A Variation on Algorithms for Pairwise Global Alignments

DIPLOMARBEIT
zur Erlangung des akademischen Grades
Magister rerum naturalium

Vorgelegt der
Fakultät für Naturwissenschaften und Mathematik
der Universität Wien

von
Ulrike Mückstein

am Institut für
Theoretische Chemie und Molekulare Strukturbiologie

im September 2001

Dank an alle,

die zum Gelingen dieser Arbeit beigetragen haben:
Peter Stadler, Ivo Hofacker, Peter Schuster.
Ingrid Abfalter, Aderonke Babajide, Stephan Bernhart, Martin Fekete, Christoph Flamm, Dagmar Friede, Kurt Grünberger, Jörg Hackermüller, Christian Haslinger, Philipp Kobel, Michael Kospach, Stefan Müller, Bärbel Stadler, Roman Stocsits, Andreas Svrcek-Seiler, Caroline Thurner, Günther Weberndorfer, Andreas Wernitznig, Stefanie Widder, Christina Witwer, Micheal Wolfinger, Daniela Dorigoni, Judith Jakubetz.
Irene und Heinz Dorfwirth, Eva und Heinz Mückstein, Lukas Endler, Raphael Mückstein und Ferdinand Eder.

Abstract

Alignment algorithms normally generate only a single optimal alignment. No information about possible alternative alignments or the reliability of the result is provided. We address this problem by calculating match probabilities and by generating a fairly distributed ensemble of optimal and suboptimal alignments.
The partition function of all possible alignments of two sequences is computed by means of a dynamic programming algorithm incorporating affine gap models. The algorithm is designed to avoid redundant solutions by determining the order of gaps in regions were gaps in one sequence are adjacent to gaps in the other sequence. This is important to avoid combinatorial explosion trough the generation of duplicate or non-canonical solutions.
The partition function can be used to determine the probability of an alignment as well as the probability of each possible match between two sequence positions. To obtain the probability of each match between positions i and j of the two sequences, respectively, the partition function over all alignments, which contain this match, is divided by the partition function over all possible alignments of the two sequences. For the calculation of match probabilities a detailed derivation of the recursion relations for partition functions of alignments of subsequences is needed.
Furthermore the partition function is used for stochastic backtracking generating a properly weighted ensemble of optimal and suboptimal alignments. In this way one acquires an overview of the alignment landscape and the entropy of the ensemble of stochastic alignments, that is a measure of the variety of the optimal and suboptimal alignments generated.
These methods were used to implement a program in C.

Zusammenfassung

Typische Alignmentalgorithmen generieren nur ein einziges optimales Alignment. Es werden keine Informationen über mögliche Alternativen zum optimalen Alignment oder über die Zuverlässigkeit der Ergebnisse geliefert. Wir lösen dieses Problem durch die Berechnung der Wahrscheinlichkeit eines Matches von zwei Resten in einem Alignment und indem wir ein fair verteiltes Ensemble von optimalen und suboptimalen Alignments erzeugen.
Die Zustandssumme aller möglich Alignments von zwei Sequenzen wird mittels eines Dynamic Programming Algorithmus, der ein affines Gapmodell beinhalted, berechnet. Der von uns entworfene Algorithmus vermeidet redundante Ergebnisse, indem die Abfolge von Gaps in Regionen, in denen Gaps in einer Sequence direkt auf Gaps in der anderen folgen, genau festgelegt wird. Diese ist wichtig um eine kombinatorische Explosion durch die Erzeugung von duplizierten oder nicht kanonischen Ergebnissen zu verhindern.
Die Zustandssumme kann sowohl für die Berechnung der Wahrscheinlichkeit eines Alignments als auch zur Berechnung der Wahrscheinlichkeit eines Matches zwischen zwei Resten verwendet werden. Um die Match Wahrscheinlichkeit von zwei Resten zu berechnen, wird die Zustandssumme über alle Alignments, in denen diese beiden Rest aligned sind, durch die Zustandssumme aller möglichen Alignments zwischen den beiden Sequenzen geteilt. Für die Berechnung der Match Wahrscheinlichkeit ist die detaillierte Ableitung der Rekursionsbeziehungen der Zustandssummen von Alignments von Teilsequenzen nötig.
Die Zustandssumme wird auch für ein stochastisches Backtracking verwendet. Mittles stochastischen Backtrackings kann man ein korrekt gewichtetes Ensemble von optimalen und suboptimalen Alignments erzeugen. Diese Methode verschafft einen Überblick über die Alignment Landschaft und die Entropie der Ensembles von stochastischen Alignments, die ein Maßfür die Vielfalt von optimalen und suboptimalen Alignments ist.
Diese Methoden wurden in einem Programm in C implementiert.

Contents

1 Introduction

In the last decades the use of alignment programs to compare DNA or amino acid sequences has become a routine task. Computer programs to align sequences, search a database with a given sequence or compare multiple sequences are readily available.

The standard approach of most alignment programs is to find the best scoring alignment between a pair of sequences. Two types of alignment methods, that are based on dynamic programming, are commonly used: Global alignment methods, that aligne the sequences along their whole length [52] and local homology search methods, that find significantly similar segments between sequences [58]. Basically there are two ways an alignment can be scored, either maximizing the similarity or minimizing the distance between two sequences [63, 56, 59]. We will focus on global similarity alignments here.

Although alignment algorithms are designed to detect similarities, they produce alignments and scores also when applied to sequences that are not related at all [60]. An important question for a biologist faced with the results of an alignment program is therefore the question about the accuracy of the proposed alignment. One way to address this question is the use of a statistical model of evolution. In a computer simulation both sequences are scrambled using random permutations and the scrambled sequences are aligned over and over again. In this way asymptotic estimates of the mean and the variance of the optimal alignment scores can be derived. If the alignment score of the original pair of sequences is significantly better than the expected score, one accepts the computed alignment as significant [37]. Another approach is to simulate random mutation of a pair of initially identical sequences and then feed the mutated sequences into the alignment algorithm to obtain a measure of accuracy of the alignment algorithm [33].

The methods outlined above yield an overall judgment whether two sequences are related. They do not deal with the fact that there are regions of differential probability of substitution along a nucleic acid or protein sequence [61, 23]. These regions of different variability along the sequences are

reflected in regions of higher and lower reliability in an alignment. The aim of our work is to provide detailed information on what parts of an alignment are uniquely defined and what parts display ambiguities.
Alignment algorithms are designed to optimize a mathematical score [71]. In the context of a similarity scoring system the scores rewards aligning similar residues and penalizes substitutions and gaps. Therefore an optimal alignment is the alignment that exhibits the most correspondences and the least differences [19]. The "optimal solution" found by common alignments algorithms is not unique. There can be different alignments of two sequences that have the same optimal score. Furthermore, there may be relevant near-optimal solutions [18], that also contribute to the biological correct alignment. Therefore a single "optimal" answer is often unsatisfactory. Different approaches have been used to evaluate the regional difference in the reliability of a pairwise alignment by including near-optimal and suboptimal solutions [55, 51, 47]. Vingron and Argos [66], for example, evaluated near optimal alignments to introduce the concept of 'stable regions'. A stable region in an alignment is one which is used by all optimal and suboptimal alignments, whose score is at most a given threshold value away from the score of the optimal alignment. When only a small threshold value is allowed, few suboptimal alignments score within this threshold and a large fraction of the optimal alignment will be stable. Upon increasing the threshold value, the stable regions decrease in size because more suboptimal alignments are taken into account and compete with the optimal alignment. Therefore residue matches that remain stable even when a large threshold value allows for the consideration of many suboptimal alignments, are likely to be aligned reliably.
All these methods have in common that they use only a restricted set of suboptimal alignments to derive their solutions. Dynamic programming can also by applied to calculate the partition function over all solutions of a search space. McCaskill [46] calculated the full equilibrium partition function for RNA secondary structure and the probabilities of all base pairs by

a recursive scheme of polynomial order N^3 in the sequence length N. In the Vienna RNA Package [32] a partition function algorithm is used to calculate base pair probabilities in the thermodynamic ensemble. In RNA secondary structure prediction computation of the partition function is an frequently applied method. For alignments the partition function was first used by Miyazawa [49] to determine the probabilities of all possible matches and indels between two sequences. Miyazawa used the probabilities of all possible correspondences between residues to construct alignments consisting only of highly probable matches. Kschischo and Lässig [41] developed a theory of probabilistic alignments derived from a thermodynamic partition function, they called *finite-temperature* alignments. The key applications of finite-temperature alignments are estimation of the reliability of single matches in an alignment and assessing the relative significance of different local alignments with high scores.

In this work we used the partition function to calculate the probabilities of all possible matches between two sequences and for stochastic backtracking. For the calculation of the partition function a variation of a dynamic programming algorithm that is designed to avoid redundancy is employed. Furthermore, we introduced a variable parameter that influences the relative weight of alignment paths with different scores.

The match probabilities can be displayed conveniently in a dot plot (see Figure 1). In this representation conserved regions of the alignment are visualized by runs of larger dots parallel to the diagonal of the plot. Each dot symbolizes a match between different positions of each sequence, the size of the dot is proportional to the probability of this match. The partition function is also used for stochastic backtracking. Repeated application of the stochastic backtracking generates randomly selected alignments, with frequencies according to the probability of each individual alignment. Thus the stochastic backtracking provides a fairly distributed ensemble of possible alignments.

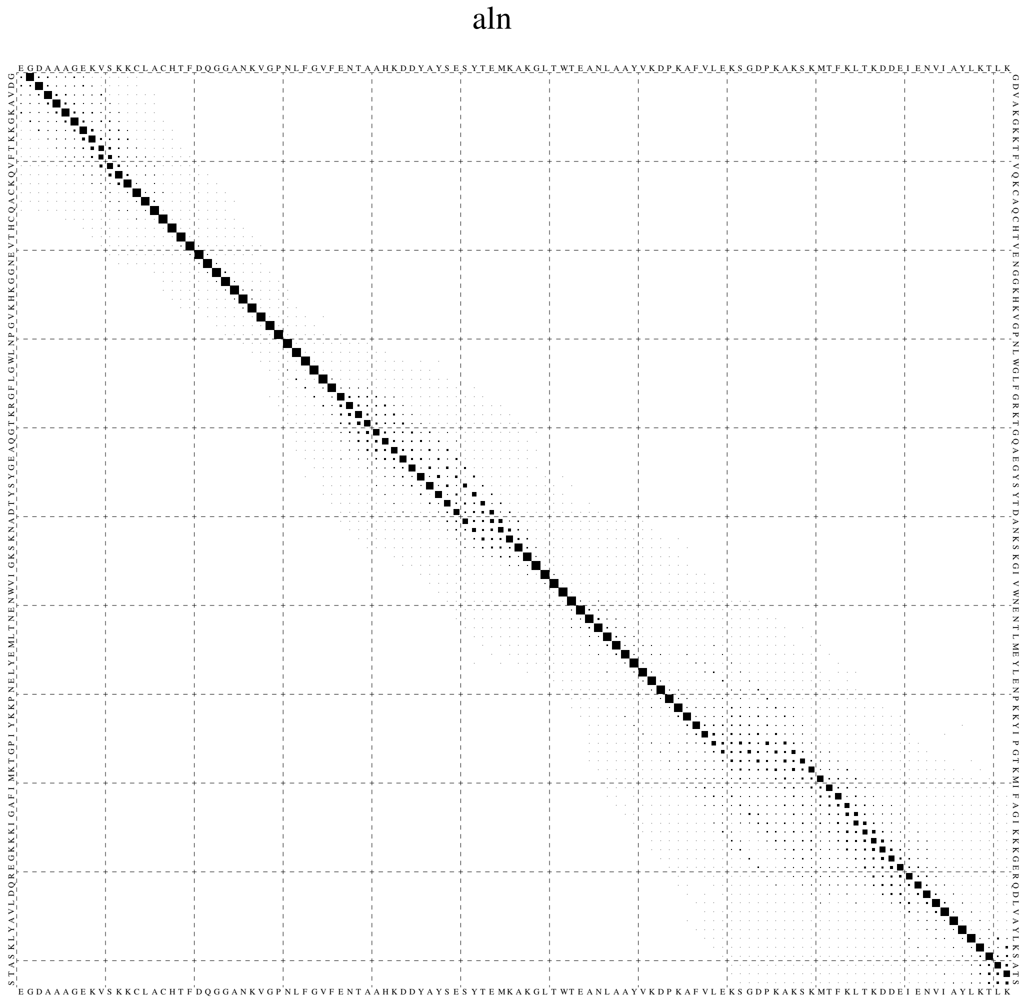


Figure 1: Dot plot of the alignment between cytochrome c from skipjack tuna, *Euthynnus pelamis*, (CCBN) and *Rhodospirillum rubrum* cytochrome c_2 (CCQF2R; position 24 - 135). The codes given in parenthesis are taken from the PIR protein sequence database [6]. The horizontal axis of the dot plot is labeled by CCQF2R, the vertical axes by CCBN. Regions of high sequence similarity are illustrated by large black dots representing high match probabilities. In regions of lower sequence similarity many different possibilities to align the two sequences exits. The different possibilities to align these regions are depicted by a multitude of small dots, each representing a match of low probability. The alignment was prepared using the score matrix Gonnet 120 (see section 3.3.1), $T = 0.4$ (see section 2.6).

2 Theory

2.1 General Concepts

The notion that genomic DNA is the blueprint for a living organism leads to the idea that evolution must be directly related to changes in DNA. The history of these changes is called molecular evolution. The simplest events that occur during the course of molecular evolution are substitution of one base for another and the insertion or deletion of one base.
An alignment represents a specific hypothesis about the evolution of the sequences. The alignment of two sequences is essentially build up by four different states. Figure 2 illustrates the four states of an alignment: If the corresponding residues in the alignment are identical, this is referred to as a match. Aligning two functionally conserved residues, such as the F-Y pair, which represents an alignment of a phenylalanin residue with a tryrosin residue, both aromatic amino acids, is called a mismatch or substitution. An insertion or deletion is one or more residues aligned to a gap symbol ('-'). It is not possible to distinguish insertions from deletions in an alignment. In the alignment in figure 2 the insertion of CT in the above sequence might also have been a deletion in the lower sequence.

```
C P S G C T N F K - C A
C P T G - - N Y K K C A
```

Figure 2: This figure shows a pairwise alignment of two protein sequences **a** = {**CPSGCTNFKCA**} and **b** = {**CPTGNWKKCA**}. The four building states of the alignment are: match (aligned residues are identic), mismatch (aligned residues are different), deletion (residues in one sequence have been deleted in the other sequence) or insertion (residues have been added to one of the sequences).

Insertions and deletions are referred to as indels. In the case of indels two states have to be distinguished: indels in the upper sequence and indels in the lower sequence.

2.2 Scoring Models

To compute the "best alignment" between two sequences, we need a way to score the alignment. The simplest way to give a score to an alignment of two sequences is to calculate their Hamming distance: For two sequences of equal length, we count the different positions [19]. This distance measure is in general not flexible enough. Sequences may have different lengths and corresponding residues may have been shifted to different positions by deletions or insertions. We need a more flexible scoring system to take this into account: The total score we assign to an alignment should be a sum of terms for each aligned pair of residues, plus terms for each indel.
The scoring systems discussed here all use similarity scores. The concept of similarity alignments is to reward similarity (e.g. matching residue a with residue a) by a positive score, s$(a, a) > 0$, whereas aligning dissimilar residues is penalized by a negative score or gets a score of zero, s$(a, b) \leq 0$. In the simplest case linear gap penalties are used, aligning a residue to a gap is penalized by a negative score, s(a, -) = s(-, a) < 0. The standard cost, $\gamma(l_g)$ of a gap of length l_g is given by:

$$\gamma(l_g) = -l_g\, d$$

where d is the gap penalty. The pairwise alignment score for two sequences **a** and **b**, S(**a**, **b**), is therefore the sum of substitution scores and gap penalties over all aligned residues, so that the best alignment is the one with the highest score, where L is the length of the alignment.

$$S(\mathbf{a}, \mathbf{b}) = \max \sum_{i=1}^{L} s(a_i, b_i),$$

2.2.1 Substitution matrices for nucleic acid alignments

In the case of nucleic acid alignments, the scoring model can be applied as described above. Only two substitution scores are used: a match between two residues is rewarded by a positive score, whereas a mismatch is penalized.

Scores can be represented in the form of a 4×4 “unitary” matrix consisting of ones (or another positive score) on the diagonal for scoring A-A, C-C, G-G and T-T matches and zeros (or a negative score) for scoring mismatches. Generally a substitution in one direction, e.g. a C-G mismatch, is similarly scored as a substitution in the other direction, in this case a G-C mismatch, the scoring matrix is therefore symmetrical, and only half of the off-diagonal scores are needed to provide a complete scoring scheme for residue substitutions. The scores for nucleic acid alignments used in this work are taken from ClustalW [62]. In ClustalW a ”unitary” matrix that scores one for matches and zero for mismatches is used. Other scoring schemes for nucleic acids use different score for transition and transversions [16]. For protein alignments a more elaborate scoring model is applied, taking into consideration how often different amino acids are replaced by other amino acids in evolution and the relative mutability of different amino acids.

