12 Sequence Alignments

Alexander K. Hartmann

Institut für Theoretische Physik
Universität Göttingen
hartmann@theorie.physik.uni-goettingen.de
Germany

Many computational methods used in Molecular Biology or Bioinformatics are in fact optimization algorithms. An example is the determination of the structure of proteins, see e.g. the contribution by U.H.E. Hansmann in chapter 13. Here, we are dealing with *sequence alignments*, which is a method to compare different biological sequences like genes or proteins. We will start with a short introduction to the basic notions in Molecular Biology, which are necessary to understand this part. Next sequence alignments are defined and different alignment algorithms are explained. In the final part, as an application, the tail of the distribution of alignment scores for random protein sequences is studied. Here additional methods from statistical physics are employed to study the distribution in the part, where sequences with probabilities as small as 10^{-40} occur.

12.1 Molecular biology

Proteins play a dominant role in life [9]. Many of them act as enzymes, which means that they catalyze various chemical reactions in cells. Muscle contraction is due to protein-protein interactions. Proteins act as channels to control transport of substances through cell membranes. As a final example, proteins play a crucial role for the immune system, see e.g. Fig. 12.1.

Chemically, proteins are polypeptide chains. They consist of a number of amino acids linked to each other. Very common are single-chain molecules, but also multiple-chain proteins exist. Typically lengths are several hundred amino acids, the longest contain about 5000 amino acids. 20 different amino acid molecules are the building blocks of proteins. Each amino acid consists of a carbon atom C to which a hydrogen atom H, an amino group (NH_2), a carboxyl group (COOH) and an amino-acid dependent group R are bound. Only one amino-acid, proline, has a slightly different structure, see below. In aqueous neutral solution, the amino acids exist in zwitterionic form, displayed in the Figs. 12.2-12.5. Please note that not all amino acids are, despite their name, acids. The smallest and simplest amino acid is Glycine, see Fig. 12.2, denoted by the letter G. Here, the R group is just a single hydrogen atom. It is neither especially hydrophilic or hydrophobic.

There are 8 different *hydrophobic*, i.e. non-polar, amino acids, see Fig. 12.3. Amino acids where the group R can be ionized in solution are hydrophilic. The 9 hydrophilic amino acids are displayed in Fig. 12.4. Two special amino acids, which fall out of the previous classifica-

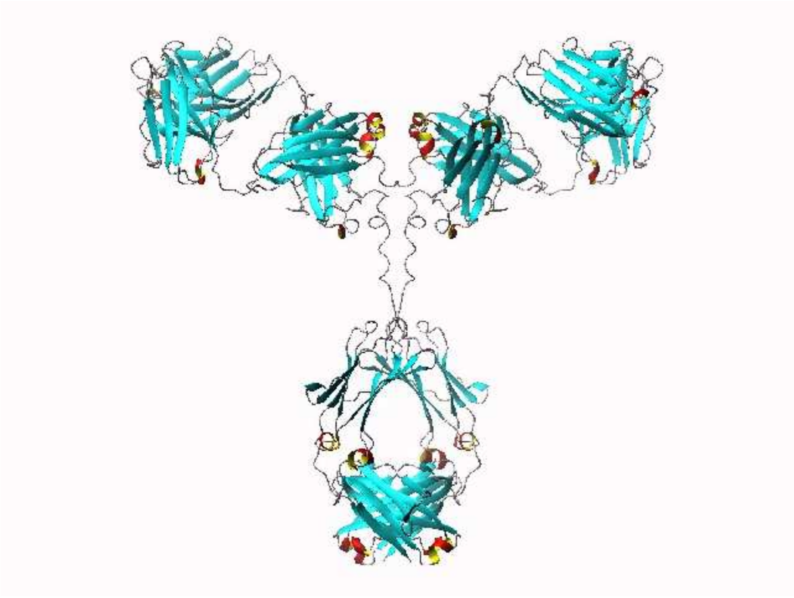
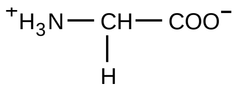


Figure 12.1: The antibody *immunoglobulin G* (IgG), which attaches to antigens, i.e. macromolecules foreign to the body, and stimulates then the immune system to generate a protective answer.



Glycine

Figure 12.2: Glycine (G) is the smallest amino acid.

tion, are Cysteine (C) and Proline (P), see Fig. 12.5.

In a protein, the different amino acids are connected by *peptide bonds*, which connect, under formal removal of H_2O, the carbon atom of the carboxyl group with the nitrogen atom the amino group. Since this process exhibits an energy barrier, in a cell usually enzymes, i.e. other proteins, perform the assembly. For the final protein, at one end an amino group (*N-terminal end*) and at the other end a carboxyl group (*C-terminal end*) remain unbounded. Usually one specifies proteins by stating the amino acid sequence starting at the N-terminal end. This sequence of amino acids is called the *primary* structure. It determines the structure of the protein in real-space[1] to a large extend. Important factors in the creation of the real-space structure are the formation of hydrogen bonds and the environment (water and enzymes) of the protein.

[1]The real-space structure is described in a hierarchical manner, i.e. by its so called *secondary, tertiary and quaternary structure*. Since we are here interested in the primary structure only, we don not go into details.

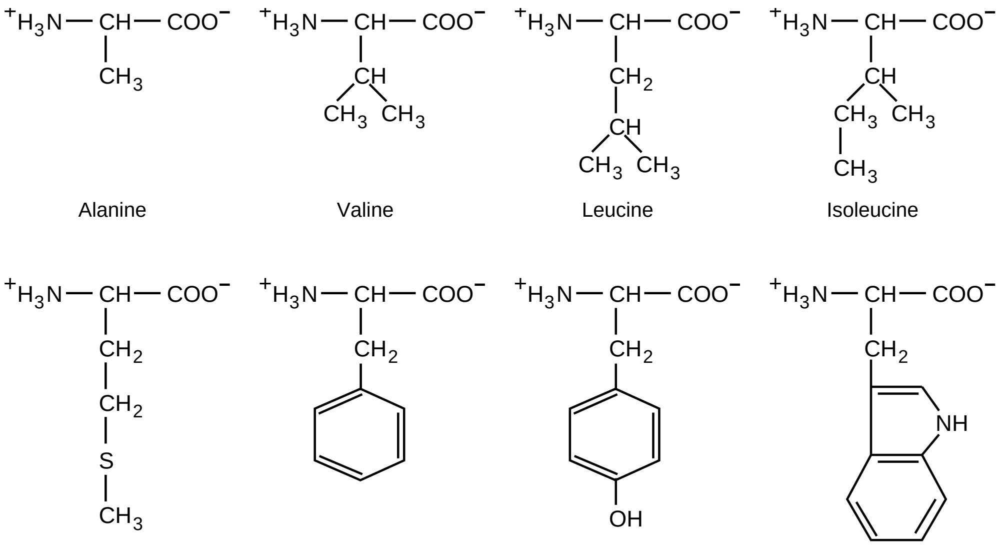


Figure 12.3: Hydrophobic amino acids: Alanine (A), Valine (V), Leucine (L), Isoleucine (I), Methionine (M), Phenylalanine (F), Tyrosine (Y) and Tryptophan (W).

The structure can be measured using X-ray diffraction or nuclear magnetic resonance (NMR). The real-space structure is very characteristic for each protein and of crucial importance for its function. For example, the mad-cow disease is believed to be due to "missfolded" proteins, called *prions*. Hence, the prediction of the structures of proteins and the study of the folding process is a very important task in Molecular Biology. A usual approach to this problem is to look for minimum energy conformations, see e.g. chapter 13.

