Principles of Sequence Similarity

M. Saleet Jafri

BINF 630 – Lecture 3

DNA Sequence Alignment – Why?

- Recognition sites might be common – restriction enzymes, start sequences, stop sequences, other regulatory sequences
- Homology – evolutionary common progenitor

  Mutations

  - Insertions
  - Deletions
  - Subsititutions

Protein sequence alignment

- Homologous proteins
  - Evolutionary common origin
  - Structural similarity
  - Functional similarity
- Conserved regions
  - Functional domains
  - Evolutionary similarity
  - Structural motif

Two different sequence alphabets

- DNA alphabet: A,C,G,T
- Four discrete possibilities - it's either a match or a mismatch
- Protein alphabet: A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y
- 20 possibilities which fall into several categories - residues can be similar without being identical

Types of Sequence Alignment

- Pairwise Alignment – compare two sequences
- Multiple Alignment – compare one sequence to many others

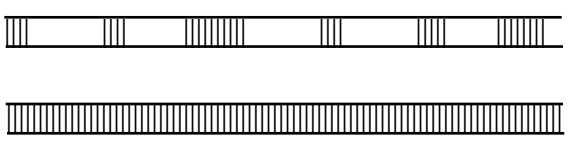
For each of the above we can do

- Local Alignment – compare similar parts of two sequences
- Global Alignment – compare the whole sequence

For the different types of alignments there are different assumptions and methods.

Global Alignment versus Local Alignment

- Local alignment: finds continuous or gapped high-scoring regions which do not span the entire length of the sequences being aligned
- Global alignment: finds the optimal full-length alignment between the two sequences being aligned
- In general, local alignment is used for database searching.



What is "sequence homology"?

- A qualitative statement
- Derived from quantitative sequence similarity data
- Assertion that two genes share a common evolutionary history
- Genes either are homologous, or they are not - there are no degrees of homology.

What is "sequence identity/similarity"?

- A quantitative measurement of the number of residues which are identical in both of the sequences being aligned
- Calculated from a sequence alignment
- Can be expressed as a percentage
- The term "sequence similarity" may also be used, especially in proteins, where the larger amino acid alphabet means that some residues are chemically similar but not identical.

Example

Start with ACGTACGT after 9540 generations with the following probabilities:

Deletion 0.0001
Insertion 0.001
Transitional substitution 0.00008
Translational substitution 0.00002

ACG – T-A - - - CG -T - - - -
ACGGTCCTAATAATGGCC

- - - AC - GTA- C- - G - T - -
CAG - GAAGATCTTAGTTC

Example (continued)

However, if we align the two sequences by superposition

- ACAC - GGTCCTAAT- - AATGGC
CAG- GAA- G- AT- - CTTAGTTC- -

or using Gotoh's algorithm with mismatch penalty 3 and gap penalty function $g(k) = 2+2k$ for length k gap

ACACG - - GTCCTAATAATGGCC
- CAGGAAGATCT - - TAGTT - - C

The alignment depends on algorithm used!

Choosing the Optimal Alignment

As shown before there are many possible alignments – which is correct?

- Every alignment has a score
- Chose alignment with highest score
- Must choose appropriate scoring function
- Scoring function based on evolutionary model with insertions, deletions, and substitutions
- Use substitution score matrix – contains an entry for every amino acid pair

Comparing Sequences

- Scoring Matrices

  Substitution score matrices - PAM (Point [or Percent] Accepted Mutation), BLOSSUM, etc

- Distance between sequences

  Minimize distance between sequence – Dynamic Programming

- Similarity between sequences

  Maximize similarity between sequences

The evolutionary basis of sequence comparison

- The simplest molecular mechanisms of evolution are substitution, insertion, and deletion.
- If a sequence alignment represents the evolutionary relationship of two sequences, residues that are aligned but do not match equal substitutions.
- Residues that are aligned with a gap in the sequence represent insertions or deletions.
- Back-substitutions are ignored because there is no way of knowing when and where they occurred.