2.2.2 Substitution matrices for protein alignments

There are several ways weights can be assigned for differences in amino acid sequences. Some methods propose scores based on chemical, functional, charge and structural properties of the amino acids [38, 26]. Other scoring systems take into account structural similarities of amino acids and the ease of genetic interchange [15, 12] or rely on structural superpositions of proteins [36, 53]. However by far the most common way is to use Dayhoff’s PAM substitution matrix series [10] and other empirically based log-odds matrix series e.g. Gonnet’s matrix series [21, 8] or Henikoff’s BLOSUM matrix series [29, 28]. The empirical log-odds matrices used in this work are similarity matrices. They contain values proportional to the probability that one amino acid mutates into another amino acid for all pairs of amino acids. Each matrix of a matrix series is defined for the alignment of proteins with certain evolutionary distances. The evolutionary distance is measured in PAM (Percent Accepted Mutations) units. A PAM 1 mutation matrix describes an amount of evolution which will change, on average, 1% of the

amino acids. To derive a mutational probability matrix for a protein sequence that has undergone N percent accepted mutations, a PAM n matrix, the PAM 1 matrix is multiplied by itself n times. This results in a family of scoring matrices.
Altschul [2] developed a statistical theory of log-odds score systems. A log-odd score is the logarithm (to an appropriate base) of the ratio between 2 probabilities. Durbin *et al.* [11] showed that in the case of alignments, a log-odds score will correspond to the logarithm of the relative likelihood that the sequences are related, compared to being unrelated. According to Durbin *et al.* identities and conservative substitutions are expected to be more likely in alignments than by chance, and therefore contribute positive score terms. Non conservative changes are expected to be observed less frequently in real alignments than by chance, and thus contribute negative score terms. Using an additive scoring scheme corresponds to the assumption that mutations at different sites in a sequence have occurred independently.
For the construct of a substitution matrix score terms are needed for each aligned residue pair. Consider a pair of sequences $\mathbf{a}$ and $\mathbf{b}$, of lengths m and n, respectively. Let a_i be the ith symbol of $\mathbf{a}$ and b_j be the jth symbol of $\mathbf{b}$. These symbols will come from some alphabet $\mathcal{A}$: In the case of DNA $\mathcal{A}$=$\{\mathsf{A},\mathsf{T},\mathsf{G},\mathsf{C}\}$ and in the case of proteins the twenty amino acids. Given a pair of aligned sequences, you want to assign a score to the alignment that gives a measure of the relative likelihood that the sequences are related as opposed to being unrelated:
The random model, R, assumes that a letter a of some alphabet $\mathcal{A}$ occurs independently with some frequency q_a and a letter b occurs independently with some frequency q_b. Therefore, the probability that two letters a and b are aligned to each other is the product of their individual frequencies:

$$P(a, b \mid R) = q_a q_b$$

Considering the random model the probability of the alignment of two sequences, $\mathbf{a}$ and $\mathbf{b}$, is hence the product of the probabilities for each letter:

$$P(\mathbf{a},\, \mathbf{b} \mid R) = \prod_i q_{a_i} \prod_j q_{b_j}$$

In the alternative match model, M, aligned pairs of residues occur with a joint probability p_{ab}. The value p_{ab} is equivalent to the probability that the residues a and b have each independently derived from the same original residue in their common ancestor. The probability that two residues are aligned is therefore given by:

$$P(a, b \mid M) = p_{ab}$$

This probability for the whole alignment between sequences **a** and **b** under the match model is then the product of the joint probabilities of all aligned letters:

$$P(\mathbf{a},\, \mathbf{b} \mid M) = \prod_i p_{a_i b_i}$$

The ratio of these two likelihoods is known as the *odds ratio*. The odds ratio for matching two letters a and b is equivalent to the probability that the two residues a and b haven been independently derived from some common ancestor in contrast to be of different origin.

$$\frac{P(a, b \mid M)}{P(a, b \mid R)} = \frac{p_{ab}}{q_a q_b} = p(a, b)$$

And the odds ratio of the alignment between sequences **a** and **b** is the product of the odd ratios for each match:

$$\frac{P(\mathbf{a}, \mathbf{b} \mid M)}{P(\mathbf{a}, \mathbf{b} \mid R)} = \prod_i \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}$$

In order to develop an additive scoring system, we take the logarithm of this ratio, known as the *log-odds ratio*. Therefore the score for aligning residue a and residue b, $s(a, b)$, is the log likelihood ratio of the residue pair (a, b) occurring as an aligned pair, as opposed to a nonaligned pair. Usually an affine transformation of this log likelihood ratio is performed, where the constants ω and k are selected as to obtain expedient scores:

$$s(a,b) = \omega + k \log \frac{p_{ab}}{q_a q_b} \tag{1}$$

The total score of the alignment of sequence a and sequence b is the sum of scores for each aligned pair of residues, plus the terms for gaps. The $s(a,b)$ scores can be arranged in a matrix. For proteins the scores form a 20×20 matrix, with $s(a_i, b_j)$ in position i, j in the matrix, where a_i, b_j are the amino acid in the ith row and the jth column of the matrix. This representation is known as a score matrix or a substitution matrix. Since it is impossible to distinguish between the two possible directions of a substitution, the matrix is symmetric about the diagonal. It is sufficient to use only one of the triangles of the 20×20 scoring matrix, because substitution scores are symmetric. An example of a substitution matrix derived essentially as described above is shown in figure 3.

2.2.3 Gap penalties

The inclusion of gaps and gap penalties is necessary to obtain the best possible alignment between two sequences. As discussed earlier (see page 11), the simplest way to penalize gaps is a linear gap penalty. In the case of linear gap penalties a fixed amount is charged for each residue aligned to a gap symbol. Thus the cost of a gap is proportional to its length, where d is the gap penalty and l_g is the length of the gap.

$$\gamma(l_g) = -l_g\, d$$

The observation that a single mutational event can delete or insert multiple residues suggests that a long gap should not cost substantially more than a short gap [14, 48, 25]. A particular mutation involving a single gap of n residues should therefore be more likely than n consecutive mutations resulting in n single gaps. The simplest way of implementing this feature of gap

Cys	C	12																			
Ser	S	0	2																		
Thr	T	−2	1	3																	
Pro	P	−3	1	0	6																
Ala	A	−2	1	1	1	2															
Gly	G	−3	1	0	−1	1	5														
Asn	N	−4	1	0	−1	0	0	2													
Asp	D	−5	0	0	−1	0	1	2	4												
Glu	E	−5	0	0	−1	0	0	1	3	4											
Gln	Q	−5	−1	−1	0	0	−1	1	2	2	4										
His	H	−3	−1	−1	0	−1	−2	2	1	1	3	6									
Arg	R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	8								
Lys	K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5							
Met	M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6						
Ile	I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5					
Leu	L	−8	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	8				
Val	V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4			
Phe	F	−4	−3	−3	−5	−4	−5	−4	−6	−5	−5	−2	−4	−5	0	1	2	−1	9		
Tyr	Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10	
Trp	W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17
		C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W

Figure 3: The PAM250 substitution matrix. The log-odds values have been scaled and rounded to the nearest integer for purposes of computational efficiency. In a log-odds matrix, positive scores indicate a given pairing is more likely to occur in related sequences than in random sequences, e.g., the score for matching a glutamate (E) with an aspartate (D) has score 3. Zero scores indicate matches, that are as likely to occur in related sequences as in random sequences and negative scores indicate matches, that are more likely to occur in random sequences than in related ones.

penalties is to use a larger gap open penalty (g_o), which is charged once for opening the gap, and a smaller gap extension penalty (g_{ext}) for each extra position in the gap. This method for penalizing gaps is called affine gap penalty [4, 67]:

$$\gamma(l_g) = -(g_o + g_{ext}(l_g - 1))$$

Durbin et. al. [11] demonstrated that gap penalties also correspond to a probabilistic model of alignment. They pointed out that the probability of a gap occurring at a particular site in a given sequence is the product of a function $f(l_g)$ of the gap-length and the combined probability of the set of inserted residues:

$$P(\mathbf{g}) = f(l_g) \prod_{i \text{ in } \mathbf{g}} q_{a_i} \tag{2}$$

The representation of the probability of a gap as the product of $f(l_g)$ with the q_{a_i} terms corresponds to an assumption that the length of the gap is not correlated to the residues it contains. The values for the q_a probabilities here are the same as those used in the random model (see page 13), because they both correspond to unmatched independent residues. If we divide the probability of the gap by the probability of the gapped region according to the random model to form the odds ratio, the q_a terms cancel out, so we are left only with a term dependent on length:

$$\gamma(l_g) = \log f(l_g)$$

If an affine transformation is performed on the substitution scores (see page 15), the same transformation has to be applied to the gap penalties, in this case gap penalties are calculated as:

$$\gamma(l_g) = \omega + k \, \log f(l_g) \tag{3}$$

In the probabilistic interpretation gap penalties correspond to the log probability of a gap of length l_g.

2.3 The Number of Alignments

Waterman [69] presents a brief combinatorial treatment of alignments to estimate the number of different alignments between two sequences. Given two sequences a with length m and b with length n, the problem is in how many ways $\mathsf{a} = \{a_1, a_2, \dots a_i, \dots a_m\}$ can be aligned with $\mathsf{b} = \{b_1, b_2, \dots b_j, \dots b_n\}$. One way to think of an alignment is that an alignment is produced when gaps, – , are inserted into the sequences, the aligned sequences must both be of the same length. It is not allowed to match a gap symbol with a gap symbol. Therefore the length of the alignment, L, of a with b has to be between $\max[m, n] \leq \mathsf{L} \leq m + n$. The case $\mathsf{L} = m + n$ comes by first deleting all a_i and then deleting all b_j:

$$\begin{array}{l} \text{— — — —}\ b_1\ b_2\ \dots\dots\ b_n \\ a_1\ a_2\ \dots\dots\ a_n\ \text{— — — —} \end{array}$$

For combinatorics on alignments it is important to recognize that alignments can end in exactly three ways:

$$\begin{array}{lll} \dots b_n & \dots b_n & \dots\text{—} \\ \dots\text{—} & \dots a_m & \dots a_m \end{array}$$

where $\binom{b_n}{-}$ corresponds to an indel of b_n, $\binom{b_n}{a_m}$ corresponds to an identity or substitution, and $\binom{-}{a_m}$ corresponds to an indel of a_m. The fate of the bases that are not displayed is not specified. Let $f(m, n)$ denote the number of alignments between the two sequences **a** and **b**. The recursion for $f(m, n)$ is:

$$f(m, n) = f(m-1, n) + f(m-1, n-1) + f(m, n-1)$$

For two sequences of length 1000, for example, we have $f(1000, 1000) \approx 10^{767}$ alignments. There are approximately 10^{80} particles in the universe; Avogadro's number is in the order of 10^{23}. It is therefore obvious that one cannot examine all alignments.

If it is agreed not to count permutations such as

$$\begin{array}{cc} \mathrm{G} & \_ \\ \_ & \mathrm{C} \end{array} \quad \text{and} \quad \begin{array}{cc} \_ & \mathrm{G} \\ \mathrm{C} & \_ \end{array}$$

as distinct, the situation improves slightly. The new way of counting alignment traces is to ignore permutations of $\begin{smallmatrix} a_l & a_{l+1} & - \\ - & - & b_k \end{smallmatrix}$. Waterman points out that the key is to realize that there must be k aligned pairs, $0 \leq \mathsf{k} \leq \min(m, n)$. There are $\binom{m}{\mathsf{k}}$ ways to choose a's and $\binom{n}{\mathsf{k}}$ ways to choose b's, so there are $\binom{m}{\mathsf{k}}\binom{n}{\mathsf{k}}$ alignments with k aligned pairs:

$$g(m, n) = \sum_{\mathsf{k} \geq 0} \binom{m}{\mathsf{k}} \binom{n}{\mathsf{k}} = \binom{m+n}{n}$$

If $g(m, n)$ is defined as above, $g(0,0) = g(1,0) = g(0,1) = 1$, $g(m,n) = \binom{m+n}{n}$ and $n = m$

$$g(n, n) = \binom{2n}{n} \sim \frac{1}{\sqrt{n\pi}} 2^{2n}, \text{ for long } n$$

Two sequences with $n = m = 1000$ have $g(1000, 1000) \approx 10^{600}$ alignment traces, so that a direct search is still impossible.

2.4 Alignment Algorithms

The first method for generating sequence alignments was described by Needleman and Wunsch [52] and was based on maximizing the similarity score between sequences. Waterman [69] pointed out that the method used by Needleman and Wunsch fits into a broad class of algorithms introduced by Richard Bellman under the name dynamic programming [7]. Dynamic programming is a programming technique that is designed for problems that

generate an exponential search space in a structurally recursive fashion. Dynamic programming can evaluate such a search space in polynomial time if subproblems are shared and the principle of subproblem optimality holds [17]. A dynamic programming algorithm can be dived into two subtasks: The "first" phase is the construction of the search space, in the case of alignments this is the set of all possible alignments between two sequences. During the "second" phase these alignments are evaluated and a choice is made according to some optimality criterion.
During the 1970s a mathematical model of the distance $d(a, b)$ between sequences was developed [63, 56]. David Mount [50] reviewed the work of the mathematician Peter Sellers [56], who constructed a metric space on the space of sequences. To model a metric space of sequences the alignment was formulated in terms of distance instead of similarity between sequences. The distance is the number of changes that must be made to convert one sequence into the other and represents the number of mutations that will have occurred following separation of the genes during evolution. Such an "edit-distance", defined as the minimum cost of converting one sequence into another via a series of edit operations, always fulfills the axioms of a metric

1. $d(a, b) = 0$ if and only if $a = b$,
2. $d(a, b) = d(b, a)$ (symmetry),
3. $d(a, b) \leq d(a, c) + d(c, b)$ for any c (triangle inequality),

provided the cost of the individual edit operations are positive and symmetric. For sequence alignments the edit operations are: point mutation (replacement of one letter by another) and insertions of gaps. An alignment thus represents the most likely evolutionary history of two sequences. Sellers also showed that the smallest number of steps required to change one sequence into the other could be calculated by the dynamic programming algorithm. Optimal alignments can be computed via the same schemes for maximum similarity by replacing the minimum distance by a maximal similarity scoring scheme.

2.4.1 The Needleman-Wunsch Algorithm

The idea for the recursion used to find an optimal alignments is to build the alignment by utilizing previous solutions for optimal alignments of smaller subsequence. The simplest way to find an optimal alignment is to use a dynamic programming algorithm with a linear gap penalty $\gamma(l_g) = -gd$, where l_g is the length of the gap and d is the gap penalty.
For the calculation of a global optimal alignment between two sequences **a** of length m and **b** of length n an $(m+1, n+1)$ matrix, M, is constructed [19]. The value $M(i,j)$ is the score of the best alignment between the initial segment $a_{1...i}$ of sequence **a** up to a_i and the initial segment $b_{1...j}$ of sequence **b** up to b_j. The matrix M is build recursively. We begin by initializing $M(0,0) = 0$, $M(i,0) = -id$ and $M(0,j) = -jd$. The values $M(i,0)$ represent an alignment of all residues of sequence **a** to gaps. If the values $M(i-1,j-1)$, $M(i-1,j)$ and $M(i,j-1)$ are known, it is possible to calculate $M(i,j)$: The three ways an alignment can be extended up to position (i,j) are shown in figure 4:

i a_i can be aligned to b_j, in which case $M(i,j) = M(i-1,j-1) + s(a_i, b_j)$,

ii a_i can be aligned to a gap, in which case $M(i,j) = M(i-1,j) - d$,

iii b_j can be aligned to a gap, in which case $M(i,j) = M(i,j-1) - d$.

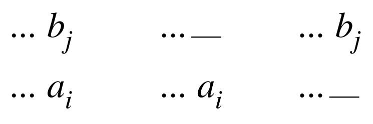


Figure 4: The alignment up to position (i,j) can be obtained by aligning a_i to b_j, by introducing a gap in sequence **b** and aligning a_i to this gap or vice versa by introducing a gap in sequence **a** and aligning b_j to this gap.

The recursion to find the optimal alignment is therefore:

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(a_i,b_j) \\ M(i-1,j) - d \\ M(i,j-1) - d \end{cases} \tag{4}$$

The best score for an alignment of $a_{1\ldots m}$ and $b_{1\ldots n}$ is, by definition, the value of the final cell of the matrix, $M(m,n)$. In order to retrieve an optimal alignment, a path of choices that led to value $M(m,n)$ has to be recovered. The method for finding this path is known as backtracking. Backtracking works by building the alignment in reverse, starting from the final cell, $M(m,n)$, we move back to the cell (or one of the cells) from which the value of $M(m,n)$ was derived. At each step of the backtracking process, there are three possible values from which the value of $M(i,j)$ could have been calculated:

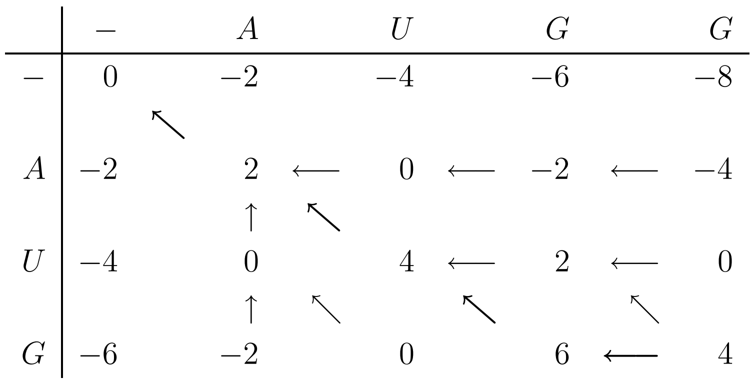


Figure 5: A global dynamic programming matrix for sequences $\mathbf{a} = \{\mathbf{AUGG}\}$ and $\mathbf{b} = \{\mathbf{AUG}\}$. The score for a match is 2, the score for a mismatch is 0, the gap penalty is -2. Arrows indicate the cell (or the cells) from which the $M(i,j)$ values have been derived. The path of choices for an optimal alignment is marked by bold arrows.

i $M(i,j)$ was generated by adding $s(a_i,b_j)$ to the value of cell $M(i-1,j-1)$, in this case residues a_i is aligned to residue b_j in the current alignment.

ii $M(i,j)$ was computed by subtracting the gap penalty (d) from the value of cell $M(i-1,j)$, in this case residue a_i is aligned to a gap.

iii $M(i,j)$ was derived by subtracting the gap penalty (d) from the value of cell $M(i,j-1)$, therefore residue b_j is aligned to a gap.

