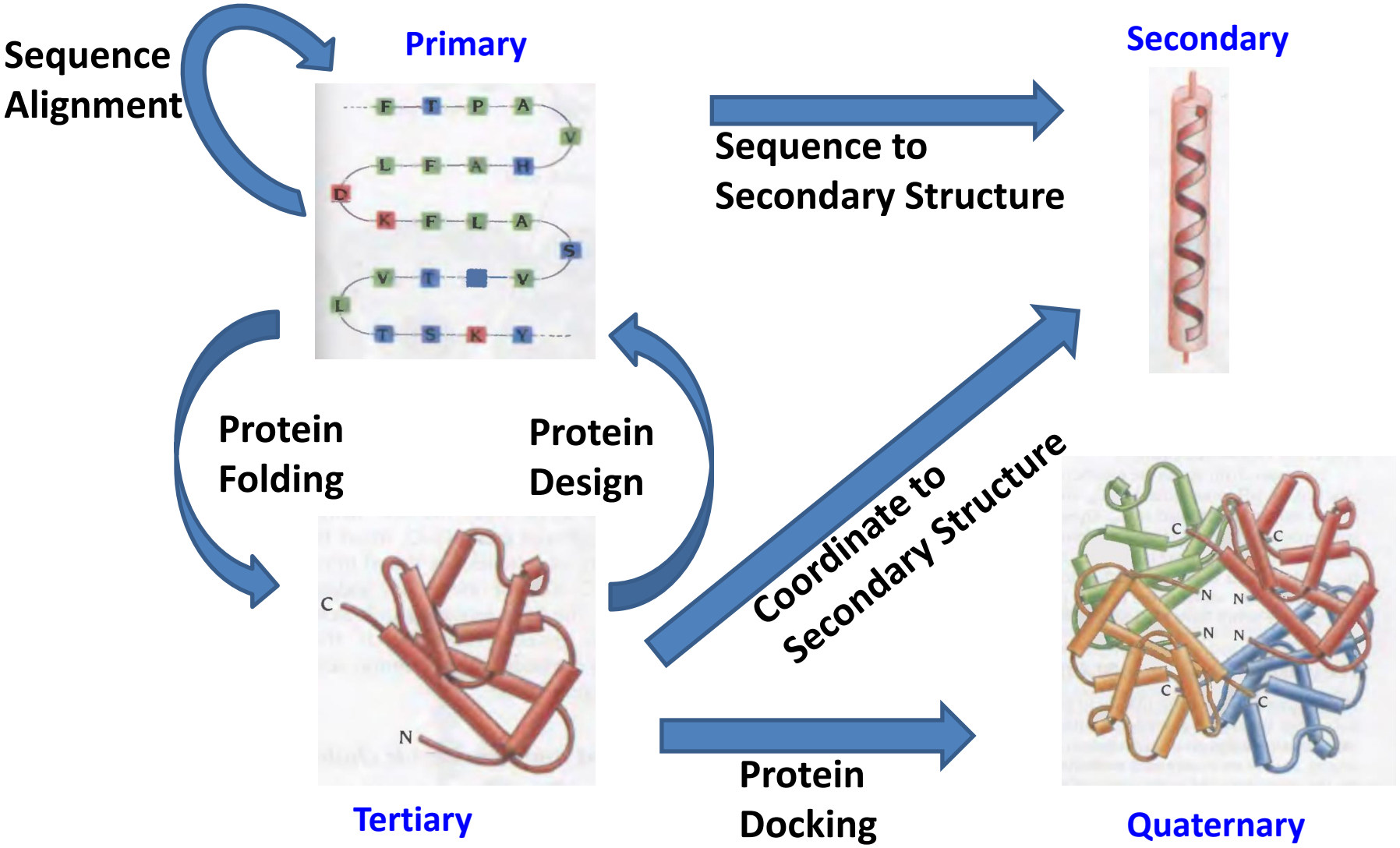
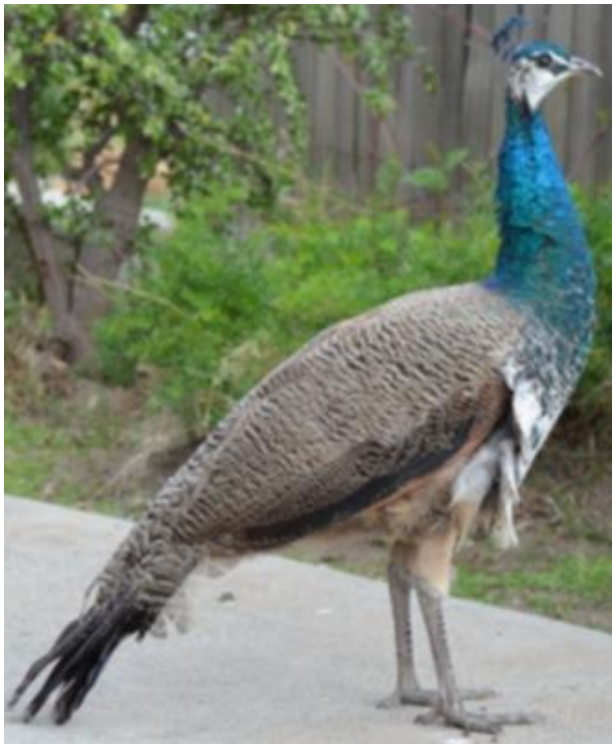
Lecture 08-09

