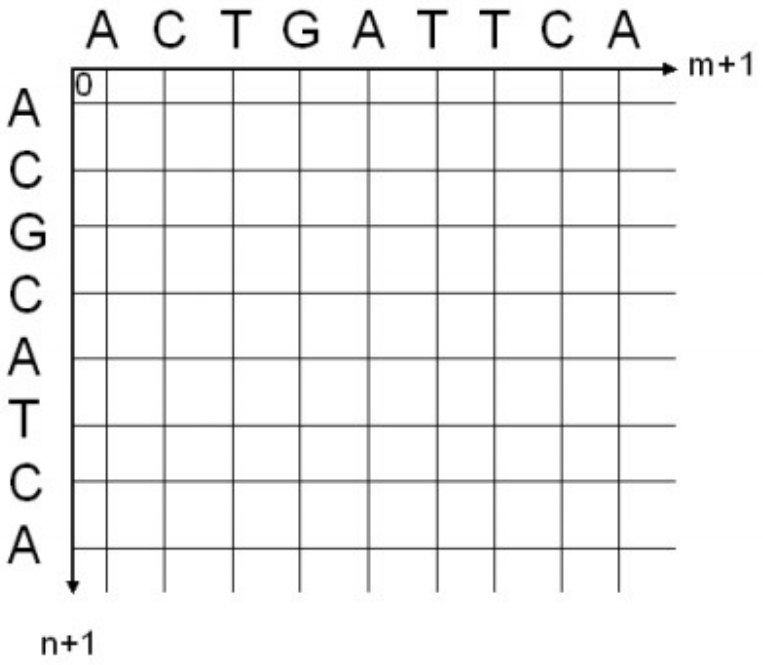
ACTGATTCA
ACGCATCA

Using -2 as a gap penalty, -3 as a mismatch penalty, and 2 as the score for a match.

Solution:

Step 1: Draw the matrix

For 2 sequences (length m and length n) what size scoring matrix is needed for their alignment? Grid dimensions must be (m+1) × (n+1). Think of each increment as a division of the sequence members:



Step 2: Assign scores

		A	C	T	G	A	T	T	C	A
	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
A	-2	2	0	-2	-4	-6	-8	-10	-12	-14
C	-4	0	4	2	0	-2	-4	-6	-8	-10
G	-6	-2	2	1	4	2	0	-2	-4	-6
C	-8	-4	0	-1	2	1	-1	-3	0	-2
A	-10	-6	-2	-3	0	4	2	0	-2	2
T	-12	-8	-4	0	-2	2	6	4	2	0
C	-14	-10	-6	-2	-4	0	4	2	6	4
A	-16	-12	-8	-4	-5	-2	2	1	4	8

Step 3: Trace back

The optimal path is traced beginning from the lower right-hand corner

		A	C	T	G	A	T	T	C	A
	0	-2	-4	-6	-8	-10	-12	-14	-16	-18
A	-2	2	0	-2	-4	-6	-8	-10	-12	-14
C	-4	0	4	2	0	-2	-4	-6	-8	-10
G	-6	-2	2	1	4	2	0	-2	-4	-6
C	-8	-4	0	-1	2	1	-1	-3	0	-2
A	-10	-6	-2	-3	0	4	2	0	-2	2
T	-12	-8	-4	0	-2	2	6	4	2	0
C	-14	-10	-6	-2	-4	0	4	2	6	4
A	-16	-12	-8	-4	-5	-2	2	1	4	8

Result:

This analysis yielded the following alignment:

```
ACTG-ATTCA
||   || ||
AC-GCAT-CA
```

The alignment score is equal to the value in the lower right-hand corner of the matrix (8).

Smith and Waterman Algorithm

- Over a decade after the initial publication of the Needleman-Wunsch algorithm, a modification was made to allow for local alignments by smith and Waterman in 1981.
- In this adaptation, the alignment path does not need to reach the edges of the search graph, but may begin and end internally.
- In order to accomplish this, 0 was added as a term in the score calculation described by Needleman and Wunsch.

$$M(i,j) = MAX(M(i-1,j-1) + S(Ai, Bj), M(i-1, j) + gap, M(i,j-1) + gap, 0)$$

Example: Find the best local alignment between these two sequences:

ATGCATCCCATGAC

TCTATATCCGT

Using -2 as a gap penalty, -3 as a mismatch penalty, and 2 as the score for a match. Solution: Traceback begins at the highest value (which is also the alignment score).

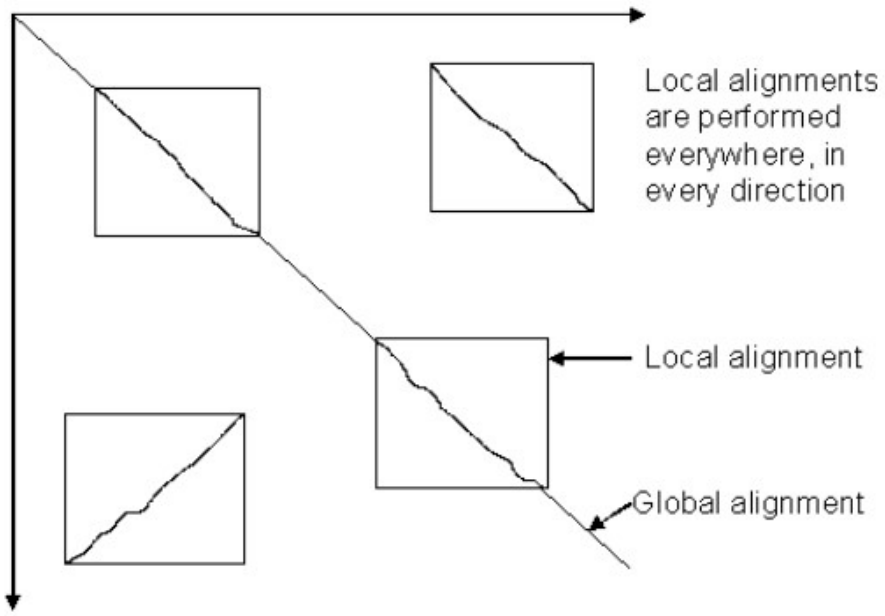
		A	T	G	C	A	T	C	C	C	A	T	G	A	C
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	2	0	0	0	2	0	0	0	0	2	0	0	0
C	0	0	0	0	2	0	0	4	2	2	0	0	0	0	2
T	0	0	2	0	0	0	0	2	1	0	0	2	0	0	0
A	0	2	0	0	0	2	0	0	0	0	2	0	0	2	0
T	0	0	4	2	0	0	2	0	0	0	0	4	2	0	0
A	0	2	0	0	0	2	0	0	0	0	2	0	0	2	0
T	0	0	4	2	0	0	4	2	0	0	0	4	0	0	0
C	0	0	2	0	4	0	0	6	4	2	0	0	0	0	2
C	0	0	0	0	2	0	0	4	8	6	4	2	0	0	2
G	0	0	0	2	0	0	0	2	6	5	3	1	4	2	0
T	0	0	2	0	0	0	2	0	4	3	2	5	3	1	0