Creating Scoring Matrices

- Ad hoc method - a biologist can set up a score matrix that gives good alignment
- Use physical/chemical properties – similarities between amino acids
- Statistical approach – need to pick appropriate evolutionary model, PAM and BLOSSUM

Substitution matrices

A substitution or scoring matrix is used to evaluate possible matches and to choose the best match between two sequences

	A	C	G	T
A	3	0	0	0
C	0	3	0	0
G	0	0	3	0
T	0	0	0	3

A	3			
C	0	3		
G	0	0	3	
T	0	0	0	3
	A	C	G	T

Unitary Matrix

PAM matrix

Problem:

To construct the PAM matrix Dayhoff and co-workers were faced with a dilemma. In order to find a good substitution matrix, you had to compare two sequences, but you needed a substitution matrix to do the comparison.

Solution:

Consider only closely related sequences (<15% difference) when making the scoring matrix.

This is good for closely related sequences.

PAM matrix

Problem:

What do you use for more distantly related proteins.

Solution:

Take evolutionary time and create matrices by multiplying the PAM matrix by itself N times where N is the number of PAM evolutionary time units that have passed

Hence the PAM250 matrix is used for distantly related proteins.

What do the scores in the matrices represent?

- **Overall, the alignment program is evaluating the likelihood that an alignment is significant, rather than random**
- **Each individual score is the logarithm of the ratio:**

$$\frac{\text{probability of meaningful occurrence of a residue pair}}{\text{probability of random occurrence}}$$

LOG ODDS

PAM Substitution matrices

- **Point Accepted Mutation (Dayhoff et al 1978)**
- **Closely related protein alignment**
- **1 PAM = 1% change**
- **Log Odds: natural log of $\frac{\text{target frequency}}{\text{background frequency}}$**

- **PAM 120: closely related proteins**
- **PAM 250: highly divergent proteins**

PAM 250

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-2	-2	0
R	-2	7	0	-1	-3	1	0	-2	0	-3	-2	3	-1	-2	-2	-1	-1	-2	-1	-2
N	-1	0	6	2	-2	0	0	0	1	-2	-3	0	-2	-2	-2	1	0	-4	-2	-3
D	-2	-1	2	7	-3	0	2	-1	0	-4	-3	0	-3	-4	-1	0	-1	-4	-2	-3
C	-1	-3	-2	-3	12	-3	-3	-3	-3	-3	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	6	2	-2	1	-2	-2	1	0	-4	-1	0	-1	-2	-1	-3
E	-1	0	0	2	-3	2	6	-2	0	-3	-2	1	-2	-3	0	0	-1	-3	-2	-3
G	0	-2	0	-1	-3	-2	-2	7	-2	-4	-3	-2	-2	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	0	-3	1	0	-2	10	-3	-2	-1	0	-2	-2	-1	-2	-3	2	-3
I	-1	-3	-2	-4	-3	-2	-3	-4	-3	5	2	-3	2	0	-2	-2	-1	-2	0	3
L	-1	-2	-3	-3	-2	-2	-2	-3	-2	2	5	-3	2	1	-3	-3	-1	-2	0	1
K	-1	3	0	0	-3	1	1	-2	-1	-3	-3	5	-1	-3	-1	-1	-1	-2	-1	-2
M	-1	-1	-2	-3	-2	0	-2	-2	0	2	2	-1	6	0	-2	-2	-1	-2	0	1
F	-2	-2	-2	-4	-2	-4	--3	-3	-2	0	1	-3	0	8	-3	-2	-1	1	3	0
P	-1	-2	-2	-1	-4	-1	0	-2	-2	-2	-3	-1	-2	-3	9	-1	-1	-3	-3	-3
S	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-1	-1	4	2	1	-2	-1
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-1	-1	2	5	-3	-1	0
W	-2	-2	-4	-4	-5	-2	-3	-2	-3	-2	-2	-2	-2	1	-3	-4	-3	15	3	-3
Y	-2	-1	-2	-2	-3	-1	-2	-3	2	0	0	-1	0	3	-3	-2	-1	3	8	-1
V	0	-2	-3	-3	-1	-3	-3	-3	-3	3	1	-2	1	0	-3	-1	0	-3	-1	5

BLOSUM Substitution matrices

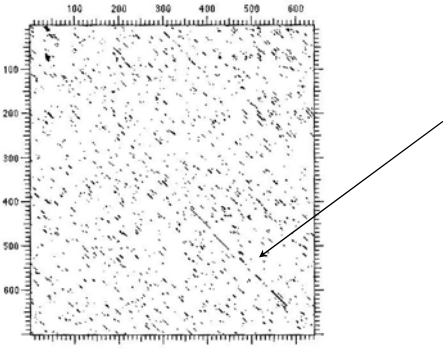
- **BLOCKS database (Henikoff & Henikoff 1991)**
- **Distantly related protein alignment**
- **Functional Motifs**
- **maximum %sequence identity that still contributes independently to model**

