



Pairwise Alignment, Multiple Alignment, and BLAST

Henrik Christensen and John Elmerdahl Olsen

- 4.1 The Pairwise Alignment Problem – 52**
 - 4.1.1 Global or Local Pairwise Alignments? – 53
 - 4.1.2 Substitution Matrices – 54
 - 4.1.3 Gaps – 57
 - 4.1.4 Dynamic Programming – 58
- 4.2 Multiple Alignment – 64**
 - 4.2.1 Clustal – 65
 - 4.2.2 Other Multiple Alignment Programs – 68
- 4.3 BLAST – 70**
 - 4.3.1 NCBI BLAST – 71
 - 4.3.2 Ortholog Detection – 72
 - 4.3.3 BLAST2 Sequences – 72
 - 4.3.4 Statistics – 74
 - 4.3.5 Variants of BLAST – 75
- 4.4 Activities – 76**
 - 4.4.1 Pairwise Alignment – 76
 - 4.4.2 Learn How Dynamic Programming Works with Pairwise Alignments – 77
 - 4.4.3 Multiple Alignment with ClustalX – 77
 - 4.4.4 BLAST – 78
- References – 79**

What You Will Learn in This Chapter

Pairwise alignments and multiple alignments are the basic tools to compare sequences. BLAST is the most frequently used bioinformatics program to compare your own sequence (query sequence) to all sequences in a database (subject sequences) based on local pairwise alignments. The outcome of the BLAST analysis provides qualitative information about homologous sequences and can quantify the identity of the query sequence to the sequences in the database. You will learn about multiple alignments and how to construct them. Multiple alignments are used for a range of applications described in the other chapters of the book.

4.1 The Pairwise Alignment Problem

A pairwise alignment is a model of the homology between two sequences considering all nucleotides or amino acids and all deletion and insertions.

Two sequences to be compared could, for instance, be the ones below:

- GCAGTAGCATGACGATAG
- GCGGTAGCATGATAC

A pairwise alignment requires a model for the evolutionary steps that led to the differences observed. First, we should test if they are homologous. Such a test can be done by BLAST (see ► Sect. 4.3), and it will tell if the sequences are from the same gene. Once this has been established, we need to model evolution to construct the pairwise alignment. The simplest assumptions are that identical nucleotide pairs and that identical amino acids pair and, further, that different amino acids with similar physiochemical properties form pairs.

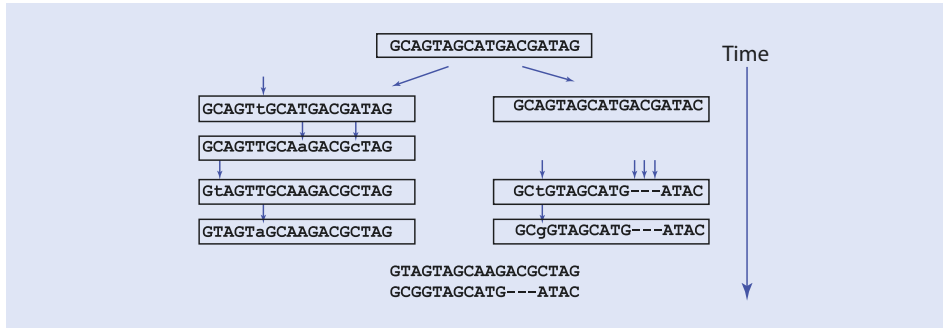
For most comparisons it is preferred to base pairwise alignments on the amino acid sequences especially if the sequences are very divergent. Pairwise nucleotide comparisons are preferred for closely related sequences only.

Gaps are used to show that an amino acid or nucleotide is without a match in the other sequence and the gaps represent insertions or deletions in an evolutionary context. Most evolutionary modeling assumes that it is more difficult to form gaps than to form mismatches and that it is more difficult to form a gap than to extend one already formed. Scoring systems are used to separate matches (high score) from mismatches (low score) and gaps (very low score). For proteins, scoring matrices (see ► Sect. 4.1.2.1) are used. At the end, scores between amino acids or nucleotides are summarized over the whole pairwise alignment, and penalties for gaps are subtracted. The result is a unit score for the pairwise alignment given the parameters used. Given the same sequences compared, the pairwise alignment with the highest total score is preferred compared to one with a lower score.

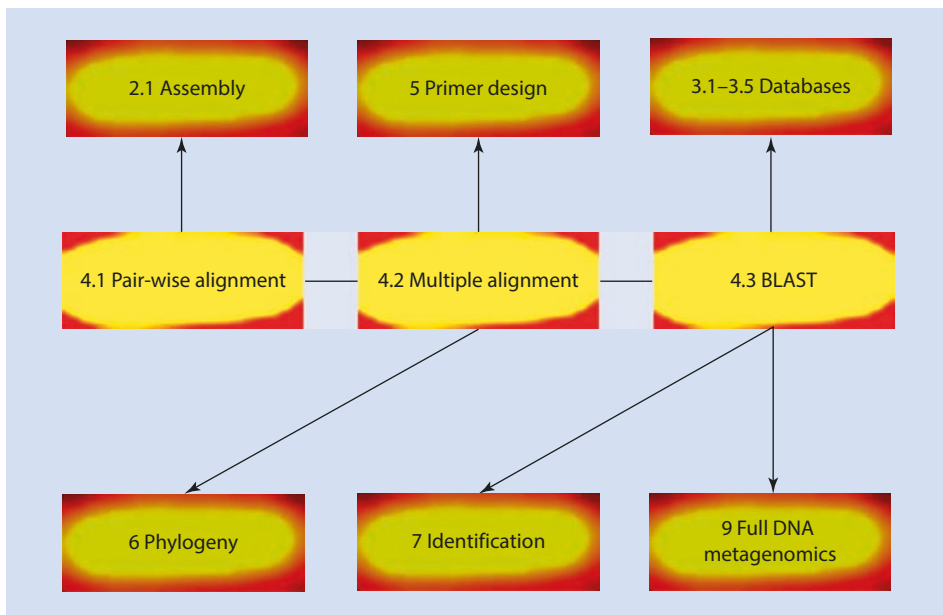
When the pairwise alignment has been constructed, the identity and similarity can be quantified. The identity is the number of nucleotides or amino acids matching in two sequences compared at all positions between the two sequences. Similarity is a further comparison also considering different types of nucleotides or amino acids as well as the gaps.

An alignment of the sequences above, and a possible scenario leading to the differences between them, is shown in ■ Fig. 4.1.

The principles of pairwise alignments are used for most multiple alignment programs and for BLAST. The three tools are fundamental to carry out most bioinformatics (■ Fig. 4.2).



■ **Fig. 4.1** Hypothetical example of a pairwise alignment of two sequences if their evolution was known. The first sequence is the ancestor sequence which then diverged into two new lineages. In the sequence to the left, five-point mutations occurred over time, and in the one to the right, two-point mutations plus a deletion of three nucleotides happened probably resulting in the loss of an amino acid. With the exact knowledge of the position of the gap, we can construct the pairwise alignment at the bottom without any assumptions needed

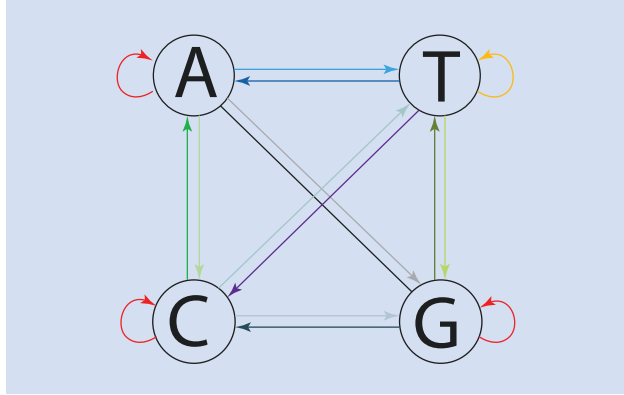


■ **Fig. 4.2** Pairwise and multiple alignments as well as BLAST introduced in this chapter provide the background for most other chapters in this book

4.1.1 Global or Local Pairwise Alignments?

Before a pairwise alignment is constructed, it needs to be decided if it should be global or local. A pairwise alignment is global if it is known that the sequences are homologous in their full length. In this situation, it is sound to align both sequences in their full length. If preliminary evidence has shown that the genes are encoding for the same proteins and the full length of the gene has been sequenced, a global alignment can be constructed.

Fig. 4.3 Markov substitution model of nucleotides. The consequence of the Markov model is that it only depends on the observed nucleotide or amino acid and not on the past changes. The arrows with different colors indicate that potentially 16 different probabilities exist for changes (unrestricted model on [Table 4.3](#)). (Modified from Durbin et al. 1999)



A local alignment is needed if it is known that one sequence is shorter than the other and that it cannot be related to the other in its full length. This can happen if the DNA sequence of one gene has not been determined in the full length or if the domain structure of the proteins that the genes encode for are rather divergent.

4.1.2 Substitution Matrices

Substitution matrices are used to model the probability of mutual amino acid or nucleotide substitutions in sequences. Amino acids or nucleotides with a lower probability of changing are given more weight in the comparisons compared to those with a higher probability of changing. The changes are assumed to occur according to a Markov model where changes only depend on the current nucleotide or amino acid observed and not on past changes ([Fig. 4.3](#)).

4.1.2.1 Amino Acid Substitution Matrices

Amino acids have different biochemical and physical properties that influence their relative replaceability during evolution ([Table 4.1](#)). The probability of replacement has been related to the physicochemical properties of the amino acids in the way that hydrophobic amino acids are easier replaced by other hydrophobic amino acids again compared to other types such as charged amino acids. The substitution matrices reflect these properties as seen from [Fig. 4.4](#). Here the score between the two hydrophobic amino acids isoleucine and histidine is 5, whereas the score between isoleucine and the positive charged arginine is only -4 . Some amino acids belong to more groups, for instance, histidine which is hydrophobic, positively charged, polar, and aromatic.

The PAM (percent accepted mutations) matrices are based on global alignments of closely related proteins. The number at the end of a PAM matrix refers to the number of steps required for a given percent change. PAM1 corresponds to 99% identity (1% change) between the aligned sequences. This means that PAM112 will recognize more distantly related sequences (40% identity) than PAM23 (80% identity). PAM matrices are based on global alignments and therefore best suited for global pairwise alignments used in evolutionary studies ([Table 4.2](#)).