In order to study proteins, one must know their sequences. Currently, two methods are used to investigate short sequences (up to – let us say – 30 amino acids), *sequenators* and *mass spectrometry*. The sequenator is an automatic device which is based on a chemical method: Using chemical reagents it detaches one amino acid after the other, starting from the N-terminal end, and identifies the amino acid in a separating column. As separation column, very often *gel electrophoresis* is used. It works in principle in the following way: charged molecules, which are marked, either radioactively or with a dye, are put in a gel and are subjected to an electric field. The molecules are driven by the field, the faster the higher their charge is, but they travel the slower the larger they are. After calibration with known amino acids, the identity of an unknown amino acid can be inferred just from the final distance it has traveled in the field (if the amount measured is large enough).

When using mass spectroscopy, one needs a collection of many fragments from the protein. Since all fragments originate from the same protein, only certain combinations of amino acids can appear as fragments, i.e. all possible subsequences of the full sequence. Each subsequence has a certain mass, which is the sum the amino acid masses it contains. Hence, for a given

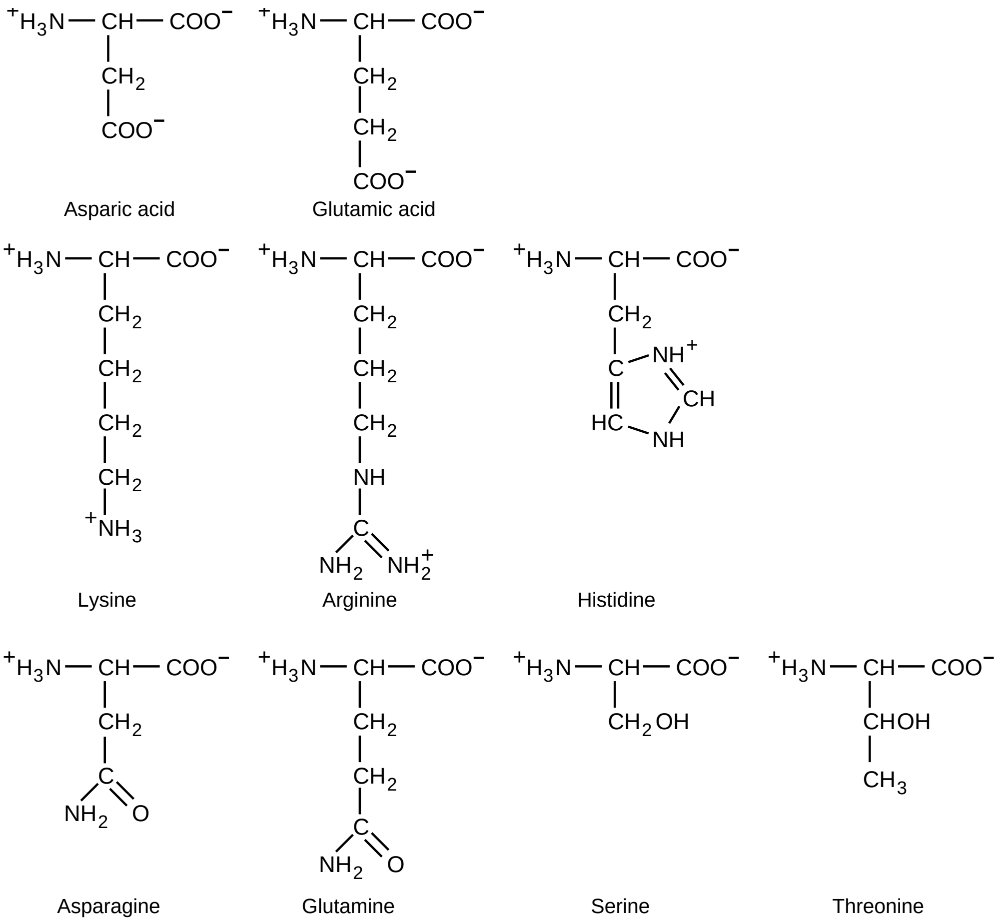


Figure 12.4: Hydrophilic amino acids: Aspartic acid (D), Glutamic acid (E), Lysine (K), Arginine (R), Histidine (H), Asparagine (N), Glutamine (Q), Serine (S) and Threonine (T).

protein, only a finite number of values for the masses of the fragments can appear. This means, the sequence of the given protein can be deduced from the set of feasible values of fragment masses, i.e. from the positions of the peaks in the spectrum.

For longer proteins, one has to break them into smaller fragments using enzymes at specific positions. Then one analyzes the different fragments, and looks for overlaps of the different fragments to obtain the full sequence. The method to find the overlapping regions is to utilize the sequence alignment algorithms, which are explained in the next section. As we will see later, sequence alignment can be used for general protein comparison as well,

The primary structures of all proteins of a cell are stored in the *genome*, which is a collection of DNA (deoxyribonucleic acid) chains. They are used to generate all proteins in a cell, one says,

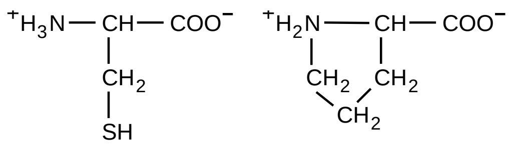


Figure 12.5: Cysteine (C) and Proline (P) are proteins for special purposes. Cystine is important for providing sulphur in its -SH group. Thus, in proteins containing Cystine, S-S bonds can be formed. They are very strong and lead to stable structures. Hence Cystine plays an important role for the protein folding process (see chapter 13). Proline differs from the general structure of the 19 other amino acids. It has an imino (-NH) group instead of an amino (-NH_2) group, i.e. actually it should be called an imino acid.

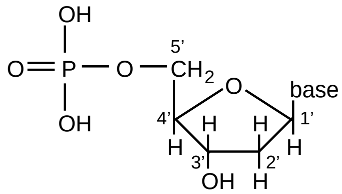


Figure 12.6: General structure for nucleotide.

the proteins currently produced are *expressed* by the cell. DNA is a double-stranded helix, where each strand consists of a chain of nucleotides. Each nucleotide has the principle form shown in Fig. 12.6. It consists of a sugar (2-deoxy-D-ribose) with an additional phosphate group and a base bound at position $1'$ of the sugar ring. The type of the nucleotide is specified by the base. For DNA, four different base types exist: Adenine (A), Guanine (G), Cytosine (C) and Thymine (T), see Fig. 12.7. Adenine and Guanine are bound to the sugar at the 9 position, while Cytosine and Thymine at the 1 position.

The different nucleotides are connected to each other via formally removing one -OH group from the phosphate group (called $5'$, see Fig. 12.6), removing a hydrogen atom at the $3'$ position and connecting the "open" bonds. Please note that this formation does not take place directly, because it involves a high free energy barrier to overcome. In a cell, again, enzymes catalyze this process. The ends of the DNA chains are uniquely defined, because at one end, called $5'$, there is an phosphate group, while at the other, called $3'$ there is none. In a cell, DNA is processed always starting at a $5'$ end.

Since there are 20 different amino acids, 3 bases ($4^3 = 64$ combinations), are needed to code an amino acid in principle. A group of three bases is called *codon*. Most amino acids are coded by several codons. The codon ATG indicates the start of a coding sequence, (but it codes Methionine as well). The sequence of codons coding a protein is called a *gene*. The three codons TAA, TAG, TGA, (called *stop codons*) are used to indicate the end of a gene. This code is almost universal for all organisms, there are only few minor variations. But there

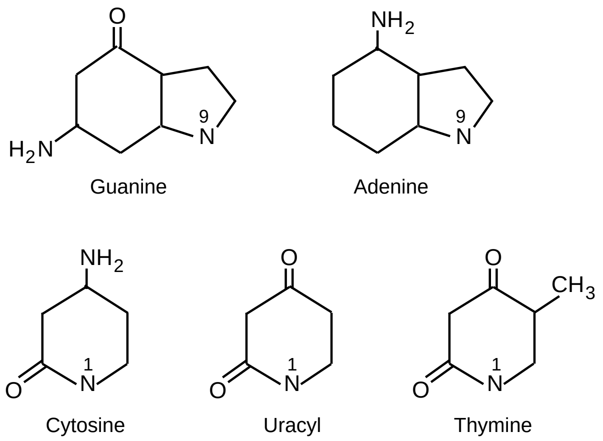


Figure 12.7: The five bases Guanine (G), Adenine (A), Cytosine (C), Uracil (U) and Thymine (T).

is not always a one-to-one correspondence between DNA and proteins, sometimes details of a protein are determined at assembly. Also, large parts of the genome do not code protein sequences, not everything of this "non-coding" regions currently well understood.