At the end of the backtracking process we reach the start of the matrix, $M(0,0)$. An example of the backtracking procedure is shown in figure 5. The backtracking procedure described here finds only one alignment with the optimal score. If for any value $M(i,j)$ two or three possibilities to compute this value exist, an arbitrary choice is made between the equal options.

2.4.2 Gotoh's Algorithm

The dynamic programming algorithm introduced by Needleman and Wunsch was subsequently improved by Gotoh [24], who adjusted the algorithm for use with affine gap penalties. In the case of affine gap penalties, there are six possibilities to extend an alignment up to position (i,j), see Figure 6.

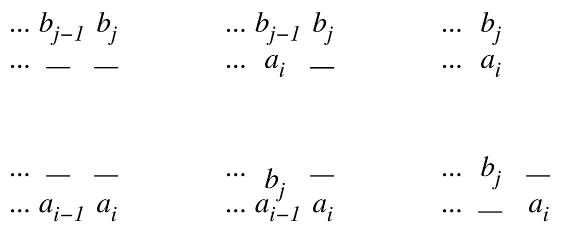


Figure 6: The six ways of extending an alignment up to position (i,j) if affine gap penalties are used.

Three of the possibilities to extend an alignment were already discussed in the case of linear gap penalties: aligning a_i with b_j, aligning a_i to a gap and aligning a gap to b_j. When using affine gap penalties of the form $\gamma(l_g) = -(g_o + g_{ext}(l_g - 1))$, opening a gap is penalized with a larger gap penalty than extending an existing gap. Two further ways to expand the alignment are therefore the extension of a gap in one of the sequences. The sixth possibility to elongate the alignment is that closing of a gap in one sequence is ensued by opening a gap in the other sequences. As already indicated (see page 19) permutations like $\begin{smallmatrix}\mathbf{G} & -\\ - & \mathbf{C}\end{smallmatrix}$ and $\begin{smallmatrix}- & \mathbf{G}\\ \mathbf{C} & -\end{smallmatrix}$ are not counted as distinct. In order to generate non-ambiguous alignments it is necessary to select one of these two similar configurations and to avoid the other.

$$M(i,j) = \max \begin{cases} M(i-1,j-1)+s(a_i,b_j) \\ E(i-1,j-1)+s(a_i,b_j) \\ F(i-1,j-1)+s(a_i,b_j) \end{cases}$$

$$E(i,j) = \max \begin{cases} M(i,j-1) & - & g_o \\ E(i,j-1) & - & g_{ext} \end{cases}$$

$$F(i,j) = \max \begin{cases} M(i-1,j) & - & g_o \\ E(i-1,j) & - & g_o \\ F(i-1,j) & - & g_{ext} \end{cases} \tag{5}$$

To compute an alignment using Equation (5), three $(m+1, n+1)$ matrices are needed: The first matrix, M, contains matches and mismatches between the two sequences: A match (or mismatch) of a_i and b_j can be added following another match $M(i,j) = M(i-1,j-1) + s(a_i,b_j)$ or the alignment can be extended with a match following a gap in either of the sequences $M(i,j) = E(i-1,j-1) + s(a_i,b_j)$ or $F(i-1,j-1) + s(a_i,b_j)$.
The auxiliary arrays E and F are necessary to keep track of the length of gaps. Array E is used to compute gaps introduced in the sequences indexed with i (sequence $\mathbf{a}$). A gap in sequence $\mathbf{a}$ can be opened following a match (or mismatch) $E(i,j) = M(i,j-1) + g_o$ or a gap can be extended after it has been opened $E(i,j) = E(i,j-1) + g_{ext}$. The auxiliary F keeps track of gaps in the sequence indexed by j (sequence $\mathbf{b}$). In addition to the options gap open after a match or mismatch, $F(i,j) = M(i-1,j) + g_o$, and gap

extend, $F(i,j) = F(i-1,j) + g_{ext}$, a third option, opening a gap in sequence **b** immediately following a gap in sequence **a**, $F(i,j) = E(i-1,j) + g_o$, is available. This asymmetry of the recursion ensures unambiguous alignments. In case a gap in one sequence is directly followed by a gap in the other sequence, the gap in sequence **a** is always located in front of the gap in sequence **b**. Such an algorithm can execute in *O(mn)* time, as was shown by Gotoh [24].

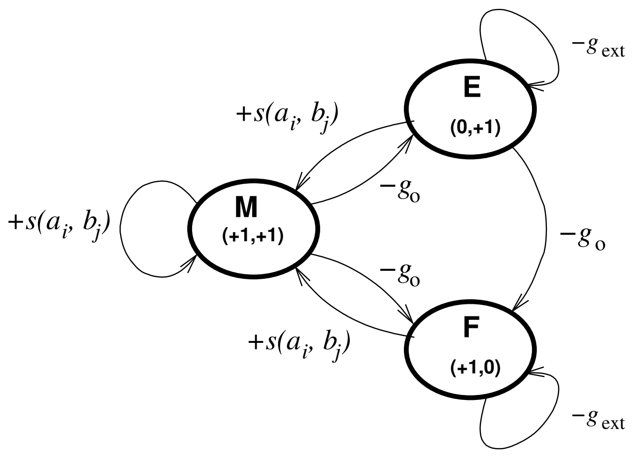


Figure 7: This diagram shows the relationship between the three states of an alignment with affine gaps penalty. The three states of the alignment are: M match (or mismatch), E gaps in the sequence indexed with i and F gaps in the sequence indexed with j.

The backtracking procedure for the algorithm with affine gap penalties is essentially the same as for an algorithm with linear gap penalties: The start position for backtracking is $\max\{M(m,n), E(m,n), F(m,n)\}$. Then we look for the cell from which $\max\{M(m,n), E(m,n), F(m,n)\}$ was derived. Backtracking now proceeds similar to the linear case:

i $M(i,j)$ was calculated by adding $s(a_i, b_j)$ to either $M(i-1, j-1)$, $E(i-1, j-1)$ or $F(i-1, j-1)$, in each case a_i is aligned to b_j.

ii $E(i,j)$ was derived either by subtracting the gap open penalty from the value in position $M(i, j-1)$ or by subtracting the gap extension penalty from the value of position $E(i, j-1)$, in either case the alignment is extended by aligning residue b_j to a gap.

iii $F(i, j)$ was computed by subtracting the gap open penalty from either $M(i-1, j)$ or $E(i-1, j)$, or by subtracting the gap extension penalty from $F(i-1, j)$, in each case a_i is aligned to a gap.

The backtracking procedure described here finds just one alignment with the optimal score. If at any point two of the derivations are equal, an arbitrary choice is made between equal options. The backtracking algorithm is easily modified to recover more than one alignment.

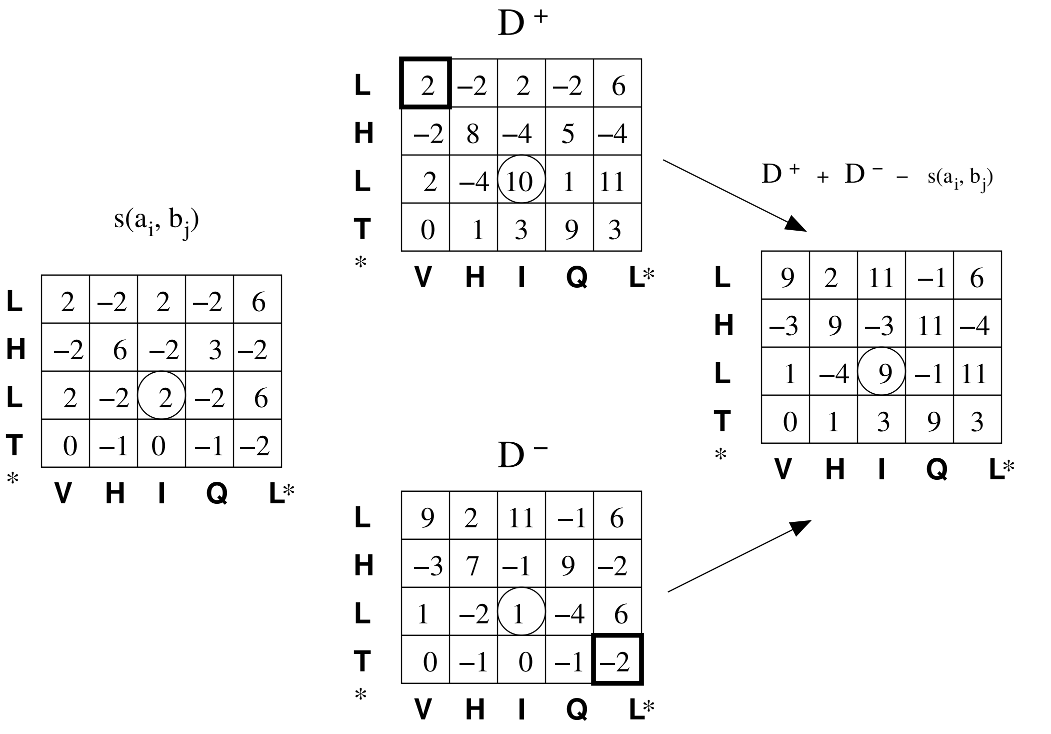
2.5 Suboptimal Alignments

Most standard sequence alignment methods generate only a single optimal alignment. For two proteins with a very close evolutionary relationship a high alignment score usually implies that their sequences have a high degree of relatedness. In this case it is often sufficient to generate just one optimal alignment to determine their homology. When the evolutionary relationship between two sequences is more distant, the optimal sequence alignment may change with small alternations of the parameters [65]. Furthermore, the dynamic programming algorithms used to derive the "optimal" alignment have an inherent ambiguity, that arises from the non uniqueness of optimal solutions and the particular scheme by which the search space is evaluated [18]. This uncertainty about sequence alignment methods has given rise to more sophisticated sequence comparison methods: One approach to improve the reliability of sequence alignments is the inclusion of near-optimal solutions. A near-optimal alignment is an alignment whose score lies within the neighborhood of the optimal score.
Saqi and Sternberg [55] developed a computer program using a variant of the Sellers algorithm [56, 54], which finds the minimum distance $\mathbf{D}(\mathbf{a}, \mathbf{b})$ of an alignment of two sequences. They generated the distance matrix $\mathbf{D}$, where

$\mathbf{D}(i, j)$ is the minimum distance of the alignment of $a_1 \ldots a_i$ and $b_1 \ldots b_j$. After obtaining an optimal alignment they modified those cells $(i.j)$ that correspond to a match or mismatch, by increasing their value by a small amount Δ. For distance alignments, increasing the value of a cell $(i.j)$ results in a lower probability that this cell is included in an optimal alignment. This procedure was repeated iteratively to obtain a set of suboptimal alignments. The stability of an alignment was given through the number of iterations through which an alignment persists.

Vingron and Argos [66] made use of suboptimal alignments to determine reliable regions in protein sequence alignments. They introduced the concept of 'stable regions' using the existence of a matched span in many reasonable alignments as the indicator for the credibility of an alignment region. Suboptimal alignments with a score close to the score of an optimal alignment are considered. The reliable regions of similarity between two sequences are those regions for which all optimal and suboptimal alignments share a regional path. Vingron and Argos used the method of forward-backward alignment to generate a set of suboptimal alignments. Using this method one can calculate the score of the best possible alignment through every residue pair of a comparison matrix. The algorithm to calculate the maximal score of an alignment path going through a point involves a modification of the classical Needleman-Wunsch algorithm [52]. It has been described first by Altschul and Erickson [3]. Zucker [70] applied it to RNA secondary structure prediction and Carillo and Lipman [9] used it to limit the search space in the simultaneous alignment of several sequences.

The core of the computation of forward-backward alignments is the calculation of two matrices, each of which is analogous to the matrix computed in the Needleman-Wunsch algorithm. The first matrix $\mathbf{D}^-$ stores in every cell (i, j) the score of an alignment that starts at the N-terminal ends of the sequences and ends with the residue pair (i, j). The second matrix $\mathbf{D}^+$ contains for every point the score of the optimal alignment from the C-terminal ends of the sequences to each point (i, j) (Figure 8). For every point (i, j), there



$s(a_i, b_j)$:

	V	H	I	Q	L*
L	2	−2	2	−2	6
H	−2	6	−2	3	−2
L	2	−2	2	−2	6
T	0	−1	0	−1	−2

D^+:

	V	H	I	Q	L*
L	2	−2	2	−2	6
H	−2	8	−4	5	−4
L	2	−4	10	1	11
T	0	1	3	9	3

D^-:

	V	H	I	Q	L*
L	9	2	11	−1	6
H	−3	7	−1	9	−2
L	1	−2	1	−4	6
T	0	−1	0	−1	−2

$D^+ + D^- - s(a_i, b_j)$:

	V	H	I	Q	L*
L	9	2	11	−1	6
H	−3	9	−3	11	−4
L	1	−4	9	−1	11
T	0	1	3	9	3

Figure 8: Principle of the algorithm for finding suboptimal points according to Vingron and Argos [66]. The computation is performed on the two example sequences **LHLT** and **VHIQL** using the Dayhoff PAM250 matrix and gap open and gap extension penalties of 5 and 1 respectively. The C-termini of the sequences are marked by an asterisk. Computation of $\mathbf{D}^-$ starts at the lower rightmost matrix position while $\mathbf{D}^+$ starts at the upper leftmost position (the starting positions of the matrices are marked by bold boxes). The encircled numbers provide an illustration of the calculation procedure for each of the matrix positions.

cannot be an overall alignment containing (i, j) that would be better than the sum of the partial alignment in $\mathbf{D}^-$ from (0,0) to (i,j) and of the partial alignment in $\mathbf{D}^+$ from (i,j) to (m,n) minus a correction involving the subtraction of the similarity score of the residue pair (i, j), as it has been counted twice. Therefore, adding equivalent positions of the matrices $\mathbf{D}^-$ and $\mathbf{D}^+$ yields for every point the maximal score that an alignment, going through this point, can possibly assume. Suboptimal alignments can be constructed

by backtracking toward the C-terminus in $\mathbf{D}^+$ and toward the N-terminus in $\mathbf{D}^-$. This method yields a small (hopefully representative) sample of suboptimal alignments.
Considering a sample of suboptimal alignments with a score close to the score of the optimal alignment Vingron and Argos determined reliably aligned regions: The reliable regions of similarity between two sequences are those regions for which suboptimal alignments, which score higher than the optimal score minus a tolerance value ϵ, share the same regional path, i.e. those regions for which a population of locally different suboptimal alignments does not exist. Sequence regions predicted to be reliably aligned at high ϵ-values are most likely to be correctly matched.

2.6 Partition Function

The construction of suboptimal alignments by the mathematical intersection of two alignment paths does not generate all suboptimal alignments but only a limited set of locally suboptimal alignments. By using the approach of foreward-backward alignment (see page 28) for retrieving suboptimal alignments, one can therefore easily miss suboptimal alignments which contribute substantially to the biological correct solution.
A method to avoid this problem is to use the partition function of all alignments between two sequences. Computation of the partition function is frequently applied in RNA secondary structure prediction [46, 32], its application for alignments was pioneered by Miyazawa [49].
The probability of an alignment $\mathcal{A}$ is the product of the probabilities of all substitutions, insertions and deletions constituting the alignment:

$$\text{Prob}(\mathcal{A}) = \prod_{i \in \text{sub}} p_{a_i^* b_i^*} \prod_{i \in \text{ins}} p_{-b_i^*} \prod_{i \in \text{del}} p_{a_i^* -}$$

where sub is an abbreviation for substitution, ins for insertion and del for deletion. The stars associated with the residue indices indicate that positions in the alignment, not positions along the sequences, are addressed.