Computational Methods in Proteins



Aligning / Matching



Matching / Alignment

- ***Longest common substring problem***

  Let Σ be an alphabet (finite set; for *DNA alphabet* (Σ = {A,C,G,T})). Search a pattern from the text where both the pattern and text are arrays of elements of Σ.

  – *Example:*
    - *Pattern: ins, india, iit, iit kharagpur*
    - *String: indian institute of technology kharagpur*

- ***Longest common subsequence problem***

  Let Σ be an alphabet (finite set; for *DNA alphabet* (Σ = {A,C,G,T})). Find the longest subsequence common to all sequences in a set of sequences (often just two sequences) constructed over alphabet set Σ.

  – *Example:*
    - *Pattern: ins, india, iit, iit kharagpur*
    - *String: indian institute of technology kharagpur*

Longest common subsequence problem

Problem Definition: We are given two strings: string *S* of length *n*, and string *T* of length *m*. Our goal is to produce their longest common subsequence: the longest sequence of characters that appear left-to-right (but not necessarily in a contiguous block) in both strings.

Example,

S = ABAZDC

T = BACBAD

- *Dynamic Programming*
  1. Initialization
  2. Matrix fill (scoring)
  3. Traceback (alignment)

Longest common subsequence problem

Dynamic Programming

Example,

S = ABAZDC

T = BACBAD

Initialization

		B	A	C	B	A	D
		0	0	0	0	0	0
A	0						
B	0						
A	0						
Z	0						
D	0						
C	0						

Longest common subsequence problem

Dynamic Programming

Example,

S = ABAZDC

T = BACBAD

$$M_{i,j} = \text{MAXIMUM} [\, M_{i-1,\, j-1} + S \,,\, M_{i-1,j} \,,\, M_{i,j-1} \,]$$

		B	A	C	B	A	D
		0	0	0	0	0	0
A	0	0	1	1	1	1	1
B	0	1	1	1	2	2	2
A	0	1	2	2	2	3	3
Z	0	1	2	2	2	3	3
D	0	1	2	2	2	3	4
C	0	1	2	3	3	3	4

Initialization
Scoring

Longest common subsequence problem

Dynamic Programming

Example,

S = ABAZDC

T = BACBAD

$M_{i,j}$ = MAXIMUM [$M_{i-1,\,j-1} + S$, $M_{i-1,j}$, $M_{i,j-1}$]

Solution???

		B	A	C	B	A	D
		0	0	0	0	0	0
A	0	0	1	1	1	1	1
B	0	1	1	1	2	2	2
A	0	1	2	2	2	3	3
Z	0	1	2	2	2	3	3
D	0	1	2	2	2	3	4
C	0	1	2	3	3	3	4

Initialization
Scoring
Alignment

Multiple Possibilities

Longest common subsequence problem

Dynamic Programming

Example,

S = ABAZDC

T = BACBAD

$$M_{i,j} = \text{MAXIMUM} [\, M_{i-1,\, j-1} + S \,,\, M_{i-1,j} \,,\, M_{i,j-1} \,]$$

		B	A	C	B	A	D
		0	0	0	0	0	0
A	0	0	1	1	1	1	1
B	0	1	1	1	2	2	2
A	0	1	2	2	2	3	3
Z	0	1	2	2	2	3	3
D	0	1	2	2	2	3	4
C	0	1	2	3	3	3	4

