Chapter 3

Sequence Analysis in Immunology

3.1 Sequence Analysis

The concept of protein families is based on the observation that, while there are a huge number of different proteins, most of them can be grouped, on the basis of similarities in their sequences, into a limited number of families. Proteins or protein domains belonging to a particular family generally share functional attributes and are derived from a common ancestor, and will most often be the result of gene duplication events.

It is apparent, when studying protein sequence families, that some regions have been more conserved than others during evolution. These regions are generally important for the function of a protein and/or the maintenance of its three-dimensional structure, or other features related to its localization or modification. By analyzing constant and variable properties of such groups of similar sequences, it is possible to derive a signature for a protein family or domain, which distinguishes its members from other unrelated proteins. Here we mention some examples of such domains that are essential to the immune response.

The immunoglobulin-like (Ig-like) protein domain is a domain of approximately 100 residues with a fold which consists of seven to nine antiparallel β strands. These β strands form a β-sandwich structure, consisting of three or four antiparallel β strands on each side of the barrel, connected by a sulfide bridge. The Ig-like domain is of special importance for the immune system. In addition to immunoglobulin, T cell receptor and MHC molecules carry Ig-like domains, i.e., the main players of the adaptive immune system have all Ig-like

domains. This is not a coincidence: the unique structure of this domain allows for maximum flexibility to interact with other molecules. This property makes the Ig-like domain one of the most widespread protein modules in the animal kingdom. This module has been observed in a large group of related proteins that function in cell-cell interactions or in the structural organization and regulation of muscles. The proteins in the Ig-like family consist of one or more of these domains.

Toll-like receptors (TLRs) are a family of pattern recognition receptors that are activated by specific components of microbes and certain host molecules. They constitute the first line of defense against many pathogens and play a crucial role in the function of the innate immune system.

That the field of immunology is almost as big, dispersed, and complicated as all the rest of the biology put together is exemplified by the fact that all the different fields of bioinformatics and sequence analysis are applied to immunological problems. Sequence alignment, structural biology, machine learning and predictive systems, pattern recognition, DNA microarray analysis, and integrative systems biology are all important tools in the research of the different aspects of the immune system and its interaction with pathogens.

3.2 Alignments

Sequence alignment is the oldest but probably the single most important tool in bioinformatics. Being one of the basic techniques within sequence analysis, alignment is, though, far from simple, and the analytic tools (i.e., the computer programs) are still not perfect. Furthermore, the question of which method is optimal in a given situation strongly depends on which question we want the answer to. The most common questions are: How similar (different) are this group of sequences, and which sequences in a database are similar to a specific query sequence. The reasoning behind the questions might, however, be important for the choice of algorithmic solution. Why do we want to know this? Are we searching for the function of a protein/gene, or do we want to obtain an estimate of the evolutionary history of the protein family? Issues like the size of database to search, and available computational resources might also influence our selection of a tool.

3.2.1 Ungapped Pairwise Alignments

From the early days of protein and DNA sequencing it was clear that sequences from highly related species were highly similar, but not necessarily identical. Aligning very closely related sequences is a trivial task and can be done manually (figure 3.1 A). In cases where genes are of different sizes and the similarity

A

```
               10        20        30        40        50        60        70
humanD MSEKKQPVDLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
       ::::::.:::::::::::::::::::.:::::::::::::::::::::::::::::::::::::::::::
gi|457 MSEKKQTVDLGLLEEDDEFEEFPAEDWTGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
               10        20        30        40        50        60        70
```

B

```
                   10        20        30        40        50        60        70
humanD ----MSEKKQPVDLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
           ....:...:::::::::::::::::::::..:::..........::....:..::..........
Anophe MSDKENKDKPKLDLGLLEEDDEFEEFPAEDWAGNKEDEEELSVWEDNWDDDNVEDDFNQQLRAQLEKHK-----
               10        20        30        40        50        60        70
```

Figure 3.1: A) The human proteasomal DSS1 subunit aligned against the zebra fish homolog using the identity matrix. B) The human proteasomal DSS1 subunit aligned to the mosquito homolog.

is less, alignments become more difficult to construct. In such cases it is also of great value to have a graduation of how related sequences are, i.e., a scoring scheme. The simplest scoring is the relative amount of identical entities, also called % identity, or %ID. This simple approach is actually too simple as ,e.g., amino acids share many physical-chemical properties, which means that they can more easily be exchanged than very unrelated amino acids. This means that a scoring system that scores different substitutions differently, a substitution matrix, is a much better approach. The most useful concept has been to estimate how often a given amino acid is exchanged for another in already aligned similar sequences. The most used are the *percentage accepted mutations* (PAM) matrix [Dayhoff et al., 1978] and the *blocks substitution matrix* (BLOSUM) [Henikoff and Henikoff, 1992].Mutations between different types of nucleotides or amino acids is not the only changes that appears in sequences during the evolution. The sequences can also loose or gain sequence entities (deletions or insertions, respectively). This also must affect a similarity score, but for simplicity these complications are left to later sections. The simplest way to calculate an alignment score is to make all the possible overlaps between two sequences, and sum the number of identical amino acids in the two sequences (ungapped alignment, figure 3.1 B).

Sequence alignment is essential to the *comparative immunology* field. The main research line in this field (so far) is to discover origins of the adaptive immune system. Thanks to the homology assessments using sequence alignments with mammalian equivalents of T cell receptors, MHC genes, cytokines, and antibodies, we now know that the adaptive immune system is well developed in the oldest jawed vertebrates, the sharks [Pasquier and Flajnik, 1999]. However, whether or not jawless invertebrates were in possession of such

adaptive immunity remains unresolved. The lamprey, which along with its cousin, the hagfish, is the only surviving jawless vertebrate, give immunologists a chance to pinpoint crucial aspects of the origin of the adaptive immune system. So far the search for antibodies, T cell receptors, and genes coding for MHC molecules has failed in these organisms. Recently, however, Pancer et al. [2004] have identified a set of uniquely diverse proteins that are only expressed by lamprey lymphocytes and named them variable lymphocyte receptors (VLRs). The sequence analysis of these proteins has revealed that the VLRs consist of multiple leucine-rich repeat (LRR) modules and an invariant stalk region that is attached to the lymphocyte plasma membrane. The remarkable VLR diversity derives from the variation in sequence and number of the LRR modules. The mature VLRs are thus generated through a process of somatic DNA rearrangement in lymphocytes. These results suggest a novel mechanism that does not involve recombinant-activating genes to generate the large diversity that an adaptive immune system is based upon.

3.2.2 Scoring Matrices

Dayhoff et al. [1978] calculated the original PAM matrices using a database of changes in groups of closely related proteins. From these changes they derived the accepted types of mutations. Each change was entered into a matrix listing all the possible amino acid changes. The relative mutability of different amino acids was also calculated, i.e., how often a given amino acid is changed to any other. The information about the individual kinds of mutations, and about the relative mutability of the amino acids were then combined into one "mutation probability matrix."

The rows and columns of this matrix represent amino acid substitution pairs, i.e., the probability that the amino acid of the column will be replaced by the amino acid of the row after a given evolutionary interval. A matrix with an evolutionary distance of 0 PAMs would have only 1s on the main diagonal and 0s elsewhere. A matrix with an evolutionary distance of 1 PAM would have numbers very close to 1 in the main diagonal and small numbers off the main diagonal. One PAM would correspond to roughly a 1% divergence in a protein (one amino acid replacement per hundred). Assuming that proteins diverge as a result of accumulated, uncorrelated, mutations a mutational probability matrix for a protein sequence that has undergone N percent accepted mutations, a PAM-N matrix, can be derived by multiplying the PAM-1 matrix by itself N times. The result is a whole family of scoring matrices. Dayhoff et al. [1978], imperically, found that for weighting purposes a 250 PAM matrix works well. This evolutionary distance corresponds to 250 substitutions per hundred residues (each residue can change more than once). At this distance

A

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

B

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

Figure 3.2: Substitution matrices. A) PAM250. B) BLOSUM62.

only one amino acid in five remains unchanged so the percent divergence has increased to roughly 80%. To avoid working with very small numbers the matrices actually used in sequence comparisons is logodds matrices. The odds matrix is constructed by taking the elements of the previous matrix and divide each component by the frequency of the replacement residue. In this way each component now gives the odds of replacing a given amino acid with another specified amino acid. Finally the log of this matrix is used as the weights in the matrix. In this it is now possible to sum up the scores for all positions to obtain the final alignment score. The PAM250 matrix is shown in Figure 3.2.

