In the Name of God, the Merciful, the Compassionate

Introduction to Bioinformatics
04: Pairwise Sequence Alignment

Instructor: Hossein Zeinali

Amirkabir University of Technology

Introduction

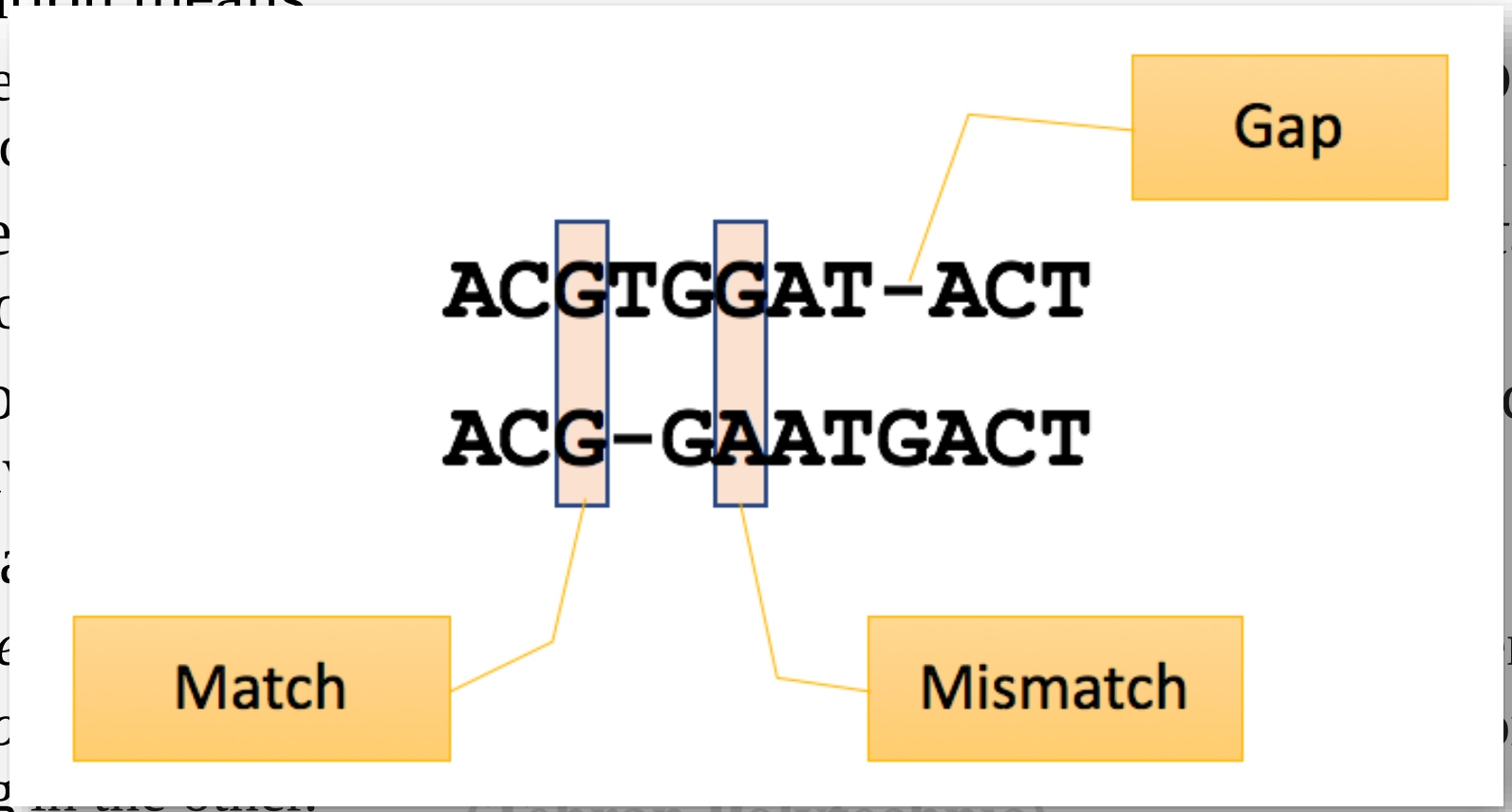
- **Sequence comparison** is an important first step toward *structural and functional analysis* of newly *determined sequences*.
  - Draw *functional and evolutionary inference* of a new protein regard to proteins already existing in the database.
- Sequence alignment is the *most fundamental* process in this type of comparison.
- *Pairwise sequence alignment* is the process of aligning two sequences and is the basis of *database similarity searching* and *multiple sequence alignment*.
- By comparing sequences through alignment, patterns of *conservation and variation* can be identified.
  - Conservation parts perform key functional and structural roles.

Introduction (Cont.)

- Conservation means:
  - Changes at a specific position of an amino acid or (less commonly, DNA) sequence that preserve the physicochemical properties of the original residue.
- Sequence alignment can be used as *basis* for prediction of *structure and function* of *uncharacterized sequences*.
- If the two sequences share *significant similarity*, the two sequences must have derived from a *common evolutionary origin*.
- By generating a correct sequence alignment:
  - *Residue substitutions (mismatch)*: regions that are aligned but not identical.
  - *Insertions or deletions*: regions where residues from one sequence correspond to nothing in the other.

Introduction (Cont.)

- Conservation means:
  - Change
    sequenc
- Sequence
  function c
- If the two
  have deri
- By genera
  - *Residue*
  - *Insertio*
    nothing



Goal of Sequence Alignment

- Find the best pairing of 2 sequences, such that there is maximum correspondence between residues

- **DNA** 4 letter alphabet (+ gap)

TTGACAC
TTTACAC

- **Proteins** 20 letter alphabet (+ gap)

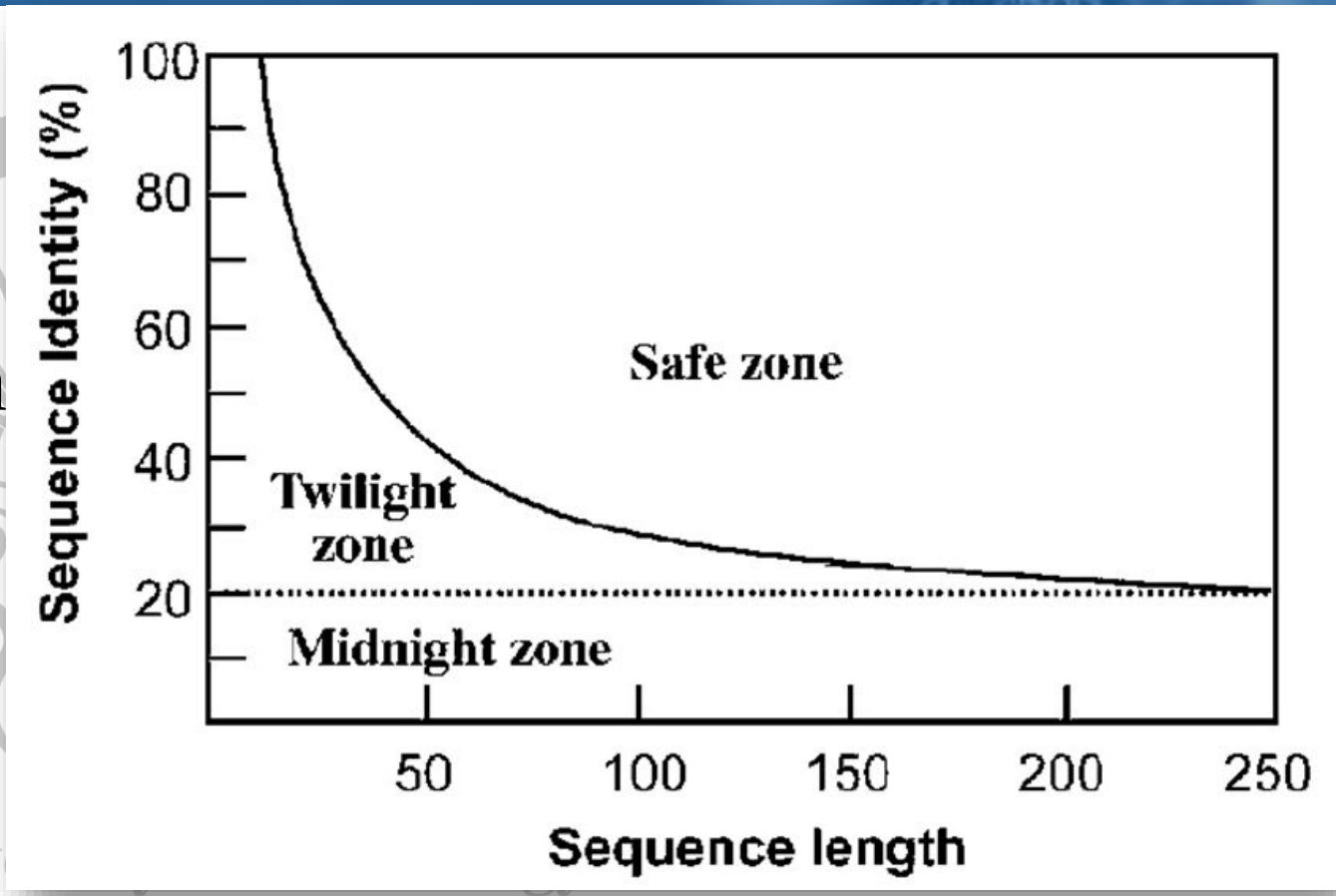
RKVA-GMA
RKIAVAMA

Sequence Homology vs Sequence Similarity

- **Homologous relationship**: When two sequences are descended from a *common evolutionary origin*.
- **Sequence similarity**: is the *percentage of aligned* residues that are similar in physiochemical properties such as *size*, *charge*, *and hydrophobicity*.
- Sequence similarity *can be quantified* (Quantitative value) using percentages; homology is a qualitative statement.
  - Two sequences share may 40% similarity.
  - They are either homologous or nonhomologous.
    - It is incorrect to say that the two sequences share 40% homology.
- If the sequence similarity level is high enough, a common evolutionary relationship can be inferred.

Zones of Protein Sequence Alignments

- **Safe zone**: inferring homologous relationships with high probability.
- **Twilight zone**: determination of homologous relationships becomes less certain.
- **Midnight zone**: homologous relationships cannot be reliably determined.



Sequence Similarity vs Sequence Identity

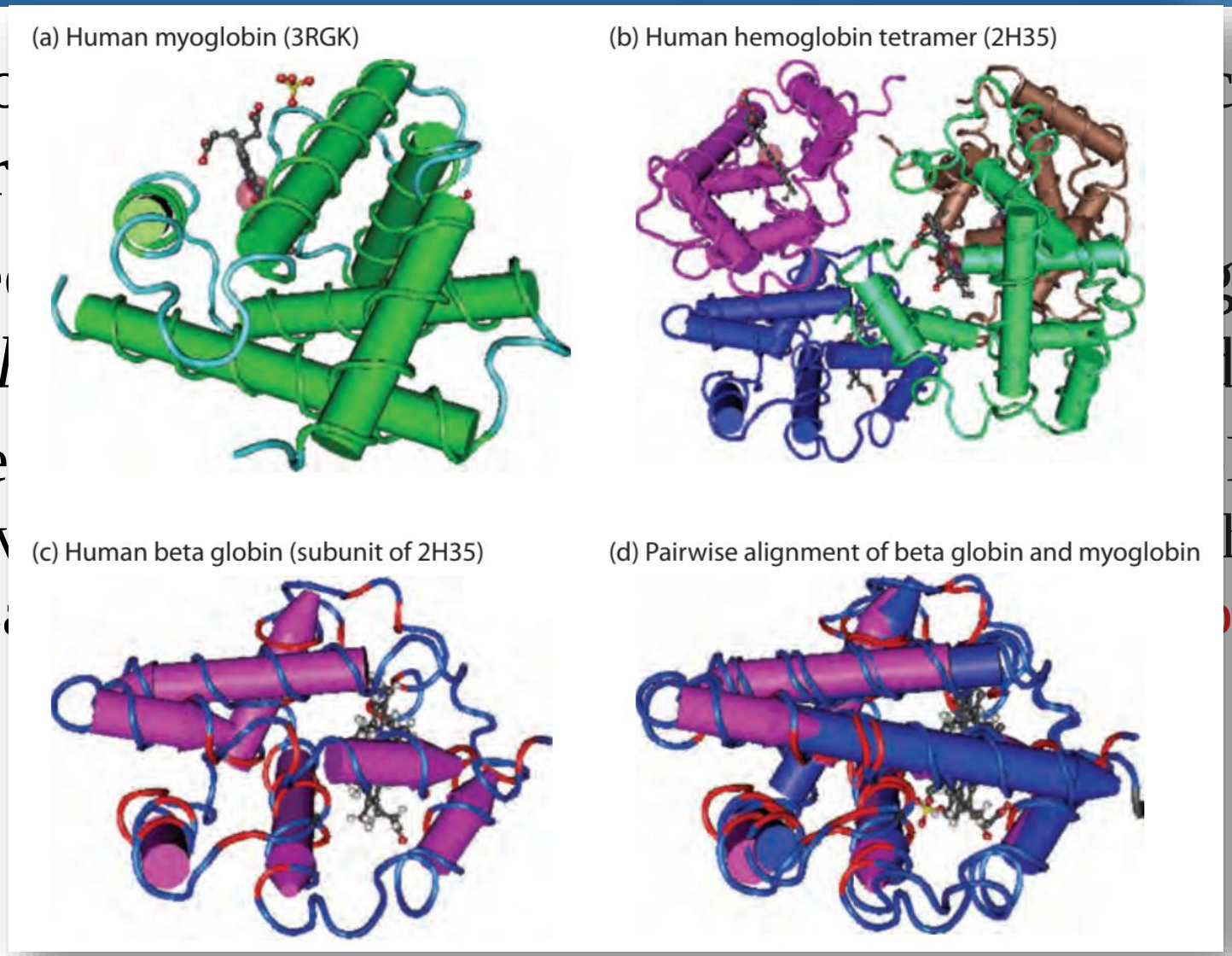
- Sequence *similarity* and sequence *identity* are synonymous for *nucleotide* sequences.
- For protein sequences, however, the two concepts are very different. In a protein sequence alignment:
  - *Sequence identity*: the percentage of matches of the same amino acid residues between two aligned sequences.
  - *Similarity*: the percentage of aligned residues that have similar physicochemical characteristics and can be more readily substituted for each other.

Sequence Homology vs Sequence Similarity 2

- Homologous proteins almost always share a significantly related three-dimensional structure.
- Two molecules may be homologous without sharing *statistically significant* amino acid (or nucleotide) identity.
  - Example: In the globin family all the members are homologous but some have sequences that have diverged so greatly that they share no recognizable sequence identity (e.g. beta globin and myoglobin).

Sequence Homology vs Sequence Similarity 2

- Homologo
  related thr
- Two mole
  statisticall
  - Example
    some hav
    recogniz



cantly

g

lentity.

logous but

hey share no

oglobin).

Ways to Calculate the Similarity or Identity

- There are two ways to calculate the sequence *similarity* (S) or *identity* (I):
  - Involves the use of the overall sequence lengths of both sequences for normalization:

$$S = \frac{L_s \times 2}{L_a + L_b} \times 100 \qquad I = \frac{L_i \times 2}{L_a + L_b} \times 100$$

where L_s is the number of aligned residues with similar characteristics, L_a and L_b are the total lengths of each individual sequence and L_i is the number of aligned identical residues.