The double-helix structure is hold together by hydrogen bonds. Hydrogen bonds can be formed only between A-T pairs (2 bonds) and between G-C pairs (3 bonds). This means that already one strand of the DNA contains the full genomic information, the second one is somehow a "backup" to make the stored information insensitive against external perturbations. Furthermore, it makes duplication of the genes easier. Due to the weak nature of hydrogen bonds, the double-helix structure can be *denaturated* in vitro by heating, i.e. the double helix separates into two single strands. In living cells, the DNA has to be unfolded to be read of. This is not done by heating but special enzymes are used by the cell. For the common case, where only a part of the DNA has to be read off to synthesize a protein, an enzyme called *RNA polymerase* splits the double strand locally and creates a copy from one strand of the DNA. This copy is called a mRNA (*messenger ribonucleic acid*). The further parts of protein synthesis use the mRNA, hence the valuable original data stored in the DNA has to be touched only rarely.

RNA is similar to DNA, i.e. a chain of nucleotides. There are two main structural differences. First, the sugar part of a nucleotide is a ribose instead of a deoxyribose, i.e. it has an -OH group at position $2'$ instead of a hydrogen (see Fig. 12.6). Second, the four bases for RNA are Guanine, Adenine, Cytosine, like for DNA, and Uracil (U) (instead of Thymine for DNA), see Fig. 12.7. RNA is single-stranded, but also folds into a three-dimensional structure which is crucial for its function.

In principle, DNA can be analyzed with similar methods like proteins. But one has the additional problem that usually a very small amount of the DNA is available. Thus, one has to amplify the probe. This is done using *polymerase chain reaction* (PCR). This method itera-

tively heats the sample up to 95°, causing the denaturation of the double helix. Then each occurrence of the desired gene section is duplicated using the enzyme RNA polymerase. The enzyme grows starting from *primers*, which attach to subsequences unique for each primer. Then the probe has to be cooled again, to allow further primers to attach to the new created sequences. This cycle can be repeated, each time doubling the amount of DNA.
The actual measurement of the DNA sequence can be performed by applying the *fluorescent method*. It works by adding modified bases called *terminators*, which do not allow the sequence to continue to grow during the duplication process beyond the terminator. These terminators are marked with four different dyes, one color for each base type. I.e. one obtains a collection of subsequences, all starting at the primer position and ending at all possible positions of the initial DNA sequence, and the color at the end of the subsequence telling the base the corresponding subsequence ends with. Then the sample is studied using gel electrophoresis (see above). The distance a segment has traveled within a certain time in the electric field decreases with the length of the segment. Therefore, from the different distances, the position where each given base was located can be immediately read off from the color visible at this position. This color detection can be performed automatically, allowing for fully automated DNA analysis. This technique works for relative short DNA, about 500 bases long. In e.g. human cells, the DNA is about 6 billion base pairs long. In this case, similar to the analysis of long proteins, one has to divide the genome into many fragments, analyze each fragment, and look for matching regions of the fragments (*shotgun method*).
For this method, but also for protein and gene comparison in general, sequence alignments are used, which are explained in the next section.

12.2 Alignments and alignment algorithms

Modern Molecular Biology, e.g. the *Human Genome Project* [16], relies heavily on the use of large databases [5, 25], where DNA or protein sequences are stored. The basic tool for accessing these databases and comparing different sequences is *sequence alignment*. The result of each comparison is a *maximum* alignment *score* S, i.e. an optimization algorithm is used to obtain the maximum score. One is interested either in *global* or *local* optimum alignments. For the first case, the score is maximized over all alignments of both complete sequences. The optimum local alignment is the optimum over all global alignments of all possible pairs of contiguous subsequences, for a precise definition see below. Next, a formal definition of alignment is given, then algorithms for several types of alignments are presented.
Let $\mathbf{x} = x_1 x_2 \dots x_n$ and $\mathbf{y} = y_1 y_2 \dots y_m$ be two sequences over a finite alphabet Σ with $r = |\Sigma|$ letters. For DNA the alphabet has 4 letters, representing the bases, for protein sequences it has 20 letters, representing the amino acids. The basic idea of using alignments to compare $\mathbf{x}, \mathbf{y}$ is to identify equal or similar subsequences in $\mathbf{x}$ and $\mathbf{y}$. An alignment is a pairing $\{(x_{i_k}, y_{j_k})\}$ ($k = 1, 2, \dots, K$, $1 \leq i_k < i_{k+1} \leq n$ and $1 \leq j_k < j_{k+1} \leq m$) of letters from the two sequences. Some letters may not be aligned in principle, i.e. *gaps* occur. In Fig. 12.8 examples for global alignment without gaps, global alignment with gaps and local alignment are given. One can extend the alignment of two sequences to *multiple* alignments of more sequences in a straightforward way [8], but here we will concentrate on pairwise alignment.
To each alignment a score is assigned, via a scoring function $S(\mathbf{x}, \mathbf{y})$. The score is chosen

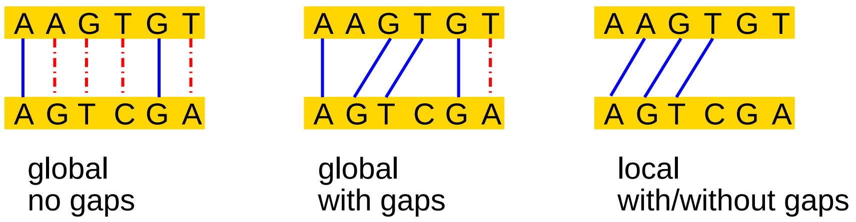


Figure 12.8: Different types of sequence alignments. Two DNA sequences are shown. Aligned symbols are joined by lines. Matching alignments are indicated by full lines, non-matches by broken lines. For global alignment without gaps (left), all letters are aligned. When allowing for gaps (middle), additional costs (i.e. negative scores) are taken into account. For local alignments (with and without gaps), the unpaired regions and the beginning and at the end of the sequences are not penalized, i.e. a zero score is contributed from there.

in such a way, that a high similarity leads to a high score. This can be done by choosing the total score as the sum of scores of all pairs of aligned letters plus the costs of all gaps $S(\mathbf{x}, \mathbf{y}) = \sum_k s(x_{i_k}, y_{j_k}) + \sum_{\text{gaps } g} f(l_g)$. The gap costs are usually taken as function of the length l_g of the gaps. Gap costs are discussed later. The part of the score for the aligned letters is a linear function of the alignment and neighboring pairings are independent, which is a simple approximation. The scores for equal or similar letters, "similar" meaning that the biological function of the corresponding units is similar, are positive, while the scores for very different letters are negative. For a moment one can assume, that the score matrices $s(a, b)$ (also called *substitution matrices*) are just given, below the biological origin of the scores is outlined. Also the part of the score for the gaps is taken to be a sum over all gaps. The scores for the gaps are chosen to be negative, because the need to leave out some letters being aligned indicates that the compared sequences are dissimilar in some way.