A

```
                    10        20        30        40        50        60        70
humanD -----MSEKKQPVDLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
            ..: .     : :.:.  :    .  .. ...:.  :::::::::::::::..::::.::::
Anophe MSDKENKDKPKLDLGLLEEDDEFEEFPAEDWAGNKEDEEELSVWEDNWDDDNVEDDFNQQLRAQLEKHK------
               10        20        30        40        50        60
```

B

```
                   10        20        30        40        50        60        70
humanD ----MSEKKQPVDLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
           ....:...:::::::::::::::::::..:::.........::....:..::.........
Anophe MSDKENKDKPKLDLGLLEEDDEFEEFPAEDWAGNKEDEEELSVWEDNWDDDNVEDDFNQQLRAQLEKHK-----
               10        20        30        40        50        60
```

Figure 3.3: (A) The human proteasomal subunit aligned to the mosquito homolog using the BLOSUM50 matrix. (B) The human proteasomal subunit aligned to the mosquito homolog using identity scores.

The BLOSUM matrix, described by Henikoff and Henikoff [1992], is another widely used amino acid substitution matrix. To calculate this, only very related blocks of amino acid sequences (conserved blocks) are considered. Originally these were taken from the BLOCKS database of prealigned sequence families [Henikoff and Henikoff, 1991]. Now the blocks are split up further in clusters, each containing the parts of the alignments that are more than X% conserved. The use of these clusters leads to a BLOSUMX matrix. That is, using clusters of down to 50% identities gives a BLOSUM50 matrix, and so forth. For every sequence in each cluster each position is compared to the corresponding position in each sequence in every other cluster. Since it is the *pairwise* number of frequencies that is calculated, the sum of all the substitutions is divided by the number of comparisons. In this way the result is the weighted probability that a given amino acid is exchanged for every other amino acid. In the final matrix, actually, the log ratio of the probability is further scaled so that the BLOSUM50 matrix is in thirds of bits, and the BLOSUM62 matrix is given in half-bits. The BLOSUM62 matrix is shown in figure 3.2.

Since the initial PAM1 matrix is made by very similar sequences, the evolutionary distances between those are very short, and most changes captured will be single base mutations leading to particular types of amino acid substitutions, while substitutions requiring more than one base mutation will be very rare. Even the calculations made to expand this matrix to longer evolution time cannot compensate for this [Gonnet et al., 1992] and therefore the BLOSUM matrices perform better when used for further distance alignment. The matrices are in a format where you can sum up the scores for each match to obtain a total alignment score, and the alignment resulting in the highest score is then the optimal one.

3.2.3 Gap Penalties

Using the BLOSUM50 matrix to align mosquito and human proteasomeal subunits (figure 3.3A) gives a slightly different alignment than just using amino acid identities (figure 3.3B). These two different alignments also reveal that there are two parts of the proteins with a high number of identical amino acids, but without inserting or deleting letters in one of the sequences they cannot be aligned simultaneously. This leads obviously to the necessity of inserting gaps in the alignments.

A gap in one sequence represents an insertion in the other sequence. First, to avoid having gaps all over the alignment these have to be penalized just like unmatching amino acids. This penalty (i.e., the probability that a given amino acid will be deleted in another related sequence) cannot be derived from the database alignments used to create the PAM and BLOSUM matrices, since these are ungapped alignments. Instead, a general gap insertion penalty is determined, usually empirically, and is often lower than the lowest match score. Having only one score for any gap inserted is called a linear gap cost, and will lead to the same total penalty for three single gaps at three different positions in the alignment as having a single stretch of three gaps. This does not make sense biologically, however, since insertions and deletions often involve a longer stretch of DNA in a single event. For this reason two different gap penalties are usually included in the alignment algorithms: one penalty for having a gap at all (gap opening penalty), and another, smaller penalty, for extending already opened gaps. This is called an affine gap penalty and is actually a compromise between the assumption that the insertion, or deletion, is created by one or more events. Furthermore, it is possible to let gaps appended at the ends of the sequences not to have a penalty, since insertions at the ends will have a much greater chance of not disrupting the function of a protein. For a more careful discussion of how to set gap penalties, see Vingron and Waterman [1994].

3.2.4 Alignment by Dynamic Programming

Introducing gaps greatly increases the number of different comparisons between two sequences and in the general case it is impossible to do them all. To compensate for that, several shortcut optimization schemes have been invented. One of the earliest schemes was developed by Needleman and Wunsch [1970] and works for global alignments, i.e., alignments covering all residues in both sequences. As an example, it is here described how to align two very short sequence stretches taken from our previous proteasome alignment. For simplicity, we will use the identity matrix (match=1, mismatch=-1) and a linear gap penalty of -2. Using the Needleman-Wunsch approach

Score matrix

		D		E		D		E		D		A		H		V		W	
	0																		
K																			
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Trace Matrix

		D		E		D		E		D		A		H		V		W	
	END																		
K																			
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Figure 3.4: Dynamic programming, global alignment. Step 1.

Score matrix

		D		E		D		E		D		A		H		V		W	
		0		-2															
K																			
		-2																	
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Trace Matrix

		D		E		D		E		D		A		H		V		W	
	END	-	right																
K	|																	
	up																		
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Figure 3.5: Dynamic programming, global alignment. Step 2.

Score matrix

		D		E		D		E		D		A		H		V		W	
		0		-2															
K																			
		-2		-1															
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Trace Matrix

		D		E		D		E		D		A		H		V		W	
	END	-	right																
K	|	\																	
	up		diagonal																
E																			
D																			
E																			
E																			
E																			
L																			
S																			
V																			
W																			

Figure 3.6: Dynamic programming, global alignment. Step 3.

Score matrix

		D		E		D		E		D		A		H		V		W	
	0		-2		-4		-6		-8		-10		-12		-14		-16		-18
K																			
	-2		-1		-3		-5		-7		-9		-11		-13		-15		-17
E																			
	-4		-3		0		-2		-4		-6		-8		-10		-12		-14
D																			
	-6		-3		-2		1		-1		-3		-5		-7		-9		-11
E																			
	-8		-5		-2		-1		2		0		-2		-4		-6		-8
E																			
	-10		-7		-4		-3		0		1		-1		-3		-5		-7
E																			
	-12		-9		-6		-5		-2		-1		0		-2		-4		-6
L																			
	-14		-11		-8		-7		-4		-3		-2		-1		-3		-5
S																			
	-16		-13		-10		-9		-6		-5		-4		-3		-2		-4
V																			
	-18		-15		-12		-11		-8		-7		-6		-5		-2		-3
W																			
	-20		-17		-14		-13		-10		-9		-8		-7		-4		-1

Trace Matrix

		D		E		D		E		D		A		H		V		W	
	END	-	left	-	left	-	left	-	left	-	left	-	left	-	left	-	left	-	left
K	|	\																	
	up		diagonal	-	left	-	left	-	left	-	left	-	left	-	left	-	left	-	left
E	|		|	\															
	up		up		diagonal		left	-	left	-	left		left	-	left	-	left	-	left
D	|	\	|		|	\													
	up		diagonal		up		diagonal	-	left	-	left		left	-	left	-	left	-	left
E	|		|	\			|	\											
	up		up		diagonal		up		diagonal	-	left		left	-	left	-	left	-	left
E	|		|		|		|		|	\								
	up		up		up		up		up		diagonal		left	-	left	-	left	-	left
E	|		|		|		|		|		\							
	up		up		up		up		up		up		diagonal		left	-	left	-	left
L	|		|		|		|		|		|	\						
	up		up		up		up		up		up		up		diagonal	-	left	-	left
S	|		|		|		|		|		|		|	\				
	up		up		up		up		up		up		up		up		diagonal	-	left
V	|		|		|		|		|		|		|	\		\		
	up		up		up		up		up		up		up		up		diagonal		diagonal
W	|		|		|		|		|		|		|		|	\	
	up		up		up		up		up		up		up		up		up		diagonal

Figure 3.7: Dynamic programming, global alignment, final matrices (Needleman-Wunsch).