Implement

Longest common subsequence problem

Dynamic Programming

Example,

S = ABAZDC

T = BACBAD

Space complexity: $O(N^2)$
Time complexity: $O(N^2)$

		B	**A**	**C**	**B**	**A**	**D**
		0	0	0	0	0	0
A	0	0	1	1	1	1	1
B	0	1	1	1	2	2	2
A	0	1	2	2	2	3	3
Z	0	1	2	2	2	3	3
D	0	1	2	2	2	3	4
C	0	1	2	3	3	3	4

Sequence Alignment

- Pairwise
  - DOT matrix
  - Dynamic programming
  - Word method (efficient heuristic method; e.g., BLAST)
- Multiple
  - Dynamic programming
  - Progressive method (e.g., CLUSTAL, T-Coffee)
  - Iterative
  - Motif finding

Dynamic Programming

S = BACBAD

T = ABAZDC

Gap Penalty:

Opening

Extension

Combined

Initialization

Matrix Fill

Score function

$M_{i,j} = Max\ [\ M_{i-1,j-1} + S_{i,j}\ ;\ M_{i-1,j}\ ;\ M_{i,j-1}]$

Traceback

$Tb_{i,j} = \{\ \nwarrow ; \uparrow\ ;\ \leftarrow\}$

Alignment Score Function

$$W(k) = c_{open} + c_{length} * k$$

$$M_{i,j} = \text{MAXIMUM} [$$
$$M_{i-1,\,j-1} + S_{i,j} ,$$
$$M_{i-1,j-k} + w(k) \ (k = 1, \ldots, j-1),$$
$$M_{i-k,j-1} + w(k) \ (k = 1, \ldots, i-1)]$$

$$M_{i,j} = \text{MAXIMUM} [$$
$$0$$
$$M_{i-1,\,j-1} + S(a_i, b_j) ,$$
$$M_{i-1,j} + w(a_i, -),$$
$$M_{i-k,j-1} + w(-, b_j)]$$

Pairwise Sequence Alignment

- Problem Statement
  - Input: Two sequences.

```
>Seq1|Dummy|SEQUENCE
GAATTCAGTTA

>Seq2|Dummy|SEQUENCE
GGATCGA
```

  - Output: Their alignment subject to optimum alignment score.

```
G _ A A T T C A G T T A
|     |   | |   |     |
G G _ A _ T C _ G _ _ A
```

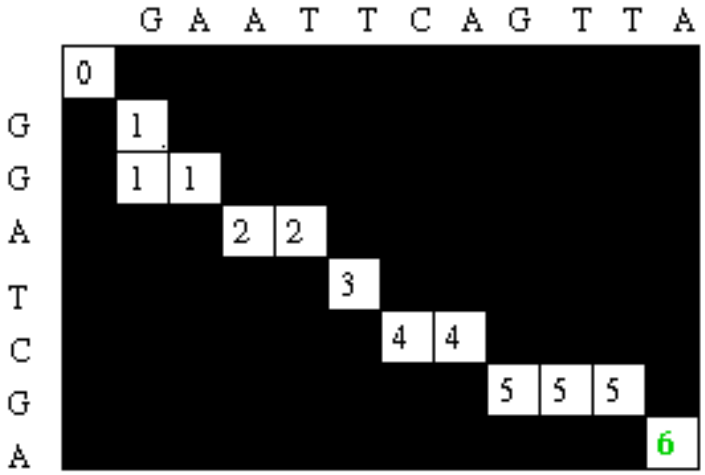
Pairwise Sequence Alignment

GAATTCAGTTA

GGATCGA

		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											

```
G _ A A T T C A G T T A
|   |   | |   |     |
G G _ A _ T C _ G _ _ A
```