Which yields the alignment:

```
ATCC
||||
ATCC
```

With an alignment score of 8.

Local alignments are performed everywhere possible along two sequences.



Pair wise sequence alignment

- This method compares two biological sequences of either protein,DNA or RNA
- Can be categorized as local or global alignment
- Comparitively simple algorithm is used for implantation
- Needle man-Wunsch algorithm is used for global alignment
- Smith-waterman algorithm is used for local alignment

Applications

- To find out conserved region between two sequences
- Similarity searches in a database

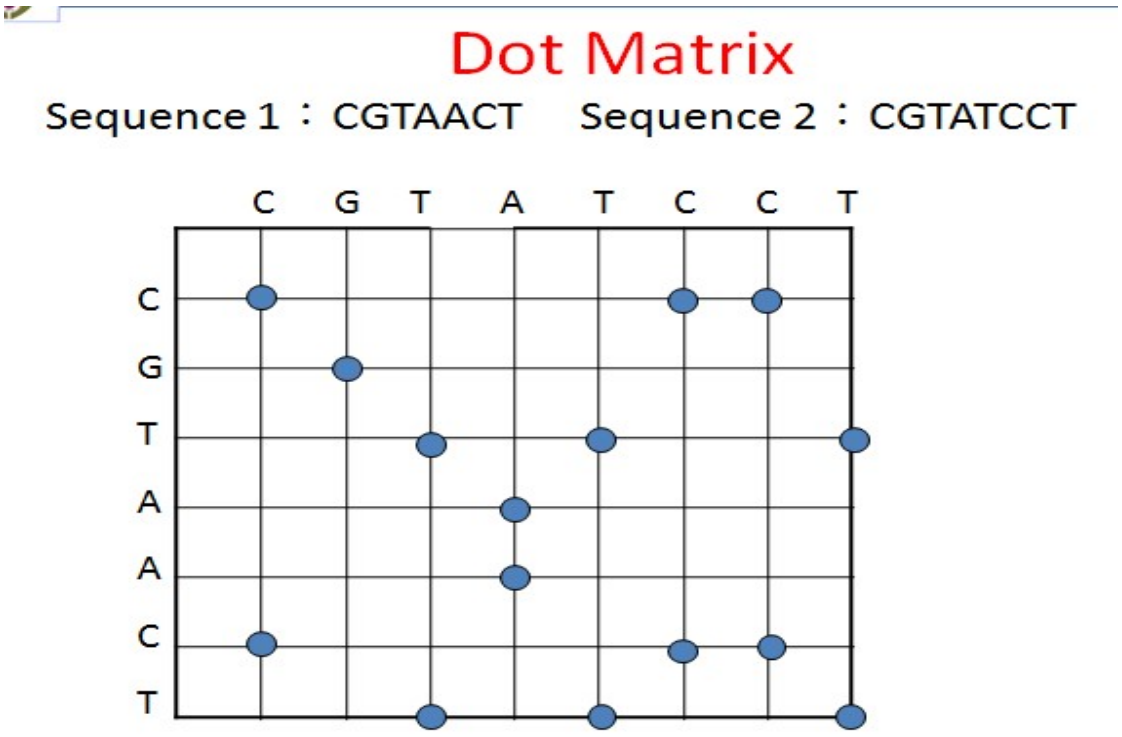
Pair-wise alignment methods

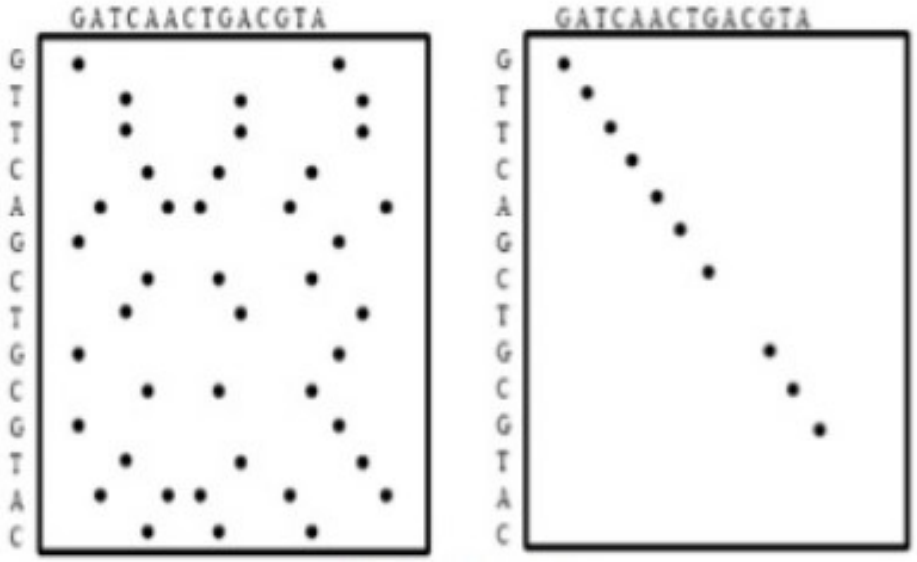
1. Dot matrix method
2. Dynamic programming
3. Word methods

