

Sequence Comparison: Pairwise Alignment

Shifra Ben-Dor

Irit Orr



PAIRWISE ALIGNMENT



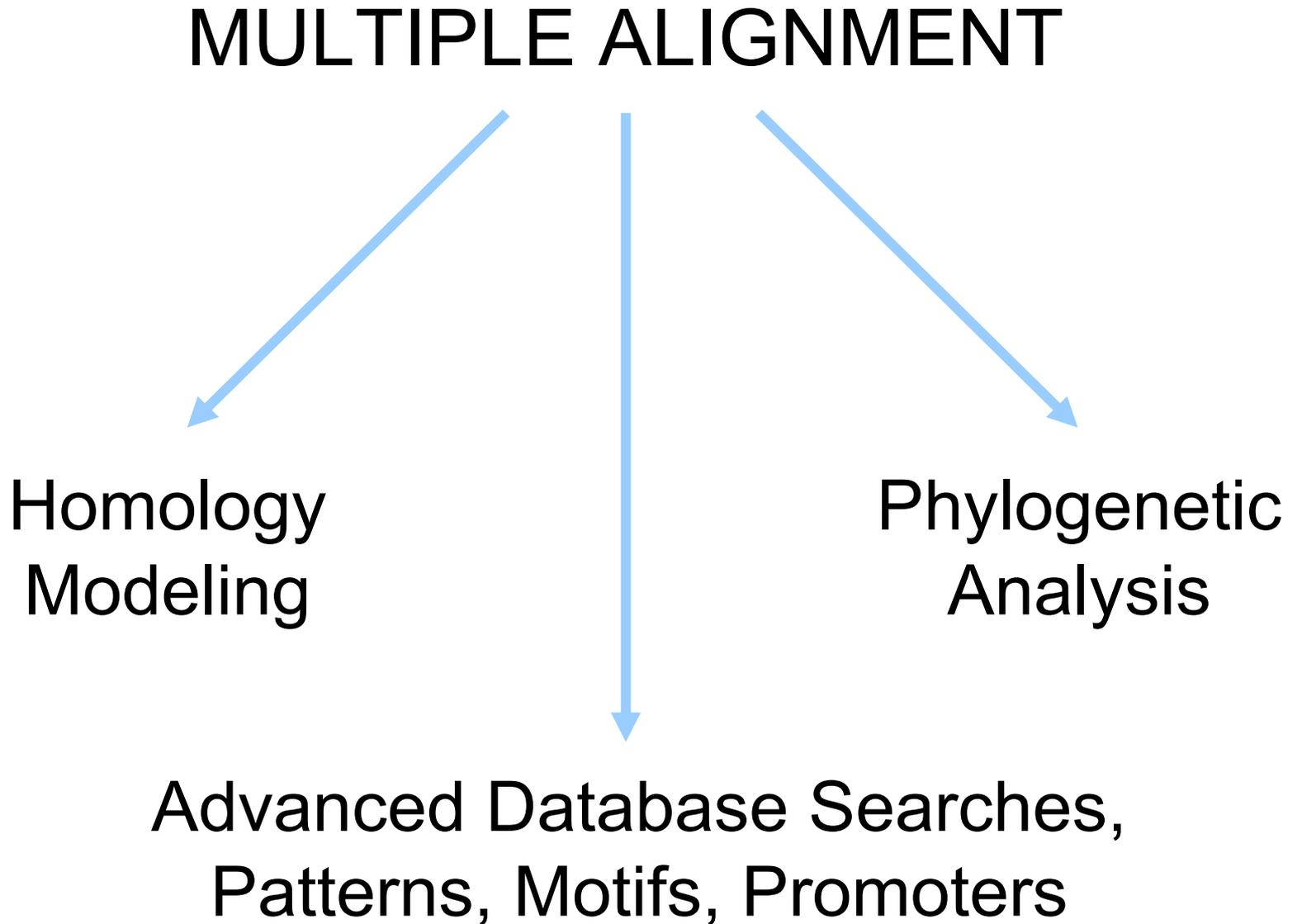
DATABASE SEARCHING



MULTIPLE ALIGNMENT



MULTIPLE ALIGNMENT



```
graph TD; A[MULTIPLE ALIGNMENT] --> B[Homology Modeling]; A --> C[Advanced Database Searches, Patterns, Motifs, Promoters]; A --> D[Phylogenetic Analysis];
```

