Alignment Methods

- Visual
- Brute Force
- Dynamic Programming
- Word-Based (k tuple)

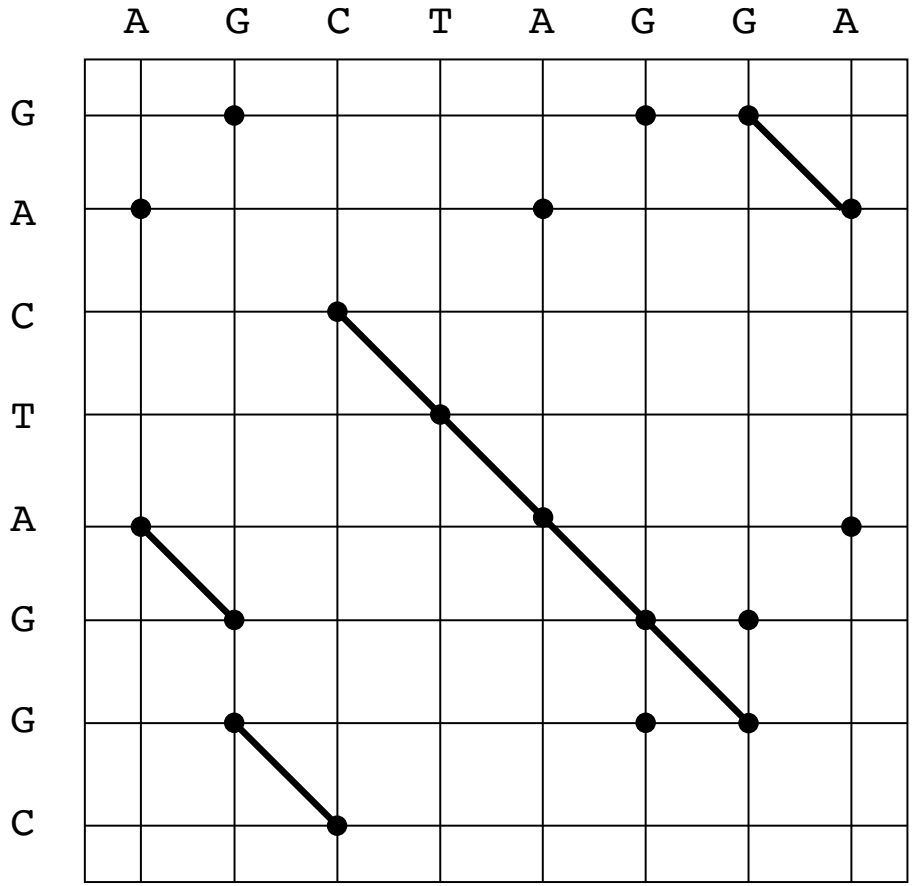
Visual Alignments (Dot Plots)

- Matrix
  - Rows: Characters in one sequence
  - Columns: Characters in second sequence

- Filling
  - Loop through each row; if character in row, col match, fill in the cell
  - Continue until all cells have been examined

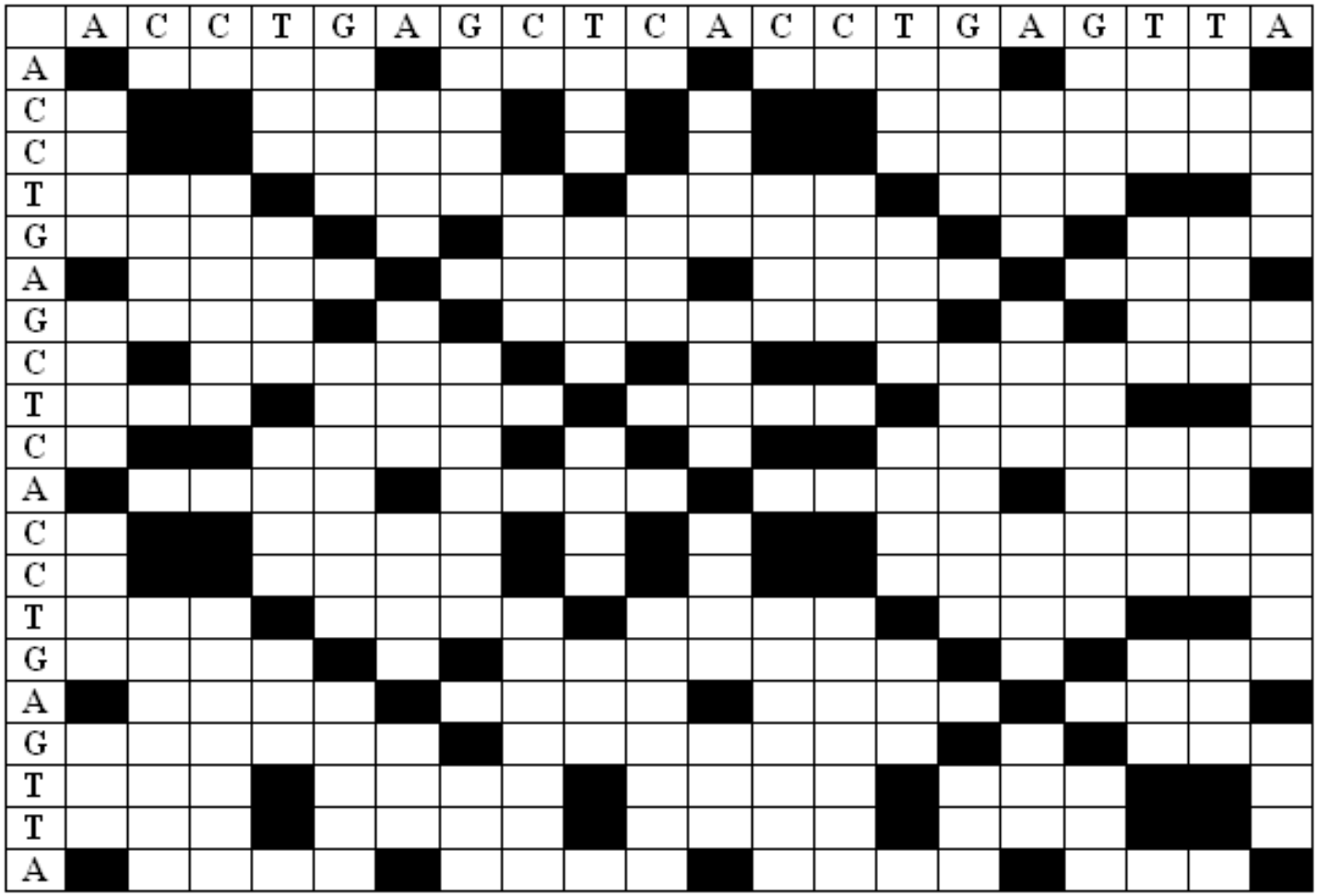
The Dot Matrix

- established in 1970 by A.J. Gibbs and G.A.McIntyre
- method for comparing two amino acid or nucleotide sequences

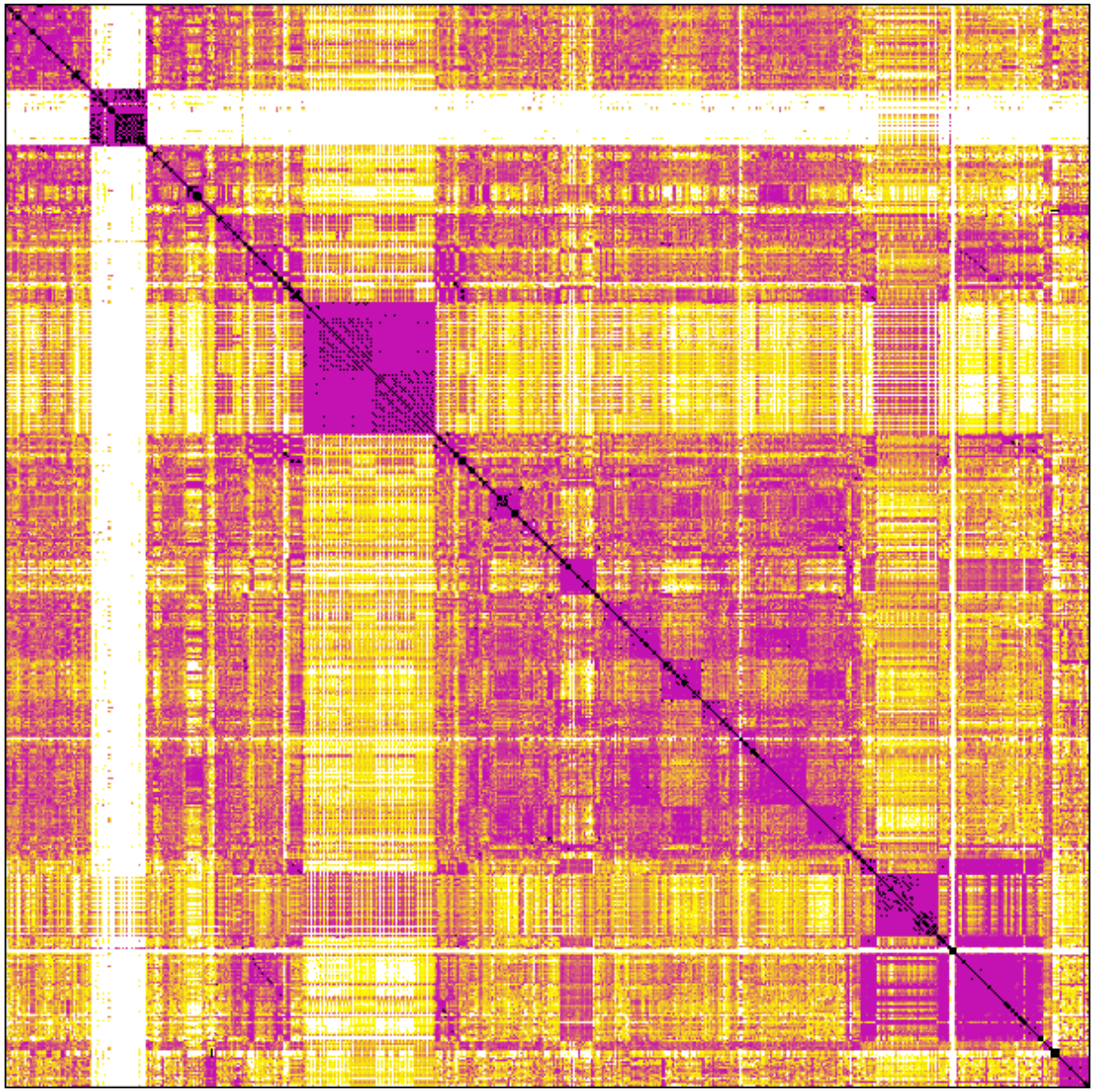


- each sequence builds one axis of the grid
- one puts a dot, at the intersection of same letters appearing in both sequences
- scan the graph for a series of dots
  - reveals similarity
  - or a string of same characters
- longer sequences can also be compared on a single page, by using smaller dots

Example Dot Plot



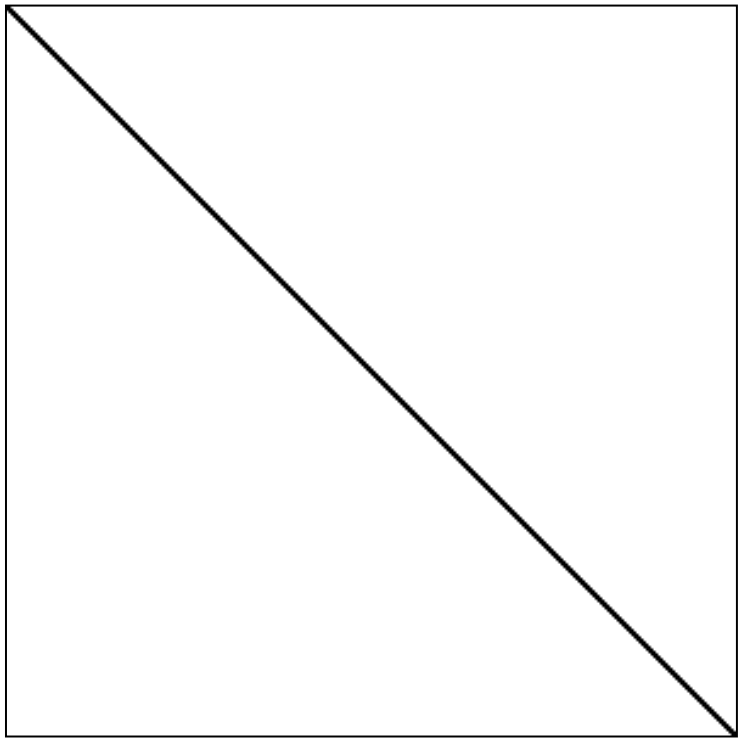
An entire software module of a telecommunications switch; about two million lines of C



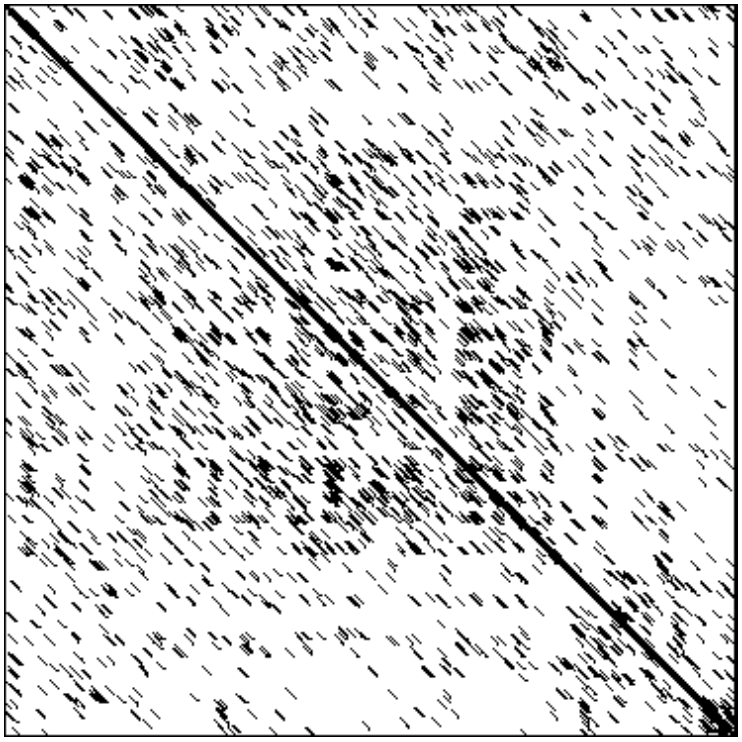
Darker areas indicate regions with a lot of matches (a high degree of similarity).
Lighter areas indicate regions with few matches (a low degree of similarity).
Dark areas along the main diagonal indicate sub-modules.
Dark areas off the main diagonal indicate a degree of similarity between sub-modules
The largest dark squares are formed by redundancies in initializations of signal-tables and finite-state machines.

The Dot Matrix

The very stringent, self-dotplot:



The non-stringent self-dotplot:



Noise in Dot Plots

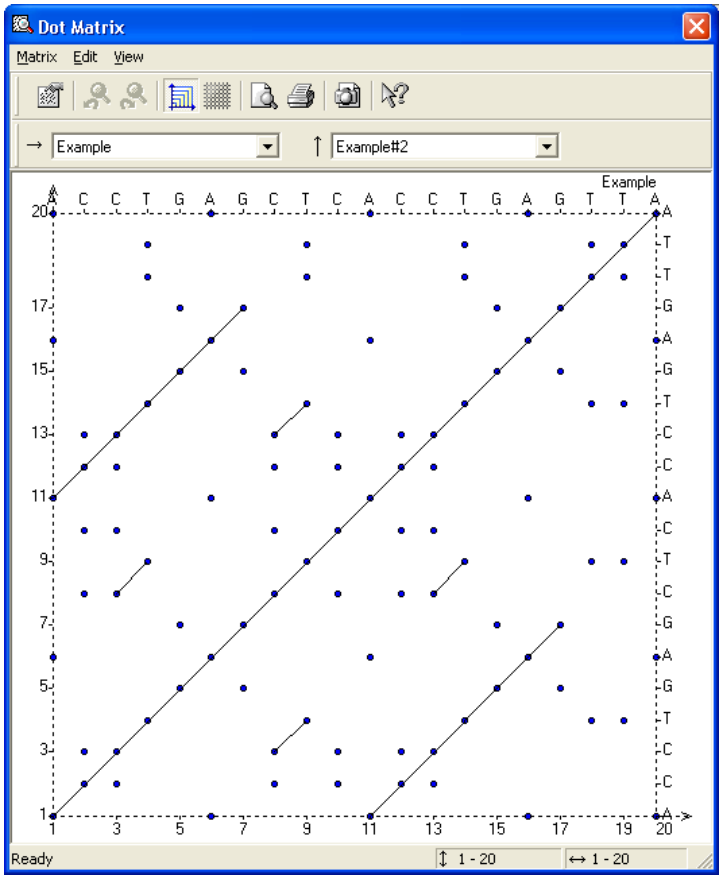
- Nucleic Acids (DNA, RNA)
  - 1 out of 4 bases matches at random

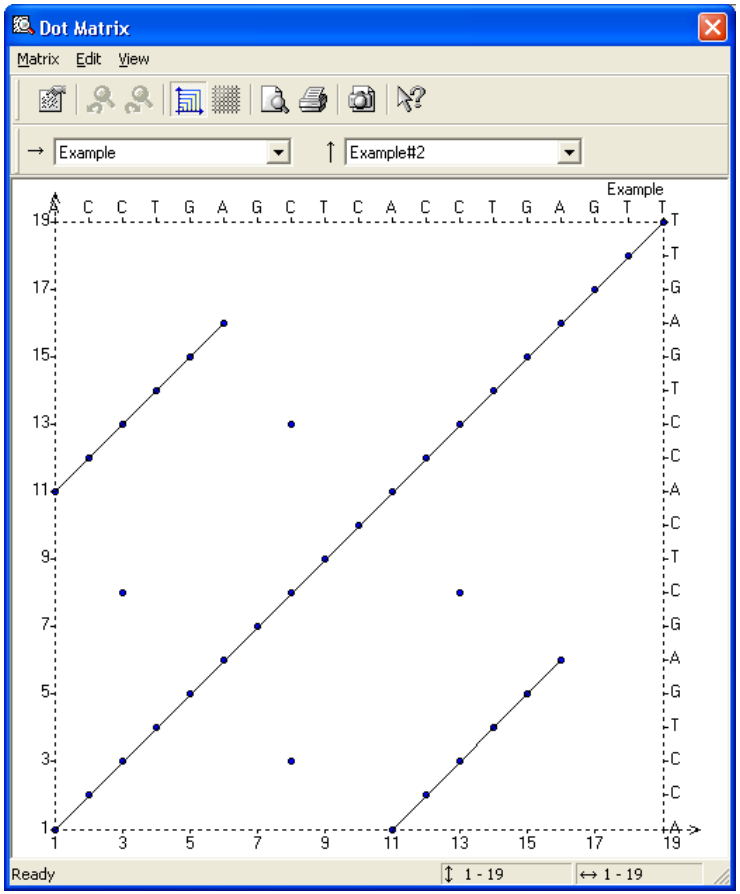
- Stringency (The condition under which a DNA sequence can bind to related or non-specific sequences. For example, high temperature and lower salt increases stringency such that non-specific binding or binding with low melting temperature will dissolve)
  - Window size is considered
  - Percentage of bases matching in the window is set as threshold

The Dot Matrix

- to filter out random matches, one uses sliding windows
- a dot is printed only if a minimal number of matches occur
- rule of thumb:
  - larger windows for DNAs (only 4 bases, more random matches)
  - typical window size is 15 and stringency of 10