A dot matrix analysis

- dot matrix analysis is a method for comparing two sequences to look for possible alignment
- The algorithm for a dot matrix:

1. One sequence (A) is listed across the top of the matrix and the other (B) is listed down the left side
2. . Starting from the first character in B, one moves across the page keeping in the first row and placing a dot in many column where the character in A is the same
3. The process is continued until all possible comparisons between A and B are made
4. Any region of similarity is revealed by a diagonal row of dots
5. Isolated dots not on diagonal represent random matches

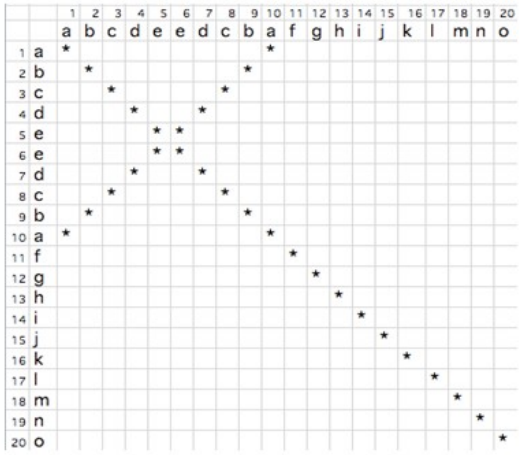




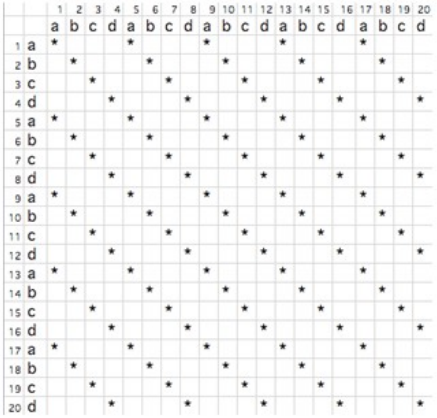
Applications

- Can use dot matrices to align two sequences
  - Regions of similarity appear as diagonal runs of dots
  - Can use to find inversions(palindromes)
  - Reverse diagonals (perpendicular to diagonal) indicate inversions
- Can use to find repetition within sequence.
  - Repeats appear as a set of diagonal runs stacked vertically

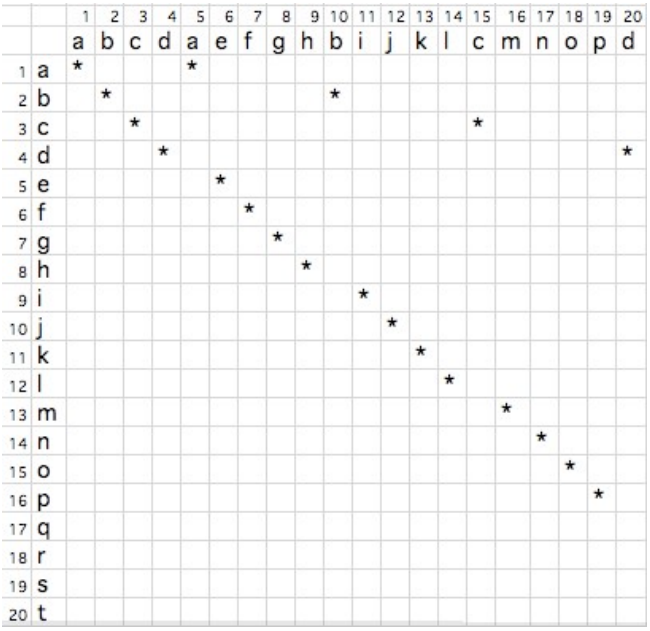
Reverse



repetition



diagonal **alignment** with "gaps"



Limitation of dot matrix

- ⇨ It's a visual aid. The human eye can rapidly identify similar regions in sequences.
- ⇨ It's a good way to explore sequence organisation.
- ⇨ It does not provide an alignment.

Dynamic Programming

- The approach compares every pair of characters in the two sequences and generates an alignment, which is the best or optimal.
- The method can be useful in aligning nucleotide to protein sequences.

Two approaches for dynamic programming: Top-down approach and Bottom-up.

Dynamic programing

1 .Global- Needleman-Wunsch algorithm

Alignment between the complete sequence A and the complete sequence B

2. semi Global Needleman-Wunsch algorithm

Alignment between 1 larger sequence and 1 smaller sequence . terminal gaps not penalised .

3 Local Smith-Waterman algorithm

Alignment between a sub-sequence of A an a sub-sequence of B

Word methods

Word methods, also known as *k*-tuple methods, are heuristic methods that are not guaranteed to find an optimal alignment solution, but are significantly more efficient than dynamic programming.

These methods are especially useful in large-scale database searches where it is understood that a large proportion of the candidate sequences will have essentially no significant match with the query sequence. Word methods are best known for their implementation in the database search tools FASTA and the BLAST family.

1. BLAST **b*asic* l*ocal* a*lignment* s*earch* t*ool***
2. FASTA **FASTA** is a DNA and protein sequence alignment software package

BLAST

The BLAST program was developed by Stephen Altschul of NCBI in 1990 and has since become one of the most popular programs for sequence analysis. BLAST uses heuristics to align a query sequence with all sequences in a database. The objective is to find high-scoring ungapped segments among related sequences. The existence of such segments above a given

threshold indicates pairwise similarity beyond random chance, which helps to discriminate related sequences from unrelated sequences in a database.

