

91學年度上學期「生物資訊學」課程

---

# Pairwise and Multiple Sequence Alignment

張傳雄

國立陽明大學 遺傳學研究所

09-23-2002



# Sequence Alignment vs. Similarity Searching

---

- Sequence alignment and similarity searching are different problems
- **Similarity searching** (Blast, Fasta, Ssearch)
  - Searching for homologies to elucidate the function of an unknown protein
  - Produces alignments, but the desired result is the **score**
- **Sequence alignment**
  - Searching for consensus sequence
  - Produces a score, but the desired result is the **sequence**

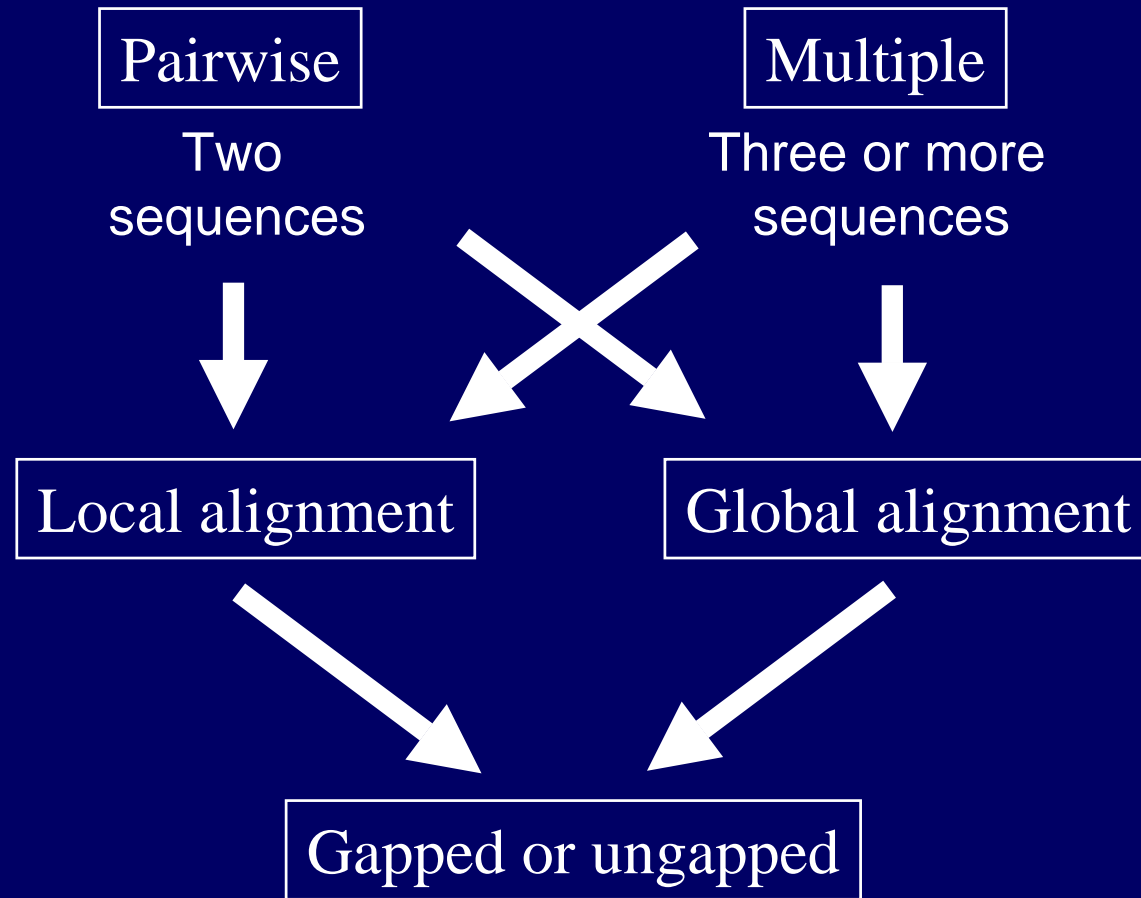
# Why do we want to align sequences ?

---

1. Assigning functions to unknown proteins
  2. Determine relatedness of organisms
  3. Identify structurally, functionally, evolutionarily similarities
  4. Make predictions about the 3D structure
- Because “similarity” may be an indicator of “homology” and thus provide some insight into function or gene identification.

# Types of Sequence Alignments

---



# Global alignment

---

## Global alignment

- match as many characters as possible from end to end
- find an alignment with **highest total score**
- regions of high local similarity may be ignored in favor of a **higher overall score**

### Example:

```
THIS-ISAGLALALIGNMENT
||  ||  ||  ||| |
THEREISTHEAL-  IGN-EDSEQ
```

# Local Alignment

---

## Local alignment

- find subsequences with highest density of matches
- find regions with **high local scores**
- sequence similarities may extend beyond the local subsequence with a lower degree of similarity

### Example:

```
-----LALIGNM-----  
          |||||  
-----EALIGNNE-----
```

- Local alignments are better for database searching and for finding similar domains since we can look for regions of similarity.

# Ungapped alignments

---

## Ungapped

- sequence comparisons are roughly proportional to the square of the average lengths

MATCHES

||

MAKERS

# Gapped alignments

---

## Gapped

If gaps of any lengths at any position would be allowed:

- computationally very expensive
- alignments would not be very meaningful

MATCHE-S

| |     | |

MA--KERS

Need a manageable number of gaps!

# Gap Penalties

---

- Inclusion of gaps and gap penalties is necessary in order to obtain the best alignment
- If gap penalty is too high, gaps will never appear in the alignment
- If gap penalty is too low, gaps will appear everywhere in the alignment
- Most alignment programs will suggest will suggest gap penalties that are appropriate for a given scoring matrix

# Gap penalties

---

- Reduce number of gaps in the alignment
- Ensure a more meaningful alignment
- **Opening** a gap is **costly**
- **Extending** a gap is **cheap**

Examples:

Gap opening penalty = 12

Gap extension penalty = 1

# Gap penalties

---

$$G = g + ln$$

$G$  = gap penalty

$g$  = cost of opening a gap

$l$  = cost of extending the gap by one

$n$  = length of the gap

\* In scoring matrices **gap penalties** are applied when not moving from  $i,j$  to  $i-1, j-1$

# Impact of gap penalties

---

**Case 1:** Gap penalty: **low** Mismatch cost: **high**

MARCHMADNESSANDBASKETBALL

-ARCHY-----ISA--BASKET-----CASE

**Case 2:** Gap penalty: **medium** Mismatch cost: **medium**

MARCHMADNESSANDBASKETBALL

-ARCHY-----ISA--BASKETCASE

**Case 3:** Gap penalty: **high** Mismatch cost: **low**

MARCHMADNESSANDBASKETBALL

-ARCHYISABASKETCASE

# Rules of thumb for gap penalties

---

- Gap **opening** penalty:  
should be 2 – 3 times larger than the most negative value in the substitution matrix that is being used
- Gap **extension** penalty:  
should be 0.1 to 0.3 times the value of the gap opening penalty

# Pairwise Sequence Alignments

---

```
HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHVDDMPNALSALSSDLHAHKL
            G+ +VK+HGKKV  A++++AH+D++ +++++LS+LH  KL
HBB_HUMAN  GNPKVKAHGKKVLGAFSDGLAHL DNLKGTFATLSELHCDKL

HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHV---D--DMPNALSALSSDLHAHKL
            ++ +++++H+ KV  + +A  ++                +L+ L+++H+ K
LGB2_LUPLU NNPELQAHAGKVFKL VYEAAIQLQVTGVVVTDATLKNLGSVHVS KG

HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHVDDMPNALSALSD----LHAHKL
            GS+ + G +   +D L  ++ H+ D+  A +AL D   ++AH+
F11G11.2   GSGYLVGDSLTFVDLL--VAQHTADLLAANAALLDEFPQFKAHQE
```



# Pairwise Alignment Methods

---

1. **Dot matrix analysis**  
(Gibbs and McIntyre)
2. **Dynamic programming algorithms**  
(Needleman-Wunsch, Smith-Waterman)
3. **Heuristic Algorithms** = Word or  $k$ -tuple methods  
(BLAST, FASTA)

# 1. Dot matrix analysis

---

## Advantages:

- **all** possible matches between 2 sequences are displayed
- readily reveals insertions & deletions
- readily identifies direct in inverted repeats
- same algorithm is used for DNA, RNA and proteins

## Disadvantages:

- doesn't show an actual sequence alignment
- qualitative evaluation of alignments
- statistical significance of alignment is not obvious

# Sources for dot matrix programs

---

- DNA Strider Mac
- MacVector Mac
- Dotlet Mac, Win, Unix  
<http://www.isrec.isb-sib.ch/java/dotlet/Dotlet.html>
- Dotter Unix  
<http://www.cgr.ki.se/cgr/gropus/sonnhammer/Dotter.html>
- EMBOSS  
<http://binf.ym.edu.tw/emboss/Apps/dotmatcher.html>
  - Dotmatcher
- GCG (Genetics Computer Group)  
<http://nun.oit.unc.edu/gcgmanual/>
  - Compare and Dotplot

# Compare and DotPlot

---

- Compare: calculation
  - Window/Strigency comparisons: use a scoring matrix and a moving window
  - Word comparison: use a hash-table or linked-list approach (perfect match)
- DotPlot: graphics

# Alignments & Scoring

---

## Global (e.g. haplotype)

ACCACACA

: : xx : : x :

ACACCATA

$$\text{Score} = 5(+1) + 3(-1) = 2$$

## Local (motif)

ACCACACA

: : : :