BLOSUM (blocks substitution matrix) matrices are based on local alignments. The number at the end of a BLOSUM matrix refers to the minimum percent identity allowed

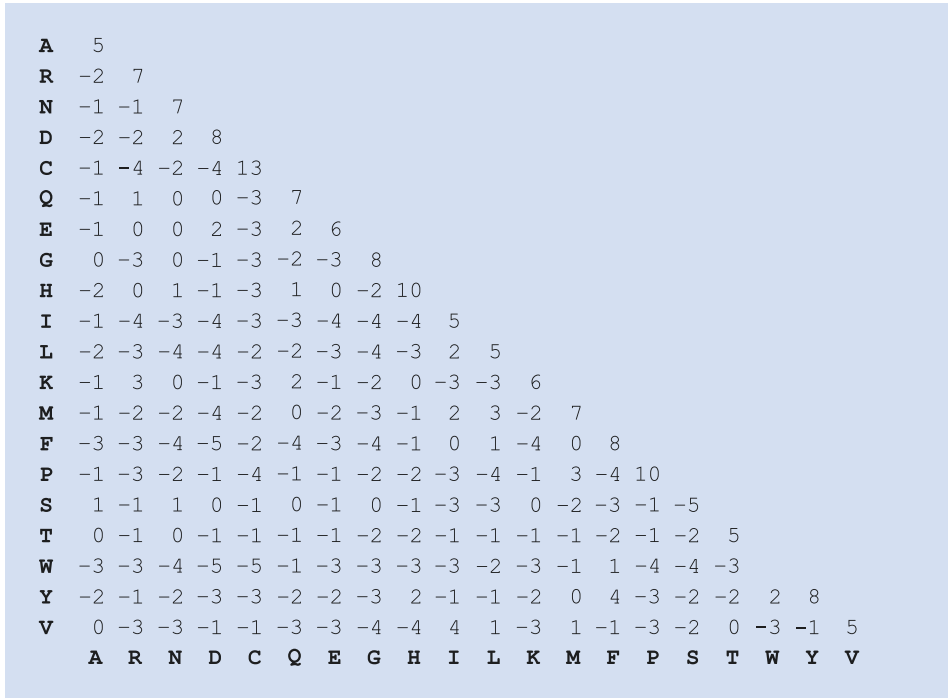
Table 4.1 Physiochemical properties of amino acids

Amino acid	Hydro-phobic	Positive	Negative	Polar	Charged	Small	Tiny	Ali-phatic	Aro-matic
Ile	+							+	
Leu	+							+	
Val	+					+		+	
Cys	+					+			
Ala	+					+	+		
Gly	+					+	+		
Met	+								
Phe	+								+
Tyr	+			+					+
Trp	+			+					+
His	+	+		+	+				+
Lys		+		+	+				
Arg		+		+	+				
Glu			+	+	+				
Gln				+					
Asp			+	+	+				
Asn				+		+			
Ser				+		+	+		
Thr	+			+		+			
Pro						+			

in the set of aligned sequences used to build the matrix. This means that BLOSUM50 will recognize more distantly related sequences than BLOSUM80 (Table 4.2).

4.1.2.2 Nucleotide Substitution Matrices

Substitution matrices for nucleotides give higher weight to transitions (G to A or C to T and vice versa) than to transversions (G to C, G to T, A to C, A to T, and vice versa) (Table 4.3). This is mainly related to the larger size of the purine compared to the pyrimidine nucleotides. The simplest model is the Jukes and Cantor model where equal probabilities for the substitution of all four nucleotides are assumed (Table 4.4). The transversion/transition bias is included in the models of HKY, Kimura, Tamura-Nei, Tamura, and the general reversible model. An additional account for the frequency of nucleotides is included in models of Tamura-Nei, the equal input, and the general reversible models. These frequencies are the observed frequencies of nucleotides under comparison. The unrestricted model allows different probabilities of changes for all 12 combinations of nucleotides (Table 4.4).



■ Fig. 4.4 The BLOSUM50 scoring matrix. Note that low scores both can be 0 and with a minus sign. Low scoring pairs occur with high frequencies in sequences. (Jesper Larsen is acknowledged for the figure)

■ Table 4.2 Practical rules for use of PAM and BLOSUM amino acid substitution matrices

Application	Default	Related sequences	Distant sequences
Search	BLOSUM62	BLOSUM80	BLOSUM45
Evolution	PAM100	PAM50	PAM250

■ Table 4.3 The sixteen different types of nucleotide pairs that can be observed between two sequences

Class	Nucleotide	Pair		
Identical nucleotides	AA	TT	CC	GG
Transition-type pair	AG	GA	TC	CT
Transversion-type pair	AT	TA	AC	CA
	TG	GT	CG	GC

Modified from Nei and Kumar (2000)

Table 4.4 Models of nucleotide substitutions

	A	T	C	G	A	T	C	G
	Jukes and Cantor				HKY			
A	–	α	α	α	–	βg_T	βg_C	αg_G
T	α	–	α	α	βg_A	–	αg_C	βg_G
C	α	α	–	α	βg_A	αg_T	–	βg_G
G	α	α	α	–	αg_A	βg_T	βg_C	–
	Kimura				Tamura-Nei			
A	–	β	β	α	–	βg_T	βg_C	$\alpha_1 g_G$
T	β	–	α	β	βg_A	–	$\alpha_2 g_C$	βg_G
C	β	α	–	β	βg_A	$\alpha_2 g_T$	–	βg_G
G	α	β	β	–	$\alpha_1 g_A$	βg_T	βg_C	–
	Equal input				General reversible			
A	–	αg_T	αg_C	αg_G	–	ag_T	bg_C	cg_G
T	αg_A	–	αg_C	αg_G	ag_A	–	dg_C	eg_G
C	αg_A	αg_T	–	αg_G	bg_A	dg_T	–	fg_G
G	αg_A	αg_T	αg_C	–	cg_A	eg_T	fg_C	–
	Tamura				Unrestricted			
A	–	$\beta\theta_2$	$\beta\theta_1$	$\alpha\theta_1$	–	a_{12}	a_{13}	a_{14}
T	$\beta\theta_2$	–	$\alpha\theta_1$	$\beta\theta_1$	a_{21}	–	a_{23}	a_{24}
C	$\beta\theta_2$	$\alpha\theta_2$	–	$\beta\theta_1$	a_{31}	a_{32}	–	a_{34}
G	$\alpha\theta_2$	$\beta\theta_2$	$\beta\theta_1$	–	a_{41}	a_{42}	a_{43}	–

Modified from Nei and Kumar (2000)

g_A , g_T , g_C , and g_G are the nucleotide frequencies, $\theta_1 = g_C + g_G$, $\theta_2 = g_A + g_T$. The nucleotide frequencies either can be estimated from the data compared or set as fixed parameters a_{ij} is the substitution rate from the nucleotide in the i -th row to the nucleotide in the j -th column

4.1.3 Gaps

Gaps are used to model insertions or deletions in sequences. If arbitrarily many gaps are inserted, this can lead to high-scoring alignments of nonhomologous sequences. To abolish this effect, gaps are penalized when alignments are constructed to obtain relatively few gaps and separate penalties used for gap opening and gap elongation. In the linear model, the penalty Υ is proportional to the number of gaps (g) given the penalty for one gap d :

Linear gap penalty score:

$$\Upsilon(g) = -gd$$

If the penalty of one gap is 8 and there are 3 gaps, the total penalty will be -24 .

To better model biological events where many gaps often occur in a row, the affine penalty score model is used where the cost is higher for inserting the first gap than to extend this gap to lengths of two and more:

$$Y(g) = -d - (g-1)e$$

where $Y(g)$ = gap penalty score of gap length g , d = gap opening penalty, and e = gap extension penalty. For a gap of length 3 with a penalty of opening the gap of 8 and of extending the gap with two more, the total penalty will be -16 .

4

4.1.4 Dynamic Programming


Dynamic programming is a way to compute the pairwise alignment? If all possibilities for the pairwise alignment of two sequences should be tested, there would be around 10^{59} possibilities for two nucleotide sequences of 100 in length. This number is approx. the same as the number of molecules in the Milky Way, and even the largest computer would not be able to handle the problem. There is also no need to consider possibilities without relevance, for instance, that one nucleotide in one sequence would pair only with gaps in the other.

To reduce the computing effort but still to perform a careful analysis, dynamic programming is used. The computer is only testing relevant comparisons (high scores) and keeping in memory the highest ones already calculated. The most relevant dynamic programming algorithms for comparison of global and local pairwise alignments were published by Needleman and Wunsch (1970) and Smith et al. (1981), respectively.



The Needleman and Wunsch algorithm is used when it is known that the sequences represent exactly a gene or protein in full length (few amino acids or nucleotide differences are tolerated) and it is used to build global pairwise alignments.

The Smith and Waterman algorithm is used with partial sequences or with sequences of unknown length to build local alignments. For both algorithms, the adjustment of gap penalty functions depends on previous knowledge about domain structures, repeats, and other properties. Both algorithms and their implementations in computer programs can be used for both amino acid and nucleotide sequences.

4.1.4.1 Needleman and Wunsch

The score function in this model is illustrated in  Fig. 4.5. This function is used on all comparisons between the two sequences. Here we will use the example with the two sequences:

- Sequence 1: HEAGAWGHEE
- Sequence 2: PAWHEAE

 Figure 4.6 shows one sequence (1) on the top of the table and sequence 2 as a vertical column. First an alignment path matrix is created. For each cell, $F(i, j)$ is calculated based on the score function in  Fig. 4.5. This matrix allows a stepwise calculation of score values. The score of the best alignment is calculated between the initial segment $x_{1\dots i}$ of x

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

Fig. 4.5 For the Needleman and Wunsch algorithm, the score function F is defined as the maximum of the three expressions. $F(i-1, j-1)$ is the score of the previous diagonal cell in the matrix which added the score for a match between an amino acid pair in the current cell. $F(i-1, j)$ is the score for the previous cell in the column subtracted the penalty of a gap (d), and $F(i, j-1)$ is the equivalent score for the previous cell in the row subtracted the gap penalty (see **Fig. 4.7** for an example of this calculation)

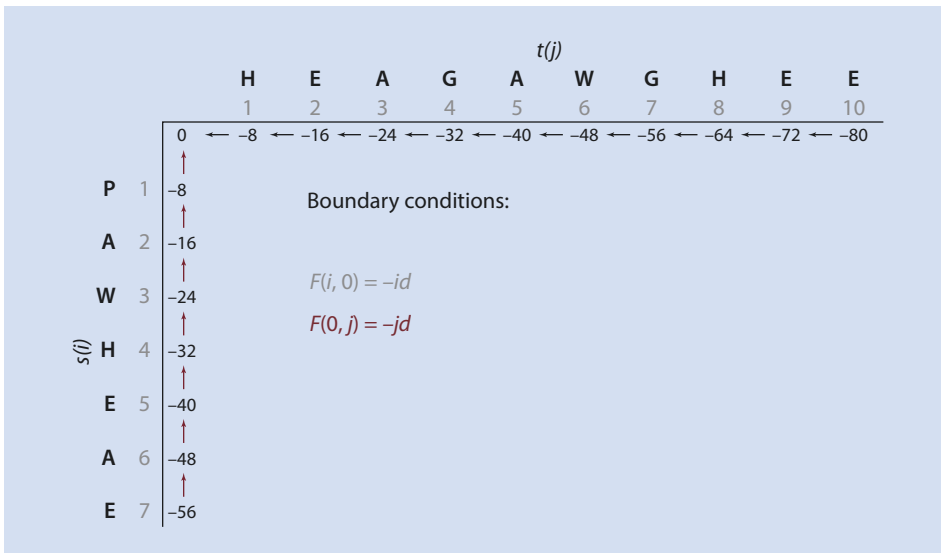


Fig. 4.6 Pairwise alignment with the Needleman and Wunsch algorithm showing the boundary conditions calculated. Here both of the sequences are paired only with gaps resulting in the lowest negative score. (Jesper Larsen is acknowledged for the figure)

up to x_i and the initial segment $y_{1..j}$ of y up to y_j . The algorithm function, $F(i, j)$, is built recursively beginning with $F(0,0) = 0$ (**Fig. 4.6, 4.7, and 4.8**). When the matrix is filled, backtracking is started (evaluation of the optimal path).