[Needleman and Wunsch, 1970], we first define two identical matrices with the same number of columns as residues in sequence 1 and as many rows as residues in sequence 2 One matrix is used to keep track of the scores and another to keep track of our route (see figures 3.4-3.7).

- **Step 1 (figure 3.4):** In the upper left field of the score matrix is written the score 0. This is the score before having aligned anything. From this field we can move in three directions: Down corresponds to inserting a gap in sequence 1, left to inserting a gap in sequence 2 and diagonal to making a match. Accordingly, a step to the right is -2, a step down is -2, and a diagonal step is $+1$ if the residues are identical, otherwise -1.
- **Step 2 (figure 3.5):** With the limits of the steps, we can easily fill in the first row and the first column of the matrix, since these fields can only be reached from one direction. So in the score matrix we write -2 in field 0,1, since this step corresponds to inserting a gap. In the trace matrix we then write *up* in field 0,1 since this was the direction we were coming from. In field 1,0 we write -2 in the score matrix and *left* in the trace matrix.
- **Step 3 (figure 3.6):** Now we would like to calculate the score of field 1,1. Coming from the left we had -2 in the previous field $(0,1)$ and will have to add -2 for making a move to the right, inserting a gap in the *other* sequence, resulting in a score of -4. We do likewise if we would come down from field 1,0. We can now also make a diagonal move which means a match between the two first residues. In this example they are not identical and the match will have the score -1. Since we came from 0,0 with the score 0 the match case will result in -1. So we have the possibility to make three different moves resulting in a score of -4, -4, or -1, respectively. We now select the move resulting in the highest score (i.e., -1), and we write this score in field 1,1 in the score matrix. In the trace matrix we write *diagonal* in field 1,1 since this was the type of move made to reach this score.
- **Final steps:** Steps 2 and 3 are repeated until both matrices are filled out (figure 3.7). In the case that two different moves to a field result in the same score, we select the move *coming* from the highest previous score to write in the trace matrix. At any field, we will finally have a score. This score is then the maximal alignment score you can get coming from the upper left diagonal and to the position in the sequences matching that field.

When the matrices are all filled out, the final alignment score is in the lower right corner of the score matrix. In the above example the final alignment score

is then -1. The score matrix has now served its purpose and is discarded, and the alignment is reconstructed using the trace matrix. To reconstruct the alignment start in the lower right corner of the final trace matrix (figure 3.7). Following the directions written in the fields, the alignment is now reconstructed backward. Here *diagonal* means a match between the two last residues in each sequence (W match W), and a move diagonal up-left. Next field: *diagonal*, i.e., V match V and a move diagonal up-left. The present field value is now *up*: This means that we introduce a gap in the first sequence to match S in the second sequence and then move one field up in the trace matrix. The rest of the trace is all diagonal, which means no gaps, and the resulting alignment will be

```
DEDEDAH-VW
KEDEEELSVW
```

This way to produce an alignment is called dynamic programming, and is still used in major alignment software packages (e.g., the ALIGN tool in the FASTA package uses the Needleman-Wunsch algorithm for global alignments). To illustrate that there *are* differences in the resulting alignments according to which scoring scheme is used, the above alignment using the BLOSUM62 matrix in figure 3.2 and a linear gap penalty of -9 results in the following alignment

```
DEDEDA-HVW
KEDEEELSVW
```

So the optimal alignment is only optimal using the chosen substitution scores and gap penalties, and there is no exact way to tell in a particular example if one set of scores gives a more “correct” alignment than another set of scores.

3.2.5 Local Alignments and Database Searches

The global alignment scheme described above is very good for comparing and analyzing the relationship between two selected proteins. Proteins, however, are often comprised of different domains, where each domain may be evolutionarily related to a different set of sequences. Thus when it comes to *searching* for sequences it is more beneficial to only look at the parts of the sequences that actually are related. A search is actually to make pairwise alignment of your query sequence to all the sequences in the database, and order the resulting alignments by the alignment score. For this purpose Smith and Waterman [1981] further developed the dynamic programming approach. The Smith-Waterman algorithm is like Needleman-Wunsch, except that the traces only continue as long as the scores are positive, Whenever a score becomes negative it is set to 0 and the corresponding trace is empty. Using the BLOSUM62 substitution matrix and a linear gap penalty of -9, the score and trace

Score matrix

		D		E		D		E		D		A		H		V		W	
	0		0		0		0		0		0		0		0		0		0
K																			
	0		0		1		0		1		0		0		0		0		0
E																			
	0		2		5		3		5		3		0		0		0		0
D																			
	0		6		4		11		5		11		2		0		0		0
E																			
	0		2		11		6		16		7		10		2		0		0
E																			
	0		2		7		13		11		18		9		10		1		0
E																			
	0		2		7		9		18		13		17		9		8		0
L																			
	0		0		0		3		9		14		12		14		10		6
S																			
	0		0		0		0		3		9		15		11		12		7
V																			
	0		0		0		0		0		0		9		12		15		9
W																			
	0		0		0		0		0		0		0		7		9		26

Trace Matrix

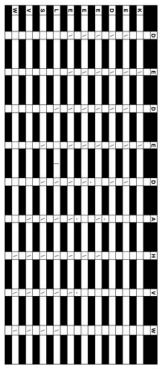

Figure 3.8: Dynamic programming, local alignment, final matrices (Smith-Waterman).

matrices will appear as in Figure 3.8. Now the backtrace of the optimal local alignment starts in the field with the highest score. There might be several equally good alignments, and there are several ways to deal with that, depending on what the goal is. If the two equally good alignments differ in length, one might, e.g., chose the longer. In this example the highest score is 26. This is accidentally again in the lower right corner so the backtrace will begin here. The backtrace will reveal that the local alignment look like this:

```
DEDEDAHVW
EDEEELSVW
```

BLAST The dynamic programming algorithm has the strength that it ensures that the optimal alignment, will always be found, given specific gap penalties and substitution scores. However, even with present-day computerpower this algorithm is far too slow to search the ever-increasing sequence databases of today. For this reason several shortcuts have been made, and one of the most successful is implemented in the widely-used alignment package, BLAST [Altschul et al., 1990, 1997, Altschul and Gish, 1996].

The basic BLAST algorithm consists of 3 steps:

1. **Make a list of words:** A list of neighbor words that have a score of at least T (default 11 for proteins) is made for each n-mer in the query sequence. Per default n=3 for proteins and n=11 for DNA. Any word in the query sequence that scores positive with itself may also be included.

2. **Search the database for the words on the list:** The database is scanned for hits to any of the N words on the list.

3. **Extend hits:** The first version of BLAST extended every hit it found. The newer version requires two nonoverlapping hits within a distance A (default 40) of each other before it extends a hit. The extension is only made until the score has dropped X (default 7) below the best score seen so far. This corresponds to saying this route looks so bad that there is no point in continuing in this direction. The locally optimal alignments are called high-scoring segment pairs (HSPs). If the score of an HSP is above a threshold S_g (default 22 bits) a gapped extension is attempted using dynamic programming. To speed the calculations this phase is only continued until the score falls X_g below the best score seen so far.

3.2.6 Expectation Values

When aligning two sequences it is not clear if a given score is really significant (i.e., might occur by chance by a certain probability). Such a measure can be

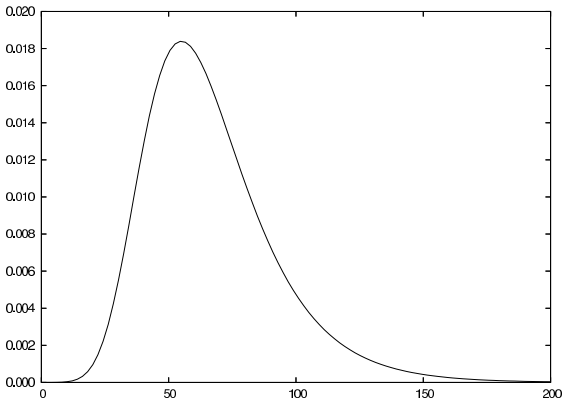

Figure 3.9: Distributions of scores, when aligning a sequence to a database of unrelated sequences.

obtained by aligning a great number of random sequences to the original sequence and from the resulting score distribution calculate the probability that a random sequence would result in a given score. This number is called the expectation-value, or E-value. The random sequences is obtained by shuffling the elements (nucleotides or amino acids) of the original sequence. In this way the score distribution will not be biased by a skewed amino acid distribution of the original sequence.