ACACCATA

$$\text{Score} = 4(+1) = 4$$

## Suffix (shotgun assembly)

ACCACACA

: : :

ACACCATA

$$\text{Score} = 3(+1) = 3$$

## Nucleic Acid

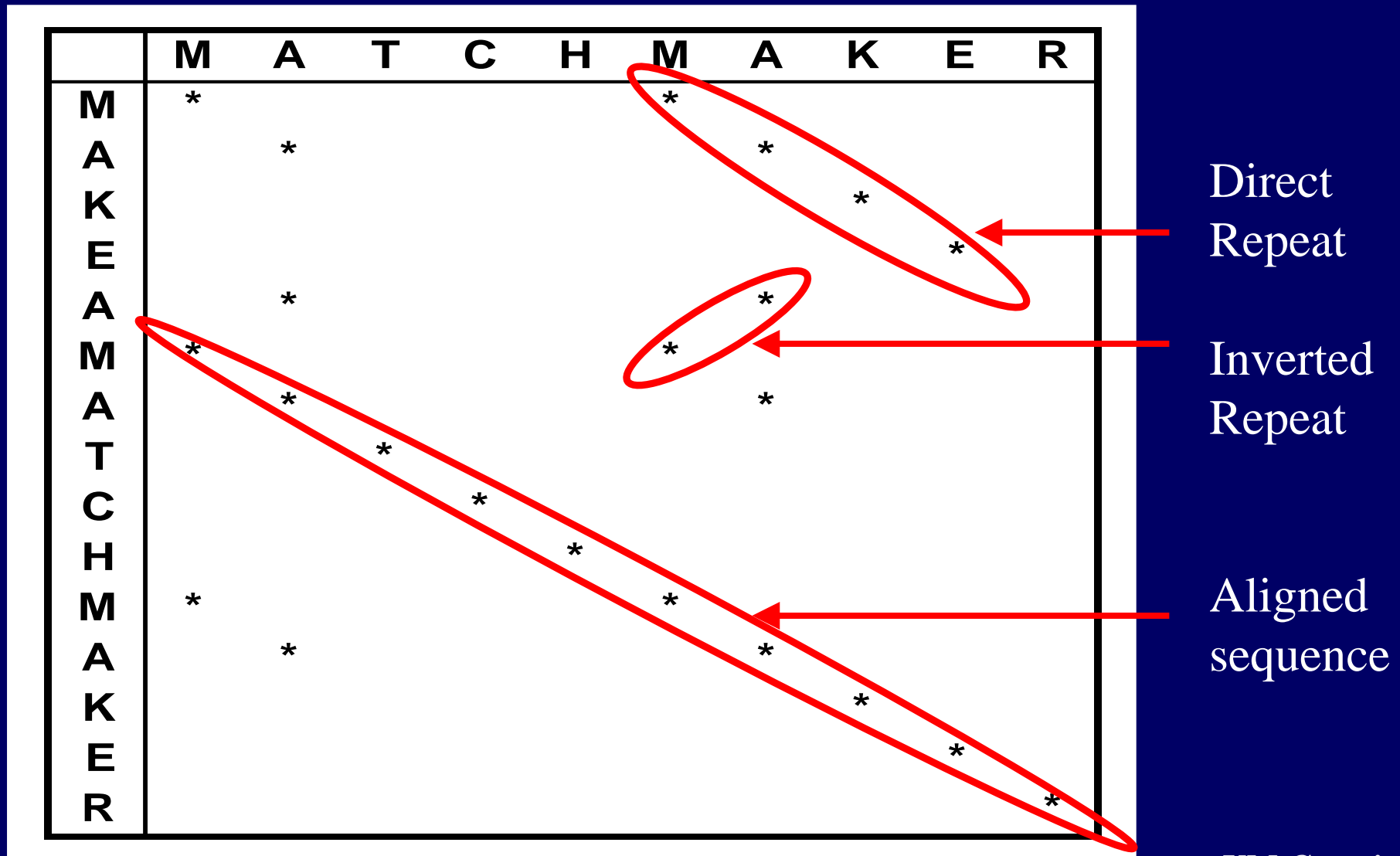
- Identical: 1
- Different: 0



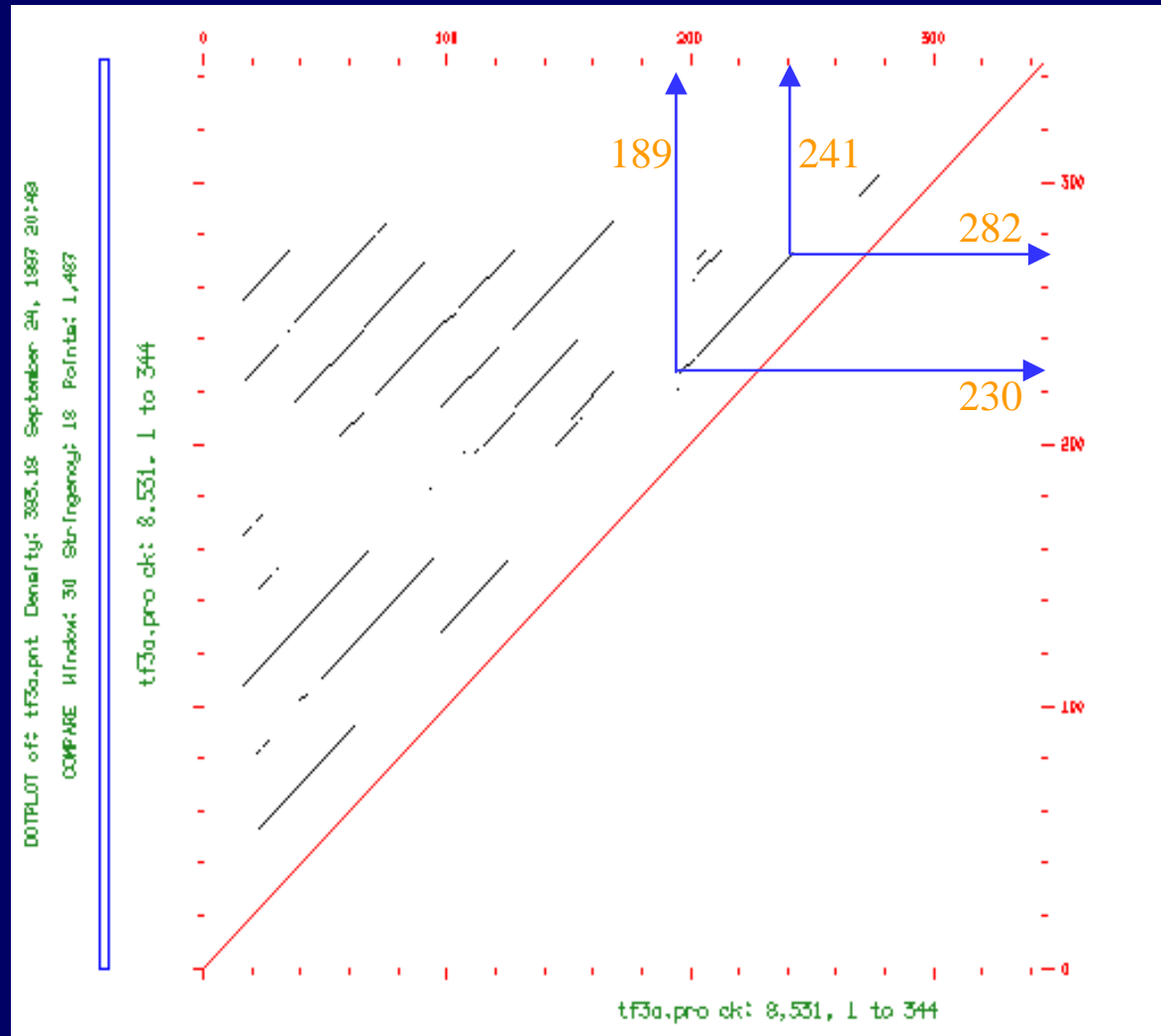
# How dot matrix analysis works

	M	A	T	C	H	M	A	K	E	R
M	*					*				
A		*					*			
K								*		
E									*	
A		*					*			
M	*					*				
A		*					*			
T			*							
C				*						
H					*					
M	*					*				
A		*					*			
K								*		
E									*	
R										*

# How dot matrix analysis works



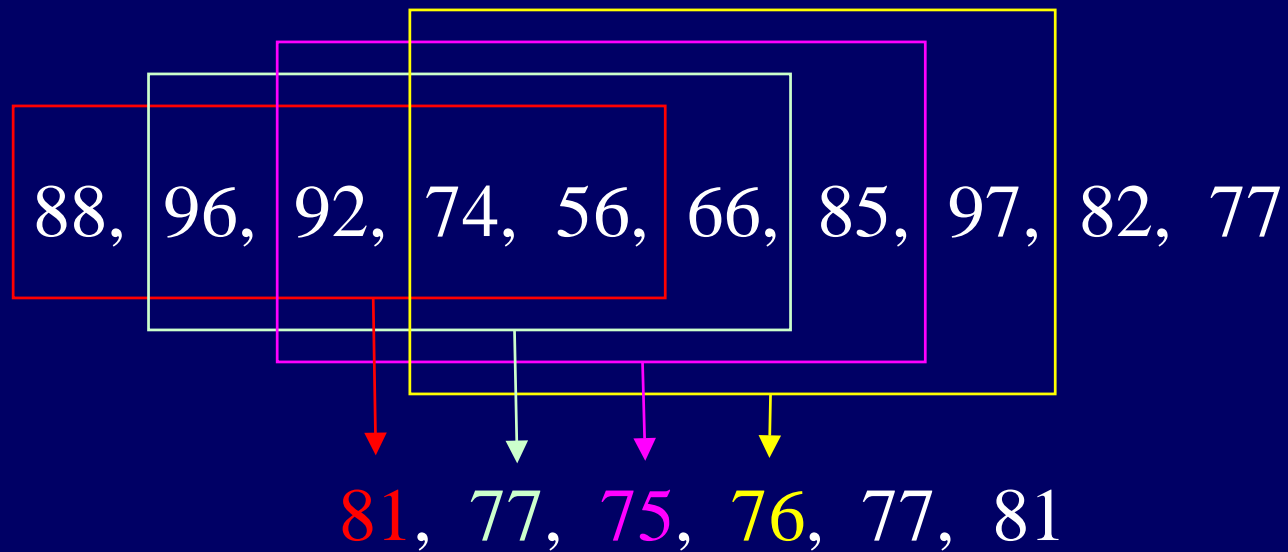
# How to read a dot matrix plot?