Homology
Modeling

Advanced Database Searches,
Patterns, Motifs, Promoters

Phylogenetic
Analysis



The problems:

I have a DNA sequence: What does it do?

possible coding region

possible regulatory region

I have a protein sequence:

What does it do??

Sequence Comparison

- Generally, sequence determines structure and structure determines function
- By studying sequence similarity, we hope to find correlations between our sequence and other sequences with known structure or function
- This approach is often successful, however many molecules have low sequence similarity, yet still share similar structure or function.



Sequence Comparison

- Motifs / Domains - similarity over small stretches
- Sequence families - similarity over longer sequences
- Comparison can help us with:
 - structure
 - function
 - evolution



Comparison Questions:

- Are the sequences related (homology)?
- Can we quantify their similarity?
- Do they have similar segments?



Terminology:

- Homology
- Identity
- Similarity



Homology

- Common ancestry
- Sequence (and usually structure) conservation
- Homology is not a measurable quantity
- Homology can be inferred, under suitable conditions



Identity

- Objective and well defined
- Can be quantified by several methods:
 - Percent
 - The number of identical matches divided by the length of the aligned region



Similarity

- Most common method used
- Not so well defined
- Depends on the parameters used (alphabet, scoring matrix, etc.)



What are we comparing?

- DNA or RNA
 - Four nucleic acids (basic set)
- Protein
 - Twenty amino acids (basic set)



Alignment

- An alignment is an arrangement of two sequences opposite one another
- It shows where they are different and where they are similar
- We want to find the optimal alignment - the most similarity and the least differences



Alignment

- Alignments have two aspects:
 - Quantity: To what degree are the sequences similar (percentage, other scoring method)
 - Quality: Regions of similarity in a given sequence



The optimal alignment of two sequences is one that finds the longest segment of high sequence similarity.



How is an alignment done?

- When we compare sequences, we take two strings of letters (nucleotides or amino acids) and align them.
- Where the characters are identical, we give them a positive score, and where they differ, a negative value.
- We count the identical and non-identical characters, and give the alignment a score (usually called the quality)



Differences in the sequence can be caused by deletions or insertions in the DNA, or by point mutations. These changes can be seen at the protein level as well (changes in the translation of the protein)



This scheme works fine as long as you assume that all possible mutations occur at the same frequency.

However, nature doesn't work this way.

It has been found that in DNA, transitions occur more often than transversions.