  - Or normalizes by the size of the shorter sequence:

$$S = \frac{L_s}{\min(L_a, L_b)} \times 100 \qquad I = \frac{L_i}{\min(L_a, L_b)} \times 100$$

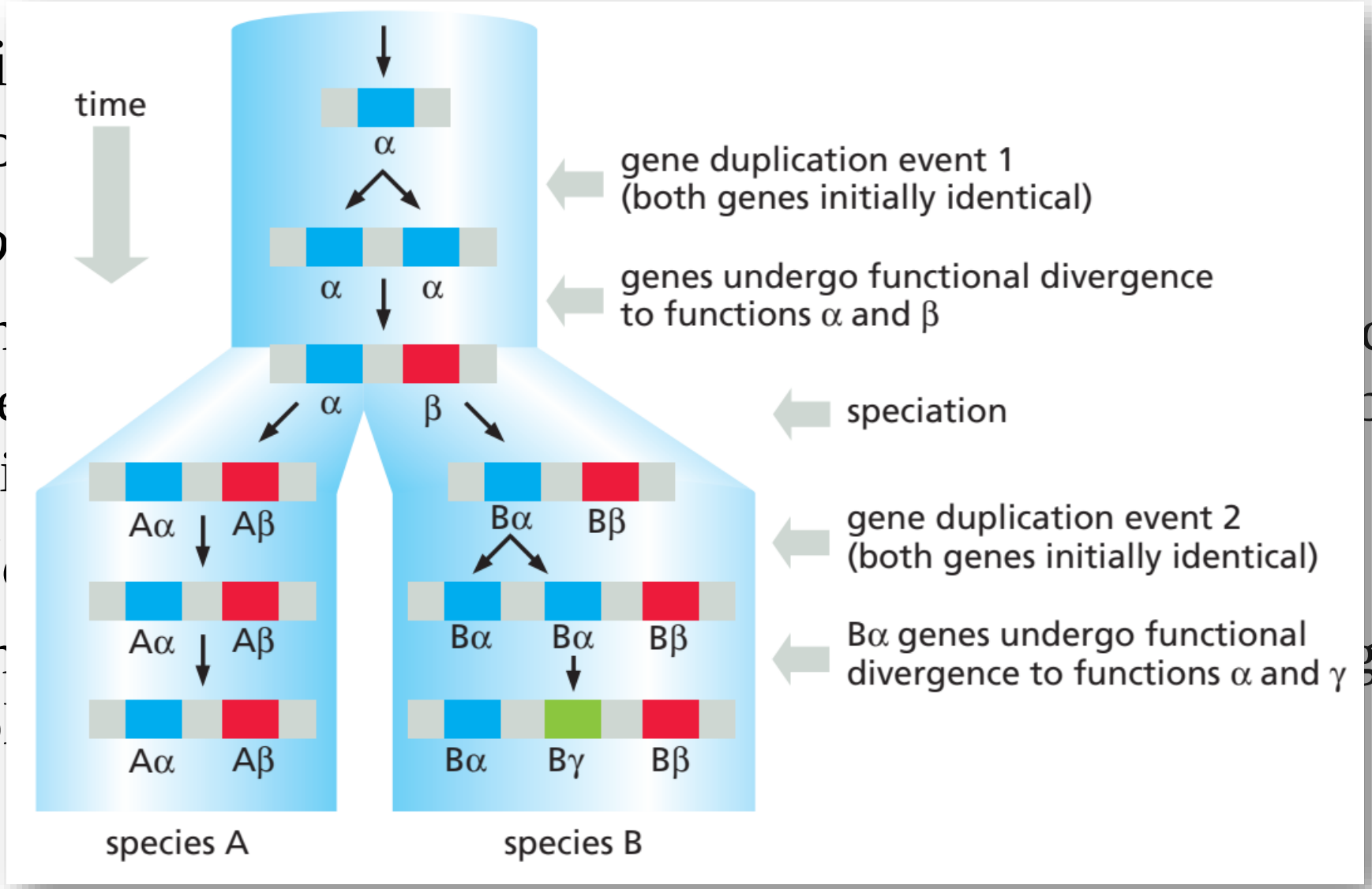
Two Types of Homology

- Proteins that are homologous may be orthologous or paralogous.
- *Orthologs*:
  - Homologous sequences in different species that arose from a common ancestral gene during speciation; may or may not be responsible for a similar function.
- *Paralogs*:
  - Homologous sequences within a single species that arose by gene duplication.

Two Types of Homology

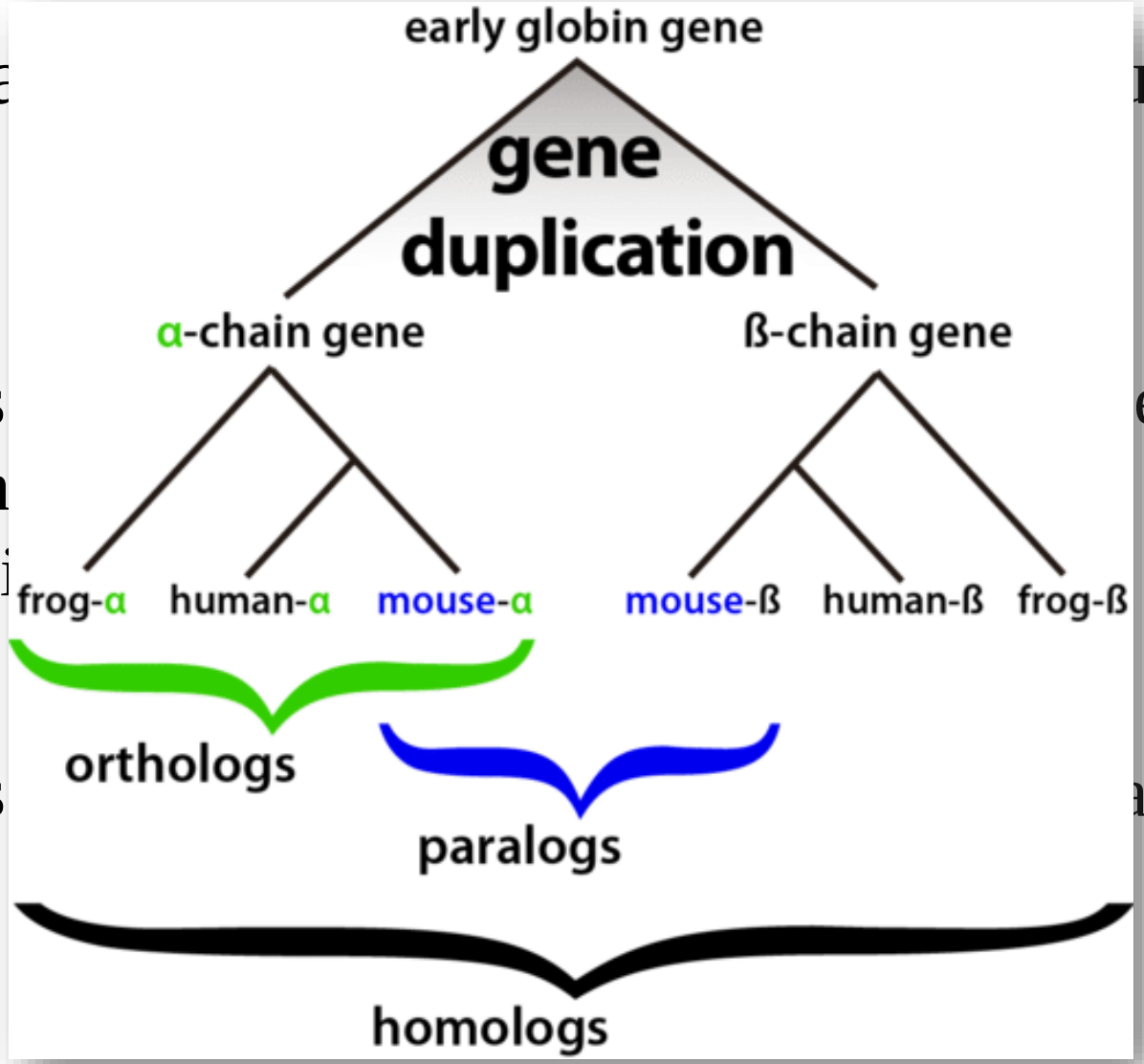
- Protei
  paralc
- *Ortho*
  - Hor
    ance
    simi
- *Paral*
  - Hor
    dup

common
ble for a
gene



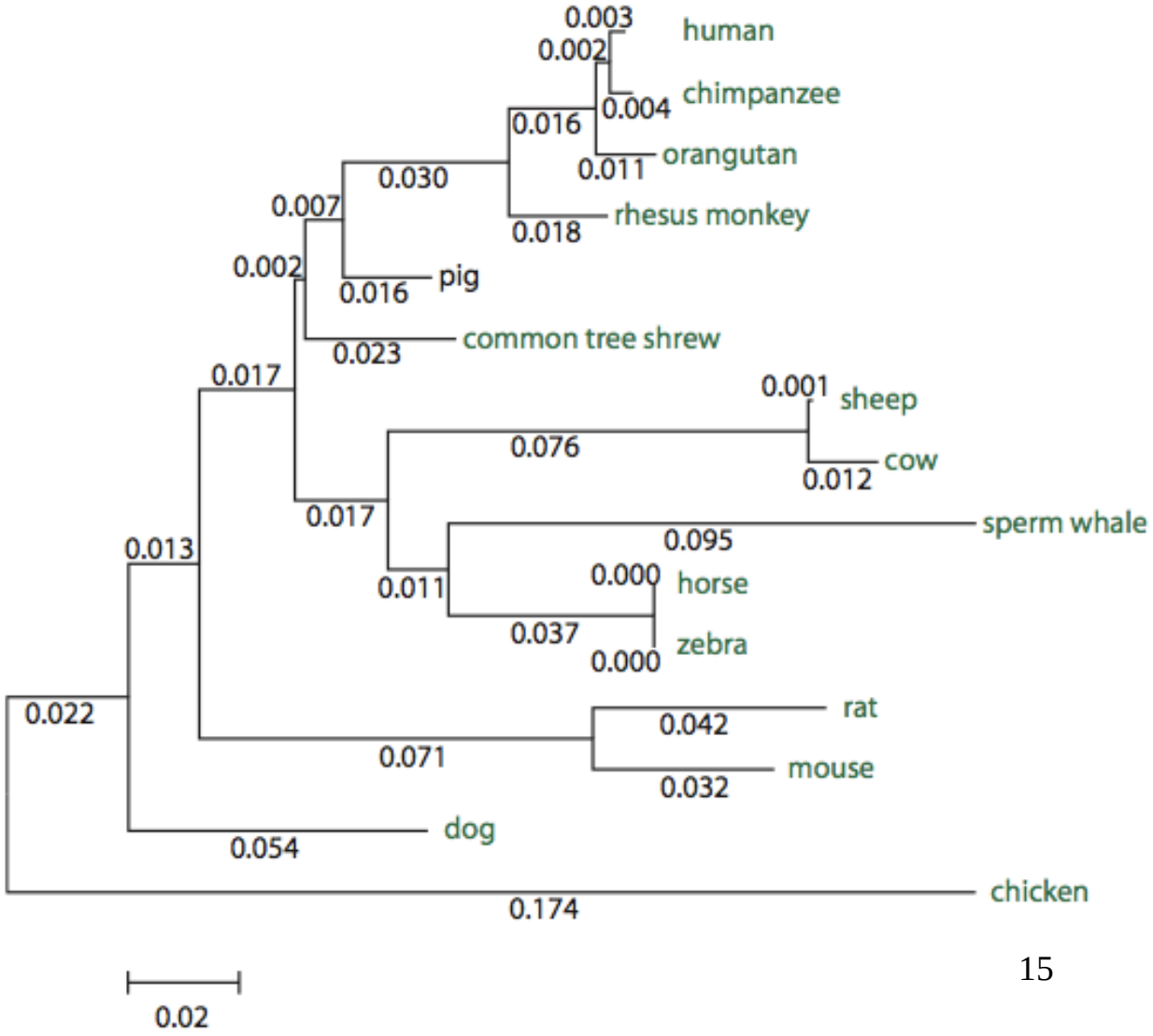
Two Types of Homology

- Proteins that a … us or paralogous.
- *Orthologs*:
  - Homologous … e from a common ancestral gen … responsible for a similar functi…
- *Paralogs*:
  - Homologous … arose by gene duplication.



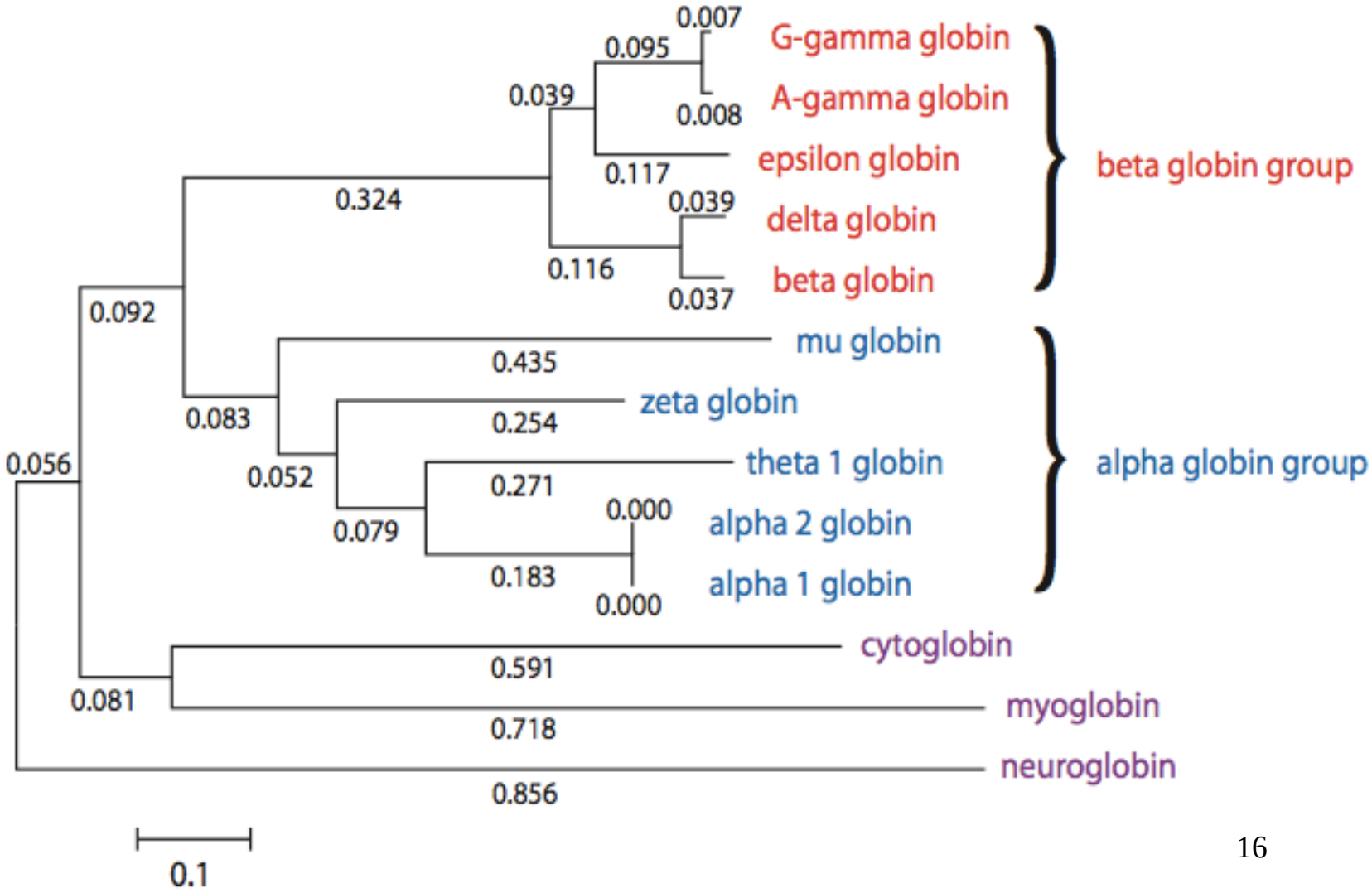
Myoglobin Proteins: Examples of Orthologs

- A group of myoglobin orthologs, visualized by multiply aligning the sequences.
- Sequences that are more closely related to each other are grouped closer together.