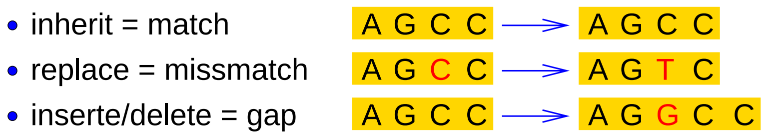


Figure 12.9: The evolution changes the DNA/amino-acid sequences. Inheritance and mutations correspond to matching resp. missmatchings and gaps of the alignment.

The score matrices cannot be "derived" in some defined way, many choices are possible, also depending the way the data is to be analyzed. The fundamental notion behind the choice of the scores is that the sequences are generated by evolution. This is directly obvious for DNA. Proteins are produced in all cells, the blueprint is encoded in their DNA. During evolution, the DNA changes, hence the expressed proteins change. Life forms, which are closer in evolution, will have proteins which are closer as well. Thus, one can also speak of the evolution of the proteins itself. This evolution of a protein or the DNA can be drawn in form of a (phylogenetic) tree. At each node of the tree, different forms originating from the same predecessor emerge. Parts of a sequence which are inherited to subsequent forms correspond to matches in the alignments, mutations where bases or amino acids are replaced correspond to mismatches, while insertions or deletions of code correspond to gaps, see also Fig. 12.9.

Here we consider score matrices for proteins. From the analysis of evolution a lot of data is already available. For the most simple approach to construct a score matrix, one can take a probabilistic approach, here no gaps and only global alignments are considered for simplicity. A sample of already known sequences and the corresponding phylogenetic tree is assumed to be available. One can now just compare all sequences in the sample and obtain the frequencies/probabilities $p_{a,b}$ that the two amino acids a, b are derived from the same ancestor c (i.e. c might be a or b or different). If one assumes independence of the different mutations, the probability that the alignment of of two sequences $\mathbf{x}, \mathbf{y}$ occur is $P(\mathbf{x}, \mathbf{y}|\mathrm{A}) \equiv \prod_i p_{x_i y_i}$. Let furthermore f_a the probability that a letter a appears at all in the sample. Then the probability that the alignment appears by pure chance is $P(\mathbf{x}, \mathbf{y}|\mathrm{R}) = \prod_i f_{x_i} f_{y_i}$. The so called *odds ratio* is the ratio of these probabilities:

$$\frac{P(\mathbf{x}, \mathbf{y}|\mathrm{A})}{P(\mathbf{x}, \mathbf{y}|\mathrm{R})} = \prod_i \frac{p_{x_i y_i}}{f_{x_i} f_{y_i}} \tag{12.1}$$

Now the score is just defined as the logarithm of the odds ratio. Due to the additivity of the logarithm for products, the score of the full alignment of the two sequences becomes a sum of scores $S = \sum_i s(x_i, y_i)$ of the paired letters with

$$s(a, b) = \log \left(\frac{p_{ab}}{f_a f_b} \right) . \tag{12.2}$$

For this approach, one must define which time window is analyzed, within the sequences have evolved, i.e. how far in evolution the sequences are away, which are compared. The larger the time interval from which the data originates, the more different the sequences will be, hence the probabilities P_{ab} change.

One frequently used score matrix which uses this approach are the PAM (*point accepted mutation*) matrices by Dayhoff, Schwartz and Orcutt [7]. Their first step was to study only the direct ancestors in the phylogenetic tree they used, leading to substitution matrices for one unit of time, called PAM 1. This matrix was extended to longer times, by raising it to the t'th power resulting in the probabilities $P(b|a, t)$ that a is substituted by b after t steps. Since $P(b|a) = p_{ab}/f_a$, the final score of the PAM t matrix according to Eq. (12.2) is then $s(a, b|t) = \log P(b|a, t)/f_b$. These values are scaled by some factor and rounded to the nearest integer. The most common form is the PAM 250 matrix with scaling factor $3/\log 2$.

Since the PAM matrices are resulting basically from studying short-time mutations, long time influences are not taken into account. E.g. changes of codons leading to mutations of proteins, where the corresponding coded differ in more than one base, are not considered. For this reason, the BLOSUM matrix set was developed [14]. Within this approach, related proteins from the BLOCK database [13] where clustered (again without allowing for gaps), where the proteins in each cluster agreed in at least a fraction L of the letters. Then the sequences from *different* clusters where compared and the frequencies of substitution measured. Then the single-letter alignment scores were obtained, after a suitable normalization, using the standard way Eq. (12.2). Finally the data is usually rescaled and rounded to the nearest integer resulting in the BLOSUM L matrix. BLOSUM62 is the standard method used for alignments.

Whatever scoring matrix is used, in case of local alignment, the expected score of two fully random sequences must be negative, i.e. $\sum_{a,b} s(a,b) f_a f_b < 0$. Otherwise, extending an alignment would on average always increase the score, hence no local alignment would occur when looking for large scores. Whenever the scores are derived from a likelihood ratio like in Eq. (12.2), this is automatically fulfilled.

The way gap costs are set up is even more heuristic than for the alignment scores. The basic idea is as follows. When due to a mutation a subsequence of a protein is removed to form a new protein, the corresponding optimum alignment of both proteins will have a gap at this position. Since a longer mutation is less likely than a shorter one, it is reasonable to increase the gap costs with the gap length. But having two independent mutations of length l is less likely than having one mutation of length $2l$. Hence the corresponding gap costs should reflect this behavior as well. This is e.g. fulfilled for the so called *affine* gap costs (α, β) where. a gap of length l has costs $g(l) = -\alpha - \beta(l-1)$. The most common cost function is (12,1) gap costs, but many other approaches exist as well.

For a *global* alignment without gaps ($n = m$), there is just one alignment, hence just one score, no algorithm is needed. When allowing for gaps, several alignments are possible, hence one can have for similar, even equal sequences $\mathbf{x}, \mathbf{y}$ a very low score when aligning the "wrong" letters. Therefore, one defines the similarity as the *optimum* global alignment $S_g(\mathbf{x}, \mathbf{y})$ with gaps, which is obtained by maximizing the score over all numbers K of aligned pairs and over all possible placements of the gaps. The optimum *local* alignment S is the maximum over all possible contiguous subsequences $\tilde{\mathbf{x}} = x_i x_{i+1} \dots x_{i+l-1}$, $\tilde{\mathbf{y}} = y_j y_{j+1} \dots y_{j+k-1}$ of the optima $S_g(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$. Here again the cases with and without allowing gaps are possible. Hence, since an alignment of zero length has score zero, the optimum local alignment is always non-negative by definition.

For both global and local alignment with gaps efficient algorithms [24, 30, 10, 8] exist, which calculate an optimum alignment in time $O(nm)$. They will be presented next. In principle all these algorithms use a dynamic programming approach. This is possible because the score function is linear, hence the optimum alignment can be decomposed into a series of optimum alignments of shorter sequences. We will start with an algorithm for global alignments with gaps. For the moment only *linear* gap costs, i.e. (0,d) affine gap costs will be considered. This is the *Needleman-Wunsch* algorithm [24].