When searching through databases the question also arises whether a given alignment score confers a relationship between the two aligned regions or not. If we align a sequence to a database of all unrelated sequences and plot the alignment score against how many alignments will have that score we will get a curve like that in figure 3.9. This is called an extreme value distribution. We can from this distribution find out how often a given alignment-score will arise by chance. Thus the E-value is the theoretically expected number of false hits per sequence query, and a lower E-value means a more significant hit. Importantly, the E-value is dependent on the size of the database searched as the chance of getting a false hit rises as the database grows.

Different alignment programs use different approaches to calculate the E-value of a given database hit. FASTA actually makes all possible alignments, and returns a real distribution curve (figure 3.10) and calculates the E-value

```
       opt      E()
< 20     0     0:
  22     0     0:            one = represents 23 library sequences
  24     0     0:
  26     0     0:
  28     0     3:*
  30     0    16:*
  32     7    64:= *
  34    75   173:====   *
  36   240   354:===========    *
  38   569   586:=======================*
  40  1127   817:===================================*=============
  42  1379   999:============================================*================
  44  1277  1102:=============================================*========
  46  1183  1122:=========================================================*===
  48   914  1074:======================================       *
  50   733   980:==============================           *
  52   753   862:================================    *
  54   661   736:============================  *
  56   516   615:=====================   *
  58   536   505:====================*==
  60   365   409:================ *
  62   335   328:==============*
  64   273   261:===========*
  66   188   206:========*
  68   168   162:=======*
  70   126   127:=====*
  72   133    99:====*=
  74    88    77:===*
  76    68    60:==*
  78    56    47:==*
  80    41    36:=*
  82    41    28:=*
  84    34    22:*=
  86    16    17:*
  88    13    13:*           inset = represents 1 library sequences
  90    12    10:*
  92     6     8:*          :====== *
  94     4     6:*          :==== *
  96     3     5:*          :=== *
  98     4     4:*          :===*
 100     2     3:*          :==*
 102     0     2:*          : *
 104     0     2:*          : *
 106     1     1:*          :*
 108     2     1:*          :*=
 110     0     1:*          :*
 112     2     1:*          :*=
 114     0     0:           *
 116     0     0:           *
 118     0     0:           *
>120     0     0:           *
4113207 residues in 11951 sequences
 Expectation_n fit: rho(ln(x))= 5.3517+/-0.00135; mu= -2.1992+/- 0.077;
 mean_var=60.8388+/-13.111, Z-trim: 5  B-trim: 3 in 1/55
 Kolmogorov-Smirnov  statistic: 0.0520 (N=29) at  46
```

Figure 3.10: Distributions of scores, from FASTA alignments of a given sequence to all sequences in a specific database.

making a fit to this curve. BLAST, however, uses a premade empirical curve to assign E-values to each alignment returned from a database search.

PSI-BLAST As described earlier, the scoring matrices used somehow represent the general evolutionary trends for mutations. However, in reality, allowed mutations are very much dependent on, and constrained by their physical context. As an example, it could be possible to insert, delete, or exchange a number of different amino acids in a flexible loop on the surface of a protein and still preserve the overall structure and function of the protein.

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

Figure 3.11: Example of a PSSM.

The corresponding number of allowed substitutions would very probably be much more limited in the core — or in a secondary structure, rich — region of the protein. So if a *general* substitution matrix works well, a matrix representing the *specific* evolutionary trend for a given position in a given protein should work even better. As described by Altschul et al. [1997], this is actually the case.

In the PSI-BLAST approach, first an ordinary BLAST search on the basis of the BLOSUM62 matrix is performed against the database. Second, a position-specific scoring matrix (PSSM) is calculated as described in chapter 4. The matrix is calculated by considering the substitutions observed in pairwise alignments made between the query sequence and the hits that have an expectation value below a selected threshold. Now the calculated matrix (figure 3.11), as a representation of the query sequence, is used to search the database again. So when the alignment score matrix is filled out, we now look in the PSSM for a given position to find the match score between the PSSM and that particular amino acid in the database sequence. For example, if we want to match position 3 in the search sequence, a glutamic acid, to an alanine, the match score is 5. However, if we want to match position 4, also a glutamic acid, to an alanine, the match score is −4. This should illustrate the higher specificity of a PSSM as compared to ordinary substitution matrices.

3.3 Multiple Alignments

When looking at several related sequences, it is often useful and informative to look at all the sequences in one alignment (multiple alignment). The simplest approach is to align all the sequences, one by one, with a single selected "master sequence," and this is what can be obtained by programs like BLAST. However, these programs make only local alignments, and often gaps and in-

A

```
Drosophila_melanogaster       MSAPDKEKEKEKEETNNKSEDLGLLEEDDEFEEFPAEDFRVGDDEEELNVWEDNWDDDNVEDDFSQQLKAHLESKKMET
Anopheles_gambiae             ----------DKENKDKPKLDLGLLEEDDEFEEFPAEDWAGneDEEELSVWEDNWDDDNVEDDFNQQLRAQLEKHK---
Zebrafish                     ------------------QTVDLGLLEEDDEFEEFPAEDWTGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELE------
HUMAN                         ----------------------DLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELE------
MOUSE                         ----------------------DLGLLEEDDEFEEFPAEDWAGLDEDEDAHVWEDNWDDDNVEDDFSNQLRAELE------
Xenopus_laevis                ----------------------DLGLLEEDDEFEEFPTEDWTGFDEDEDTHVWEDNWDDDNVEDDFSNQLRAELE------
Saccharomyces_cerevisiae      -------------------------LEEDDEFEDFPIDTWANGETIkqTNIWEENWDDVEVDDDFTNELKAELDRYKRE-
Neurospora_crassa.            ----DAKSTEPKPEQPVTEKKTAVLEEDDEFEDFPVDDWEAEDTeeAKHLWEESWDDDDTSDDFSAQLKEELK------
```

B

```
Drosophila_melanogaster       ----MSAPDKE----KEKEKEETNNKSEDLGLLEEDDEFEEFPAEDFRVG
Anopheles_gambiae             ----MS--DKEN---KDKPK-------LDLGLLEEDDEFEEFPAEDWAGN
HUMAN                         ----MS----------EKKQ------PVDLGLLEEDDEFEEFPAEDWAGL
MOUSE                         ----MS----------EKKQ------PVDLGLLEEDDEFEEFPAEDWAGL
Zebrafish                     ----MS----------EKKQ------TVDLGLLEEDDEFEEFPAEDWTGL
Xenopus_laevis                ---MSS----------DKKP------PVDLGLLEEDDEFEEFPTEDWTGF
Neurospora_crassa.            ----MASTQPKNDAKSTEPKPEQPVTEKKTAVLEEDDEFEDFPVDDWEAE
Saccharomyces_cerevisiae      MSTDVAAAQAQSKIDLTKKKNE----EINKKSLEEDDEFEDFPIDTWANG
                                   :          :          .   ********:** : :

Drosophila_melanogaster       ------DDEEELNVWEDNWDDDNVEDDFSQQLKAHLESK--KMET-
Anopheles_gambiae             K-----EDEEELSVWEDNWDDDNVEDDFNQQLRAQLEKH--K----
HUMAN                         ------DEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
MOUSE                         ------DEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
Zebrafish                     ------DEDEDAHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
Xenopus_laevis                ------DEDEDTHVWEDNWDDDNVEDDFSNQLRAELEKHGYKMETS
Neurospora_crassa.            DTEAAKGNNEAKHLWEESWDDDDTSDDFSAQLKEELKKVEAAKKR-
Saccharomyces_cerevisiae      ETIKS-NAVTQTNIWEENWDDVEVDDDFTNELKAELDRY--KRENQ
                                        :**:.*** :..***. :*: .*.
```

