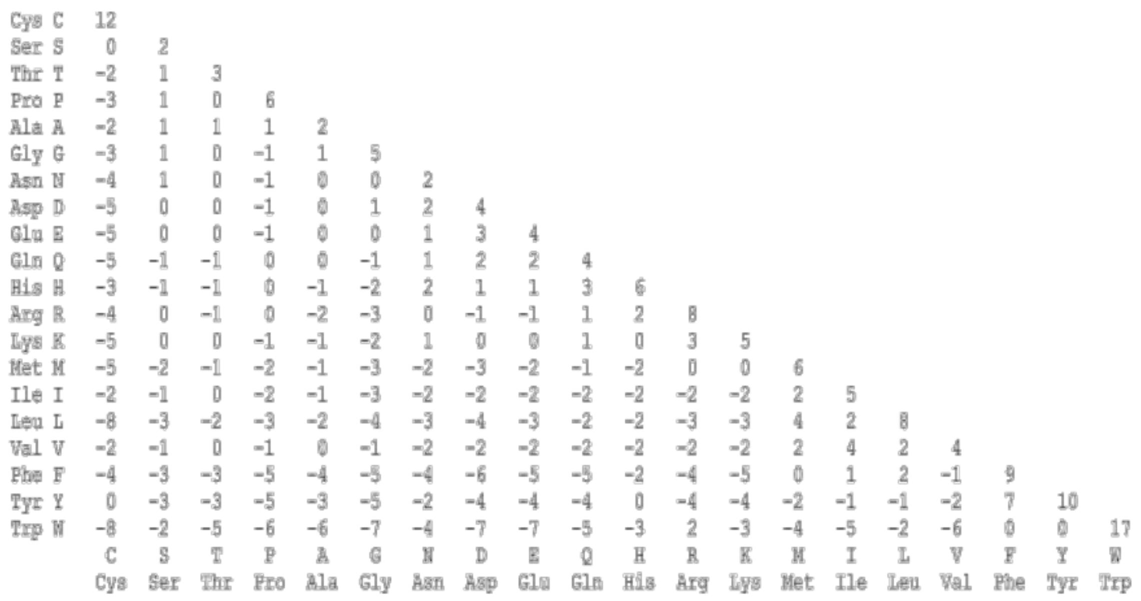
Exercises on pairwise alignment

1 Introduction

In this exercise you will have a closer look at the affine gap global alignment algorithm. Before you start this exercise you should have watched the course recordings on alignment algorithms.

In the second part of this exercise, you can perform a local alignment manually. Nothing new is introduced in this part of the exercise, but it can help you practice for the exam, or to gain a better understanding of how and why these dynamic programming algorithms work.



Cys C	12																			
Ser S	0	2																		
Thr T	-2	1	3																	
Pro P	-3	1	0	6																
Ala A	-2	1	1	1	2															
Gly G	-3	1	0	-1	1	5														
Asn N	-4	1	0	-1	0	0	2													
Asp D	-5	0	0	-1	0	1	2	4												
Glu E	-5	0	0	-1	0	0	1	3	4											
Gln Q	-5	-1	-1	0	0	-1	1	2	2	4										
His H	-3	-1	-1	0	-1	-2	2	1	1	3	6									
Arg R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	8								
Lys K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5							
Met M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6						
Ile I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5					
Leu L	-8	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	8				
Val V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4			
Phe F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9		
Tyr Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10	
Trp 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
	Cys	Ser	Thr	Pro	Ala	Gly	Asn	Asp	Glu	Gln	His	Arg	Lys	Met	Ile	Leu	Val	Phe	Tyr	Trp