Purines (A,G) are 2-ring bases

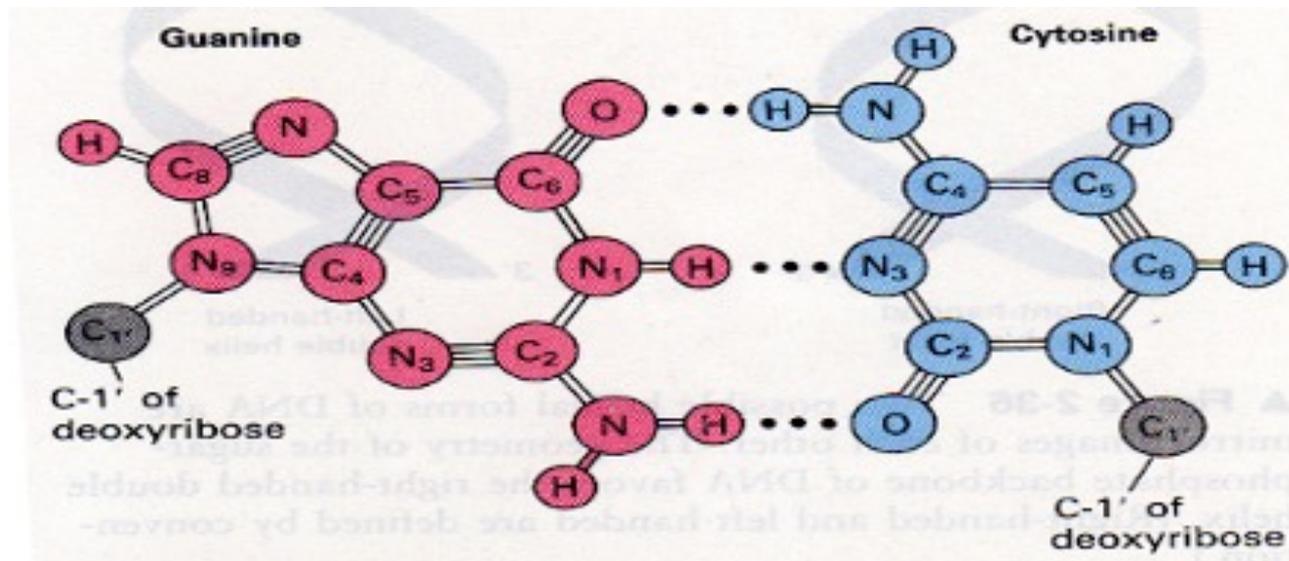
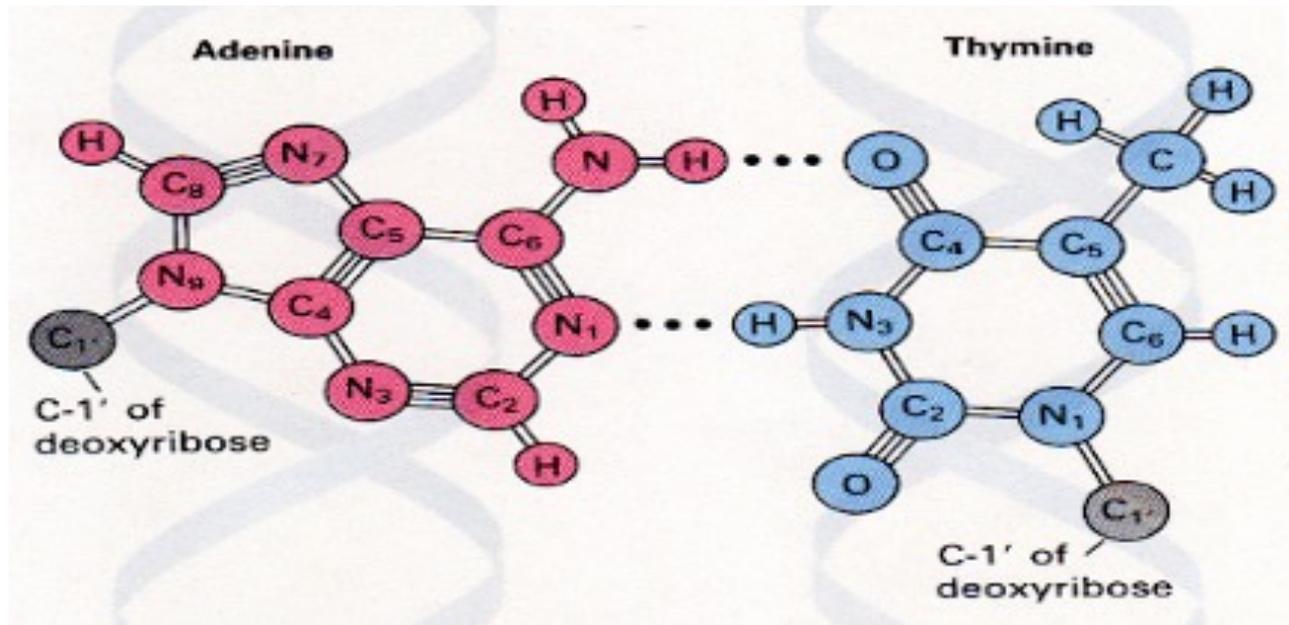
Pyrimidines (C,T) are 1-ring bases