C

```
HUMAN              1   ---------- ---------- ------MSEK KQPVDLGLLE EDDEFEEFPA
MOUSE              1   ---------- ---------- ------MSEK KQPVDLGLLE EDDEFEEFPA
Zebrafish          1   ---------- ---------- ------MSEK KQTVDLGLLE EDDEFEEFPA
Drosophila_m       1   ----MSapDK Ek-------E KEKEET-NNK SE--DLGLLE EDDEFEEFPA
Neurospora_c       1   ----MA--ST QPKNDAKSTE PKPEQpVTEK KTAV----LE EDDEFEDFPV
Xenopus_laev       1   m--------- ---------- -----S-SDK KPPVDLGLLE EDDEFEEFPT
Saccharomyce       1   mstdVA--AA QAQSKIDLTK KKNEEI-NKK S-------LE EDDEFEDFPI
Anopheles_ga       1   ----MS--DK ENKD------ ---------- KPKLDLGLLE EDDEFEEFPA

HUMAN             25   EDWAGLDE-- ----DED-AH VWEDNWDDDN VEDDFSNQLR AELEK----H
MOUSE             25   EDWAGLDE-- ----DED-AH VWEDNWDDDN VEDDFSNQLR AELEK----H
Zebrafish         25   EDWTGLDE-- ----DED-AH VWEDNWDDDN VEDDFSNQLR AELEK----H
Drosophila_m      37   EDFRVGDD-- ----EEE-LN VWEDNWDDDN VEDDFSQQLK AHLES----K
Neurospora_c      41   DDWEAEDtEA AKGNNEA-KH LWEESWDDDD TSDDFSAQLK EELKKveaaK
Xenopus_laev      26   EDWTGFDE-- ----DED-TH VWEDNWDDDN VEDDFSNQLR AELEK----H
Saccharomyce      41   DTWAng--ET IKSNavtqTN IWEENWDDVE VDDDFTNELK AELDR----Y
Anopheles_ga      29   EDWAGNKE-- ----DEEeLS VWEDNWDDDN VEDDFNQQLR AQLEK----H

HUMAN             64   GYKMETS
MOUSE             64   GYKMETS
Zebrafish         64   GYKMETS
Drosophila_m      76   --KMET-
Neurospora_c      90   --Kr---
Xenopus_laev      65   GYKMETS
Saccharomyce      85   --KRENQ
Anopheles_ga      69   --K----
```

Figure 3.12: Multiple alignments of the proteasome DSS1 subunit from different organisms using A) PSI-BLAST, B) ClustalW, and C) DIALIGN. Lower case letters means a part of the sequence that is not significantly aligned.

sertions will be placed differently in the master sequence depending on which other sequence it is aligned with. Another approach is to align all sequences pairwise with all other sequences and establish the difference between every pair. Such a map is called a distance matrix, and from this it is possible to obtain an estimate of which sequences are most related (a cluster), and aligning those first, and then align all the prealigned clusters against each other. This is basically what is implemented in the most used multiple alignment program, ClustalW alias ClustalX [Thompson et al., 1994]. First is calculated a score for the alignment between each pair of the sequences. These scores are then used to calculate phylogenetic tree, or a dendrogram, using the clustering method UPGMA (see Chapter 5). Having calculated the dendrogram, the sequences are aligned in larger and larger groups. Each of these alignments consists of aligning 2 alignments, using profile alignments, which are the alignment of 2 groups of already aligned sequences. The method is an extension of the profile method of Gribskov et al. [1987] for aligning a single sequence with an aligned group of sequences. With a sequence-to-sequence alignment, a weight matrix such as BLOSUM62 is used to obtain a score for a particular substitution between the pairs of aligned residues. In profile alignments, however, each of the two input alignments are treated as a single sequence, but you calculate the score at aligned positions as the average substitution matrix score of all the residues in one alignment vs. all those in the other, e.g., if you have 2 alignments with I and J sequences respectively the score at any position is the average of all the I times J scores of the residues compared separately. Any gaps that are introduced are placed in all of the sequences of an alignment at the same position. However, all gaps in the ends of the sequences are free. This might give some artifacts, especially when sequences of different length are aligned. Newer multiple alignment algorithms implemented in programs such as T-Coffee [Notredame et al., 2000] and DIALIGN [Morgenstern, 1999] handle these problems much better, but the algorithms behind them will not be described in this book. Figure 3.12 is an example of the differences in the results, using different alignment algorithms/programs. Note that PSI-BLAST will only return local alignments, and that the result is based on pairwise alignments to the query sequence, i.e., no clustering has been involved.

3.4 DNA Alignments

Untill now only protein alignments have been described. The basic algorithms and programs used for DNA alignment, however, are the same as for proteins. DNA alignments are much more difficult since at each position, we can have one of only four different bases as opposed to one of twenty in peptide alignments. So we will not have a specific substitution matrix like BLOSUM or PAM

but rather take a step back and use a general substitution score for any match or mismatch but still using affine gap penalties. This makes the probability of any given substitution equally high, and so the significance of the final alignment will be lower. Some nucleotide matrices, however, do have different substitution scores for transitions (Dealing with DNA/RNA sequences from coding regions, however, gives an opportunity to shortcut the alignment by actually aligning the translation products, rather than the actual DNA sequences. This approach has been implemented in most alignment software packages, including FASTA (tfasta [Pearson and Lipman, 1988, Pearson, 1996]) and BLAST (tblast [Altschul et al., 1990, Altschul and Gish, 1996]). In this basic but strong approach, gaps in the aligned DNA sequences will only occur in multiples of triplets. This will, however, not catch examples correctly where frameshifts have actually happened, leading to major changes of larger or smaller parts of the translated protein. For such investigations the programs GenA1 [Hein and Støvlbaek, 1994, 1996] and COMBAT [Pedersen et al., 1998] can be used, but only for pairwise alignments. For multiple alignments an automatic method exists that will translate DNA to peptide, do the multiple alignment using DIALIGN [Morgenstern, 1999], and return the final alignment at the DNA level [Wernersson and Pedersen, 2003]. Multiple DNA alignments are especially useful for investigating the evolution on the molecular level (molecular evolution). With such alignments it is possible to examine exactly which positions in the DNA are more or less likely to undergo mutations that survive and are transferred to the progeny. We can also calculate the chance that a given codon will only allow mutations that will not lead to an amino acid change (silent mutations or synonymous mutations) and compare it to the chance that a substitution leads to an amino acid change (nonsynonymous mutations). This ratio is called dN/dS and an example of such a calculation is given in chapter 7.

3.5 Molecular Evolution and Phylogeny

Phylogenies reveal evolutionary relationships between organisms and specific sequences. In recent years molecular phylogenies have started to play a major role in epidemiological studies of pathogens. These studies provide information about where and when a virulent strain can arise. Not only human pathogens but also viral and bacterial disease-causing agents of livestock are of importance, as such outbreaks can cause great economic loss, as well as increase the chance of a possible cross-species infection. Recent developments of new methods for isolating, amplifying, and sequencing RNA isolated from small samples of blood or tissue have made the molecular phylogeny of pathogens a rapidly expanding research field. Moreover, since many pathogens can mutate at much higher rates than eukaryotes, it is possible to obtain the

phylogeny of sequences that diverged only recently.

One interesting application of molecular phylogeny is represented by analysis of the origins of HIV epidemics. Exactly when simian immunodeficiency virus (SIV) was transmitted from nonhuman primates to humans, giving rise to the human immunodeficiency virus (HIV), is still under investigation. Korber et al. [2000] used a phylogenetic analysis of the viral sequences with a known date of sampling to estimate the year of origin for the main group of HIV viruses (HIV-1 M), the principal cause of acquired immunodeficiency syndrome (AIDS). AIDS is caused by two divergent viruses, HIV-1 and HIV-2. HIV-1 is responsible for the global pandemic, while HIV-2 has, until recently, been restricted to West Africa and appears to be less virulent in its effects. SIV viruses related to HIV have been found in many species of nonhuman primates. By analyzing the molecular divergence of the envelope gene, and applying a model which assumes constant mutation rates through time and across lineages, Korber et al. [2000] estimated that the last common ancestor of the HIV-1 M group appeared in 1931 (with a confidence interval of 1916–1941). Using a different molecular clock analysis, where the mutation rate is allowed to change at splitting events, and also when analyzing a different protein, the same estimates were obtained. This approach only identifies when the common ancestor began to diversify; it does not identify the exact time of transmission. Still, given this estimate, one is able to come up with more precise hypotheses about the transmission event.