The basic idea is that an optimum alignment of the partial sequences $\tilde{\mathbf{x}} = x_1 x_2 \dots x_i$, $\tilde{\mathbf{y}} = y_1 y_2 \dots y_j$ can be obtained by either extending the optimum alignment of $\tilde{\mathbf{x}} = x_1 x_2 \dots x_{i-1}$, $\tilde{\mathbf{y}} = y_1 y_2 \dots y_{j-1}$ by aligning x_i and y_j, or by extending either the optimum alignment of $\tilde{\mathbf{x}} = x_1 x_2 \dots x_{i-1}$, $\tilde{\mathbf{y}} = y_1 y_2 \dots y_j$ resp. $\tilde{\mathbf{x}} = x_1 x_2 \dots x_i$, $\tilde{\mathbf{y}} = y_1 y_2 \dots y_{j-1}$ by a gap. The real optimum alignment is the maximum over all three possible scores. The optimum score of the sequences up to letters i and j is denoted by $F(i,j)$, hence a recursive function is obtained

$$F(i,j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) & \text{align} \\ F(i, j-1) - d & \text{gap in x} \\ F(i-1, j) - d & \text{gap in y} \end{cases} \tag{12.3}$$

Since the alignment of one sequence with nothing is a gap, the recursion is started by $F(0,l) = F(l,0) = -d \times l$. The optimum alignment of the full sequences is then the rightmost entry in the bottom line $F(n,m)$.

		A	A	G	T	G	T
	0	−2	−4	−6	−8	−10	−12
A	−2	1	−1	−3	−5	−7	−9
G	−4	−1	−2	0	−2	−4	−6
T	−6	−3	−4	−2	1	−1	−3
C	−8	−5	−6	−4	−1	−2	−4
G	−10	−7	−8	−5	−3	0	−2
A	−12	−9	−6	−7	−5	−2	−3

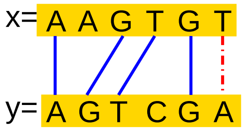


Figure 12.10: Example of how the Needleman-Wunsch algorithm works. The dynamic-programming matrix $F(i,j)$ and the resulting optimum alignment are shown.

As a simple example, we consider the (DNA) sequences $\mathbf{x}$ =AAGTGT and $\mathbf{y}$ =AGTCGA. For a match of two equal bases, the score $s(a,a) = 1$ is used, while for a mismatch a $s(a,b) = -3$ penalty is given. A gap has cost $-d = -2$. In Fig. 12.10 the resulting dynamic-programming matrix $F(i,j)$ is shown together with the resulting maximum alignment. As an example, we consider how the score $F(1,1)$ is calculated. Either x_1 aligned to a gap (score -2) or y_1 aligned to a gap (score -2) is extended by a y_1 resp. x_1 to a gap (score -2), leading to the total score of -4. Or the two letters x_1 =A and y_1 =A (score s(A,A) $= 1$) are aligned. Since $1 > -4$ the later possibility is taken. The remaining matrix is calculated in the same way.

Since the dynamic-programming matrix $F(i,j)$ has $n \times m$ entries, and the calculation of each entry takes only constant time, the full algorithm runs in $O(nm)$ steps. Note that the actual alignment can be inferred from the calculation of the matrix as well, by always storing a pointer backwards to the previous optimum, from which the current optimum was obtained. Then the optimum alignment can be read of the matrix by starting at the element (m,n) and going backwards following the pointers. In the example in Fig. 12.10 the "path" when going backwards in the matrix is indicated by shaded squares. The actual alignment may not be unique in case several possibilities in (12.3) lead to the same score.

Indeed the Needleman-Wunsch algorithm is a shortest path algorithm for a special geometry. The nodes of the underlying graph are arranged in a rectangular array. Edges exist only between nearest neighbors and for two of the four possible next-nearest neighbors (pointing right downwards). Since the edges are directed and point into the direction of increasing sequence lengths, the corresponding graph contains no loops. Hence a very simple shortest path algorithm can be applied. Note that other pattern-matching algorithms, like the UNIX command *grep*, work in a similar fashion.

The extension to find the optimum local alignment with linear gap costs is straightforward and was given by Smith and Waterman [30]. Since one can always align nothing, which gives score 0, it does not make sense to extend a subsequence with negative score, instead a new subsequence is started. This leads to the following recursion:

$$F(i,j) = \max \begin{cases} 0 & \text{restart} \\ F(i-1,j-1) + s(x_i,y_j) & \text{align} \\ F(i,j-1) - d & \text{gap in x} \\ F(i-1,j) - d & \text{gap in y} \end{cases} \tag{12.4}$$

Since gaps have always negative cost, the recursion is started by $F(0,l) = F(l,0) = 0$. The final optimum alignment is the maximum over all matrix elements $S = \max_{i,j} F(i,j)$. When storing again backpointers, the optimum alignment can be obtained by starting from the element (i_0, j_0) where the maximum is found. Again, the optimum alignment might not be unique.

		A	A	G	T	G	T
	0	0	0	0	0	0	0
A	0	1	1	0	0	0	0
G	0	0	0	2	0	1	0
T	0	0	0	0	3	1	2
C	0	0	0	0	1	0	0
G	0	0	0	1	0	2	0
A	0	1	1	0	0	0	0

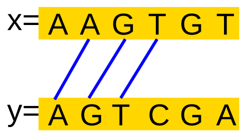


Figure 12.11: Example of how the Smith-Waterman algorithm works. The dynamic-programming matrix $F(i,j)$ of partial scores and the resulting optimum local alignment are shown.

In Fig. 12.11 the resulting dynamic-programming matrix $F(i,j)$ along with the optimum local alignment for the same two sequences as above are shown.
The most common form used to compare protein sequences is local alignment with (α, β) affine gap cost. The algorithm [10] is also a dynamic programming algorithm. But now one recursion relation is not sufficient here. The reason is that the costs for opening a gap are different from the costs of extending a gap. Therefore, one uses three dynamic-programming matrices, i.e.

- $A(i,j)$, the optimal score up to (i,j), given that x_i and y_j are aligned.
- $G_x(i,j)$, the optimal score up to (i,j), given that x_i is aligned to a gap.
- $G_y(i,j)$, the optimal score up to (i,j), given that y_i is aligned to a gap.

The recursive equations, for the case of local alignment, read as follows

$$A(i,j) = \max\begin{cases} 0 & \text{restart} \\ A(i-1,j-1)+s(x_i,y_j) & \text{align} \\ G_x(i-1,j-1)+s(x_i,y_j) & \text{align} \\ G_y(i-1,j-1)+s(x_i,y_j) & \text{align} \end{cases} \tag{12.5}$$

$$G_x(i,j) = \max\begin{cases} 0 & \text{restart} \\ A(i-1,j)-\alpha & \text{new gap} \\ G_y(i-1,j)-\alpha & \text{new gap} \\ G_x(i-1,j)-\beta & \text{extend gap} \end{cases} \tag{12.6}$$

$$G_y(i,j) = \max\begin{cases} 0 & \text{restart} \\ A(i,j-1)-\alpha & \text{new gap} \\ G_x(i,j-1)-\alpha & \text{new gap} \\ G_y(i,j-1)-\beta & \text{extend gap} \end{cases} \tag{12.7}$$

The actual total optimal score up to (i,j) is $F(i,j) = \max\{A(i,j), G_x(i,j), G_y(i,j)\}$ and the local optimum is obtained again as the optimum of $F(i,j)$ over all elements of the dynamic-programming matrix. Retrieving the actual alignment is performed again by pointers, but now one has to store for each matrix one pointer, this one pointing to the element in the actual matrix which was previous in the optimal subsequence. E.g. if opening a new gap after a pair of aligned letters led to the optimal score for $G_y(i,j)$, then there will pointer from $G_y(i,j)$ back to $A(i,j-1)$.

There are several algorithms for other types of alignment [8]. When looking for *repeated matches*, one allows that a subsequence of $\mathbf{x}$ is aligned several times to different subsequences of $\mathbf{y}$. For *overlap alignments*, one allows that one sequence contains the other, or that they overlap. This is a special form of local alignment, with the restriction that at least one prefix and one postfix of the two sequences have to be aligned. One can obtain even more complex algorithms, by describing them by finite-state automates. E.g. for local alignment with affine gap cost, one would have three states *aligned*, *x-gaped* and *y-gaped*. With more states, more evolved algorithms can be realized, e.g. if one wants to score different regions of a protein in different ways. All these algorithms have in common that they run in $O(nm)$. In the case of multiple alignment, i.e. when one is interested in patterns that appear in more than two sequences, no polynomial-time algorithms are known. Due to this slow running time, multiple alignment is not used for large databases.