- **BLOSUM 90: closely related proteins**
- **BLOSUM 30: highly divergent proteins**

Dot Matrix Sequence Comparison

- Method for comparing two sequences
- Can be used to find direct or inverted repeats
- All possible matches shown – investigator picks significant ones

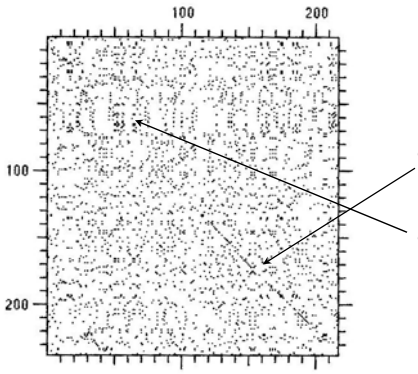
Dot plot



- Dot matrix analysis of human LDL receptor against DNA Strider (window=11, stringency=7)
- Dots on the diagonal indicate sequence similarity
- Horizontal or Vertical lines indicate repeated bases

From *Bioinformatics* by D. W. Mount

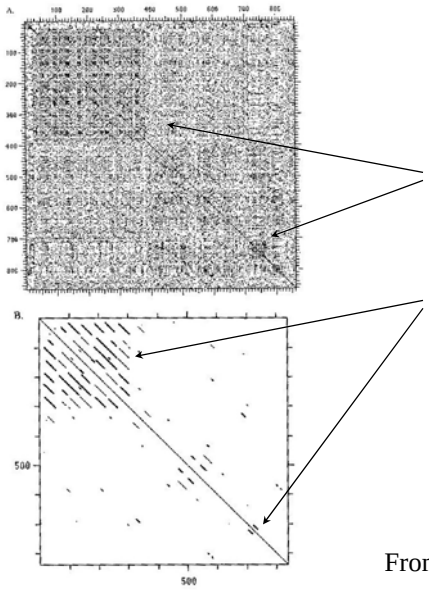
Dot plot



- Dot matrix analysis of human LDL receptor against DNA Strider (window=1, stringency=1)
- Dots on the diagonal indicate sequence similarity
- Horizontal or Vertical lines indicate repeated bases

From *Bioinformatics* by D. W. Mount

Dot Plots – Sequence Repeats



- Analyze a human LDL receptor sequence against itself.
- Top plot indicates repeats with high dot density (window=1, stringency=1)
- Bottom plot indicates repeats by diagonal lines off the diagonal (window=23, stringency=7)
- Note the overlap of high density dots and diagonal lines

From *Bioinformatics* by D. W. Mount

Pairwise Sequence Alignment

Typical operations:

- Find differences between two similar sequences
  - insertions, deletions, substitutions
  - may need to compare large sequences (over 10000 characters)
- Find local similarities
  - look for a few hundred characters in each string
  - need to identify partial matches
  - useful for searching large databases
- Is one sequence a prefix of another?
  - useful in DNA fragment assembly
- Find the similarities between two sequences with same evolutionary background

Gapped matching vs. ungapped matching

- Ungapped matching is less demanding than gapped matching. There is only one optimal way in which COMPARE and COMPLETE can be aligned without introducing gaps.
- Introducing gaps into either sequence means multiple permutations of the alignment are allowed
- Increase state or solution space

```
COMPARE      COMP-ARE     COMPARE
****         ****  *      ****  *
COMPLETE     COMPLETE     COMPL-ETE
```

Alignments

Given two sequences u and v, an alignment is a pair of sequences u' and v' such that:

1. u' is obtained from u by inserting gap character '-'
2. v' is obtained from v by inserting gap character '-'
3. u' and v' have same length: | u' | = | v' |
4. No position has gap characters in both u' and v'

Example:

u = ATGGCT
v = TGCTA
u' = ATGGCT-
v' = -TG-CTA

Goal: given two sequences, find the "best" alignment according some scoring function.