The scoring parameters are given by the BLOSUM 50 matrix (**Fig. 4.4**) and a linear gap penalty of $d = -8$. We will align the whole sequence length of the sequences meaning that we need to form a global alignment and use the Needleman and Wunsch algorithm. The procedure can be followed on **Fig. 4.6, 4.7, 4.8, and 4.9**.

A real example of the use of the Needleman and Wunsch algorithm is shown on **Fig. 4.10**. Here realistic long sequences are aligned by the algorithm implemented as the **needle** program in the EMBOSS package (Rice et al. 2000).

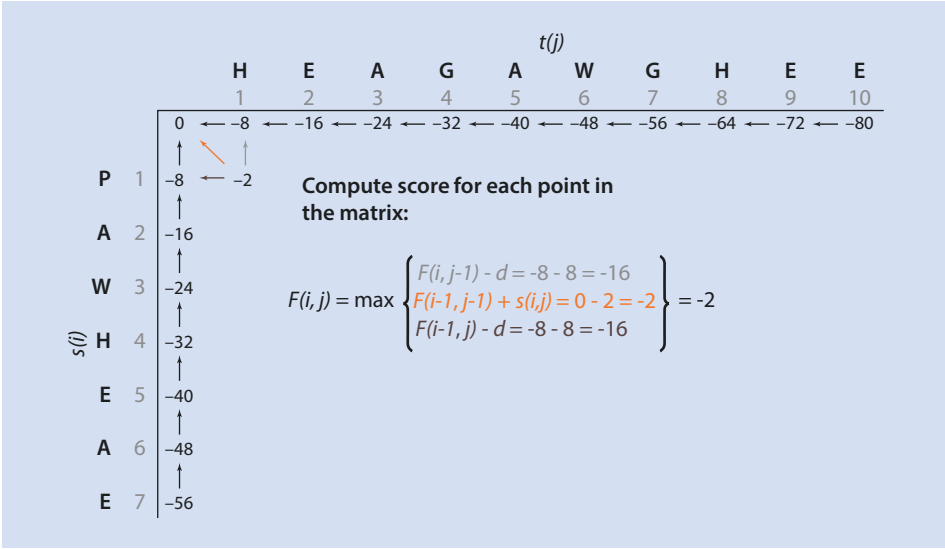


Fig. 4.7 Pairwise alignment with the Needleman and Wunsch algorithm. Here the first real position of the matrix is calculated. The three possibilities, H pairing with a gap, P pairing with a gap, or H pairing with P are considered. The one with the highest score: H pairing with P (-2) (see Fig. 4.4) gives the maximum score and is preferred. (Jesper Larsen is acknowledged for the figure)

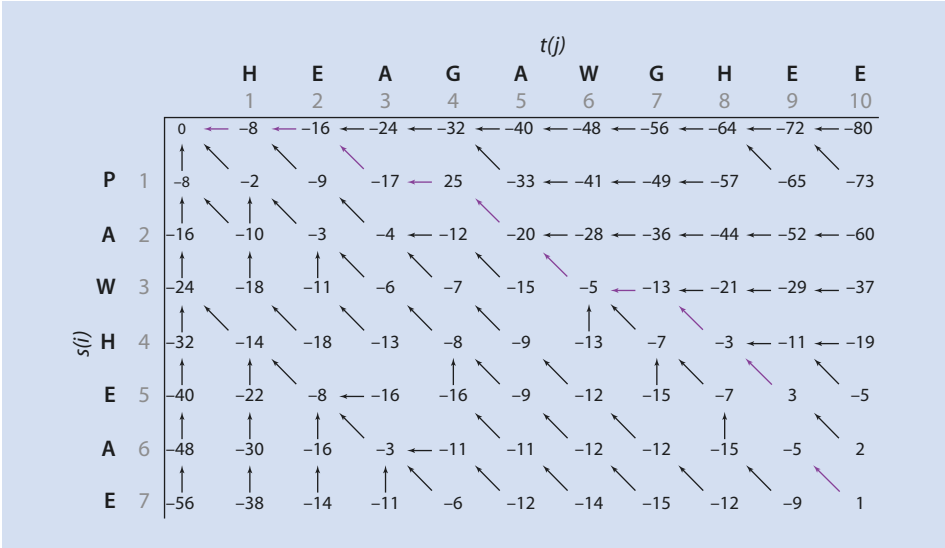
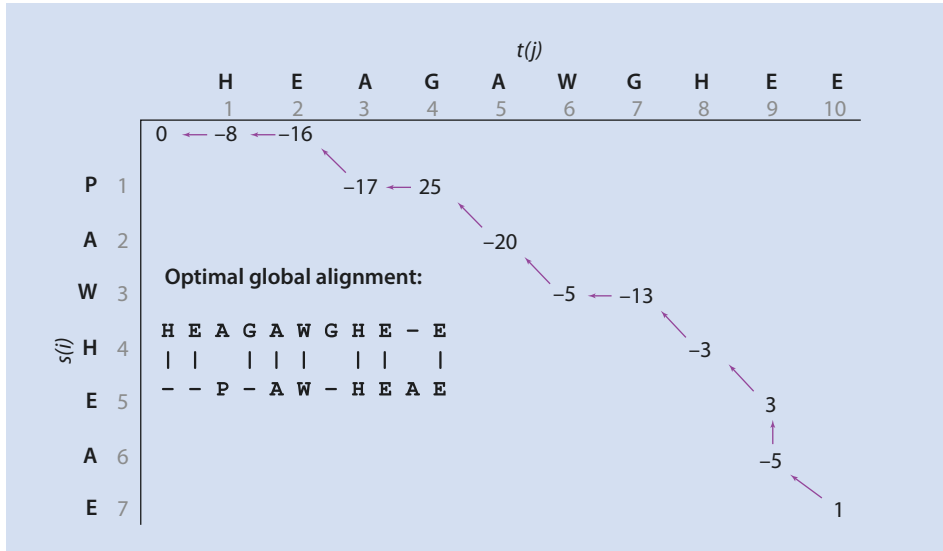


Fig. 4.8 Pairwise alignment with the Needleman and Wunsch algorithm. Here the maximum score has been inserted for all cells in the matrix, and backtracking from the bottom right corner has been performed following the path which maximizes the score. After the score of 1 for the cells to the lowest right, -5 is chosen since it was marked as a path to 1 in the forward tracking. The higher score of 2 cannot be selected since it was not marked in the forward tracking. (Jesper Larsen is acknowledged for the figure)



■ **Fig. 4.9** Pairwise alignment with the Needleman and Wunsch algorithm. The optimal path for backtracking is shown again (■ Fig. 4.8), and the pairwise alignment has been built. Note the combination of H and P considered in ■ Fig. 4.7 has not been selected since the backtrack path did not pass through this cell. The total score for the multiple alignments is 1 (bottom left corner). (Jesper Larsen is acknowledged for the figure)

4.1.4.2 Smith and Waterman

Now we will compute a pairwise alignment of the sequences using the Smith and Waterman algorithm. There are two differences in the algorithm compared to Needleman and Wunsch. The first is that the alignment can start/end anywhere in the matrix, and the other that backtracking is started at the highest value rather than in the lower right corner. Backtracking is terminated as soon as zero is encountered. The score function is shown in ■ Fig. 4.11.

We will align the same two short model protein sequences with Smith and Waterman:

- Sequence 1: HEAGAWGHEE
- Sequence 2: PAWHEAE

And use the same parameters of BLOSUM50 and a linear gap penalty of $d = -8$.

Using the score function in ■ Fig. 4.11 results in 0 in most cells when the matrix is filled in the forward direction (■ Fig. 4.12). The reason is that the three other possibilities in ■ Fig. 4.11 result in negative values and we then have to use 0. Backtracking starts with the highest score (28) and terminates when 0 is reached. Note that an alternative path is also possible. However, the maximum score of this one is lower (21), and therefore the first is chosen to construct the final pairwise alignment (■ Fig. 4.13). Similar to the example with Needleman and Wunsch (■ Fig. 4.10), a more realistic example is shown in ■ Fig. 4.14 with longer sequences. The sequences are the same that we used in ■ Fig. 4.10, and we see little difference between the two pairwise alignments probably because they were of nearly the same length.

```

#####
# Program: needle
# Rundate: Thu Feb 12 14:49:37 2004
# Align_format: srspair
# Report_file: outfile.align
#####
#=====#
#
# Aligned_sequences: 2
# 1: AAA85484.1
# 2: AAA85485.1
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 163
# Identity: 113/163 (69.35)
# Similarity: 137/163 (84.0%)
# Gaps: 1/163 (0.6%)
# Score: 607.0
#
#
#=====#

AAA85484.1      1      MKFFAVLALCIVGALAHPLTSDEAALVKSSWAQVKHNEVDILYTBFKAYP 50
|||||||:|||||.||:|:||||:|||||.|||||||..|.|
AAA85485.1      1      MKFFAVLALCVGALASPLSADEAAIVKSSWDQVKHNEVDILAAVFAAYP 50

AAA85484.1     51      DIQARFPQFAGKDLDTIKTSGQFATHATRIVSFSELLALSGSESNLSAI 100
||||:|||||||:|:|. . . | ||||| . . :|:|:||||:||||
AAA85485.1     51      DIQAKFPQFAGKDLASIKDTAAAFATHATRIVSFFTEVISLSGNQANLSAV 100

AAA85484.1    101      YGLISKMGTDHKNRGITQTQFNKFRITALVSYISSNVAWGDNVAAAWTHAL 150
|. :|:|. |. |||. |||. . . :| ||||| . . :|:|:||||:||||
AAA85485.1    101      YALVSKLGVDPKARGTSAAQFGEFRTALVSYLQAHVSWGDNVAAAWNHAL 150

AAA85484.1    151      DNVYTAVFQIVTA 163
||. |. . . . . .
AAA85485.1    151      DNTYAVALSKLE 162