# Window

---

Assume window Size = 5

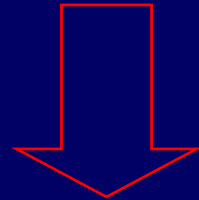


# Stringency

---

88, 96, 92, 74, 56, 66, 85, 97, 82, 77

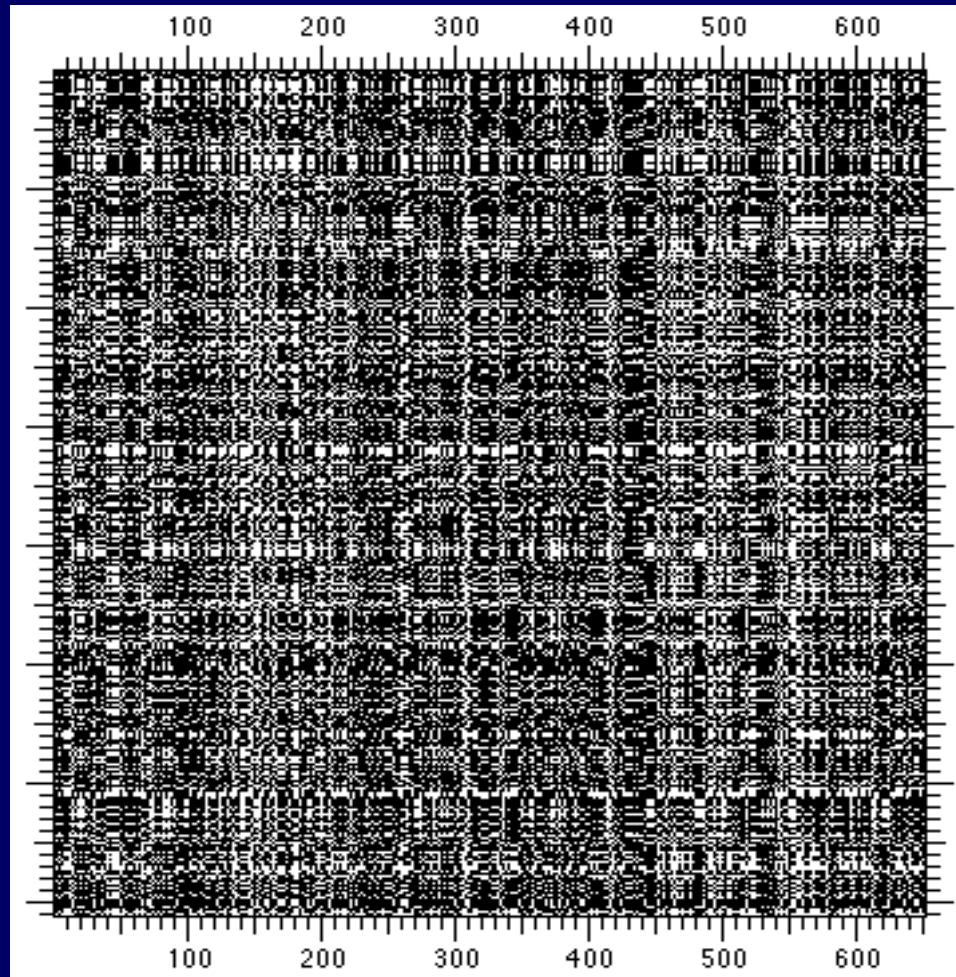
Stringency = 79



01, 01, 01, 00, 00, 00, 01, 01, 01, 00

# Dot matrix analysis with DNA

---



## Settings:

Vertical scale: lambda cI

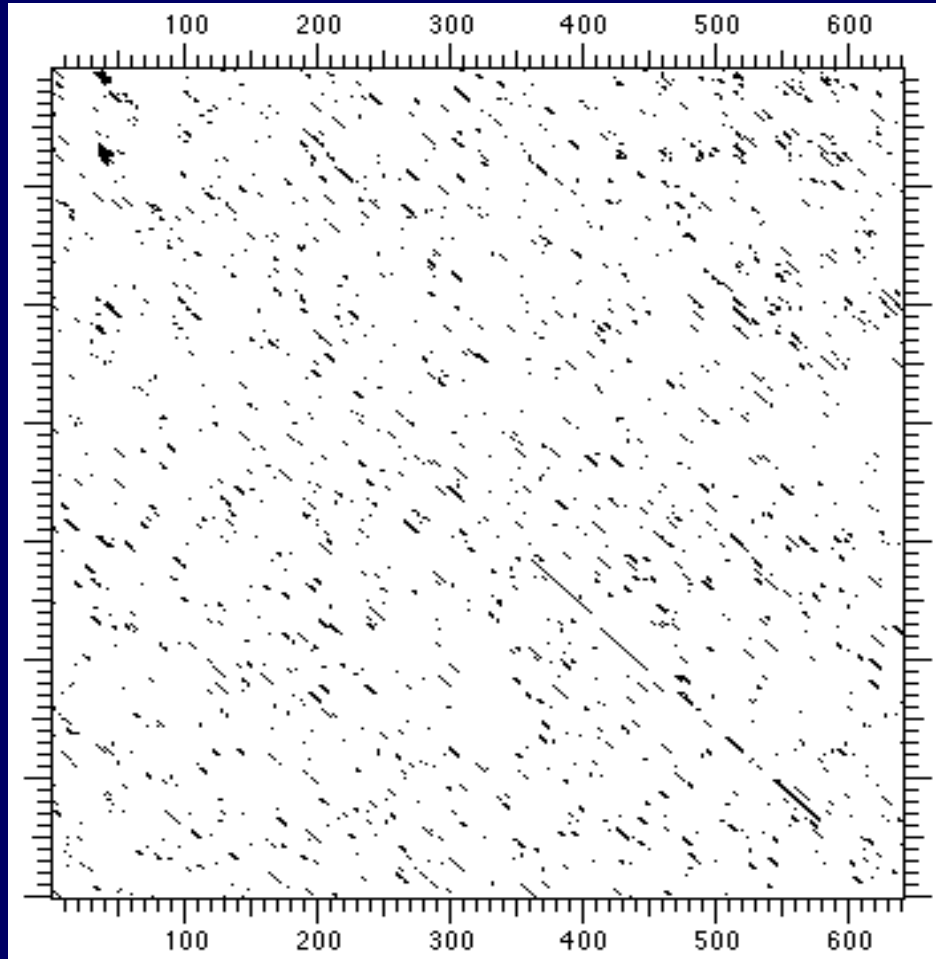
Horizontal scale: phage P22 c2

Window size: 1

Stringency: 1

# Dot matrix analysis with DNA

---



## Settings:

Vertical scale: lambda cI