BLAST performs sequence alignment through the following steps. The first step is to create a list of words from the query sequence. Each word is typically three residues for protein sequences and eleven residues for DNA sequences. The list includes every possible word extracted from the query sequence. This step is also called *seeding*. The second step is to search a sequence database for the occurrence of these words. This step is to identify database sequences containing the matching words. The matching of the words is scored by a given substitution matrix. A word is considered a match if it is above a threshold. The fourth step involves pairwise alignment by extending from the words in both directions while counting the alignment score using the same substitution matrix. The extension continues until the score of the alignment drops below a threshold due to mismatches (the drop threshold is twenty-two for proteins and twenty for DNA). The resulting contiguous aligned segment pair without gaps is called *high-scoring segment pair* (HSP). In the original version of BLAST, the highest scored HSPs are presented as the final report. They are also called maximum scoring pairs.

The BLAST output includes a graphical overview box, a matching list and a text description of the alignment (Fig). The graphical overview box contains colored horizontal bars that allow quick identification of the number of database hits and the degrees of similarity of the hits. The color coding of the horizontal bars corresponds to the ranking of similarities of the sequence hits (red: most related; green and blue: moderately related; black: unrelated). The length of the bars represents the spans of sequence alignments relative to the query sequence. Each bar is hyperlinked to the actual pairwise alignment in the text portion of the report. Below the graphical box is a list of matching hits ranked by the *E*-values in ascending order. Each hit includes the accession number, title (usually partial) of the database record, bit score, and *E*-value. This list is followed by the text description, which may be divided into three sections: the header, statistics, and alignment.

The header section contains the gene index number or the reference number of the database hit plus a one-line description of the database sequence. This is followed by the summary of the statistics of the search output, which includes the bit score, *E*-value, percentages of identity, similarity ("Positives"), and gaps. In the actual alignment section, the query sequence is on the top of the pair and the database sequence is at the bottom of the pair labeled as *Subject.* In

between the two sequences, matching identical residues are written out at their corresponding positions, whereas nonidentical but similar residues are labeled with "+". Any residues identified as LCRs in the query sequence are masked with *X*s or *N*s so that no alignment is represented in those regions.

FASTA

FASTA (FAST ALL, www.ebi.ac.uk/fasta33/) was in fact the first database similarity search tool developed, preceding the development of BLAST. FASTA uses a "hashing" strategy to find matches for a short stretch of identical residues with a length of *k*. The string of residues is known as *ktuples* or *ktups*, which are equivalent to words in BLAST, but are normally shorter than the words. Typically, a ktup is composed of two residues for protein sequences and six residues for DNA sequences. The first step in FASTA alignment is to identify ktups between two sequences by using the hashing strategy. This strategy works by constructing a lookup table that shows the position of each ktup for the two sequences under consideration. The positional difference for each word between the two sequences is obtained by subtracting the position of the first sequence from that of the second sequence and is expressed as the offset. The ktups that have the same offset values are then linked to reveal a contiguous identical sequence region that corresponds to a stretch of diagonal in a two-dimensional matrix (Fig.).

1. Given two amino acid sequences for comparision:

sequence 1 **AMPSDGL**
sequence 2 **GPSDNAT**

2. Construct a hashing table:

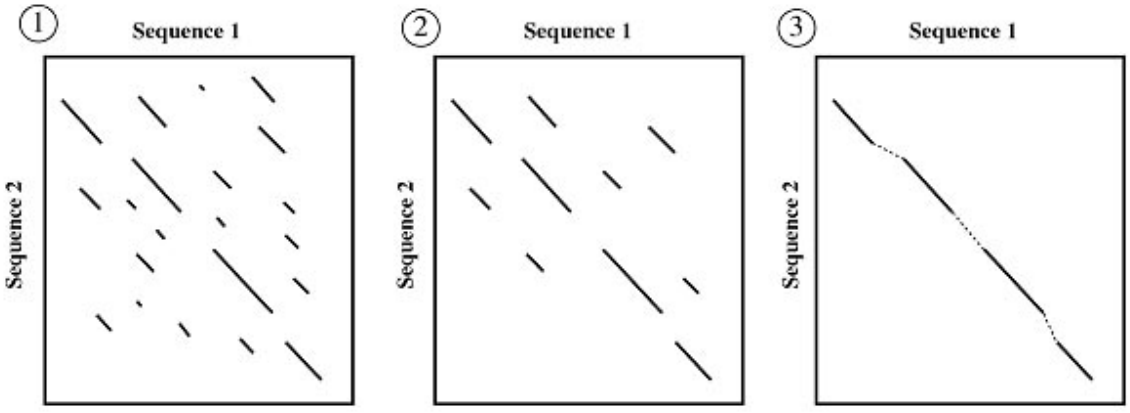
amino acid	sequence position		offset
	seq 1	seq 2	
A	1	6	-5
D	5	4	1
G	6	1	5
L	7	-	-
M	2	-	-
N	-	5	-
P	3	2	1
S	4	3	1
T	-	7	-

3. Identify residues with the same offset values (highlighted in grey).

4. Find the matching word of three residues in the order of 3, 4 and 5 in one sequence and 2, 3,and 4 in the other.

5. This allows establishment of alignment between the two sequences.

```
sequence 1   AMPSDGL-
                |||
sequence 2   -GPSDNAT
```



The second step is to narrow down the high similarity regions between the two sequences. Normally, many diagonals between the two sequences can be identified in the hashing step. The top ten regions with the highest density of diagonals are identified as high similarity regions. The diagonals in these regions are scored using a substitution matrix. Neighboring high-scoring segments along the same diagonal are selected and joined to forma single alignment. This step allows introducing gaps between the diagonals while applying gap penalties. The score of the gapped alignment is calculated again. In step 3, the gapped alignment is refined further using the Smith–Waterman algorithm to produce a final alignment (Fig.). The last step is to perform a statistical evaluation of the final alignment as in BLAST, which produces the *E*-value.