Reduction of Dot Plot Noise

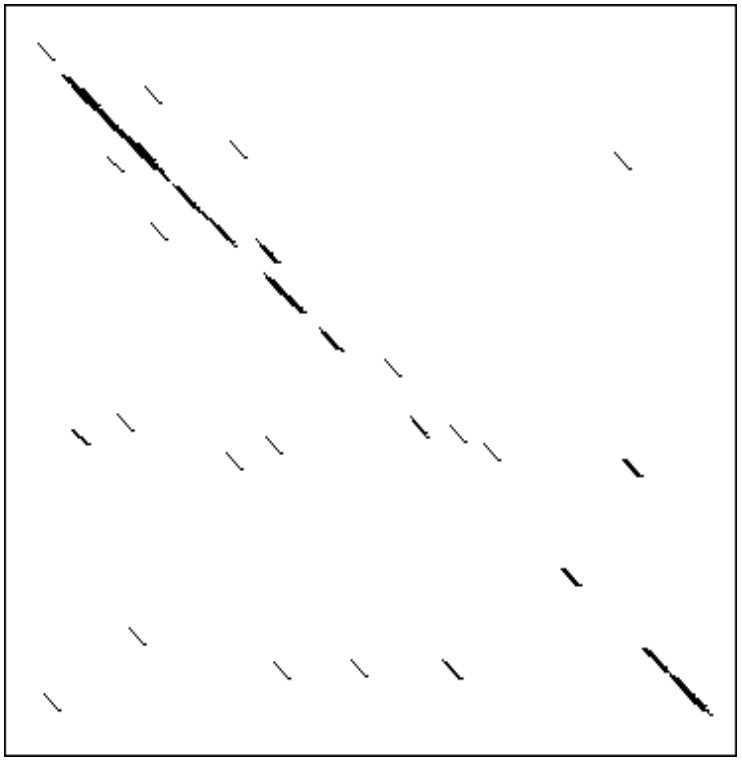




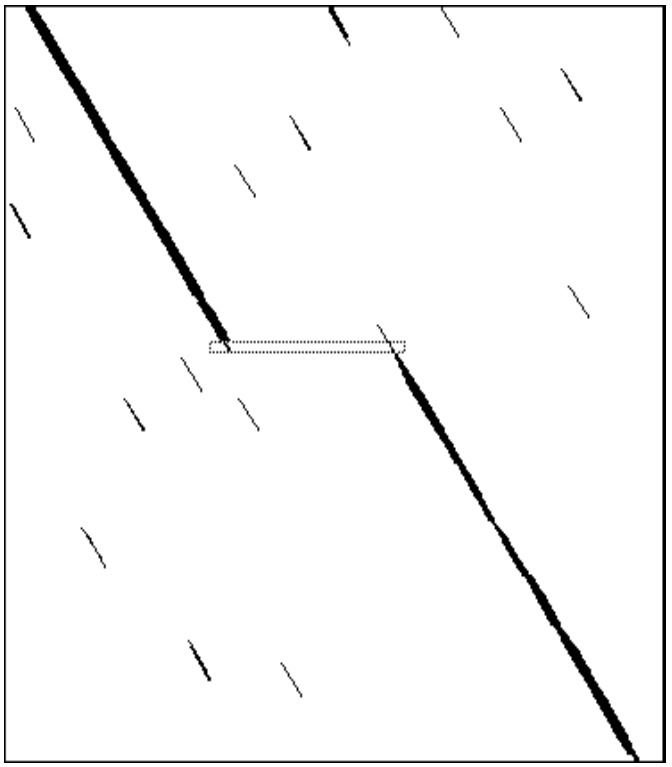
Self alignment of ACCTGAGCTCACCTGAGTTA

The Dot Matrix

Two similar, but not identical, sequences

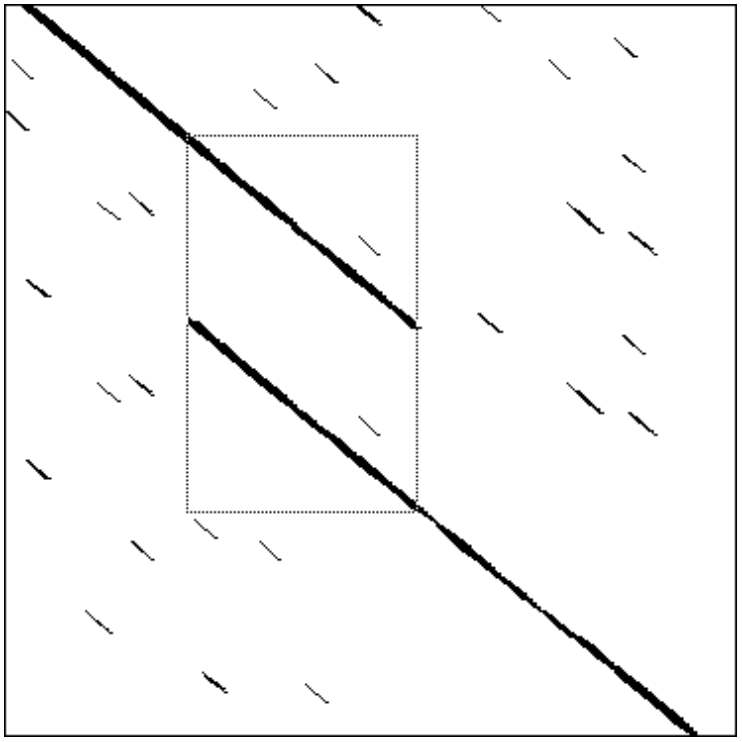


An indel (insertion or deletion):

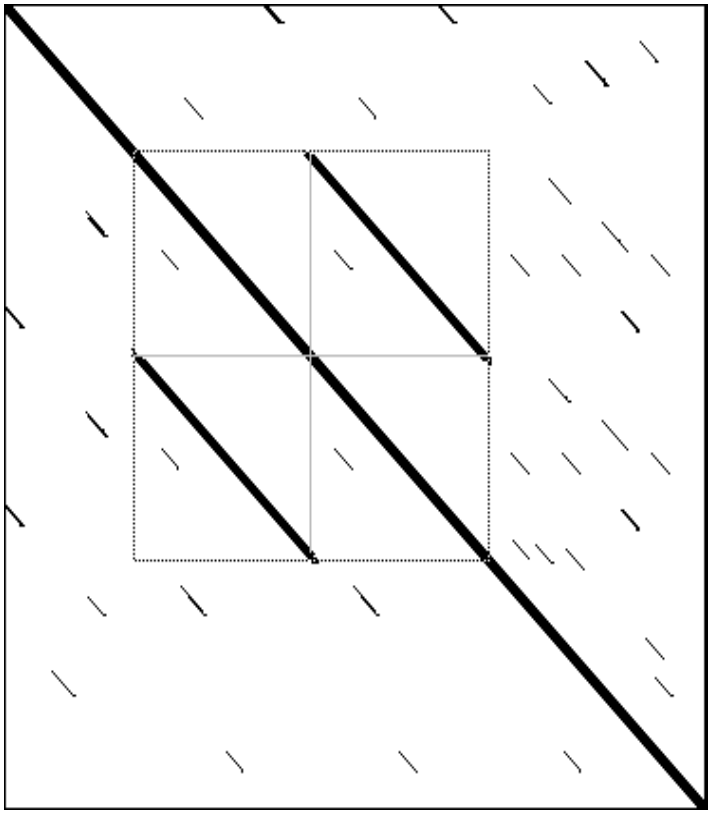


The Dot Matrix

A tandem duplication:

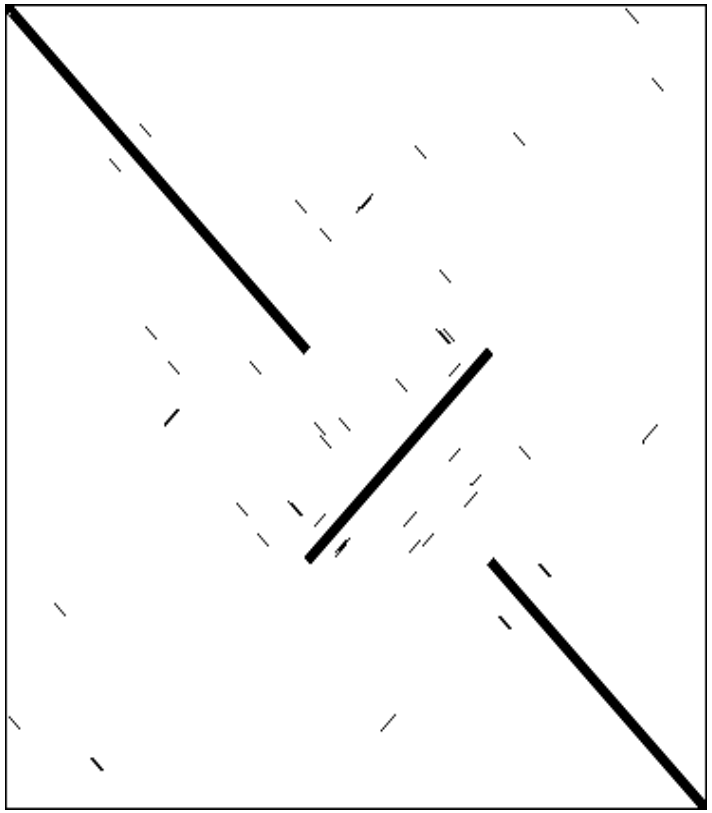


Self-dotplot of a tandem duplication:

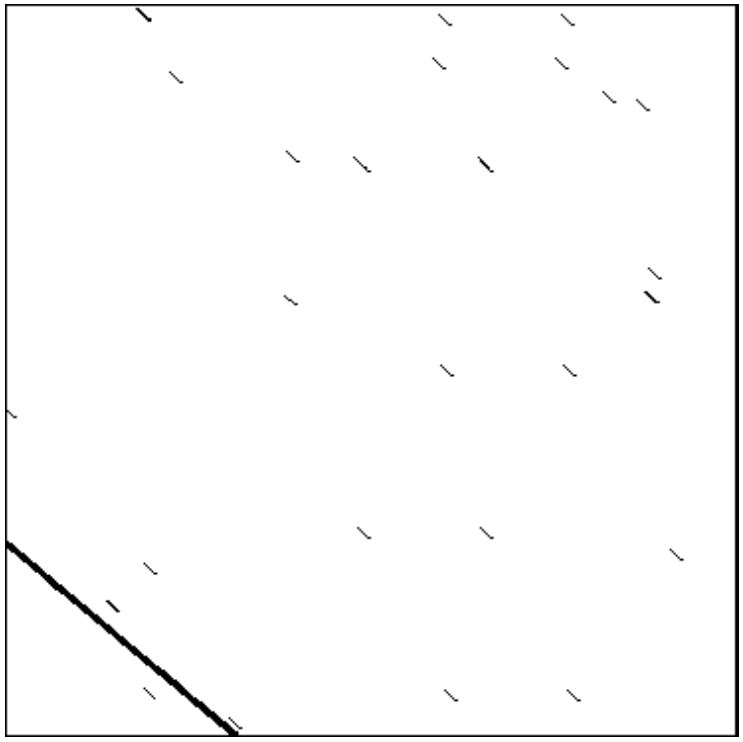


The Dot Matrix

An inversion:

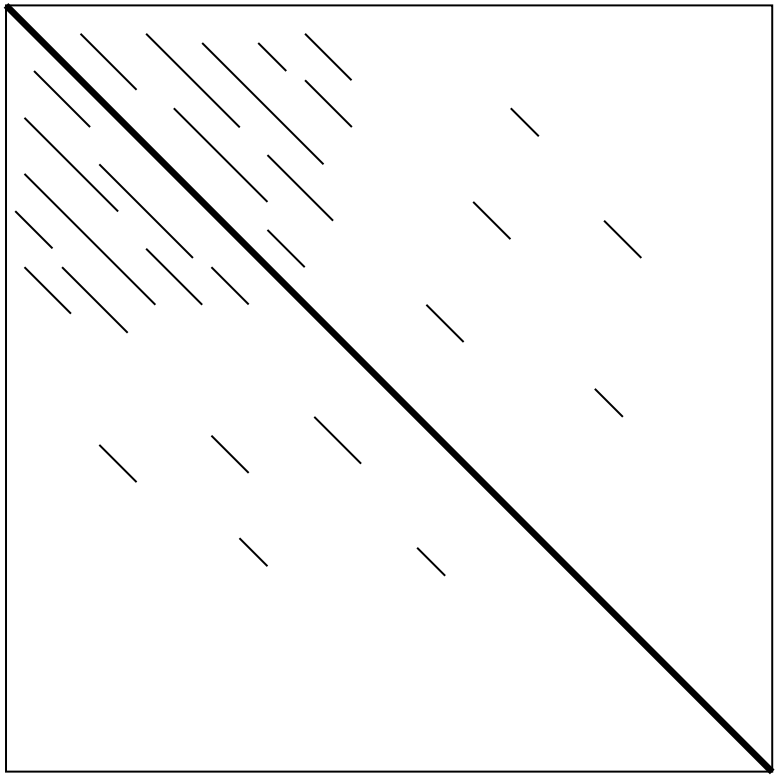


Joining sequences:



The Dot Matrix

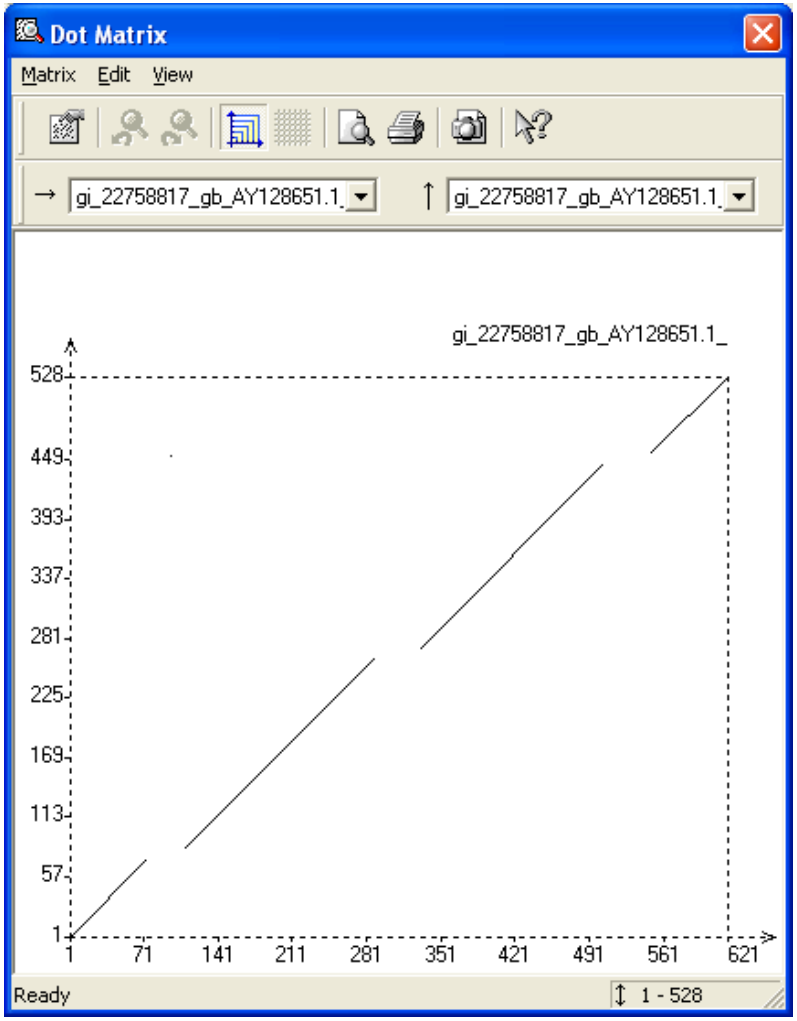
Self dotplot with repeats:



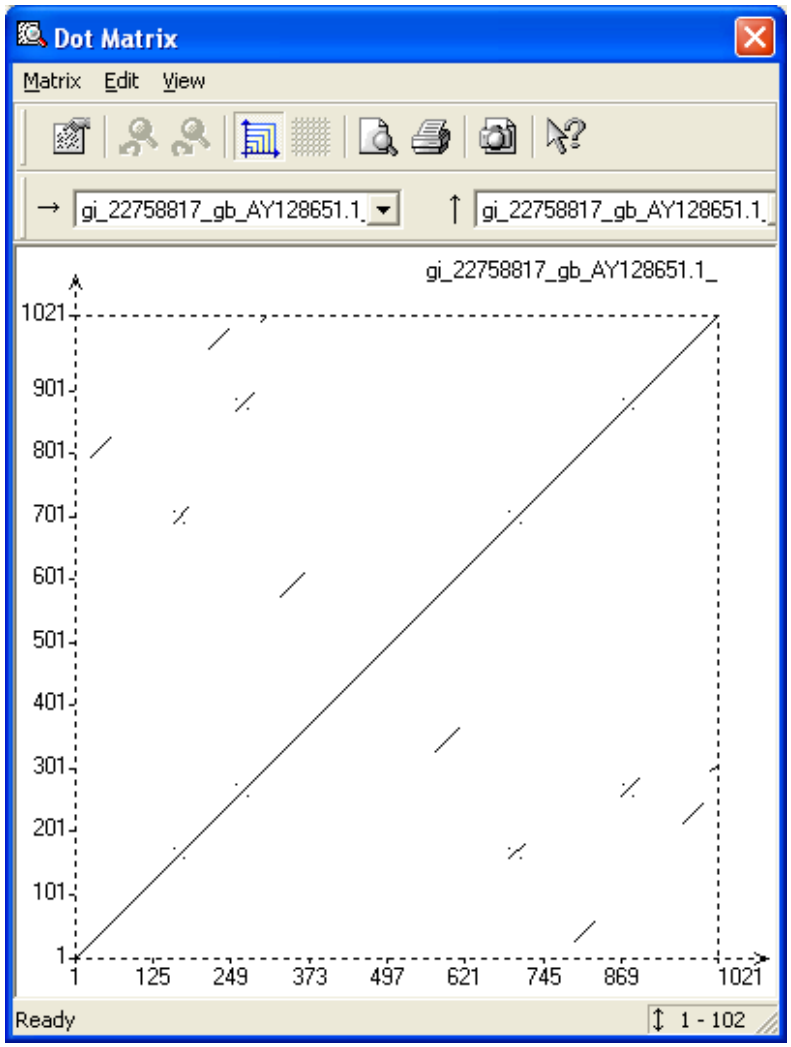
The Dot Matrix

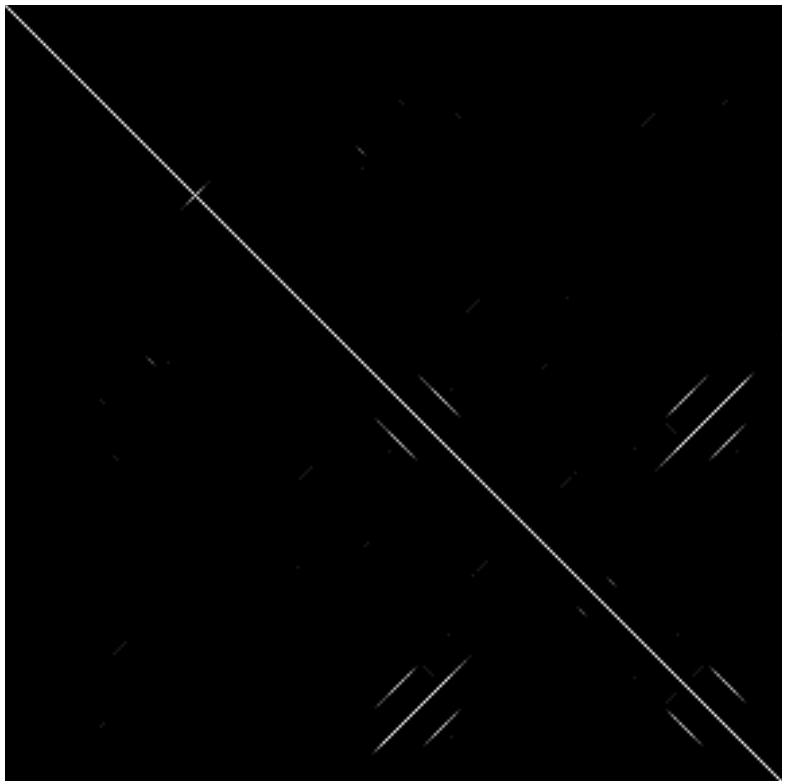
- the dot matrix method reveals the presence of insertions or deletions
- comparing a single sequence to itself can reveal the presence of a repeat of a subsequence
  - Inverted repeats = reverse complement
    - Used to determine folding of RNA molecules
- self comparison can reveal several features:
  - similarity between chromosomes
  - tandem genes
  - repeated domains in a protein sequence
  - regions of low sequence complexity (same characters are often repeated)