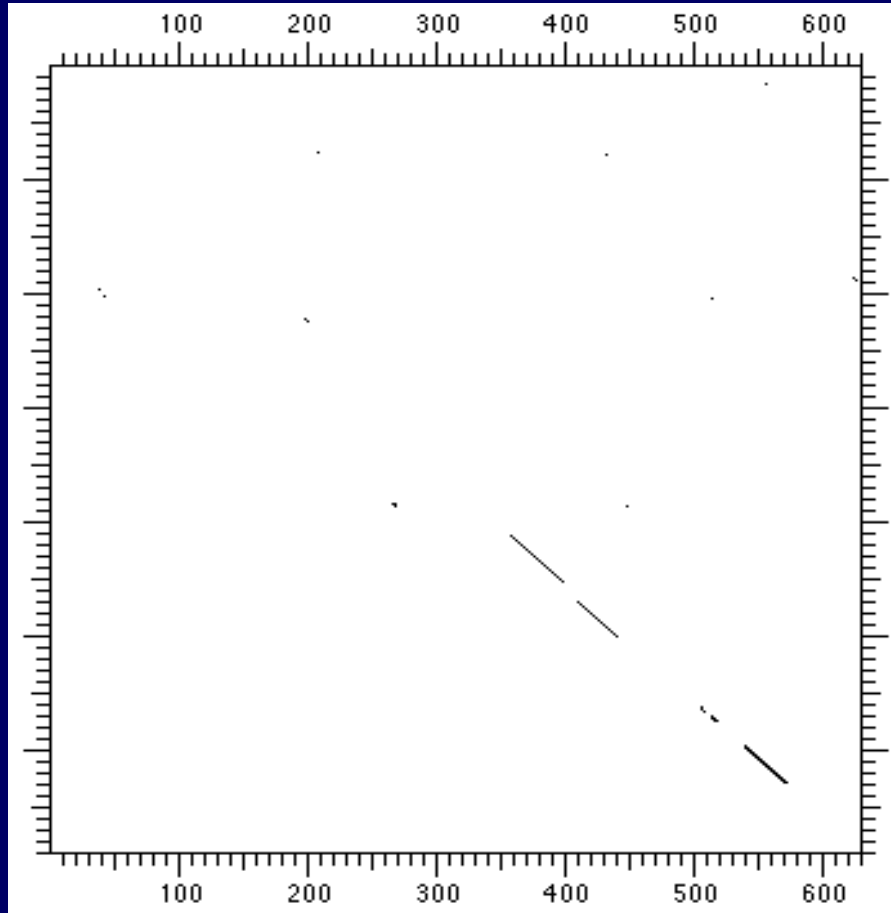
Horizontal scale: phage P22 c2

Window size: **11**

Stringency: **7**

# Dot matrix analysis with DNA

---



## Settings:

Vertical scale: lambda cI

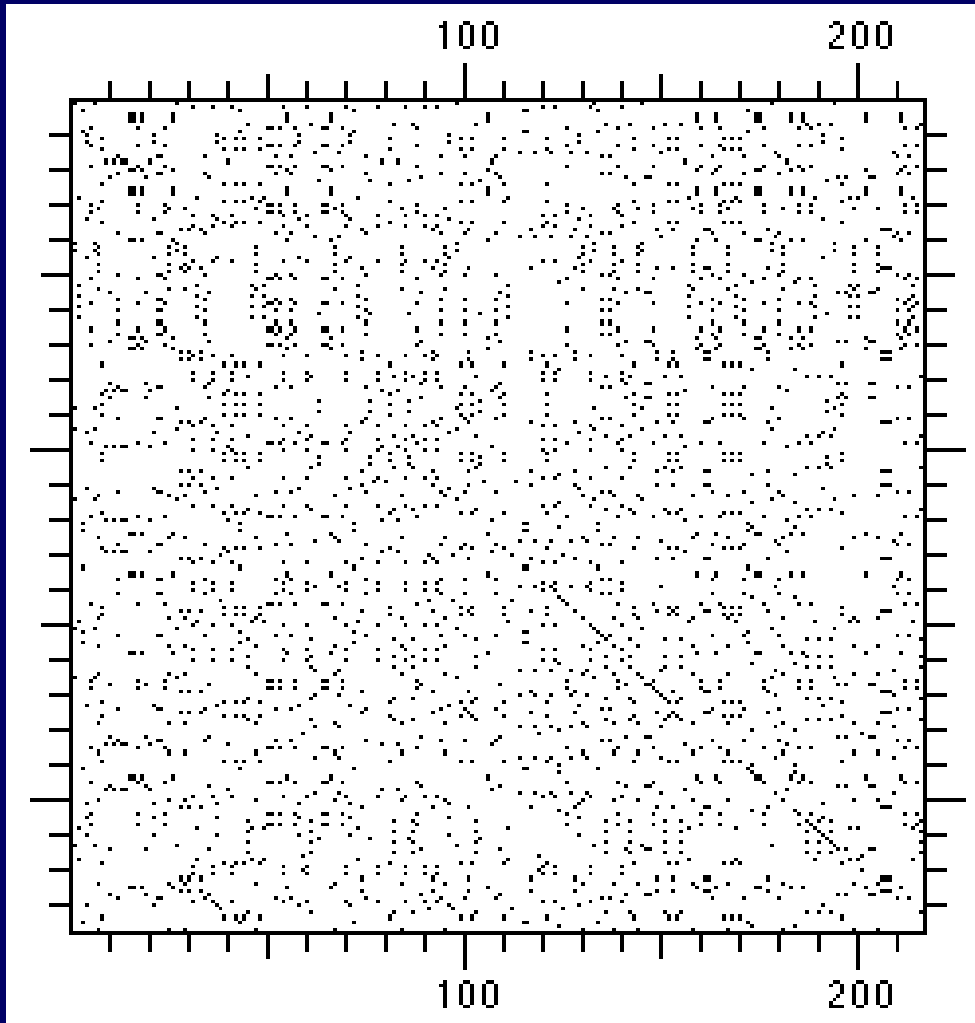
Horizontal scale: phage P22 c2

Window size: **23**

Stringency: **15**

# Dot matrix analysis with proteins

---



## Settings:

Vertical scale lambda cI

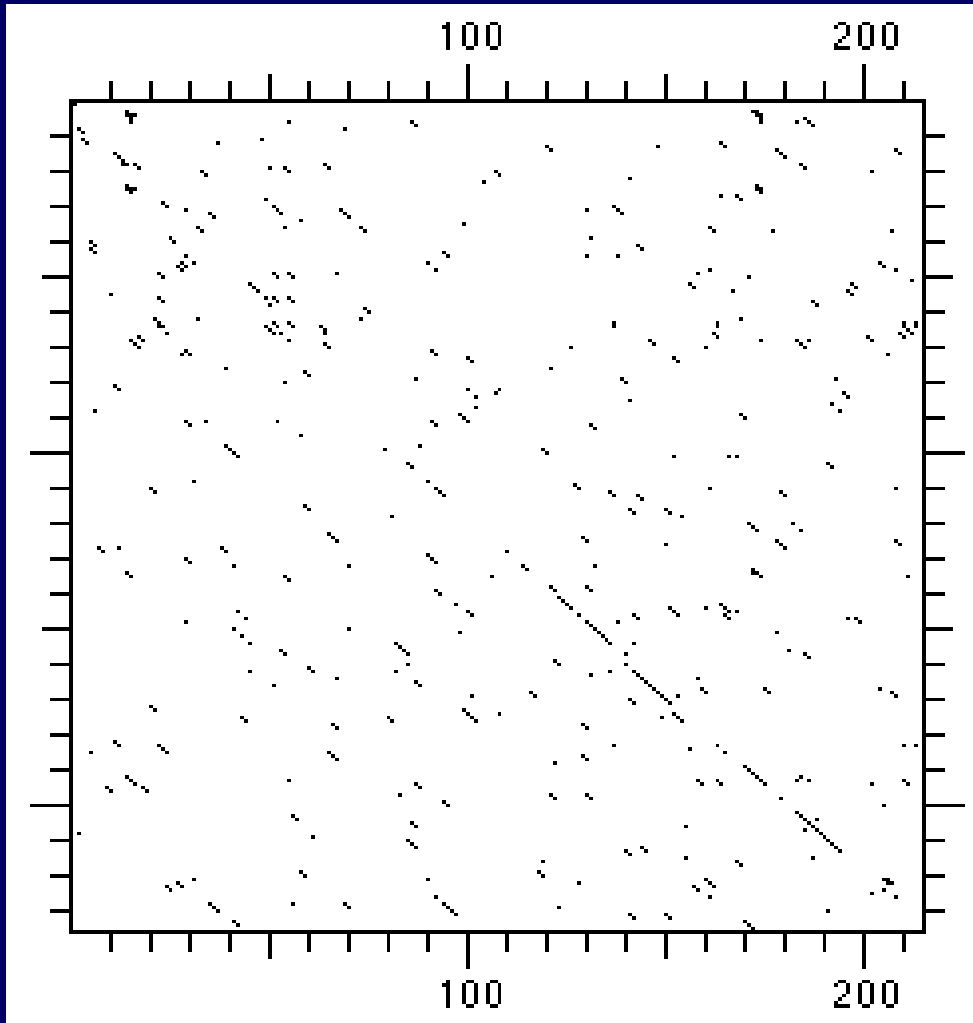
Horizontal scale: phage P22 c2.