```

Fig. 4.10 Pairwise alignment with the Needleman and Wunsch algorithm. A real example with sequences of realistic length has been computed using implementation of the algorithm in EMBOSS as the needle program. (see Activity 4.4.1 for details about this program)

Fig. 4.11 Score function for Smith and Waterman. An extra possibility of 0 can be selected in addition to the three in Fig. 4.5. The 0 is chosen if the other three are negative. This way cells are marked as 0 to account for different length of sequences in the local alignment (Fig. 4.12)

$$F(i, j) = \max \left\{ \begin{array}{l} 0 \\ F(i-1, j-1) + s(i, j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{array} \right.$$

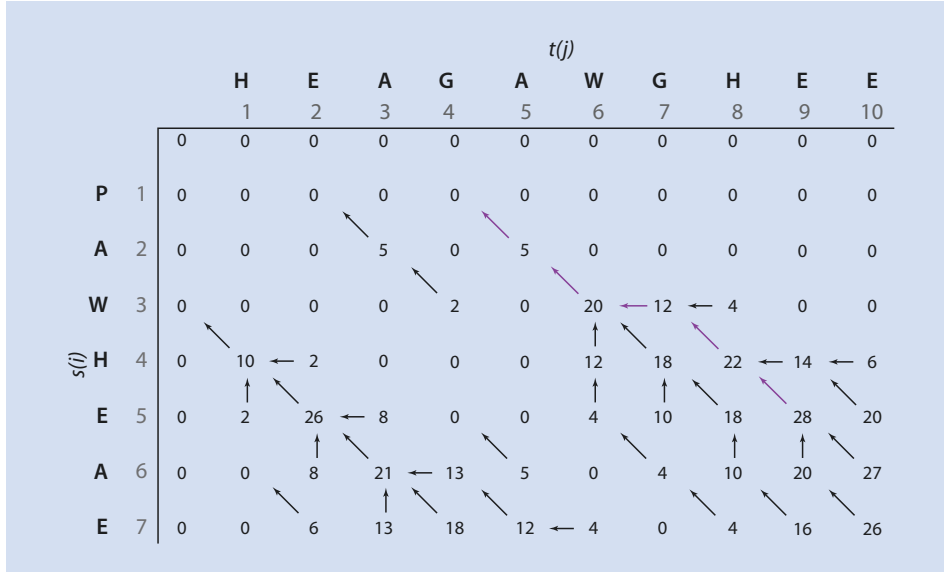


Fig. 4.12 Smith and Waterman alignment. Here the matrix has been filled with combinations based on the score function in Fig. 4.11. Backtracking started from the highest score in the matrix (28) and continued to join the highest possibilities until 0 was reached. (Jesper Larsen is acknowledged for the figure)

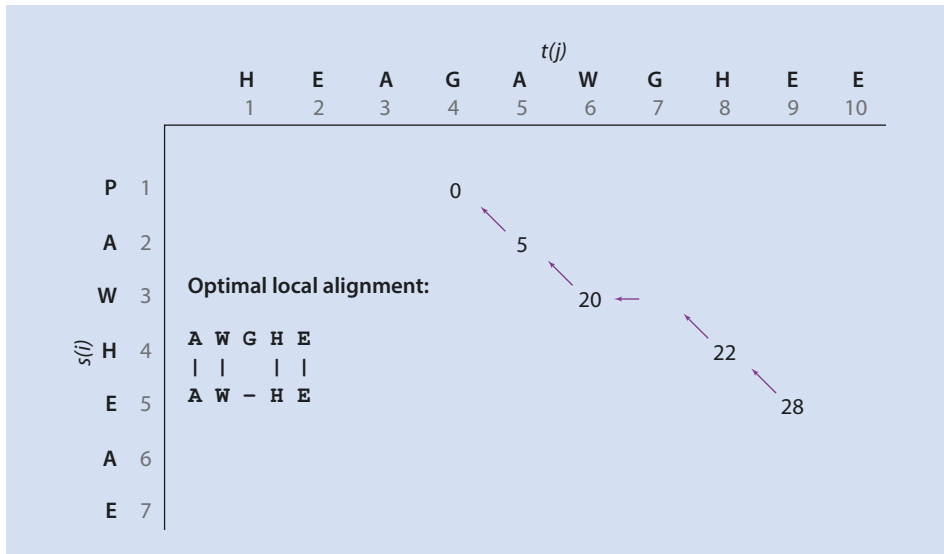


Fig. 4.13 Smith and Waterman algorithm. Here the backtrack path is shown and the resulting pairwise alignment has been constructed. Note the difference to global alignment generated by the Needleman and Wunsch algorithm in Fig. 4.9. The total score for the pairwise alignment is 28. (Jesper Larsen is acknowledged for the figure)

```

#####
# Program: water
# Rundate: Fri Feb 13 13:11:09 2004
# Align_format : srspair
# Report_file: outfile.align
#####
#=====
#
# Aligned_sequences: 2
# 1: AAA85484.1
# 2: AAA85485.1
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 159
# Identity: 113/169 (71.1%)
# Similarity: 136/159 (85.5%)
# Gaps: 0/159 (0.0%)
# Score: 609.0
#
#
#=====

AAA85484.1      1 MKFFAVLALCIVGALAHPLTSDEAALVKSSWAQVKHNEVDILYTBFKAYP 50
  |||:|||||:|||||.||::|||:|||||.|||||||.|||.|||
AAA85485.1      1 MKFFAVLALCVVGALASPLSADEAAIVKSSWDQVKHNEVDILAAVFAAYP 50

AAA85484.1     51 DIQARFPQFAGKDLDTIKTSGQFATHATRIVSFLSELLALSGSESLSAI 100
  |||:|||||||.||:|.:.|||:|||||||.||:|:|:|:|:|:|:|:|:|
AAA85485.1     51 DIQAKFPQFAGKDLASIKDTAAAFATHATRIVSFFTEVISLSGNQANLSAV 100

AAA85484.1     101 YGLISKMGTDHKNRGITQTQFNKFR TALVSYISSNVAWGDNVAAAWTHAL 150
  |.:|:|:|.|||.|||.|||.|||.|||.|||.|||.|||.|||.|||.|||
AAA85485.1     101 YALVSKLGVDPKARGISAAQFGEFR TALVSYLQAHVSWGDNVAAAWNHAL 150

AAA85484.1     151 DNVYTAVFQ 159
  |||.|||.|||.|||.|||.|||.|||.|||.|||.|||.|||.|||.|||.
AAA85485.1     151 DNTYAVALK 159

#-----
#-----

```

Fig. 4.14 Pairwise alignment with the Smith and Waterman algorithm. A real example with sequences of realistic length has been computed using the water program of EMBOSS which has implemented the Smith and Waterman algorithm (see ▶ Activity 4.4.2 for details about this program). Note that there are only small differences to the pairwise alignment generated by Needleman and Wunsch (▶ Fig. 4.10). It is related to the comparable length of the two sequences

4.2 Multiple Alignment

A multiple alignment is the simultaneous alignment of three or more nucleic acid or amino acid sequences. The procedure involves the insertion of gaps in the sequences so as to maximize the overall similarity (Higgins and Sharp 1988). Multiple alignments are rarely used for their own sake but are usually created for another purpose – for instance, primer design (▶ Chap. 5) or analysis of phylogeny (▶ Chap. 6). Users select a favorite program or program package and try to optimize program settings for that.

To start the construction of a multiple alignment, we will use the same criteria as for the pairwise alignment problem. We will assume that they shared ancestors (homologous). We need to model evolution the way that every column represents only orthologous amino acids or nucleotides that have evolved from a common ancestor. The simplest assumptions are again that identical nucleotides or identical amino acids pair, that different amino acids with similar physiochemical properties pair, that it is more difficult to form a gap than to form a mismatch, and that it is more difficult to form a gap than to extend one already formed. The scoring between nucleotides and amino acids is based on the system above for pairwise alignment (► Sect. 4.1.2).

4.2.1 Clustal

To form a multiple alignment of the clustal type, first pairwise alignments between sequences 1–2, 1–3, 1–4, 1–5, 2–3, 2–4, 2–5...4–5 are formed (■ Fig. 4.15). This is the progressive alignment part of the process, and there are $(n(n - 1))/2$ combinations where n is the number of sequences. In this example with five sequences, there are ten combinations. The next step is to construct a “guide tree” (■ Fig. 4.16) uniting the pairs with highest score. The final step is to construct the multiple alignments from the guide tree which is called profile alignment (■ Fig. 4.17). The clustal type of multiple alignments is therefore performing progressive profile alignment. Clustal has been called “a quick and dirty version of Feng and Dolittle (1987)” where “only residues that are part of the matches of a given length (k -tuple matches) are scored” (Higgins and Sharp 1988).

The developments in the clustal programs were started with Clustal (1988) (Higgins and Sharp 1988) and continued with ClustalV (five) (1992), ClustalW (weights) (1994), and ClustalX 2.0 (2007) (Larkin et al. 2007). ClustalX is the implementation of the program for PC (■ Figs. 4.18 and 4.19) and is further described in ► Activity 4.4.2. Clustal

```

CLUSTAL W (1.81) Multiple Sequenc Alignments

Sequence format is Pearson
Sequence 1: 001          238 bp
Sequence 2: 002          228 bp
Sequence 3: 005          227 bp
Sequence 4: 018          227 bp
Sequence 5: 120          228 bp
Start of Pairwise alignments
Aligning...
Sequences (1:2) Aligned. Score: 98
Sequences (1:3) Aligned. Score: 86
Sequences (1:4) Aligned. Score: 75
Sequences (1:5) Aligned. Score: 80
Sequences (2:3) Aligned. Score: 86
Sequences (2:4) Aligned. Score: 76
Sequences (2:5) Aligned. Score: 81
Sequences (3:4) Aligned. Score: 76
Sequences (3:5) Aligned. Score: 81
Sequences (4:5) Aligned. Score: 76
Guide tree      file created: [/ebi/extserv/clustalw-work/interactive/clustalw-20040304-09194253.dnd]
Start of Multiple Alignment
There are 4 groups
Aligning...
Group 1: Sequences:  2      Score:4197
Group 2: Sequences:  3      Score:3797
Group 3: Sequences:  4      Score:3662
Group 4: Sequences:  5      Score:3496
Alignment Score 10389
CLUSTAL-Alignment file created [/ebi/extserv/clustalw-work/interactive/clustalw-20040304-09194253.aIn