Insertions/Deletions

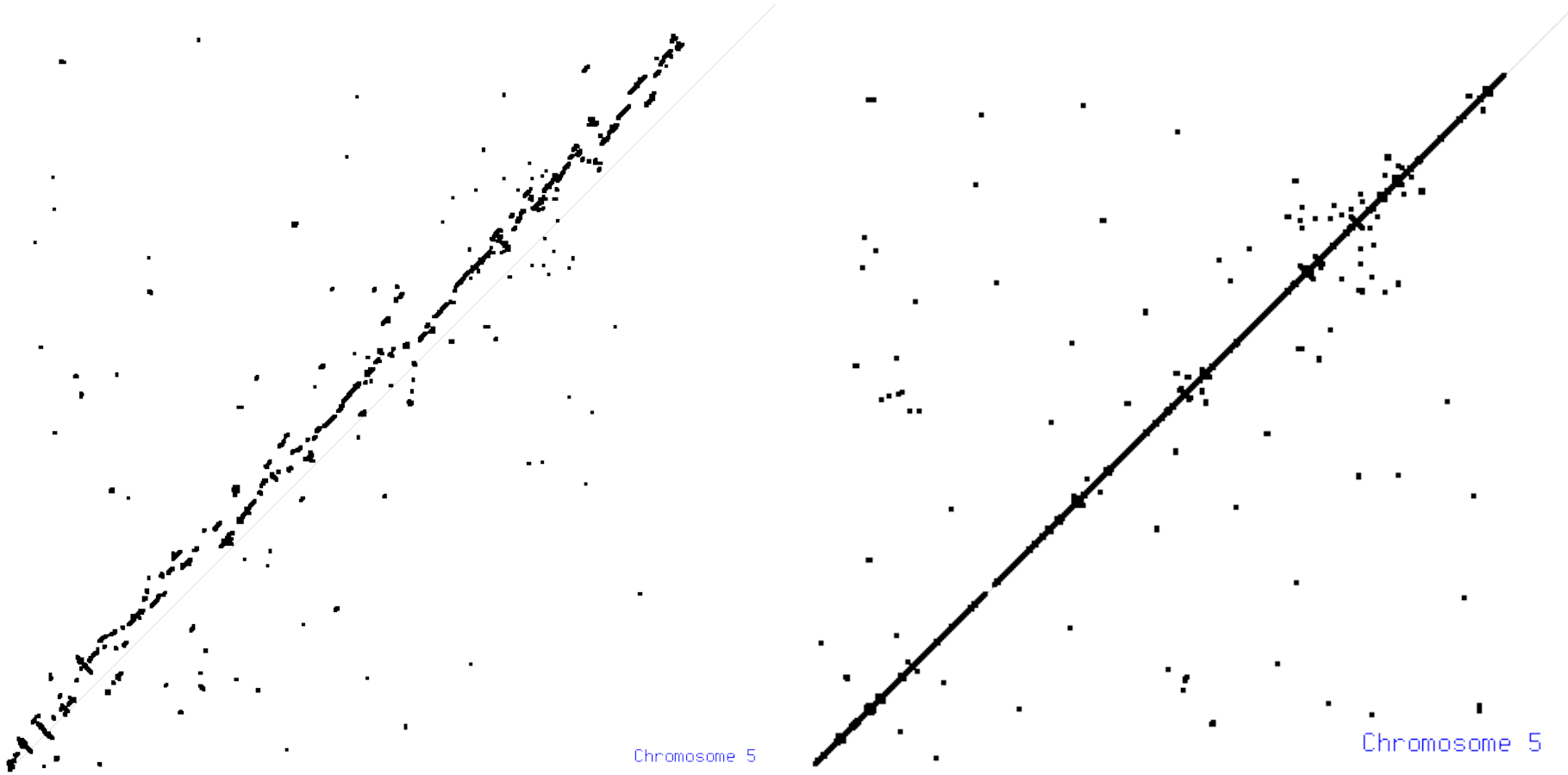


Repeats/Inverted Repeats

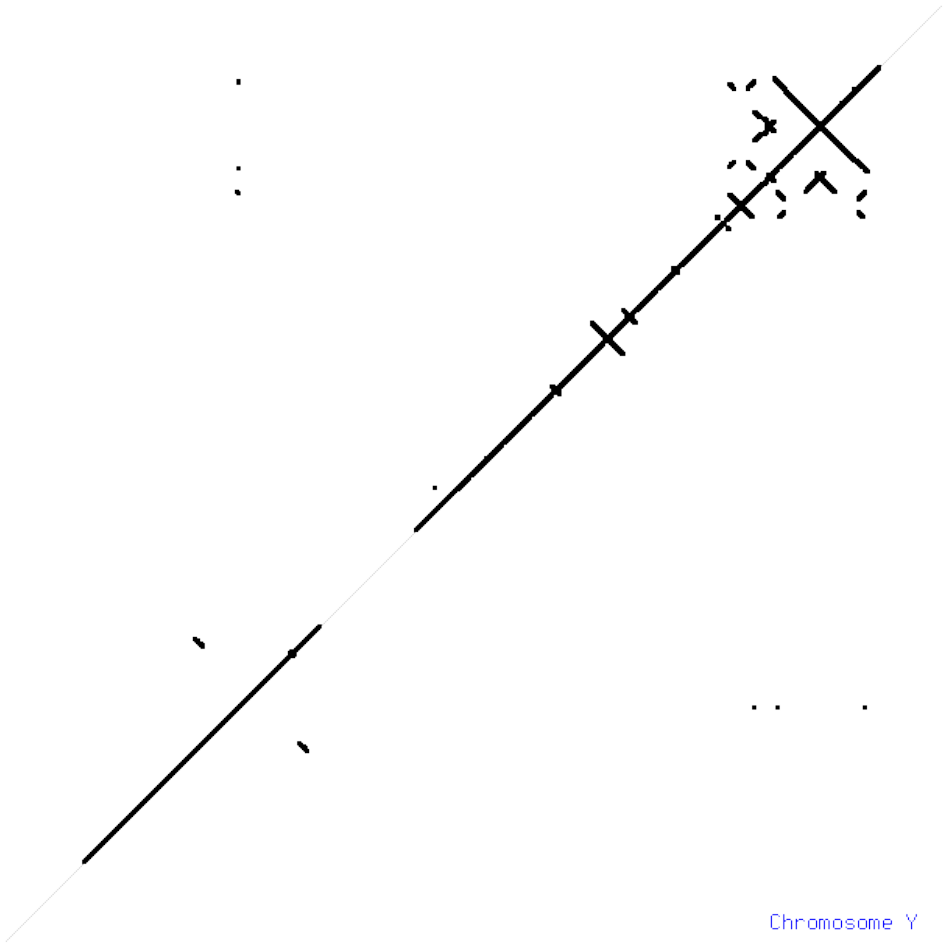




Comparing Genome Assemblies

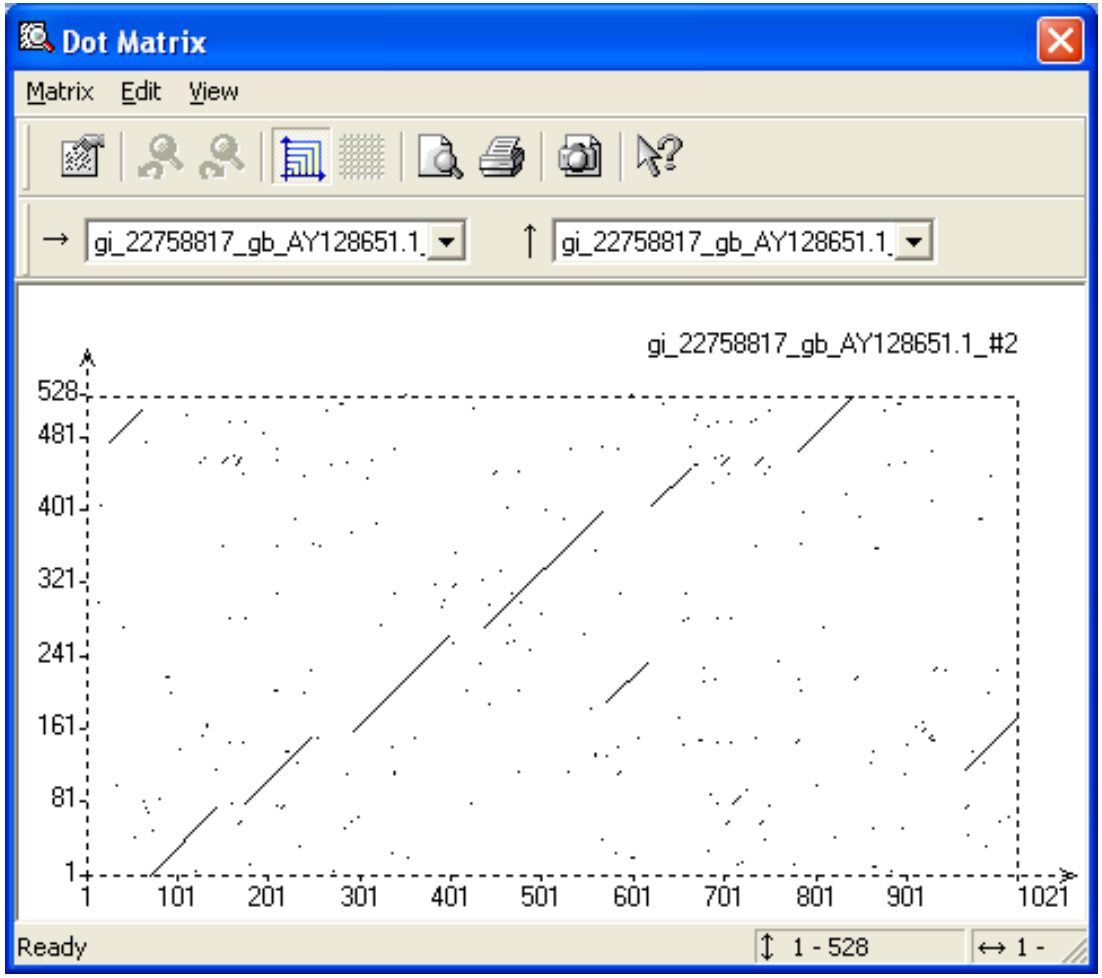


Chromosome Y self comparison



Available Dot Plot Programs

- Vector NTI software package (under AlignX)



Available Dot Plot Programs

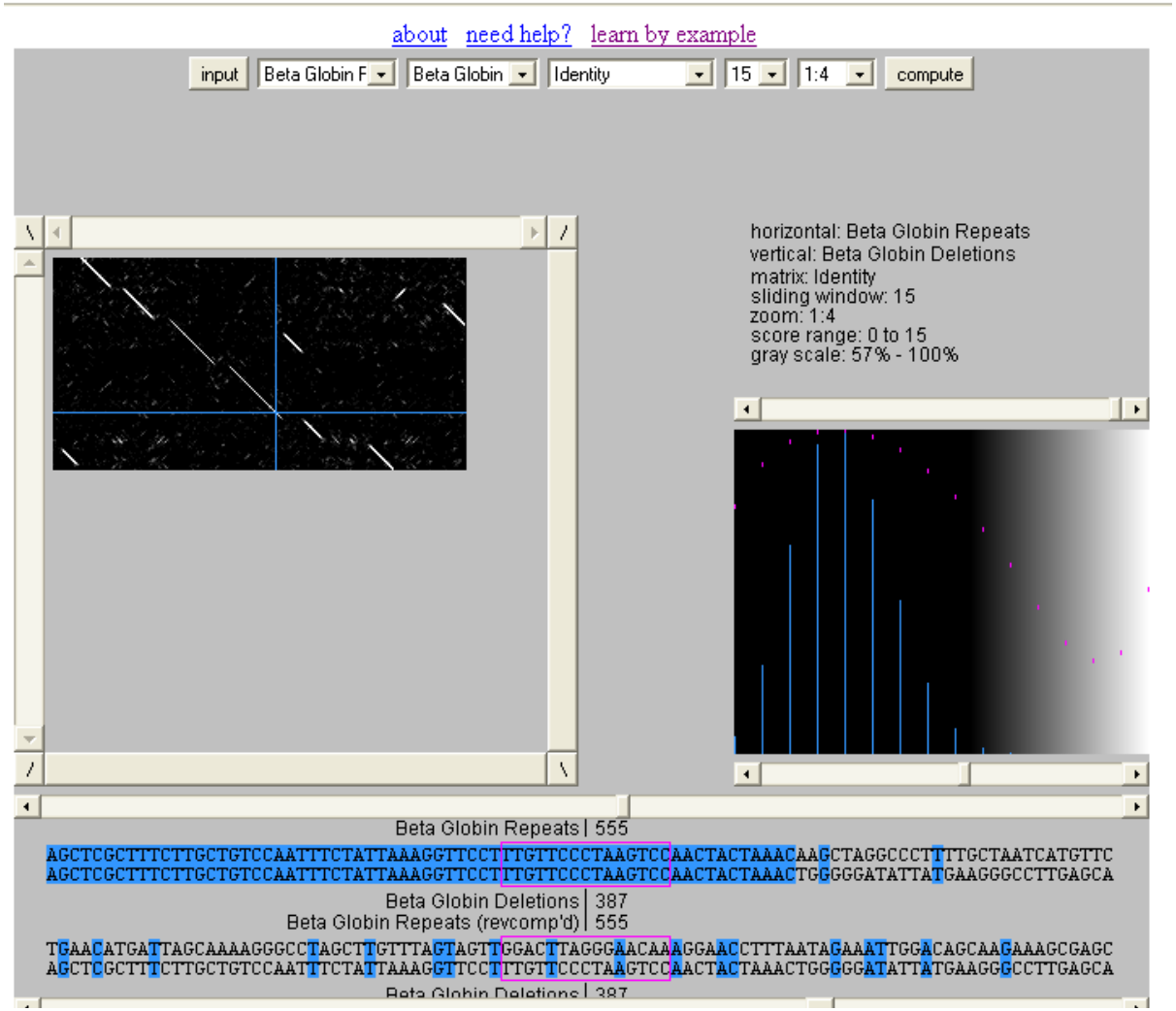
- Vector NTI software package (under AlignX)

GCG software package:

- Compare http://www.hku.hk/bruhk/gcgdoc/compare.html
- DotPlot+ http://www.hku.hk/bruhk/gcgdoc/dotplot.html
- http://www.isrec.isb-sib.ch/java/dotlet/Dotlet.html
- http://bioweb.pasteur.fr/cgi-bin/seqanal/dottup.pl
- Dotter (http://www.cgr.ki.se/cgr/groups/sonnhammer/Dotter.html)

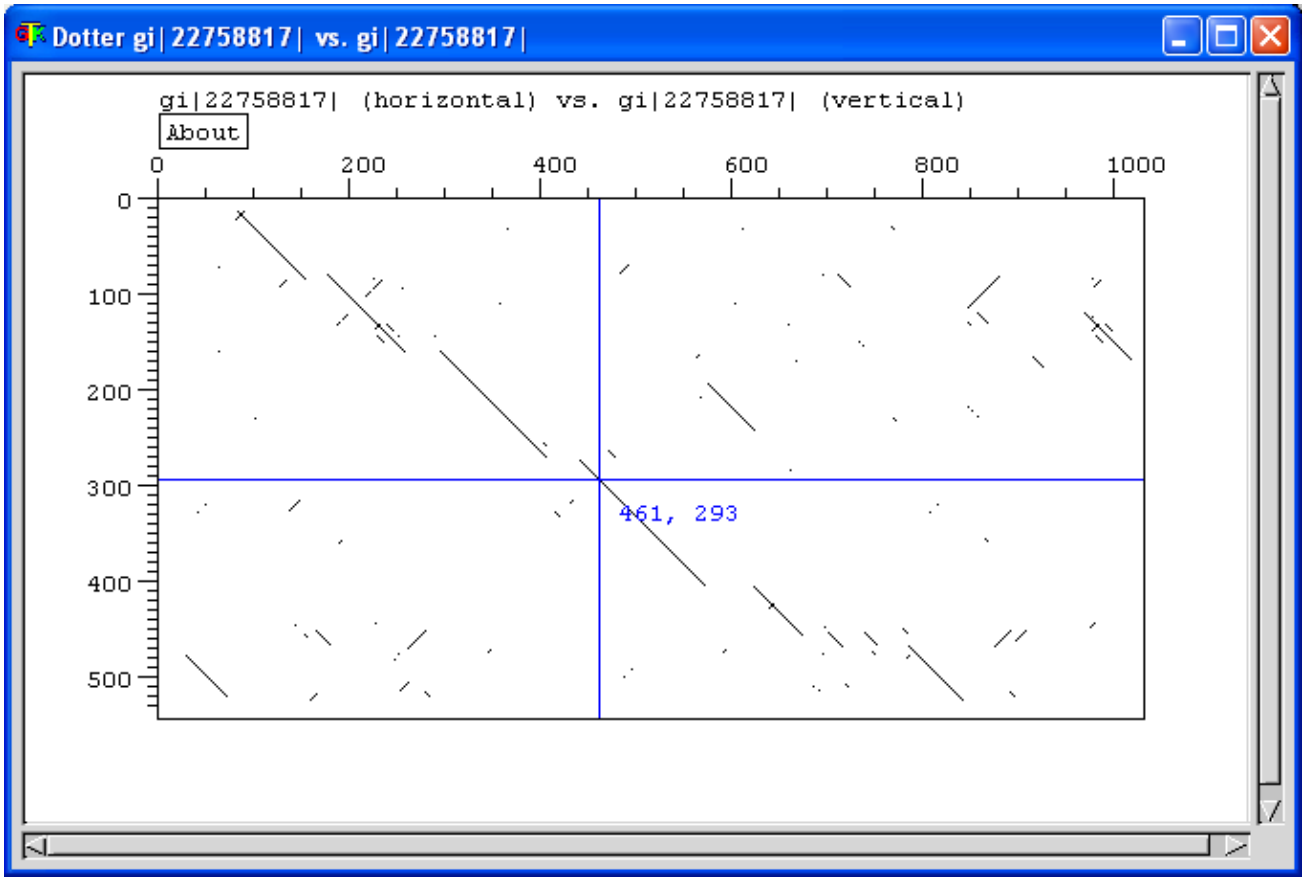
Available Dot Plot Programs

Dotlet (Java Applet) http://www.isrec.isb-sib.ch/java/dotlet/Dotlet.html



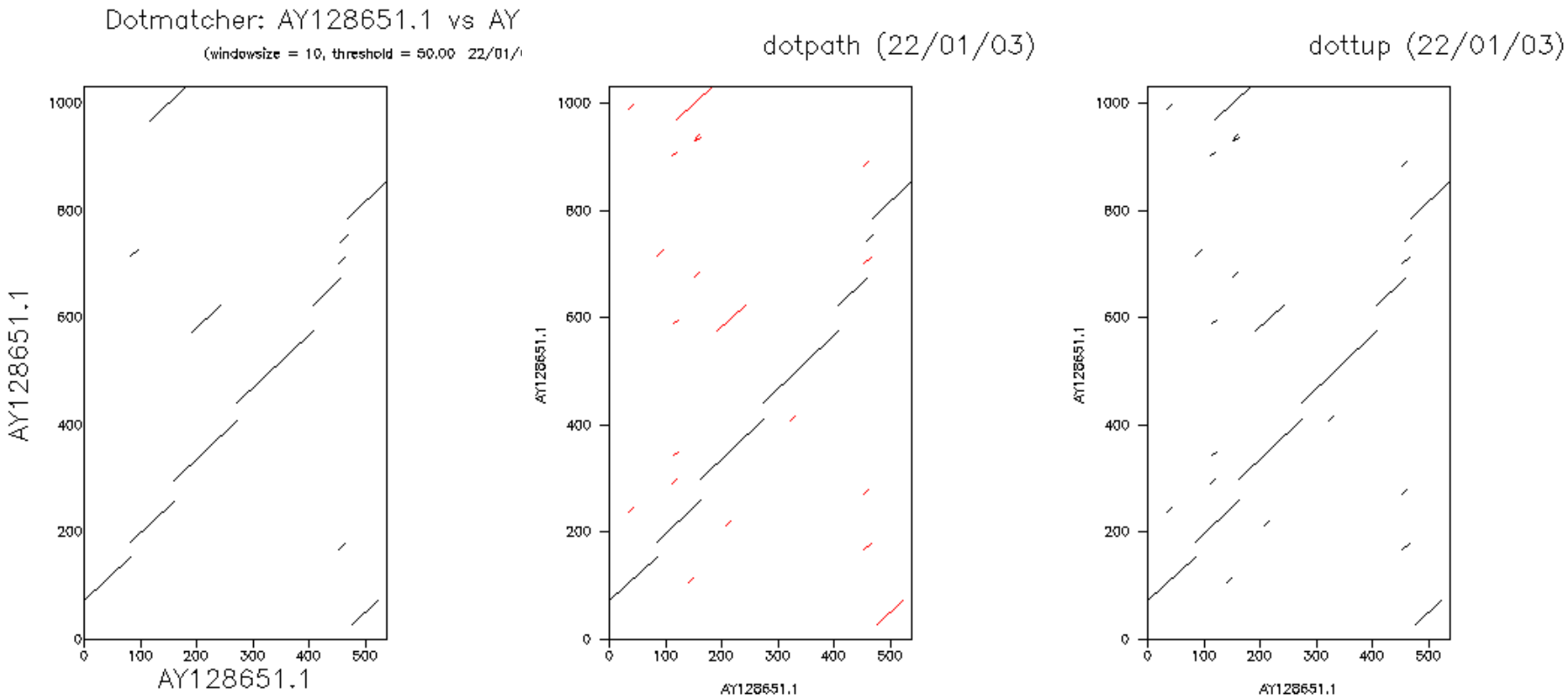
Available Dot Plot Programs

Dotter (http://www.cgr.ki.se/cgr/groups/sonnhammer/Dotter.html)



Available Dot Plot Programs

EMBOSS DotMatcher, DotPath,DotUp



The Dot Matrix

- When to use the Dot Matrix method?
  - unless the sequences are known to be very much alike
- limits of the Dot Matrix
  - doesn't readily resolve similarity that is interrupted by insertion or deletions
  - Difficult to find the best possible alignment (optimal alignment)
  - most computer programs don't show an actual alignment

Dot Plot References

Gibbs, A. J. & McIntyre, G. A. (1970).
The diagram method for comparing sequences. its use with amino acid and nucleotide sequences.
Eur. J. Biochem. **16**, 1-11.