Window size: 1

Stringency: 1

# Dot matrix analysis with proteins

---



## Settings:

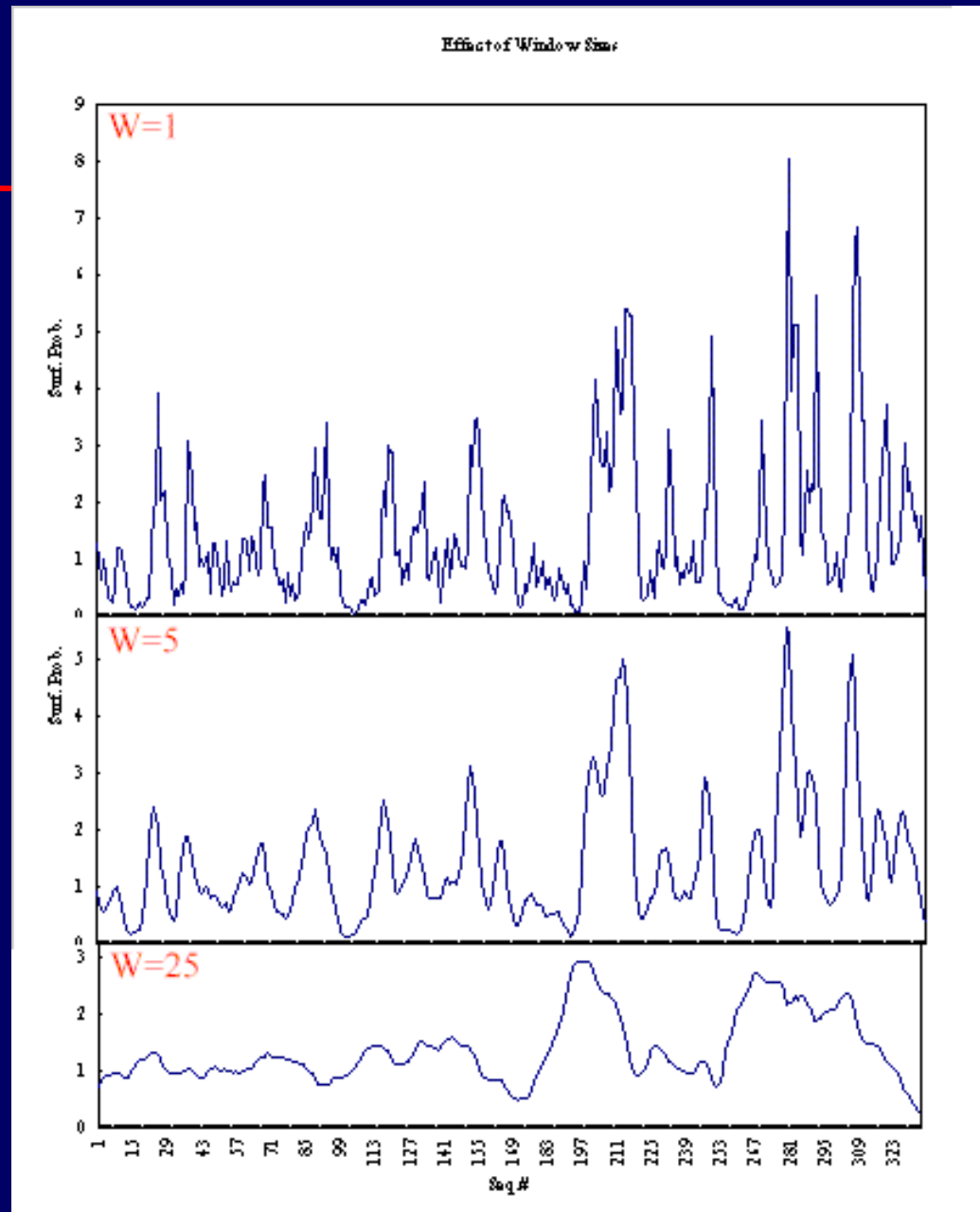
Vertical scale: lambda cI

Horizontal scale: phage P22 c2

Window size: 3

Stringency: 2

# Effect of window size



# Rules of thumb

---

- **DNA sequence alignments**

- use **large** windows (7 - 11)
- use **high** stringencies

- **Protein sequence alignments**

- use **small** windows (1 - 3)
- use **lower** stringencies

## 2. Dynamic Programming

---

- Decompose a large problem into sub-problems
- Each sub-problem is identical to the original problem except the size is smaller
- Use the same strategy to solve sub-problems and store answers in a table
- Combine solutions of the sub-problems by table “look-up”
- used when many solutions are possible and an **optimal solution** needs to be found

# Dynamic Programming (DP) Algorithms

---

## Advantages:

- guaranteed to provide the **optimal** (i.e. highest scoring) alignment (mathematically proven)
- user defined choice of **substitution matrix**
- user defined **gap penalties**
- may provide one or more sequence alignments

## Disadvantages:

- relatively slow, computational steps increase as the square or cube of the sequence lengths

# Dynamic Programming

---

- First used by Needleman and Wunsch (1970) for global alignment and for local alignment by Smith and Waterman (1981)
- Alignment is generated by starting at the ends of the sequences and by following a scoring scheme for matches, mismatches and gaps
- Breaks down a large problem into a series of small problems
- The process of finding the best path through a simple dot plot comparison of the two sequences

# Implementations of DP methods

---

## **Global alignment** (Needleman & Wunsch)



- Compare two sequences in their entirety
- Insert gaps as necessary to make the sequences the same lengths

## **Local alignment** (Smith-Waterman)



- Compare a portion of one sequence to a portion of another
- Look for the “best” possible alignment of sub-regions

# Smith-Waterman algorithm

---

- **Local** alignment method
- Does not place any restrictions on the evolutionary model
- Better at finding alignments in diverged sequences
- Most rigorous method
- Very sensitive
- Computationally expensive

# How DP works (3 steps)

---

1. Generate a sequence vs. sequence matrix; fill in the **best scores** from  $[0,0]$  to  $[n,m]$ . Keep track of pointers to allow trace-back.
2. Identify highest score in matrix
3. Trace back to start to get alignment position by position

# Step 1. Create and fill matrix

## GLOBAL

Seq. T		<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>	
		M	A	T	C	H	E	S	
Seq. S		0	-2	-4	-6	-8	-10	-12	-14
<i>i</i>	T	-2							
<i>i</i> +1	H	-4							
...	A	-6							
...	T	-8							
...	C	-10							
...	H	-12							
...	E	-14							
<i>m</i>	R	-16							

Penalize 1<sup>st</sup> column and row  
Position \* gap penalty

## LOCAL

Seq. T		<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>
		M	A	T	C	H	E	S
Seq. S		0	0	0	0	0	0	0
<i>i</i>	T	0						
<i>i</i> +1	H	0						
...	A	0						
...	T	0						
...	C	0						
...	H	0						
...	E	0						
<i>m</i>	R	0						

No penalty for 1<sup>st</sup> col. or row  
No negative numbers allowed  
(minimum score is zero)

# Step 1. Create and fill matrix

$$\text{BestScore}[ij] = \text{BestScore}[<i,<j] + \text{Match}[i,j] + \text{GapPenalty}$$

Seq. T		<i>j</i>	<i>j+1</i>	...	...	...	...	<i>n</i>	
		M	A	T	C	H	E	S	
Seq. S		0	-2	-4	-6	-8	-10	-12	-14
<i>i</i>	T	-2	-1	-3	-3	-5	-7	-9	-11
<i>i+1</i>	H	-4							
...	A	-6							
...	T	-8							
...	C	-10							
...	H	-12							
...	E	-14							
<i>m</i>	R	-16							

**Scoring contributions:**  
 Vertical: -2 (gap in T)  
 Horizontal: -2 (gap in S)  
 Diagonal: +1 if match  
 -1 if mismatch

There are only three ways of pairing at each step

1. One residue from each sequence, either a match or mismatch
2. One residue from sequence T and a gap in sequence S
3. One residue from sequence S and a gap in sequence T

**NOTE:** Gaps don't align with gaps

## Step 2. Find highest score

---

**Global alignment:** (Needleman-Wunsch)

Find highest score in final row and final column

**Local alignment:** (Smith-Waterman)

Highest score anywhere in the matrix

(Trackback begins at highest score in matrix)

# Step 2. Find highest score

## GLOBAL

## LOCAL

Seq. T		<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>	
		M	A	T	C	H	E	S	
Seq. S		0	-2	-4	-6	-8	-10	-12	-14
<i>i</i>	T	-2	-1	-3	-3	-5	-7	-9	-11
<i>i</i> +1	H	-4	-3	-2	-4	-4	-4	-6	-8
...	A	-6	-5	-2	-3	-5	-5	-5	-7
...	T	-8	-7	-4	-1	-3	-5	-6	-6
...	C	-10	-9	-6	-3	0	-2	-4	-6
...	H	-12	-11	-8	-5	-2	1	-1	-3
...	E	-14	-13	-10	-7	-4	-1	2	0
<i>m</i>	R	-16	-15	-12	-9	-6	-3	0	1

Seq. T		<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>
		M	A	T	C	H	E	S
Seq. S		0	0	0	0	0	0	0
<i>i</i>	T	0	0	0	5	0	0	2
<i>i</i> +1	H	0	0	0	0	2	10	2
...	A	0	0	5	0	0	2	9
...	T	0	0	0	10	2	0	9
...	C	0	0	0	2	23	15	7
...	H	0	0	0	0	15	33	25
...	E	0	0	0	0	7	25	39
<i>m</i>	R	0	0	0	0	0	17	31

Highest score in last row  
and last column

Highest score anywhere  
(0 = end of aligned subsequence)

## Step 3. Trace back and align

---

- start at **highest score** and create alignment in **reverse order**
- print sequence  $S[i]$  and sequence  $T[j]$  as aligned
- trace **pointer** back to previous highest score
  - if sequence  $S[i-1]$  and sequence  $T[j-1]$  then print
  - if sequence  $S[i-1]$  and sequence  $T[j-N]$  then report matches to gaps for  $S[j-1] \dots T[j-(N-1)]$
  - if sequence  $S[i-N]$  and sequence  $T[j-1]$  then print matches to gaps for  $S[i-1] \dots T[i-(N-1)]$

# Global alignment

		Seq. T	<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>
			M	A	T	C	H	E	S
Seq. S		0	-2	-4	-6	-8	-10	-12	-14
<i>i</i>	T	-2	-1	-3	-3	-5	-7	-9	-11
<i>i</i> +1	H	-4	-3	-2	-4	-4	-4	-6	-8
...	A	-6	-5	-2	-3	-5	-5	-5	-7
...	T	-8	-7	-4	-1	-3	-5	-6	-6
...	C	-10	-9	-6	-3	0	-2	-4	-6
...	H	-12	-11	-8	-5	-2	1	-1	-3
...	E	-14	-13	-10	-7	-4	-1	2	0
<i>m</i>	R	-16	-15	-12	-9	-6	-3	0	1