```

■ Fig. 4.15 Output from ClustalW showing the pairwise combinations and the grouping of sequences based on the guide tree (■ Fig. 4.16)

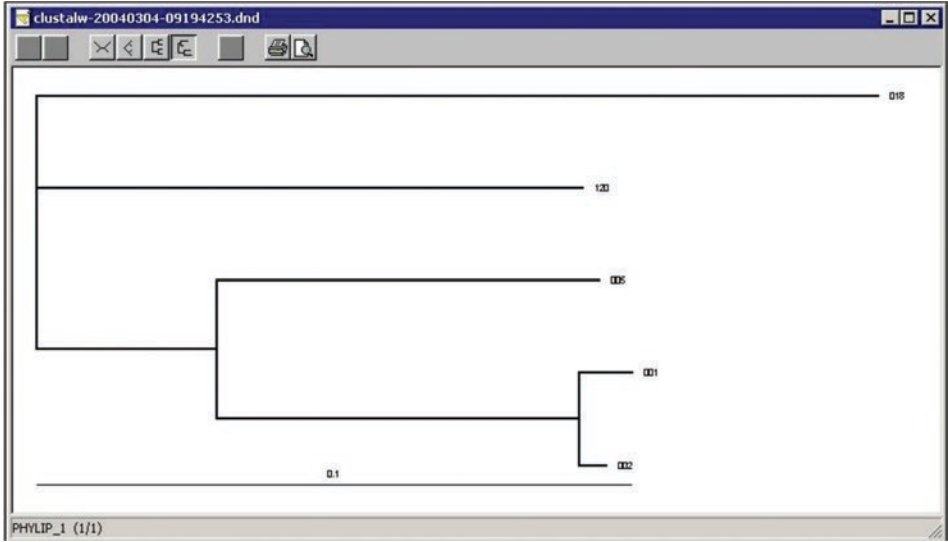


Fig. 4.16 The guide tree of ClustalW which is used by the program to perform the full multiple alignment

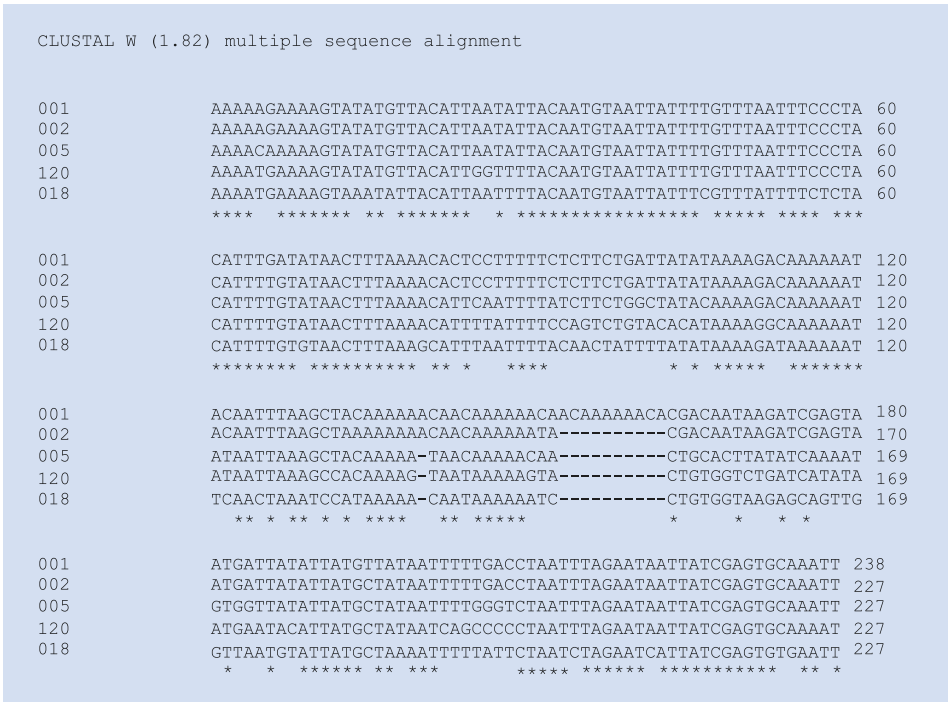
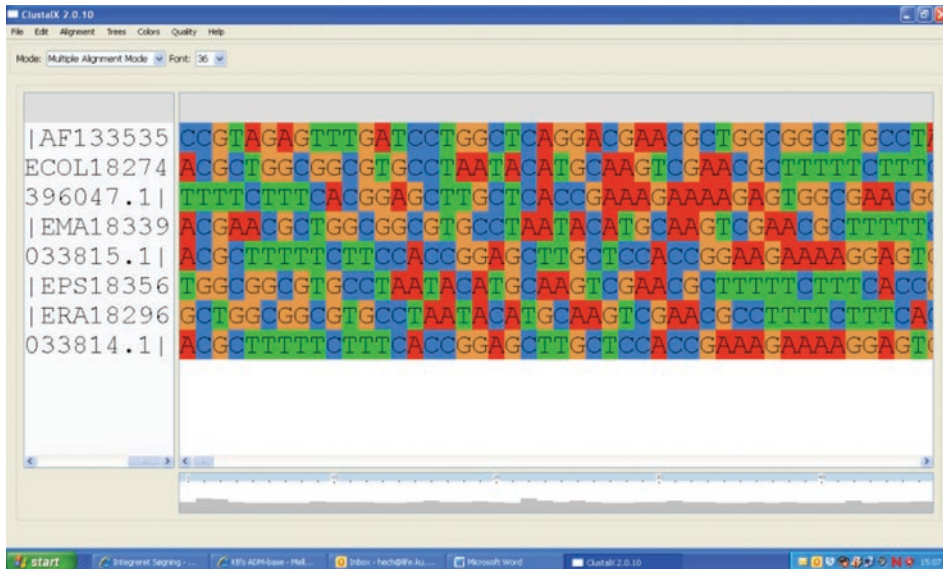
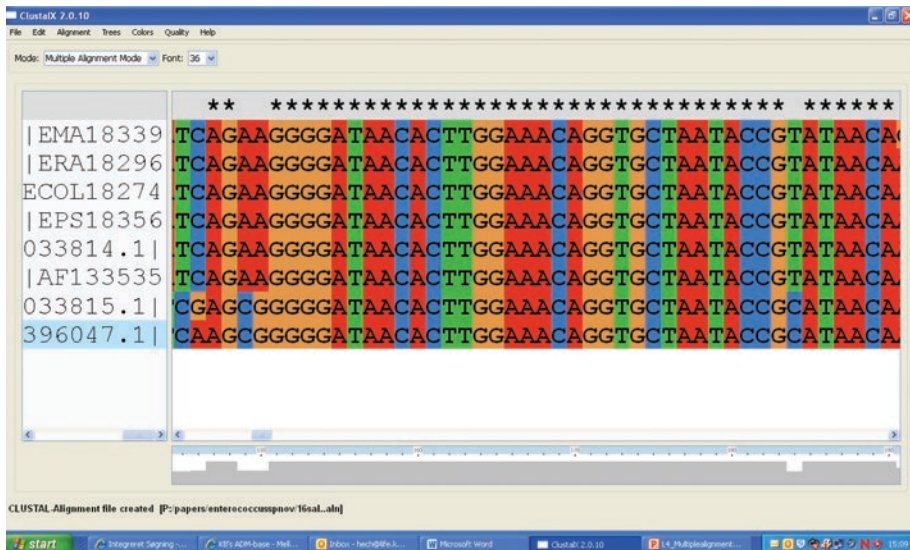


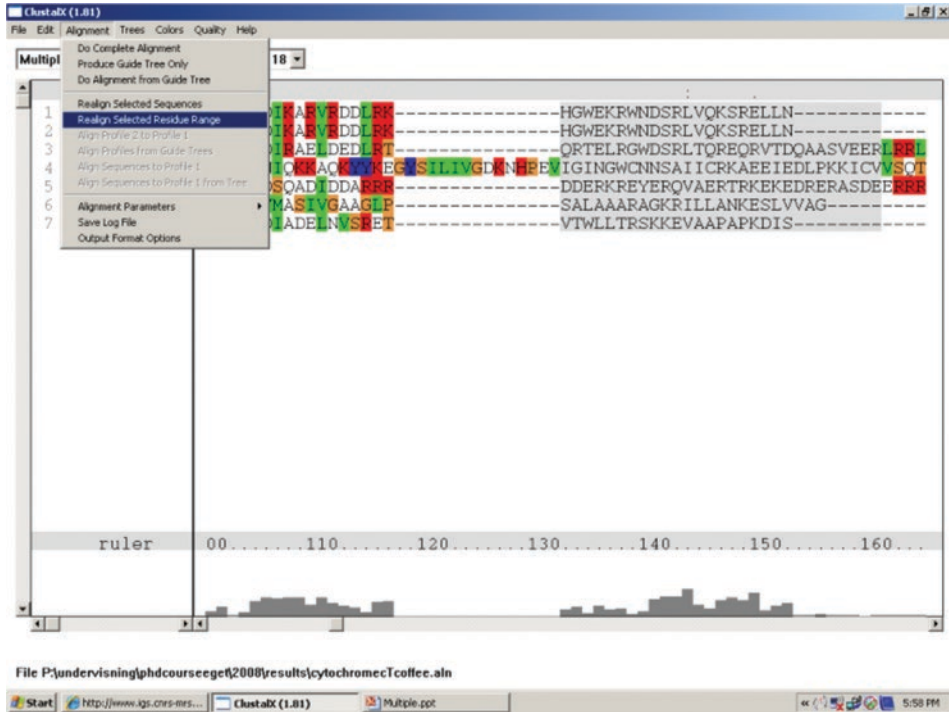
Fig. 4.17 The final multiple alignments made with ClustalW



■ Fig. 4.18 CLUSTALX (Larkin et al. 2007) showing the sequences not yet aligned. This is seen from the unordered arrangement of the nucleotides in the columns



■ Fig. 4.19 CLUSTALX (Larkin et al. 2007) showing aligned sequences where most of the columns show only one type of nucleotide