3.5.1 Phylogenetic Methods

The starting point of any phylogenetic work is a collection of sequences that might be evolutionarily related. Such a set could be extracted from public databases using some of the tools described previously, or it could be data from one's own work. These sequences must now be aligned by the use of multiple alignment software, such as ClustalW. ClustalW also calculates a distance matrix of your sequences, i.e., the relation of each of your sequences to the other sequences in your alignment. A way to visualize the distances in a distance matrix is a tree-like drawing where the distances along the branches correlates with the distances in the distance matrix. Such a drawing is called a phylogenetic tree. One important point about trees is that they are only useful if the described system has been under vertical evolution (i.e., no horizontal gene transfers and recombination), otherwise a simple tree makes no sense. To calculate the grouping and the branch lengths of such a tree, two major approaches are applicable. One approach is optimization methods that will find the tree that gives the optimal fit to the matrix, e.g., the minimal sum of squared errors. Another approach is clustering methods that is related to the

optimization methods, but is much faster. The clustering methods, however, do not guarantee the optimal solution.

Two major types of trees exist: rooted and unrooted trees. With rooted trees a common ancestor point is used as the origin of the tree, no matter if this is really scientific sane with the given data. In rooted trees the horizontal distance from the leaves to the origin is directly proportional to the amount of changes. Unrooted trees are used to show relations where no common ancestor is given, and only the evolutionary distance between the leaves can be inferred. In both rooted and unrooted trees, the leaves are grouped in clusters. This grouping depends heavily on the algorithm used. Some algorithms just give one of potentially many, more or less equally probable, outputs. Other approaches actually calculate many different solutions and give the most probable outcome with some indication of how reliable a particular solution is.

As a simple example, we will investigate the phylogenetic relationship between HIV and SIV using a set consisting of 27 different gp120 protein sequences from isolates of HIV-1, HIV-2, chimpanzee SIV, and macaque monkey SIV. The gp120 protein of HIV is crucial for binding of the virus particle to target cells. It is the specific affinity of gp120 for the CD4 protein that targets HIV to those cells of the immune system that express CD4 on their surface (e.g., helper T lymphocytes, monocytes, and macrophages). ClustalW is used to align the sequences (figure 3.13) and, as mentioned earlier, ClustalW also clusters the most related sequences. The information from this clustering can subsequently be used to produce a phylogenetic tree (figure 3.14).

The phylogenetic tree from the analysis (see figure 3.14) shows two separate clusters. One contains SIV from chimpanzee (SIVCZ) together with the HIV-1 sequences, while the other contains SIV from macaque/sooty mangabey together with HIV-2. This indicates that HIV-1 originated from one event where the virus was transmitted from (presumably) chimpanzee to human, while HIV-2 originated from a second, independent event where the virus was transmitted from (presumably) macaque to human.

3.6 Viral Evolution and Escape: Sequence Variation

Coexistence of pathogens with their hosts imposes an evolutionary pressure both for the host immune systems and the pathogens. The coexistence depends on a delicate balance between the replication rate of the pathogen and the clearance rate by the host immune response. Throughout the animal and plant kingdoms we see several quite different strategies developed by the host immune systems to defend themselves against intruders. Similarly, the pathogens have developed an array of immune evasion mechanisms to escape their elimination by the host's immune system.

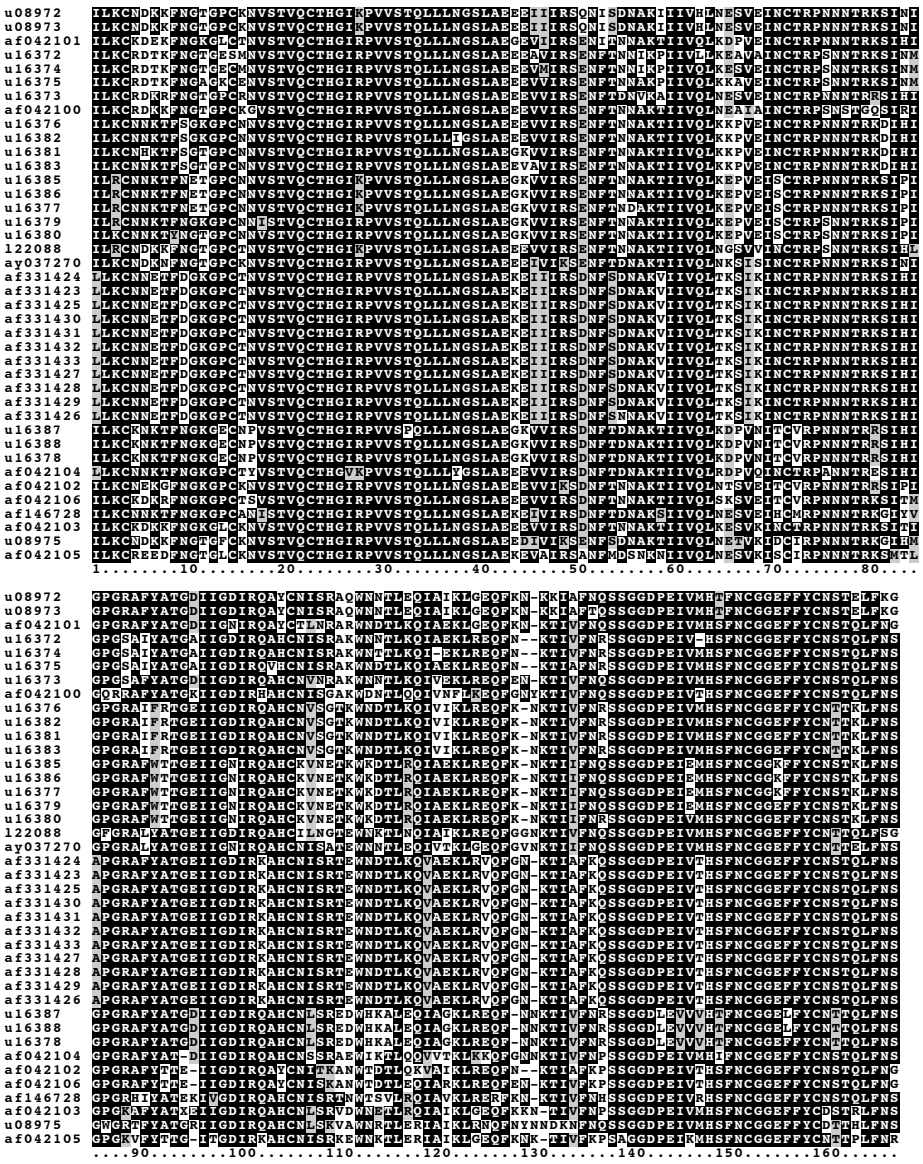

Figure 3.13: ClustalW alignment of 27 HIV/SIV gp120 sequences. The output is modified with the BOXSHADE program.

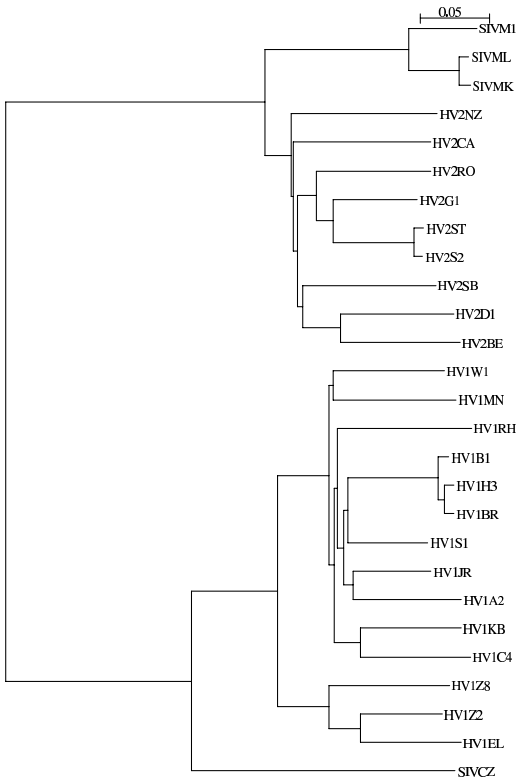

Figure 3.14: A rooted tree of 27 aligned HIV/SIV gp120 sequences. HV1XX=HIV-1 sequences, HV2XX=HIV-2 sequences, SIVMX=SIV (macaque), SIVSX=SIV (sooty mangabey), SIVCZ=SIV (chimpanzee).