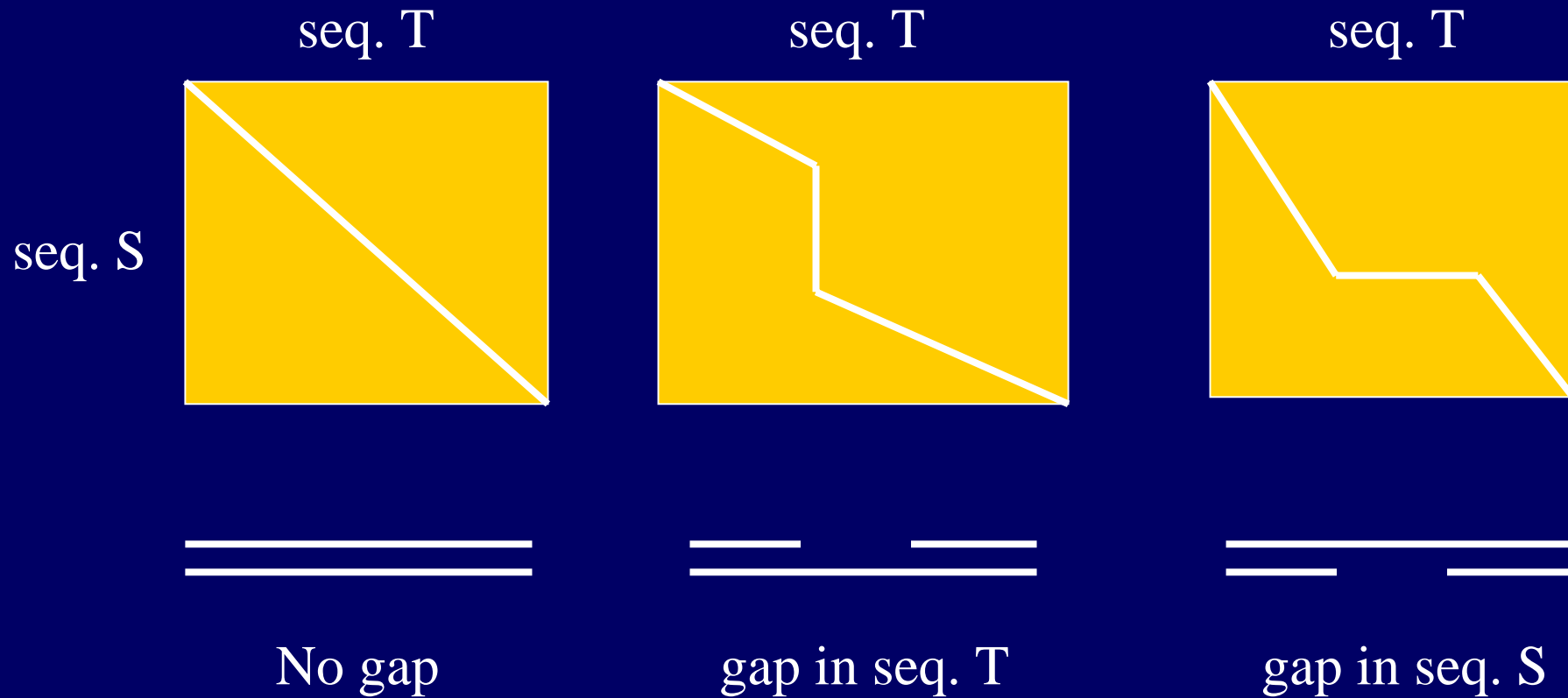
  

M	-	A	T	C	H	E	S
T	H	A	T	C	H	E	R

**Scoring contributions:**  
 Vertical: -2 (gap in T)  
 Horizontal: -2 (gap in S)  
 Diagonal: +1 if match  
 -1 if mismatch

# Alignment paths & gap placement

---



# Local alignment

		Seq. T	<i>j</i>	<i>j</i> +1	...	...	...	...	<i>n</i>
			M	A	T	C	H	E	S
Seq. S		0	0	0	0	0	0	0	0
<i>i</i>	T	0	0	0	5	0	0	0	2
<i>i</i> +1	H	0	0	0	0	2	10	2	0
...	A	0	0	5	0	0	2	9	3
...	T	0	0	0	10	2	0	9	3
...	C	0	0	0	2	23	15	7	3
...	H	0	0	0	0	15	33	25	17
...	E	0	0	0	0	7	25	39	31
<i>m</i>	R	0	0	0	0	0	17	31	38

A T C H E  
 A T C H E

**Scoring contributions:**  
 Vertical: -2 (gap in T)  
 Horizontal: -2 (gap in S)  
 Diagonal: +1 if match  
 -1 if mismatch

Stop alignment when BestScore[*ij*] is zero

# How to calculate the scores?

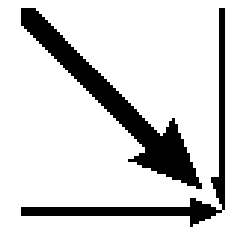
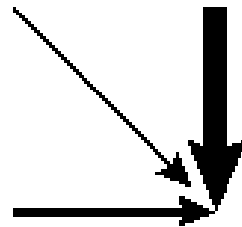
---

	T	C
T	1	0
C	0	

	T	C
T	1	0
C	0	-1

	T	C
T	1	0
C	0	-1

	T	C
T	1	0
C	0	2



	C	G
T	0	-1
C	2	

	C	G
T	0	-1
C	2	-2

	C	G
T	0	-1
C	2	1

	C	G
T	0	-1
C	2	-1

# Back Tracing

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

# A scoring system is needed

---

Calculate **the probabilities** that:

1. a particular aa pair is found in the alignment
2. the same aa is aligned by chance
3. the insertion of a gap of one or more residues in one of the sequence would improve the alignment

1 and 2 are retrieved from **a substitution matrix**

# Scoring schemes

---

- Given a scoring scheme,
  - an optimal alignment between two sequences is one with the *best* score (there might be more than one optimal alignment).
  - the *score of the sequence pair* is such a best score.
- Using the scores of sequence pairs one can:
  - investigate the hypothesis that two sequences diverged from a common ancestor
  - use the alignment of a pair of sequences that are judged to be related in order to discover common patterns.
  - by comparing scores among different species, get information to help reconstruct the phylogenetic tree that relates them all.

# Nucleic Acid Scoring Matrices

---

- Incorporates information from mutational analysis which reveals transitions are more probable than transversions
- Matrices can be used to produce global or local alignments of nucleic acid sequences

# Scoring Matrices

---

- Scoring Matrices are designed to detect signal above background, to detect similarities beyond what would be observed by chance alone
- The simplest scoring mechanism is match = 1, mismatch = -1, but these values don't work well for biological data.
- Because amino acids affect structure and reactivity, not all of the 400 aa pairs can be treated via a unitary match/mis-match matrix

# Significance of scoring matrices

---

- Sequence is not necessarily critical to protein function (3-D structure)
  - Humans can metabolize pig insulin
- Scoring matrices reflect the likelihood that one amino acid may be exchanged for another over some evolutionary distance and still preserve function
  - Empirical
- Some amino acids are critical
  - Cysteine almost never substitutes
- Expressed as logarithm of the ratio of the probabilities of two residues being aligned due to homology vs. due to random chance

# Similarity or substitution matrices

---

- attempts to quantify whether a mutation preserves or disrupts the function of a protein
- reflect different degrees of evolutionary divergence
- provide a quantifiable measure for amino acid residue substitutions

## Examples:

- a) Point Accepted Mutations (PAM)**
- b) Block sum (BLOSUM)**

# Amino Acid Substitution Matrices

---

- Amino acid substitutions commonly occur in related proteins from different species
- Knowing the types of changes that are most and least common in a large number of proteins can assist with predicting alignments for any set of protein sequences
- If related protein sequences are quite similar, one can readily determine single-step amino acid changes
- In matrix, each position is filled with a score that reflects how often one amino acid would have been paired with another in an alignment of related protein sequences

# PAM and BLOSUM Matrices

---

- 1) **PAM** (percent accepted mutation) – lists the likelihood of change from one amino acid to another in homologous sequences during evolution
- 2) **BLOSUM** – matrix values are based on a large set of ~2000 conserved amino acid patterns called **blocks**. Blocks come from a database of protein sequences representing more than 500 families of related proteins.
- 3) PAM matrices were the first matrices, BLOSUM matrices came later. For most applications, BLOSUM 62 is the default scoring matrix.