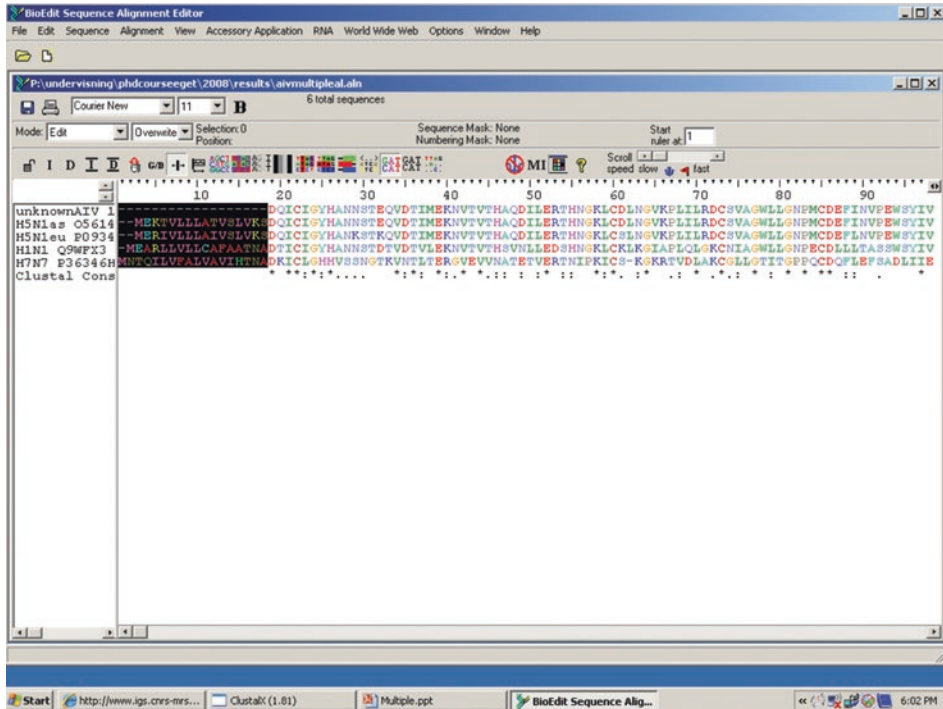
■ Fig. 4.20 CLUSTALX (Larkin et al. 2007) showing optimization by realignment of a region of the multiple alignment to improve local regions

Omega is the most recent version of the program for multiple alignments of very large protein datasets (Sievers et al. 2011).

ClustalX/W produces global alignments which include benefits and drawbacks inherited from the pairwise alignment part included in the construction of the alignment. Domain structures of proteins can be optimized by ClustalW/X (■ Fig. 4.20). ClustalW/X invokes different “hidden” functions during alignment and tends to be cluster gaps and take hydrophobic/hydrophobic properties into account. An inherent problem with the progressive approach used in ClustalX/W is that mistakes made in the initial alignment cannot be corrected later. To account for this and other problems, other programs have been constructed.

4.2.2 Other Multiple Alignment Programs

MUSCLE (Multiple Sequence Comparison by Log-Expectation) (Edgar 2004) (► <https://www.drive5.com/muscle/downloads.htm>) includes no guide tree. MUSCLE is claimed both to achieve better average accuracy and better speed than ClustalW2 or T-Coffee (Edgar 2004). The main focus is on protein multiple alignments, but it works also on DNA. The principle is based on K-mer comparison between sequences. K-mers are contiguous subsequence of short length (k-tuple). Related sequences tend to have more k-mers in common than expected by chance. The program can be used from MEGA7 (Tamura et al.



■ Fig. 4.21 BIOEDIT (Hall 1999) ([▶ http://www.mbio.ncsu.edu/BioEdit/bioedit.html](http://www.mbio.ncsu.edu/BioEdit/bioedit.html)) used to trim a region of a multiple alignment with many gaps inserted as a consequence of lack of data (short sequences). The box marked in black was deleted

2011) ([▶ https://www.megasoftware.net/](https://www.megasoftware.net/)) used in ▶ Chaps. 6 and 11. Like the following programs, T-Coffee and MAFFT have options to adjust alignment parameters for a range of applications.

T-Coffee ([▶ http://tcoffee.crg.cat/](http://tcoffee.crg.cat/)) (Notredame et al. 2000) incorporates a “tree-based consistency objective function for alignment evaluation.” The benefit should be high accuracy. The program comes in a range of flavors suitable for different applications.

MAFFT (Yamada et al. 2016 and other papers) ([▶ https://mafft.cbrc.jp/alignment/software/](https://mafft.cbrc.jp/alignment/software/)) is also a multiple alignment package that includes applications for very large datasets.

Multiple alignment programs can be compared with sets of reference sequence (BaliBASE) (Thompson et al. 1999).

To account for the secondary structures in 16S rRNA sequence, the ARB package (Ludwig et al. 2004) ([▶ http://www.arb-home.de/](http://www.arb-home.de/)) has been developed. It has been used to construct precomputed multiple alignments in the SILVA database ([▶ www.arb-silva-de](http://www.arb-silva.de)) which is further considered in ▶ Chap. 8. Folding patterns in 16S rRNA genes can be predicted with Mfold ([▶ http://unafold.rna.albany.edu/?q=mfold](http://unafold.rna.albany.edu/?q=mfold)) (Zuker and Jacobson 1998) or similar program.

Trimming should only be done with limited datasets of closely related organisms. Trimming can be done in BioEdit (■ Fig. 4.21). For Mac users Jalview can be used. ([▶ http://www.jalview.org](http://www.jalview.org)) (Waterhouse et al. 2009)

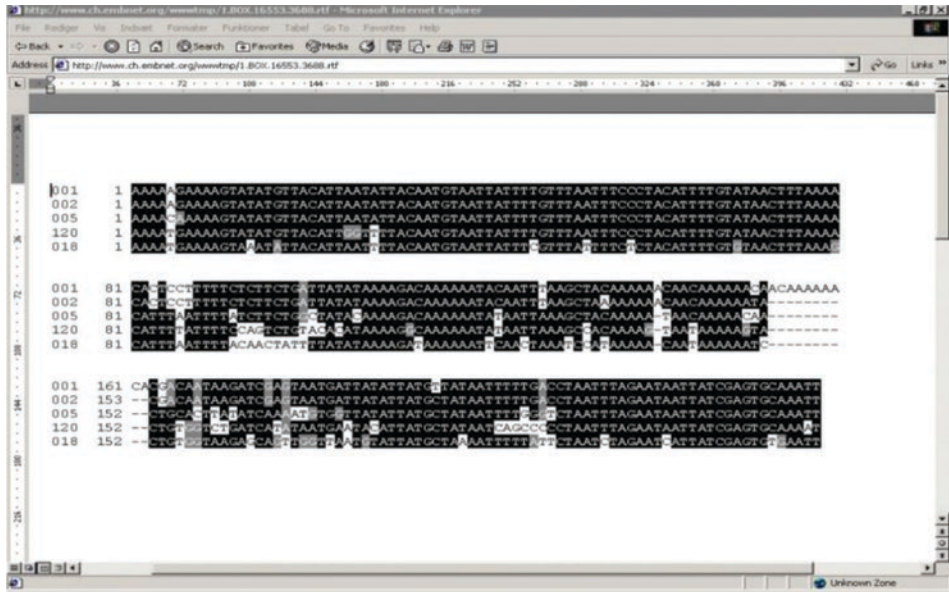


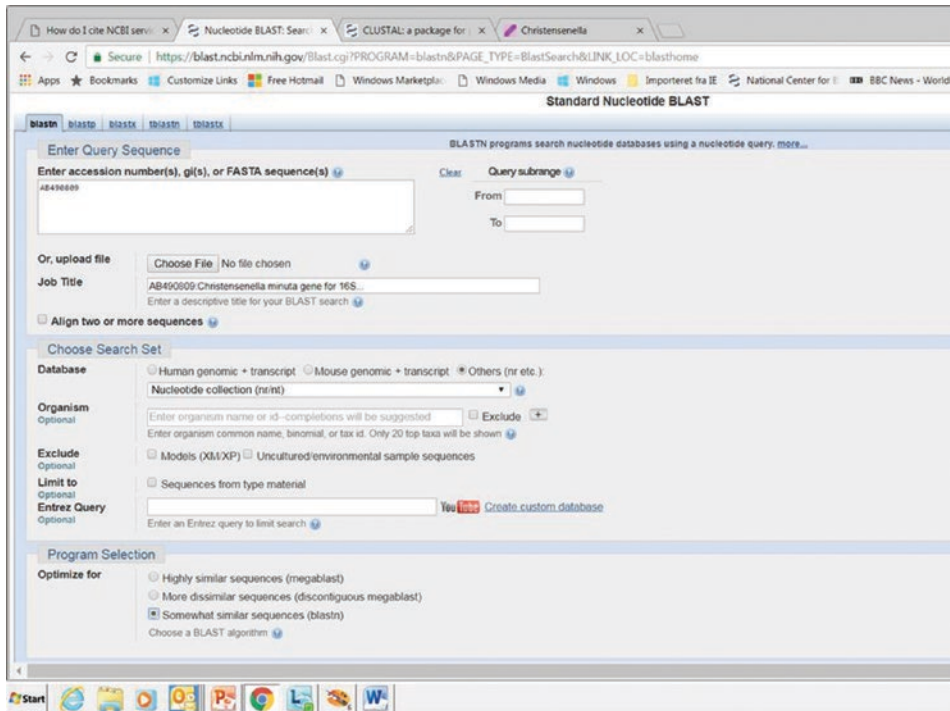
Fig. 4.22 Output from BoxShade ([▶ http://www.ch.embnet.org/software/BOX_form.html](http://www.ch.embnet.org/software/BOX_form.html)). This output can be included for presentation in a publication

Graphic output of multiple alignments can be made in BoxShade ([▶ http://www.ch.embnet.org/software/BOX_form.html](http://www.ch.embnet.org/software/BOX_form.html)) (Fig. 4.22).

4.3 BLAST

BLAST (Basic Local Alignment Search Tool) was originally described by Altschul et al. (1990), and this tool enabled fast search in the electronic nucleotide and protein databases during the 1990s when the Internet was invented. Later an improved algorithm was described allowing gaps to be inserted during the analysis (Altschul et al. 1997). Both versions of BLAST are in use as the “ungapped” (1990) and “gapped” (1997), respectively.

BLAST is based on a heuristics search algorithm which is able to search through the ever-growing sizes of databases. BLAST can be compared to the search tool Google in popularity within the bioinformatics community including the formation of the verb “to blast.” BLAST is based on the initial identification of the users’ query sequence by short pieces of sequence (words). These words are compared to the database with all the sequences (subjects). If the word is identified in a sequence of the database, the match of the word can be extended and will form a high-scoring pair (HSP). When no further extension is possible, the result is returned as a hit list to be used with indication of similarity between the query and the closest related subjects in the database. The “gapped” version of BLAST is most frequently used, and its “two-hit method” requires two non-overlapping “words” on the same diagonal with a distance of A before an extension is invoked. Gapped BLAST should only require 1/3 computer time compared to ungapped version because of less extensions of the words. BLAST uses dynamic programming to extend residues in both directions; BLAST is most suitable





■ Fig. 4.23 BLAST search (Altschul et al. 1990, 1997). (BLAST [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – [cited 2018 04 17]. Available from: ► <https://www.ncbi.nlm.nih.gov/Blast.cgi>)

for finding of subject DNA and protein sequences in databases that match with a reasonable similarity and a comparable length to the query. BLAST is good for sequences of at least 100 in length that are well represented in the databases. BLAST is not suitable to make really precise determination of similarity between sequences, and pairwise alignment programs should instead be used for that (► Sect. 4.1). BLAST is not suitable for sequence-based identification of bacterial species since there are simply too many sequences in GenBank and the type strains are not always clearly labeled (► Chap. 7). BLAST is not suitable for databases beyond a certain size and cannot be used to search very large metagenomics databases (► Chap. 9). In principle the database search of BLAST is based on local alignments. It needs to be local since we are never certain if our query sequence will be represented by a homolog in the database with similar length. BLAST is most often used on the NCBI server (► <http://www.ncbi.nlm.nih.gov/BLAST/>); however, the BLAST program can also be installed on your local computer and used with your own database.