For the largest public databases, having millions of sequences stored, even the $O(nm)$ algorithms are too slow, since an answer within few seconds is desired. For this reason often fast heuristics are applied, which do not guarantee to give the exact optimum alignment. For example the BLAST package [1] works by first searching for short subsequences with match exactly, and then by trying to extend these subregions to form larger high scoring subregions. BLAST is e.g. used in the protein database Swiss-Prot [33].

In the next chapter the statistics of alignment scores for random sequences are studied. Using a special algorithm originating in statistical physics, one can reduce a lot the number of sequences one has to compare in order to get good results. Thus, no heuristic algorithm is necessary, an exact $O(nm)$ algorithm can be applied instead.

12.3 Low-probability tail of alignment scores

The result of a sequence alignment is a maximum score, e.g. $S = 200$. Now one wants to estimate the meaning of this score, does it mean a high similarity or a low? Thus, one wants to know the *significance* of the result. One way to estimate the significance, is to compare against fully randomness, i.e. to calculate the probability $p(S)$ that such a (large) score can be obtained by pure chance.
For biologically relevant models, e.g. for protein sequences with BLOSUM62 substitution scores [14] and affine gap costs [10], $p(S)$ is not known in the interesting region, where $p(S)$ is small. A number of empirical studies [31, 6, 23, 35, 36, 2] for local alignment, in the region where $p(S)$ is large, suggest that $p(S)$ is an *extreme value* (or Gumbel) distribution [12]

$$p_G(S) = \lambda e^{-\lambda(S-u)} \exp(-e^{-\lambda(S-u)}) \,, \tag{12.8}$$

where u denotes the maximum of the distribution and λ characterizes the behavior for large values of S, i.e. the tail of the distribution.
As an attempt to check whether (12.8) holds, one can easily generate e.g. $N \approx 10^5$ samples of pair of sequences according the frequencies f_a, obtain each time the optimum alignment, and calculate a histogram of the optimum scores S, as done in the above mentioned previous studies [2, 6, 23, 31, 35, 36]. This *simple sampling* allows one to calculate $p(S)$ in the region were the probabilities are large (e.g. $p(S) \geq 10^{-4}$). Hence it is actually the region of the distribution, where the relevant scores are *not* found, since the proteins generated by evolution are indeed *not* random. Recently, the *island method* [3] was introduced, which allows a speed up of several orders of magnitudes for very long sequences like $n = 10^4$, but still the far end of the distribution is out of reach. Also, please note that biologically relevant protein sequences have lengths of only few hundred amino acids.
Here, to determine the tail of $p(S)$, a *rare event simulation* is applied. For dynamical problems, like investigating queuing systems or studying the reliability of technical components, several techniques [27] have been developed. Related methods have been introduced in physics [4, 11].
By simply changing perspective, one can apply these standard techniques to many other problems. Here, the method it is applied to sequence alignment. The basic idea is that one uses a physical system, which has a state given by a pair of sequences and is held at temperature T, instead of directly drawing the random sequences. This idea is similar to the simulated annealing approach [19], used to find approximate solutions of hard optimization problems. But the method presented here goes much beyond simulated annealing, because not only the minimum of one system but the whole distribution over all random instances is sampled during one run. The state of the system changes in time, governed by the rules of statistical mechanics. The energy E of the system is defined as $E = -S$. Therefore, at low temperatures the system prefers pairs of sequences with high score value S. Since the thermodynamic properties of such a system are known in principle, it is possible to extract from the measured distribution $p^*_T(S)$ of scores the target distribution $p(S)$.
To determine the behavior of $p(S)$ at the rare event tail (e.g. $p(S) \approx 10^{-40}$), one views each pair $c = (\mathbf{x}, \mathbf{y})$ of sequences as the state of physical system, which behaves according the rules of statistical mechanics, with $-S$ being the energy of the system. More precisely, instead of

considering many independent pairs of fixed sequences, a Markov chain [20] $c(0) \to c(1) \to c(2) \to \ldots$ of pairs is used to generate the instances. For each instance $c(i)$ the optimum local alignment score S is calculated. Let $p(c \to c')$ denote the transition probability from state c to state c'. Changing the sequences dynamically is similar to annealed disorder simulations [32, 29, 34]. There, the quenched disorder (in this case the sequences) *and* the dynamic variables (in this case the alignment) are degrees of freedom, i.e. allowed to change during the annealed simulation. While the physics of an annealed system is different from the physics of the related quenched system, here an annealed-disorder-like simulation is used via applying a simple transformation (see below) to obtain the *true* behavior of the quenched system.

The simplest rule for the transition is, to choose randomly a position in one of the sequences with all positions being equiprobable and to choose randomly a new letter from the alphabet, the letters having probabilities f_a, i.e. $p(c \to c') = f_a/(n+m)$ if c, c' differ by at most one letter, and $p(c \to c') = 0$ otherwise. So far no temperature T and no statistical mechanics is involved. With this choice of the transition probabilities, iterating this step t times, for $t \to \infty$ all possible pairs of sequence have the probability $P(c) = \prod_i f_{x_i} \prod_j f_{y_j}$ of occurrence. Hence, simple sampling is reproduced.

To increase the efficiency, one can change the sampling distribution [22, 27], a standard method for simulating rare events [18], which allows one to concentrate the sampling in small regions in configuration space. A good choice for the transition rule of the sequence-alignment problem is first to change one position of one sequence randomly as above and recalculate the optimum alignment $S(c')$ with a standard algorithm as described in the previous section. This move $c \to c'$ is accepted with the Metropolis probability [22]

$$P_{\mathrm{Metr}} = \max(1, \exp(\Delta S/T)), \quad \text{where } \Delta S = S(c') - S(c)\,. \tag{12.9}$$

This leads to the equilibrium state of a physical system at temperature T with energy $E = -S$, with the distribution weighted by the sequence probabilities $P(c)$. The advantage of this approach is that the equilibrium distribution $Q(c)$ is known from statistical physics [26]: $Q(c) = P(c)\exp(S(c)/T)/Z$ with $Z(T) = \sum_c P(c)\exp(S(c)/T)$ being the partition function. Thus, the estimator for the probability to have score S in the ensemble at temperature T is

$$p^*_T(S) = \sum_c{}' Q(c) = \frac{\exp(S/T)}{Z(T)} \sum_c{}' P(c)\,, \tag{12.10}$$

where the sum $\sum'$ runs over all sequences with score S.

Thus, from the measured histogram of scores $p^*_T(S)$ one obtains the estimator for the unbiased distribution through

$$p(S) = \sum_c{}' P(c) = p^*_T(S) Z(T) \exp(-S/T)\,. \tag{12.11}$$

$Z(T)$ is unknown a priori, but can be determined very easily, as shown below.

Please note a striking difference to conventional Monte-Carlo (MC) simulations of random systems. For the conventional approach when studying quenched-disorder systems, different random samples of the quenched disorder are studied by MC, each sample having the predefined probability $P(c)$. Within the method presented here, a biased simulation is done *on the* *disorder*, while the behavior of each random sample is determined exactly, resulting finally in the unbiased distribution over the disorder. Note that to describe the behavior of $p(S)$ over a wide range, the model must be simulated at several temperatures. Hence, to enhance the efficiency of the simulation, the *parallel tempering* method [15, 21] may be applied, where several realizations are kept in parallel at different temperatures and exchanges between neighboring temperatures occur.