The probability of an individual match or mismatch, p_{ab}, is the product of the probability $p(a,b)$ that the two residues have a common ancestor, as opposed to be of different origin, times the frequencies of these residues (see also page 15).

$$p_{ab} = p(a,b)\, q_a q_b$$

The probability of a gap is the product of a function $f(l_g)$ of the length of the gap, l_g, and the combined probabilities of the residues in the gap (see page 17).

$$p_{\mathbf{g}} = f(l_g) \prod_{i \in \mathbf{g}} q_{a_i}$$

The probability of the alignment is therefore given by:

$$\begin{aligned}
\text{Prob}(\mathcal{A}) &= \prod_{i \in \text{ sub}} p(a_i^*, b_i^*)\, q_{a_i^*} q_{b_i^*} \prod_{i \in \text{ ins}} f(l_{\text{ins}}) \prod_{i \in \text{ ins}} q_{b_i^*} \prod_{i \in \text{ del}} f(l_{\text{del}}) \prod_{i \in \text{ del}} q_{a_i^*} \\
&= \prod_{i \in \mathbf{a}} q_{a_i} \prod_{i \in \mathbf{b}} q_{b_i} \prod_{i \in \text{ indel}} f(l_{\text{indel}}) \prod_{i \in \text{ sub}} p(a_i^*, b_i^*) \\
&= \text{prob}(\mathbf{a})\ \text{prob}(\mathbf{b}) \prod_{i \in \text{ indel}} f(l_{\text{indel}}) \prod_{i \in \text{ sub}} p(a_i^*, b_i^*) \qquad (6)
\end{aligned}$$

where the product of the probabilities of all residues of sequence **a** that are matched in the alignment, $\prod_{i \in \text{ sub}} q_{a_i^*}$, and the probabilities off all residues of sequence **a** that are aligned to a gap (indel) $\prod_{i \in \text{indel}} q_{a_i^*}$ constitutes the probability of sequence **a**, $\text{prob}(\mathbf{a}) = \prod_{i \in \mathbf{a}} q_{a_i}$. The gap length dependent function $f(l_{\mathbf{g}})$ is the same for insertions and deletions. Therefore the products of $f(l_{\text{ins}})$ for all insertions and $f(l_{\text{del}})$ for all deletions can be combined to account for the product of $f(l_{\text{indel}})$ for all residues in the alignment that are aligned to a gap, $\prod_{i \in \text{ indel}} f(l_{\text{indel}}) = \prod_{l_g} f(l_g)$.

The score for the alignment is the sum of the scores for all gaps in the alignment, $\gamma(l_g) = \omega + k \log f(l_g)$ (see page 17) plus the sum of the scores for all substitutions, $s(a,b) = \omega + k \log p(a,b)$ (see page 15). Therefore we have:

$$\begin{aligned} S(\mathcal{A}) &= \sum_{l_g} \gamma(l_g) + \sum_{(i,j)\in\mathcal{A}} s(a_i, b_j) \\ &= \omega + k \left\{ \sum_{l_g} \log f(l_g) + \sum_{(i,j)\in\mathcal{A}} \log p(a_i, b_j) \right\} \end{aligned} \tag{7}$$

$$\begin{aligned} e^{S(\mathcal{A})} &= e^{\omega} \prod_{l_g} e^{k \, \log f(l_g)} \prod_{(i,j)\in\mathcal{A}} e^{k \, \log p(a_i,b_j)} \\ &= e^{\omega} \left\{ \prod_{l_g} f(l_g) \prod_{(i,j)\in\mathcal{A}} p(a_i, b_j) \right\}^k \\ &= e^{\omega} \left\{ \frac{\text{Prob}(\mathcal{A})}{\text{prob}(\mathbf{a}) \ \text{prob}(\mathbf{b})} \right\}^k \end{aligned}$$

The quantity $e^{\frac{S(\mathcal{A})}{k}}$ is therefore directly proportional to the probability of the alignment:

$$e^{S(\mathcal{A})} = \frac{e^{(\omega/k)}}{\text{prob}(\mathbf{a}) \ \text{prob}(\mathbf{b})} \text{Prob}(\mathcal{A}) \tag{8}$$

We subsume the constant factors in a constant c, that is:

$$c = \frac{e^{(\omega/k)}}{\text{prob}(\mathbf{a}) \ \text{prob}(\mathbf{b})}$$

The probability of the alignment is therefore given by:

$$\text{Prob}(\mathcal{A}) = \frac{e^{(\omega/k)}}{\text{prob}(\mathbf{a}) \ \text{prob} \ (\mathbf{b})} \, e^{\frac{S(\mathcal{A})}{k}} = c \, e^{\frac{S(\mathcal{A})}{k}}$$

The sum over the probabilities of all possible alignments between the two sequences **a** and **b** has to be 1:

$$\sum_{\mathcal{A}} \mathrm{Prob}(\mathcal{A}) = c \sum_{\mathcal{A}} e^{\frac{S(\mathcal{A})}{k}} = 1$$

In statistical mechanics it is customary to introduce the *partition function* Z as the sum of the Boltzmann factors for all possible states. In the present context we therefore set:

$$Z = \sum_{\mathcal{A}} e^{\frac{S(\mathcal{A})}{k}} \tag{9}$$

Thus we see that $c = \frac{1}{Z}$ and we recover the relationship between "energies" and probabilities of states that is familiar from statistical mechanics:

$$\mathrm{Prob}(\mathcal{A}) = \frac{1}{Z} e^{\frac{S(\mathcal{A})}{k}} \tag{10}$$

Miyazawa [49] used the substitution matrix of Dayhoff et al. [10] to derive the value of the scoring matrix dependent parameter k. Like any substitution matrix that makes statements about the probability of observing ab pairs in real alignments, Dayhoff's matrix is defined as a log-odds matrix [37, 5]. For computational convenience the entries of the score matrix were multiplied by 10 and rounded to the nearest integer. The score for a substitution, $s(a, b)$, is therefore:

$$s(a, b) = 10 \log_{10} \left(\frac{p_{ab}}{q_a q_b} \right)$$

Thus, the substitution score, $s(a, b)$, is equivalent to the log-likelihood ratio of this substitution times a constant factor, which, in case of Dayhoff's PAM matrix, is given by:

$$k = \frac{10}{\log_e 10} \approx 4.3429$$

Kschischo and Lässig [41] developed a theory of probabilistic alignments derived from a thermodynamic partition function. They called these probabilistic alignments *finite-temperature* alignments. A finite-temperature alignment is a probability distribution over all alignment paths $\mathcal{A}$. The probability of a given alignment $\mathcal{A}$ is shown in Equation (10). Kschischo and Lässig pointed out that a finite temperature alignment is controlled by three parameters: The average gap frequency and the length of the alignment path are determined by the scoring scheme used while the parameter k governs the relative weight of alignment paths with different scores.
We multiply the parameter k by a variable factor T. The inverse of this composed parameter is named β. In the case of Dayhoff's PAM matrix β is therefore:

$$\beta = \frac{1}{k\,T} = \frac{\log_e 10}{10\,T} \tag{11}$$

The relative weight of alignment paths with different scores can easily be modified by changing the value of T. At low T values optimal and near optimal alignment are preferentially generated. For $T = 1$ we obtain the "true" probability of the alignment. Increasing the temperature T decrease the weight given to the optimal alignments. In the limit of $T \to \infty$ random alignments with a uniform probability distribution are generated.
In a thermodynamic interpretation the score of the alignment, $S(\mathcal{A})$, is treated as if it were negative energy, the matrix-dependent constant k corresponds to Boltzmann's constant and the variable factor T is analogous to the temperature. We used a similarity scoring scheme to compare sequences, the scores for reliable alignments thus have positive values. Therefore the the algebraic sign of energies and entropy is opposite to the thermodynamic convention.
Including the variable T in our considerations the partition function at constant T is given by:

$$Z(T) = \sum_{\mathcal{A}} e^{\frac{S(\mathcal{A})}{k\,T}} = \sum_{\mathcal{A}} e^{\beta\, S(\mathcal{A})} \tag{12}$$

And the probability of the alignment is:

$$\mathrm{Prob}(\mathcal{A}) = \frac{1}{Z(T)} e^{\beta\, S(\mathcal{A})} \tag{13}$$

2.7 Match Probabilities

Using the partition function, the probability of each match (i, j) between the two sequences can be calculated. To derive the match probabilities, we define a class Ω of alignments that meet certain criteria. The probability to find an alignment that belongs to the class Ω is:

$$\mathrm{Prob}(\Omega) = \frac{1}{Z} \sum_{\mathcal{A} \in \Omega} e^{\beta S(\mathcal{A})} = \frac{Z(\Omega)}{Z}$$

The probability that i and j are matched is then given by:

$$p_{ij} = \mathrm{Prob}(\Omega_{i,j})$$

where $\Omega_{i,j}$ is the class of alignments in which a_i is matched to b_j.

$$\Omega_{i,j} = \{\mathcal{A} \mid (i, j) \in \mathcal{A}\}$$

For each $\mathcal{A} \in \Omega_{i,j}$ the score of the whole alignment is the sum of the score of the partial alignment $\mathcal{A}_{1,1}^{i,j}$ from position $(1, 1)$ to position (i, j) and the score of the partial alignment $\mathcal{A}_{i,j}^{m,n}$ from position (i, j) to position (m, n), minus the score of the match (i, j), $s(a_i, b_j)$.

$$S(\mathcal{A} \in \Omega_{i,j}) = S(\mathcal{A}_{1,1}^{i,j}) + S(\mathcal{A}_{i,j}^{m,n}) - s(a_i, b_j)$$

The partition function over all alignments containing the match (i, j) is therefore:

$$\begin{aligned}
Z(\Omega_{i,j}) &= \sum_{\mathcal{A}\in\Omega_{i,j}} e^{\beta\, S(\mathcal{A}_{1,1}^{i,j})+\beta\, S(\mathcal{A}_{i,j}^{m,n})-\beta\, s(a_i,b_j)} && (14)\\
&= \underbrace{\sum_{\mathcal{A}\in\mathfrak{A}_{1,1}^{i,j}} e^{\beta\, S(\mathcal{A}_{1,1}^{i,j})}}_{Z_{ij}^{\mathrm{M}}} \times \underbrace{\sum_{\mathcal{A}\in\mathfrak{A}_{i,j}^{m,n}} e^{\beta\, S(\mathcal{A}_{i,j}^{m,n})}}_{\widehat{Z}_{ij}^{\mathrm{M}}} \times e^{-\beta\, s(a_i,b_j)}\\
&= Z_{ij}^{\mathrm{M}}\ \widehat{Z}_{ij}^{\mathrm{M}}\ e^{-\beta\, s(a_i,b_j)} && (15)
\end{aligned}$$

where $Z_{ij}^{\mathrm{M}} = Z(\mathfrak{A}_{1,1}^{i,j})$ is the partition function of the set $\mathfrak{A}_{1,1}^{i,j}$ of all alignments of the partial sequences of $a_1\ldots a_i$ and $b_1\ldots b_j$ that end with a match of a_i and b_j. Analogously $\widehat{Z}_{ij}^{\mathrm{M}} = Z(\mathfrak{A}_{m,n}^{i,j})$ is the partition function of the set $\mathfrak{A}_{m,n}^{i,j}$ of all alignments of the partial sequences of $a_m\ldots a_i$ and $b_n\ldots b_j$ with a match of a_i and b_j.
The probability that residue a_i is aligned to residue b_j, $p(i,j)$, is therefore:

$$p(i,j) = \frac{Z_{ij}^{\mathrm{M}}\widehat{Z}_{ij}^{\mathrm{M}}}{Z}\ e^{-\beta\, s(a_i,b_j)} \qquad (16)$$

where the normalization factor Z is the partition function over all alignments between the two sequences. In the course of the calculation of the partition function the Z_{ij}^{M} values are calculated. To compute the $\widehat{Z}_{ij}^{\mathrm{M}}$ values, the partition function of all alignments is calculated in reverse order. We start the calculation of the partition function at the C-terminus, that is at positions a_m and b_n and proceed in the direction of the N-terminus (see page 37).

2.8 Calculation of the partition function

The partition function can easily be calculated using dynamic programming. Giegerich [18] pointed out that dynamic Programming Algorithms have an inherent ambiguity that arises from two independent sources. One source is the non-uniqueness of optimal solution, the other emerges from the particular recursion scheme by which the search space is evaluated. By using the partition function to generate suboptimal alignments, one extracts the information provided by the ambiguity of the search space to improve the results. The second source of ambiguity, which is caused by the recursion scheme used, has effects that antagonize an exact solution of the partition function. These effects are the generation of duplicate and non-canonical solutions. Duplicate solutions arise from the fact that the algorithm used may produce the same solution several times. This may lead to a combinatorial explosion of redundancy. Often, the search space exhibits additional redundancy in terms of solutions that are represented differently, but are equivalent from a more semantic point of view. Therefore canonization is important in evaluating statistical significance [42], and also in reducing redundancy among near-optimal solutions [18]. Examples of ambiguity in alignments arising from solutions that are represented differently but are semantically equivalent are given in Figure 9.

```
AAAC C – – TTAA        AAA– – C CTTAA        AAAC – – CTTAA
AAA– – G GTTAA         AAAG G– –TTAA         AAA– GG –TTAA
```

Figure 9: Examples of ambiguity in alignments after Giegerich [18]. The first and the second alignment are equivalent under most scoring schemes. The third alignment shows two deletions separated by an insertion and is considered a mal-formed alignment.

For dynamic programming algorithms designed to retrieve an optimal solution, it is not necessary to avoid the generation of redundant alignment. The results of the partition function, on the other hand, can be seriously distorted if redundant or non canonical alignments are not excluded. In our work we used a variation of the Needleman-Wunsch algorithm with affine gap penalties, that is designed to avoid redundancy. Figure 9 shows three alignments that are semantically equivalent but represented differently. The three alignments differ only in the relative order of adjacent gaps. The third alignment, which contains two deletions flanked by two insertions in the upper sequence is mal-formed and should not be generated at all. The first and the second alignment are equivalent, permutations of the location of adjacent gaps are

not counted as distinct, and one has to decide which one of this two equal representations should be used. The variation of the Needleman-Wunsch algorithm, that is used in our work, employs an asymmetric recursion scheme to avoid this kind of ambiguity (see Equation (5) and Figure 7): If two or more gaps are adjacent to each other, the gaps in the sequence indexed with i are always located in front of the gaps in the other sequence (the one indexed with j), therefore only one of these two possible representations of the alignment is generated.
The calculation of the partition function was done analogous to Miyazawa [49], using our variation of the Needleman-Wunsch algorithm: A partition function for the scoring scheme of equation (7) can be derived by changing the maximum operations in equation (5) to summations:

$$\begin{aligned} Z^M_{i,j} &= \left(Z^M_{i-1,j-1} + Z^E_{i-1,j-1} + Z^F_{i-1,j-1}\right)e^{\beta s(a_i,b_i)} \\ Z^E_{i,j} &= Z^M_{i,j-1}e^{\beta g_o} + Z^E_{i,j-1}e^{\beta g_{\text{ext}}} \\ Z^F_{i,j} &= \left(Z^M_{i-1,j} + Z^E_{i-1,j}\right)e^{\beta g_o} + Z^F_{i-1,j}e^{\beta g_{\text{ext}}} \end{aligned} \tag{17}$$

$$Z_{i,j} = Z^M_{i,j} + Z^E_{i,j} + Z^F_{i,j} \tag{18}$$

where $Z^M_{i,j}$ is the partition function over all alignments of the partial sequences from $a_1 \dots a_i$ and $b_1 \dots b_j$, in which a_i is aligned with b_j. $Z^E_{i,j}$ is the partition function over all alignments between these two sequence fragments in which residue b_j is aligned to a gap (all alignments where a gap is introduced in sequence **a**) and $Z^F_{i,j}$ contains the partition function over all alignments which end with a gap in sequence **b** (a_i is aligned to a gap). The partition function over all alignments up to position a_i and b_j is therefore $Z_{i,j} = Z^M_{i,j}+Z^E_{i,j}+Z^F_{i,j}$. The boundary conditions are: $Z^M_{0,0} = 1$, $Z^M_{i,0}$ and $Z^M_{0,j}$ are not defined. $Z^E_{0,0} = 1$, $Z^E_{0,1} = e^{\beta g_o}$ and $Z^E_{0,j>1} = e^{(\beta g_o+(j-1)g_{\text{ext}})}$. $Z^E_{i,0}$ is not defined. $Z^F_{0,0} = 1$, $Z^F_{1,0} = e^{\beta g_o}$ and $Z^F_{i>1,0} = e^{(\beta g_o+(i-1)g_{\text{ext}})}$. $Z^F_{0,j}$ is not defined.
For the calculation of the Z^{rev} values, the same recursion is used. We start the calculation with the index i set to m and the index j set to n and finish

when i and j are 0. The boundary conditions are: $Z^M_{m,n} = 1$, $Z^E_{m,n} = 1$, $Z^E_{0,n-1} = e^{\beta g_o}$ and $Z^E_{0,j<n-1} = e^{(\beta g_o + (n-j) g_{\text{ext}})}$. $Z^F_{m,n} = 1$, $Z^F_{m-1,0} = e^{\beta g_o}$ and $Z^F_{i<m-1,0} = e^{(\beta g_o + (m-i) g_{\text{ext}})}$.

2.9 Stochastic Backtracking

The results of the partition function can be used to generate optimal and suboptimal alignments by stochastic backtracking. Stochastic backtracking is a process that generates randomly selected alignments, with a distribution that corresponds to the probability of each individual alignment. The stochastic backtracking thus provides a fairly distributed ensemble of possible alignments.The probability of an alignment is given by:

$$\text{Prob}(\mathcal{A}) = \frac{e^{\beta S(\mathcal{A})}}{Z}$$

where $S(\mathcal{A})$ is the score of the alignment and β is a composite factor that contains the inverse of a matrix dependent constant times the inverse of the variable factor (see Equation (11)). In the formalism of the thermodynamic partition function $S(\mathcal{A})$ is analogous to a negative energy and β is the analog of $\frac{1}{kT}$.