Figure 1: Scoring matrix for the simple affine gap alignment

2 Affine Gap

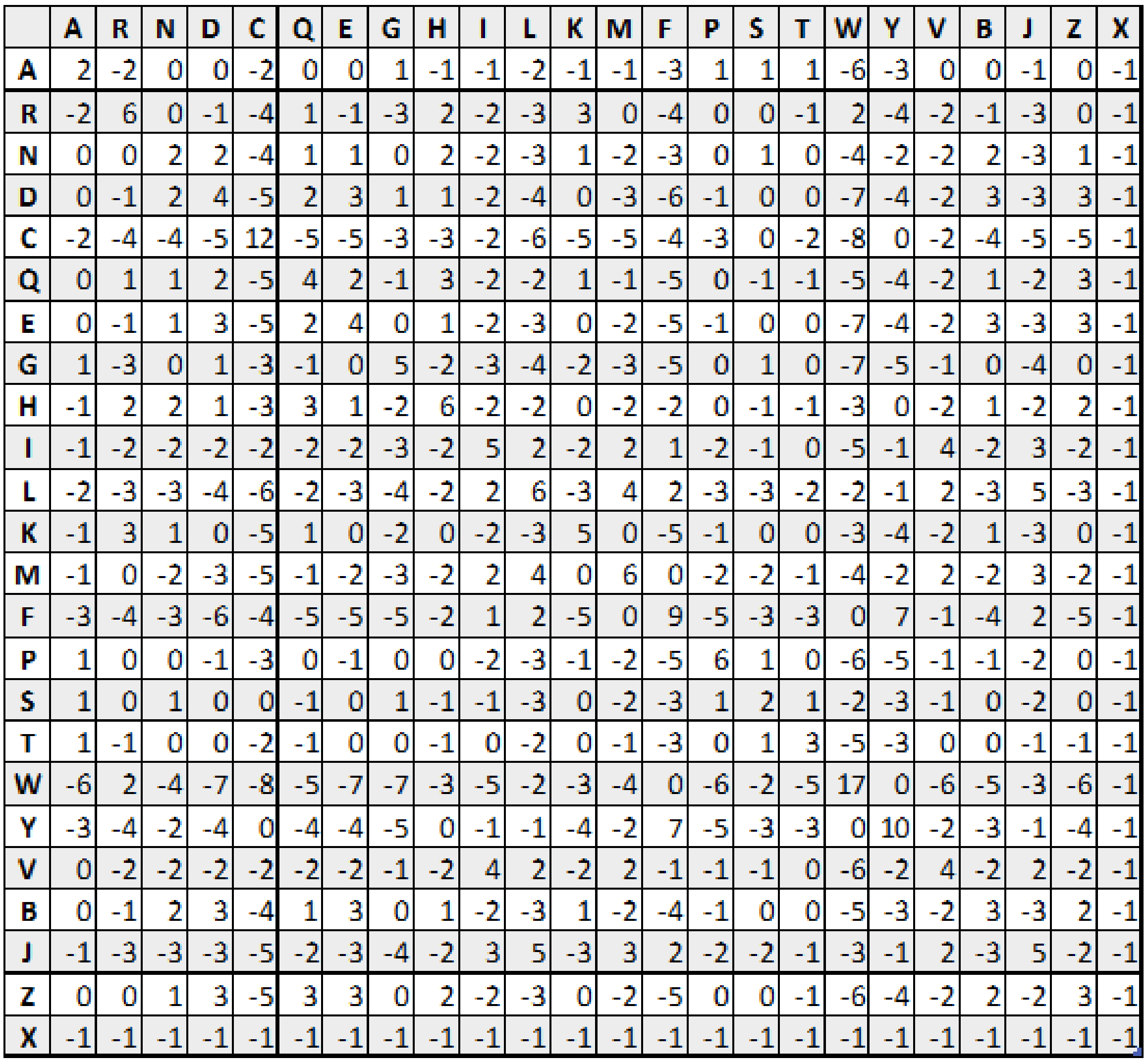
2.1 A simple example

In this simple example of an affine gap alignment, we will align the sequence LSR to itself, with the substitution scores as in fig. 1, a gap open penalty of -12, and a gap extension penalty of -4. You can watch the recording of this example in "completing_alignment_matrix.mp4" (17 minutes). Make sure you understand how we fill in this matrix. You can use the file "Simple_LSR_example.doc" to complete this matrix yourself.

2.2 A more complex example

Now open the powerpoint presentation "3_pairwisealignment_ppt_student_part2_DB.pptx" and look at the alignment matrix in slides 14 and further. In these slides, we align two protein sequences MNALSDRT and MGSDRTTET, with PAM250 substitution scores (fig. 2), a gap open penalty of -12, and a gap extension penalty of -4. Listen to the voice recording and try to understand the backtracking in this matrix.