Transition: purine to purine or
pyrimidine to pyrimidine

Transversion: purine to pyrimidine or
pyrimidine to purine

Transitions conserve ring number

Transversions change ring number



taken from Molecular Cell Biology, Darnell Lodish Baltimore 1990



For proteins, the situation is far more complex

- Amino acids can be grouped by a number of classifications:
 - Chemical: aromatic, aliphatic, sulphuric
 - Functional: hydrophobic, hydrophilic, acidic, basic
 - Charge: positive, negative, neutral
 - Structural: internal, external



Scoring Matrices

- Scoring matrices are used to assign a score to each comparison of a pair of characters
- The scores in the matrix are integer values which assign a positive score to identical or similar character pairs, and a negative value to dissimilar pairs
- The matrices were constructed by analyzing known families of proteins

An example: Blosum62

Henikoff & Henikoff

	A	B	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	X	Y	Z
A	4	-2	0	-2	-1	-2	0	-2	-1	-1	-1	-1	-2	-1	-1	-1	1	0	0	-3	-1	-2	-1
B	-2	6	-3	6	2	-3	-1	-1	-3	-1	-4	-3	1	-1	0	-2	0	-1	-3	-4	-1	-3	2
C	0	-3	9	-3	-4	-2	-3	-3	-1	-3	-1	-1	-3	-3	-3	-3	-1	-1	-1	-2	-1	-2	-4
D	-2	6	-3	6	2	-3	-1	-1	-3	-1	-4	-3	1	-1	0	-2	0	-1	-3	-4	-1	-3	2
E	-1	2	-4	2	5	-3	-2	0	-3	1	-3	-2	0	-1	2	0	0	-1	-2	-3	-1	-2	5
F	-2	-3	-2	-3	-3	6	-3	-1	0	-3	0	0	-3	-4	-3	-3	-2	-2	-1	1	-1	3	-3
G	0	-1	-3	-1	-2	-3	6	-2	-4	-2	-4	-3	0	-2	-2	-2	0	-2	-3	-2	-1	-3	-2
H	-2	-1	-3	-1	0	-1	-2	8	-3	-1	-3	-2	1	-2	0	0	-1	-2	-3	-2	-1	2	0
I	-1	-3	-1	-3	-3	0	-4	-3	4	-3	2	1	-3	-3	-3	-3	-2	-1	3	-3	-1	-1	-3
K	-1	-1	-3	-1	1	-3	-2	-1	-3	5	-2	-1	0	-1	1	2	0	-1	-2	-3	-1	-2	1
L	-1	-4	-1	-4	-3	0	-4	-3	2	-2	4	2	-3	-3	-2	-2	-2	-1	1	-2	-1	-1	-3
M	-1	-3	-1	-3	-2	0	-3	-2	1	-1	2	5	-2	-2	0	-1	-1	-1	1	-1	-1	-1	-2
N	-2	1	-3	1	0	-3	0	1	-3	0	-3	-2	6	-2	0	0	1	0	-3	-4	-1	-2	0
P	-1	-1	-3	-1	-1	-4	-2	-2	-3	-1	-3	-2	-2	7	-1	-2	-1	-1	-2	-4	-1	-3	-1
Q	-1	0	-3	0	2	-3	-2	0	-3	1	-2	0	0	-1	5	1	0	-1	-2	-2	-1	-1	2
R	-1	-2	-3	-2	0	-3	-2	0	-3	2	-2	-1	0	-2	1	5	-1	-1	-3	-3	-1	-2	0
S	1	0	-1	0	0	-2	0	-1	-2	0	-2	-1	1	-1	0	-1	4	1	-2	-3	-1	-2	0
T	0	-1	-1	-1	-1	-2	-2	-2	-1	-1	-1	-1	0	-1	-1	-1	1	5	0	-2	-1	-2	-1
V	0	-3	-1	-3	-2	-1	-3	-3	3	-2	1	1	-3	-2	-2	-3	-2	0	4	-3	-1	-1	-2
W	-3	-4	-2	-4	-3	1	-2	-2	-3	-3	-2	-1	-4	-4	-2	-3	-3	-2	-3	11	-1	2	-3
X	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Y	-2	-3	-2	-3	-2	3	-3	2	-1	-2	-1	-1	-2	-3	-1	-2	-2	-2	-1	2	-1	7	-2
Z	-1	2	-4	2	5	-3	-2	0	-3	1	-3	-2	0	-1	2	0	0	-1	-2	-3	-1	-2	5