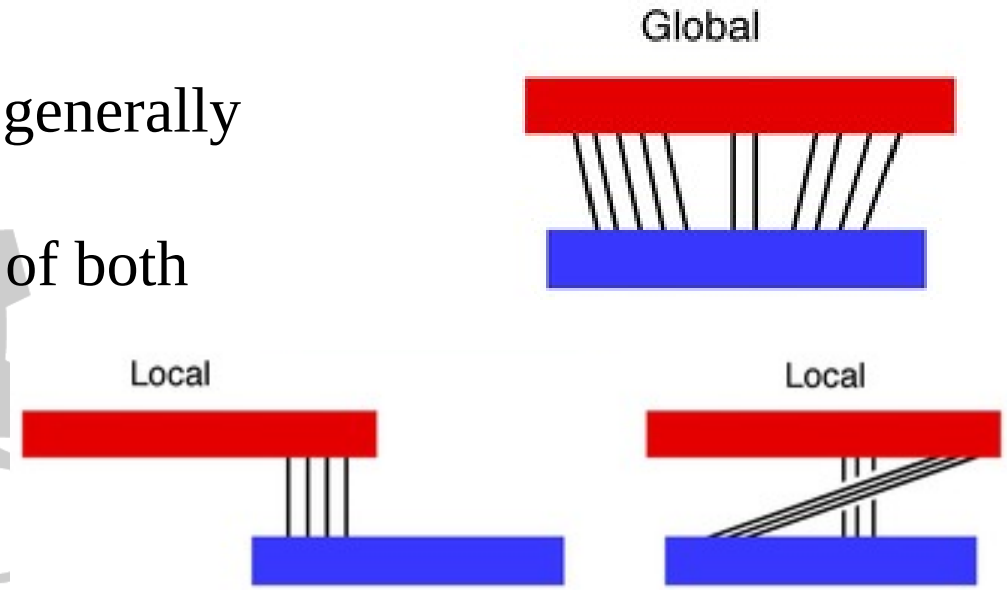
Human Globin Paralogs

- Each of these proteins is human, and each is a member of the globin family.



Global Alignment and Local Alignment

- ***Global Alignment***:
  - Two sequences to be aligned are assumed to be generally similar over their entire length.
  - Alignment is carried out from beginning to end of both sequences.
  - Is more applicable for aligning two closely related sequences of roughly the same length.

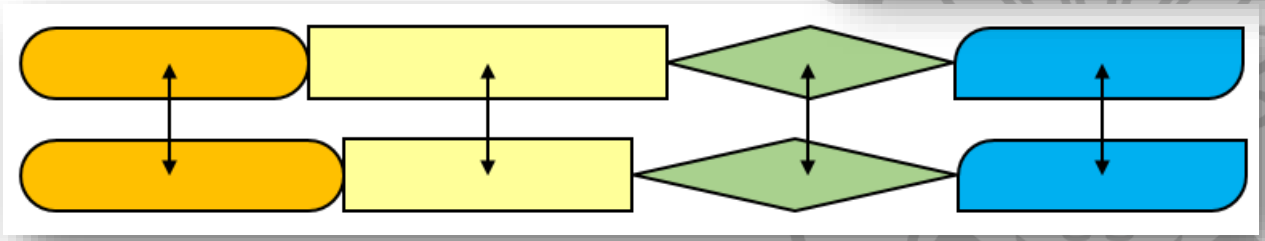


- ***Local Alignment***:
  - Does not assume that the two sequences have similarity over the entire length.
  - It only finds local regions with the highest level of similarity.
  - Is more appropriate for aligning divergent biological sequences containing only modules (*domains* or *motifs*) that are similar.

Global Alignment and Local Alignment

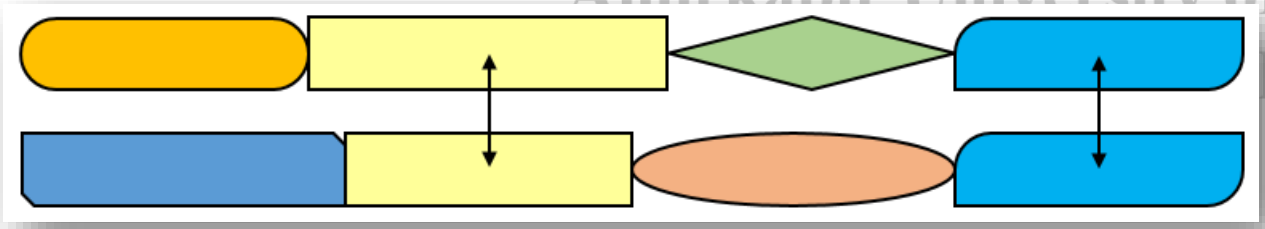
- **Global Alignment**:

```
seq1   EARDF-NQYYSSIKRSGSIQ
          . :  .:::::::::.  .  .
seq2   LPKLFIDQYYSSIKRTMG-H
```



- **Local Alignment**:

```
seq1   NQYYSSIKRS
          .:::::::::.
seq2   DQYYSSIKRT
```



Global vs Local Alignment – Example 1

1 = **CTGTCGCTGCACG**

2 = **TGCCGTG**

Global alignment

```
CTGTCGCTGCACG
-TG-C-C-G--TG
```

```
CTGTCGCTGCACG
-TGCCG-T----G
```

Local alignment

```
CTGTCGCTGCACG
-TGCCG-TG----
```

Which one is better?

Global vs Local Alignment – Example 2

1 = **FTFTALILLLAVAV**

2 = **FTALLLAAV**

Global

```
FTFTALILLLAVAV
F--TAL-LLA-AV
```

Local

```
FTFTALILL-AVAV
--FTAL-LLAAV--
```

Global vs Local Alignment – Example 3

1 = **ATCTGATG**

2 = **TGCATAC**

Global

```
AT-C-TGATG
-TGCAT-A-C
```

4 matches

1 mismatch

2 insertions

2 deletions

Global vs Local Alignment

Which should we use for?

1. Searching for conserved motifs in DNA or protein sequences? Local
2. Aligning two closely related sequences with similar lengths? Global
3. Aligning highly divergent sequences? Local (at least initially)
4. Generating an extended alignment of closely related sequences? Global
5. Generating an extended alignment of closely related sequences with very different lengths? We'll work on that

Global Alignment: Scoring

Reward for matches: α

Mismatch penalty: β

Space/gap penalty: γ

Score = $\alpha w + \beta x + \gamma y$

w = #matches x = #mismatches y = #spaces

Example:

Reward for matches: 8

Mismatch penalty: -2

Space/gap penalty: -5

```
CTGTCG-CTGC

-TGC-CG-TG-
```

4*8 + 2*-2 + 5*-5=+3

Alignment Algorithms

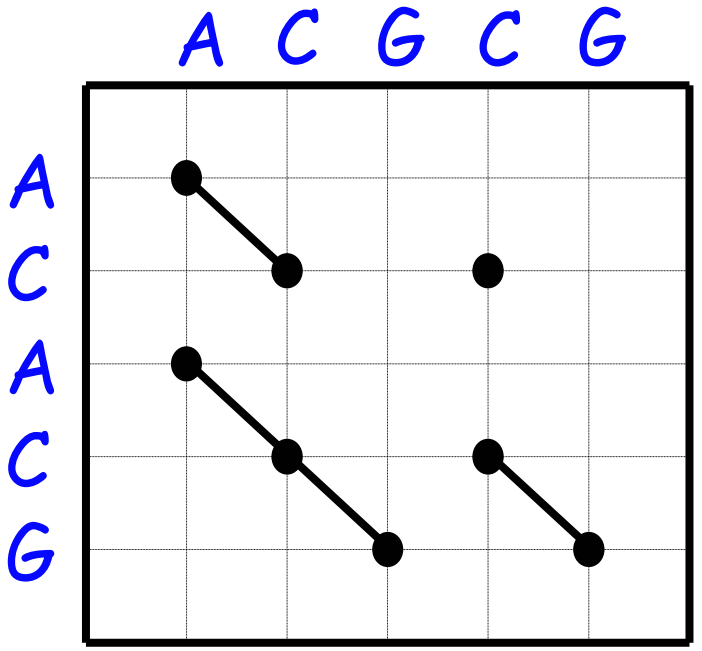
- Alignment algorithms, both global and local, are *fundamentally similar* and only differ in the *optimization strategy* used in aligning similar residues.
- Both types can be based on one of the following three methods:
  - Dot matrix method
  - Dynamic programming method
  - Word method

Dot Matrix Method (Dot Plots)

- The most basic sequence alignment method, also known as the *dot plot method.*
- It is a graphical way of comparing two sequences.
- A problem exists when comparing *large sequences* using the dot matrix method, namely, *the high noise level*.
  - Mostly happened for DNA sequences
- The method is also restricted to pairwise alignment. It is difficult for the method to scale up to multiple alignment.

Dot Matrix Method (Dot Plots)

- Place 1 sequence along top row of matrix
- Place 2nd sequence along left column of matrix
- Plot a dot each time there is a match between an element of *row sequence* and an element of *column sequence*
  - For proteins, usually use more sophisticated scoring schemes than "identical match"
- *Diagonal lines* indicate areas of match



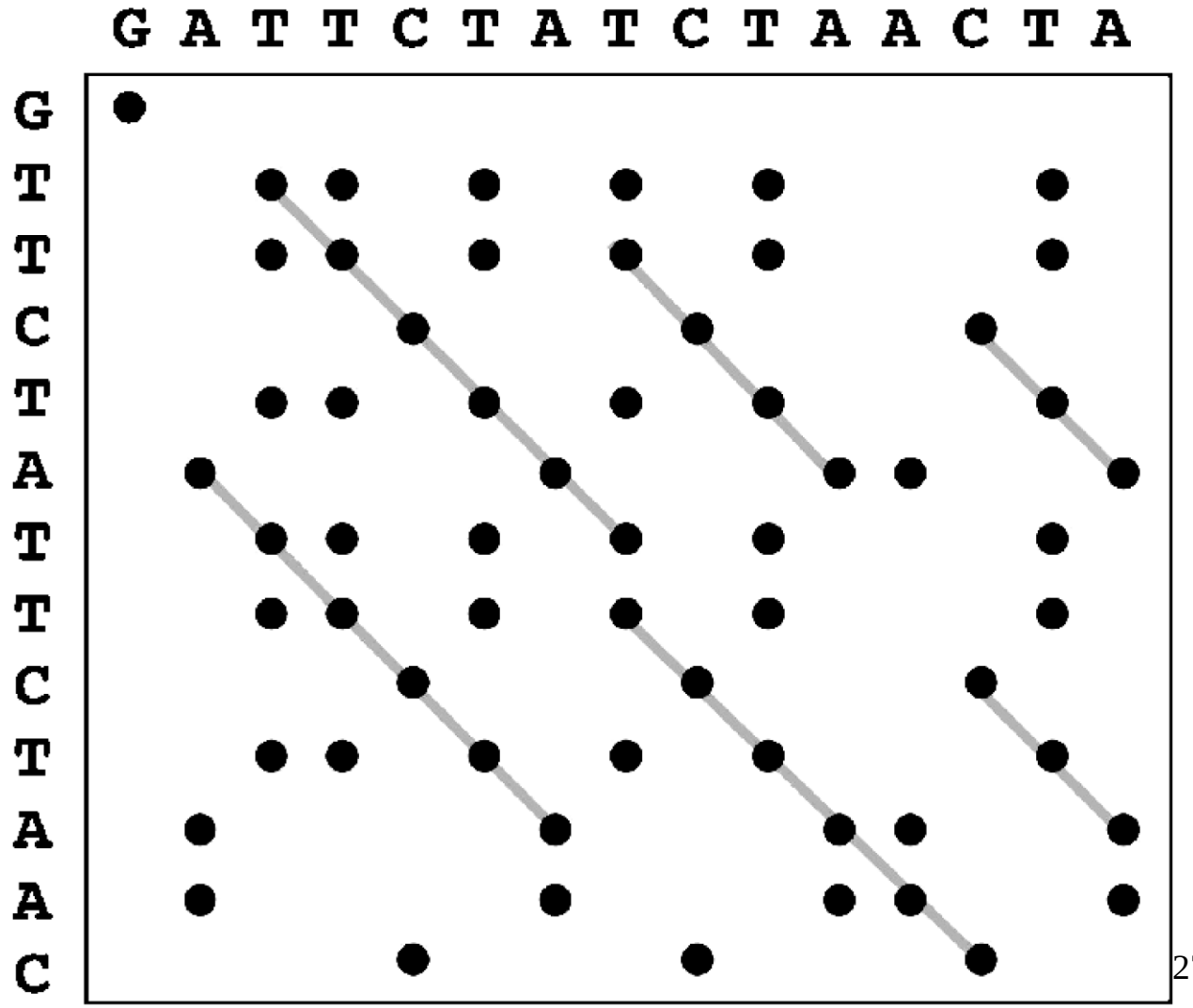
- **Contiguous diagonal lines reveal alignment; "breaks" = gaps (indels)**

Dot Matrix Method: Example 1

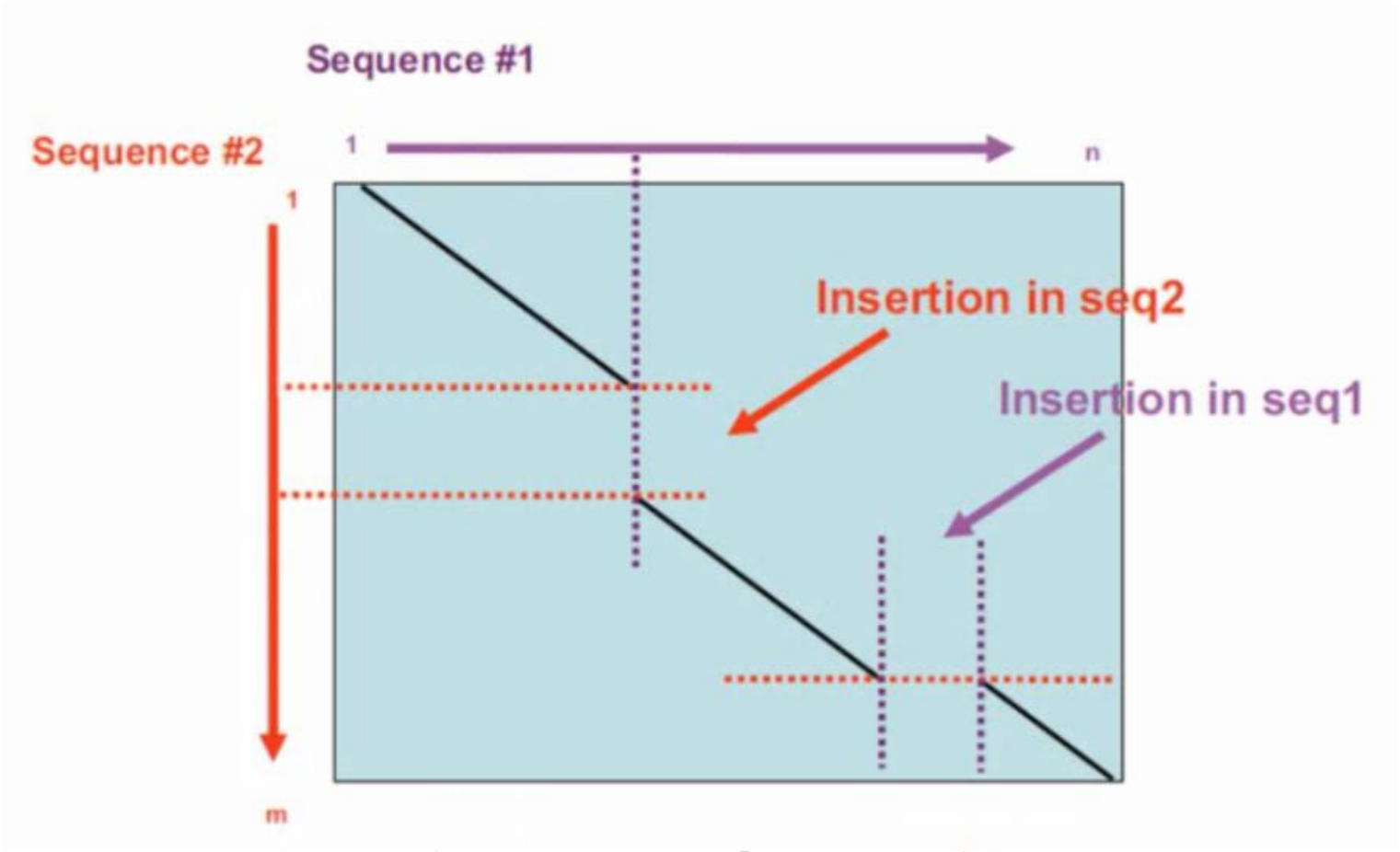
Seq1: **GATTCTATCTAACTA**
Seq2: **GTTCTATTCTAAC**

When comparing 2 sequences:

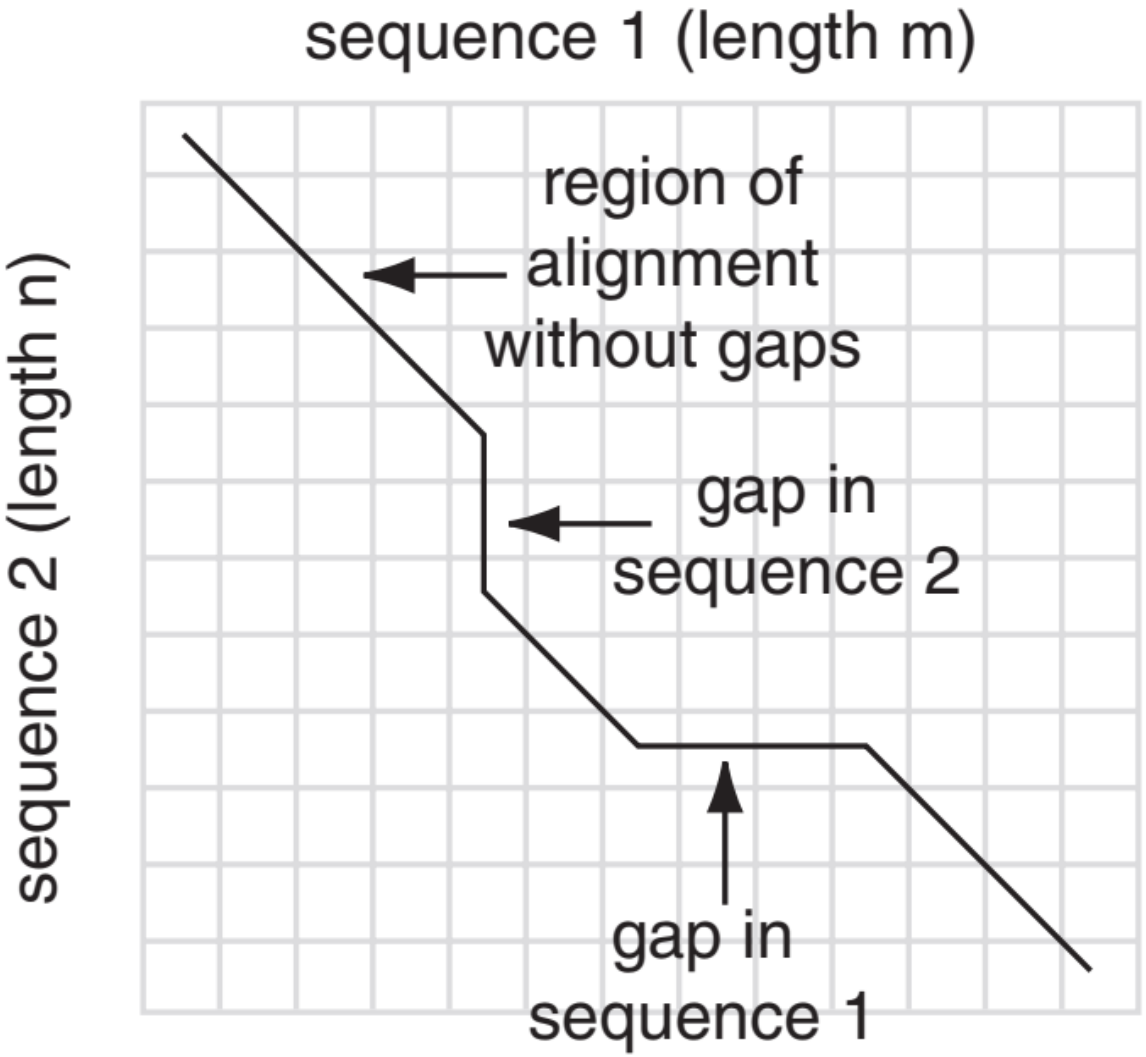
- *Diagonal lines of dots* indicate ***regions of similarity*** between 2 sequences
- *Reverse diagonals* (perpendicular to diagonal) indicate ***inversions***



Dot Matrix Method

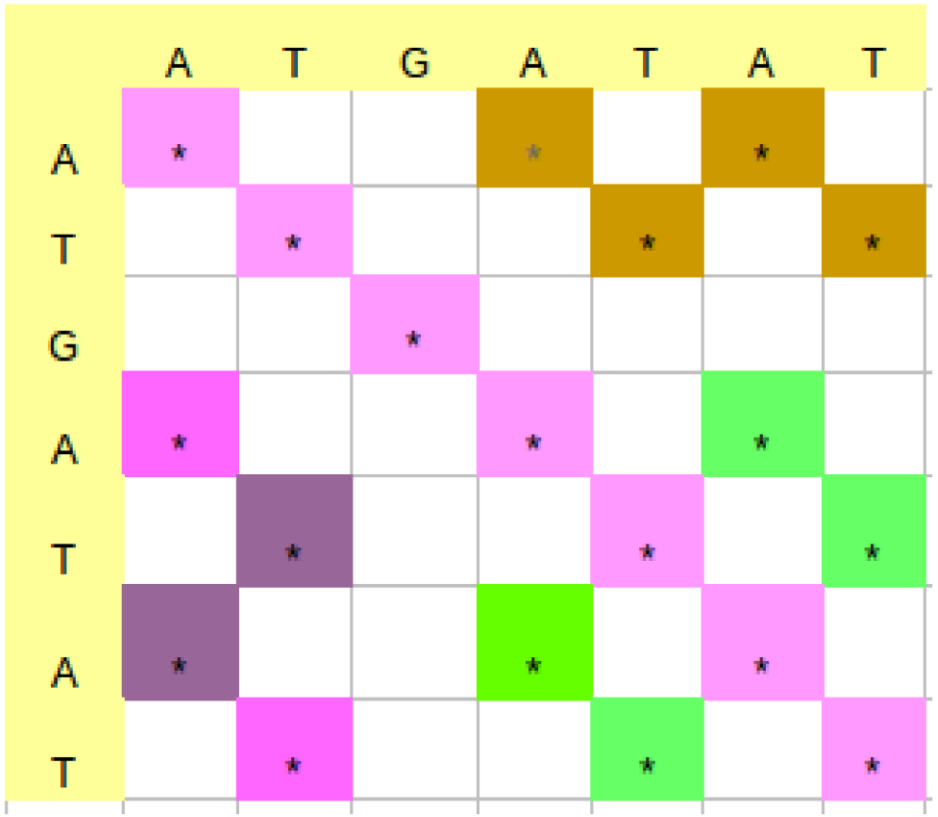


Dot Matrix Method



Dot Matrix Method: Usage 1

- Identify internal repeat elements:
  - A sequence should be aligned with itself
  - If repeats are present, short parallel lines are observed above and below the main diagonal.



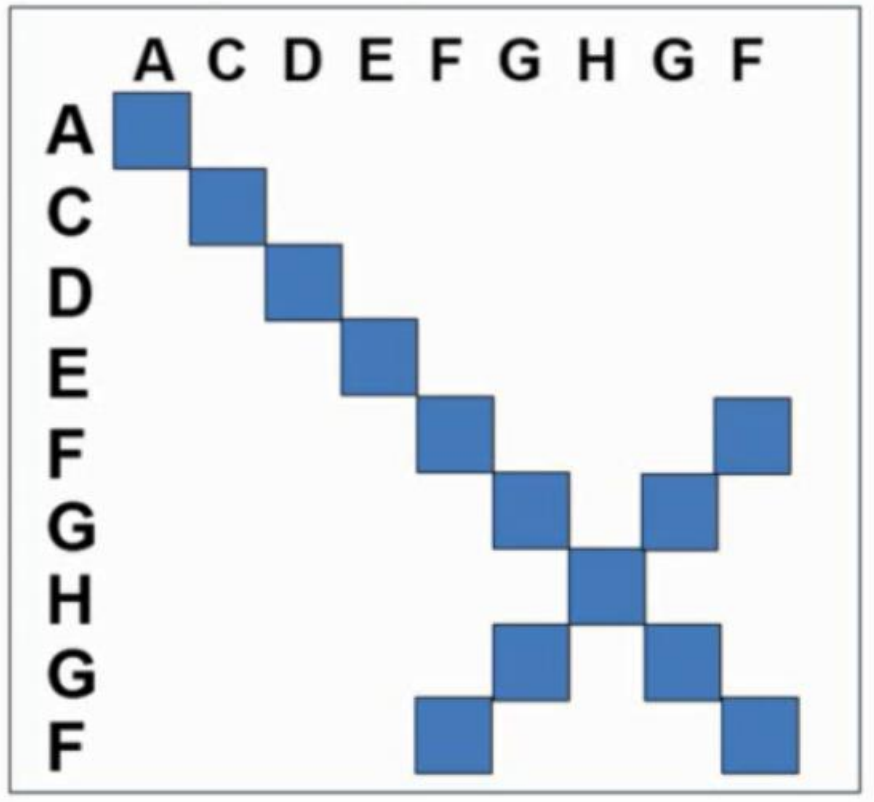
Dot Matrix Method: Usage 2

- Identify *palindromic* sequences:
  - A palindromic sequence is a nucleic acid sequence (DNA or RNA) that is same whether read 5' to 3' on one strand or 3' to 5'.
  - A DNA sequence is compared with its reverse sequence.

	A	T	G	C	G	T	A
A	*						*
T		*				*	
G			*		*		
C				*			
G			*		*		
T		*				*	
A	*						*

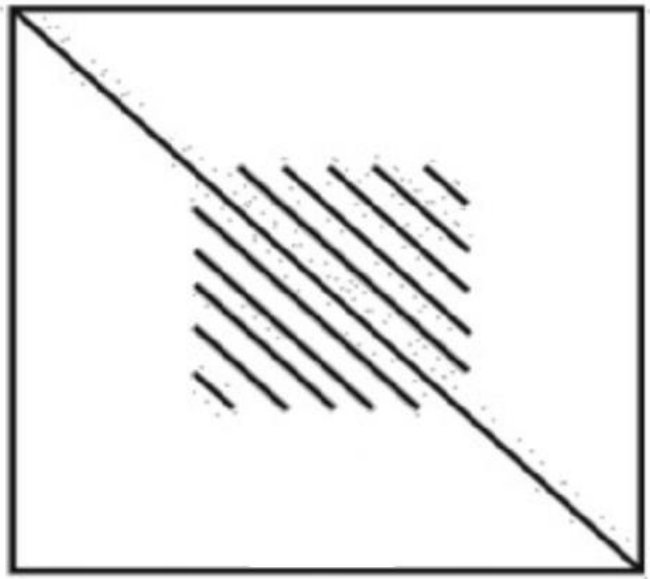
Dot Matrix Method: Usage 3

- Identify *inverted repeats* sequences



Dot Matrix Method: Usage 4

- Identify *Low complexity region*:
  - Regions around the diagonal which all obtaining a high score.



	C	D	E	E	E	E	E	F	G
C	o								
D		o							
E			o	o	o	o	o		
E			o	o	o	o	o		
E			o	o	o	o	o		
E			o	o	o	o	o		
E			o	o	o	o	o		
F								o	
G									o

Dot Matrix Limitation

- For longer sequence, memory required for the graphical representation is very high. So long sequence cannot be aligned.
- Lots of insignificant matches makes it noisy (so many off diagonal appear).
- Time required to compare two sequences is proportional to the product of length of the sequences:
  - Higher efficiency of short sequence.
  - Low efficiency of long sequence

Dot plot software

- Dotmatcher: bioweb.pasteur.fr/seqanal/interfaces/dotmatcher.html
- Dotpath
- Polydot
- Dottup: emboss.bioinformatics.nl/cgi-bin/emboss/dottup
- Dothelix: www.genebee.msu.su/services/dhm/advanced.html
- MatrixPlot: www.cbs.dtu.dk/services/MatrixPlot

Gap Penalties

- Performing *optimal alignment* between sequences often involves applying gaps that represent *insertions and deletions*.
- In *natural evolutionary* processes insertion and deletions are relatively rare in comparison to substitutions:
  - Introducing gaps should be made more difficult computationally
- *Too low penalty*: gaps can become too numerous to allow even nonrelated sequences to be matched up with high similarity scores.
- *Too high penalty*: gaps may become too difficult to appear, and reasonable alignment cannot be achieved.

Gap Penalties (Cont.)

- It is easier to extend a gap that has already been started:
  - Gap opening should have a much higher penalty than gap extension.
- The total gap penalty (W) is a linear function of gap length as follow:

$$W = \gamma + \delta \times (k - 1)$$

  where γ is the gap opening penalty, δ is the gap extension penalty, and k is the length of the gap.
- Besides the affine gap penalty, a *constant gap penalty* is sometimes also used.

Gap Penalty Example

```
Seq1 = GACGCCGAACG
Seq2 = GACGCACG
```

Scores
Match: +2
Gap opening: -5
Gap extension: -1

```
GACGCCGAACG
|||||   |||
GACGC---ACG
```

```
GACGCCGAACG
||||  |  |  ||
GACG-C-A-CG
```

What's a better alignment?

8*2 - 5 - 2 = 9

8*2 - 3*5 = 1

Dynamic Programming Method

Dynamic Programming Method

- For pairwise sequence alignment
- **Idea:** Display one sequence above another with spaces inserted in both to reveal similarity.
- It is fundamentally similar to the dot matrix method.
  - It finds alignment in a more quantitative way by converting a dot matrix into a scoring matrix
- **Global:** Needleman-Wunsch (NW)
- **Local**: Smith-Waterman (SW)

```
CAT-TCA-C
|  |   ||  |
C-TCGCAGC
```

- Both NW and SW use ***dynamic programming***
- Variations:
  - Gap penalty functions
  - Scoring matrices

Dynamic Programming - Key Idea

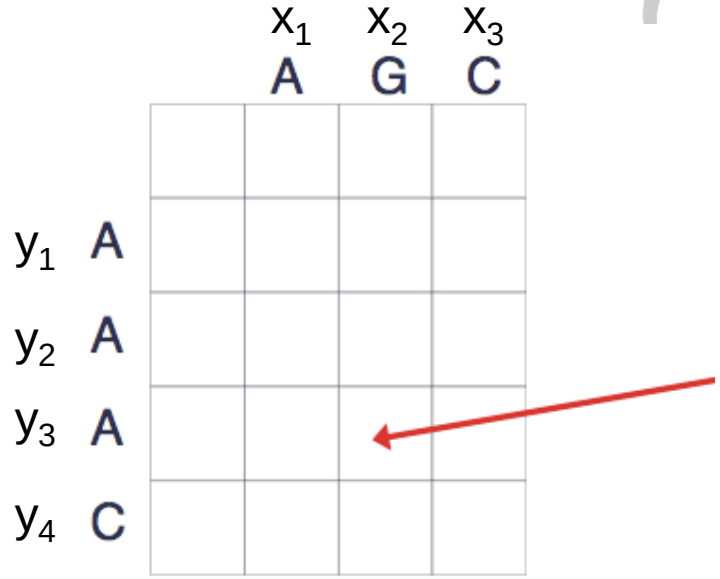
- The score of the best possible alignment that ends at a given pair of positions (i, j) is equal to:
  - the score of best alignment ending just previous to those two positions (i.e., ending at i-1, j-1)

  PLUS

  - the score for aligning x_i and y_j

Global Alignment: Formulation & Notations

- Given two sequences (strings)
  - $X = x_1x_2...x_N$ of length N $\quad x = \text{AGC} \quad N = 3$
  - $Y = y_1y_2...y_M$ of length M $\quad y = \text{AAAC} \quad M = 4$
- Construct a matrix with $(N+1)$ x $(M+1)$ elements, where
  - $S(i,j)$ = Score of best alignment of $x[1..i]=x_1x_2...x_i$ with $y[1..j]=y_1y_2...y_j$



Which means: Score of best alignment of a *prefix* of X and a *prefix* of Y

S(2,3) = score of best alignment of AG (X_1X_2) to AAA ($Y_1Y_2Y_3$)

Dynamic Programming - 4 Steps

1. ***Define score of optimum alignment***, using recursion
2. ***Initialize and fill in a DP matrix*** for storing optimal scores of subproblems, by solving smallest subproblems first (bottom-up approach)
3. ***Calculate score of optimum alignment(s)***
4. ***Trace back through matrix to recover optimum alignment(s)*** that generated optimal score

1- Define Score of Optimum Alignment

Define:

$x_{1..i}$ = Prefix of length i of x

$y_{1..j}$ = Prefix of length j of y

$S(i, j)$ = Score of optimum alignment of $x_{1..i}$ and $y_{1..j}$

Initial conditions:

$$S(i,0) = -i \cdot \gamma \quad S(0,j) = -j \cdot \gamma$$

α = Match Reward
β = Mismatch Penalty
γ = Gap penalty

Recursive definition:

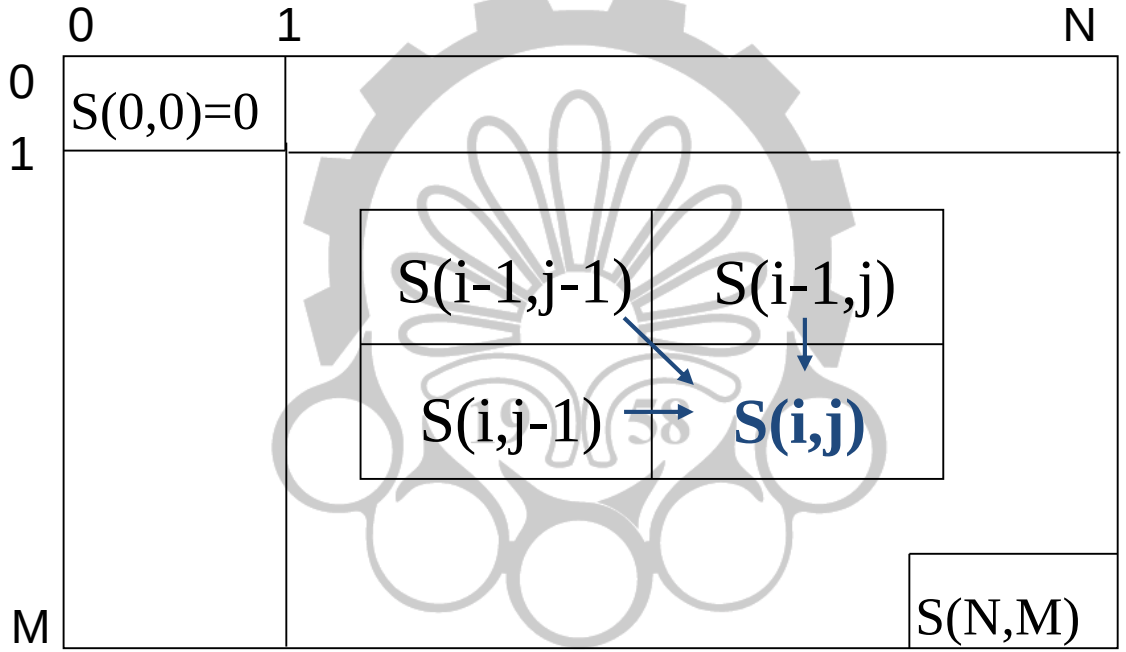
$\sigma(x_i, y_j)$ = **α *or* β**

For $1 \le i \le N$, $1 \le j \le M$:

$$S(i,j) = \max \begin{cases} S(i-1, j-1) + \sigma(x_i, y_j) \\ S(i-1, j) - \gamma \\ S(i, j-1) - \gamma \end{cases}$$

2- Initialize & Fill in DP Matrix

- Construct sequence *vs* sequence matrix:



Recursion

$$S(i,j)=\max\begin{cases} S(i-1,j-1)+\sigma(x_i,y_j) \\ S(i-1,j)-\gamma \\ S(i,j-1)-\gamma \end{cases}$$

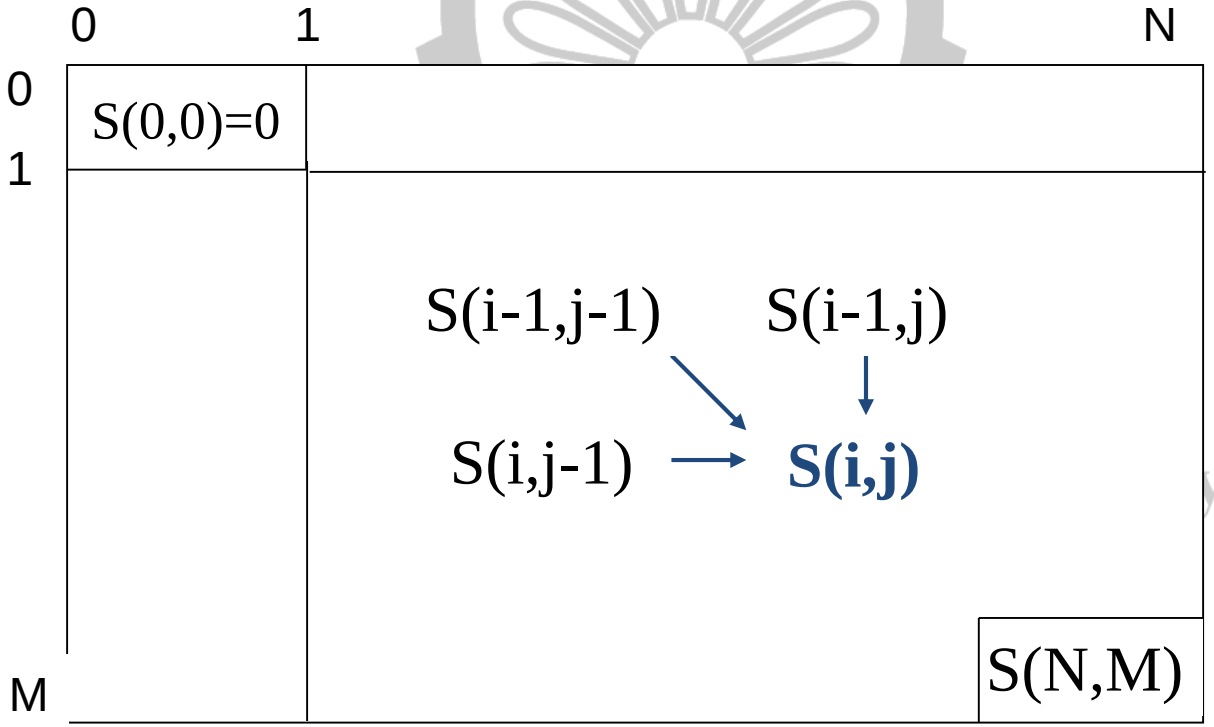
Initialization

$$S(i,0)=-i\cdot\gamma$$

$$S(0,j)=-j\cdot\gamma$$

2- Fill in DP Matrix

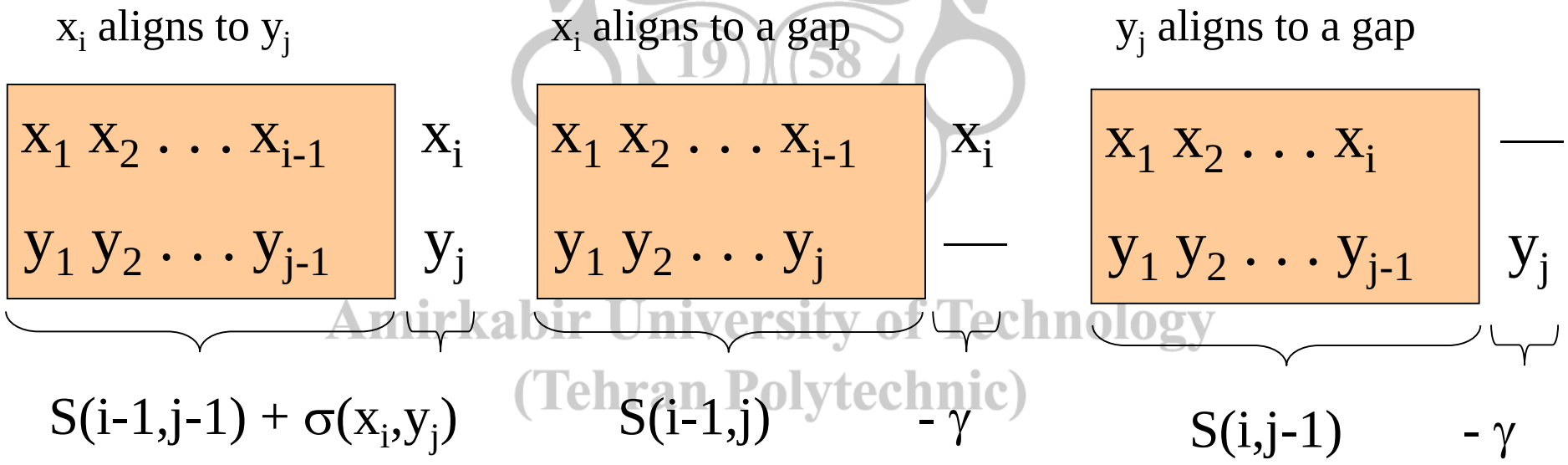
- Fill in from [0,0] to [N,M] (row by row), calculating best possible score for each alignment including residues at [i,j]
- Keep track of dependencies of scores (in a pointer matrix).



3- Calculate Score S(N,M) of Optimum Alignment

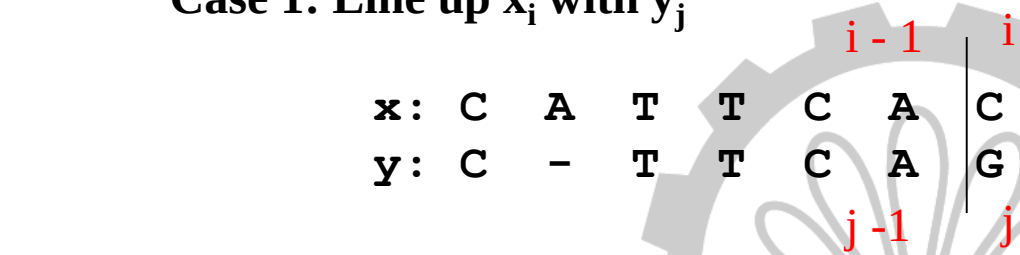
What happens in *last step* in alignment of x[1..i] to y[1..j]?

1 of 3 cases applies:

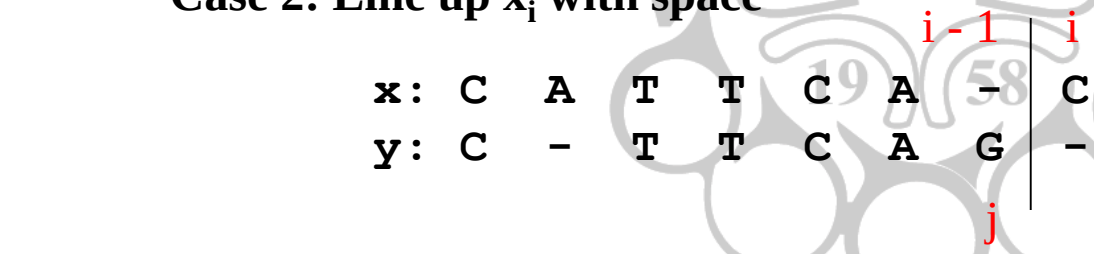


Example

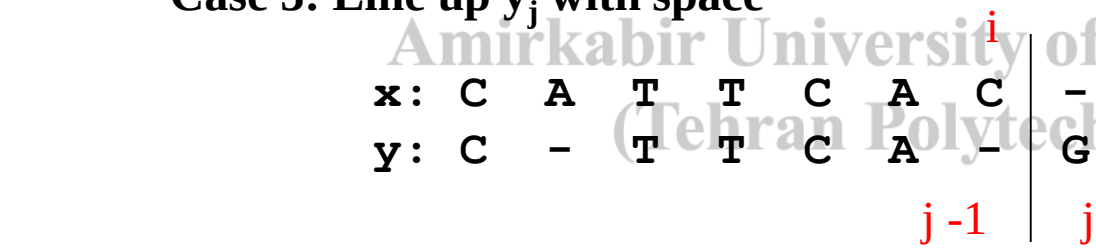
Case 1: Line up x_i with y_j



Case 2: Line up x_i with space



Case 3: Line up y_j with space



Fill in the Matrix

	λ	C	T	C	G	C	A	G	C
λ	0	-5	-10	-15	-20	-25	-30	-35	-40
C	-5	10	5						
A	-10								
T	-15								
T	-20								
C	-25								
A	-30								
C	-35								

+10 for match, -2 for mismatch, -5 for space

Calculate Score of Optimum Alignment

	λ	C	T	C	G	C	A	G	C
λ	0	−5	−10	−15	−20	−25	−30	−35	−40
C	−5	10	5	0	−5	−10	−15	−20	−25
A	−10	5	8	3	−2	−7	0	−5	−10
T	−15	0	15	10	5	0	−5	−2	−7
T	−20	−5	10	13	8	3	−2	−7	−4
C	−25	−10	5	20	15	18	13	8	3
A	−30	−15	0	15	18	13	28	23	18
C	−35	−20	−5	10	13	28	23	26	33

+10 for match, -2 for mismatch, -5 for space

4- Trace Back Through Matrix for Global Alignment

- **How?** "Repeat" alignment calculations in reverse order, starting at from position with highest score and following path, position by position, back through matrix
- **Result?** Optimal alignment(s) of sequences
- Start in lower right corner & trace back to upper left
- Each arrow introduces one character at end of sequence alignment:
  - A horizontal move puts a gap in *left* sequence
  - A vertical move puts a gap in *top* sequence
  - A diagonal move uses one character from *each* sequence

Trace Back to Recover Alignment

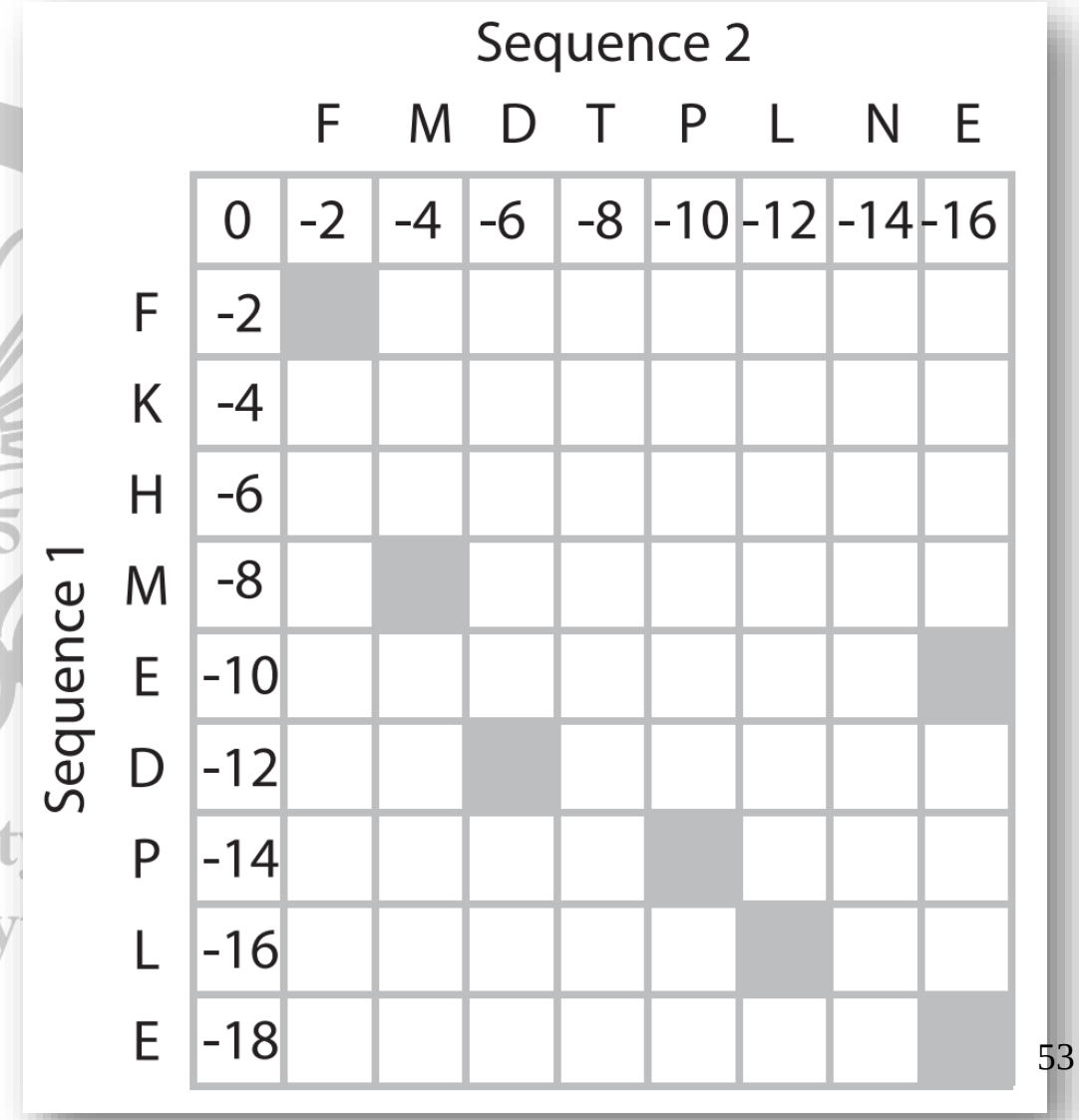
		λ	C	T	C	G	C	A	G	C
	0	−5	−10	−15	−20	−25	−30	−35	−40
λ	−5	10	5	0	−5	−10	−15	−20	−25
C	−10	5	8	3	−2	−7	0	−5	−10
A	−15	0	15	10	5	0	−5	−2	−7
T	−20	−5	10	13	8	3	−2	−7	−4
T	−25	−10	5	20	15	18	13	8	3
C	−30	−15	0	15	18	13	28	23	18
A	−35	−20	−5	10	13	28	23	26	33
C									