For every position (i, j) of an alignment the probability for matching residues a_i and b_j, for introducing a gap in sequence **a** and for introducing a gap in sequence **b** can be calculated. The stochastic backtracking starts at final positions, (m, n), of the Z matrices. We start the stochastic backtracking by determining the probabilities of the three possible alignment states: The probability of matching a_m and b_n, $p(\text{match})$, is the partition function over all alignments between sequences **a** and **b** that end in a match, $Z^M_{m,n}$, divided by the partition function over all possible alignments between these two sequences, $Z_{m,n} = Z^M_{m,n} + Z^E_{m,n} + Z^F_{m,n}$:

$$p(\text{match}) = \frac{Z^M_{m,n}}{Z_{m,n}}$$

The probability, for introducing a gap in sequence $\mathbf{a}$, $p(\text{gap in }\mathbf{a})$, and the probability for introducing a gap in sequence $\mathbf{b}$, $p(\text{gap in }\mathbf{b})$ are calculated in the same way. The probability to insert a gap in sequence $\mathbf{a}$ is given by the partition function over all alignments that end with a gap in sequence $\mathbf{a}$, $Z^E_{m,n}$, divided by the partition function over all possible alignments. And the probability to introduce a gap in sequence $\mathbf{b}$ is calculated by dividing the partition function over all alignments that end with a gap in sequence $\mathbf{b}$, $Z^F_{m,n}$, by the partition function over all alignments.

$$p(\text{gap in }\mathbf{a}) = \frac{Z^E_{m,n}}{Z_{m,n}}$$

$$p(\text{gap in }\mathbf{b}) = \frac{Z^F_{m,n}}{Z_{m,n}}$$

To select one of the three possible states of the alignment, a random number between 0 and 1 is generated. Residue a_m is matched to residue b_n if the random number is $< p(\text{match})$, a gap is introduced in sequence $\mathbf{a}$, if the random number is $< (p(\text{match}) + p(\text{gap in }\mathbf{a}))$, else a gap is inserted in sequence $\mathbf{b}$. In the following steps of the stochastic backtracking, the probability of each state is dependent on the previous choice. If the residues were matched, the state probabilities are given by:

$$p(\text{match}) = \frac{Z^M_{i-1,j-1}\, e^{\beta s(a_i,b_j)}}{Z_{i,j}}$$

$$p(\text{gap in }\mathbf{a}) = \frac{Z^E_{i-1,j-1}\, e^{\beta s(a_i,b_j)}}{Z_{i,j}}$$

$$p(\text{gap in }\mathbf{b}) = \frac{Z^F_{i-1,j-1}\, e^{\beta s(a_i,b_j)}}{Z_{i,j}}$$

where $Z_{i,j} = (Z^M_{i-1,j-1} + Z^E_{i-1,j-1} + Z^F_{i-1,j-1})e^{\beta s(a_i,b_j)}$. If the previous state of the alignment was a gap in sequence $\mathbf{a}$, there are only two possibilities to extend the alignment, either return to the match state or to add another gap in sequence **a**. The probability to introduce a gap in sequence **b**, $p(\text{gap in } \mathbf{b})$ is zero, because the algorithm is designed to arrange gaps in order gaps in **a** $\Rightarrow$ gaps in **b**.

$$p(\text{match}) = \frac{Z^M_{i,j-1}\, e^{\beta g_o}}{Z^E_{i,j}}$$

$$p(\text{gap in } \mathbf{a}) = \frac{Z^E_{i,j-1}\, e^{\beta g_{\text{ext}}}}{Z^E_{i,j}}$$

where $Z^E_{i,j} = Z^M_{i,j-1} e^{\beta g_o} + Z^E_{i,j-1} e^{\beta g_{\text{ext}}}$ is the sum over all alignments up to position (i, j) that end with a gap in sequence **a**.
In the case that the previous state of the alignment was a gap in sequence **b**, three possible states to continue the alignment are available: return to the match state, switch to a gap in sequence **a**, or continue the alignment with another gap in sequence **b**.

$$p(\text{match}) = \frac{Z^M_{i-1,j}\, e^{\beta g_o}}{Z^F_{i,j}}$$

$$p(\text{gap in } \mathbf{a}) = \frac{Z^E_{i-1,j}\, e^{\beta g_o}}{Z^F_{i,j}}$$

$$p(\text{gap in } \mathbf{b}) = \frac{Z^F_{i,j-1}\, e^{\beta g_{\text{ext}}}}{Z^F_{i,j}}$$

where $Z^F_{i,j} = (Z^M_{i-1,j} + Z^E_{i-1,j}) e^{\beta g_o} + Z^F_{i-1,j} e^{\beta g_{\text{ext}}}$ is the sum of all possible alignments up to position (i, j) that end with a gap in sequence **b**.
At each step of the backtracking process the selection of the next alignment states is done stochastically. Repeated application of this procedure yields an equilibrium sample of alignments.

3 Program

3.1 Introduction

The variations of a dynamic programming algorithm for global pairwise alignments (2.4.2) and for the partition function (2.8) were used to implement a program in C. The program was named probA for probability alignments and provides several distinct features. probA can be used to calculate a global pairwise alignment, to determine the PAM distance (see page 12) and/or the pairwise identity (see below) between two sequences, for computation of the partition function, the match probabilities and for stochastic backtracking.

3.2 Global Pairwise Alignment

propA starts by calculating an initial alignment of the input sequences. The typ of the input polymer can be specified by the user, using the command line options **-DNA** or **-prot** to identify nucleic acid or protein sequences, respectively. If the type of the input polymer is not specified probA automatically calls the function `check_polymer`, which decides whether the input polymers are nucleic acid or protein sequences. Function `check_polymer` is taken from ClustalW [62, 31]. The decision is based on counting all occurrences of the characters $\{\mathbf{A}, \mathbf{U}, \mathbf{T}, \mathbf{G}, \mathbf{C}\}$ in the input polymers. If $\geq 85\%$ of all characters of the longer input sequence are $\{\mathbf{A}, \mathbf{U}, \mathbf{T}, \mathbf{G}, \mathbf{C}\}$ the polymer is treated as nucleic acid, else as protein.
This initial alignment is used to determine the PAM distance and/or the pairwise identity of the input sequences. The calculation of the PAM distance is done similar as in ClustalW [62, 31]. To determine the pairwise identity between two sequences the number of matches is divided by the sum of mismatches and matches (gaps are ignored). The observed distance is (1 - observed identity) and has values between between 0.0 (for identical sequences) and 1.0 (for totally different sequences). One problem of the estimation of distances arises from the fact that during divergence sequences

become saturated with mutations. Various positions along the sequences can be subjected to substitutions more than once. The calculated distances will therefore underestimate the actual divergence time. To correct for multiple substitutions we used the formula of Motoo Kimura [40]:

$$c = -\ln(1 - d - \frac{d^2}{5}) \tag{19}$$

where c is is corrected distance and d is the observed distance. This formula gives the mean number of estimated substitutions per site and, in contrast to the observed distance, can be greater than 1 (i.e. more than one substitution per site, on average). To express the corrected distance in PAM units, the corrected distance is multiplied by 100. PAM units measure the mean number of substitutions per 100 residues. Dayhoff *et al.* derived a table relating observed distances to predicted PAM distances. Kimura's formula is just a "curve fitting" approximation to this table. It is very accurate in the range $0.0 > d > 0.75$ but becomes increasingly inaccurate at $d > 0.75$ and fails completely at around $d = 0.85$. Observed distances > 0.75 are determined using Dayhoff's table. The determined PAM distance or the percentage identity is used to select a scoring matrix appropriate for the evolutionary distance of the input sequences.

3.3 Scoring the Alignment

Each alignment of two sequences is assigned a score. The score is based on the number of matches and mismatches for aligned elements and on the number of gaps used. Maximization of this score is then used to select the optimal alignment. However, it is well known that the optimal alignment of a given pair of sequences can depend strongly on the scoring parameters used. With increasing evolutionary distance maximum similarity alignments tend to become sensitive to the choice of the scoring parameters and therefore less reliable.

3.3.1 Substitution Matrices for Protein Sequences

Methods for alignments of protein sequences typically measure similarity by using a substitution matrix with scores for all possible exchanges of one amino acid with any other (see 2.2.2). The sensitivity of most protein sequence alignments depends strongly on the quality of the scoring matrix and the penalties used for insertions and deletions. probA therefore provides three different sets of scoring matrices, the Dayhoff's PAM matrix series, the `BLOSUM` series and the Gonnet series of scoring matrices. Each log-odds matrix in a series is defined for a specific evolutionary distance measured in PAM units in the case of PAM and Gonnet matrices and as percentage identity in the case of `BLOSUM` matrices.

Dayhoff *et al.* collected mutational data from 72 sequence families consisting of more than 1300 related protein sequences (distance < 15 PAM units, this is equivalent to a percentage identity $> 85\%$) to calculate the probability of amino acid exchanges. Alignments between that closely related sequence pairs are indisputable. Hence the elements of the mutation data matrix can be tabulated directly without needing to worry about 'successive accepted mutations at one site' [10].

To convert this data from closely related sequence pairs into a matrix that describes mutations between protein pairs 250 PAM units divergent, a process of matrix powering is used. A matrix for protein pairs m PAM units distant is converted to one applying to proteins separated by n PAM units by raising the matrix to the $\frac{n}{m}$th power.

In the Dayhoff model the scoring matrices were derived from a small set of proteins that were very similar in sequence, furthermore substitution matrices for distantly related proteins were derived by successive matrix powering from a scoring matrix for closely related protein sequences. Several authors pointed out that Dayhoff's substitution matrices are therefore unsuitable for alignments of more distantly related sequences and provided methods to calculate substitution matrices more suitable for this purpose [29, 21, 8].

Henikoff and Henikoff [29] derived substitution matrices from more than 2000

blocks of aligned sequence segments characterizing more than 500 families of related protein sequences. Local alignments were represented as ungapped blocks with each row a different protein segment and each column an aligned residue position. To reduce multiple contributions to amino acid pair frequencies from the most closely related members of a family, sequences were clustered within blocks and each cluster was weighted as a single sequence in counting pairs [30]. This was done by specifying a clustering percentage in which sequence segments identical for at least that percentage of amino acids were grouped together. For example, if the percentage is set at 80% sequence segments identical at $\geq$ 80% of their aligned positions are clustered and their contributions are averaged in calculating pair frequencies. A consequence of clustering is that the contribution of closely related segments to the frequency table is reduced. Varying the clustering percentage in this way leads to a family of matrices. The matrix derived from a database of blocks in which sequence segments that are identical at $\geq$ 80% of aligned residues are clustered is referred to as `BLOSUM` 80, and so forth.
Gonnet *et al.* [21] determined empirical probabilities of mutations between amino acid by matching an entire protein sequence database with 8,344.353 amino acid residues using a Needleman Wunsch algorithm. Each sequence was compared against the entire database, such that 1.7x10^6 subsequence matches resulted for significant alignments. The key to matching an entire database in a reasonable time was the reorganization of the sequence data by indexing on a patricia tree [22] preceding the application of the Needleman Wunsch algorithm. In an indexed database, pairs of identical sequences are found instantaneously because they lie together on the tree. Similar sequences lie near each other on the tree. After exhaustive matching of the database and several rounds of refinement the remaining matches, each optimally aligned, were used to calculate new mutation data matrices and a model for scoring gaps.
For each of the matrix series introduced above a set of scoring matrices for different distances is provided by probA (see Table 1). probA provides the

Table 1: Scoring Matrices in probA

matrix series	PAM distance [a] / percentage identity [b]						
Gonnet series	40	80	120	160	250	300	350 [a]
PAM series	20	60	120	350 [a]			
BLOSUM series	80	62	50	30 [b]			

[a] PAM distance is used for the PAM and the Gonnet series

[b] percentage identity is used for the BLOSUM series

options to either select one of the matrix series or to pick out a single substitution matrix. To select one of the matrix series use the command line option **-score_matrix** [*name*] with the first three letters specifying the matrix series of choice as the argument (e.g. **-score_matrix** *gon*; specifies the Gonnet matrix series). The default matrix series is the Gonnet series. In case a matrix series was selected probA computes an initial alignment using the substitution matrix gonnet init. [21]. The matrix gonnet init. was compiled for protein pairs separated by a PAM distance between 6.4 and 100 and extrapolated by exponential fitting to a PAM distance of 116.5. It was selected for the initial alignment because it performed best in a comparison of different amino acid exchange matrices done by Vogt *et al.* [68]. The initial alignment is then used to estimate the PAM distance (or the percentage identity) of the input sequences. The estimated PAM distance (or percentage identity) is then used to select the scoring matrix for this evolutionary distance. To select one specific matrix directly, use the command line options **-score_matrix** [*name*] to specify the matrix series and **-pam** [*value*] to particularize the PAM distance (or the percentage identity)(e.g. the combination **-score_matrix** *gon* **-pam** *80*; specifies the Gonnet 80 matrix).

3.3.2 Gap penalties for Protein Scoring Matrices

The sensitivity of protein sequence alignments depends not only on the quality of the scoring matrix used, but is also strongly dependent on gap penalties. Gap penalties have to be optimized for the evolutionary distance of each matrix to achieve maximal performance. Vogt *et al.* [68] used a set of amino acid sequences matched by superposition of known protein tertiary topologies to test the alignment accuracy of different matrix-penalty combinations. Vogt *et al.* observed that the behavior of a Needleman Wunsch algorithm changed drastically if the utilized scoring matrices contains only positive values or positive and negative values. The effect of a changed matrix offset can be neutralized by changing the gap extension penalty if terminal gaps are penalized similar to gaps in the interior of the alignment. If terminal gaps are not penalized the performance of a Needleman Wunsch algorithm depends on the matrix offset. In our program the user can specify if terminal gaps are penalized similar to gaps in the interior of the alignment or not (see 3.3.3). We therefore provide substitution matrix variants with only positive values as well as the original matrices with positive and negative values. Matrices containing positive and negative scores were made positive by subtracting the smallest value in the matrix from all elements. Table 2 lists the gap penalties used for all available scoring matrices.

3.3.3 Terminal Gaps

In most dynamic programming applications, gap opening and gap extension penalties are applied equally at every position in the sequence, regardless of the location of the gap, except for terminal gaps which are usually allowed at no cost. Terminal gaps with no or low cost are particularly useful if two sequences of very different lengths are aligned. In this case the usage of terminal gaps with a neural or low score prevents that the alignment starts or ends with one or only a few aligned residues separated from the rest of the aligned residues by one extended gap. probA scores terminal gaps different from gaps in the interior of the alignment. By default the score for terminal

Table 2: Substitution matrices and gap penalties

score matrix		+ \ − matrix		all + matrix	
		open	ext.	open	ext.
Gonnet [a]	init.	14.00 [b]	0.20 [b]	6.00 [b]	0.80 [b]
Gonnet [c]	40	25.730 [d]	1.396 [d]		
	80	23.492 [d]	1.396 [d]		
	120	22.183 [d]	1.396 [d]		
	160	21.255 [d]	1.396 [d]		
	250	19.814 [d]	1.396 [d]		
	300	19.225 [d]	1.396 [d]		
	350	18.727 [d]	1.396 [d]		
PAM [c]	20	7.00 [b]	1.00 [b]	12.50 [b]	3.25 [b]
	60	7.00 [b]	1.00 [b]	14.00 [b]	1.50 [b]
	120	6.00 [b]	1.40 [b]	12.50 [b]	1.00 [b]
	350	9.00 [b]	2.40 [b]	12.50 [b]	0.40 [b]
`BLOSUM` [c]	80	7.00 [b]	1.50 [b]	15.50 [b]	0.04 [b]
	62	7.50 [b]	0.90 [b]	10.00 [b]	0.60 [b]
	50	9.50 [b]	1.20 [b]	9.50 [b]	0.60 [b]
	30	10.00 [b]	1.50 [b]	9.00 [b]	1.00 [b]

[a]Matrix Gonnet int. was published by Gonnet *et al.* [21]
[b]values for the gap penalties were taken from Vogt *et al.* [68]
[c]All other matrices were copied from ClustalW [62, 31]
[d]Gap penalties were calculated using Darwin [20]. Darwin is a partially interpreted language tailored for bioinformatics research. For the Gonnet matrix series no gap penalties for all + matrices are available.

gaps is zero. To change the cost of terminal gaps set the argument for the command line option **-endgaps** [*value*] to the selected value. If you want to score terminal gaps similar to gaps in the interior of the alignment use the command line option **-noEg**.

3.3.4 Scoring Nucleic Acid Alignments

To score nucleic acid alignments an unitary matrix is applied, scoring 1 for a match and zero for a mismatch. A gap open penalty of -3 and a gap extension penalty of -1 is used.