During this alignment we had to construct two copies of the dynamic programming matrix, because we had two different ways to optimally align the prefixes MNA and MGSD. Continuing from a partial optimal alignment that ends in a gap versus continuing from one that ends in a (mis)match, will result in different values when using an affine gap penalty. The continuation from a (mis)match will open a new gap (-12), while the continuation from a gap can extend a gap (-4) or open a new gap in the other sequence (-12).



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

Figure 2: PAM250

However, this means that we potentially have to duplicate our dynamic programming matrix multiple times. When dealing with large protein or DNA sequences, a single dynamic programming matrix can already contain thousands or even millions of cells. Duplicating such a matrix several times is not desirable. The approach we used in the slides is therefore not used in practice. Instead a more compact approach, with limited memory and CPU usage is used.

2.3 Affine Gap Algorithm

We will now discuss the gapped alignment. Denote with $s(v, w)$ the (mis)match score between characters v and w, with *gapOpen* the gap open penalty, and with *gapExtend* the gap extension penalty. The gapped alignment algorithm between two sequences V and W then proceeds as follows:

- Instead of 1 dynamic programming matrix that stores the score of the optimal alignment of two prefixes, store 3 dynamic programming matrices:
  - M, a matrix for the best score of an alignment ending in a (mis)match. M is initialized as follows: $M[0, 0] = 0$, and the remainder of the first row and column are $-\infty$.
  - X, a matrix that stores the best score of alignments ending in a gap in the first (top) sequence V. X is initialized with the appropriate gap costs in the first column, and the first row is filled with $-\infty$.
  - Y, a matrix that stores the best score of alignments ending in a gap in the second (left) sequence W. Y is initialized with the appropriate gap costs in the first row, and the first column is filled with $-\infty$.

  The scores with value $-\infty$ indicate an invalid position in the matrix, these will never contribute to the final score.

- The score in $M[i, j]$ is the maximum of
  - $M[i-1, j-1] + s(W[i-1], V[j-1])$,
  - $X[i-1, j-1] + s(W[i-1], V[j-1])$, and
  - $Y[i-1, j-1] + s(W[i-1], V[j-1])$.

  In all three cases this is a simple extension of the previous alignment, with a (mis)match.

- The score in $X[i, j]$ is the maximum of
  - $M[i-1, j] + gapOpen$,
  - $X[i-1, j] + gapExtend$, and
  - $Y[i-1, j] + gapOpen$.

  Here we open a new gap when coming from the M matrix or the Y matrix, but we extend the already existing gap in the X matrix.

- The score in $Y[i, j]$ is the maximum of
  - $M[i, j-1] + gapOpen$,
  - $X[i, j-1] + gapOpen$, and
  - $Y[i, j-1] + gapExtend$.

Here we open a new gap when coming from the M matrix or the X matrix, but we extend the already existing gap in the Y matrix.

- The final score is the maximum of the final scores in M, X, and Y.
- The traceback proceeds by appropriately jumping between the three matrices. As usual, it can be very convenient to store the traceback information while filling the dynamic programming matrices.

2.4 Example

When applying the above algorithm to the sequences MNALSDRT and MGSDRTTET, with PAM250 substitution scores, a gap open penalty of -12, and a gap extension penalty of -4, we get the following three matrices:

Table 1: Dynamic programming matrix M for scores of alignments ending in a (mis)match.

		M	N	A	L	S	D	R	T
	0	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
M	$-\infty$	6	-14	-17	-16	-26	-31	-32	-37
G	$-\infty$	-15	6	-5	-14	-13	-17	-25	-26
S	$-\infty$	-18	-5	7	-8	-8	-13	-17	-21
D	$-\infty$	-23	-8	-5	3	-5	-4	-14	-17
R	$-\infty$	-24	-14	-10	-8	3	-6	2	-15
T	$-\infty$	-29	-18	-13	-11	-7	3	-7	5
T	$-\infty$	-33	-22	-17	-15	-10	-7	2	-4
E	$-\infty$	-38	-25	-22	-20	-15	-7	-8	2
T	$-\infty$	-41	-30	-24	-23	-19	-15	-8	-5

Table 2: Dynamic programming matrix X for scores of alignments ending in a gap in the first sequence.