Alignment algorithms

- Visual alignment
 - allows integration of relevant data not available to computerized algorithms
 - Time consuming, not feasible for all but the shortest sequences
- Fixed length algorithms
 - do not consider insertions and deletions
 - insertions and deletions are needed even for closely related sequences



Alignment Algorithms

- The naïve approach:
 - generate all possible alignments for 2 sequences (including gaps) and choose the alignment with the highest score
 - Too time consuming



Dynamic programming algorithms

- Each character along both sequences is evaluated. At each position there are four possibilities:
 - identity
 - substitution
 - deletion in sequence 1
 - deletion in sequence 2



Dynamic programming

- Identical characters (matches) or substitutions (mismatches) are scored according to a matrix.
- Deletions in either of the sequences are called gaps.
- Gaps are given a negative score, referred to as the gap penalty



The alignment is given a score, called the quality

Quality = matches - (mismatches + gap penalty)

The program will find the alignment with the highest quality.

The choice between gaps and substitutions is made to give the higher quality of the two.

The Gap Penalty

Consider the two following alignments:

V I T K L G T C V G S

V I T K L G T C V G S

V I T . . . T C V G S

V . T K . G T C V . S

According to the algorithm these 2 cases will get the same gap penalty:

Match = 3

Gap = -2

$$8(3) + 3(-2) = 18$$

$$8(3) + 3(-2) = 18$$



However nature is different. In most cases insertions/deletions are longer than a single residue, even for very similar sequences.



To compensate for this, and to differentiate between cases like the one above, the gap penalty is made up of two factors:

The gap creation penalty - subtracted from the alignment quality whenever a gap is opened.

The gap extension penalty - subtracted from the alignment quality according to the length of the gap.



Thus we have:

Quality =

matches - (mismatches + gap penalty)

Gap penalty =

gap creation penalty +
(gap extension penalty X gap length)

The Gap Penalty

So now we have:

V I T K L G T C V G S

V I T . . . T C V G S

V I T K L G T C V G S

V . T K . G T C V . S

Match = 3

Gap open = - 4

Gap extension = -1

$$8(3) + [1(-4) + 3(-1)] = 17$$

$$8(3) + [3(-4) + 3(-1)] = 9$$



Gap penalty parameters

Insertion of a gap must improve the quality of the alignment (raise the quality score).

If the gap creation and gap extension penalties are high, less gaps will be inserted into the alignment.

If the gap creation and gap extension penalties are low, more gaps will be inserted into the alignment.



So if you are interested in an alignment between two very similar sequences the gap penalties should be raised, to reduce the chances of getting something random.

If you are interested in detecting homology (finding a weak similarity) between two distantly related sequences the gap penalties should be lowered.