Similar to BLAST, FASTA has a number of subprograms. The web-based FASTA program offered by the European Bioinformatics Institute (www.ebi.ac.uk/) allows the use of either DNA or protein sequences as the query to search against a protein database or nucleotide database. Some available variants of the program are FASTX, which translates aDNAsequence and uses the translated protein sequence to query a protein database, and TFASTX, which compares a protein query sequence to a translated DNA database.

Steps of the FASTA alignment procedure. In step 1 (*left*), all possible ungapped alignments are found between two sequences with the hashing method. In step 2 (*middle*), the alignments are scored according to a particular scoring matrix. Only the ten best alignments are selected. In step 3 (*right*), the alignments in the same diagonal are selected and joined to form a single gapped alignment, which is optimized using the dynamic programming approach.

Scoring Matrices

- Sequence alignment and database searching algorithm compares sequence with each other as a series of characters
- All the above relay on some scoring schemes for that
- It is used to assign a score a score to each comparison of a pair of sequences
- The score in the matrices are integer values
    - A + or 0 is assigned for a match
    - -ve is assigned for a mismatch

Types

Scoring matrices for nucleic acid

1. **Identity scoring matrix**

Is the simplest type of matrix were the characters are classified as identical(score 1) and non identical(score 0).this is rarely used

	A	T	C	G
A	1	0	0	0
T	0	1	0	0
C	0	0	1	0
G	0	0	0	1

Identity

2. DNA scoring matrix

-use in the alignment of DNA/RNA sequences

- is a 4*4 matrix since there are only 4 basis in DNA and RNA

DNA - **Adenine ('A'), Thymine ('T'), Guanine ('G') and Cytosine ('C').**

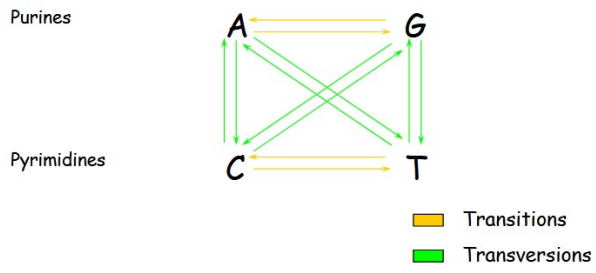
RNA -Adenine ('A'), Guanine ('G') and Cytosine ('C') AND Uracil ('U')

-this consider the changes as transitions and traversions.The following matrix scores 0 for identity,5 for transitions and 1 for transversions

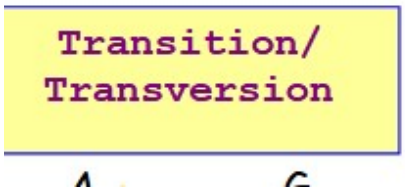
Transition- Mutation From one Purine to other Purine(A->T)

Transversion-Mutation from Purine to Pyramidine(A->C)

Base mutations (general): definitions



	A	T	C	G
A	0	5	5	1
T	5	0	1	5
C	5	1	0	5
G	1	5	5	0



3. **Chemical similarity scoring matrix**

Provides greater weight to aminoacids with similar chemical properties(ed shape ,sizetec..)

4. **Observed matrices**
   - Are mose common in use
   - Constructed by analyzing the substitution frequencies seen in the alignment of known family of proteins
   - Every possible identity and substitution is assigned a score
     Identities assigned higher scores
     Substitution assigned lower scores
   - Mismatches or matches that are unlikely to have seen as the result of evolution is assigned a negative score

Scoring matrices for amino acids/ proteins

- We know that there are 20 amino acids so the scoring matrix is a 20*20 matrix that shows scores of substitution
- The most frequently used observed scoring matrix are PAM and BLOSUM

PAM matrix

- A **point accepted mutation** (PAM) also known as a PAM is the replacement of a single amino acid in the primary structure of a protein with another single amino acid, which is accepted by the processes of natural selection.

- PAM is the substitution of one amino acid by other such that the protein function stays conserved

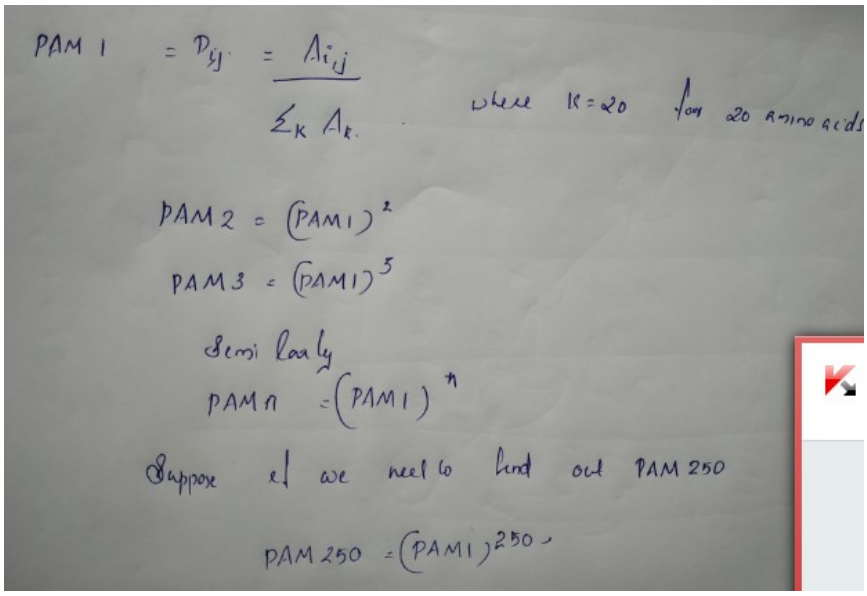
- This definition does not include all point mutations in the DNA of an organism. A PAM matrix is a matrix where each column and row represents one of the twenty standard amino acids.
- In bioinformatics, PAM matrices are regularly used as substitution matrices to score sequence alignments for proteins. Each entry in a PAM matrix indicates the likelihood of the amino acid of that row being replaced with the amino acid of that column through a series of one or more point accepted mutations during a specified evolutionary interval, rather than these two amino acids being aligned due to chance. Different PAM matrices correspond to different lengths of time in the evolution of the protein sequence.
- PAM unit is the time in which about 1 percentage of amino acid in a sequence undergo acceptable mutation