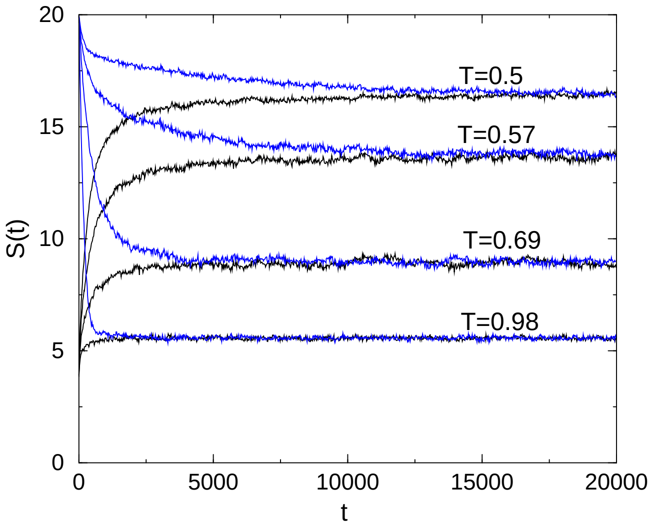


Figure 12.12: Average alignment score S as a function of step t for a toy model ($n, m = 20$, 4 letters, local alignment without gaps) for different temperatures T. For each temperature, 1000 independent simulations were started with two random sequences (low scores) and 1000 simulations with two equal sequences (high scores).

For the Monte Carlo simulation, one has to ensure *equilibration*. To check equilibration in our case, two types of runs are performed:

(a) Initially, all pairs of sequences are random, i.e. the score is low at the beginning.

(b) Initially, each pair consists of two equal sequences. Thus, the score is maximal at the beginning, i.e. the system is in the ground state.

In this way equilibration can be checked easily: if after time t_0 the score averaged over several independent runs for both initial configuration agree within error bars, the simulation is long enough.

Next, a simple example is given, illustrating how the method works. Optimum local alignments without gaps for sequences of equal length $m = n = 20$ and $r = 4$ letters, all having the same probability $1/4$, are calculated. For the test the following score is applied: $s(x, y) = 1$ if $x = y$ and $s(x, y) = -3$ otherwise. This is the same model as considered in the previous chapter to illustrate the alignment algorithms.

In Fig. 12.12 the average optimum score S for the beginning 10% of the running time of 1000 independent runs and four different temperatures T is shown. For not too small temperatures, the system equilibrates quickly. For very low temperatures (not shown in the figure), equilibration may not be possible, hence the data for these temperatures cannot be used. This means basically, that the artificial dynamics of the disorder exhibits a glassy phase, which might be worth of its own study. Anyway, as we will see, the data obtained at higher temperatures is sufficient to obtain the true distribution $p(S)$ over many orders of magnitude.
To obtain weakly correlated samples when measuring $p_T^*(S)$, only values at t_0, $t_0+\tau$, $t_0+2\tau$ etc. are taken, where τ is the characteristic time in which the score-score correlation $c_S(t_0,t) = (\langle S(t_0)S(t)\rangle - \langle S\rangle^2)/(\langle S^2\rangle - \langle S\rangle^2)$ decreases to $1/e$.

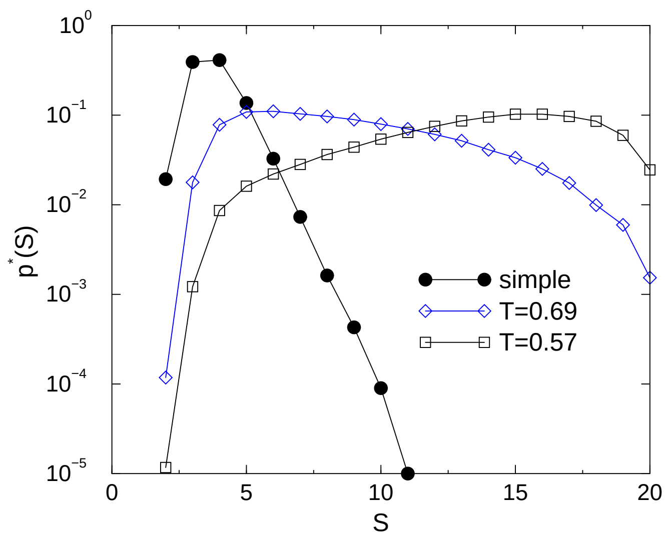


Figure 12.13: For the four-letter toy model: Raw distribution of alignment scores S for the direct simulation and the distributions $p_T^*(S)$ obtained at finite temperatures $T = 0.57$ and $T = 0.69$.

In Fig. 12.13 the raw distributions $p_T^*(S)$ for two temperatures is shown together with a distribution from a simple sampling of $N = 10^4$ realizations. Clearly, with the statistical mechanics approach, the region of high scores is sampled much more frequently.
The first step, to obtain the final distribution $p(S)$ according to Eq. (12.11), is to rescaled the measured distributions $p_T^*(S)$ with the Boltzmann factor $\exp(-S/T)$. The result is displayed in Fig. 12.14.
Hence, only the partition functions $Z(T)$ remain to be determined to obtain the final result. The starting point is the unbiased distribution which is known to be correct in the region of high probabilities. Now, in an interval $[S_1, S_2]$, the data of the simple sampling and the highest temperatures overlap. Hence, the finite-temperature data must be shifted (on a logarithmic scale) such that it becomes identical with the simple-sampling data, resulting in $Z(T)$ for

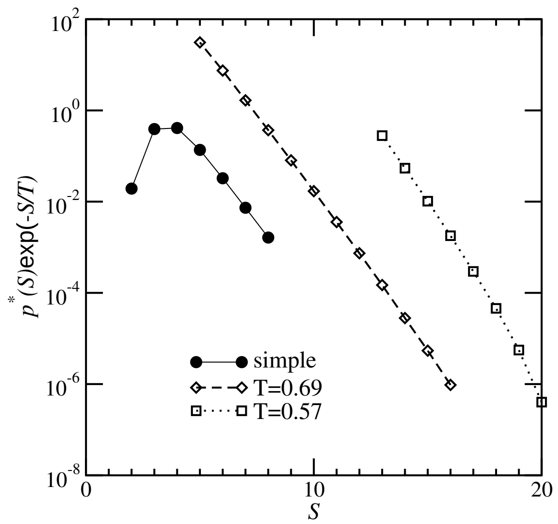


Figure 12.14: Distribution of alignment scores for the direct simulation and and rescaled distributions $p^*_T(S)\exp(-S/T)$ for $T = 0.57$ and $T = 0.69$. Only the regions with the best statistics are displayed.

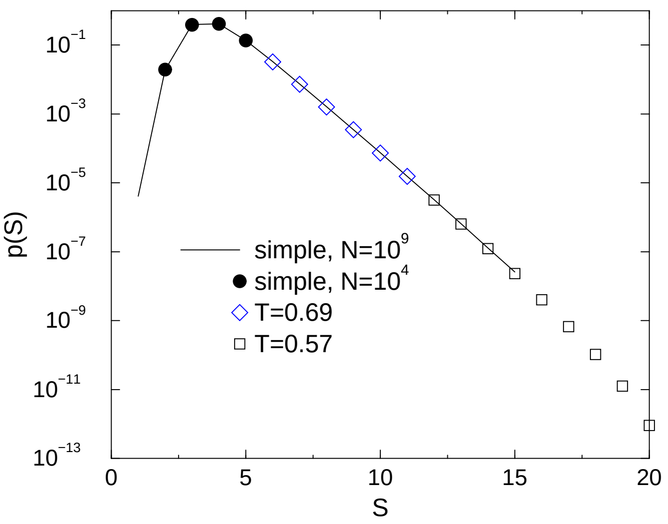


Figure 12.15: Rescaled distribution $p(S)$ for the direct simulation and for $T = 0.57$, $T = 0.69$. The solid line is the result of a large simple sampling simulation with $N = 10^9$ samples.