If you don't know what to expect, start off with the default parameters



To summarize:

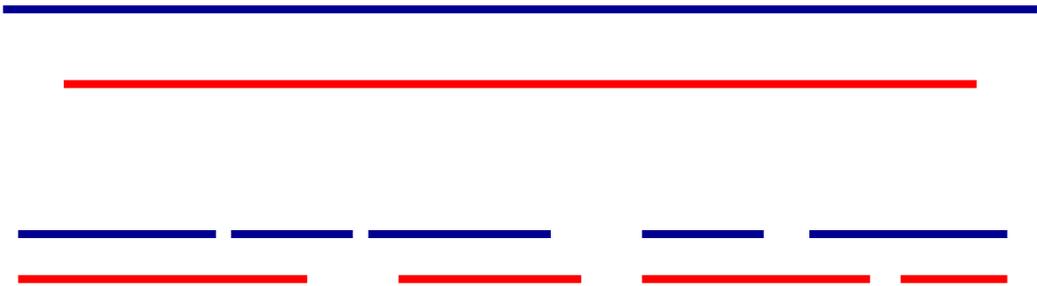
- Alignment scores are dependent on what we choose for: matches, mismatches, substitutions and gaps.
- Dynamic programming can be used for global or local alignment



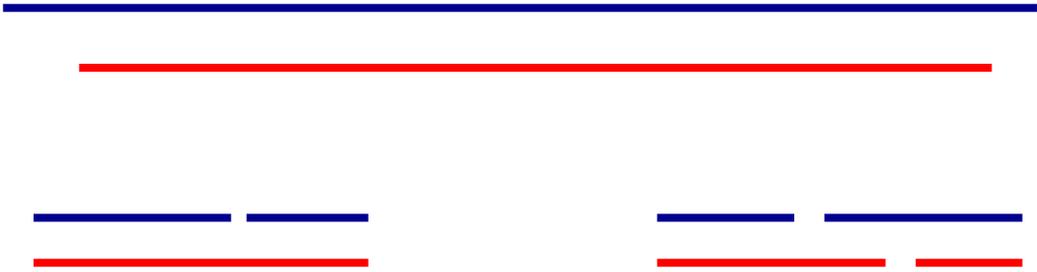
Two types of alignment:

- Global alignment
- Local alignment

Global alignment



Local alignment





Global alignment

A **global** pairwise alignment is one where it is assumed that the two sequences have diverged from a common ancestor and that the program should try to stretch the two sequences, introducing gaps where necessary, in order to show the alignment over the whole length of the two sequences that best illustrates their similarities.



Global alignment

- Compares sequences and gives best overall alignment
- May fail to find the best local region of similarity (such as a shared motif) among distantly related sequences
- Will (generally) return only the best matching segment for a given pair of sequences



Global alignment – End Gaps

- Since a global alignment can only give one overall output, the question arises of how we deal with overhanging ends, also known as ‘end gaps’
- There is an optional penalty for end gaps in most global alignment programs, though they are not necessarily on by default



Global alignment

- The classical algorithm for global alignment is the Needleman-Wunsch

A general method applicable to the search for similarities in the amino acid sequence of two proteins.

Needleman SB, Wunsch CD

J Mol Biol 1970 Mar;48(3):443-53



Local Alignment

- Searches for **regions of local similarity** between two sequences and need not include the entire length of the sequences.
- Finds regions of (ungapped) sequence with a high degree of similarity
- Better at finding motifs, especially for sequences that are different overall
- Can return more than one matching segment for a given pair of sequences



Local Alignment

- The classical algorithm for local alignment is the Smith-Waterman

Identification of common molecular subsequences

Smith TF, Waterman MS

J Mol Biol 1981 Mar 25;147(1):195-7



Sequence Comparison Programs

- Global
 - Needle (EMBOSS)
 - Stretcher (EMBOSS) – modified to conserve memory, good for long sequences



Sequence Comparison Programs

- Local
 - Lalign (Fasta) – can return more than one segment
 - Matcher (EMBOSS) - based on lalign, can return more than one segment
 - Water (EMBOSS) - Smith-Waterman, only one hit
 - Bl2Seq – Blast 2 sequences