Staden, R. (1982).
An interactive graphics program for comparing and aligning nucleic-acid and amino-acid sequences.
Nucl. Acid. Res. **10** (9), 2951-2961.

Next step

We must define quantitative measures of sequence similarity and difference!

- *Hamming* distance:
  - *# of positions with mismatching characters*

```
AGTC
CGTA
```

Hamming distance = 2

- *Levenshtein* (or *edit*) distance:
  - *# of operations required to change one string into the other (deletion, insertion, substitution)*

```
AG-TCC
CGCTCA
```

Levenshtein distance = 3

Scoring

- +1 for a match -1 for a mismatch?
- should gaps be allowed?
  - if yes how should they be scored?
- what is the best algorithm for finding the optimal alignment of two sequences?
- is the produced alignment significant?

Scoring Matrices

- match/mismatch score
  - Not bad for similar sequences
  - Does not show distantly related sequences

- Likelihood matrix
  - Scores residues dependent upon likelihood substitution is found in nature
  - More applicable for amino acid sequences

Parameters of Sequence Alignment

Scoring Systems:

- Each symbol pairing is assigned a numerical value, based on a symbol comparison table.

Gap Penalties:

- Opening: The cost to introduce a gap
- Extension: The cost to elongate a gap

DNA Scoring Systems-very simple

Sequence 1

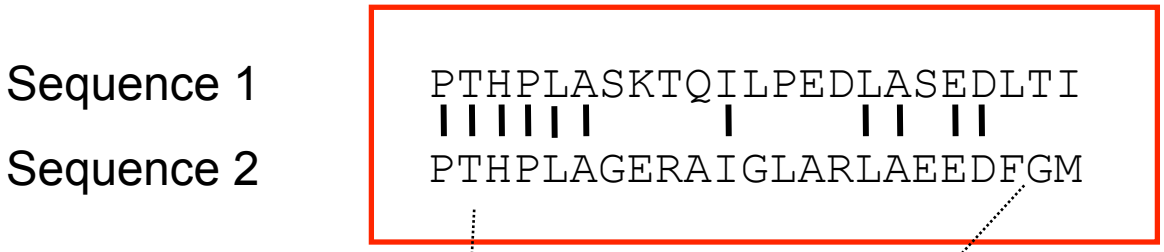
Sequence 2

```
actaccagttcatttgatacttctcaaa
     | |     |        ||
    taccattaccgtgttaactgaaaggacttaaagact
```

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

Match: 1
Mismatch: 0
Score = 5

Protein Scoring Systems



T:G = -2
T:T = 5
Score = 48

Scoring matrix

	C	S	T	P	A	G	N	D	.	.
C	9									
S	-1	4								
T	-1	1	5							
P	-3	-1	-1	7						
A	0	1	0	-1	4					
G	-3	0	-2	-2	0	6				
N	-3	1	0	-2	-2	0	5			
D	-3	0	-1	-1	-2	-1	1	6		
.										
.										

A scoring matrix is a table of values that describe the probability of a residue pair occurring in alignment.

Amino acid exchange matrices

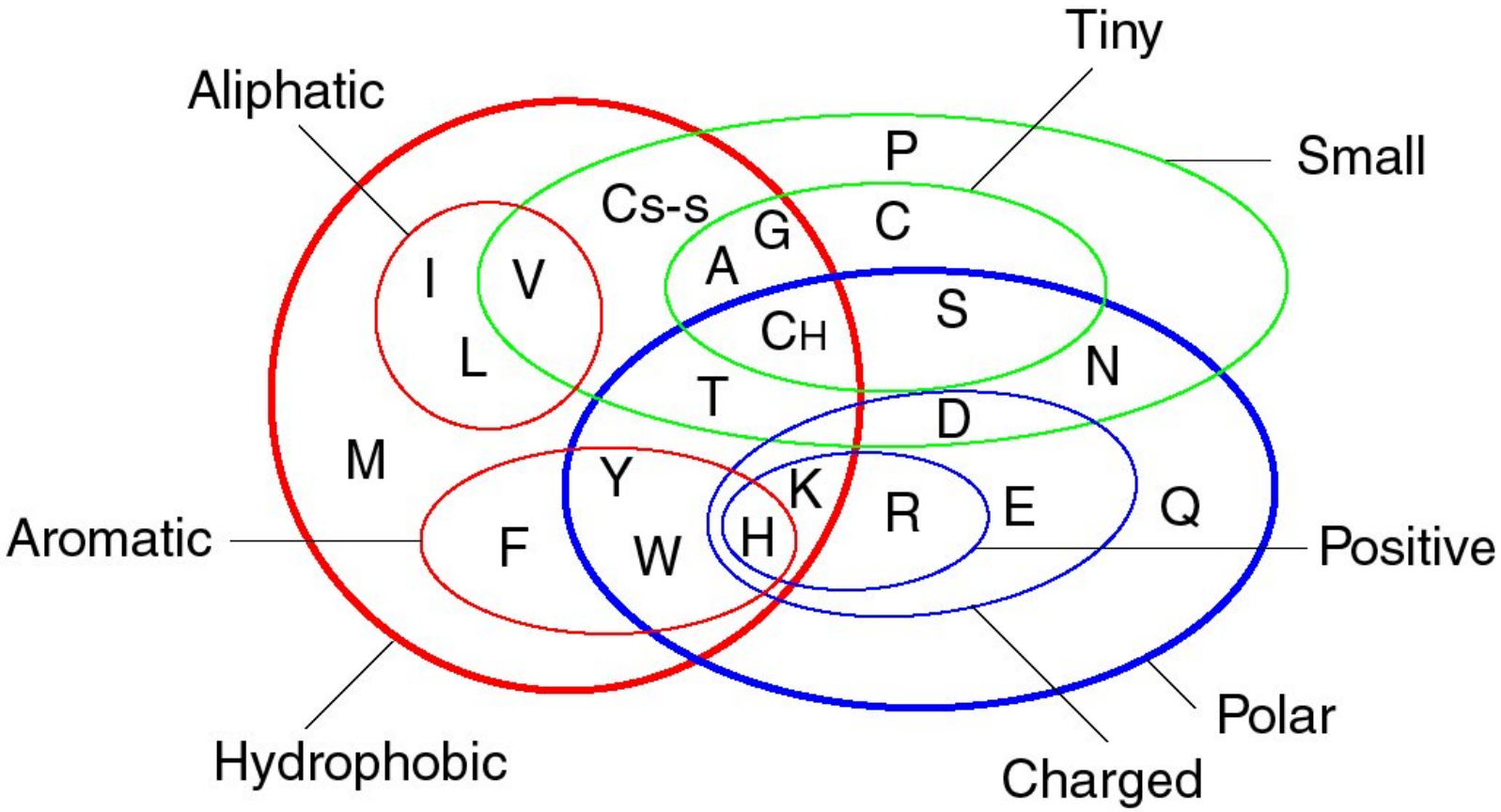
Amino acids are **not** equal:

1. Some are easily substituted because they have similar:
   - physico-chemical properties
   - structure
2. Some mutations between amino acids occur more often due to similar codons

The two above observations give us ways to define *substitution matrices*

Properties of Amino Acids

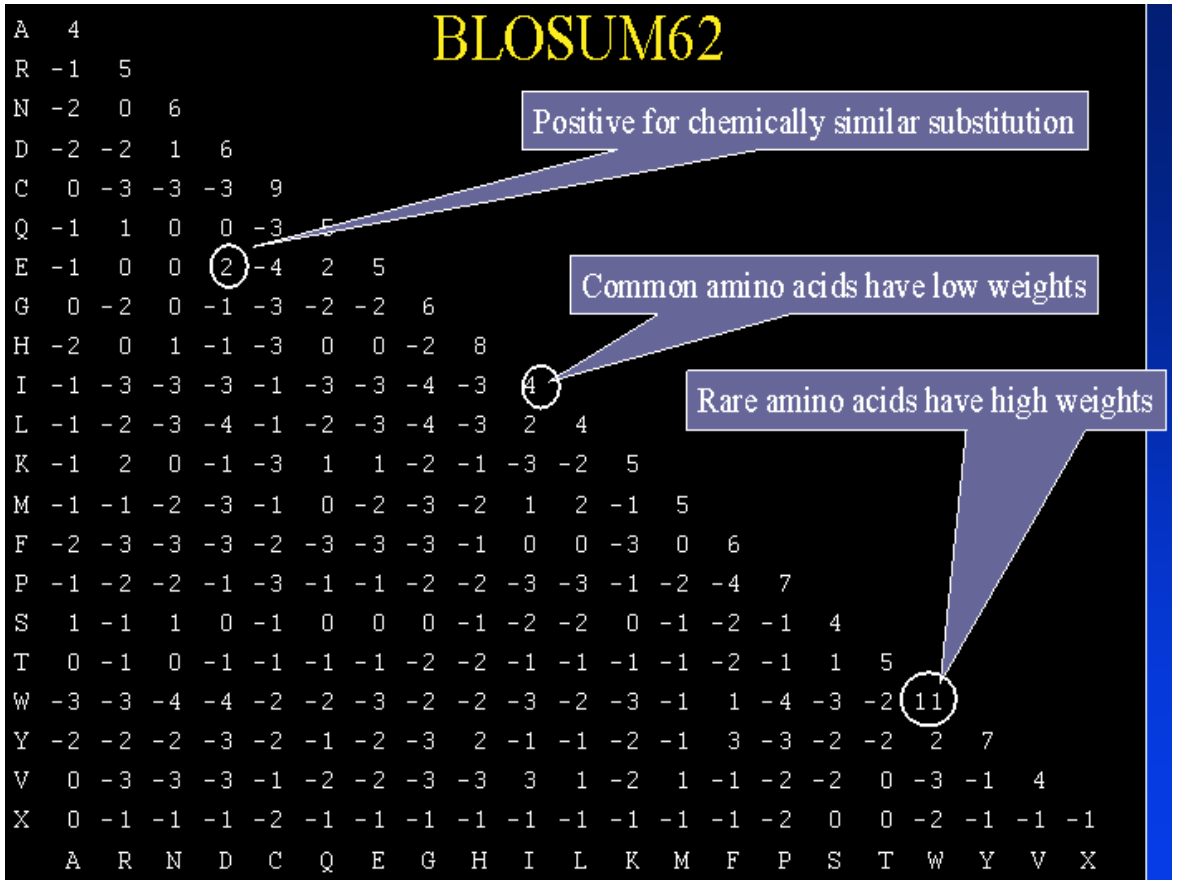
- Substitutions with similar chemical properties
- Amino acids have different biochemical and physical properties that influence their relative replaceability in evolution.



Scoring Matrices

- table of values that describe the probability of a residue pair occurring in an alignment
- the values are logarithms of ratios of two probabilities

1. probability of random occurrence of an amino acid (diagonal)
2. probability of meaningful occurrence of a pair of residues



Protein Scoring Systems

- Scoring matrices reflect:
  - # of mutations to convert one to another
  - chemical similarity
  - observed mutation frequencies
- Log odds matrices:
  - the values are logarithms of probability ratios of the probability of an aligned pair to the probability of a random alignment.
- Widely used scoring matrices:
  - PAM
  - BLOSUM

Scoring Matrices

Widely used matrices

- **PAM** (Percent Accepted Mutation) / **MDM** (Mutation Data Matrix) / **Dayhoff**
  - Derived from *global* alignments of *closely* related sequences.
  - Matrices for greater evolutionary distances are extrapolated from those for lesser ones.
  - The number with the matrix (PAM40, PAM100) refers to the evolutionary distance; greater numbers are greater distances.
  - PAM-1 corresponds to about 1 million years of evolution
  - for distant (global) alignments, Blosum50, Gonnet, or (still) PAM250

- **BLOSUM** (Blocks Substitution Matrix)
  - Derived from *local, ungapped alignments of distantly related sequences*
  - *All matrices are directly calculated; no extrapolations are used*
  - *The number after the matrix (BLOSUM62) refers to the minimum percent identity of the blocks used to construct the matrix; greater numbers are lesser distances.*
  - *The BLOSUM series of matrices generally perform better than PAM matrices for local similarity searches.*
  - For local alignment, Blosum 62 is often superior

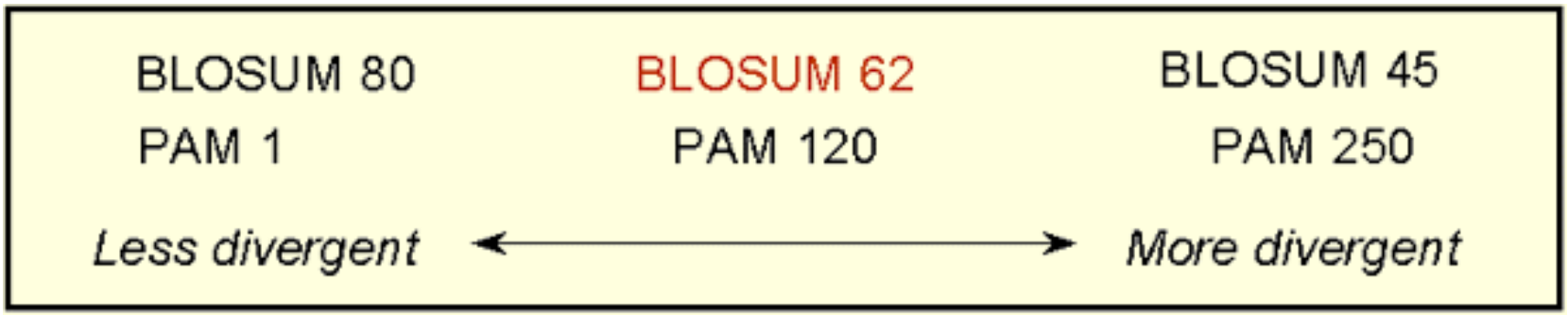
- ***Structure-based matrices***

- ***Specialized Matrices***

Scoring Matrices

The relationship between BLOSUM and PAM substitution matrices

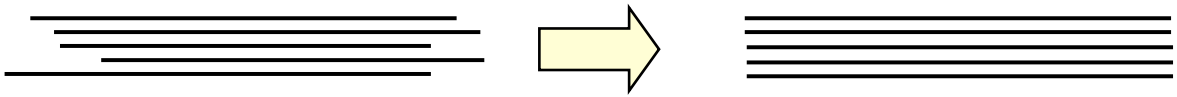
- BLOSUM matrices with higher numbers and PAM matrices with low numbers are designed for comparisons of closely related sequences.
- BLOSUM matrices with low numbers and PAM matrices with high numbers are designed for comparisons of distantly related proteins.



http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/Scoring2.html

Percent Accepted Mutation (PAM or Dayhoff) Matrices

- Studied by Margaret Dayhoff (Dayhoff *et al.*, 1978).
- Amino acid substitutions
  - Alignment of common protein sequences
  - 1572 amino acid substitutions
  - 71 groups of protein, 85% similar



Derived from global alignments of protein families. Family members share at least 85% identity.

- "Accepted" mutations – do not negatively affect a protein's fitness
- Similar sequences organized into phylogenetic trees
- Number of amino acid changes counted
- Relative mutabilities evaluated
- 20 x 20 amino acid substitution matrix calculated

Percent Accepted Mutation (PAM or Dayhoff) Matrices

- PAM 1: 1 accepted mutation event per 100 amino acids; PAM 250: 250 mutation events per 100 …
- PAM 1 matrix can be multiplied by itself N times to give transition matrices for sequences that have undergone N mutations
- PAM 250: 20% similar; PAM 120: 40%; PAM 80: 50%; PAM 60: 60%

PAM1 matrix

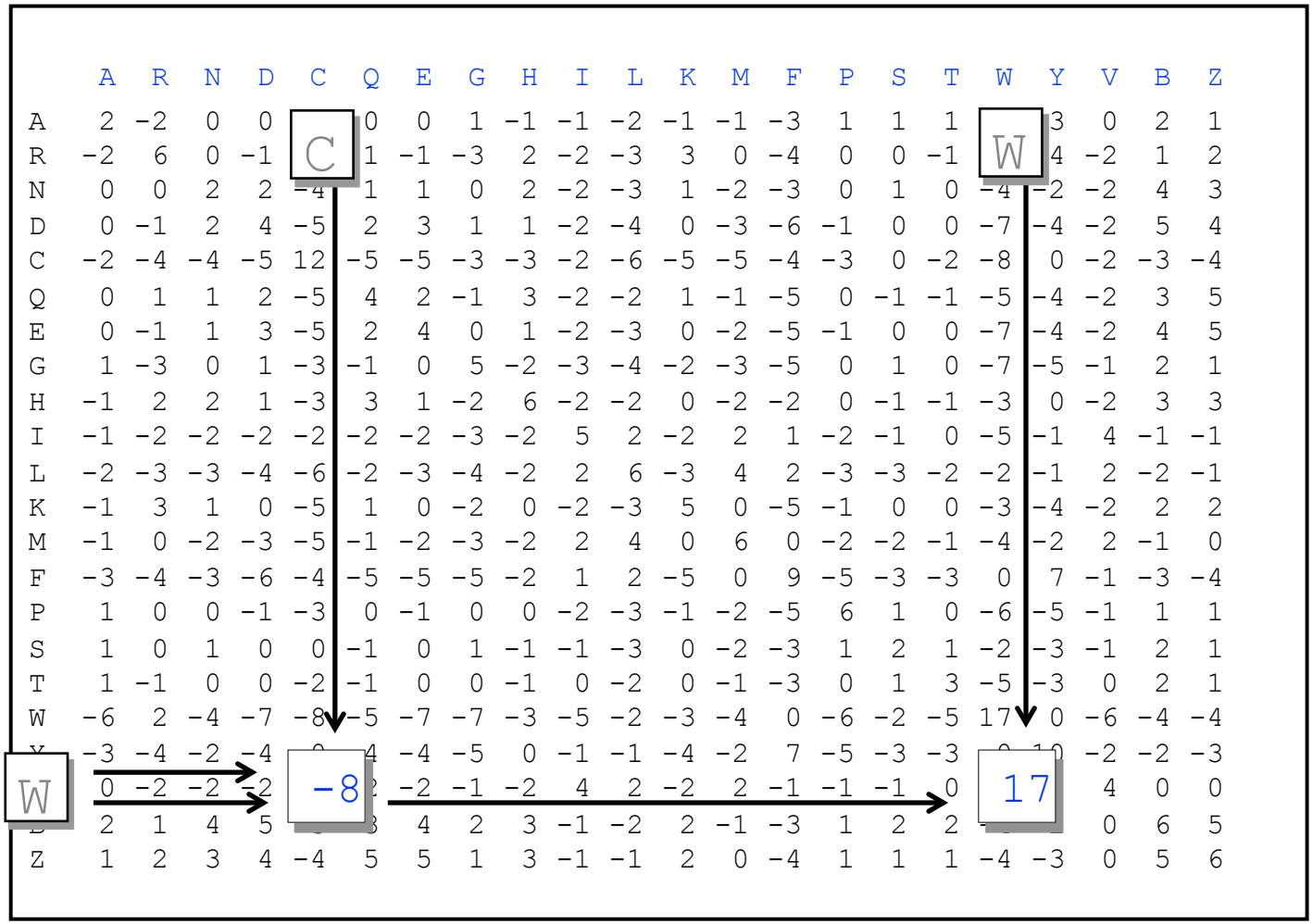
normalized probabilities multiplied by 10000

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	9867	2	9	10	3	8	17	21	2	6	4	2	6	2	22	35	32	0	2	18
R	1	9913	1	0	1	10	0	0	10	3	1	19	4	1	4	6	1	8	0	1
N	4	1	9822	36	0	4	6	6	21	3	1	13	0	1	2	20	9	1	4	1
D	6	0	42	9859	0	6	53	6	4	1	0	3	0	0	1	5	3	0	0	1
C	1	1	0	0	9973	0	0	0	1	1	0	0	0	0	1	5	1	0	3	2
Q	3	9	4	5	0	9876	27	1	23	1	3	6	4	0	6	2	2	0	0	1
E	10	0	7	56	0	35	9865	4	2	3	1	4	1	0	3	4	2	0	1	2
G	21	1	12	11	1	3	7	9935	1	0	1	2	1	1	3	21	3	0	0	5
H	1	8	18	3	1	20	1	0	9912	0	1	1	0	2	3	1	1	1	4	1
I	2	2	3	1	2	1	2	0	0	9872	9	2	12	7	0	1	7	0	1	33
L	3	1	3	0	0	6	1	1	4	22	9947	2	45	13	3	1	3	4	2	15
K	2	37	25	6	0	12	7	2	2	4	1	9926	20	0	3	8	11	0	1	1
M	1	1	0	0	0	2	0	0	0	5	8	4	9874	1	0	1	2	0	0	4
F	1	1	1	0	0	0	0	1	2	8	6	0	4	9946	0	2	1	3	28	0
P	13	5	2	1	1	8	3	2	5	1	2	2	1	1	9926	12	4	0	0	2
S	28	11	34	7	11	4	6	16	2	2	1	7	4	3	17	9840	38	5	2	2
T	22	2	13	4	1	3	2	2	1	11	2	8	6	1	5	32	9871	0	2	9
W	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	9976	1	0
Y	1	0	3	0	3	0	1	0	4	1	1	0	0	21	0	1	1	2	9945	1
V	13	2	1	1	3	2	2	3	3	57	11	1	17	1	3	2	10	0	2	9901