We can divide the immune evasion mechanisms (mainly of viruses) broadly into three categories that allow:

1. avoiding the humoral immune response,

2. interfering with the cellular immune response,

3. disrupting the immune effector functions, e.g., by expressing some cytokines.

The humoral response is impaired whenever the antibody binding sites on a protein (often on the surface) mutate in such a way that binding is no longer possible. Especially neutralizing antibodies, i.e., the antibodies that can block infection of the cells by the pathogen, cause a high selection pressure on the virus to mutate. The most straightforward way of identifying such mutants is via sequence analysis of the pathogenic samples. The first step is to align the sequences to pinpoint which regions of the pathogen are mutating. This may be the region that is under the strongest selection pressure by the antibodies. However, it could also be areas with no constrains. Such alignments demonstrate that the most typical examples of escape from antibody response occur in the influenza virus and HIV. The human body can rapidly mount neutralizing antibodies against the major surface protein of the influenza surface protein, hemagglutinin. The influenza virus evades this humoral response by two mechanisms [Gorman et al., 1992]. First, using point mutations, the viral variants can escape neutralization, but this does not cause severe disease, since there will still be some unaltered epitopes that can be recognized. Second, if RNA segments are exchanged between different strains, the hemagglutinin protein can gain a totally different structure. In such a case, the antibodies made during previous infections are no longer functional and severe pandemics can occur [Claas and Osterhaus, 1998]. Interestingly, the phylogenetic analysis of the hemagglutinin protein shows that the antigenic evolution of the influenza virus is punctuated, i.e., some mutants cause epidemics for almost eight consecutive years, while others last only for two or three years [Smith et al., 2004]. Since the 1960s (when the first sequences were collected) every viral mutant has been able to cause an epidemic for at least two years, after which enough individuals will have acquired immunity to limit the spread significantly (herd immunity).

Similarly, the cytotoxic T lymphocyte (CTL) response can be abrogated whenever peptide binding of MHC molecules or binding of the T cell receptor to the MHC-peptide complex is disturbed. It is relatively difficult to observe such escapes, because they are different for each individual, depending on her or his MHC background. Therefore many CTL escape variants can be circulating in a host population without one becoming the dominant mutant. Only in chronic infections like HIV and hepatitis B is it possible to find these escape mutants in a patient. Again, for HIV we have an extensive amount of data to analyze CTL escape mutants. Using sequence analysis it is possible to see that escape mutations are not spread all over the viral genome, because HIV is not able to tolerate changes equally well in all proteins. HIV has very flexible proteins like the envelope protein, gp160, where up to 35% of the sequence can be different from the wild-type virus [Gaschen et al., 2002]. On the other hand, for some proteins, like capsid protein p24, the surface cannot tolerate point mutations without a severe loss of viral fitness [von Schwedler et al., 2003,

Leslie et al., 2004].

An effective vaccine should be able to target the parts of a pathogenic genome that are quite conserved even under the above-mentioned selection pressures. For example, given that less than a 2% amino acid change can cause a failure in cross-reactive immunity of the influenza vaccine [Korber et al., 2001b], it is obvious that for an HIV vaccine to use the envelope protein would be futile. One approach to deal with such large diversity is to use the consensus or the ancestral virus sequence as a vaccine. Such sequences have the advantage of being central and most similar to circulating strains. Another, safer approach would be to design epitope vaccines, which again requires choosing the most conserved epitopes. But the selection of such epitopes also requires computational analysis that goes beyond what simple sequence comparison techniques can handle, as the binding specificities are influenced by correlations between amino acids present at different peptide positions. A solution to this problem is to use machine learning techniques (see chapter 5).

3.7 Prediction of Functional Features of Biological Sequences

During experimental analysis of the immune system, proteins of unknown function are typically being identified as key players using high-throughput gene expression or proteomics data. The functional assignment of such immune system–related proteins also often requires sequence analysis that goes beyond what can be solved by simple sequence alignment methods. In most genomes no more than 40 to 60% of the proteins can be assigned a functional role based on sequence similarity to proteins with known function. Traditionally, protein function has been related directly to the 3D structure of the protein chain of amino acids, which currently, for an arbitrary sequence, is quite hard (in the general case, impossible) to compute. As the sequence, in a given biochemical context, determines the structure, functional information between two sequences can be transferred by comparing the sequence of amino acids by aligning the two against each other. This method is fast and powerful, but only solves part of the problem: it is still impossible to determine that two quite different sequences encode proteins with essentially the same biochemical function.

Several different methods have been developed which do not rely on direct sequence similarity, but on features which go beyond sequence-wide similarity, such as the gene position in the genome, or integration of local or global protein features. One such method, ProtFun, does not, like sequence alignment, compare any two sequences, but operates in the “feature” space of all sequences. ProtFun is therefore complementary to methods based on alignment and the inherent, position-by-position quantification of similarity

between two sequences and their amino acids [Jensen et al., 2002, 2003]. This particular method is still entirely sequence-based and does not require prior knowledge of gene expression, gene fusion, or protein-protein interaction.

For any function assignment method, the ability to correctly predict the functional relationship depends strongly on the function classification scheme used. One would, e.g., not expect that a method based on coregulation of genes will work well for a category like "enzyme," since enzymes and the genes coding for their substrates or substrate transporters often display strong coregulation at the gene and protein levels.

The ProtFun approach to function prediction is based on the fact that a protein is not alone when performing its biological task. It will have to operate using the same cellular machinery for modification and sorting as all the other proteins do. Essential types of post-translational modifications (PTMs) include glycosylation, phosphorylation, and cleavage of N-terminal signal peptides controlling the entry to the secretory pathway, but hundreds of other types of modification exist (a subset of these will be present in any given organism). Many of the PTMs are enabled by local consensus sequence motifs, while others are characterized by more complex patterns of correlation between the amino acids close or far apart in the sequence.

This suggests an alternative approach to function prediction, as one may expect that proteins performing similar functions would share some attributes even though they are not at all related at the global level of amino acid sequence. As several powerful predictive methods for PTMs and localization have been constructed, a function prediction method based on such attributes can be applied to all proteins where the sequence is known.

3.7.1 The ProtFun Method

The ProtFun method integrates (using an artificial neural network approach; see chapter 5 for a general introduction) many individual attribute predictions and calculated sequence statistics (out of many more tested for discriminative value) (see figure 3.15). The integrated method predicts functional categories which can be defined in various ways. The method predicts, e.g., whether a sequence is likely to function as an enzyme, and if so, its category according to the classes defined by the Enzyme Commission. The same scheme can be used to predict any other set of functional classes, including highly specific ones, such as "ligand gated ion channel." It can, for example, be used to identify hormones, growth factors, receptors, and ion channels in the human genome as defined by the Gene Ontology Consortium gene function classification scheme. Obviously, even though such methods produce predictions with false positives and false negatives, they can provide essential clues, e.g., to selecting an assay

if the confidence scores are sufficiently high.

The method uses combinations of attributes as input to the neural network for predicting the functional category of a protein. Combinations of attributes can be selected by evaluating their discriminative value for a specific functional category, say proteins involved in transcription or proteins being transporters. Attributes useful for function prediction must not only correlate well with the functional classification scheme, but must also be predictable from the sequence with reasonable accuracy.

Interestingly, the combinations of attributes selected for a given category also implicitly characterize a particular functional class in an entirely new way. This type of method identifies, without any a priori ranking of their importance, the biological features relevant to a particular type of functionality, say attributes which are discriminative for two different categories of ion channels.

The success of the method indicates that (even predicted) PTMs correlate strongly with the functional categories and this fits well with general biological knowledge. For proteins with "regulatory function" one of the most important features turned out to be phosphorylation, consistent with the fact that reversible phosphorylation is a well-known and widely used regulatory mechanism. Glycosylation was also found to be a strong indicator for regulatory proteins. The most important single feature for distinguishing between enzymes and nonenzymes turned out to be predicted protein secondary structure. This also makes sense, as enzymes are known to be overrepresented among all-alpha proteins where the amino acid chain forms an alpha-helix structure, and more rarely are found to be all-β proteins, where the structure is rich in β-sheet.