  Steps to compute scoring matrix

  1: Align protein sequence which are 1 PAM unit diverged

  2:Let $A_{i,j}$ be the number of time A_i is substituted by A_j

  3:Compute the frequency of amino acid A_i



	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Fig PAM 250

BLOSUM matrix

- In bioinformatics, the **BLOSUM** (**BLO**cks **SU**bstitution **M**atrix) matrix is a substitution matrix used for sequence alignment of proteins.
- BLOSUM matrices are used to score alignments between evolutionarily divergent protein sequences.
- They are based on local alignments.
- BLOSUM matrices were introduced er by Steven Henikoff and Jorja Henikoff.
- They scanned the BLOCKS database for very conserved regions of protein families (that do not have gaps in the sequence alignment) and then counted the relative frequencies of amino acids and their substitution probabilities. Then, they calculated a log-odds score for each of the 210 possible substitution pairs of the 20 standard amino acids. All BLOSUM matrices are based on observed alignments; they are not extrapolated from comparisons of closely related proteins like the PAM Matrices.
- BLOSUM matrices are obtained by using blocks of similar amino acid sequences as data, then applying statistical methods to the data to obtain the similarity scores. Statistical Methods Steps : [8]

1. Eliminating Sequences

Eliminate the sequences that are more than r% identical. There are two ways to eliminate the sequences. It can be done either by removing sequences from the block or just by finding similar sequences and replace them by new sequences which could represent the cluster. Elimination is done to remove protein sequences that are more similar than the specified threshold.

2. Calculating Frequency & Probability

- A database storing the sequence alignments of the most conserved regions of protein families. These alignments are used to derive the BLOSUM matrices. Only the sequences with a percentage of identity lower than the threshold are used. By using the block, counting the pairs of amino acids in each column of the multiple alignment.

3. Log odd ratio

It gives the ratio of the occurrence each amino acid combination in the observed data to the expected value of occurrence of the pair. It is rounded off and used in the substitution matrix.

$$LogOddRatio = 2\log_2\left(\frac{P(O)}{P(E)}\right)$$

In which $P(O)$ is the possibility of observed and $P(E)$ is the possibility of expected.

The odds for relatedness are calculated from log odd ratio, which are then rounded off to get the substitution matrices BLOSUM matrices.

Scores within a BLOSUM are log-odds scores that measure, in an alignment, the logarithm for the ratio of the likelihood of two amino acids appearing with a biological sense and the likelihood of the same amino acids appearing by chance. The matrices are based on the minimum percentage identity of the aligned protein sequence used in calculating them. Every possible identity or substitution is assigned a score based on its observed frequencies in the alignment of related proteins. A positive score is given to the more likely substitutions while a negative score is given to the less likely substitutions.

To calculate a BLOSUM matrix, the following equation is used:

$$S_{ij} = \left(\frac{1}{\lambda}\right)\log\left(\frac{p_{ij}}{q_i * q_j}\right)$$

Here, p_{ij} is the probability of two amino acids i and j replacing each other in a homologous sequence, and q_i and q_j are the background probabilities of finding the amino acids i and j in any protein sequence. The factor λ is a scaling factor, set such that the matrix contains easily computable integer values.

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Fig:BLOSUM45 matrix

Multiple sequence alignment

- A multiple sequence alignment is an alignment of than two sequences obtained by inserting gaps ("-") into sequences such that the resulting sequences have all length L and can be arranged in a matrix of N rows and L columns where each column represents a homologous position.

- The principle is that multiple alignments are achieved by successive application of pairwise methods

Application of MSA

- In order to characterize protein families, identify shared regions of homology in a multiple sequence alignment
- Determination of the consensus sequence of several aligned sequences.
- Consensus sequences can help to develop a sequence "finger print" which allows the identification of members of distantly related protein family (motifs)
- MSA can help us to reveal biological facts about proteins, like analysis of the secondary/tertiary structure
- Phylogenetic analysis
- To detect regions of variability or conservation in a family of proteins
- Detection of homology between a new sequence and an existing gene family

The msa can be viewed as an extention of pairwise alignment.The first step is pair wise alignment of all sequences,here we have 4 sequences so we need 6 comparisons.The next step is to perform cluster analysis on these pairwise data to generate a heierarchy for alignment.
Methods of MSA

1.Dynamic programming

2.Progressive method

3.Iteration methods

Dynamic Programming method

Computes an optimal alignment for a given score function. Because of its high running time , it is not typically used in practice.A dynamic programming method is SUM OF PAIR method

Sum of Pairs(SP) Method

- Is a dynamic programming method
- Instead of 2 sequence at a time we need to align 3 or more sequence at a time
- Sp functions scores each position in the protein that is each column,as the sum of pair wise scores.For k sequences there are k(k-1)/2 unique pairwise comparisons for k=4 there are 6 possible pair wise comparisons
  - Consider aligning the following 4 protein sequences
    S1 = AQPILLLV
    S2 = ALRLL
    S3 = AKILLL
    S4 = CPPVLILV
  - Next consider the following MSA

```
A Q P I L L L V
A L R - L L - -
A K - I L L L -
C P P V L I L V
```

- Assume c(match) = 1 , c(mismatch) = -1 , and c(gap) = -2 , also assume c(-, -) = 0 to prevent the double counting of gaps.
- Then the SP score for the 4th column of the MSA would be $SP(m_4) = SP(I, -, I, V)$
  $= c(I,-) + c(I,I) + c(I,V) + c(-,I) + c(-, V) + c(I,V)$
  $= -2 + 1 + (-1) + (-2) + (-2) + (-1)$
  $= -7$
- To find SP(MSA) we would find the score of each m_i and then SUM all $SP(m_i)$ scores to get the score MSA.