Can **have > 1 optimum alignment; this example has 2**

Example 2

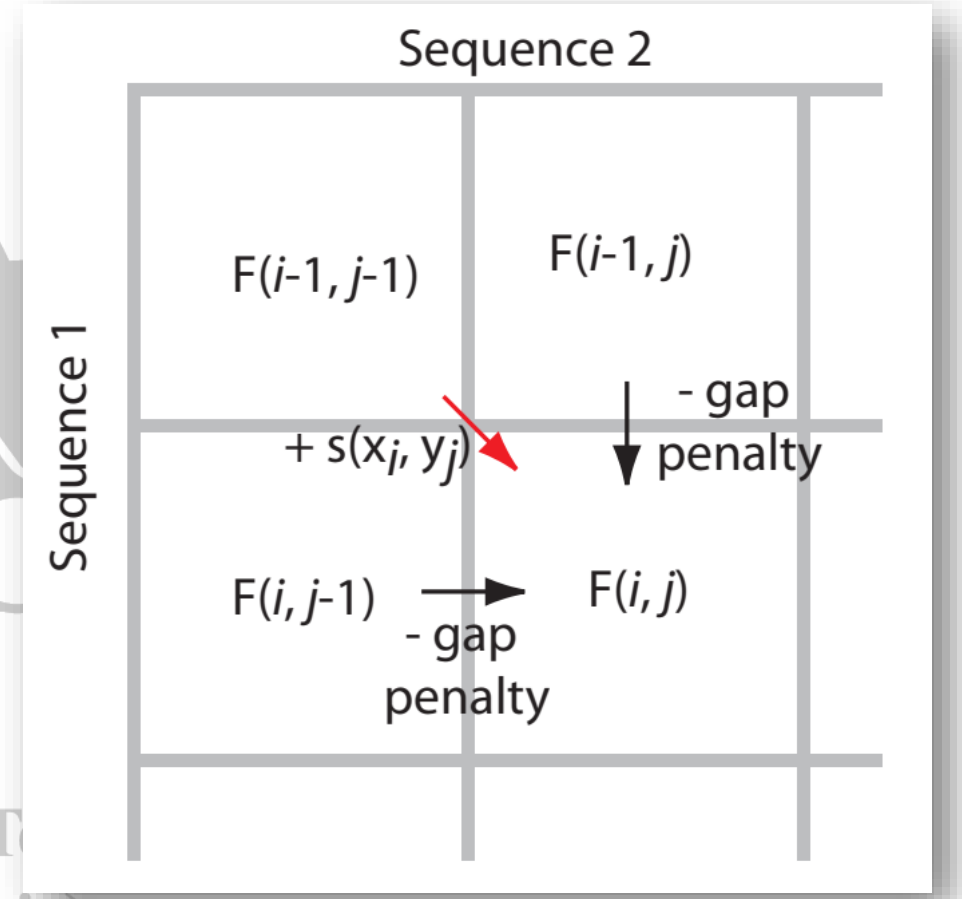
Seq1 = **FKHMEDPLE**
Seq2 = **FMDTPLNE**

Score (this example) = +1 (match)
-2 (mismatch)
-2 (gap penalty)

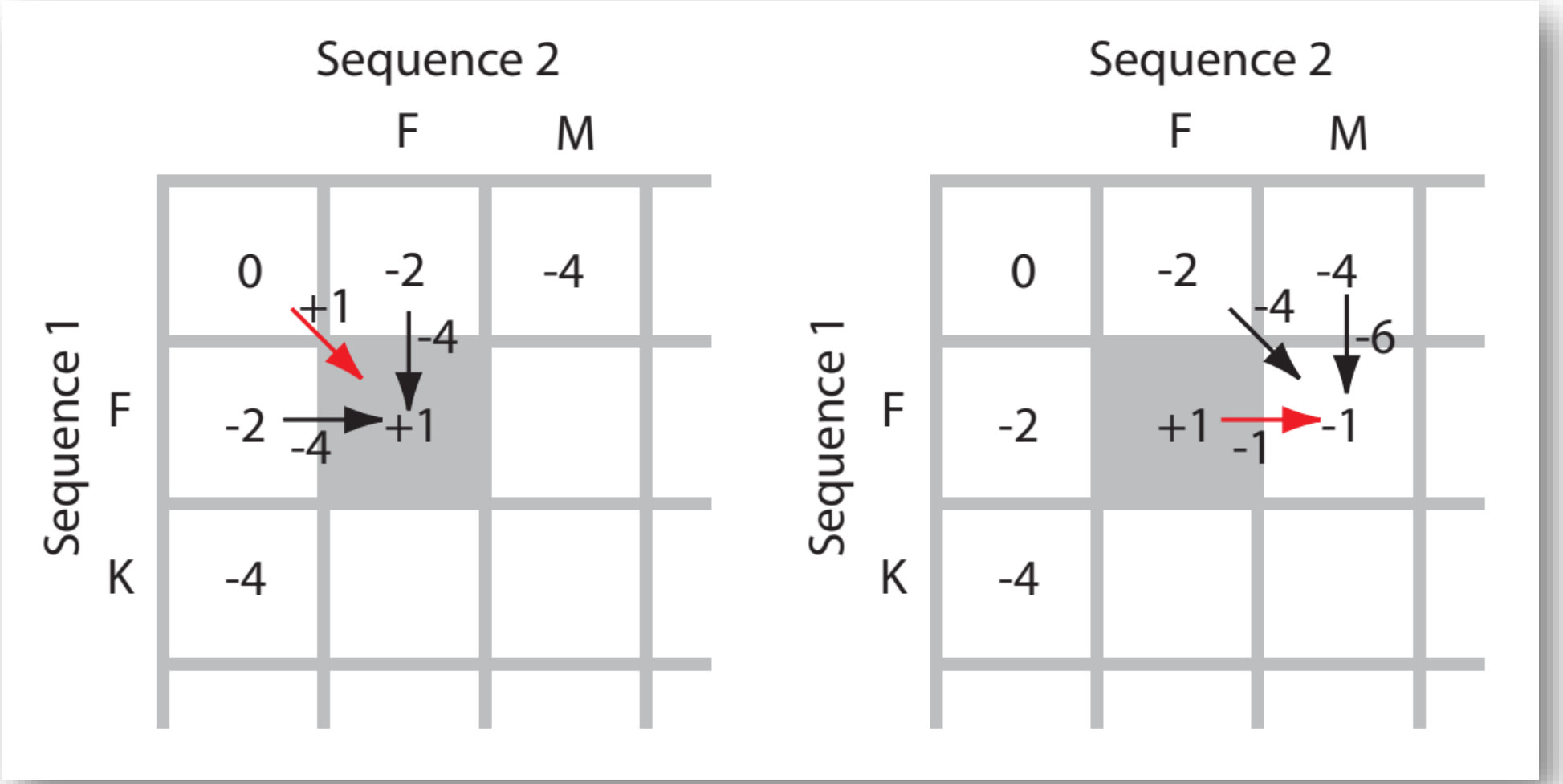


Example 2: Cont.

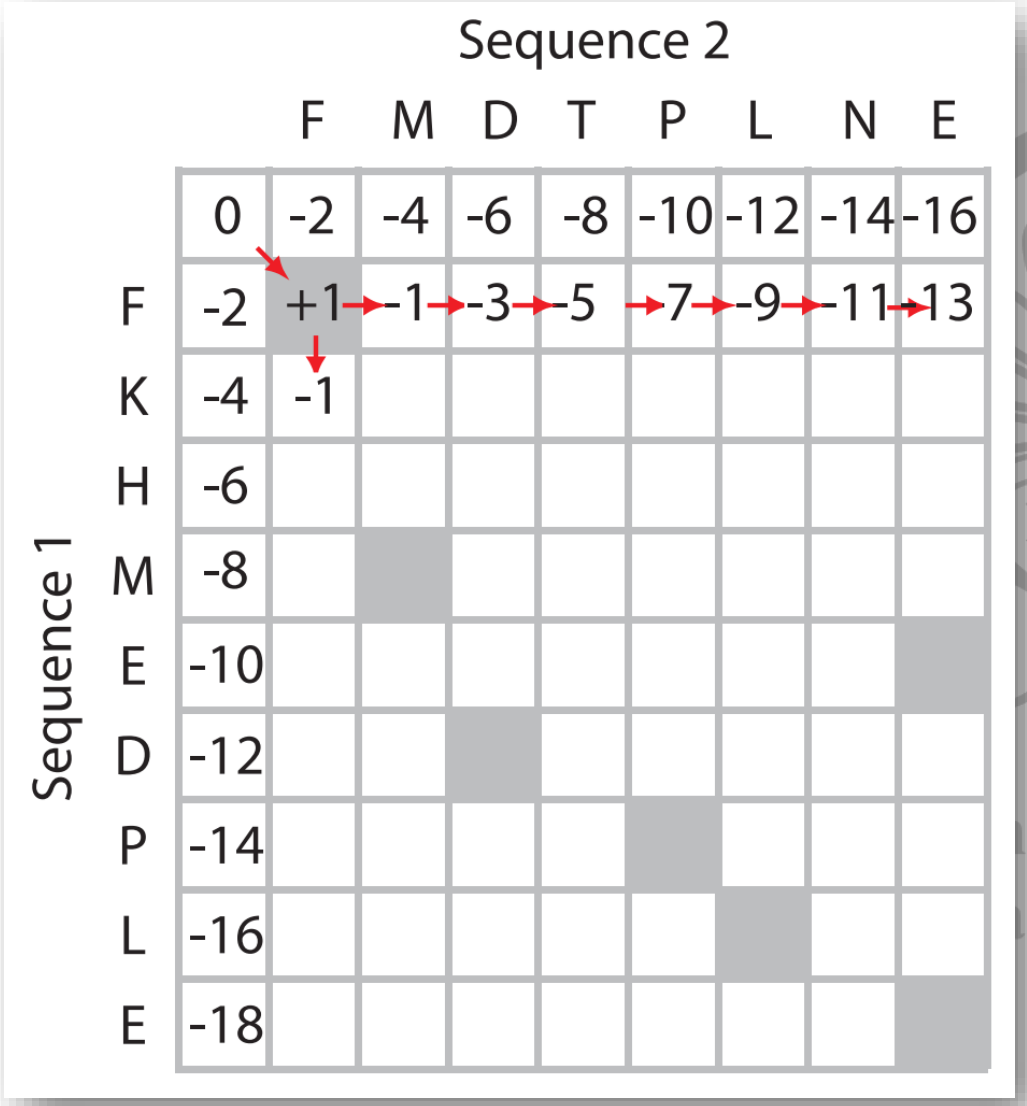
$$\text{Score} = \text{Max} \begin{cases} F(i-1, j-1) + s(x_i, y_i) \\ F(i-1, j) - \text{gap penalty} \\ F(i, j-1) - \text{gap penalty} \end{cases}$$

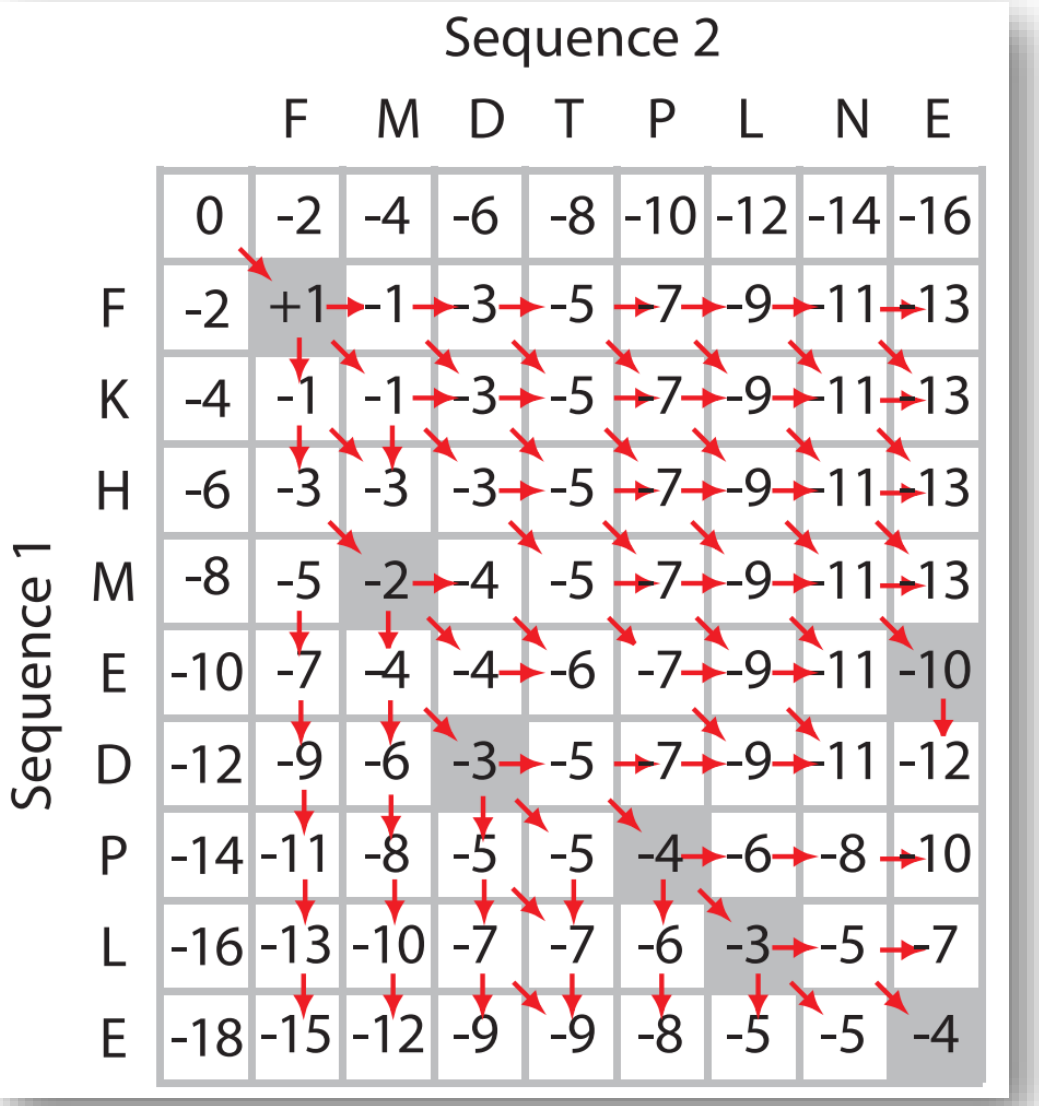


Example 2: Cont.