# Comparison of PAM and BLOSUM Matrices

---

- PAM matrices are based on the prediction of the first changes that occur as proteins diverge from a common ancestor during evolution of a protein family
- PAM model is designed to track the evolutionary origins of proteins
- BLOSUM matrices are derived from considering all amino acid changes observed in an aligned region of related family of proteins
- BLOSUM model is designed to find their conserved domains
- In BLOSUM, not all mutations are counted equally (similar sequences are clustered and together)
- PAM matrices based on mutations observed throughout a global alignment, BLOSUM is based on conserved regions (blocks) which contain no gaps

# PAM vs. BLOSUM matrix

---

## PAM matrix (Dayhoff)

1. Based on mutations in conserved and variable regions in **global** alignments
2. Limited # of observations
3. Derived from an explicit evolutionary model

## BLOSUM matrix (Henikoff)

1. based exclusively on mutations in **local**, highly conserved regions w/o gaps
2. Large # of observations
3. Derived with a sum-of pairs evolutionary model

# Multiple Sequence Alignments

## A. Block alignment

```

VRALFDF KGDILRI WUNA GMIPVPYV
FVALYDF KGEKLRV WCEA GWVPSNYI
VQALFDF RGDFIHV WWKG GMFPRNYV
VVALYDY KGDEYFI WWRA GYIPSNYV
FRAMYDY DGDALIN WMYG GMLPANYV
VKALFDY KSAIIQN WWRG LWFPSNYV
YRALYDY LGDILTV WLNG GDFPGTYV
    
```

## B. Segment alignment

```

EYVRALFDFNGNDEEDLPFKKGDILRIRDKPEEQ.....WUNAEDSEGKR.GMIPVPYVEK
NLFVALYDFVASGDNTLSITKGEKLRVLGYNHNGE.....WCEAQTKNQ..GWVPSNYITP
TYVQALFDFDPQEDGELGFRRGDFIHVMDNSDPN.....WWKGACHGQT..GMFPRNYVTP
KKVVALYDYMPMNANDLQLRKGDEYFILEESNLP.....WWRARDKNGQE.GYIPSNYVTE
KIFRAMYDYMAADADEVSFKDGDALINVQAIDEG.....WMYGTVQRTGRTGMLPANYVEA
CAVKALFDYKAQREDELTFIKSAIIQNVEKQEGG.....WWRGDYGGKKQ.LWFPSNYVEE
YQYRALYDYKKEREEDIDLHLGDILTVNKGSLVALGFSGDQEARPEEIGWLNGYNETTGERGDFPGTYVEY
    
```

## C. Local alignment

```

.....aeyVRALFDFngndeedlpfkKGDILRIrdkpee.....WUNAedsegkr.GMIPVPYVek.....
.....nlFVALYDFvasgdntlsitKGEKLRVlgynhng.....WCEAqtknqg..GWVPSNYItpvns.....
lvdyhrstsvsrnqqiflrldieqvpqqptyVQALFDFdpqedgelgfrRGDFIHVmdnsdpn.....WWKGachgqt..GMFPRNYVtpvnrnv....
.....gsmstselkkVVALYDYmpmnandlqlrKGDEYFIleesnl.....WWRARdkngqe.GYIPSNYVteaeds.....
.....tagkiFRAMYDYmaadadevsfkDGDALINvqaideg.....WMYGTvqrtgrtGMLPANYVeai.....
.....gsptfkcaVKALFDYkaqredeltfiKSAIIQNvekqegg.....WWRGDyggkkq..LWFPSNYVeemvnpegihrd
.....gyqYRALYDYkkereedidlhlLGDILTVnkgsivalgfsdgqearpeeigWLNGynettgerGDFPGTYVeyigrkkisp..
    
```

## D. Global alignment

```

.....AEYVRALFDFNGNDEEDLPFKKGDILRIRDK.....EEQWUNAEDS.EGKRGMI PVPYVEK.....
.....NLFVALYDFVASGDNTLSITKGEKLRVLGYN.....HNGEWCEAQT..NGQGWVPSNYITPVNS.....
LVDYHRSTSVSRNQQIFLRDIEQVPQQPTYVQALFDFDPQEDGELGFRRGDFIHVMDNS.....DPNWWKGACH..GQTGMFPRNYVTPVNRNV....
.....GSMSTSELKKVVALYDYMPMNANDLQLRKGDEYFILEES.....NLPWWRARDK.NGQEGYIPSNYVTEAEDS.....
.....TAGKIFRAMYDYMAADADEVSFKDGDALINVQAI.....DEGWMYGTVQRTGRTGMLPANYVEAI.....
.....GSPTFKCAVKALFDYKAQREDELTFIKSAIIQNVEKQ.....EGGWWRGDY.GKKQLWFPSNYVEEMVNPEGIHRD
.....GYQYRALYDYKKEREEDIDLHLGDILTVNKGSLVALGFSGDQEARPEEIGWLNGYNETTGERGDFPGTYVEYIGRKKISP..
    
```

# What can a MSA be used for?

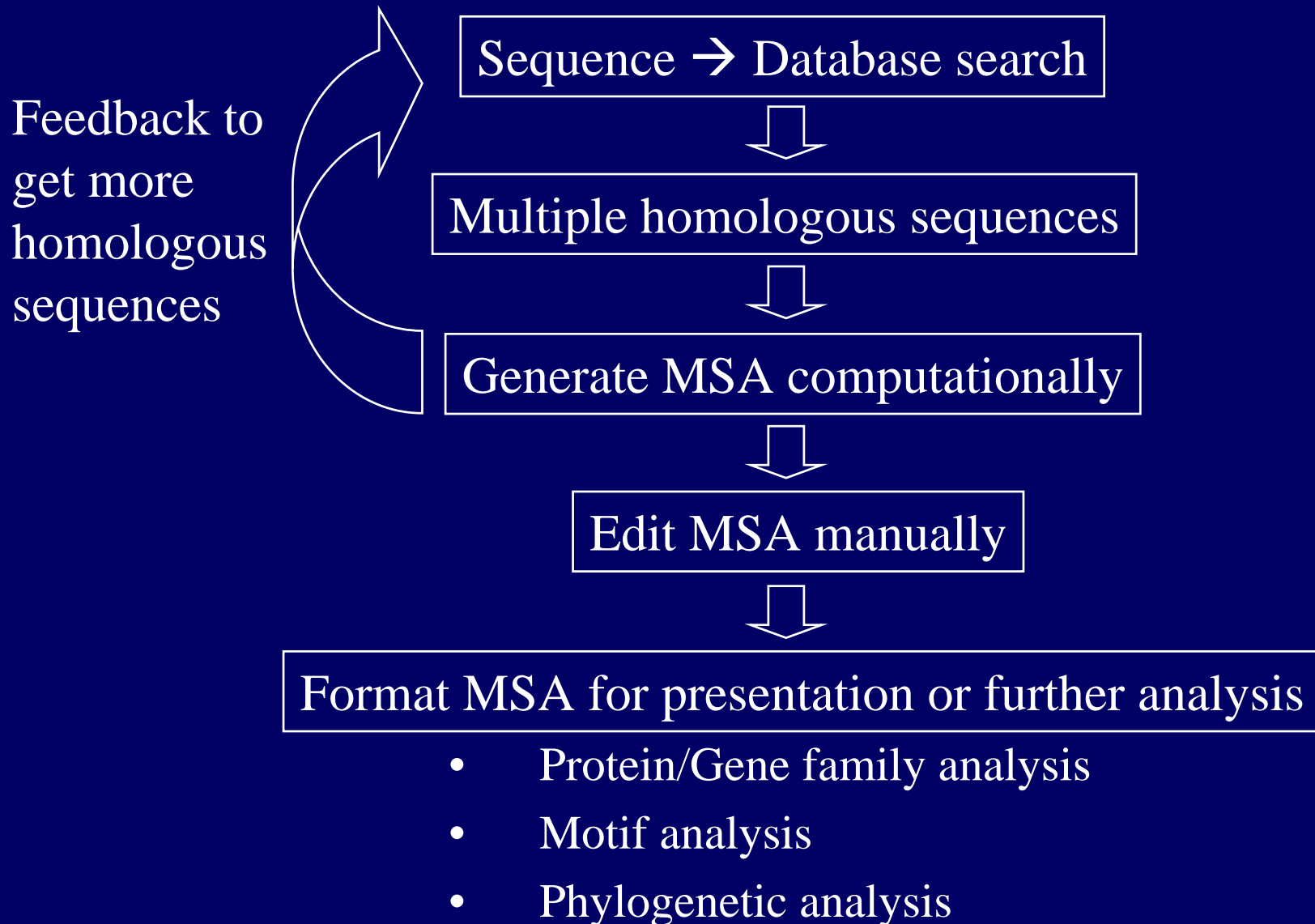
---

- Valuable for both DNA and Protein
  - Identification of highly conserved regions of homologous sequences (e.g., identify structural & functional domains or protein families, for secondary structure prediction, & design of primers for PCR, drugs and vaccines)
  - Residue conservation and acceptable amino acid substitutions (e.g., for evaluation of crucial residues for enzyme action and ligand binding)
  - Reconstruction of evolutionary relationships- phylogenetic trees
  - Structural alignment (Protein 3-D Modeling)

\* The alignment of a family of sequences provides more information than the alignment of any pair of those sequences.

# Analysis using MSA

---



# Correct Alignment? Optimal Alignment?

---

- Except for highly identical sequences, it is impossible to unambiguously create a single correct MSA.
- For a given group of sequences, there is no single "correct" alignment, only an alignment that is "optimal" according to some set of calculations.
- Determining what alignment is best for a given set of sequences is really up to the judgement of the investigator.
- Success of the alignment will depend on the similarity of the sequences. If sequence variation is great it will be very difficult to find optimal alignment.

# Algorithms for MSA

---

- Progressive alignment
  - Profile alignment
  - Iterative refinement methods
- Profile HMM training
- Multi-dimensional dynamic programming

# How do we do a MSA? (Methods)

---

- Progressive global alignment of the sequences starting with an alignment of the most alike sequences and then building an alignment by adding more sequences.
- Iterative methods that make an initial alignment of groups of sequences and then revise the alignment to achieve a better result.
- Alignments based on locally conserved patterns found in the same order in the sequences.
- Use of statistical methods and probabilistic models of the sequences.

# Progressive Pairwise Methods

---

- Most of the available multiple alignment programs use some sort of incremental or progressive method that makes pairwise alignments, then adds new sequences one at a time to these aligned groups.
- Sequences are chosen based on a “guide tree”.
- This is an approximate method!

# Pairwise vs Multiple Sequences

---

- Pairs of sequences typically aligned using exhaustive algorithms (dynamic programming)
- Multiple sequence alignment using heuristic methods

# Multiple Sequence Alignments

---

## **Global** alignment methods

- ClustalW (most popular)
- PileUp

## **Local** alignment methods

- Dialign

# Progressive Pairwise Programs

---

- PILEUP is the multiple alignment program in the GCG package.
- CLUSTAL is another popular program that uses a similar algorithm.