M(i,j) = MAXIMUM [

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

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

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

Workout

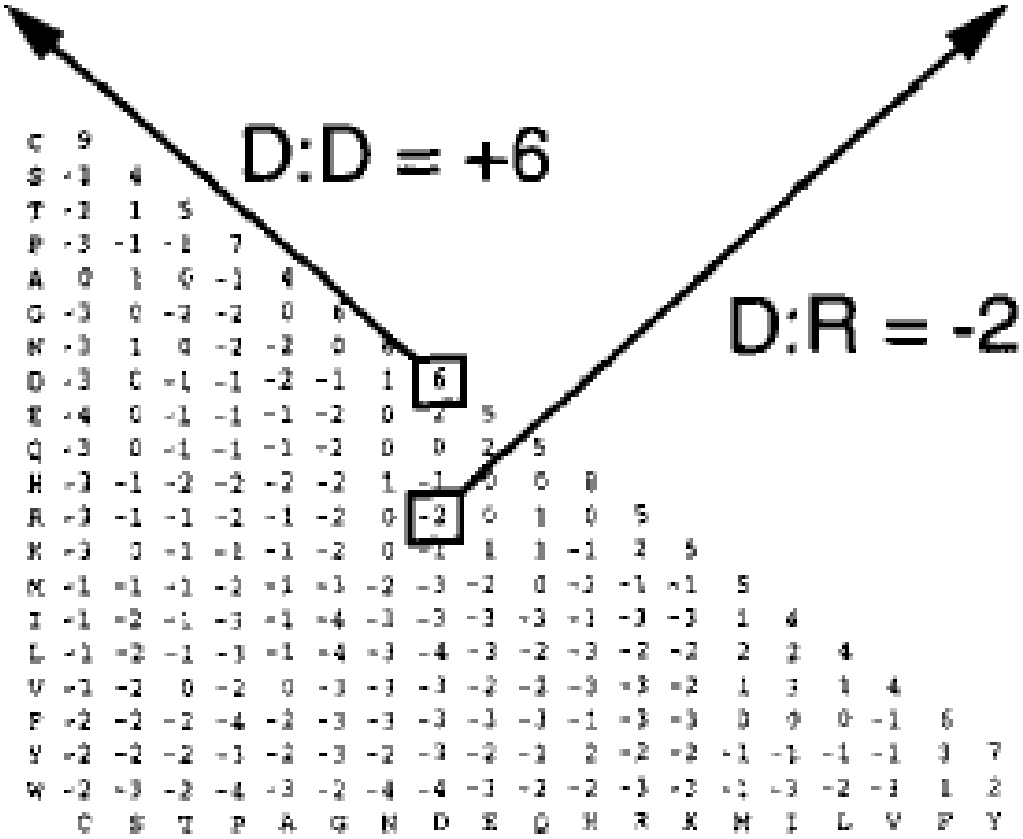
Source: http://www.avatar.se/molbioinfo2001/dynprog/dynamic.html

Workout

- Implement Dynamic programming for pairwise sequence alignment.

Pairwise alignment

```
METR:   134 LQQGELDLVMTSDILPRSELHYSPMFDFEVRLVLAPDHPLASKTQITPEDLASETLLI
                |     |||           |         |          |||||     |     ||  ||
RBCR:   137 LDSNSVDLVLMGVPPRNVEVEAEAFMDNPLVVIAPPDHPLAGERAISLARLAEETFVM
```



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

Substitution matrix

Substitution Matrix

BLOSUM62

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	-1	5																		
Asn	-2	0	6																	
Asp	-2	-2	1	6																
Cys	0	-3	-3	-3	9															
Gln	-1	1	0	0	-3	5														
Glu	-1	0	0	2	-4	2	5													
Gly	0	-2	0	-1	-3	-2	-2	6												
His	-2	0	1	-1	-3	0	0	-2	8											
Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

(**BLO**cks **SU**bstitution **M**atrix)

$$S_{ij} = \left(\frac{1}{\lambda}\right) \log\left(\frac{p_{ij}}{q_i \times qj}\right)$$

p_{ij} is the probability of two amino acids *i* and *j* replacing each other in a homologous sequence,

q_i and q_j are the background probabilities of finding the amino acids *i* and *j* in any protein sequence,

λ is a scaling factor.

(**BLO**cks **SU**bstitution **M**atrix)

Henikoff, S.; Henikoff, J.G. (1992). Amino Acid Substitution Matrices from Protein Blocks. *PNAS* **89** (22):10915-10919.

Eddy, S. R. (2004) Where did the BLOSUM62 alignment score matrix come from? *Nature Biotechnology* **22**:1035-1036.

Styczynski, M. P.; Jensen, K. L.; Rigoutsos, I.; Stephanopoulos, G. (2008). BLOSUM62 miscalculations improve search performance**.** *Nature Biotechnology* **26**:274 - 275.

(**BLO**cks **SU**bstitution **M**atrix)

- BLOSUM matrices with high numbers are designed for comparing closely related sequences, while those with low numbers are designed for comparing distant related sequences.

- BLOSUM80 is used for less divergent alignments, and BLOSUM45 is used for more divergent alignments.

- The matrices were created by merging (clustering) all sequences that were more similar than a given percentage into one single sequence and then comparing those sequences (that were all more divergent than the given percentage value) only; thus reducing the contribution of closely related sequences.

- The percentage used was appended to the name, giving BLOSUM80 for example where sequences that were more than 80% identical were clustered.