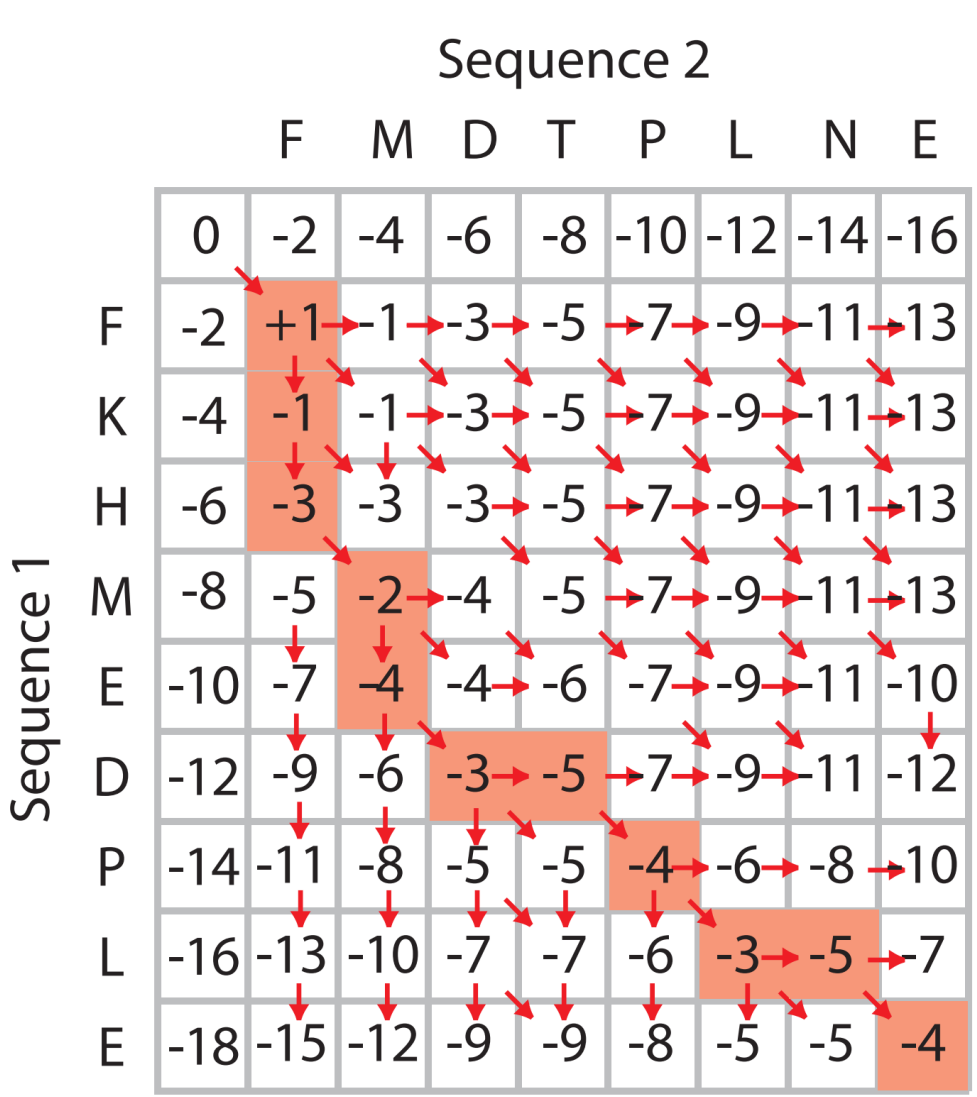


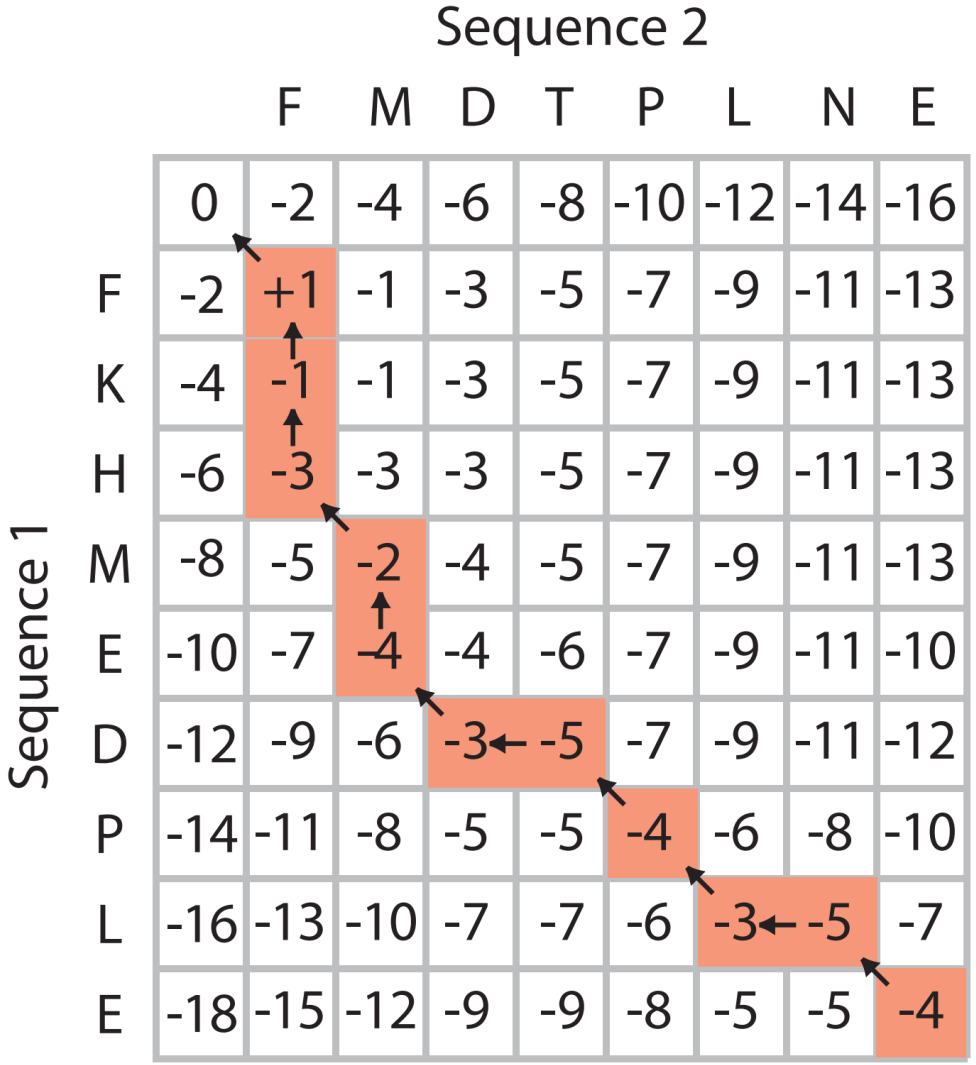
Example 2: Cont.





Example 2: Cont.





Example 2: Cont.

Seq1 = **FKHMEDPLE**
Seq2 = **FMDTPLNE**

	+1	-1	-3	-2	-4	-3	← -5	-4	-3	← -5	-4
Sequence 1	F	K	H	M	E	D	-	P	L	-	E
Sequence 2	F	-	-	M	-	D	T	P	L	N	E

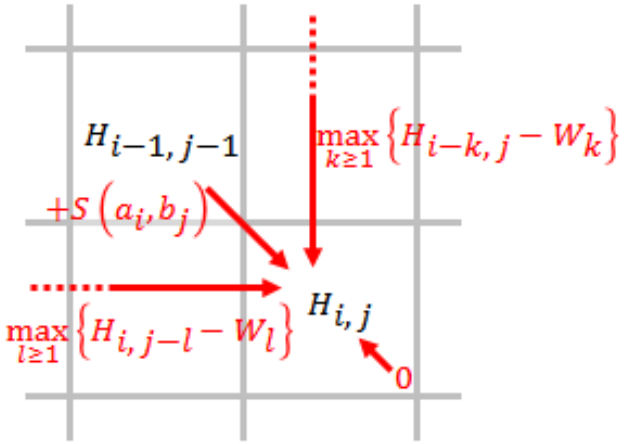
Sequence 2

Sequence 1		F	M	D	T	P	L	N	E
	0	-2	-4	-6	-8	-10	-12	-14	-16
F	-2	+1	-1	-3	-5	-7	-9	-11	-13
K	-4	-1	-1	-3	-5	-7	-9	-11	-13
H	-6	-3	-3	-3	-5	-7	-9	-11	-13
M	-8	-5	-2	-4	-5	-7	-9	-11	-13
E	-10	-7	-4	-4	-6	-7	-9	-11	-10
D	-12	-9	-6	-3	← -5	-7	-9	-11	-12
P	-14	-11	-8	-5	-5	-4	-6	-8	-10
L	-16	-13	-10	-7	-7	-6	-3	← -5	-7
E	-18	-15	-12	-9	-9	-8	-5	-5	-4

Local Alignment: Smith-Waterman

- Given two sequences (strings)
  - $A = a_1a_2...a_n$ of length n and $B = b_1b_2...b_m$ of length m
- Construct a matrix with $(n+1)$ x $(m+1)$ elements, where
  - $H(i,j)$ = Score of alignment of $x[1..i]=x_1x_2...x_i$ with $y[1..j]=y_1y_2...y_j$
- Initialize top row & leftmost column of matrix with "0“

$$H_{ij} = \max \begin{cases} H_{i-1,j-1} + s(a_i, b_j), \\ \max_{k \geq 1}\{H_{i-k,j} - W_k\}, \\ \max_{l \geq 1}\{H_{i,j-l} - W_l\}, \\ 0 \end{cases} \qquad (1 \leq i \leq n, 1 \leq j \leq m)$$



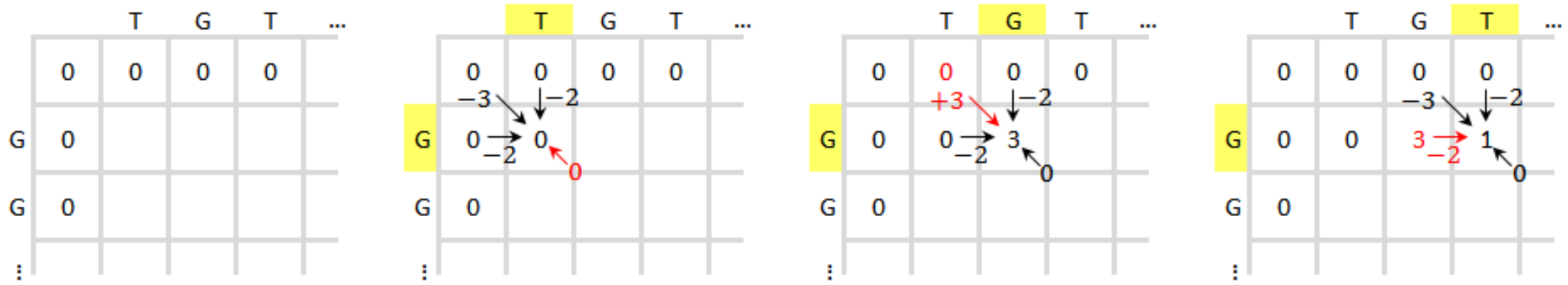
where W_k is the penalty of a gap that has length k (affine gap penalty) and $s(a, b)$ is similarity score of two elements.

- Traceback: starting at the highest score in the scoring matrix H and ending at a matrix cell that has a score of 0.

Example

```
Seq1 = TGTTACGG
Seq2 = GGTTGACTA
```

Match: +3
Mismatch: -3
Gap penalty: -2



Example (Cont.)

		T	G	T	T	A	C	G	G
	0	0	0	0	0	0	0	0	0
G	0	0	3	1	0	0	0	3	3
G	0	0	3	1	0	0	0	3	6
T	0	3	1	6	4	2	0	1	4
T	0	3	1	4	9	7	5	3	2
G	0	1	6	4	7	6	4	8	6
A	0	0	4	3	5	10	8	6	5
C	0	0	2	1	3	8	13	11	9
T	0	3	1	5	4	6	11	10	8
A	0	1	0	3	2	7	9	8	7

		T	G	T	T	A	C	G	G
	0	0	0	0	0	0	0	0	0
G	0	0	3	1	0	0	0	3	3
G	0	0	3	1	0	0	0	3	6
T	0	3	1	6	4	2	0	1	4
T	0	3	1	4	9	7	5	3	2
G	0	1	6	4	7	6	4	8	6
A	0	0	4	3	5	10	8	6	5
C	0	0	2	1	3	8	13	11	9
T	0	3	1	5	4	6	11	10	8
A	0	1	0	3	2	7	9	8	7

```
GTT-AC
|||  ||
GTTGAC
```

Initialize the scoring matrix

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

Substitution matrix: $S(a_i, b_j) = \begin{cases} +3, & a_i = b_j \\ -3, & a_i \neq b_j \end{cases}$

Gap penalty: $W_k = kW_1$

$W_1 = 2$

Example 2

D	E	-	S

D
E
A
S

	-	D	E	S	I	G	N
-	0	0	0	0	0	0	0
I	0	0	0	0	5	4	3
D	0	5	4	3	4	4	3
E	0	4	10	9	8	7	6
A	0	3	9	9	8	7	6
S	0	2	8	14	13	12	11

Match = +5

Mismatch = -1

Gap = -1

1: DESIGN

2: IDEAS

Aligned:

1: DE-S

2: DEAS

Semi-global Alignment

- Not penalize gaps at the beginning and/or end of the alignment.
- Semi-global alignment is a modification of global alignment that allows the user to specify that gaps will be penalty-free at the beginning and/or at the end of one of the sequences.
- Usage:
  - Searching a specified domain in a newly discovered protein
  - Searching a gen (or a segment of a DNA) in a whole genome

```
CTGTCGTGCACG
  ||  ||||
--TGCCGTG----
```

Semi-global Alignment

- Not to penalize start gaps
  - can be accounted for by initializing the first row and first column of the dynamic programming table to zeros.

	-	y_1	y_2	$\cdots$	y_n
-	0	$-g$	$-2g$	$\cdots$	$-ng$
x_1	$-g$				
x_2	$-2g$				
$\vdots$	$\vdots$				
x_m	$-mg$				

$\Rightarrow$

	-	y_1	y_2	$\cdots$	y_n
-	0	0	0	$\cdots$	0
x_1	0				
x_2	0				
$\vdots$	$\vdots$				
x_m	0				

- Gaps at the end of the alignment should be ignored
  - No more modifications are needed in the dynamic programming table.
  - The optimal alignment score is now detected as the maximum value on the last row (when **x** is aligned with any prefix of **y**) or column (when **y** is aligned with any prefix of **x**).

	-	y_1	y_2	$\cdots$	y_n
-					$\uparrow$
x_1					.
x_2					.
$\vdots$					.
x_n	$\leftarrow$	.	.	.	.

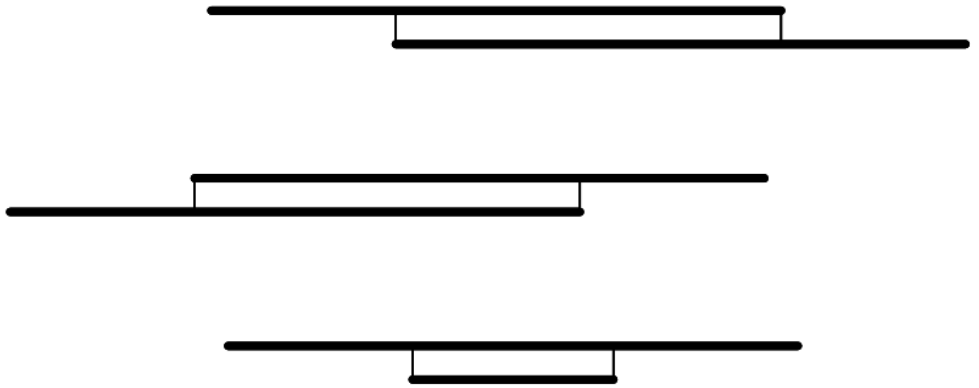
General semi-global alignment

- Variations of this optimal alignment algorithm can accommodate situations where gaps are ignored only for **x** and/or **y** or only at the start or end of the sequence.