Log Odds Matrices

- PAM matrices converted to log-odds matrix
  - Calculate odds ratio for each substitution
    - Taking scores in previous matrix
    - Divide by frequency of amino acid
  - Convert ratio to log10 and multiply by 10
  - Take average of log odds ratio for converting A to B and converting B to A
  - Result: Symmetric matrix

PAM 250



	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z
A	2	-2	0	0	-2	0	0	1	-1	-1	-2	-1	-1	-3	1	1	1	-6	-3	0	2	1
R	-2	6	0	-1	-4	1	-1	-3	2	-2	-3	3	0	-4	0	0	-1	2	-4	-2	1	2
N	0	0	2	2	-4	1	1	0	2	-2	-3	1	-2	-3	0	1	0	-4	-2	-2	4	3
D	0	-1	2	4	-5	2	3	1	1	-2	-4	0	-3	-6	-1	0	0	-7	-4	-2	5	4
C	-2	-4	-4	-5	12	-5	-5	-3	-3	-2	-6	-5	-5	-4	-3	0	-2	-8	0	-2	-3	-4
Q	0	1	1	2	-5	4	2	-1	3	-2	-2	1	-1	-5	0	-1	-1	-5	-4	-2	3	5
E	0	-1	1	3	-5	2	4	0	1	-2	-3	0	-2	-5	-1	0	0	-7	-4	-2	4	5
G	1	-3	0	1	-3	-1	0	5	-2	-3	-4	-2	-3	-5	0	1	0	-7	-5	-1	2	1
H	-1	2	2	1	-3	3	1	-2	6	-2	-2	0	-2	-2	0	-1	-1	-3	0	-2	3	3
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5	2	-2	2	1	-2	-1	0	-5	-1	4	-1	-1
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6	-3	4	2	-3	-3	-2	-2	-1	2	-2	-1
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5	0	-5	-1	0	0	-3	-4	-2	2	2
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6	0	-2	-2	-1	-4	-2	2	-1	0
F	-3	-4	-3	-6	-4	-5	-5	-5	-2	1	2	-5	0	9	-5	-3	-3	0	7	-1	-3	-4
P	1	0	0	-1	-3	0	-1	0	0	-2	-3	-1	-2	-5	6	1	0	-6	-5	-1	1	1
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	2	1	-2	-3	-1	2	1
T	1	-1	0	0	-2	-1	0	0	-1	0	-2	0	-1	-3	0	1	3	-5	-3	0	2	1
W	-6	2	-4	-7	-8	-5	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	17	0	-6	-4	-4
Y	-3	-4	-2	-4	0	-4	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	0	10	-2	-2	-3
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	-2	4	0	0
B	2	1	4	5	-3	3	4	2	3	-1	-2	2	-1	-3	1	2	2	-4	-3	0	6	5
Z	1	2	3	4	-4	5	5	1	3	-1	-1	2	0	-4	1	1	1	-4	-3	0	5	6

A value of 0 indicates the frequency of alignment is random

log(freq(observed)/freq(expected))

PAM250 Log odds matrix

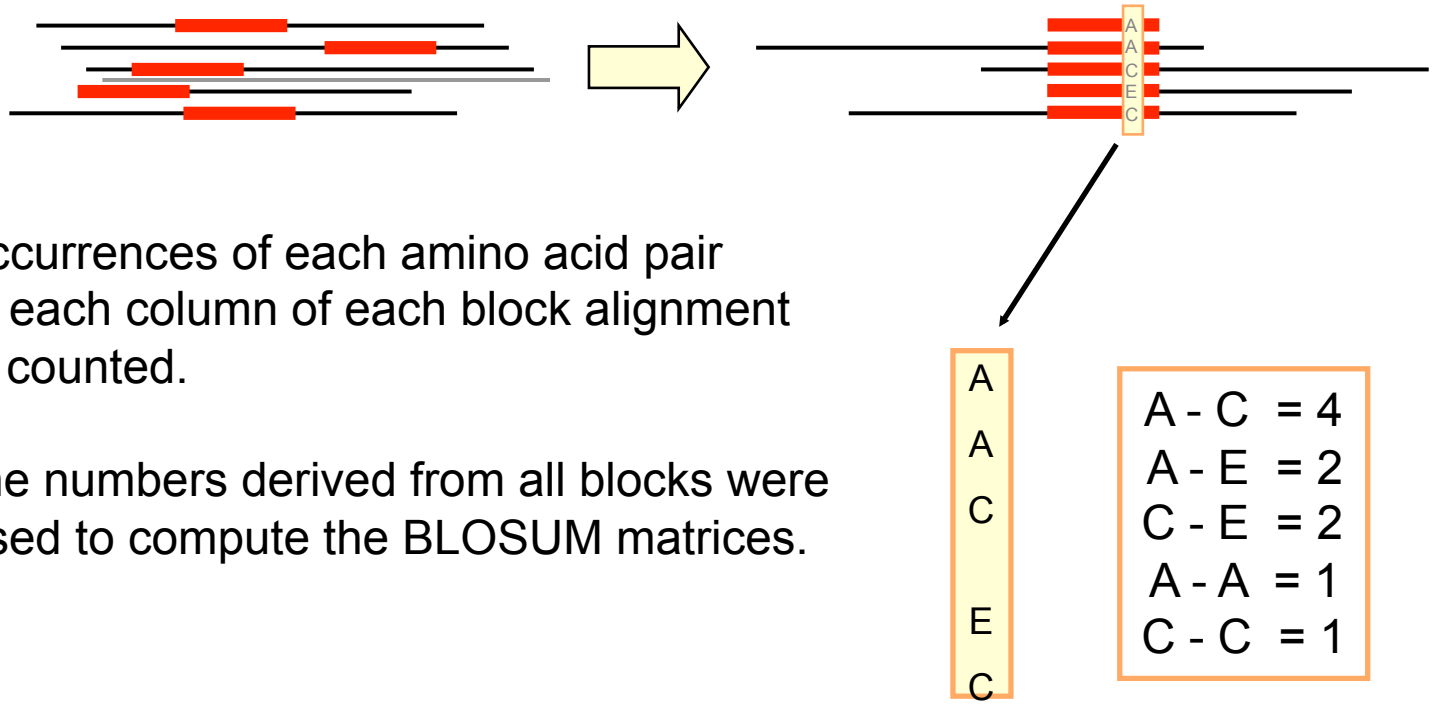
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	12																				C
S	0	2																			S
T	-2	1	3																		T
P	-3	1	0	6																	P
A	-2	1	1	1	2																A
G	-3	1	0	-1	1	5															G
N	-4	1	0	-1	0	0	2														N
D	-5	0	0	-1	0	1	2	4													D
E	-5	0	0	-1	0	0	1	3	4												E
Q	-5	-1	-1	0	0	-1	1	2	2	4											Q
H	-3	-1	-1	0	-1	-2	2	1	1	3	6										H
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6									R
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5								K
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6							M
I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5						I
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6					L
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4				V
F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9			F
Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10		Y
W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Blocks Amino Acid Substitution Matrices (BLOSUM)

- Larger set of sequences considered
- Sequences organized into signature blocks
- Consensus sequence formed
  - 60% identical: BLOSUM 60
  - 80% identical: BLOSUM 80

BLOSUM (Blocks Substitution Matrix)

- Derived from alignments of domains of distantly related proteins (Henikoff & Henikoff,1992).



- Occurrences of each amino acid pair in each column of each block alignment is counted.

- The numbers derived from all blocks were used to compute the BLOSUM matrices.

BLOSUM (Blocks Substitution Matrix)

- Sequences within blocks are clustered according to their level of identity.
- Clusters are counted as a single sequence.
- Different BLOSUM matrices differ in the percentage of sequence identity used in clustering.
- The number in the matrix name (e.g. 62 in BLOSUM62) refers to the percentage of sequence identity used to build the matrix.
- Greater numbers mean smaller evolutionary distance.

AMINO ACID FREQUENCY

- Genetic information contained in mRNA is in the form of codons, which are translated into amino acids which then combine to form proteins.
- At certain sites in a protein's structure, amino acid composition is not critical.
- Yet certain amino acids occur at such sites up to six times more often than other amino acids.
- Are frequencies of particular amino acids simply a consequence of random permutations of the genetic code or instead a product of natural selection?

AMINO ACID FREQUENCY

- If a particular amino acid is in some way adaptive, then it should occur more frequently than expected by chance.
- This can easily be tested by calculating the expected frequencies of amino acids and comparing to observed.
- The codons and observed frequencies of particular amino acids are given in the next slide.

The codons and observed frequencies of amino acids

Amino Acids	Codons	Observed Frequency in Vertebrates
Alanine	GCU, GCA, GCC, GCG	7.4 %
Arginine	CGU, CGA, CGC,CGG, AGA, AGG	4.2 %
Asparagine	AAU, AAC	4.4 %
Aspartic Acid	GAU, GAC	5.9 %
Cysteine	UGU, UGC	3.3 %
Glutamic Acid	GAA, GAG	5.8 %
Glutamine	CAA, CAG	3.7 %
Glycine	GGU, GGA, GGC, GGG	7.4 %
Histidine	CAU, CAC	2.9 %

The codons and observed frequencies of amino acids

Isoleucine	AUU, AUA, AUC	3.8 %
Leucine	CUU, CUA, CUC, CUG, UUA, UUG	7.6 %
Lysine	AAA, AAG	7.2 %
Methionine	AUG	1.8 %
Phenylalanine	UUU, UUC	4.0 %
Proline	CCU, CCA, CCC, CCG	5.0 %
Serine	UCU, UCA, UCC, UCG, AGU, AGC	8.1 %
Threonine	ACU, ACA, ACC, ACG	6.2 %
Tryptophan	UGG	1.3 %
Tyrosine	UAU, UAC	3.3 %
Valine	GUU, GUA, GUC, GUG	6.8 %
Stop Codons	UAA, UAG, UGA	---

AMINO ACID FREQUENCY

- The frequencies of DNA bases in nature are
  - 22.0% uracil,
  - 30.3% adenine,
  - 21.7% cytosine,
  - 26.1% guanine.
- The expected frequency of a particular codon can then be calculated by
  - multiplying the frequencies of each DNA base comprising the codon.

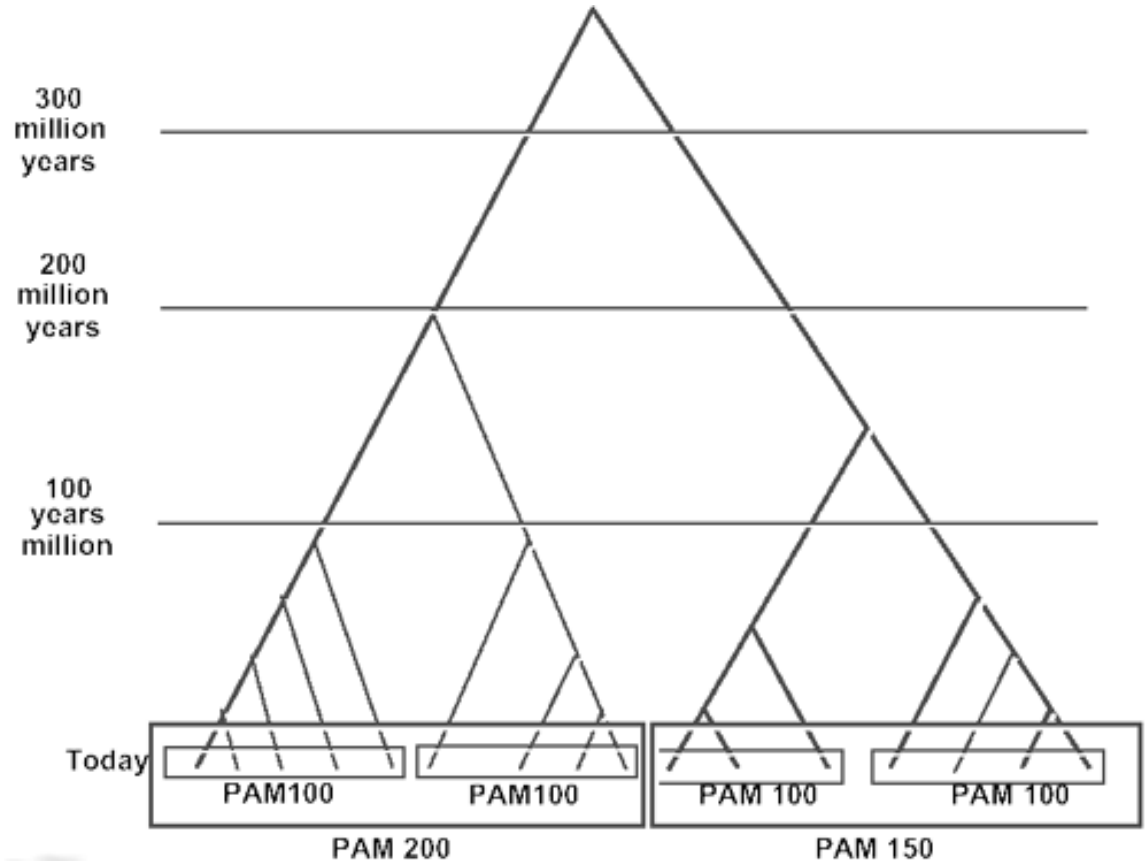
AMINO ACID FREQUENCY

- The expected frequency of the amino acid can then be calculated by adding the frequencies of each codon that codes for that amino acid.
- As an example,
  - the RNA codons for tyrosine are UAU and UAC, so the random expectation for its frequency is
    - (0.220)(0.303)(0.220) + (0.220)(0.303)(0.217) = 0.0292.
  - Since 3 of the 64 codons are nonsense or stop codons, this frequency for each amino acid is multiplied by a correction factor of 1.057.

TIPS on choosing a scoring matrix

- Generally, BLOSUM matrices perform better than PAM matrices for local similarity searches (Henikoff & Henikoff, 1993).
- When comparing closely related proteins one should use lower PAM or higher BLOSUM matrices,
- For distantly related proteins higher PAM or lower BLOSUM matrices.
- For database searching the commonly used matrix is BLOSUM62.

Use Different PAM's for Different Evolutionary Distances



(Adapted from D Brutlag, Stanford)

Nucleic Acid Scoring Scheme

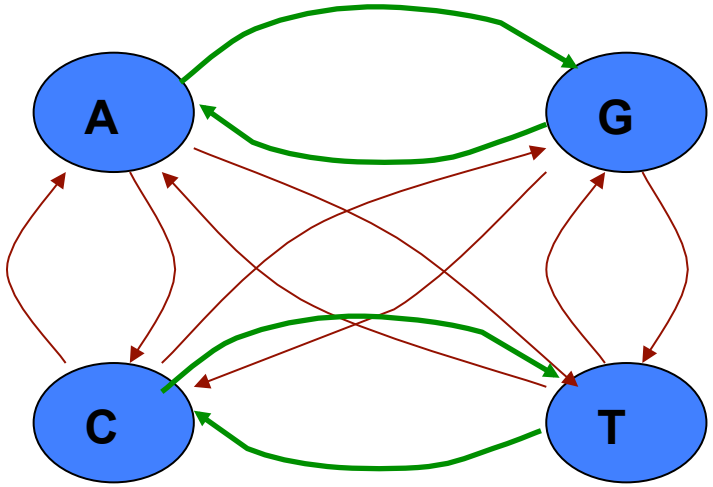
- Transition mutation (more common)
  - purine ⟷ purine A ⟷ G
  - pyrimidine ⟷ pyrimidine T ⟷ C

- Transversion mutation
  - purine ⟷ pyrimidine A, G ⟷ T, C

	A	G	T	C
A	20	10	5	5
G	10	20	5	5
T	5	5	20	10
C	5	5	10	20

DNA Mutations

In addition to using a match/mismatch scoring scheme for DNA sequences, nucleotide mutation matrices can be constructed as well. These matrices are based upon two different models of nucleotide evolution: the first, the Jukes-Cantor model, assumes there are uniform mutation rates among nucleotides, while the second, the Kimura model, assumes that there are two separate mutation rates: one for transitions (where the structure of purine/pyrimidine stays the same), and one for transversions. Generally, the rate of transitions is thought to be higher than the rate of transversions.



PURINES: A, G
PYRIMIDINES C, T

Transitions: A↔G; C↔T
Transversions: A↔C, A↔T, C↔G, G↔T

Nucleic Acid Scoring Matrices

- Two mutation models:
  - Jukes-Cantor Model of evolution: α = common rate of base substitution
  - Kimura Model of Evolution: α = rate of transitions; β = rate of transversions
    - Transitions
    - Transversions

$$R = \begin{array}{c} \\ A \\ C \\ G \\ U \end{array} \begin{array}{c} \begin{array}{cccc} A & C & G & U \end{array} \\ \begin{pmatrix} * & 0.25\alpha & 0.25\alpha & 0.25\alpha \\ 0.25\alpha & * & 0.25\alpha & 0.25\alpha \\ 0.25\alpha & 0.25\alpha & * & 0.25\alpha \\ 0.25\alpha & 0.25\alpha & 0.25\alpha & * \end{pmatrix} \end{array}$$

$$R = \begin{array}{c} \\ A \\ C \\ G \\ U \end{array} \begin{array}{c} \begin{array}{cccc} A & C & G & U \end{array} \\ \begin{pmatrix} * & 0.25\beta & 0.25\alpha & 0.25\beta \\ 0.25\beta & * & 0.25\beta & 0.25\alpha \\ 0.25\alpha & 0.25\beta & * & 0.25\beta \\ 0.25\beta & 0.25\alpha & 0.25\beta & * \end{pmatrix} \end{array}$$

Nucleotide substitution matrices with the equivalent distance of 1 PAM

A. Model of uniform mutation rates among nucleotides.

	A	G	T	C
A	0.99			
G	0.00333	0.99		
T	0.00333	0.00333	0.99	
C	0.00333	0.00333	0.00333	0.99

B. Model of 3-fold higher transitions than transversions.

	A	G	T	C
A	0.99			
G	0.006	0.99		
T	0.002	0.002	0.99	
C	0.002	0.002	0.006	0.99

Nucleotide substitution matrices with the equivalent distance of 1 PAM

E.g. take ~'ln' (logarithm base e) of the matrix in the previous slide (for non-diagonal entries):

A. Model of uniform mutation rates among nucleotides.

	A	G	T	C
A	2			
G	-6	2		
T	-6	-6	2	
C	-6	-6	-6	2

B. Model of 3-fold higher transitions than transversions.

	A	G	T	C
A	2			
G	-5	2		
T	-7	-7	2	
C	-7	-7	-5	2

Determining Optimal Alignment

- Two sequences: X and Y
  - $|X| = m$; $|Y| = n$
  - Allowing gaps, $|X| = |Y| = m+n$

- Brute Force
- Dynamic Programming

Brute Force

- Determine all possible subsequences for X and Y
  - 2^{m+n} subsequences for X, 2^{m+n} for Y!
- Alignment comparisons
  - $2^{m+n} * 2^{m+n} = 2^{(2(m+n))} = 4^{m+n}$ comparisons
- Quickly becomes impractical

Dynamic Programming

- Used in Computer Science
- Solve optimization problems by dividing the problem into independent subproblems
- Sequence alignment has optimal substructure property
  - Subproblem: alignment of prefixes of two sequences
  - Each subproblem is computed once and stored in a matrix

Dynamic Programming

- Optimal score: built upon optimal alignment computed to that point

- Aligns two sequences beginning at ends, attempting to align all possible pairs of characters

Dynamic Programming

- Scoring scheme for matches, mismatches, gaps
- Highest set of scores defines optimal alignment between sequences
- Match score: DNA – exact match; Amino Acids – mutation probabilities

- Guaranteed to provide optimal alignment given:
  - Two sequences
  - Scoring scheme

Steps in Dynamic Programming

- Initialization
- Matrix Fill (scoring)
- Traceback (alignment)

DP Example:

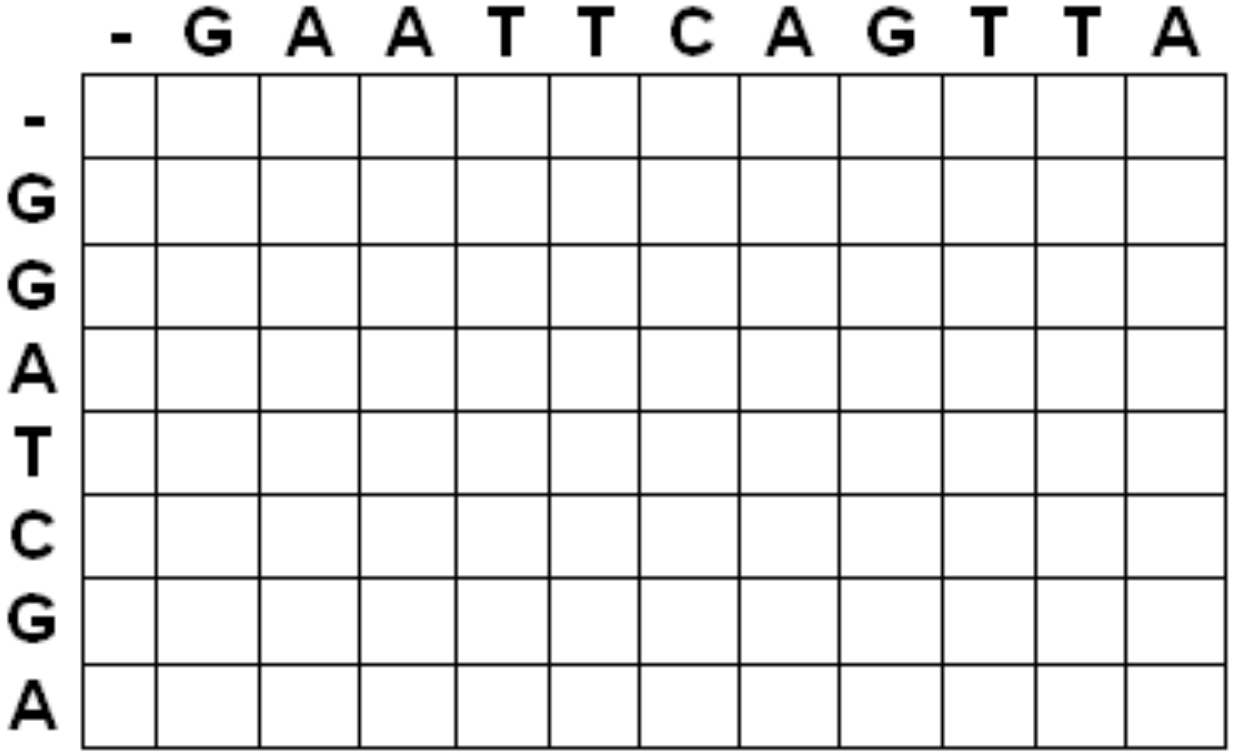
Sequence #1: GAATTCAGTTA; M = 11

Sequence #2: GGATCGA; N = 7

- $s(a_i b_j) = +5$ if $a_i = b_j$ (match score)
- $s(a_i b_j) = -3$ if $a_i \neq b_j$ (mismatch score)
- $w = -4$ (gap penalty)

View of the DP Matrix

- M+1 rows, N+1 columns



Global Alignment (Needleman-Wunsch)

- Attempts to align all residues of two sequences

- ***INITIALIZATION***: First row and first column set
- $S_{i,0} = w * i$
- $S_{0,j} = w * j$

Initialized Matrix(Needleman-Wunsch)

		-	G	A	A	T	T	C	A	G	T	T	A
	0	-4	-8	-12	-16	-20	-24	-28	-32	-36	-40	-44
-	-4											
G	-8											
G	-12											
A	-16											
T	-20											
C	-24											
G	-28											
A												

Matrix Fill (Global Alignment)

$$S_{i,j} = \text{MAXIMUM}[$$

$$S_{i-1,\,j-1} + s(a_i, b_j) \text{ (match/mismatch in the diagonal)},$$

$$S_{i,j-1} + w \text{ (gap in sequence \#1)},$$

$$S_{i-1,j} + w \text{ (gap in sequence \#2)}$$

$$]$$

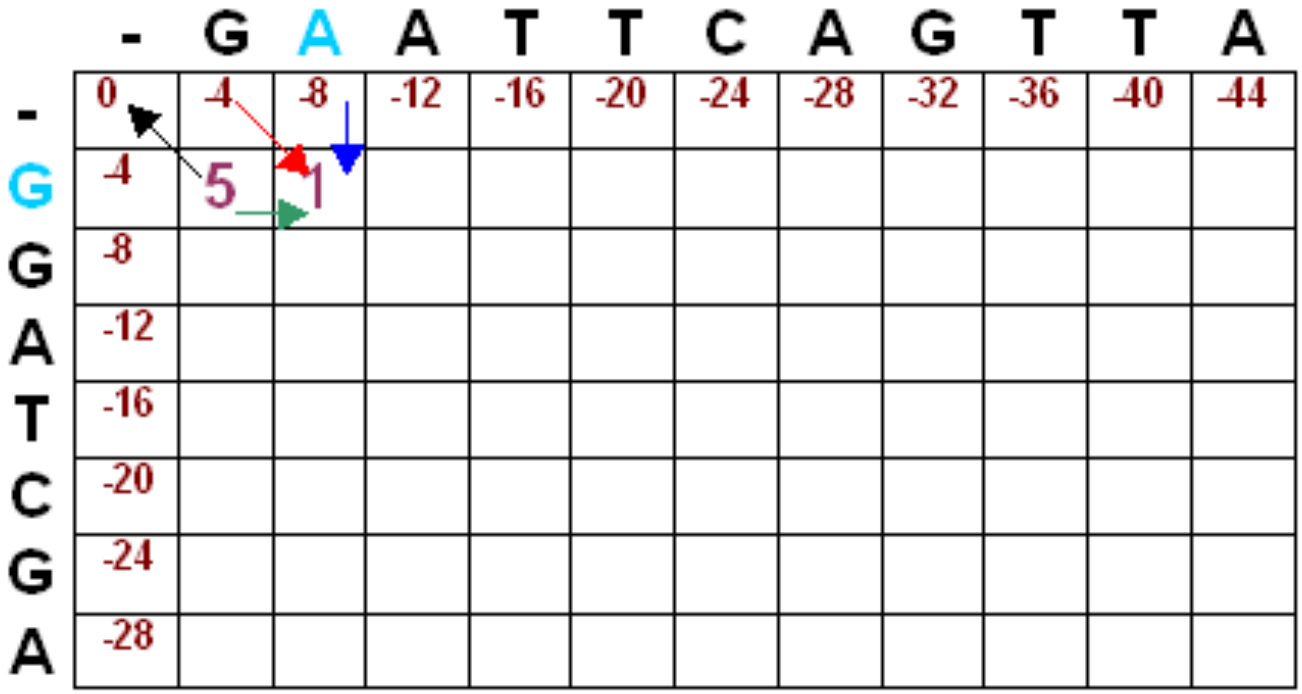
Matrix Fill (Global Alignment)

- $S_{1,1} = MAX[S_{0,0} + 5, S_{1,0} - 4, S_{0,1} - 4] = MAX[5, -8, -8]$

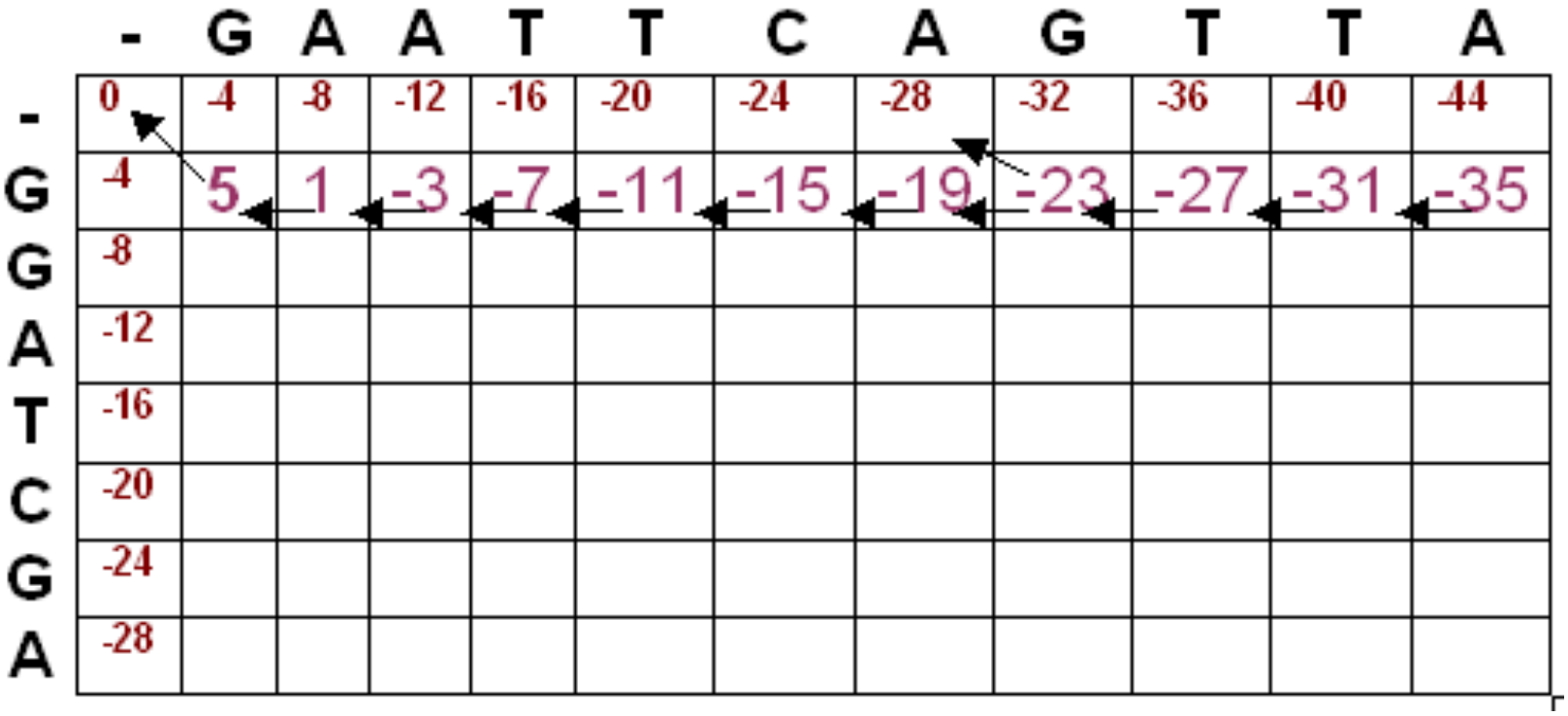
	-	G	A	A	T	T	C	A	G	T	T	A
-	0	-4	-8	-12	-16	-20	-24	-28	-32	-36	-40	-44
G	-4	5										
G	-8											
A	-12											
T	-16											
C	-20											
G	-24											
A	-28											

Matrix Fill (Global Alignment)

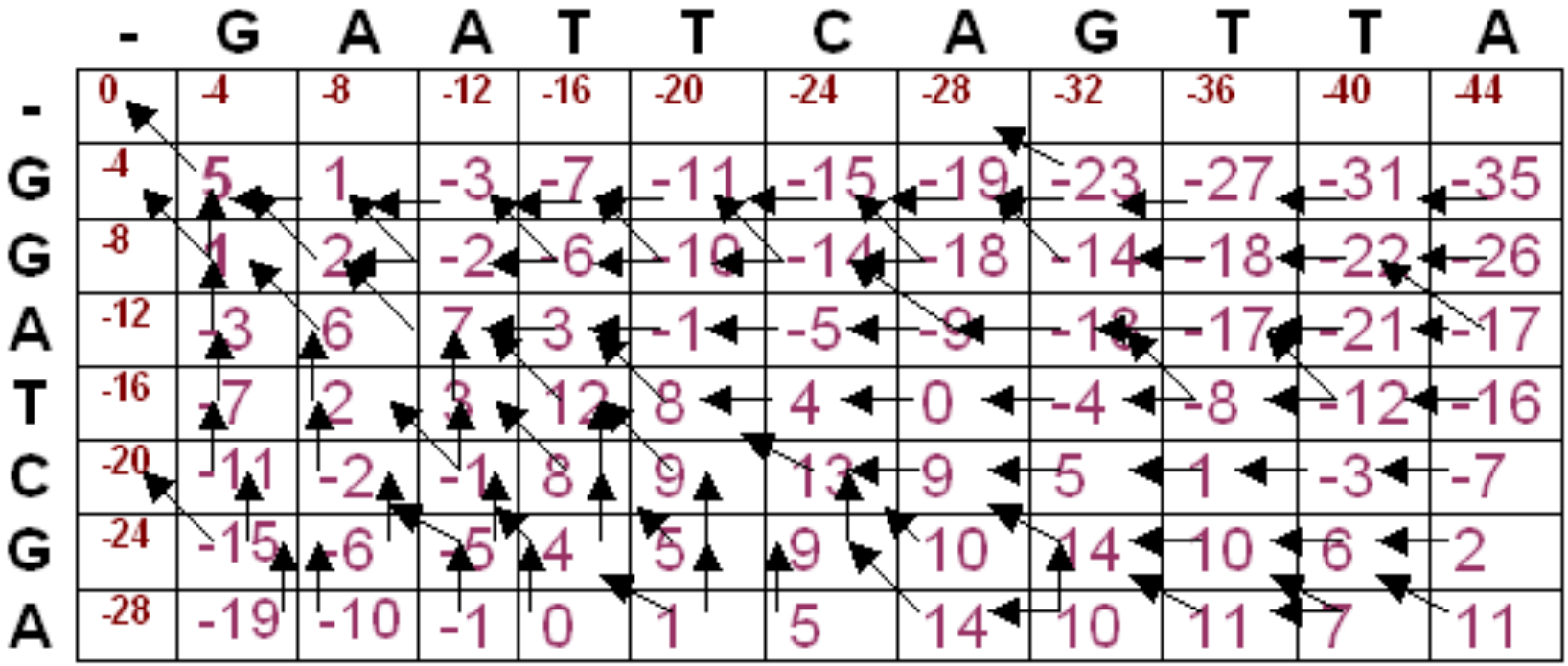
- $S_{1,2} = MAX[S_{0,1} -3, S_{1,1} - 4, S_{0,2} - 4] = MAX[-4 - 3, 5 – 4, -8 – 4] = MAX[-7, 1, -12] = 1$



Matrix Fill (Global Alignment)



Filled Matrix (Global Alignment)



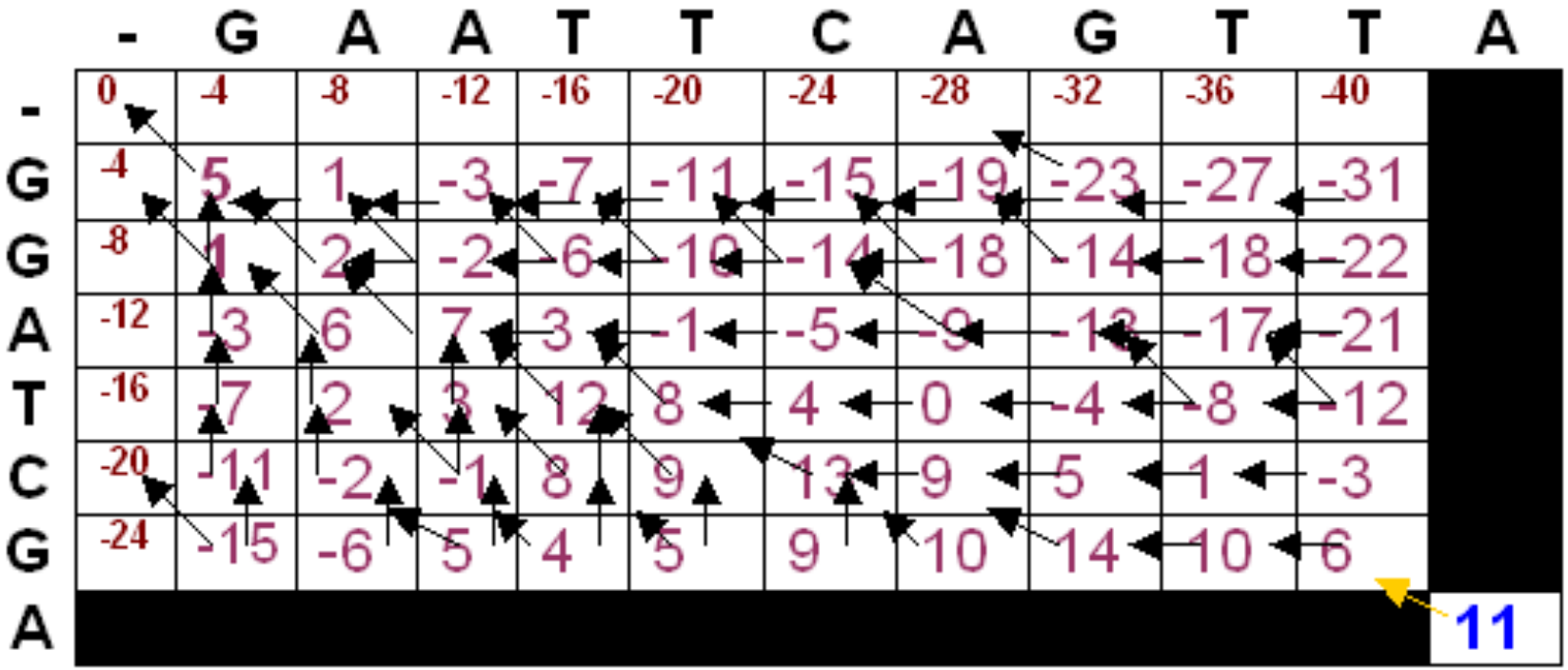
Trace Back (Global Alignment)

- maximum global alignment score = 11 (value in the lower right hand cell).

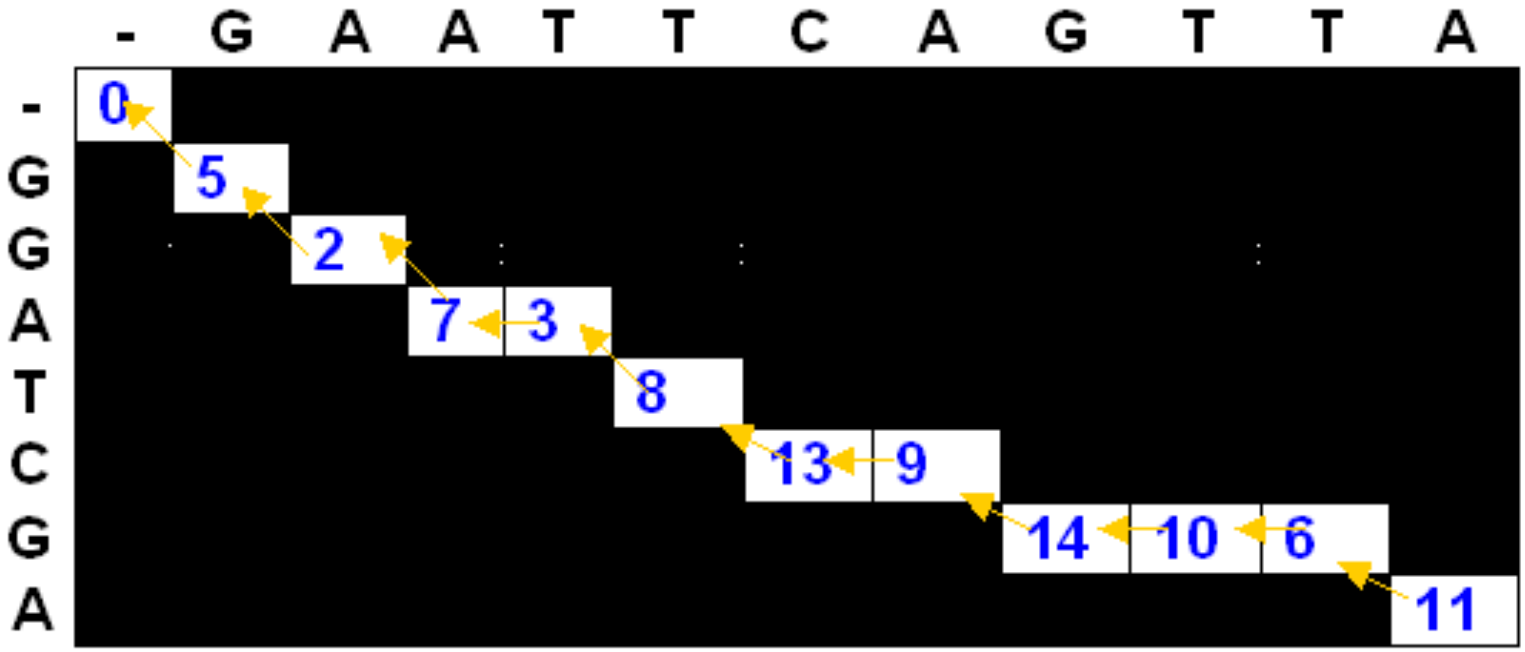
- Traceback begins in position $S_{M,N}$; i.e. the position where both sequences are globally aligned.

- At each cell, we look to see where we move next according to the pointers.