3.7.2 Individual Sequence Prediction

The ProtFun method can be used to characterize the entire genome, but it is perhaps best suited for obtaining functional hints for individual sequences for later use in assay selection and design. As an example we can take the human prion sequence which is being associated with the Creutzfeldt-Jacob disease. The functionality of this protein, which seems to produce no phenotype when knocked out in mice, was for a long time not fully understood. The ProtFun method predicts (see figure 3.16) with high confidence that the human prion sequence belongs to the transport and binding category, and also that it is very unlikely to be an enzyme. Indeed, prions have now been shown to be able to bind and transport copper, while no catalytic activity has ever been observed. Interestingly, as the prion is a cell surface glycoprotein (expressed by neural cells) it has a distinct pattern of post-translational modification, which most likely contains information which can be exploited by the prediction method

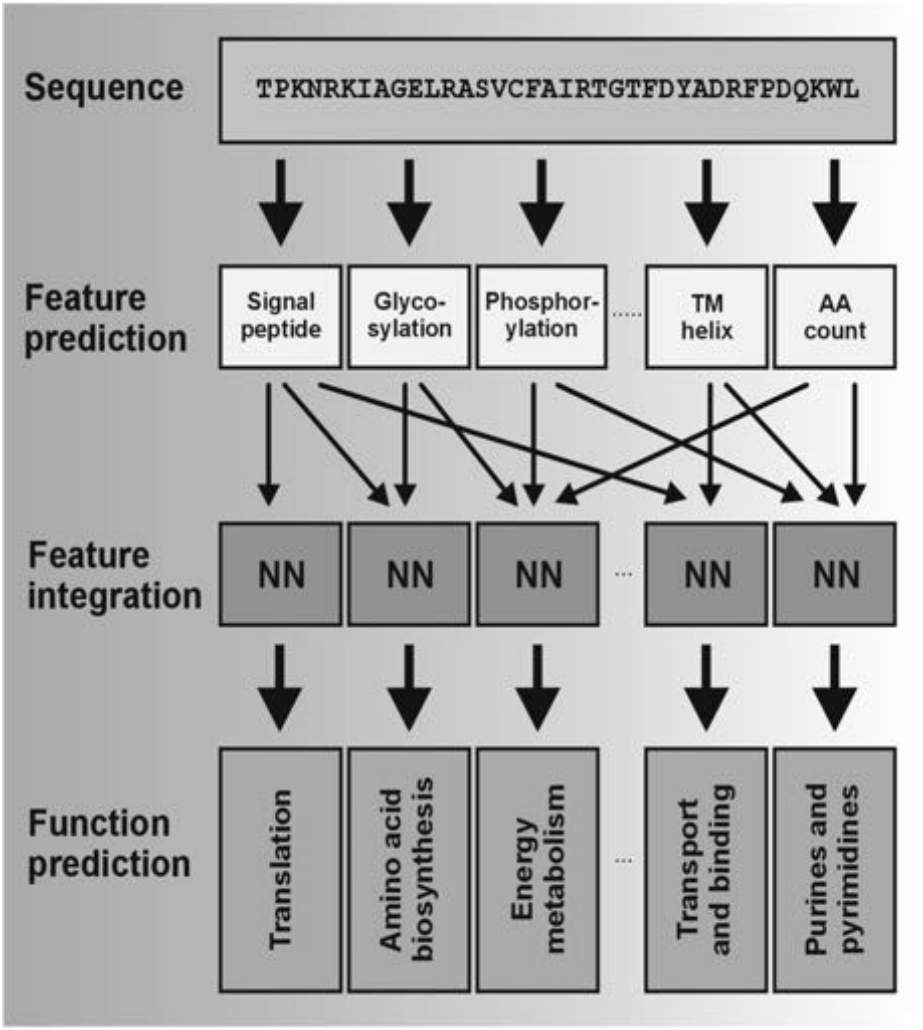

Figure 3.15: The ProtFun neural networks that predict the function of proteins in protein feature space. Each sequence is converted into features and then the networks (NN) integrate these features and provide a prediction for the affinity toward different functional categories. For different categories different protein features will have discriminatory value. During training (using experimentally characterized data) the most discriminative features are determined for each category.

```
######### ProtFun 1.1 predictions ##########

>PRIO_HUMAN

# Functional category                   Prob

  Amino_acid_biosynthesis               0.020
  Biosynthesis_of_cofactors             0.032
  Cell_envelope                         0.146
  Cellular_processes                    0.053
  Central_intermediary_metabolism       0.130
  Energy_metabolism                     0.029
  Fatty_acid_metabolism                 0.017
  Purines_and_pyrimidines               0.528
  Regulatory_functions                  0.013
  Replication_and_transcription         0.020
  Translation                           0.035
  Transport_and_binding              => 0.831

# Enzyme/nonenzyme                      Prob
  Enzyme                                0.250
  Nonenzyme                          => 0.750

# Enzyme class                          Prob
  Oxidoreductase (EC 1.-.-.-)           0.070
  Transferase    (EC 2.-.-.-)           0.031
  Hydrolase      (EC 3.-.-.-)           0.057
  Isomerase      (EC 4.-.-.-)           0.020
  Ligase         (EC 5.-.-.-)           0.010
  Lyase          (EC 6.-.-.-)           0.017
```

Figure 3.16: The prediction output from the ProtFun method for the human prion protein, PRIO_HUMAN. The method produces three types of output for functional categories: broad cellular role, enzyme classes, and Gene Ontology categories, only the two first are included here for reasons of space. The number of Gene Ontology categories predicted is growing and is currently around 75. The numerical output can be used, for example, to select an assay, or the order in which different assays should be selected, when confirming experimentally the function of an uncharacterized protein. The ProtFun method is made available at www.cbs.dtu.dk/services.

for functional inference.

The neural network was not transferring functional information just by identifying by sequence similarity from the nearest neighbor in sequence space used to train the system, as the maximal similarity between the prion sequence and the data set used to train and test the ProtFun method was only 14.8% identity at the amino acid level to a proline-arginine-rich repeat protein. Predictions like these are very useful when resolving protein function, because they can be used to generate specific hypotheses and direct laboratory experiments for sequences where no information at all can be obtained by alignment.

3.7.3 Predicting Functional Categories for Systems Biology: the Cell Cycle as an Example

Characterization of the immune system also requires that genes and proteins are grouped into subsystems, where the biochemical task of each protein may be highly different. The ProtFun method can also be used to group sequences in this manner. As an example with relevance for the immune system, we describe here a version of the method that predicts whether a protein is encoded by a periodically transcribed, cell cycle regulated gene, or not. The ability of a cell to replicate itself is one of the most fundamental features of life, and also of disease, most importantly in relation to cancers. The hundreds of genes maintaining the cell cycle work together in a highly robust manner, making it possible for cells to divide under many different growth conditions and other influences from the environment. The robustness is achieved by sophisticated regulation making the periodic gene expression highly stable. The eukaryotic cell cycle is regulated at many levels, from transcription and translation to posttranslational modification and targeted protein degradation. Proteins need not only be produced, but also be removed again when no longer needed. The cell cycle molecular machinery consists of highly diverse proteins, with little sequence similarity.

A key technique being used to elucidate which genes are involved in a given subsystem is the DNA microarray method (see section 5.1). This is also the case for the cell cycle, where gene expression measurements are made during many different time points of the cycle. Unfortunately, many of the "lists" of genes, which have been produced in this way do not agree as much as expected, even if these studies have produced highly valuable information de Lichtenberg et al. [2003, 2004]. Part of the disagreement relates to differences in experimental conditions and procedures, but a large fraction is presumably related to basic noise problems in the DNA microarray technology when measuring the expression level of weakly expressed genes.

The ProtFun function classification technique described above can be used to predict, in feature space, such systems biology related categories de Lichtenberg et al. [2003]. Not all cell cycle related genes are periodic, but many of the key factors enabling the final formation of protein complexes are. The fact that the method with a reasonable high performance is able to separate such two highly diverse categories, demonstrates that many cell cycle proteins indeed display correlations between their features, which are different from those of other proteins. These features include phosphorylation, glycosylation, stability and/or disposition for targeted degradation, as well as localization in the cell.

In relation to the immune system many other sets of proteins creating a given subsystem may also display feature based similarities that can be ex-

ploited in a prediction approach like ProtFun. One aim is of course to identify novel components involved, but also to discover whether such biochemically diverse proteins share features which can be used to describe the biology behind their functionality.