Local pairwise alignment using BL2SEQ at NCBI

- This tool produces the alignment of two given sequences using **BLAST** algorithm for local alignment.
- Reference:
Tatiana A. Tatusova, Thomas L. Madden (1999),
"Blast 2 sequences - a new tool for comparing
protein and nucleotide sequences", FEMS
Microbiol Lett. 174:247-250



Local pairwise alignment using BL2SEQ

- This tool utilizes the BLAST engine for pairwise sequence comparison and is based on the same algorithm and statistics of local alignments that have been described in the BLAST paper.
- The BLAST algorithm generates a gapped alignment by using dynamic programming to extend the central segment of aligned residues.
- Because the parameters were based on database searching, some may have to be changed to find a match



Statistical Evaluation of Alignments

The problem with these programs is no matter how dissimilar the sequences you compare, the programs will always align them.

Even a 5% identity will be displayed as a valid result.

So how can you tell if the alignment is statistically valid???



Statistics by randomization

- A program will take the second sequence you input and shuffle it, to obtain a random sequence with the same character composition.
- This random sequence will be compared to the first sequence, using either a global or local algorithm (the same that you used originally), and a quality score will be obtained.



Randomization

- This process is repeated many times, (number of times generally specified by the user) in order to obtain a population of sequences that can be used for statistical analysis.
- The quality of these alignments is plotted in a distribution and compared to the original quality, and then be used to give a statistically meaningful answer to the alignment.



In the FASTA package, the PRSS program can perform shuffling of sequences

It can be done uniformly throughout the sequence, or using windows (which is useful if there are non-random windows in a sequence, like a transmembrane domain, which will be skewed towards hydrophobic amino acids).



Dotplots

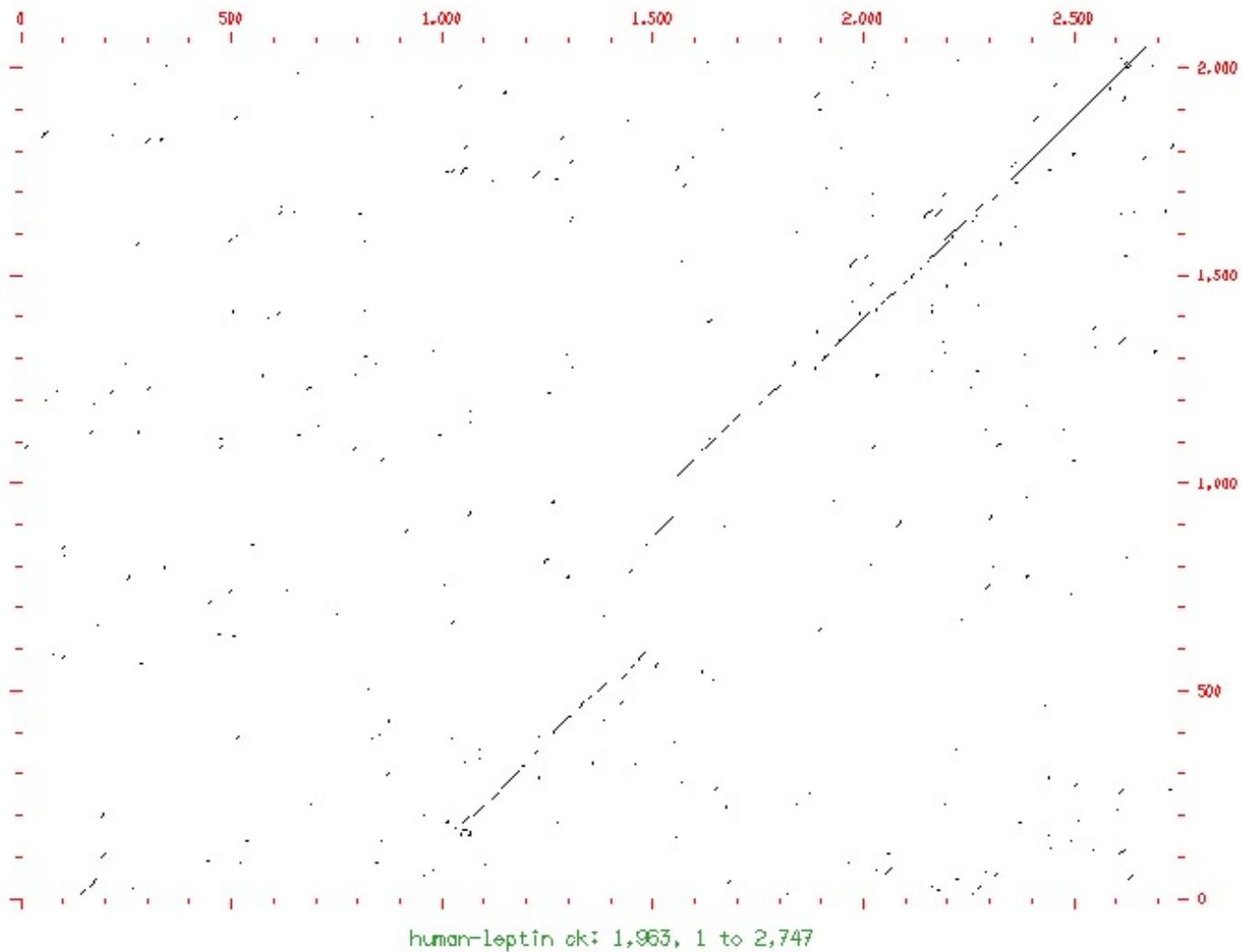
Dot plots are two dimensional graphs, showing a comparison of two sequences.