Trace Back (Global Alignment)



Global Trace Back




```
G A A T T C A G T T A
|   |   | |   |     |
G G A - T C - G - - A
```

Checking Alignment Score

```
G A A T T C A G T T A
|   |   | |   |     |
G G A - T C - G - - A
```

```
+ - + - + + - + - - +
5 3 5 4 5 5 4 5 4 4 5
```

$5 - 3 + 5 - 4 + 5 + 5 - 4 + 5 - 4 - 4 + 5 = 11$ ✓

Local Alignment

- Smith-Waterman: obtain highest scoring local match between two sequences

- Requires a modification:
  - When a value in the score matrix becomes negative, reset it to zero (begin of new alignment)

Local Alignment Initialization

- Values in row 0 and column 0 set to 0.

	-	G	A	A	T	T	C	A	G	T	T	A
-	0	0	0	0	0	0	0	0	0	0	0	0
G	0											
G	0											
A	0											
T	0											
C	0											
G	0											
A	0											

Matrix Fill (Local Alignment)

$S_{i,j}$ = MAXIMUM [

$S_{i-1, j-1} + s(a_i, b_j)$ (match/mismatch in the diagonal),

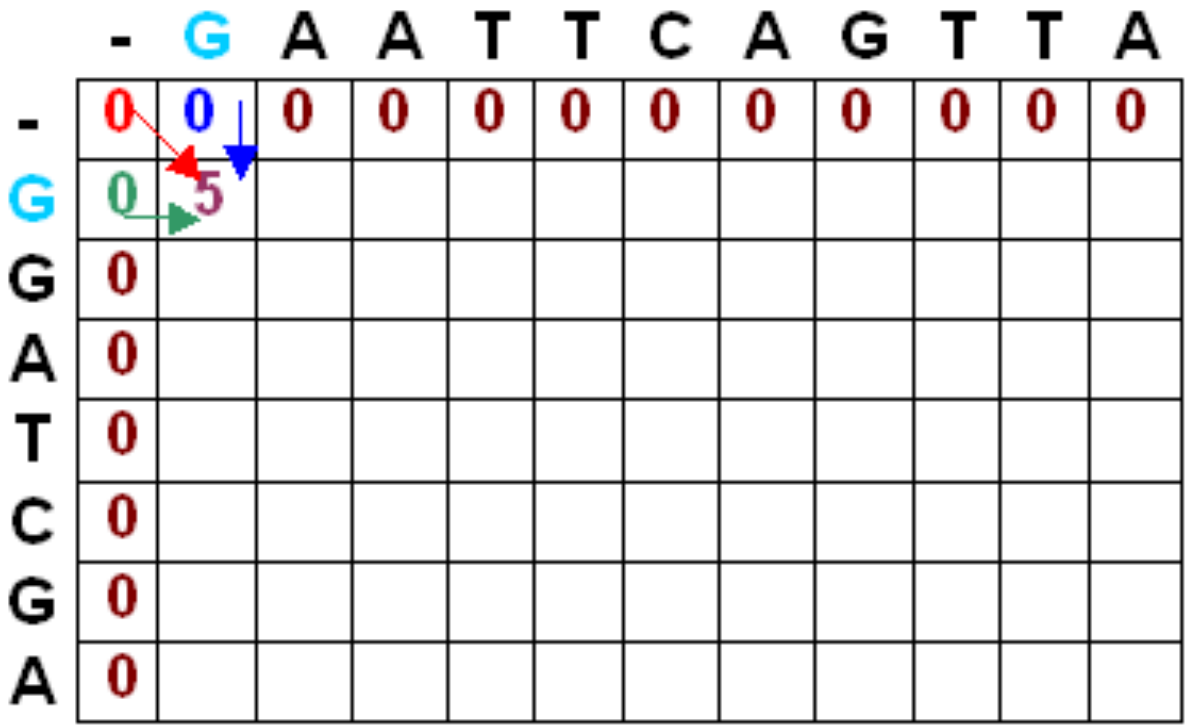
$S_{i,j-1} + w$ (gap in sequence #1),

$S_{i-1,j} + w$ (gap in sequence #2),

0]

Matrix Fill (Local Alignment)

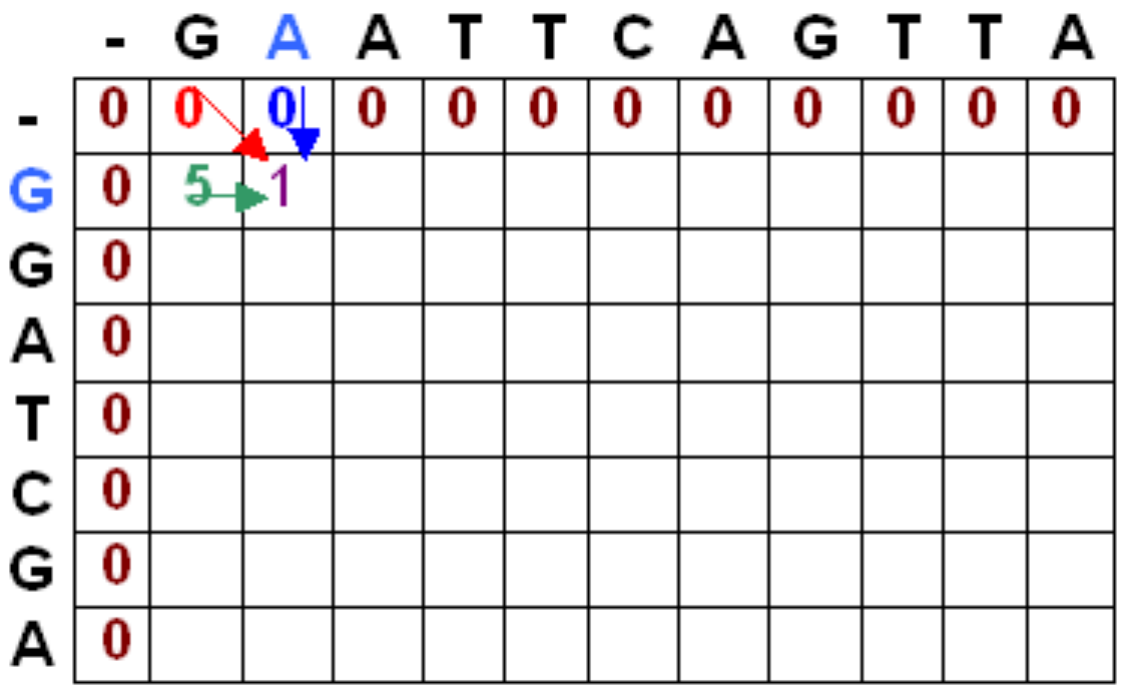
$S_{1,1} = MAX[S_{0,0} + 5, S_{1,0} - 4, S_{0,1} - 4, 0] = MAX[5, -4, -4, 0] = 5$



Matrix Fill (Local Alignment)

$S_{1,2} = MAX[S_{0,1} -3, S_{1,1} - 4, S_{0,2} – 4, 0] = MAX[0 - 3, 5 – 4, 0 – 4, 0] =$

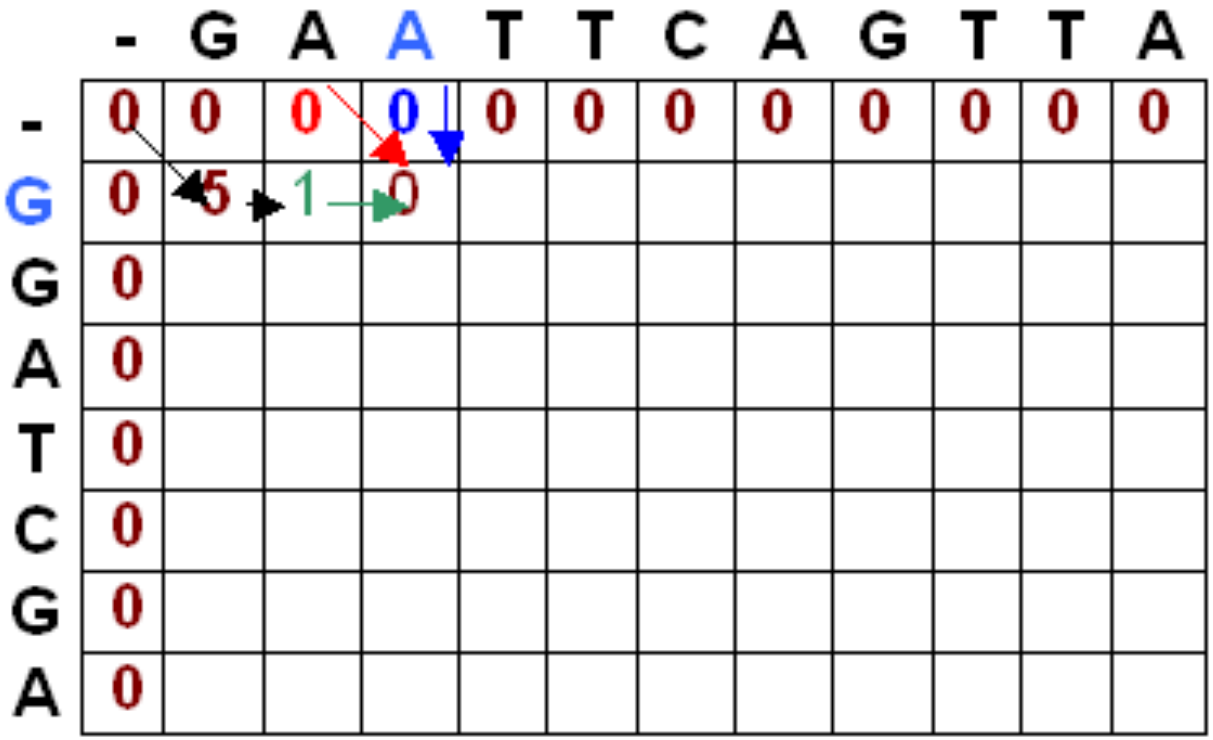
$MAX[-3, 1, -4, 0] = 1$



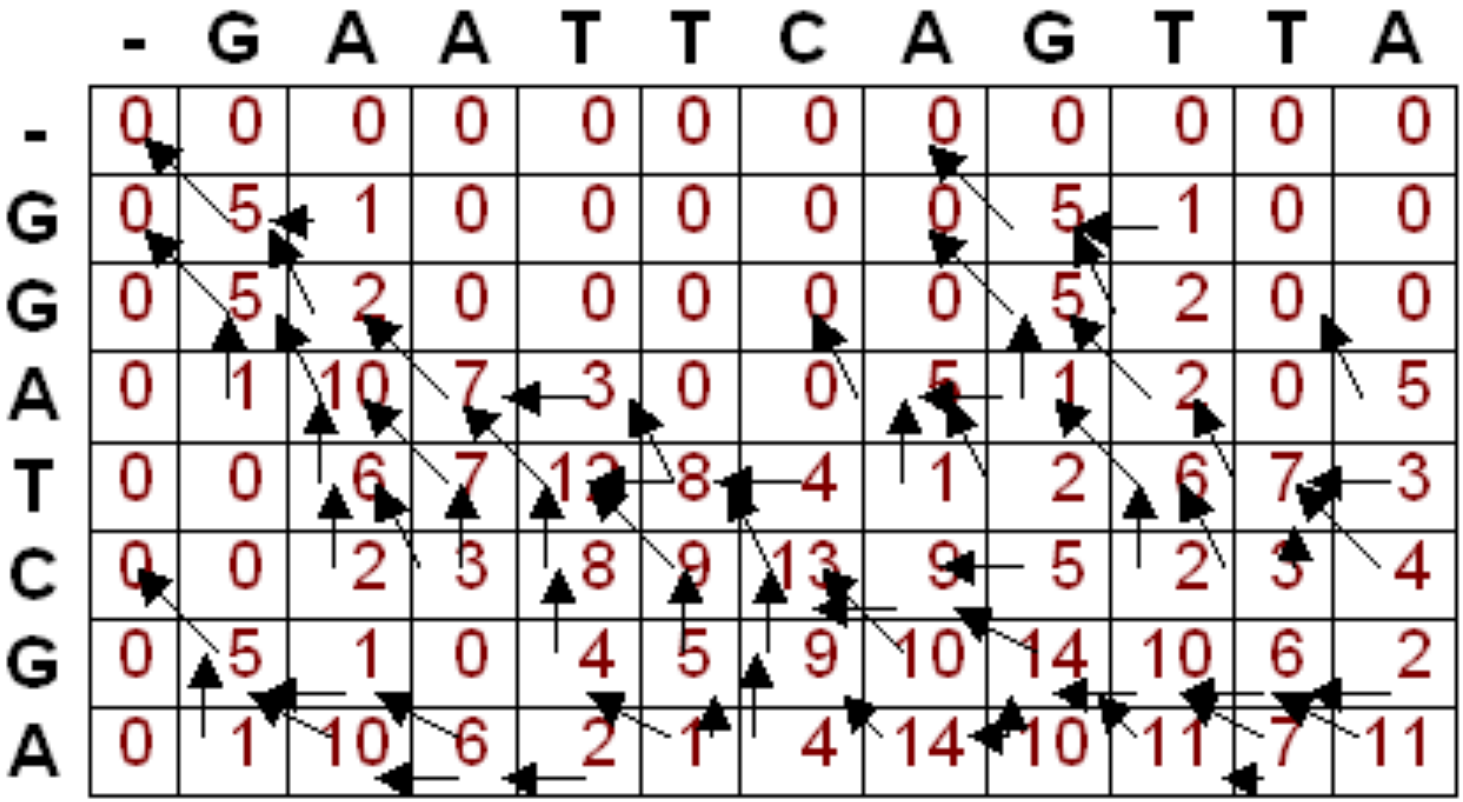
Matrix Fill (Local Alignment)

$S_{1,3} = MAX[S_{0,2} - 3, S_{1,2} - 4, S_{0,3} - 4, 0] = MAX[0 - 3, 1 - 4, 0 - 4, 0] =$

$MAX[-3, -3, -4, 0] = 0$



Filled Matrix (Local Alignment)



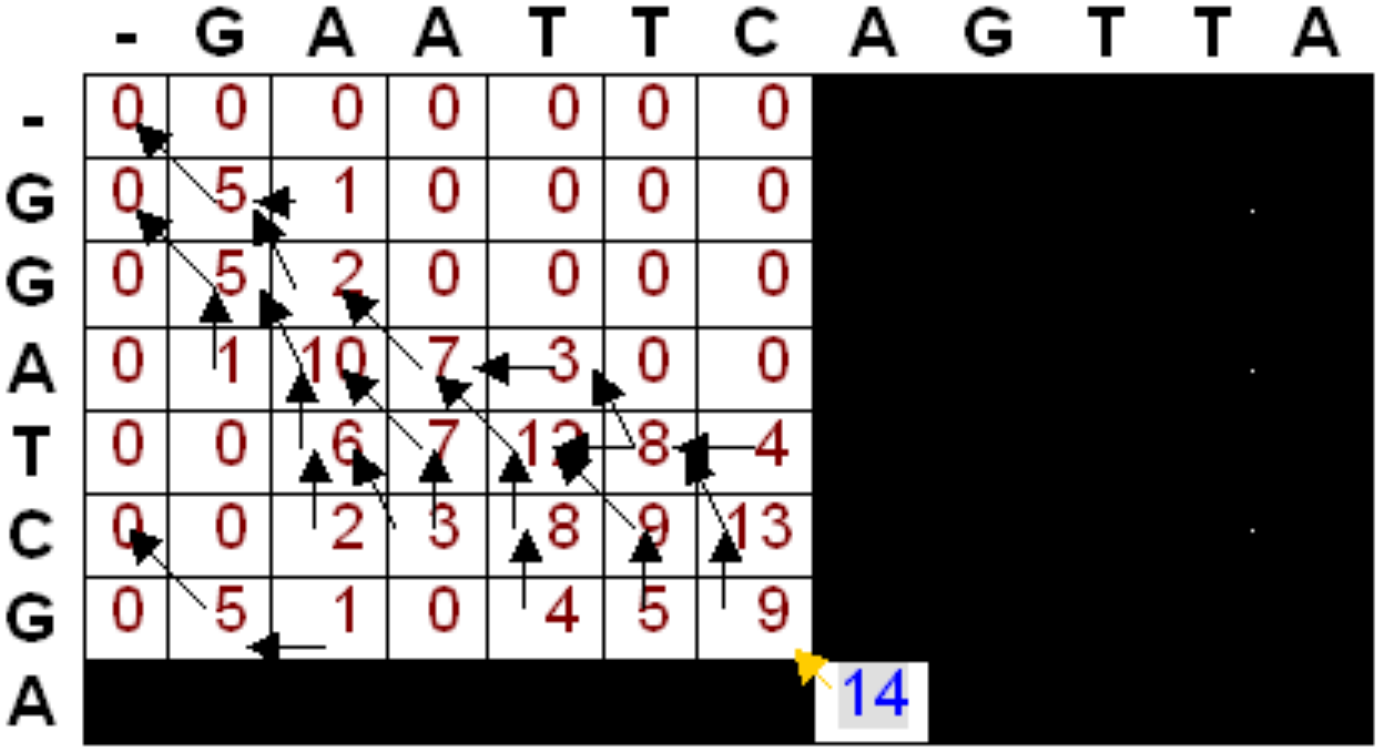
Trace Back (Local Alignment)

- maximum local alignment score for the two sequences is 14
- found by locating the highest values in the score matrix
- 14 is found in two separate cells, indicating multiple alignments producing the maximal alignment score

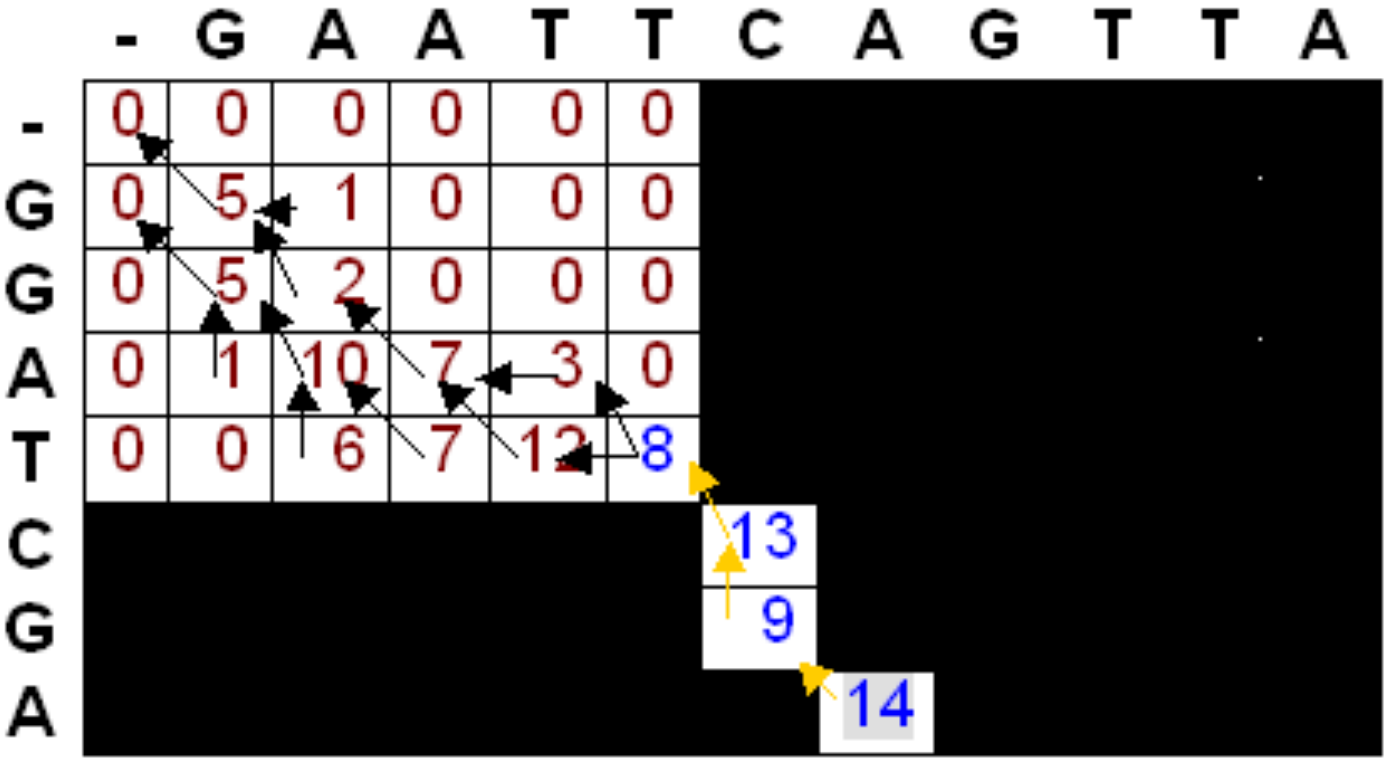
Trace Back (Local Alignment)