3.4 Match Probabilities

probA calculates the partition function over all alignments of the two input sequences. The partition function is then used to calculate the match probability of each match i, j between the two sequences (see 2.6, Equation 16). The match probabilities are depicted as a dot plot. Figure 1 shows a dot plot of the alignment between cytochrome c from skipjack tuna, *Euthynnus pelamis*, (CCBN[1]) and *Rhodospirillum rubrum* cytochrome c_2 (CCQF2R[1]). In the dot plot regions of high sequence similarity are illustrated by large black dots along or parallel to the diagonal of the dot plot. The size of each dot is governed by the match probability of the match it represents. In regions with lower sequence similarity many possibilities to align the input sequences exists. In this case all possible matches are depicted as small dots, which decrease in size with an increasing number of possibilities to align the sequence fragments in this region.
In our work we introduced a variable factor T governing the relative weight of alignment paths with different scores (see Equation 11, page 33). The value of T can be controlled using the command line option **-T** [*value*]. The default value for T is 1. At $T \rightarrow 0$ optimal are generated preferentially. Increasing the value of T decreases the weight given to the optimal alignment.

[1]Codes are taken from the PIR protein sequence database [6].

In the limit of $T \rightarrow \infty$ random alignments with a uniform probability distribution are generated. The effect of the variation of T on match probabilities is depicted in Figures 10 - 15, page 50. Figures 10 - 15 show the alignment of cytochrome c from skipjack tuna (CCBN[1]) to *Rhodospirillum rubrum* cytochrome c_2 (CCQF2R[1]). For the alignment scoring matrix Gonnet 120 was used. The value of T is given as caption of the figures. At low T values reliable matches have the highest probabilities. Increasing the value of T results in an increase of the probabilities of less reliable matches, whereas the weights given to very probable matches decrease. In the limit of $T \rightarrow \infty$ match weights are equally distributed.

3.5 Stochastic Backtracking

The results of the partition function are used for stochastic backtracking. The stochastic backtracking calculates a fairly distributed ensemble of possible alignments (see 2.9, page 38). By default the stochastic backtracking returns one stochastic alignment. To increase the number of alignments returned use the command line option **-N** [*number*]. probA returns each stochastic alignment encoded as a string of digits to STDOUT. The representation of alignments used by probA is different from the commonly used alignment representation. Generally an alignment is represented as two strings of letters written above each other. Similar or evolutionary related letters are written in one column, this is refered to as a match or mismatch, respectively. If letters of one string have been deleted in the other or if extra letters were inserted in one of the strings a gap symbol (-) is used to indicate the missing letters. An alignment is therefore explicitly described if the order of the two sequences and, for every position of the alignments, the state of this position is known. The states, which can be assigned to individual positions in an alignment, are straightforward: let 1 code for a match, 2 for a mismatch, 3 for a missing letter (gap) in the lower string and 4 for a gap in the upper string. An example of the output of probA is given in Figure 16.

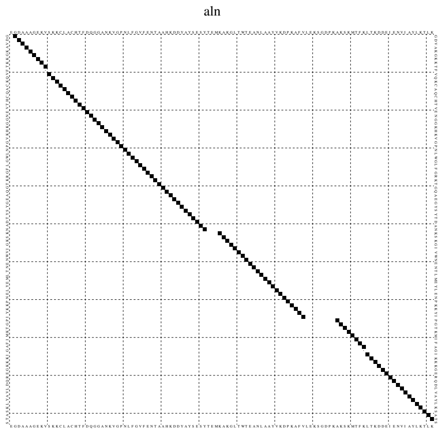


Figure 10: $T = 0.008$

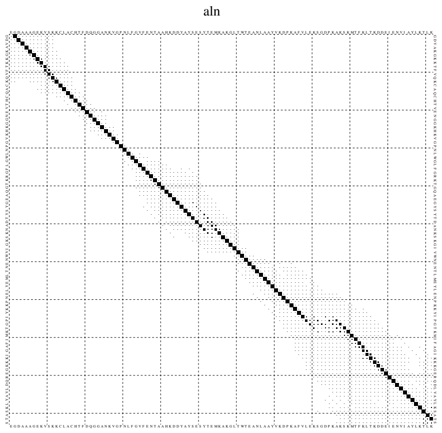


Figure 11: $T = 0.3$

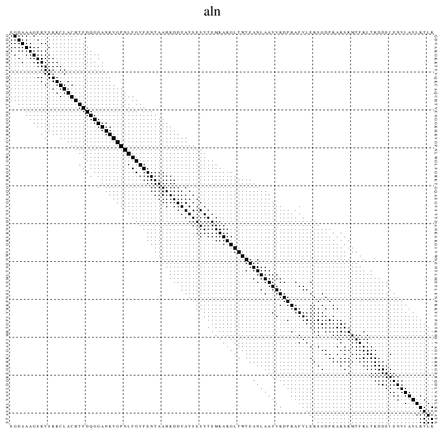


Figure 12: $T = 0.5$

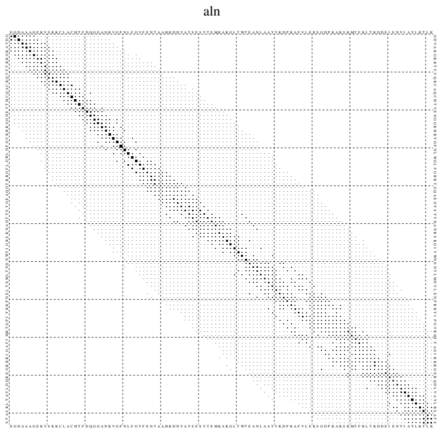


Figure 13: $T = 0.7$

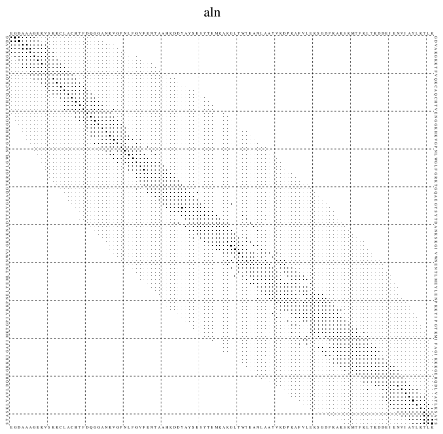


Figure 14: $T = 1$

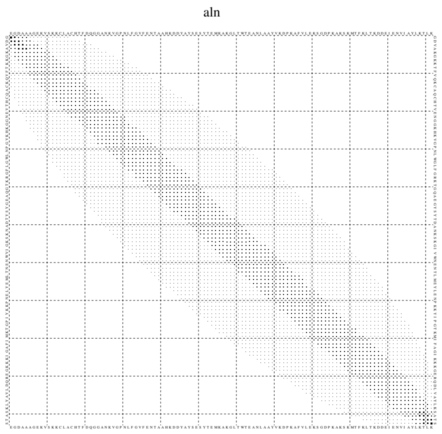


Figure 15: $T = 10$

```
#optimal alignment
#
#long           CPSGCTNF-KCA
#short          CPTG--NYKKCA
#
#               112133124111
#
#parameters
#T = 1.00
#scoring matrix: gonnet_40;
#gap penalties open:-25.73, extend:-1.396000;
#endgap penalty: 0.00
#
#score of the optimal alignment =  38.04
#entropy of the stochastic ensemble =  -36.46
#
#number of stochastic alignments = 1
#upper sequence
#long CPSGCTNFKCA
#lower sequence
#short CPTGNYKKCA
#
#S(A)     Prob(A)
38.04     1.80e+11   112133121411
Exit 1
```

Figure 16: Example of the output of probA. The output begins with the optimal alignment represented in the common way and as a string of digits: 1 match; 2 mismatch; 3 gap in the lower sequence; 4 gap in the upper sequence. Next the parameters used for the alignment are shown, follwowed by the score of the optimal alignment and the entropy of the stochastic ensemble. After the number of the stochastic alignments the order of the sequences in the stochastic alignments, which are represented as a strings of digits, is given. For each stochastic alignment, the first value is the score of the alignment, $S(\mathcal{A})$, the second the probability of this alignemnt, Prob$(\mathcal{A})$.

4 Results

4.1 Entropy of Ensembles of Stochastic Alignments

Stochastic backtracking provides an ensemble of stochastic alignments distributed according to the probability of each alignment. In order to quantify the diversity of alignments within the stochastic ensemble the entropy of the ensemble was calculated. The entropy of a stochastic ensemble corresponds to the difference between the score of the optimal alignment $S(\mathcal{A}_{\text{opt}})$ and the product of $kT\log Z$:

$$\Delta S^{\text{ensemble}} = S(\mathcal{A}_{\text{opt}}) - kT\ln Z$$

In our work we used a similarity scoring scheme to compare sequences. Recall that good alignment scores are therefore positive, i.e. scores correspond to negative energies. Thus the entropy of an ensemble of stochastic alignments has always a negative value.

To compare the entropies of stochastic alignment ensembles random sequences of different lengths were generated and the mean entropy for each length was calculated. This was done for protein as well as nucleic acid sequences. The entropy of a stochastic ensemble is dependent on the length of the compared sequences as well as on their % identity. For each type of polymer we calculated the mean entropies for stochastic ensembles of two identical sequences and of two randomly generated sequences. The mean % identity for random nucleic acid sequences is 41.1%, random protein sequences have a mean % identity of 13.3%. The results of this calculations are depicted in Figure 17, which shows that identical sequences have low entropy values. The optimal alignment dominates the stochastic ensemble, only a small number of suboptimal alignments is generated. Sequences with lower % identity values have noticeably higher entropy values. In this case the optimal alignment accounts for only a small fraction of the stochastic ensemble, the large majority of alignments are different suboptimal alignments.

To test whether the results of alignments of random sequences can be transferred to biological sequences, the entropy of pairs of functionally similar biological sequences of different length was calculated. The red downwards pointing triangles in Figure 17 denote the entropy of stochastic ensembles of different alignments of tRNA sequences. The % identity of this sequences varies between between 68% and 43%. The entropy values fluctuate around the curve calculated from the alignments of random sequences. The deviations from the curve of random sequences are caused by the different % identities of the tRNA sequences and also by the length differences between the tRNA sequences. The random sequences were all of the same length, differences in sequence length increases the entropy of the stochastic ensemble. The results for alignments of 5S RNA are indicated by red squares. Most of the alignments have a % identity between 40% to 50% and their entropies oszillate around the random sequence curve. The outlier value with an entropy of -3.9 (in the plot the sign of the entropy values was reverted) correspond to an alignment of sequences with 60% identity. The red stars point at the alignments of RNase P genes. The mean % identity of this sequences is 46%, the values are within the range of the values for random sequences. The same is true for the entropies of the partial TFIIA,E,D genes which are indicated by red upwards pointing triangles. The 16S rRNAs, which are referred to by red circles, have high identities (between 60% to 90%), this is reflected in low entropy values for these stochastic ensembles. The data from biological sequences are in good agreement with the results from stochastic ensembles of random sequences.

4.2 Generation of a 3D structure alignment

As an application of the algorithm the alignment between leghaemoglobin from yellow lupin [27], *Lupinus luteus*, (1GDJ[2]) and chain A of human deoxyhaemoglobin [13] (2HHB_A[2]) was analysed. For this analysis 1 million

[2]protein codes are those in Protein Data Bank (PDB), `http://www.rcsb.org/pdb/`

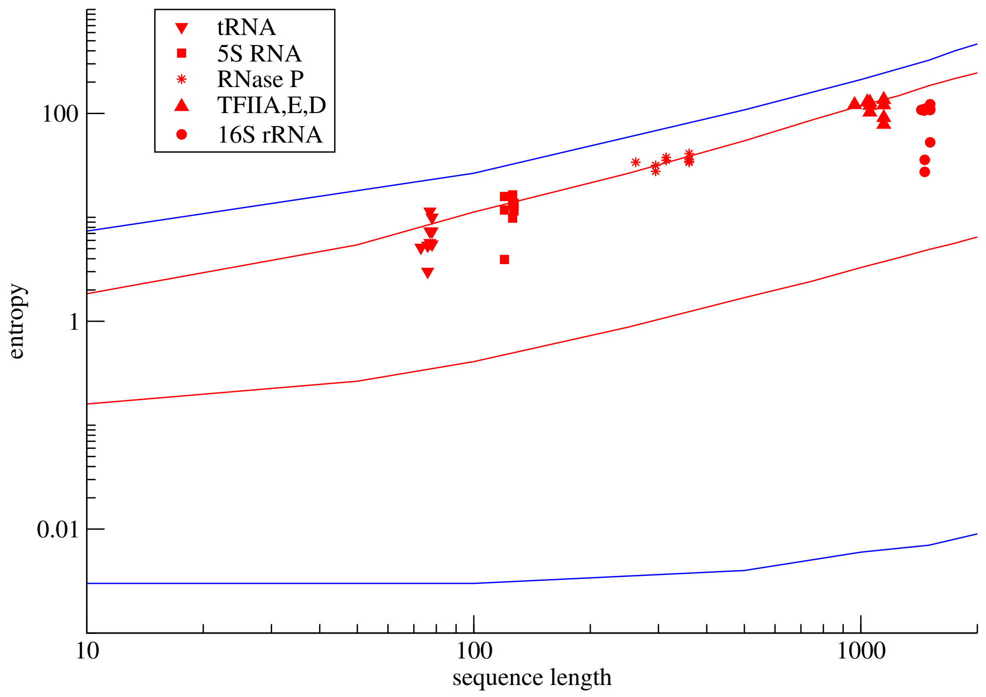


Figure 17: Entropy of an ensemble of stochastic alignments. The blue lines correspond to the mean entropies of protein sequences. The upper blue curve shows the entropies for alignments of two random protein sequences. The lower blue curve for alignments of two identical sequences. The red curves are for nucleic acids. Again the upper curve shows the values for two random sequences, the lower for identical sequences. The red symbols indicate entropies for alignments of different biological sequences. As stated in the text entropy values for similarity alignments have negative score. In order to obtain a graph that corresponds to thermodynamic conventions the sign of the entropy values was reversed.

stochastic alignments between 1GDJ and 2HHB_A were generated. To determine the reliability of the alignments, they were compared to an alignment based on the three-dimensional (3D) structures of these molecules. The global 3D alignment of two proteins has been characterized as NP hard [43]. Methods to find 3D alignments use approximations to make the problem computationally tractable. There is no exact solution to the protein structure alignment problem, but only the best solution for the heuristics used in the calculation. Therefore different heuristics were applied for the generation of the 3D structure alignment to reduce the influence of the various heuristic methods. The underlying assumption was that a region that is identically aligned by various methods is, in fact, reliably aligned.
1GDJ and 2HHB_A are dissimilar in sequence, the pairwise identity is 14 %, but rather similar in structure (see Figure 18). The quality of the crystal structures of 1GDJ and 2HHB_A were examined using the WHAT_CHECK program [34], then three-dimensional (3D) structure alignments of the two proteins were calculated.
For the calculation of pairwise 3D structure alignments of the two proteins we employed different Web-accessible structure alignment programs to extract reliably aligned regions. The different structure alignment programs apply distinct methods to align 3D structures. The subsequent paragraphs contain a short description of the structure alignment methods used, in order to give an overview of employed heuristics.
The Combinatorial Extension (CE) [57] method uses similarity in local geometry of C_α positions to generate structure alignments. Global features, such as overall topology, are not employed. Structure alignments utilizing CE are available via the Web at `http://cl.sdsc.edu/ce.html`. The CE algorithm calculates protein structure alignments by incremental combinatorial extension (CE) of the optimal path defined by aligned fragment pairs (AFPs). Each AFP confers geometric similarity between two fragments of C_α positions. Combinations of AFPs that represent possible continuous alignment paths are selectively extended or discarded leading to a single optimal

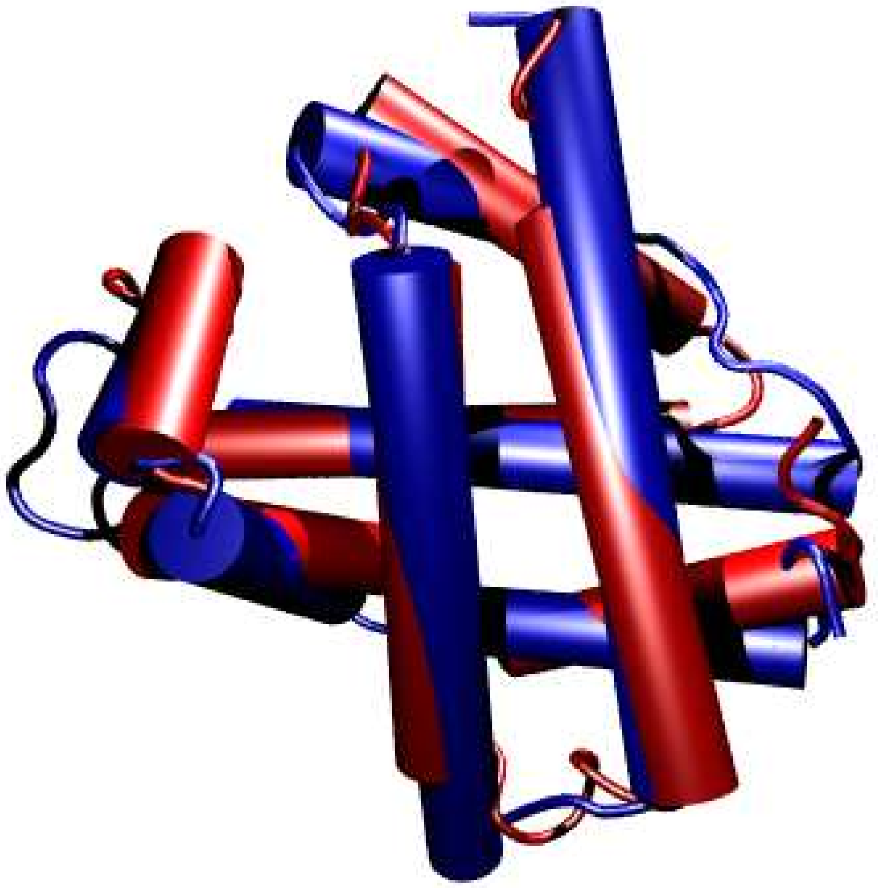


Figure 18: 3D structure superposition of 1GDJ and 2HHB_A generated by vmd [35]. 1GDJ is colored blue, 2HHB_A is colored red.