4.3.1 NCBI BLAST

The most common way of performing a BLAST search is to access the program at NCBI and search in their databases. A search can be performed by a DNA or protein sequence in FASTA format, by uploading a sequence file in FASTA format or by inserting an accession number as AB490809 shown in ■ Fig. 4.23 if the sequence is already included with

GenBank. In the example shown on  Fig. 4.23, default setting has been selected for a BLAST search with a DNA sequence. Under **Program Selection** and **Optimize for different versions of BLAST** can be selected. In this case **megablast**, **discontiguous megablast**, and **blastn** can be selected. The three versions differ by the scores given to matches and their way of penalizing gaps. For shorter sequences **blastn** should be used.

When the search is terminated, you will see the familiar graphical overview in  Fig. 4.24 – the colored section with horizontal bars. You can see details of subject sequences by mousing over the bars in this section.

In the next section comes the descriptions with one line for each subject which gives a more detailed information about the subject sequences providing **Score**, **Query cover**, **E value**, and **accession number**. These parameters will be explained in the following.


The hit list is arranged by decreasing **Score**. In the example you see that hits are sorted by decreasing score. The identities are also decreasing but not systematically since the score is not always directly related to the identity. The longer the match between two sequences, the higher the score given the same identity. E values cannot be directly seen here since they are very small and they are just shown as “0.0.”

In the alignments section, the pairwise alignments are shown. This is the most detailed information of comparison between your query sequence and the subjects in the database. The parameters from the description section are available, and additional information of gaps and visual location of both gaps and mismatches can be observed.

4.3.2 Ortholog Detection

Sequences are orthologs if they are homologs and have the same function in different species. The paralogs are also homologs but have been duplicated in evolution, and they have got divergent functions in the same species, and the sequences are often quite divergent. BLAST can be used to sort the orthologs from the paralogs. The principle of reciprocal best hits (RBH) is used as explained in Moreno-Hagelsieb and Latimer (2008) “Two genes residing in two different genomes are deemed orthologs if their protein products find each other as the best hit in the opposite genome” (Moreno-Hagelsieb and Latimer 2008). Further investigation of orthologs and paralogs can be done by phylogeny (► Chap. 6).

4.3.3 BLAST2 Sequences

BLAST2 sequences is a version of NCBI BLAST that allows pairwise comparisons on server or on stand-alone BLAST installed on your PC. The program is available on both DNA and protein level, and it can be used to build your own database for many purposes by including many sequences in the 2nd window ( Fig. 4.25). For instance, if the query DNA or protein sequence is included with the 1st search window and a range of genomes included in the 2nd search, you can identify the query sequence in

Pairwise Alignment, Multiple Alignment, and BLAST

The figure displays three screenshots of the NCBI BLAST web interface. The top screenshot shows the 'BLAST Results' page for a search of 'Christensenella minuta gene for 16S...'. It includes a 'Graphic Summary' section with a bar chart titled 'Distribution of the top 12 Blast Hits on 10 subject sequences'. The chart shows alignment scores for 10 sequences, with a color key indicating score ranges: <math>K < 40</math> (black), 40-50 (blue), 50-80 (green), 80-200 (red), and > 200 (dark red). The middle screenshot shows the 'Descriptions' section, listing sequences producing significant alignments with columns for Description, Max score, Total score, Query cover, E value, Ident, and Accession. The bottom screenshot shows the 'Alignments' section, displaying a detailed view of sequence alignments between the query and subject sequences, including sequence coordinates and alignment scores.

■ Fig. 4.24 BLAST (Altschul et al. 1990, 1997) output showing three screenshots of the output from BLAST search: **Graphic Summary**, **Description**, and **Alignments**, respectively. The number of Max target sequences was reduced to ten in the alignment parameters section to show it all on one page. (BLAST [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – [cited 2018 04 17]. Available from: ► <https://www.ncbi.nlm.nih.gov/Blast.cgi>)

■ Fig. 4.25 BLAST2 sequences (Altschul et al. 1990, 1997). (BLAST [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – [cited 2018 04 17]. Available from: ► <https://www.ncbi.nlm.nih.gov/Blast.cgi>)

the whole genomic sequences including genomes not yet deposited with the databases. The same approach can be used to build a small MLST database (► Chap. 11). BLAST2 sequences is started from the ordinary BLAST page NCBI by marking “Align two or more sequences.” It can be used for fast draft comparisons between two sequences for example if you want to look up the location of a primer or gene on a genome. This is your “Swiss knife” always available if you just can access the Internet. Like for an ordinary knife, be careful with BLAST2 sequences! If you need real precise similarities between sequences, they should be obtained by the pairwise alignment programs described in ► Sect. 4.1.

4.3.4 Statistics

Evaluation of the results in the BLAST output is based on the Karlin and Altschul formula $E = k \times m \times N \times e^{-\lambda S}$ (Karlin and Altschul 1990) where m is letter in query meaning the number of nucleotides or amino acids, N is the total letters in database, and S is the actual score.

For amino acids, S is defined by the BLOSUM matrix. S is scaled to bit score to better fit the computer. The bit score = $((\lambda \times S) - \ln \kappa) / \ln 2$. In the pairwise section (■ Fig. 4.24 above), you can see the raw score in parenthesis along with the bit score,

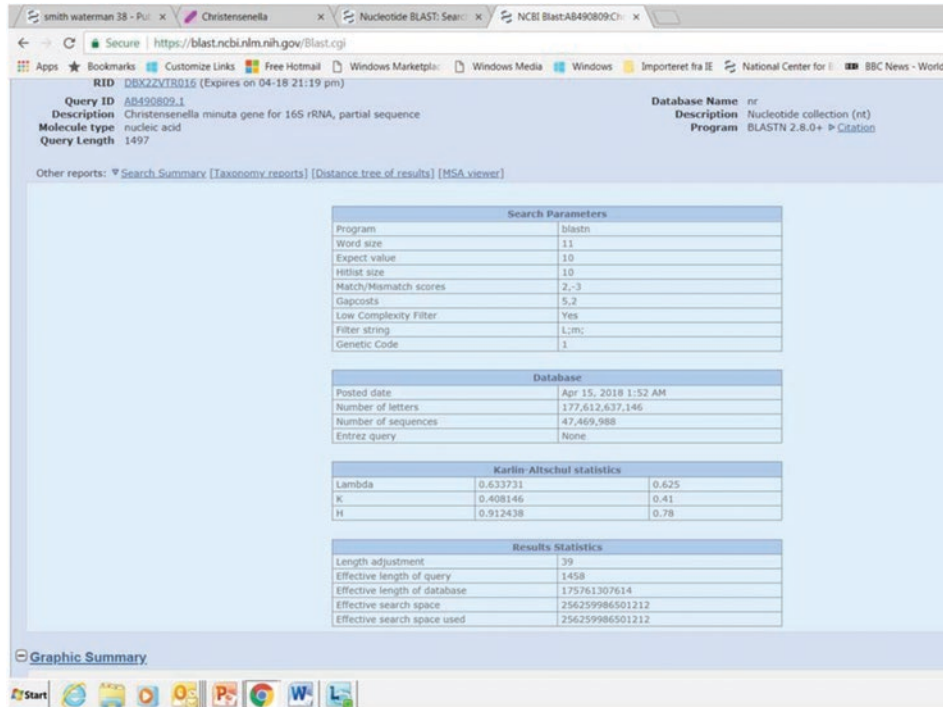


Fig. 4.26 The Search Summary section of the output showing all parameters used for the search (Altschul et al. 1990, 1997). (BLAST [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – [cited 2018 04 17]. Available from: ► <https://www.ncbi.nlm.nih.gov/Blast.cgi>)

the constants κ and λ depending on the database. E reflects that we find the sequence in the database by chance (chance for false positives). E is this way related to the size of the database. The smaller E the less likely is it found by chance and the more unique is the sequence – “The smaller E the better.” For really “unique” sequences, E is so small that it cannot be represented on the computer; this is the reason for “0.0” with the output.

All parameters related to the search can be found in **Search Summary** (Fig. 4.26); just press the bottom **Graphic summary** on the output page (Fig. 4.24). Here you can find parameters used for calculating the statistics in the Karlin and Altschul formula, and you can understand everything about the BLAST output if you compare these parameters to the statistics just explained.

Note that parameters for ungapped blast (Altschul et al. 1990) differ from gapped blast (Altschul et al. 1997) (Fig. 4.26) and λ and κ are different for gapped BLAST compared to ungapped.

4.3.5 Variants of BLAST

A BLAST search can be based on a query DNA sequence that the program translates to protein and compares to all protein sequence in the database (BLASTx). In this case the search takes more time, and it should only be done when it is absolutely needed (rarely).