Progressive methods

- This approach repeatedly aligns two sequences, two alignments, or a sequence ce with an alignment.
- The most widely used approach
- Builds up a final MSA by combining pairwise alignments beginning with the most similar pair and progressing to the most distantly related

Progressive alignment methods require two stages:

-A first stage in which the relationships between the sequences are represented as a tree, called a *guidetree*

-Second step in which the MSA is built by adding the sequences sequentially to the growing MSA according to the guide tree

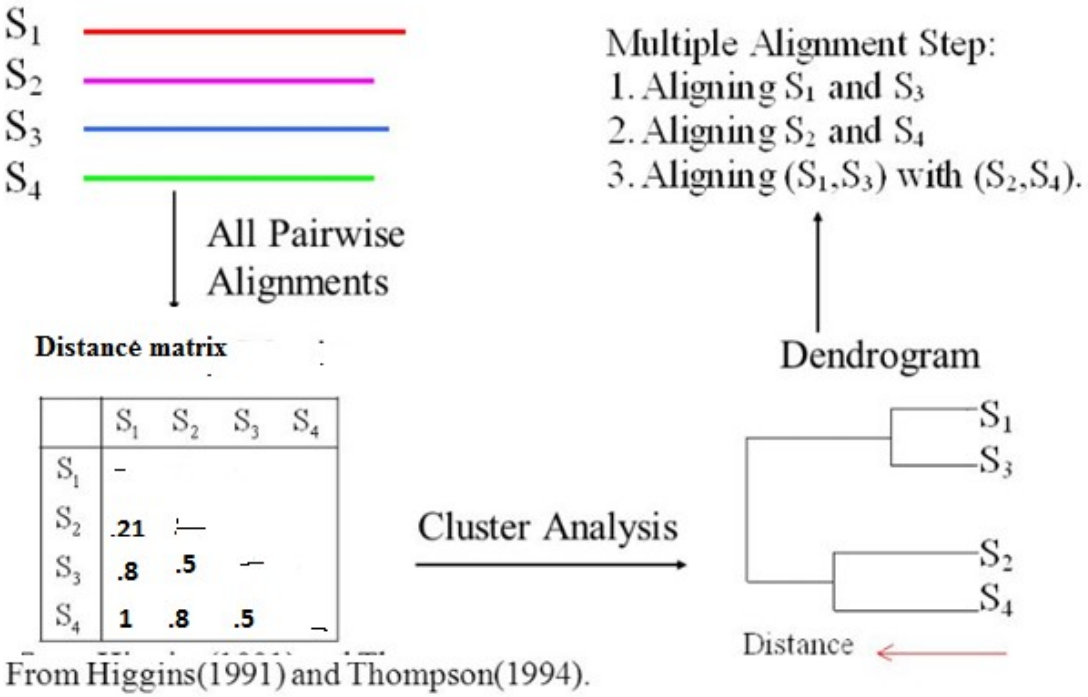
Following are the two progressive alignment methods

1. **CLUSTAL W**
2. **CLUSTAL X**

CLUSTAL W

- Works by progressive alignment.
- ClustalW was introduced by Julie D. Thompson and Toby Gibson of EMBL, EBI.
- Most closely related sequences are aligned first, and then additional sequences and groups of sequences are added, guided by the initial alignments.
- Uses alignment scores to produce a phylogenetic tree.
- Is command driven

- ClustalW is a progressive method for MSA
- The term "progressive" is used because ClustalW starts with using a pairwise method to determine the most related sequences and then progressively adding less related sequences or groups of sequences to the initial alignment.



From Higgins(1991) and Thompson(1994).

CLUSTALW Algorithm

- Step 1 : Determine all pairwise alignment between sequences and determine degrees of similarity between each pair.
- Step 2 : Construct a similarity tree.
- Step 3 : Combine the alignments starting from the most closely related groups to the most distantly related groups, using the "once a gap always a gap" rule .

Once a gap always a gap means The gaps introduced earlier in the alignment remain valid as new sequences are added

- Use a pairwise alignment method to compute pairwise alignment amongst the sequences.
- Using the pairwise alignments compute a "distance" between all pairs of sequences. A method commonly used is as follows:

  for each pairwise alignment look at the non-gapped position and count the number of mismatches between the two sequences, then divide this value by the number of non-gapped pairs to calculate the distance, for example

```
NKL-ON      distance = 1/4 = .25
-MLNON
```

Step1 of the algorithm is as follows

- After computing the "distance" between all pairs of sequences we put them in to a matrix. For example if we consider a set of 7 sequences we could have the following matrix:

Seq.	S1	S2	S3	S4	S5	S6	S7
S1	-						
S2	.17	-					
S3	.59	.60	-	(SYMMETRIC)			
S4	.59	.59	.13	-			
S5	.77	.77	.75	.75	-		
S6	.81	.82	.73	.74	.80	-	
S7	.87	.86	.86	.88	.93	.90	-

Step 2 of algorithm works as follows

- Construct a similarity tree. ClustalW uses a distance matrix and the Neighbor Joining method to construct the similarity tree. The root is placed a the midpoint of the longest chain of consecutive edges.

Step 3 of algorithm works as follows

- Combine the alignments starting from the most closely related groups to the most distantly related groups by going from tip of tree to the root of the tree.
- In our example we first align S1 with S2 (profile 1) then S3 with S4 (profile 2), then align grp1 with grp2, we continue until the root is reached.