alignment.
TOP and COMPARER both utilize topological features for the computation of 3D structure alignments: TOP [45, 44] is a protein topological comparison program which detects structure similarities between two proteins. It superimposes two protein structures automatically without any previous knowledge of sequence alignment. TOP identifies the residues that share similar positions of both main-chain and side-chain atoms. Two protein structures can be compared via the Web at `http://bioinfo1.mbfys.lu.se/TOP/webtop.html`.
COMPARER [53] defines general topological equivalence in protein structures by a procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. For protein structure superposition the DiCE structure alignment program (unpublished) is used. COMPARER on-line from Robert Steward is available on the Web at `http://www-cryst.bioc.cam.ac.uk/~robert/cpgs/COMPARER`
SARF2 and MATRAS employ secondary structure information to generate structure alignments: SARF2 [1] detects 3D similarities of the backbone fragments considering only the protein's secondary structure elements (SSEs), no topological restrictions are used. The SARF2 algorithm consists of four steps: In the first step C_α-traces are used to assign secondary structure elements (SSEs). The fragment of the C_α-trace is assigned a particular secondary structure if it superimposes with a small enough rmsd to one of the prototypes of the α-helix and the β-strand along the protein backbone [64]. In the next step distance and angle constrains are used to filter out superimposable pairs of SSEs. Then the largest ensemble of the compatible pairs of SSEs is searched. The last step comprises extension and refinement of the matches by an iterative procedure. Structure alignments using SARF2 are available on the Web at `http://www-lmmb.ncifcrf.gov/~nicka/run2.html`.
MATRAS by Kawabata and Nishikawa [39] proposes a novel theory to evaluate protein structure similarity, which is based on the Markov transition model of evolution. The similarity score between structures i and j is defined as $\log(P(j \to i)/P(i))$, where $P(j \to i)$ is the probability that structure

j changes to structure i during the evolutionary process, and $P(i)$ is the probability that structure i appears by chance. MATRAS was developed using these scores. It employs a hierarchical alignment algorithm, in which a rough alignment is first obtained by SSEs, and is then improved with more detailed functions. Pairwise 3D alignment using MATRAS is available on the Web at `http://bongo.lab.nig.ac.jp/~takawaba/Matras_pair.html`.

The 3D structure alignments computed by the different on-line programs displayed 3 regions which were aligned without ambiguity. A region was accepted as reliably aligned if at least four of the five structure alignment programs used, identified this region as aligned and non of the programs disagreed with the alignment of this region. The reliably aligned regions are position 3 to 44 in 1GDJ with position 2 to 43 in 2HHB_A, position 90 to 98 in 1GDJ with position 80 to 88 in 2HHB_A and position 101 to 150 in 1GDJ with position 92 to 140 in 2HHB_A. Figure 19 shows the different structure alignments and the reliably aligned regions extracted using the procedure described earlier. In the structure alignments regions that are non-ambiguously aligned are colored yellow. In the consensus reliably aligned region are indicated in red. About 65% of the position of the consensus were reliably aligned. In the regions that are not colored in Figure 19 no consistent alignment exists, these regions were not considered. The reliably aligned regions in the consensus of Figure 19 were used as a standard measure of reliability in the examination of an ensemble of stochastic alignments.

```
CE          ------QAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSEVPQNN-PELQAHAGKVFKLVYEAAIQLE
            ------DKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYF--PHFDLSHGSAQVKGHGKKVADALTNAVAHV----
TOP 6.7     --LTE-QAALVKSSWEEFN-----HTHRFFILVLE-APAAK----------------------HAGK------------
            --LSP-DKTNVKAAWGKVG-----YGAEALERMFL-FPTTK----------------------KVAD------------
SARF2       --LTESQAALVKSSWEEFNANI-KHTHRFFILVLEIAPAAKDLF----KGTSEVP--NPELQAHAGKVFKLVYE------
            --LSPADKTNVKAAWGKVGAHA-EYGAEALERMFLSFPTTKTYF----FDLSHGS--K-GHGKKVADALTNAVA------
MATRAS      -ALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE-VPQNNPELQAHAGKVFKLVYEAAIQLE
            -VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFD-----LSHGSAQVKGHGKKVADALTNAVAHV-
COMPARER    GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE-VPQNNPELQAHAGKVFKLVYEAAIQLE
            -VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFD-----LSHGSAQVKGHGKKVADALTNAVAHVD

consensus   GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSKLKGTSEVPQNN-PELQAHAGKVFKLVYEAAIQLE
            -VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYF--PHFDLSHGSAQVKGHGKKVADALTNAVAHV----
```

```
CE          VTGVVVTDATLKNLGSVHV-SKGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMD---
            ----DDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---
TOP 6.7     ------------NLG--------VADAHFPVVKEAILKTIKEVVG-KWSEELNSAWTIAYDELAIVIKKEM----
            ------------ALS--------VDPVNFKLLSHCLLVTLAAHLP-EFTPAVHASLDKFLASVSTVLTSKY----
SARF2       ----------LKNLGSVHV-SKGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMD---
            ----------LSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---
MATRAS      VTGVVVTDATLKNLGSVHVS-KGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMD---
            --DDMPNA--LSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---
COMPARER    VTGVVVTDATLKNLGSVHVSK-GVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMDDAA
            D-----MPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---

consensus   VTGVVVTDATLKNLGSVHV-SKGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMDDAA
            ----DDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---
```

Figure 19: Reliably aligned region were extracted from 5 pairwise structure alignments computed with different on-line 3D structure alignment programs. For each pairwise alignment the upper sequence is 1GDJ, the lower 2HHB_A. The structure alignment program used is indicated in the first column. The second column displays the structure alignments, reliably aligned regions are indicated in yellow. The lowest pairwise alignment of each block indicates the consensus alignment, reliably aligned residues are colored red. The alignment in the regions were no consistent alignment between the different structure alignment programs exists is like in the structure alignment generated with CE.

4.3 Evaluation of the stochastic alignment space

To evaluate the stochastic alignment space an ensemble of 1 million stochastic alignments between sequences 1GDJ and 2HHB_A was generated. For the computation of the stochastic alignments scoring matrix Gonnet 300 was used. The variable T (see equation 11) was set to 1. The penalty for opening a gap was 19.225, the gap extension penalty 1.396 (see table 2, page 47). Terminal gaps were treated differently from gaps inside the alignment. For terminal gaps a score of zero was applied.

To get an overview of the stochastic alignment space the frequency of each alignment was determined. Of the 1 million stochastic alignments generated 2.48% have a score that lies within $1kT$ of the score of the optimal alignment. The frequency of different stochastic alignments is shown in Figure 20. The number of identical alignments generated by stochastic backtracking increases noticeably with increasing score of the alignment. However relatively high numbers (more than 50) of identical alignments are only generated if the score of the alignment is in the range of $1kT$ from the optimal alignment. The frequency of identical alignment decreases exponentially with decreasing score. Around a score of 40 the maximal number of identical alignments generated drops to about 20. Below a score of 30 each alignment is only generated once or twice and below a score of 0 each alignment is generate only once. The insert in Figure 20 shows that many different alignments with the same score are found. This alignments have different frequencies as well as varying % identities to the structure alignment.

The match probability distribution that governs the selection of the stochastic alignments was analysed to determine the correlation between the probability of a match and the appearance of this match in the structure alignment. The dot plot was generated using the alignment conditions described above. Figure 21 shows the dot plot of the match probabilities for the comparison between lupin leghaemoglobin, 1GDJ, and chain A of human deoxyhaemoglobin, 2HHB_A. The dot plot displays three region of high match probabilities. The region of high match probabilities at the N-terminus of

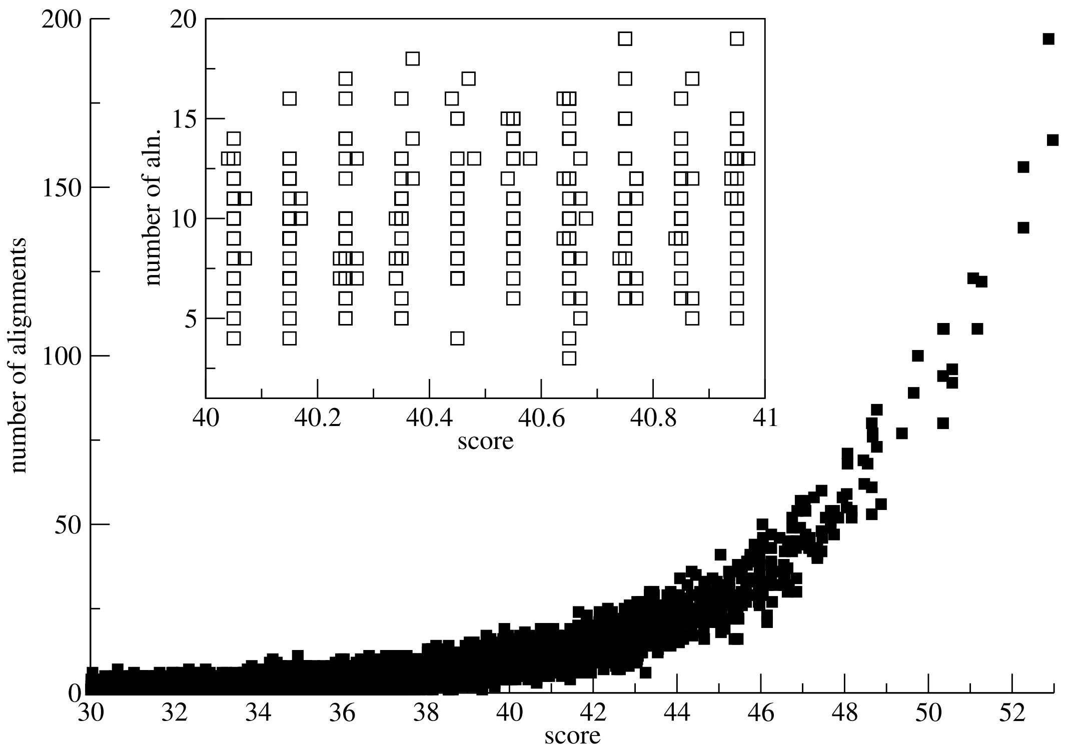


Figure 20: Frequencies of stochastic alignments with scores in the range of $30 < \text{score} \leq S_{\text{opt}}$ are shown. Below a score of 30 each alignment is generated maximal 3 times, most alignments are only found once or twice. Alignments with scores < 0 were generally generated only once. In the insert alignments with scores between 40 and 41 are shown. Each square in the insert represents one alignment. It is clearly visible that many different alignments with the same score are generated. These alignments vary not only in respect to their frequency, they also have different degrees of similarity to the structure alignment.

both sequences, starting in the upper left corner of the dot plot and proceeding to about position 45 of both sequences, as well as the region of high match probabilities at the C-terminus, starting with a match of position 102 of 1GDJ and position 93 of 2HHB_A and proceeding to the C-terminal end of sequence 2HHB_A at the lower right corner of the dot plot, are in good agreement with the structure alignment. In the middle of the dot plot a third region of high match probabilities emerges. In this region more alternatives to the matches with high match probabilities exists. The different structure alignments in Figure 19 show no consistent solution in this region. The partial agreement between the structure alignment and the dot plot of match probabilities demonstrates that regions of high match probabilities can identify reliably aligned regions. It is however important to take into account the number and probabilities of alternative matches in the vicinity of each match.

The results of the dot plot of match probabilities were compared to a null model, which assumes that the two sequences are unrelated. To generate the null model, the sequence of residues in 1GDJ and 2HHB_A were shuffled to produce randomized sequences with the same overall composition as the original sequence. The randomized sequences were used to compute a match probability dot plot and 1 million stochastic alignments. The dot plot of the match probabilities after shuffling 1GDJ and 2HHB_A is depicted in Figure 22. In Figure 22 no regions of high match probabilities exists. For each position of one of the two sequences a multitude of possible matches to different positions of the other sequence can be found. The comparison of the match probability dot plot of the null model to the dot plot of probabilities of all possible matches between 1GDJ and 2HHB_A emphasizes the fact, that match probability dot plots reveal conserved features between related sequences.

Next we examined how the stochastic alignments are distributed in respect to the % identity with the structure alignment. Figure 23 shows that the vast majority of the stochastic alignments exhibit more than 60% identity to

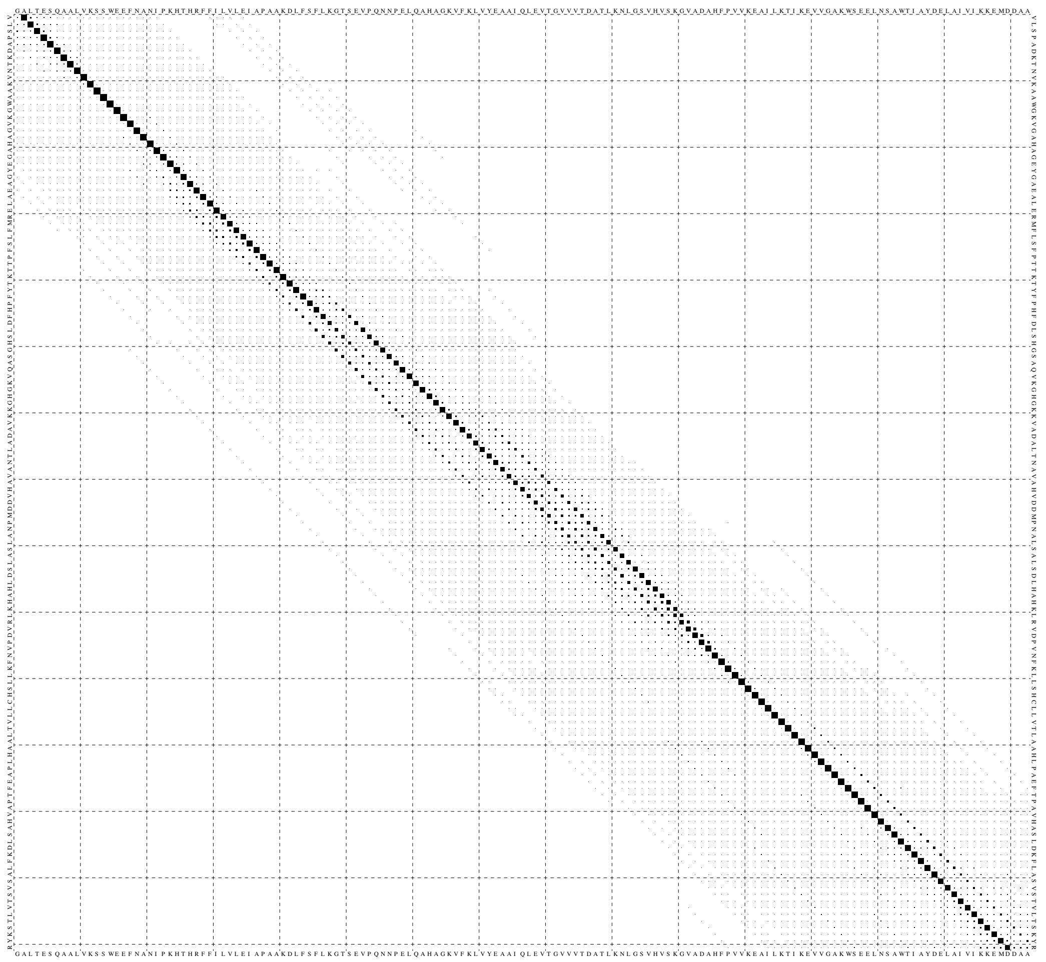


Figure 21: Dot plot of the match probabilities for the comparison between lupin leghaemoglobin, 1GDJ, and chain A of human deoxyhaemoglobin, 2HHB_A.