- Traceback begins in the position with the highest value.
- At each cell, we look to see where we move next according to the pointers
- When a cell is reached where there is not a pointer to a previous cell, we have reached the beginning of the alignment

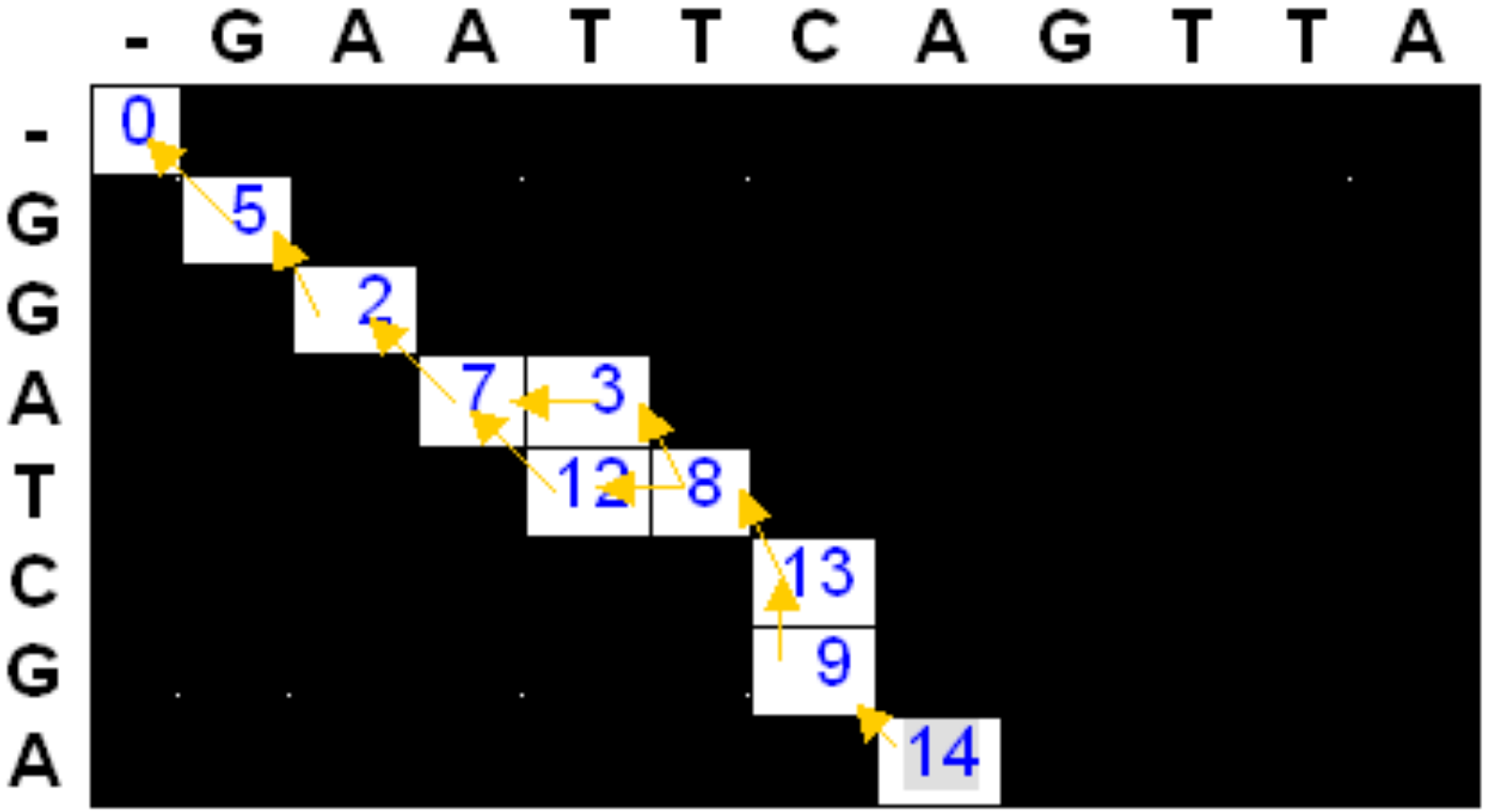
Trace Back (Local Alignment)



Trace Back (Local Alignment)



Trace Back (Local Alignment)



Maximum Local Alignment

```
G A A T T C - A      G A A T T C - A
|   | |   |   |      |   |   | |   |
G G A T - C G A      G G A - T C G A

+ - + + - + - +      + - + - + + - +
5 3 5 5 4 5 4 5      5 3 5 4 5 5 4 5
```

Overlap Alignment

Consider the following problem:

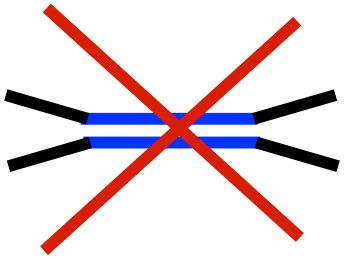
- Find the most significant overlap between two sequences?
- Possible overlap relations:

a. 

b. 

Difference from local alignment:

Here we require alignment between the endpoints of the two sequences.



Overlap Alignment

Initialization: $S_{i,0} = 0$, $S_{0,j} = 0$

Recurrence: as in global alignment

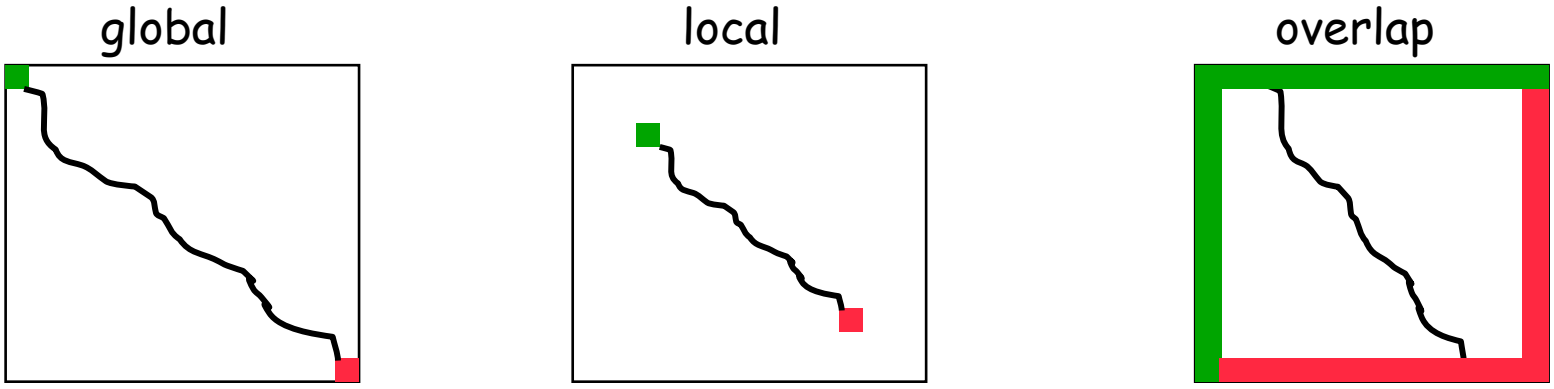
$$S_{i,j} = \text{MAXIMUM} [$$

$$S_{i-1,\,j-1} + s(a_i, b_j) \text{ (match/mismatch in the diagonal)},$$

$$S_{i,j-1} + w \text{ (gap in sequence \#1)},$$

$$S_{i-1,j} + w \text{ (gap in sequence \#2) }]$$

Score: maximum value at the bottom line and rightmost line



Overlap Alignment - example

PAWHEAE
HEAGAWGHEE

Scoring scheme :
Match: +4
Mismatch: -1
Gap penalty: -5

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0										
A	0										
W	0										
H	0										
E	0										
A	0										
E	0										

Overlap Alignment

PAWHEAE
HEAGAWGHEE

Scoring scheme :
Match: +4
Mismatch: -1
Gap penalty: -5

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
A	0	-1									
W	0	-1									
H	0	4									
E	0	-1									
A	0	-1									
E	0	-1									

Overlap Alignment

PAWHEAE
HEAGAWGHEE

Scoring scheme:
Match: +4
Mismatch: -1
Gap penalty: -5

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
A	0	-1	-2	3	-2	3	-2	-2	-2	-2	-2
W	0	-1	-2	-2	2	-2	7	2	-3	-3	-3
H	0	4	-1	-3	-3	1	2	6	6	1	-4
E	0	-1	8	3	-2	-3	0	1	5	10	5
A	0	-1	3	12	7	2	-2	-1	0	5	9
E	0	-1	3	7	11	6	1	-3	-2	4	9

Overlap Alignment

The best overlap is:

P A W H E A E - - - - - -

- - - H E A G A W G H E E

Pay attention!

A different scoring scheme could yield a different result, such as:

Scoring scheme :
Match: +4
Mismatch: -1
Gap penalty: -2

- - - P A W - H E A E

H E A G A W G H E E -

Sequence Alignment Variants

- Global alignment (The Needelman-Wunsch Algorithm)
  - Initialization: $S_{i,0} = i*w$, $S_{0,j} = j*w$
  - Score: $S_{i,j} = \mathbf{MAX}\,[S_{i-1,\,j-1} + s(a_i, b_j),\ S_{i,j-1} + w,\ S_{i-1,j} + w]$
- Local alignment (The Smith-Waterman Algorithm)
  - Initialization: $S_{i,0} = 0$, $S_{0,j} = 0$
  - Score: $S_{i,j} = \mathbf{MAX}\,[S_{i-1,\,j-1} + s(a_i, b_j),\ S_{i,j-1} + w,\ S_{i-1,j} + w,\ 0]$
- Overlap alignment
  - Initialization: $S_{i,0} = 0$, $S_{0,j} = 0$
  - Score: $S_{i,j} = \mathbf{MAX}\,[S_{i-1,\,j-1} + s(a_i, b_j),\ S_{i,j-1} + w,\ S_{i-1,j} + w]$

Linear vs. Affine Gaps

- The scoring matrices used to this point assume a linear gap penalty where each gap is given the same penalty score.
- However, over evolutionary time, it is more likely that a contiguous block of residues has become inserted/deleted in a certain region (for example, it is more likely to have 1 gap of length k than k gaps of length 1).
- Therefore, a better scoring scheme to use is an initial higher penalty for opening a gap, and a smaller penalty for extending the gap.

Linear vs. Affine Gaps

- Gaps have been modeled as linear
- More likely contiguous block of residues inserted or deleted
  - 1 gap of length k rather than k gaps of length 1
- Scoring scheme should penalize new gaps more

Affine Gap Penalty

- $w_x = g + r(x\text{-}1)$

- w_x : total gap penalty; g: gap open penalty; r: gap extend penalty; x: gap length

- gap penalty chosen relative to score matrix
  - Gaps not excluded
  - Gaps not over included
  - Typical Values: g = -12; r = -4

Affine Gap Penalty and Dynamic Programming

$$M_{i,j} = \max \{ D_{i-1,j-1} + \text{subst}(A_i, B_j) ,$$
$$M_{i-1,j-1} + \text{subst}(A_i, B_j) ,$$
$$I_{i-1,j-1} + \text{subst}(A_i, B_j) \}$$

$$D_{i,j} = \max \{ D_{i,j-1} - \text{extend}, \; M_{i,j-1} - \text{open} \}$$

$$I_{i,j} = \max \{ M_{i-1,j} - \text{open}, \; I_{i-1,j} - \text{extend} \}$$

where M is the match matrix, D is delete matrix, and I is insert matrix

Drawbacks to DP Approaches

- Dynamic programming approaches are guaranteed to give the optimal alignment between two sequences given a scoring scheme.
- However, the two main drawbacks to DP approaches is that they are compute and memory intensive, in the cases discussed to this point taking at least $O(n^2)$ space, between $O(n^2)$ and $O(n^3)$ time.
- Linear space algorithms have been used in order to deal with one drawback to dynamic programming. The basic idea is to concentrate only on those areas of the matrix more likely to contain the maximum alignment.
- The most well-known of these linear space algorithms is the Myers-Miller algorithm.

Alternative DP approaches

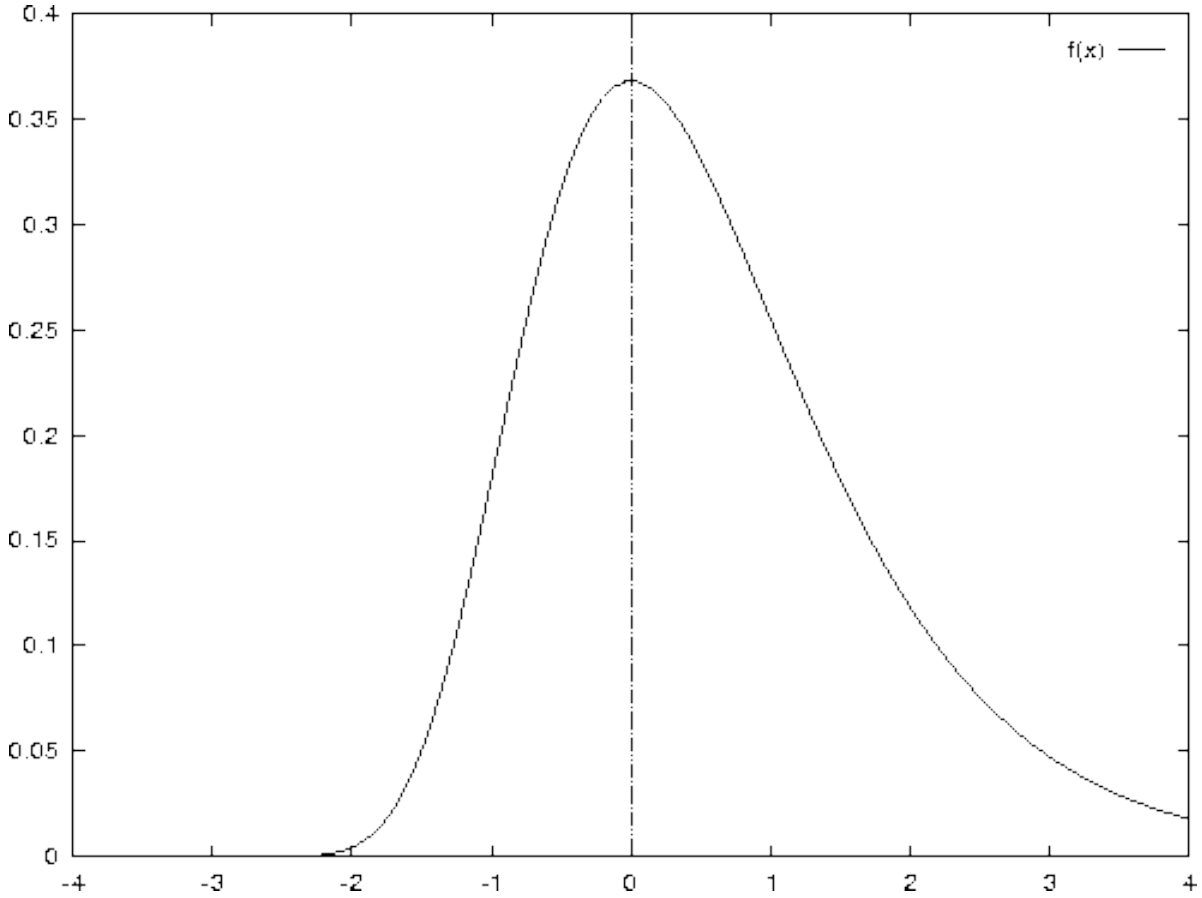
- Linear space algorithms Myers-Miller
- Bounded Dynamic Programming
- Ewan Birney's Dynamite Package
  - Automatic generation of DP code

Assessing Significance of Alignment

- When two sequences of length m and n are not obviously similar but show an alignment, it becomes necessary to assess the significance of the alignment.

- The alignment of scores of random sequences has been shown to follow a Gumbel extreme value distribution.

Gumbel Extreme Value Distribution



- http://roso.epfl.ch/mbi/papers/discretechoice/node11.html
- http://mathworld.wolfram.com/GumbelDistribution.html
- http://en.wikipedia.org/wiki/Generalized_extreme_value_distribution

Probability of Alignment Score

- Using a Gumbel extreme value distribution, the expected number of alignments (E-value) with a score at least S is:

$$E = Kmn\, e^{-\lambda S}$$

  - m,n: Lengths of sequences
  - *K* ,λ: statistical parameters dependent upon scoring system and background residue frequencies

- Recall that the log-odds scoring schemes examined to this point normally use a $S = 10*\log_{10}x$ scoring system.

- We can normalize the raw scores obtained using these non-gapped scoring systems to obtain the amount of bits of information contained in a score (or the amount of nats of information contained within a score).

Converting to Bit Scores

A raw score can be normalized to a bit score using the formula:

$$S' = \frac{\lambda S - \ln K}{\ln 2}$$

- The E-value corresponding to a given bit score can then be calculated as:

$$E = mn\,2^{-S'}$$

- Converting to nats is similar.
- We just substitute e for 2 in the above equations.
- Converting scores to either bits or nats gives a standardized unit by which the scores can be compared.

P-Value

- P values can be calculated as the probability of obtaining a given score at random.

- P-values can be estimated as:

$$P = 1 - e^{-E}$$

A quick determination of significance

- If a scoring matrix has been scaled to bit scores, then it can quickly be determined whether or not an alignment is significant.
- For a typical amino acid scoring matrix, K = 0.1 and lambda depends on the values of the scoring matrix.
- If a PAM or BLOSUM matrix is used, then lambda is precomputed.
- For instance, if the log odds matrix is in units of bits, then lambda = $\log_e 2$, and the significance cutoff can be calculated as $\log_2(mn)$.

Significance of Ungapped Alignments

- PAM matrices are $10 * \log_{10}x$
- Converting to $\log_2 x$ gives bits of information
- Converting to $\log_e x$ gives nats of information

Quick Calculation:

- If bit scoring system is used, significance cutoff is:

$$\log_2(mn)$$

Example

- Suppose we have two sequences, each approximately 250 amino acids long that are aligned using a Smith-Waterman approach.
- Significance cutoff is:

$\log_2(250 * 250) = 16$ bits

Example

- Using PAM250, the following alignment is found:

- F W L E V E G N S M T A P T G
- F W L D V Q G D S M T A P A G

PAM250 matrix

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	12																				C
S	0	2																			S
T	−2	1	3																		T
P	−3	1	0	6																	P
A	−2	1	1	1	2																A
G	−3	1	0	−1	1	5															G
N	−4	1	0	−1	0	0	2														N
D	−5	0	**0**	−1	0	1	2	4													D
E	−5	**0**	**0**	−1	0	0	1	3	4												E
Q	−5	−1	−1	**0**	0	−1	1	2	2	4											Q
H	−3	−1	−1	0	−1	−2	2	1	1	3	6										H
R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	6									R
K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5								K
M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6							M
I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5						I
L	−6	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	6					L
V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4				V
F	−4	−3	−3	−5	−4	−5	−4	−6	−5	−5	−2	−4	−5	0	1	2	−1	9			F
Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10		Y
W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Example

- Using PAM250, the score is calculated:

- F W L E V E G N S M T A P T G
- F W L D V Q G D S M T A P A G

- S = 9 + 17 + 6 + 3 + 4 + 2 + 5 + 2 + 2 + 6 + 3 + 2 + 6 + 1 + 5 = 73

Significance Example

- S is in 10 * $\log_{10}x$, so this should be converted to a bit score:

- $S = 10 \log_{10}x$
- $S/10 = \log_{10}x$
- $S/10 = \log_{10}x * (\log_2 10/\log_2 10)$
- $S/10 * \log_2 10 = \log_{10}x / \log_2 10$
- $S/10 * \log_2 10 = \log_2 x$
- $1/3\ S \sim \log_2 x$

- $S' \sim 1/3S$

Significance Example

- S’ = 1/3S = 1/3 * 73 = 24.333 bits

- The significance cutoff is:

  log2(mn) = log2(250 * 250) = 16 bits

- Since the alignment score is above the significance cutoff, this is a significant local alignment.

Estimation of E and P

- When a PAM250 scoring matrix is being used, K is estimated to be 0.09, while lambda is estimated to be 0.229.
- For PAM250, $K = 0.09$; $\lambda = 0.229$

- We can convert the score to a bit score as follows:
- $S' = \lambda S - \ln Kmn$
- $S' = 0.229 * 73 - \ln 0.09 * 250 * 250$
- $S' = 16.72 - 8.63 = 8.09$ bits
- $P(S' >= x) = 1 - e^{(-e^{-x})}$
- $P(S' >= 8.09) = 1 - e^{(-e^{-8.09})} = 3.1 * 10^{-4}$

- Therefore, we see that the probability of observing an alignment with a bit score greater than 8.09 is about 3 in 1000.

Significance of Gapped Alignments

- Gapped alignments make use of the same statistics as ungapped alignments in determining the statistical significance.

- However, in gapped alignments, the values for λ and K cannot be easily estimated.

- Empirical estimations and gap scores have been determined by looking at the alignments of randomized sequences.