# PILEUP

---

- PILEUP is the MSA program that is part of the Genetics Computer Group (GCG) sequence analysis package
- Sequences are aligned pairwise using dynamic programming algorithm
- The scores are used to produce a phylogenetic tree, which is then used to guide the alignment of the most closely related sequences and groups of sequences
- Resulting alignment is a global alignment produced by the Needleman-Wunsch algorithm

# PILEUP Drawbacks

---

- No recent enhancements such as gap modifications or sequence weighting comparable to those introduced for CLUSTALW
- As with other progressive alignment programs, does not guarantee an optimal alignment
- Major problem with progressive alignment programs such as CLUSTAL and PILEUP is the dependence of the final MSA on the initial pairwise alignments
- For closely related sequences, CLUSTAL is designed to provide an adequate alignment of a large number of sequences

# The PILEUP Algorithm

---

- First, **PILEUP** calculates approximate pairwise similarity scores between all sequences to be aligned, and they are clustered into a dendrogram (tree structure).
- Then the most similar pairs of sequences are aligned.
- Averages (similar to consensus sequences) are calculated for the aligned pairs.
- New sequences and clusters of sequences are added one by one, according to the branching order in the dendrogram.

# How ClustalW works

---

- based on Progressive Pairwise Alignment (PPA)
  1. **globally** align most similar sequences first
  2. construct a tree using **neighbor-joining**  
(determines the order in which subsequent seq. are incorporated into the alignment)
  3. align the sequences sequentially, guided by the phylogenetic relationships indicated by the tree

# CLUSTAL

---

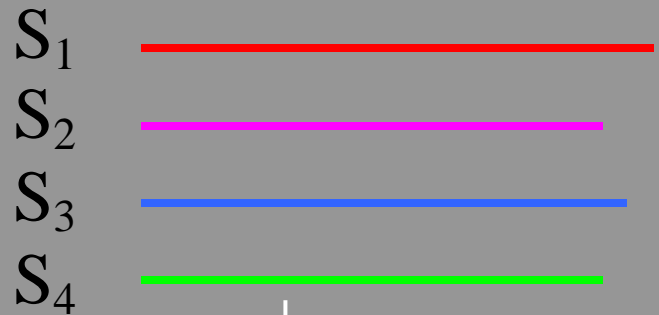
- **CLUSTAL** is a stand-alone (i.e. not integrated into GCG) multiple alignment program that is superior in some respects to **PILEUP**
  - Gap penalties can be adjusted based on specific amino acid residues, regions of hydrophobicity, proximity to other gaps, or secondary structure.
  - it can re-align just selected sequences or selected regions in an existing alignment
  - It can compute phylogenetic trees from a set of aligned sequences.

# CLUSTALW Features

---

- Gap penalties can be adjusted based on specific amino acid residues, regions of hydrophobicity, proximity to other gaps, or secondary structure.
- It can re-align just selected sequences or selected regions in an existing alignment
- It can compute phylogenetic trees from a set of aligned sequences.

# ClustalW: Progressive Multiple Alignment



- Multiple Alignment Step:
1. Aligning  $S_1$  and  $S_3$
  2. Aligning  $S_2$  and  $S_4$
  3. Aligning  $(S_1, S_3)$  with  $(S_2, S_4)$ .

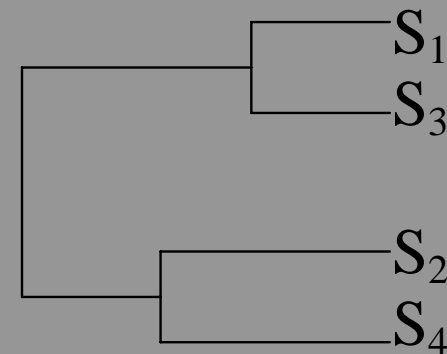
All Pairwise Alignments

Similarity Matrix

	$S_1$	$S_2$	$S_3$	$S_4$
$S_1$		4	9	4
$S_2$			4	7
$S_3$				4
$S_4$				

Cluster Analysis

Dendrogram



Distance

From Higgins(1991) and Thompson(1994).

# Clustal Format

---

CLUSTAL X (1.81) multiple sequence alignment

```
CAS1_BOVIN      MKLLI LTCLVAVALARPKHPI KHQGLPQ-----EVLNEN-
CAS1_SHEEP      MKLLI LTCLVAVALARPKHPI KHQGLSP-----EVLNEN-
CAS1_PIG        MKLLI FICLAAVALARPKPPLRHQEHLONEPDSRE-----
CAS1_HUMAN      MRLLI LTCLVAVALARPKLPLRYPERLONPSESSE-----
CAS1_RABBIT     MKLLI LTCLVATALARHKFHLGHLKLTQEQPESSEQEILKERK
CAS1_MOUSE      MKLLI LTCLVAAAFAMPRLHSRNAVSSQTQ-----QQHSSSE
CAS1_RAT        MKLLI LTCLVAAALALPRAHRRNAVSSQTQ-----
*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:
```

# Alignment of coding regions

---

- Nucleotide sequences much harder to align accurately than proteins
- Protein coding sequences can be aligned using the protein sequences
- Proteins are easier to align than DNA because they are more conserved
- Hint, if you need a DNA alignment, generate the protein alignment first and then use it to guide your DNA alignment

MMGSTLV	ATGATGGGGGAGUCCCCUCGUI
M-GSTIV	ATG---GGCGCCCCUAUUGUG

# How Dialign works ?

---

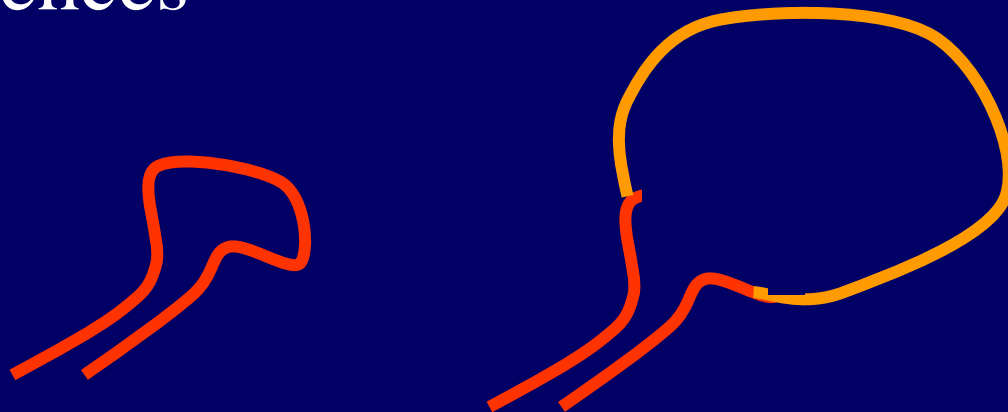
- **Local** alignment approach
- identify gap-free fragments (called diagonals) of high similarity
- built segments into multiple alignment using an iterative approach
- works with DNA and proteins

# When to use Dialign ?

---

Dialign performs well when:

- sequences have **long terminal extensions**
- sequences have **large insertions**
- useful for finding **conserved blocks** within a set of sequences



# Other commonly used programs for pairwise alignment

---

- GCG
  - gap (global)
  - bestfit (local)
- EMBOSS
  - matcher, est2genome (local)
  - stretcher (global)
- Others
  - Blast2sequences (local)
  - SIM4 (cDNA-genomic DNA)

# Profile Analysis

---

- Profiles are made by performing the global msa of a group of sequences and then removing the highly conserved regions in the alignment in a smaller msa
- A scoring matrix for the msa, called a profile is made

# Iterative Methods

---

- Attempt to correct initial alignment problems by repeatedly aligning subgroups of the sequences and then by aligning these subgroups into a global alignment of all the sequences

# Steps to perform local alignments using a global MSA program

---

- Perform a databank search, or a seq. comparison
- Cut out sequences ranges that are homologous on the basis of the above results
- Perform global MSA on the select sequence ranges

# In GCG, you use ...

---

**Preliminary alignment**

**FastA / Local Blast**

**Compare / DotPlot**

**Alignment**

**FileUp**

**Display**

**Pretty**

**PlotSimilarity**

# Local Sequence Alignment

---

1	YICSFADCGAAYNKNWKLQ*AHLC*KH	37
2	TGEK*PFPCKEEGCEKGFTSLHHLT*RHSL*TH	67
3	TGEK*NFTCDSDGCDLRFTTKANMK*KHFNRFH	98
4	NIKICVYVCHFENCGKAFKKHNQLK*VHQF*SH	129
5	TQQL*PYECPHEGCDKRFSLPSRLK*RHEK*VH	159
6	AG--*YPCCKDDSCSFVGKTWTLYLKHVAECH	188
7	QD--*LAVC--DVCNRKFRHKDYLR*DHQK*TH	214
8	EKERTVYLCPRDGCDRSYTTAFNLR*SHIQSFH	246
9	EEQR*PFVCEHAGCGKCFAMKKSLE*RHSV*VH	276
	TGEK*PYVC..DGCDKRETKK..LK*RH..* .H	Consensus

# Multiple Alignment Strategies

---

- Align pairs of sequences using an optimal method
- Choose representative sequences to align carefully
- Choose sequences of comparable lengths
- Progressive alignment programs such as Clustal for multiple alignment
- Progressive alignment programs may be combined
- Review alignment by eye and edit