the highest temperature. In the same way $Z(T)$ at lower temperatures can be obtained by matching to distributions obtained before at higher temperatures. The final distribution is shown in Fig. 12.15. For each data point, the distribution with the highest accuracy was taken. For comparison, a simple sampling distribution obtained using a huge number of samples ($N = 10^9$) is shown. Both results agree very well. Please note that the distribution from the finite-T approach spans almost the entire interval $[0, 20]$. In principal, the region for very small score S can be investigated also using the new method by simulating at *negative* temperatures. How powerful the new method is can be seen by looking at the right border of the interval, where a value $p(20) = 9.13(20) \times 10^{-13}$ was obtained. This agrees within error bars with the exact result $0.25^{20} \approx 9.09 \times 10^{-13}$ (only when $\mathbf{x} = \mathbf{y}$ the highest possible score is achieved). Also the same model with $(3, 1)$ gap costs was tried and again a perfect agreement with a huge simple sampling simulation was found. This example illustrates that the method presented here is indeed able to calculate accurately the distribution $p(S)$ of optimum alignment scores in regions where $p(S)$ is very small.
In Ref. [17] the method was applied for a biologically relevant case. Sequences of amino acids distributed according the background frequencies by Robinson and Robinson [28] were used together with the BLOSUM62 scoring matrix [14] for $(12, 1)$ affine gap costs. This type of system was previously studied in Ref. [2] in the region where $p(S)$ is large. Sequences of length $n = m$ in the range $[40, 400]$ were considered. The main result was that for high scores significant deviations from the pure Gumbel behavior are visible, in contrast to the earlier predictions. Since the deviations occur at high score values, they could not be detected before using conventional methods. From an theoretic point of view it is interesting that with increasing lengths n, m, on a scale of scores proportional to $u \sim \log n$, the measured $p(S)$ approaches the Gumbel distribution more and more. For practical applications, where short and medium length sequences are relevant, the measured distribution instead of a fitted Gumbel should be used in databases like Swiss-prot [33] to achieve the highest possible accuracy.

Acknowledgements: The authors is grateful to Bernd Burghardt and Joachim Schmidt-Brauns for critically reading the manuscript. The author obtained finacial support from the VolkswagenStiftung (Germany) within the program "Nachwuchsgruppen an Universitäten".

Bibliography

[1] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman, *Basic local alignment search tool*, J. Mol. Biol. **215**, 403 (1990).

[2] S.F. Altschul and W. Gish, *Local alignment statistics*, Methods in Enzymology **266**, 460 (1996).

[3] S.F. Altschul, R. Bundschuh, R. Olsen, and T. Hwa, *The estimation of statistical parameters for local alignment score distributions*, Nucl. Acid Res. **29**, 351 (2001).

[4] B.A. Berg and T. Neuhaus, *Multicanonical ensemble: a new approach to simulate first-order phase transitions*, Phys. Rev. Lett. **68**, 9 (1992).

[5] S.M. Brown, *Bioinformatics*, (Eaton Publishing, Natick (MA) 2000).

[6] J.F. Collins, A.F.W. Coulson, and A. Lyall, *The significance of protein-sequence similarities*, CABIOS **4**, 67 (1988).

[7] M.O. Dayhoff, R.M. Schwartz, and B.C. Orcutt, *A model of evolutionary change in proteins*; in: M.O. Dayhoff (ed), *Atlas of Protein Sequence and Structure* **5**, suppl. 3, (National Biomedical Research Foundation, Washington D.C., 1978).

[8] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis*, (Cambridge University Press, 1998).

[9] W.H. Elliott and D.C. Elliott, *Biochemistry and Molecular Biology*, (Oxford University Press, Oxford 2001).

[10] O. Gotoh, *An improved algorithm for matching biological sequences*, J. Mol. Biol. **162**, 705 (1982).

[11] P. Grassberger, *Prune-enriched Rosenbluth method: Simulations of Θ polymers of chain length up to 1000000*, Phys. Rev. E **56**, 3682 (1997).

[12] E.J. Gumbel, *Statistics of Extremes*, (Columbia Univ. Press, New York 1958).

[13] S. Henikoff and J.G. Henikoff, *Automated assembly of protein blocks for database searching*, Nucl. Acids. Res. **19**, 6565 (1991).

[14] S. Henikoff and J.G. Henikoff, *Amino-acid substitution matrices from protein blocks*, Proc. Natl. Acad. Sci. USA **89**, 10915 (1992).

[15] K. Hukushima and K. Nemoto, *Exchange Monte Carlo method and application to spin glass simulation*, J. Phys. Soc. Jap. **65**, 1604 (1996).

[16] International Humane Genome Sequencing Consortium, Nature **409**, 860 (2001).

[17] A.K. Hartmann, *Sampling rare events: statistics of local sequence alignments*, Phys. Rev. E 65, 056102 (2002).

[18] V. Kalashnikov, *Geometric Sums: Bounds for Rare Events with Applications*, (Kluwer Academic Publishers, Dordrecht 1997).

[19] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, *Optimization by simulated annealing*, Science **220**, 671 (1983).

[20] S. Lipschutz and M. Lipson, *Probability*, (McGraw-Hill, New York 2000).

[21] E. Marinari and G. Parisi, *Simulated tempering — a new Monte-Carlo scheme*, Europhys. Lett. **19**, 451 (1992).

[22] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, J. Chem. Phys. **21**, 1087 (1953).

[23] R. Mott, *Maximum-likelihood estimation of the statistical distribution of Smith-Waterman local sequence similarity scores*, Bull. Math. Biol. **54**, 59 (1992).

[24] S.B. Needleman and C.D. Wunsch, *A general method applicable to search for similarities in amino acid sequence of 2 proteins* J. Mol. Biol. **48**, 443 (1970).

[25] H.H. Rashidi and L.K. Buehler, *Bioinformatics Basics*, (CRC Press, Boca Raton (FL) 2000).

[26] L.E. Reichl, *A Modern Course in Statistical Physics*, (Wiley, New York 1998).

[27] B.D. Ripley, *Stochastic simulation*, (Wiley, New York 1987).

[28] A.B. Robinson and L.R. Robinson, *Distribution of Glutamine and Asparagine residues and their near neighbors in peptides and proteins*, Proc. Natl. Acad. Sci. USA, **88**, 8880 (1991).

[29] P.J. Shah and O.G. Mouritsen, *Dynamics of ordering process in annealed diluted systems — island formation, vacancies at domain boundaries, and compactification* Phys. Rev. B **41**, 7003 (1990).

[30] T.F. Smith and M.S. Waterman, *Identification of common molecular subsequences*, J. Mol. Biol. **147**, 195 (1981).

[31] T.F. Smith, M.S. Waterman, and C. Burks, *The statistical distribution of nucleic-acid similarities*, Nucleic Acids Res. **13**, 645 (1985).

[32] R.B. Stinchcombe, in C. Domb and M.S. Green (eds.) *Phase Transitions and Critical Phenomena* **7**, (Academic, New York 1983).

[33] Siss-Prot, e.g. at the European Molecular Biology Laboratory (EMBL) in Heidelberg, see `http://dove.embl-heidelberg.de/Blast2/`.

[34] R. Voča, *Transport on an annealed disordered lattice*, Phys. Rev. E **60**, 3516 (1999).

[35] M.S. Waterman and V. Vingron, *Rapid and accurate estimates of statistical significance for sequence data-base searches*, Proc. Natl. Acad. Sci. USA **91**, 4625 (1994).

[36] M.S. Waterman and V. Vingron, *Sequence comparison significance and Poisson approximation*, Stat. Sci. **9**, 367 (1994).