Places where gaps are not penalized	Action
Start of x	Initialize first row to zeroes
End of x	Look for max in last row
Start of y	Initialize first column to zeroes
End of y	Look for max in last column

Overlap detection

- An overlap of two sequences is considered an alignment where start and end gaps are ignored.
- When detecting the optimal overlap of **x** and **y**, the possibilities are:
  - a suffix of **x** aligns with a prefix of **y**
  - a suffix of **y** aligns with a prefix of **x**
  - **y** aligns with a substring of **x**
  - **x** aligns with a substring of **y**



Alignment Comparison

```
1 2 3 4 5 6 7 8 9 10 11
C G T C C G A A G T G
    | . | | | |
⋆ ⋆ T A C G A A ⋆ ⋆ ⋆
```

(a) Global alignment

```
3 4 5 6 7 8
T C C G A A
| . | | | |
T A C G A A
```

(b) Local alignment

```
1 2 3 4 5 6 7 8
C G T C C G A A
    | . | | | |
⋆ ⋆ T A C G A A
```

(c) Semi-global alignment

Example

BLOSUM50, linear gap penalty d=8

GAWGHEE

PAW-HEA

		H	E	A	G	A	W	G	H	E	E
	0	0	0	**0**	0	0	0	0	0	0	0
P	0	-2	-1	-1	**-2**	-1	-4	-2	-2	-1	-1
A	0	-2	-2	4	-1	**3**	-4	-4	-4	-3	-2
W	0	-3	-5	-4	1	-4	**18**	**10**	2	6	-6
H	0	10	2	6	-6	-1	10	16	**20**	12	4
E	0	2	16	8	0	7	2	8	16	**26**	18
A	0	-2	8	21	13	5	3	2	8	18	**25**
E	0	0	4	13	18	12	4	4	2	14	24

Dynamic Programming Software

- Global alignment (Needleman–Wunsch algorithm):
  - GAP: http://bioinformatics.iastate.edu/aat/align/align.html

- Local alignment (Smith–Waterman algorithm)
  - SIM: http://bioinformatics.iastate.edu/aat/align/align.html
  - SSEARCH: http://pir.georgetown.edu/pirwww/search/pairwise.html
  - LALIGN: www.ch.embnet.org/software/LALIGN form.html

"Scoring" or "Substitution" Matrices

- The alignment procedure in DP algorithm has to make use of a *scoring system*:
  - A set of values for quantifying the likelihood of one residue being substituted by another.
  - The scoring systems is called a *substitution matrix*.
- Scoring matrices for *nucleotide sequences* are relatively simple:
  - A *positive value* or high score for a match and a *negative value* or low score for a mismatch.
  - It is based on the assumption that frequencies of mutation are equal for all bases.
  - This assumption may not be realistic:
    - *Transitions* (substitutions between purines and purines or between pyrimidines and pyrimidines) occur more frequently than *transversions* (substitutions between purines and pyrimidines)

Scoring Matrices for Nucleotide Sequences

- Transition: A ↔ G C ↔ T
- Transversion: a purine (A or G) is replaced by a pyrimidine (C or T) or vice versa

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

Identity

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

BLAST

	A	T	C	G
A	1	-5	-5	-1
T	-5	1	-1	-5
C	-5	-1	1	-5
G	-1	-5	-5	1

Transition/Transversion

Amino Acid Substitution Matrices

- Substitution matrices for *amino acids* are more complicated:
  - Scoring has to reflect the physicochemical properties of amino acid residues.
  - As well as the likelihood of certain residues being substituted among true homologous sequences.
- Substitution matrices are 20 × 20 matrices for amino acid.
- There are essentially two types of amino acid substitution matrices:
  - One type is based on interchangeability of the genetic code or amino acid properties.
    - is less accurate than the second approach.
  - The other is derived from empirical studies of amino acid substitutions (PAM and BLOSUM matrices).

Amino Acid Substitution Matrices (Cont.)

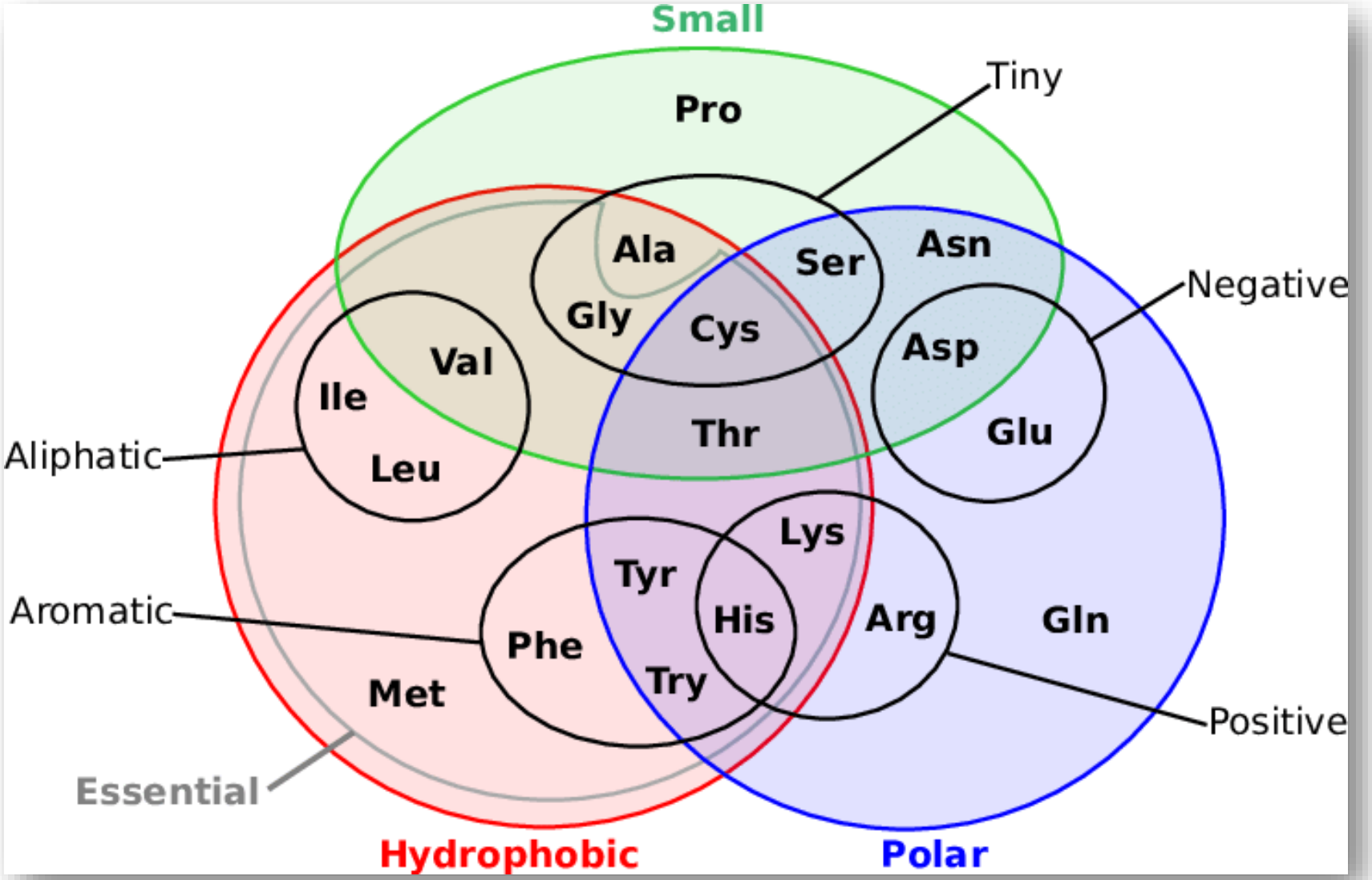
- For a given substitution matrix:
  - *Positive score*: means the frequency of substitutions is greater than would have occurred by random chance.
    - Represent substitutions of very similar residues or identical residues
  - *Zero score*: means the frequency of substitutions is equal to that expected by chance.
    - The amino acids is weakly similar at best in terms of physicochemical properties.
  - *Negative score*: means the frequency of substitutions is less than would have occurred by random chance.
    - Occurs with substitutions between dissimilar residues.

Log-odds Scores

- The substitution matrices apply logarithmic conversions to describe the probability of amino acid substitutions.
  - The converted values are called *log-odds scores (or* ratios*)*.
- *Log-odds ratios*: are logarithmic ratios of the *observed mutation frequency* divided by the *probability of random substitution*.
  - The conversion can be either to the base of 10 or to the base of 2.
- Example: in an alignment that involves 10 sequences, each having *only one aligned* position, 9 of the sequences are F (phenylalanine) and the remaining one I (isoleucine):
  - The observed frequency of I being substituted by F is 0.1
  - The probability of I being substituted by F by random chance is 0.05

$$log - odd = \log \frac{1/10}{1/20} = \log 2 = 1$$

Amino Acid Similarity



"Scoring" or "Substitution" Matrices

- 2 Major types for Amino Acids: **PAM & BLOSUM**

  - ***PAM = Point Accepted Mutation***

    relies on "evolutionary model" based on observed differences in alignments of *closely related proteins*

  - ***BLOSUM = BLOck SUbstitution Matrix***

    based on % aa substitutions observed in blocks of conserved sequences within *evolutionarily divergent proteins*

PAM Matrices

- One PAM unit is defined as 1% of the amino acid positions that have been changed or one mutation per 100 residues.
- Construction of the PAM1 matrix involves alignment of full-length sequences and subsequent construction of phylogenetic trees using the parsimony principle.
- PAM1: probability of one substitution in 100 amino acids
- PAMN: is produced by values of the PAM1 matrix multiplied by itself N times.
- Suffix number (N) reflects amount of "time" passed: rate of expected mutation if N% of amino acids had changed.
- PAM1 - for more closely related sequences (shorter time)
- PAM250 - for more divergent sequences (longer time)

PAM250

- Corresponds to 20% amino acid identity
- Represents 250 mutations per 100 residues
- Corresponds to an expected evolutionary span of 2,500 million years.

PAM Number	Observed Mutation Rate (%)	Sequence Identity (%)
0	0	100
1	1	99
30	25	75
80	50	50
110	40	60
200	75	25
250	80	20

PAM250 Matrix

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

EMBOSS Programs

STEP 1 - Enter your protein sequences

Enter a pair of

PROTEIN

sequences. Enter or paste your first **protein** sequence in any supported format:

DCYQWKSA

Or, upload a file: Browse... No file selected.

AND

Use a example sequence | Clear sequence | See more example inputs

Enter or paste your second **protein** sequence in any supported format:

FPCYWCPNHL

Or, upload a file: Browse... No file selected.

```
#=======================================
#
# Aligned_sequences: 2
# 1: EMBOSS_001
# 2: EMBOSS_001
# Matrix: EPAM250
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 11
# Identity:        3/11 (27.3%)
# Similarity:      4/11 (36.4%)
# Gaps:            4/11 (36.4%)
# Score: 24.0
#
#
#=======================================

EMBOSS_001          1 -DCYQWKSA--       8
                        .|| |.:.
EMBOSS_001          1 FPCY-WCPNHL      10
```

BLOSUM Matrices

- Derived based on direct observation for every possible amino acid substitution in *multiple sequence alignments*.
- Constructed based on more than 2,000 blocks representing 500 groups of protein sequences.
- *Blocks* (sequence patterns) are ungapped alignments of less than sixty amino acid residues in length.
- The frequencies of amino acid substitutions of the residues in these blocks are calculated to produce a substitution matrix.

```
AABCDA...BBCDA
DABCDA.A.BBCBB
BBBCDABA.BCCAA
AAACDAC.DCBCDB
CCBADAB.DBBDCC
AAACAA...BBCCC
```

BLOSUM Matrices (Cont.)

- BLOSUM score for a particular residue pair:
  - Log ratio of *observed residue substitution frequency* versus the *expected probability* of a particular residue.
- Suffix number (n) reflects expected similarity: average identity n% in the MSA from which the matrix was generated
  - BLOSUM45 - for more divergent sequences
  - BLOSUM62 - for less divergent sequences

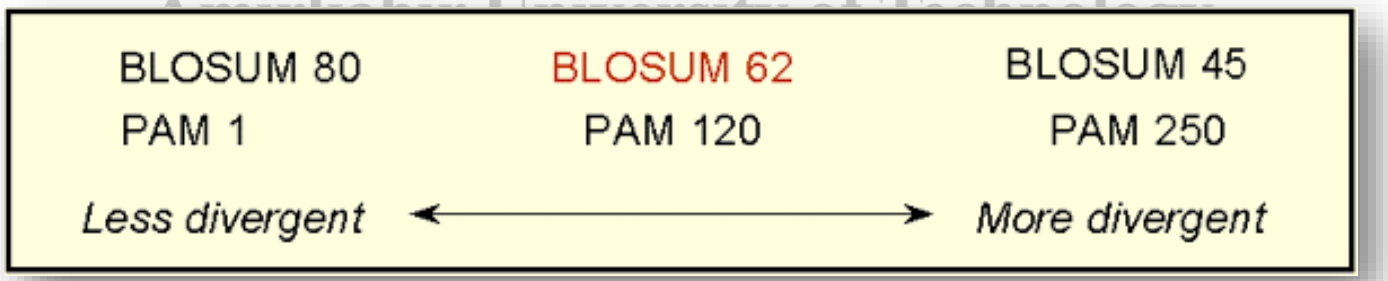
BLOSUM62

- Indicates that the sequences selected for constructing the matrix share an average identity value of 62%

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

Comparison between PAM and BLOSUM

- PAM matrices are used most often for reconstructing phylogenetic trees.
- BLOSUM matrices may be more advantageous in searching databases and finding conserved domains in proteins.
- BLOSUM matrices perform better for local alignment
- PAM matrices perform better for global alignment
- The numbers after PAM and BLOSUM have opposite meaning

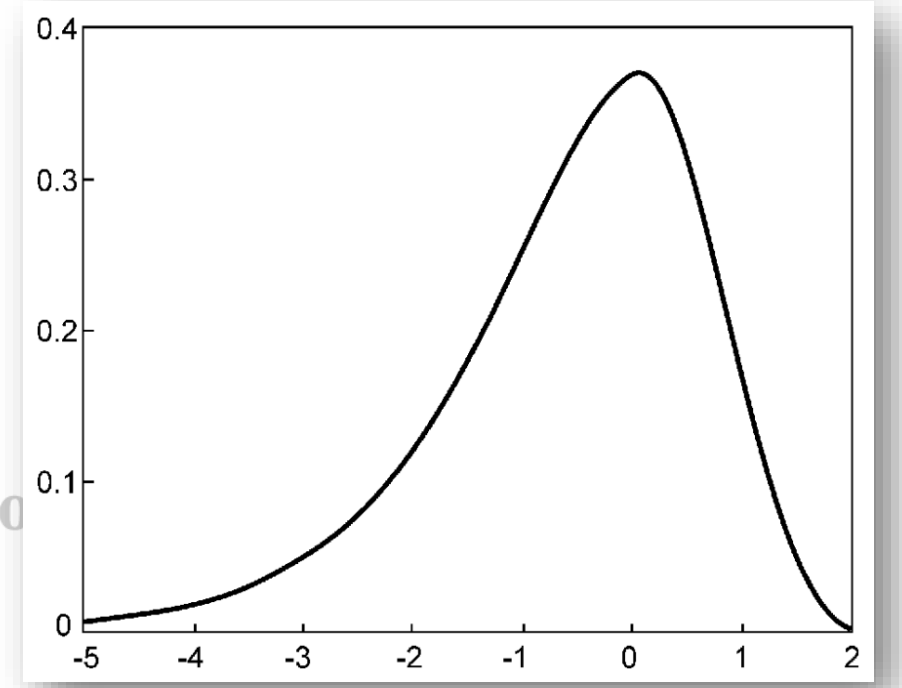


Sequence Alignment Statistics

- Distribution of similarity scores in sequence alignment is not a simple "normal" distribution
- **"Gumbel extreme value distribution"** - a highly skewed normal distribution with a long tail
- The distribution can be expressed as:

$$P = 1 - e^{-Kmne^{-\lambda x}}$$

- where *m* and *n* are the sequence lengths, λ is a scaling factor depends on score matrix, and *K* is a constant that depends on the gap penalty.



Statistical Significance of Sequence Alignment

- Compare score of an alignment with distribution of scores of alignments for many *'randomized'* (one of the two sequences is randomly shuffled) versions of the original sequence
- If score is in extreme margin, then unlikely due to random chance
- **P-value** = probability that original alignment is due to random chance (lower P is better)

$P < 10^{-100}$	indicates an exact match
$P = 10^{-50} - 10^{-100}$	nearly identical match
$P = 10^{-5} - 10^{-50}$	sequences have clear homology
$P = 10^{-1} - 10^{-5}$	possible distant homologs
$P > 10^{-1}$	may be randomly related

Check out: **PRSS (Probability of Random Shuffles)**
http://www.ch.embnet.org/software/PRSS_form.html

References

- Mostly used:
  - Essential bioinformatics, Chapter 3 (Pairwise Sequence Alignment)
- Second reference:
  - Bioinformatics and functional genomics, Chapter 3 (Pairwise Sequence Alignment)

- IP notice: some slides were selected from Drena Dobbs' slides.

Thanks for your attention