Point **A**ccepted **M**utation

- PAM is the replacement of a single amino acid in the primary structure of a protein with another single amino acid, which is accepted by the processes of natural selection.

- In particular, silent mutations are not point accepted mutations, nor are mutations which are lethal or which are rejected by natural selection in other ways.

- The calculation of PAM substitution matrix were based on 1572 observed mutations in the phylogenetic trees of 71 families of closely related proteins. The proteins to be studied were selected on the basis of having high similarity with their predecessors. The protein alignments included were required to display at least 85% identity.

Dissimilar matrices can provide comparable performance with optimized gap penalties

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
	1	-1	-1	-1	-1	-1	0	-1	0	-1	-2	-2	0	-1	-1	0	-1	-1	-3	-4	C
		3	0	-1	0	0	0	0	-1	0	-1	-1	0	-1	-1	-1	-1	0	0	-1	S
C	13		3	-1	-1	-1	0	-1	-1	-1	-2	-1	-1	0	0	0	0	0	0	1	T
S	-1	5		2	-1	0	-1	0	-1	-1	-1	-2	0	-1	0	-2	-1	0	0	1	P
T	-1	2	5		3	0	-1	-2	-1	-1	-1	-1	-1	0	0	-1	0	-1	0	1	A
P	-4	-1	-1	10		1	0	-1	-2	-1	-1	-2	-1	1	0	0	-1	1	1	1	G
A	-1	1	0	-1	5		3	0	-1	-1	0	-1	-1	0	0	-1	-1	-1	-1	0	N
G	-3	0	-2	-2	0	8		3	-1	-1	-1	-2	-1	-1	0	0	-1	-1	0	0	D
N	-2	1	0	-2	-1	0	7		2	0	0	0	0	0	-1	0	-1	1	1	1	E
D	-4	0	-1	-1	-2	-1	2	8		4	0	-1	0	1	-1	0	-1	-1	1	2	Q
E	-3	-1	-1	-1	-1	-3	0	2	6		4	-1	-1	0	-2	-1	-2	-1	0	-2	H
Q	-3	0	-1	-1	-1	-2	0	0	2	7		2	0	0	-2	-1	-1	0	1	-1	R
H	-3	-1	-2	-2	-2	-2	1	-1	0	1	10		3	-1	-1	-1	-1	-1	0	1	K
R	-4	-1	-1	-3	-2	-3	-1	-2	0	1	0	7		3	0	0	-1	-2	0	0	M
K	-3	0	-1	-1	-1	-2	0	-1	1	2	0	3	6		1	-1	1	-1	0	-1	I
M	-2	-2	-1	-3	-1	-3	-2	-4	-2	0	-1	-2	-2	7		1	-1	-1	-1	-1	L
I	-2	-3	-1	-3	-1	-4	-3	-4	-4	-3	-4	-4	-3	2	5		2	-1	0	0	V
L	-2	-3	-1	-4	-2	-4	-4	-4	-3	-2	-3	-3	-3	3	2	5		1	-1	-3	F
V	-1	-2	0	-3	0	-4	-3	-4	-3	-3	-4	-3	-3	1	4	1	5		0	-2	Y
F	-2	-3	-2	-4	-3	-4	-4	-5	-3	-4	-1	-3	-4	0	0	1	-1	8		1	W
Y	-3	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-1	-2	0	-1	-1	-1	4	8		
W	-5	-4	-3	-4	-3	-3	-4	-5	-3	-1	-3	-3	-3	-1	-3	-2	-3	1	2	15	
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Other variations of scoring matrices

- Position Specific Scoring Matrix
- Structural information
- 3D-1D mapping

Global Pairwise Sequence Alignment

- Needleman-Wunch

$$W(k) = c_{open} + c_{length} * k$$

$$M(i,j) = \text{MAXIMUM} [$$

$$M_{i-1,\,j-1} + S_{i,j} \text{ (substitution matrix)},$$

$$M_{i-1,j-k} + w(k) \text{ (k = 1, …, j-1)},$$

$$M_{i-k,j-1} + w(k) \text{ (k = 1, …, i-1) }]$$

Local Pairwise Sequence Alignment

- Smith-Waterman

$$W(k) = c_{open} + c_{length} * k$$

$$M(i,j) = \text{MAXIMUM}\ [$$
$$0$$
$$M_{i-1,\,j-1} + S(a_i, b_j) \text{ (match/mismatch)},$$
$$M_{i-1,j} + w(a_i, -),$$
$$M_{i-k,j-1} + w(-, b_j)\ ]$$