Dynamic Programming

- Compares two sequences and generates an alignment
- Alignment contains matched and mismatched characters as well as gaps
- Can be used for both local (Smith-Waterman) and global (Needleman-Wunch) alignments
- Generates an alignment score so that significance of or optimal alignment can be found
- Depends on choice of scoring system

Practical Considerations

- Goal of alignment will determine the type of scoring matrix used
- PAM based on model of evolutionary change
- BLOSUM are defined to identify members of the same family
- Different types of gap penalties

Concept of Distance or Similarity

- Distance
  - The distance between two sequences, based on an evolutionary model, describes when the two sequences had a common ancestor.
  - We want to minimize the distance.
- Similarity
  - The similarity between two sequence described how closely related two sequences are.
  - We want to maximize the similarity
- Either can be used and get the same result

Metrics

- Any notion of distance or similarity must be a metric
- A metric d must satisfy the following
  - D(x,y) = 0 if x = y
  - D(x,y) = D(y,x) (symmetry)
  - D(x,z) <= D(x,y) + D(y,z) (triangle inequality)
- The concept of distance between two points satisfies this (Euclidean distance or Euclidean metric)

How gapped matches are scored

- The scoring function is expanded to include a penalty for gaps
- The penalty value is generally chosen to be costly enough, in terms of the current scoring matrix, that adding a gap will not be too easy (resulting in meaningless alignments) or too difficult (resulting in no gaps).
- It costs less to extend a gap once it's opened than to open it in the first place.

```
ACGTAGTGT-CACT        -ACGTAGTGTCA-C-T
 *  **  **             * * * **  * * *
GAGA--TGAGCATG        GA-G-A-TG--AGCATG
```

Gap penalties

- Linear gap penalty function
- Subadditive gap penalty function
  $g(k+l) <= g(k) + g(l)$
- Affine gap penalty function
  $g(k) = a + kb$
- Different gap penalty at the ends of sequences

Steps to Dynamic Programming

- Compute the similarity/distance matrix for two sequences
- Perform the trace back to find the optimal alignment
- We can use distance or similarity and get the same result

Example: Consider the words a = AT and b = AAGT and a similarity score function s(x,y) = 0 if x ><y and w(x,x) =1

Example: Consider the words a = AT and b = AAGT and a cost score function d(x,y) = 1 if x ><y and w(x,x) = 0

- In these examples, the gap and mismatch penalty are equal.
- The text minimizes similarity in their algorithm

Needleman-Wunch Algorithm

$$D_{0,,0} = 0$$

$$D_{0,j} = \sum_{k=1}^{j} w(-, b_k)$$

$$D_{i,0} = \sum_{k=1}^{i} w(a_k, -)$$

$$\forall i, j > 0 \quad D_{i,j} = \min \begin{Bmatrix} D_{i,j-1} + w(-, b_j) \\ D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \end{Bmatrix}$$

where w is the weight of a gap

Needleman-Wunch Algorithm Example

Example: Consider the words a = AT and b = AAGT and a cost function s(x,y) = 1 if x ><y and w(x,x) =0

Needleman-Wunch Algorithm Example

		A	A	G	T
A					
T					

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1				
T	2				

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1				
T	2				

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1	0 2			
		2 0			
T	2				

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1	0 2			
		2 0			
T	2	2 1			
		3 1			

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3				
		2	0	1	1				
T	2	2	1						
		3	1						

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3				
		2	0	1	1				
T	2	2	1	1	2				
		3	1	2	1				

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4		
		2	0	1	1	2	2		
T	2	2	1	1	2				
		3	1	2	1				

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4		
		2	0	1	1	2	2		
T	2	2	1	1	2	2	3		
		3	1	2	1	2	2		

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3		
		3	1	2	1	2	2		

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

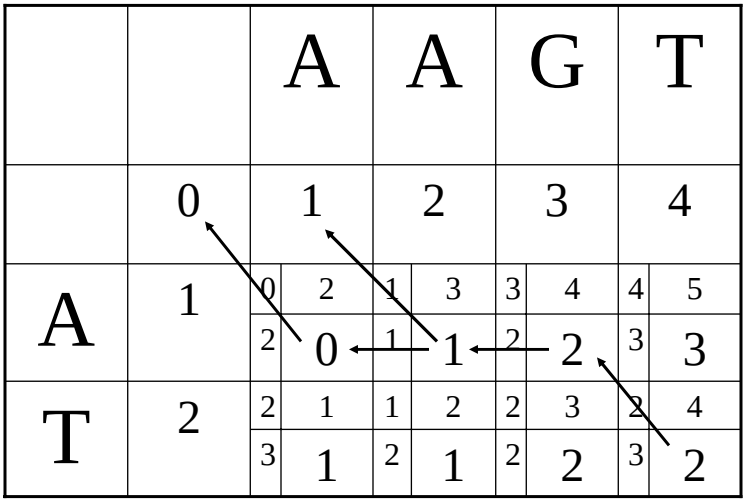
Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	3	4
		3	1	2	1	2	2	3	2

Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	3	4
		3	1	2	1	2	2	3	2

Traceback



Getting Alignment from Trace back

- The alignments can be determined from the traceback
- Horizontal arrows denote a gap in the sequence on the left
- Vertical arrows denote a gap in the sequence on the top
- Diagonal arrows denote a match if there is no penalty
- Diagonal arrows denote a mismatch if there is a penalty

Alignment

AAGT
– A –T

		A	A	G	T
	0 ←	1	2	3	4
A	1	0 2 | 1 3 | 3 4 | 4 5			
		2 | 0 ← | 1 ← | 2 ← | 3			
T	2	2 | 1 | 1 | 2 | 2 | 3 | 2 | 4			
		3 | 1 | 2 | 1 | 2 | 2 | 3 | 2			

Alignment

AAGT
A– – T

		A	A	G	T
	0	1	2	3	4
A	1	0 2 | 1 3 | 3 4 | 4 5			
		2 | 0 ← | 1 ← | 2 ← | 3			
T	2	2 | 1 | 1 | 2 | 2 | 3 | 2 | 4			
		3 | 1 | 2 | 1 | 2 | 2 | 3 | 2			

Waterman-Smith-Beyer Algorithm

$$D_{0,,0} = 0$$

$$D_{0,j} = g(j)$$

$$D_{i,0} = g(i)$$

$$\forall i, j > 0 \quad S_{i,j} = \min\begin{Bmatrix} \min_{1 \le k \le j}(D_{i,j-k} + g(k)) \\ D_{i-1,j-1} + w(a_i, b_j) \\ \min_{1 \le k \le j}(D_{i-k,j} + g(k)) \end{Bmatrix}$$

where $g(k)$ is the gap penalty function and
w is the similarity score function

Waterman-Smith-Beyer Algorithm Example

Example: Consider the words a = AT and b = AAGT and a cost function s(x,y) = 1 if substitution and s(x,x) =0. We will assume an affine gap penalty function g(k) = 1 + k

Waterman-Smith-Beyer Algorithm Example

		A		A		G		T	
	0	2		3		4		5	
A	2	0	4	2	5	4	6	5	7
		4	0	2	2	3 4	3	4	4
T	3	3	2	1	4	3	5	3	6
		5	2	4	1	3	3	4 5	3

Waterman-Smith-Beyer Algorithm Example

		A		A		G		T	
	0	2		3		4		5	
A	2	0	4	2	5	4	6	5	7
		4	0	2	2	3	3	4	4
T	3	3	2	1	4	3	5	3	6
		5	2	4	1	3	3	4	3

Alignment

AAGT
A– – T

		A	A	G	T
	0	2	3	4	5
A	2	0 4	2 5	4 6	5 7
		4 0	2 2	3 3	4 4
T	3	3 2	1 4	3 5	3 6
		5 2	4 1	3 3	4 3

Enhancements to Dynamic Programming

- Needleman and Wunch (1970) – global alignment
- Smith and Waterman (1980) – local alignment ie. alignment does not have to start at the ends
- Gotoh (1982) – decreased number of steps
- Waterman and Eggert (1987) – find alternative alignments ie., can start alignment in different places
- Myers and Miller (1988) – decreased memory required
- Schwartz (1991)– long sequence alignment
- Chao (1994) – near-optimal alignments

These methods are constantly evolving.

Significance of Alignment

- Dayhoff evaluated Needleman-Wunch alignment scores for many randomized and unrelated protein sequences using their log odds scoring matrix at 250 PAMs and a constant gap penalty.
- Result were normally distributed.
- For a score of an alignment to be significant, it must be at least 3-5 standard deviations greater than the mean of the random scores
- Caveats: computationally expensive and assumes random distribution of characters in alphabet.