  Each alignment (sequence-sequence, sequence-profile, profile-profile) involves dynamic programming by the SP score method.

In order to compensate for biased representation in large subfamilies, ClustalW reduces the weight of related sequences.

To assign a weight to a sequence, ClustalW does the following:

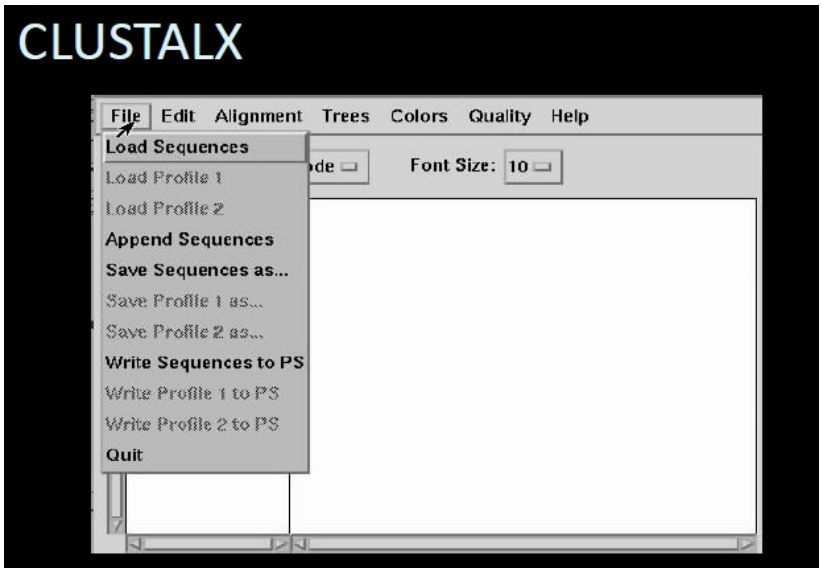
(1) It uses the values on the NJ-tree from the sequence to the root of the tree.

(2) If two or more sequences share a branch, which may indicate an evolutionary relationship, its value is split amongst the sequences.

CLUSTAL X

- ClustalXprovides a new window-based user interface to the ClustalWprogram.

- It uses the Vibrant multi-platform user interface development library, developed by the National Center for Biotechnology as part of their NCBI software development toolkit



Advantages

- Efficient enough to implement on a large scale for many (100s to 1000s) sequences.
- Progressive alignment services are commonly available on publicly accessible web servers, so users need not locally install the applications of interest.
- Most widely used method of multiple sequence alignment because of speed and accuracy.

Disadvantages

- Progressive alignments are not guaranteed to be globally optimal.
- The primary problem is that when errors are made at any stage in growing the MSA, these errors are then propagated through to the final result.
- Performance is also particularly bad when all of the sequences in the set are rather distantly related

Iterative method

Works similarly to progressive methods but repeatedly realigns the initial sequences as well as adding new sequences to the growing MSA.

Difference between Distance and Similarity matrix:

Trees that represent protein or nucleic acid sequences usually display the differences between various sequences. One way to measure distances is to count the number of mismatches in a pairwise alignment. Another method, employed by the Feng and Doolittle progressive alignment algorithm, is to convert similarity scores to distance scores. Similarity scores are calculated from a series of pairwise alignments among all the proteins being multiply aligned. The similarity scores S between two sequences (i, j) are converted to distance scores D using the equation

$$D = -\ln Seff$$

Where

$$Seff = \frac{Sreal(ij) \_ Srand(ij)}{Siden(ij) \_ Srand(ij)} \times 100$$

Here, Sreal(ij) describes the observed similarity score for two aligned sequences i and j, Siden(ij) is the average of the two scores for the two sequences compared to themselves (if score i compared to i receives a score of 20 and score j compared to j receives a score of 10, then Siden(ij) =15); Srand(ij) is the mean alignment score derived from many (e.g., 1000) random shufflings of the sequences; and Seff is a normalized score. If sequences i, j have no similarity, then Seff = 0 and the distance is infinite. If sequences i, j are identical, then Seff = 1 and the distance is 0.

Similarity matrices are used for database searching, while distance matrices are naturally used for phylogenetic tree construction. The distance score (D) is usually calculated by summing up of mismatches in an alignment divided by the total number of matches and mismatches, which represents the number of changes required to change one sequence into the other, ignoring gaps.

$$D = \frac{\text{Matches}}{(\text{Matches} + \text{Mismatches})}$$

By determining the number of mutational changes by sequence alignment methods, a quantitative measure can be obtained of the distance between any pair of sequences. These values can be used to reconstruct a phylogenetic tree, which describes a relationship between the gene sequences. The more mutations required changing one sequence into the other, the more unrelated the sequences and the lower the probability that they share a recent common ancestor sequence. Conversely, the more alike a pair of sequences, the fewer the number of changes

required to change one sequence into the other, and the greater the likelihood that they share a recent common ancestor sequence.

Distances between DNA sequences are relatively simple to compute as the sum between two sequences, that is the least number of steps required to change one sequence into the other (D = X + Y). It is preferable in phylogenetic tree analysis because (i) the pattern of mutations, insertions and deletions at the nucleotide level are definitive and (ii) silent mutations at the DNA level do not result in an amino acid substitution (because of the redundancy of the genetic code). A simple matrix of the frequencies of the 12 possible types of replacement (each base can be replaced by any of the three other bases) can be used. Differences due to insertions/deletions are generally given a large score than substations.

Simple similarity matrix for DNA:

The simplest matrix in use is the identity (similarity) matrix. If two letters are the same they are given +1 and 0 if they are not the same.

Simple similarity matrix for DNA:

	A	*T*	*G*	*C*
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

Simple distance matrix for DNA:

	A	*T*	*G*	*C*
A	0	5	5	1
T	5	0	1	5
G	5	1	0	5
C	1	5	5	0