The two axes of the graph represent the two sequences being compared.

Every region of the sequence is compared to every region of the other sequence.

DOTPLOT Density: 2343.18 May 14, 2006 09:01
COMPARE Window: 21 Stringency: 14 Points: 1,714

cow-leptin ck: 5,544, 1 to 2,060





Dotplots

Dotplotting is the best way to see all of the structures in common between two sequences.

Dotplotting can also be used to view repeated structures or inverted repeats in a single sequence. This is accomplished by comparing a sequence to itself.

Dotplotting helps recognize large regions of similarity. In most cases it is not sensitive enough to see small structures.



Comparison Criteria

The match criterion can be met in two different ways:

The window/stringency method.

The word method.



The window/stringency method

Searches for all the places where a given number of matches (stringency) occur within a given range (window).

This method is more time-consuming, but more sensitive.

Comparisons are done according to a scoring matrix.



The word method

Must be specified on the command line
(`-wordsize=X`, where X is the size you choose).

Searches for short perfect matches of a set length
(words).

This method is about 1000 times faster than the
window/stringency method, but is much less
sensitive.

If the sequences do not contain short perfect
matches then this method will find nothing.



Hints

If you have long sequences, try a word comparison first. This is much faster, and will give you an idea of what the dot plot for the more sensitive window/stringency method will look like.

When using the word method, start off with a word size of 6 for nucleic acid sequences of up to 1,000 bases, or 8 for sequences of up to 10,000.



Hints

For peptide sequences, start off with a word size of 2-3.

When using the window/stringency method start off with a window of 21 and a stringency of 14 for nucleic acids.

For peptide sequences start off with a window of 30 and a stringency of 11.



Programs for dotplots

- FASTA

- PLALIGN

- EMBOSS

- Dotmatcher - window/stringency
- Dottup - word plot
- Dotpath- non-overlapping word plot
- Polydot - all against all word plot



Alternative “dotplots”

Dotter is a graphical dotplot program for detailed comparison of two sequences.

To make the score matrix more intelligible, the pairwise scores are averaged over a sliding window which runs diagonally. The averaged score matrix forms a three-dimensional landscape, with the two sequences in two dimensions and the height of the peaks in the third.



This landscape is projected onto two dimensions by aid of greyscales - the darker grey of a peak, the higher it is.

Dotter provides a tool to explore the visual appearance of this landscape, as well as a tool to examine the sequence alignment it represents.