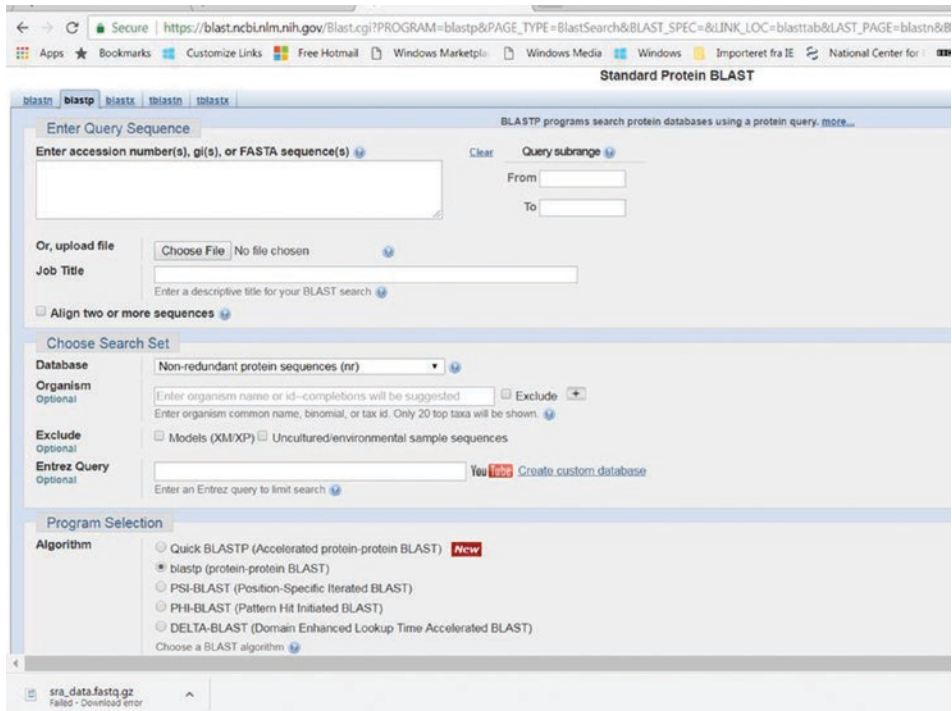


Fig. 4.27 Protein BLAST and BLASTp; note the additional options PSI- PHI- and delta BLAST (Altschul et al. 1990, 1997). (BLAST [Internet]. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004 – [cited 2018 04 17]. Available from: ► <https://www.ncbi.nlm.nih.gov/Blast.cgi>)

For such comparison, the codon table should be defined according to the type of organisms. Prokaryotes are using codon table 11 (Appendix). tBLASTn searches a translated nucleotide databases using a protein query. tBLASTx searches a translated nucleotide query against the translated nucleotide databases.

For BLASTp it is sometimes convenient to select Swiss-Prot or ref_seq_protein if you want a really precise information about the hits. For protein searches note the additional options PSI-, PHI-, and delta BLAST (► Fig. 4.27); they can be used to build up a profile used to search for query proteins which are rather distantly related to the subject proteins in the database.

4.4 Activities

4.4.1 Pairwise Alignment

We will construct local and global pairwise alignments by use of the program “water” and “needle,” respectively, available as EMBOSS programs (Rice et al. 2000) on the EBI server. The protein sequences AAA85484 and AAA85485 can be downloaded from NCBI as described in ► Chap. 3 (see ► Sect. 3.3.1 including Activity 3.8.1).

Decide if you want to use “needle” for global – or “water” for local alignment.

Open the FASTA format files, and paste them in the windows of the relevant EMBOSS program on the server:

- ▶ http://www.ebi.ac.uk/Tools/psa/emboss_needle/
- ▶ http://www.ebi.ac.uk/Tools/psa/emboss_water/

If you need to do comparisons of nucleotide sequences, the similar programs can be found here:

- ▶ http://www.ebi.ac.uk/Tools/psa/emboss_needle/nucleotide.html
- ▶ http://www.ebi.ac.uk/Tools/psa/emboss_water/nucleotide.html

You can also install the **mEMBOSS** package on your PC and select the programs **water** and **needle** from the menu. Download **mEMBOSS** from ▶ <ftp://emboss.open-bio.org/pub/EMBOSS/windows/> Click on ▶ **mEMBOSS-6.3.1.2-setup.exe**

And download to local computer. Click on file and install.

Use **Explorer** from **Windows** to navigate the folder with **mEMBOSS**.

Go into the **Jemboss** folder and click **Jar** and **Jemboss MS**; this will allow you to run **EMBOSS** in **Windows** from a graphic interface. All commands to activate the programs are found on the left menu bar. Most useful are **water**, **transeq**, and **revseq** which will do pairwise Smiths and Water alignments, translate from DNA to protein, and reverse and complement sequences, respectively.

4.4.2 Learn How Dynamic Programming Works with Pairwise Alignments

Use the tool at: ▶ <http://www.itu.dk/~sestoft/bsa/graphalign.html>.

Use the short protein sequences shown already. Try the different substitution matrices, let the gap costs stay at “linear, gap score –8” to reduce the complexity, or make your own choice. Try with or without traceback. The tool has been designed and is maintained by Peter Sestoft.

4.4.3 Multiple Alignment with ClustalX

From ▶ <ftp://ftp.ebi.ac.uk/pub/software/clustalw2/2.0.10/>, download ▶ [clustalx-2.0.10-win.msi](ftp://ftp.ebi.ac.uk/pub/software/clustalw2/2.0.10/win.msi), install, and locate the icon to the desktop. Open the program by double-clicking on the icon. **File | Load sequences**, and select the sequences in FASTA format from the location on your computer. You need to edit the input file yourself. The sequences that you want to use as input should be in one file only with all sequences in FASTA format:

```
>sequence1
ATGACGATAC...
>sequence2
GATAGATAGACS...
etc.
```

Select **Alignment | Do complete Alignment | ok**

In the lower left corner, you can follow how the program works.

Scroll through the alignment when it is completed.

At the bottom, the bars show the degree of conservation of columns. Above the alignment the signature * shows columns with conserved nucleotides (for amino acids: with the same properties).

4.4.4 BLAST

4

We will use this activity to learn how to perform an ordinary BLAST search in GenBank with a DNA sequence. However, we also learn some more about BLAST in this activity. We will perform a search with the sequence with acc. no. AF445297 from GenBank. The expectation is that this will be the best hit from the BLAST output. To a surprise it is not so.

First perform a BLAST search at NCBI: ► <https://blast.ncbi.nlm.nih.gov/Blast.cgi>. On the graphics select **nucleotide BLAST**, insert the acc. no. AF445297 in the window, mark “**somewhat similar sequences blastn**,” mark “**show results in a new window**,” and press the blue **BLAST** bottom. You will get a nice output and 100% identity to a sequence from an uncultured organism. Where is your query sequence? Usually you would expect the query as the best hit if it comes from the database. The reason can be found if you look up AF445297 at NCBI, since it is labeled as unverified. The unverified sequences are not included as databases in any BLAST search.

Take-Home Messages

- Pairwise alignments and multiple alignments are the basic tools for sequence comparison.
- A quantitative pairwise comparison of two sequences can be performed by aligning two sequences based on considerations of gaps representing insertions or deletions and matches between nucleotides or amino acids.
- Pairwise comparisons can be performed as global alignments if it is known that the sequences are homologous in their full length or by local alignments if it is known that one sequence is shorter than the other.
- BLAST is the most frequently used bioinformatics program to compare your own sequence (query sequence) to all sequences in a database (subject sequences).
- BLAST provides qualitative information about homologous sequences and can quantify the identity of the query sequence to the subject sequences in the database.
- Substitution matrices provide the probability of nucleotide or amino acid substitutions when two or more sequences are compared.
- Dynamic programming is the most frequently used procedure to perform pairwise comparisons of nucleotide or amino acid sequences and can be done based on the Needleman and Wunsch algorithm for global alignments and by the Smiths and Waterman algorithm for local alignments.
- Multiple alignments are most frequently constructed by the progressive profile procedure as implemented in the clustal family of programs.

References

- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215, 403–410.
- Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* 25:3389–3402.
- Durbin, R., Eddy, S., Krogh, A. and Mitchison, G. 1999. *Biological sequence analysis*. Cambridge Univ. Press.
- Edgar, R. C. 2004. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.* 32(5):1792–1797
- Feng, D. F. & Dolittle, R. F. 1987. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.* 25, 351–60.
- Hall, T. A. 1999. BioEdit: a user-friendly biological sequence alignment editor and analysis program for Windows 95/98/NT. *Nucl Acids Symp Ser* 1999;41:95–98.
- Higgins, D. G. & Sharp, P. M. 1988. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene* 73, 237–244.
- Karlin, S., & Altschul, S. F. 1990. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc Natl Acad Sci USA* 87, 2264–8.
- Larkin, M. A., Blackshields, G., Brown, N. P., Chenna, R., McGettigan, P. A., McWilliam, H., Valentin, F., Wallace, I. M., Wilm, A., Lopez, R., Thompson, J. D., Gibson, T. J., Higgins, D. G. 2007. Clustal W and Clustal X version 2.0. *Bioinformatics* 23:2947–2948.
- Ludwig, W., Strunk, O., Westram, R., Richter, L., Meier, H., Yadhukumar, Buchner, A., Lai, T., Steppi, S., Jobb, G., Förster, W., Brettske, I., Gerber, S., Ginhart, A. W., Gross, O., Grumann, S., Hermann, S., Jost, R., König, A., Liss, T., Lüssmann, R., May, M., Nonhoff, B., Reichel, B., Strehlow, R., Stamatakis, A., Stuckmann, N., Vilbig, A., Lenke, M., Ludwig, T., Bode, A., Schleifer, K. H. 2004. ARB: a software environment for sequence data. *Nucleic Acids Res.* 32, 1363–1371.
- Moreno-Hagelsieb, G. & Latimer, K. 2008. Choosing BLAST options for better detection of orthologs as reciprocal best hits. *Bioinformatics* 24:319–24.
- Needleman, S. B. & Wunsch, C. D. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* 48, 443–453.
- Nei, M. & Kumar, S. 2000. *Molecular evolution and phylogenetics*. Oxford University Press.
- Notredame, C., Higgins, D.G., Heringa, J. 2000. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.* 302: 205–217.
- Rice, P., Longden, I., & Bleasby, A. 2000. EMBOSS: The European Molecular Biology Open Software Suite. *Trends Genetics* 16:276–277.
- Sievers, F., Wilm, A., Dineen, D., Gibson, T. J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Söding, J., Thompson, J. D. & Higgins, D. G. 2011. Fast, scalable generation of high-quality protein multiple sequence alignments using ClustalOmega. *Mol. Syst. Biol.* 7:539.
- Smith, T. F., Waterman, M. S. & Fitch, W. M. 1981 Comparative biosequence metrics. *J. Mol. Evol.* 18, 38–46.
- Tamura, K., Peterson, D., Peterson, N., Stecher, G., Nei, M. et al. MEGA5: molecular evolutionary genetics analysis using maximum likelihood, evolutionary distance, and maximum parsimony methods. *Mol Biol Evol* 2011; 28: 2731–2739.
- Thompson, J. D., Plewniak, F. & Poch, O. 1999. BALIbase: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics* 15:87–88.
- Waterhouse, A. M., Procter, J. B., Martin, D. M., Clamp, M., Barton, G. J. 2009. Jalview Version 2--a multiple sequence alignment editor and analysis workbench. *Bioinformatics* 25, 1189–1191.
- Yamada, K.D., Tomii, K., Katoh, K. 2016. Application of the MAFFT sequence alignment program to large data-reexamination of the usefulness of chained guide trees. *Bioinformatics* 32, 3246–3251.
- Zuker, M. & Jacobson, A. B. 1998. Using Reliability Information to Annotate RNA Secondary Structures. *RNA* 4, 669–679, 1998.