		M	N	A	L	S	D	R	T
	0	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
M	-12	-24	-28	-32	-36	-40	-44	-48	-52
G	-16	-6	-18	-22	-26	-30	-34	-38	-42
S	-20	-10	-6	-17	-22	-25	-29	-34	-38
D	-24	-14	-10	-5	-17	-20	-25	-29	-33
R	-28	-18	-14	-9	-9	-17	-16	-26	-29
T	-32	-22	-18	-13	-13	-9	-18	-10	-22
T	-36	-26	-22	-17	-17	-13	-9	-14	-7
E	-40	-30	-26	-21	-21	-17	-13	-10	-11
T	-44	-34	-30	-25	-25	-21	-17	-14	-10

Table 3: Dynamic programming matrix Y for scores of alignments ending in a gap in the second sequence.

		M	N	A	L	S	D	R	T
	0	-12	-16	-20	-24	-28	-32	-36	-40
M	$-\infty$	-24	-6	-10	-14	-18	-22	-26	-30
G	$-\infty$	-28	-18	-6	-10	-14	-18	-22	-26
S	$-\infty$	-32	-22	-17	-5	-9	-13	-17	-21
D	$-\infty$	-36	-26	-20	-17	-9	-13	-16	-20
R	$-\infty$	-40	-30	-26	-21	-20	-9	-13	-10
T	$-\infty$	-44	-34	-30	-25	-23	-19	-9	-13
T	$-\infty$	-48	-38	-34	-29	-27	-22	-19	-10
E	$-\infty$	-52	-42	-37	-33	-32	-27	-19	-20
T	$-\infty$	-56	-46	-42	-36	-35	-31	-27	-20

2.5 Questions

1. Use the three matrices above to find the optimal alignment. Do you find the same result(s) as in the slides from section 2.2?
2. Now that you have performed a traceback manually, how would you store the traceback information to make traceback easier?
3. How would you adapt this algorithm such that gaps longer than 3 positions get a fixed gap penalty of -20?
   (a) Approach 1: for each score in X and Y, keep track of the number of gaps at the end of the alignment that lead to that score. What potential problems do you see with this approach?
   (b) Approach 2: add additional matrices for alignments that end in a *specific* number of gaps in the first or in the second sequence (cf. X and Y, these contain scores for alignments that end in a *non-zero* number of gaps). Give a high level description of which matrices you would define, and of the recursions that relate them to eachother.

3 Smith-Waterman

In this example you will align two sequences:

- FTFIR, and
- FFTR.

You will use the Smith-Waterman algorithm for local alignment with the following scores: (mis)matches are scored with BLOSUM62, and there is a fixed gap penalty -5. The empty alignment matrix is given in table 4.

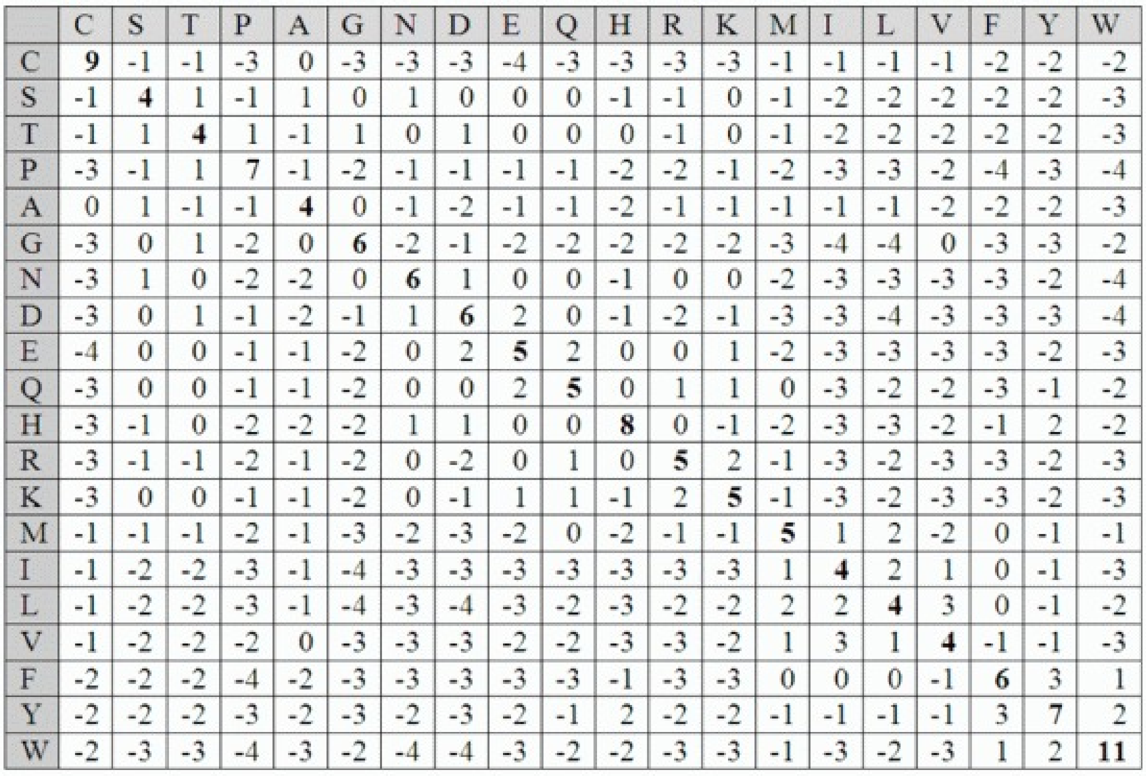
In each alignment cell we will write 4 values, as follows:

- the match score (topleft: match score + score in $[i-1, j-1]$),
- the two gap scores (topright: gap penalty + score in cell $[i-1, j]$, bottomleft: gap penalty + score in cell $[i, j-1]$), and
- the actual entry of the DP matrix (bottomright: max of the three auxilliary values).

The auxilliary values are only here for the sake of clarity in this exercise, the actual DP matrix is just the values that we write in the bottomright of each cell. Table 1 shows the (empty) DP matrix with additional room for the auxilliary values. The initial gap scores have already been filled in.

3.1 Questions

1. Fill in the dynamic programming matrix with the Smith-Waterman algorithm, using BLOSUM62 scoring matrix (fig. 3) and a fixed gap penalty of -5.
2. Determine the optimal score.
3. Perform a traceback, which optimal alignments do you find?



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	**9**	-1	-1	-3	0	-3	-3	-3	-4	-3	-3	-3	-3	-1	-1	-1	-1	-2	-2	-2
S	-1	**4**	1	-1	1	0	1	0	0	0	-1	-1	0	-1	-2	-2	-2	-2	-2	-3
T	-1	1	**4**	1	-1	1	0	1	0	0	0	-1	0	-1	-2	-2	-2	-2	-2	-3
P	-3	-1	1	**7**	-1	-2	-1	-1	-1	-1	-2	-2	-1	-2	-3	-3	-2	-4	-3	-4
A	0	1	-1	-1	**4**	0	-1	-2	-1	-1	-2	-1	-1	-1	-1	-1	-2	-2	-2	-3
G	-3	0	1	-2	0	**6**	-2	-1	-2	-2	-2	-2	-2	-3	-4	-4	0	-3	-3	-2
N	-3	1	0	-2	-2	0	**6**	1	0	0	-1	0	0	-2	-3	-3	-3	-3	-2	-4
D	-3	0	1	-1	-2	-1	1	**6**	2	0	-1	-2	-1	-3	-3	-4	-3	-3	-3	-4
E	-4	0	0	-1	-1	-2	0	2	**5**	2	0	0	1	-2	-3	-3	-3	-3	-2	-3
Q	-3	0	0	-1	-1	-2	0	0	2	**5**	0	1	1	0	-3	-2	-2	-3	-1	-2
H	-3	-1	0	-2	-2	-2	1	1	0	0	**8**	0	-1	-2	-3	-3	-2	-1	2	-2
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	**5**	2	-1	-3	-2	-3	-3	-2	-3
K	-3	0	0	-1	-1	-2	0	-1	1	1	-1	2	**5**	-1	-3	-2	-3	-3	-2	-3
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	**5**	1	2	-2	0	-1	-1
I	-1	-2	-2	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	**4**	2	1	0	-1	-3
L	-1	-2	-2	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	**4**	3	0	-1	-2
V	-1	-2	-2	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	**4**	-1	-1	-3
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	**6**	3	1
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	**7**	2
W	-2	-3	-3	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	**11**

Figure 3: BLOSUM62

Table 4: Dynamic programming matrix with auxilliary tables.

		F		T		F		I		R	
		0	0	0	0	0	0	0	0	0	0
F	0										
	0										
F	0										
	0										
T	0										
	0										
R	0										
	0										