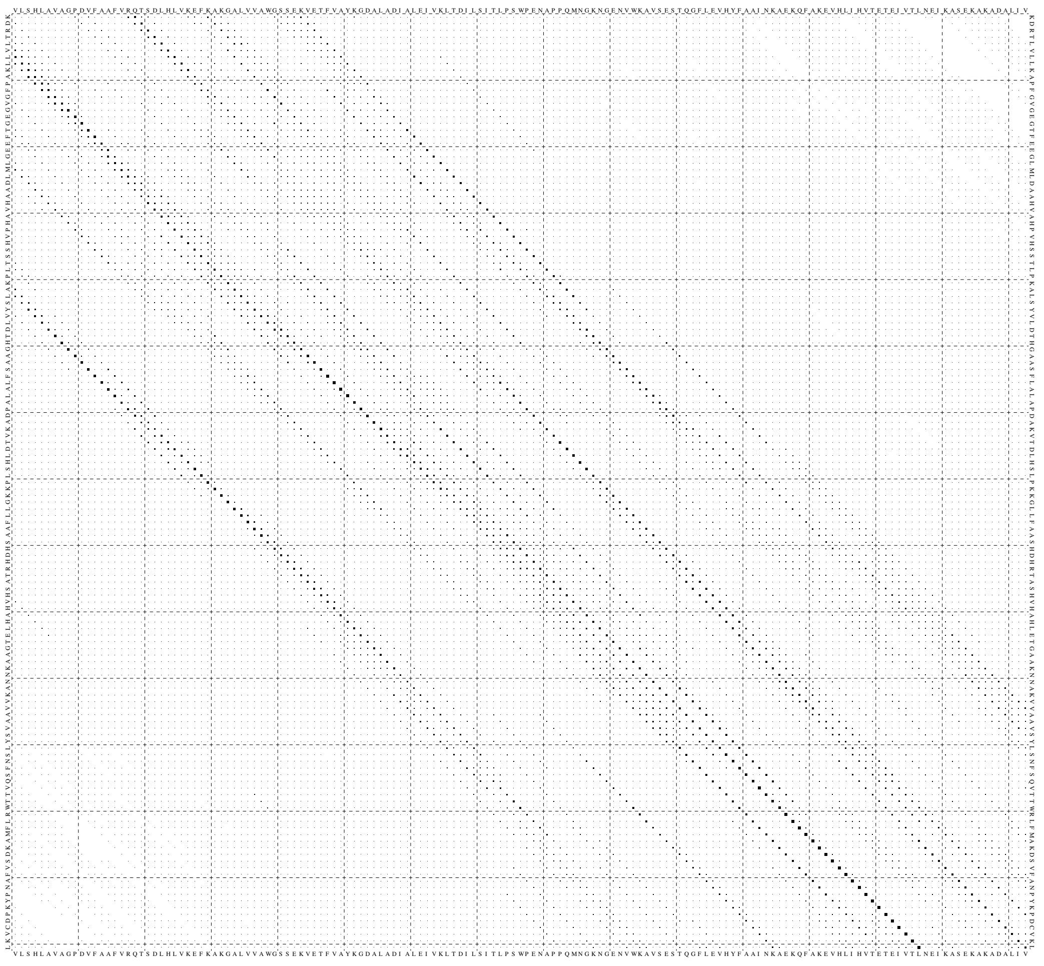


Figure 22: Dot plot of the match probabilities after shuffling 1GDJ and 2HHB_A.

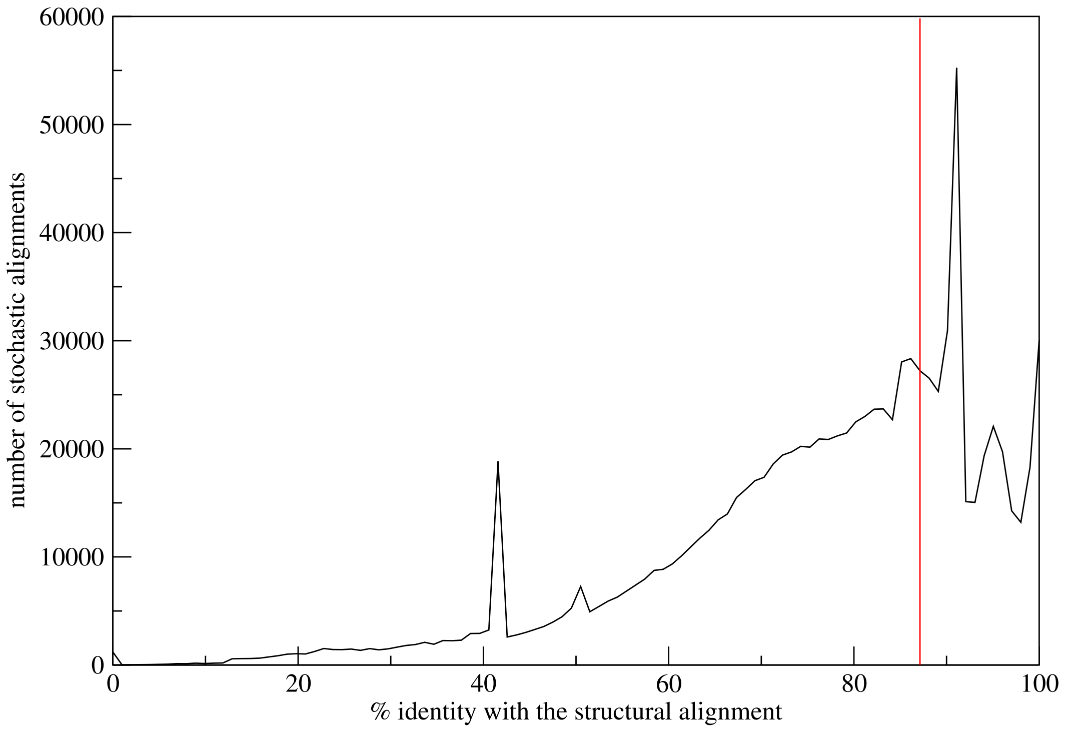
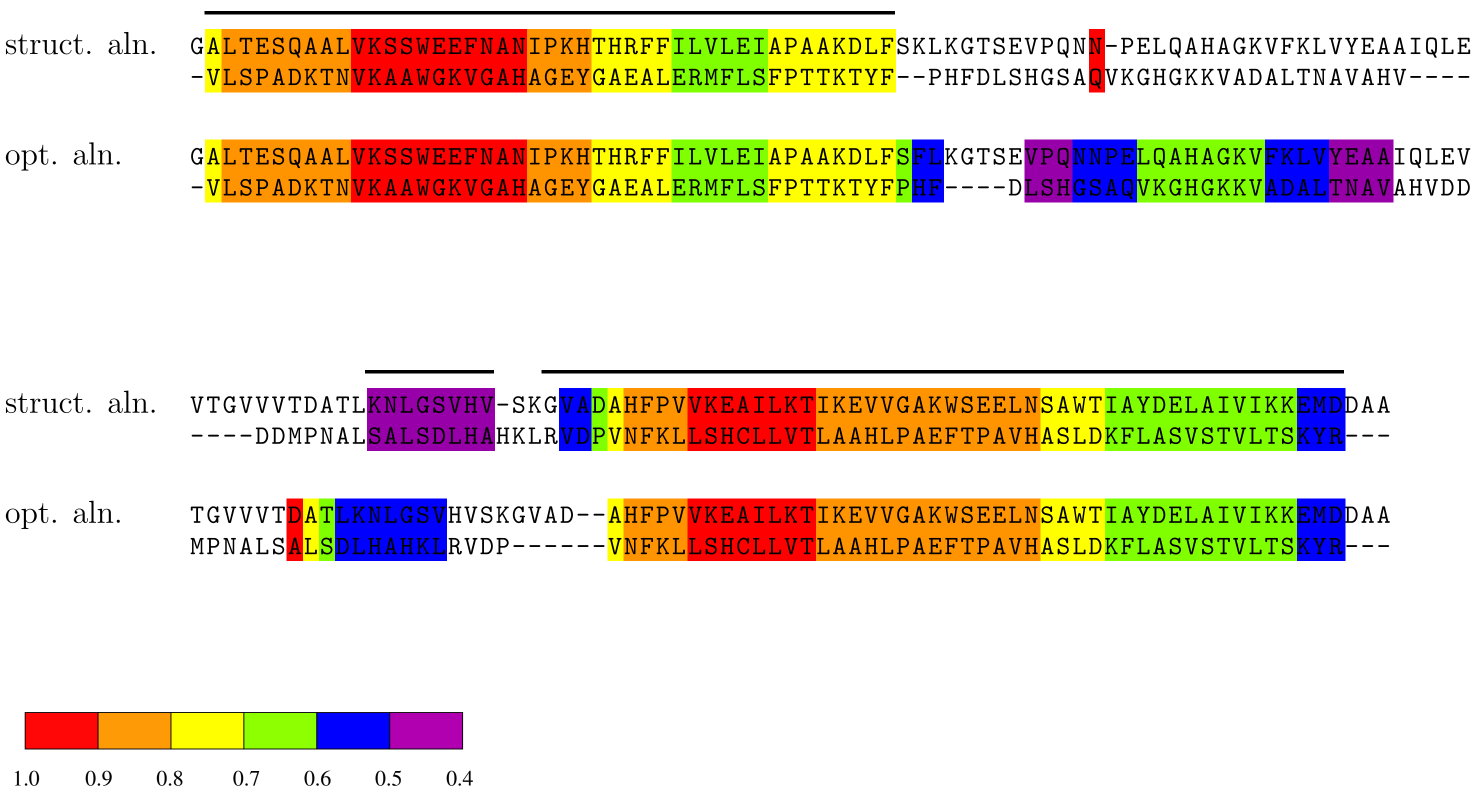


Figure 23: Distribution of stochastic alignment with different % identities to the structure alignment. The red line indicates the position of the optimal alignment.

the structure alignment. The distribution of the number of alignment with a certain % identity to the structure alignment shows a sharp peak at 41.58 % identity, corresponding the percentage of all reliably aligned residues of the first reliably aligned fragment in Figure 19. The small peak at 50.5 % identity to the structure alignment is in accordance with the sum of the percentages of the first and the small second reliably aligned region. The sharp peak at 91.09% is commensurate to the sum of the percentages of the first and the third reliably aligned region. The peaks in Figure 23 therefore correspond to alignments that contain some of the reliable fragments of the structure alignment. The reason for the higher number of alignments with % identities to the structure alignment of 41.58%, 50.5% and 91.09% is easily visible in Figure 24. Figure 24 shows that the reliably aligned fragments of the structure alignment have a significantly higher match probability than regions in which no consistent structure alignment exists. The probability to generate an alignments which contains whole fragments of the structure alignment is therefore higher than the probability to include only some matches of one of the reliably aligned region. Figure 24 depicts also the match probability distribution in the optimal alignment. The optimal alignment exhibits only 87.13% identity to the structure alignment. The reason for the low identity of the optimal alignment to our standard of truth is also visible in Figure 24. Region of very high match probability exist at both termini of the sequences, this regions are included in the optimal as well as in the structure alignment. The fragment in the middle of each sequence, however, is not that easy to align. The structure consensus contains only one match with a match probability > 0.4 in this region. The optimal alignment, however, includes a lot of residues with a match probability > 0.4 in the alignment of this sequence fragments. Figure 23 demonstrates the important contribution of suboptimal alignments to the evolutionary correct solution to the alignment problem. In the case of the alignment between 1GDJ and 2HHB_A the optimal alignment contains all residues with high match probabilities that are in agreement with the construction of an alignment - but fails in retrieving the correct solution.

A great number of near optimal and suboptimal alignments exists that bear a greater resemblance to the structure alignment. However in the absence of a standard of truth no mean exists to judge with ones of the many suboptimal alignments provide a better solution than the optimal one.

To study the correlation between the % identity to the structure alignment and the alignment store, the joint probability of getting an alignment with a certain score and a certain % identity to the structure alignment was calculated. Figure 25 show that for low alignment scores and low % identities to the structure alignment a direct correlation between the alignment score and the % identity with the structure alignment exits. For a % identity higher than 85%, however, this correlation does not exist any more. The optimal alignment has a % identity of 87% and this fact causes the disappearece of a correlation between the score of an alignment and the % identity to the structure alignment. The highest scoring alignment with an identity to the structure alignment has a score of 50.35, which is very similar to the score of the optimal alignment, 52.97. However, alignments with a % identity of 100% can also score below 0. This demonstrates that the alignment score is no absolute measure of the biological correctness of an alignment and that a direct correlation between the score of an alignment and the evolutionary correct solution does not exists.



```
struct. aln.   GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSKLKGTSEVPQNN-PELQAHAGKVFKLVYEAAIQLE
               -VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYF--PHFDLSHGSAQVKGHGKKVADALTNAVAHV----

opt. aln.      GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSEVPQNNPELQAHAGKVFKLVYEAAIQLEV
               -VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF----DLSHGSAQVKGHGKKVADALTNAVAHVDD


struct. aln.   VTGVVVTDATLKNLGSVHV-SKGVADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMDDAA
               ----DDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---

opt. aln.      TGVVVTDATLKNLGSVHVSKGVAD--AHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMDDAA
               MPNALSALSDLHAHKLRVDP------VNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR---
```

Figure 24: Match probabilities in the 3D structure alignments and in the optimal alignment are indicated by different colors. The color code for match probabilities are shown below the alignments. Reliably aligned regions of the structure alignment are indicated by a black line above the alignment.

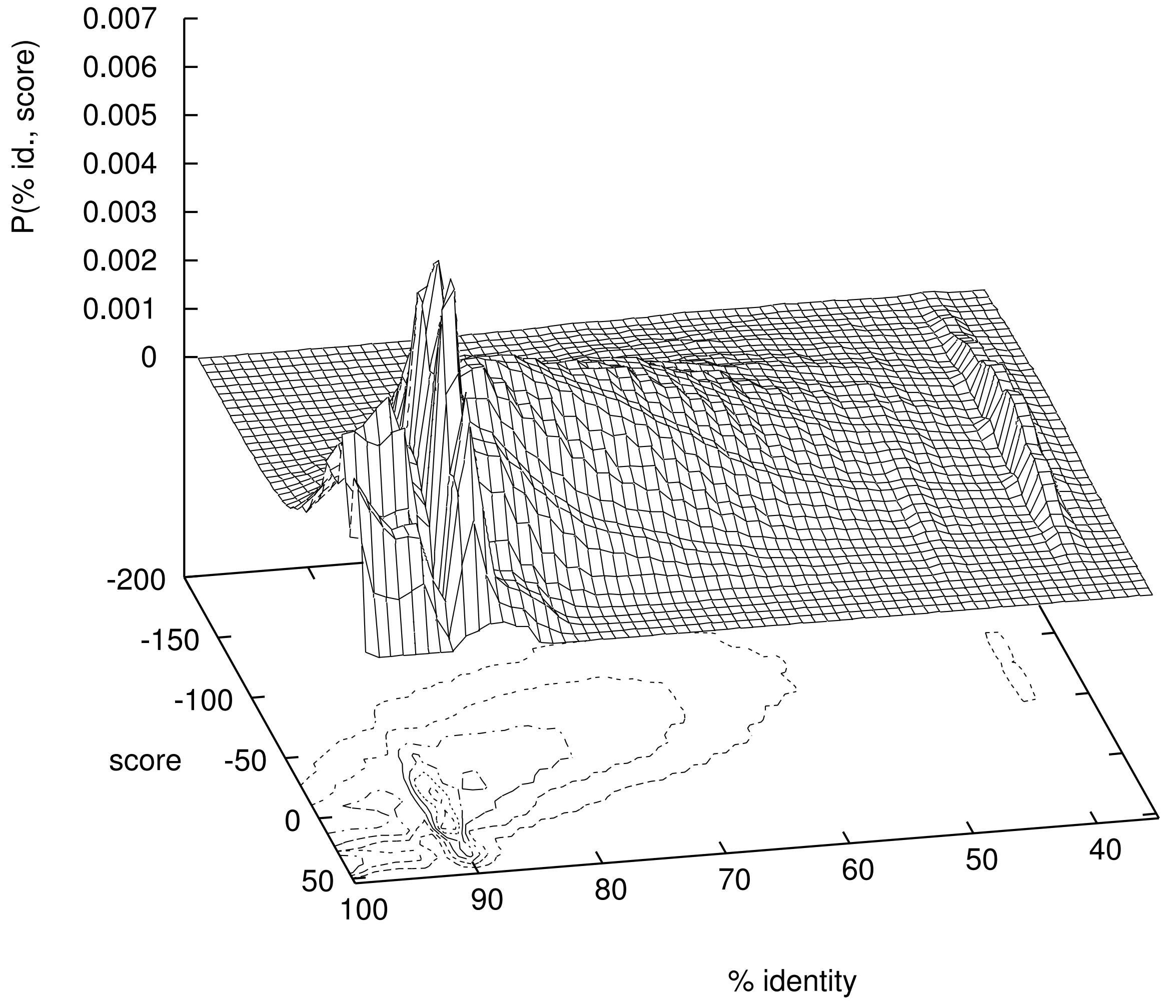


Figure 25: 3D plot of the joint probability of getting an alignment with a certain score and a certain %identity to the structure alignment

5 Conclusion and Outlook

Most alignment programs try to find the evolutionary correct solution of the alignment problem by providing one optimal alignment. Alignment algorithms, however, produce an "optimal alignment" also when applied to sequences that are not related at all. It is therefore important to find a measure of the accuracy of an alignment. Statistical models of evolution or the simulation of random mutation of a pair of initially related sequences were employed to evaluate the accuracy of optimal alignments. This methods provide an overall measure of alignment accuracy but they do not provide any information about regional differences of alignment quality. The probability of substitutions and the overall mutation rate varies in different regions of biological sequences. In our work we provide detailed information on what parts of an alignment are reliably aligned and what part display a high degree of ambiguity.

For the evaluation of the regional difference in the reliability of pairwise alignments several methods have been used. In one approach the information contained in near optimal and suboptimal alignments is used to detect reliably aligned regions [55, 51]. One example of this method was provided by Vingron and Argos [66]. They constructed suboptimal alignments by computing the optimal alignment trough every possible match between two sequences. Aligned sequences fragments were considered "stable" (reliably aligned) if they appeared in the optimal as well as in near optimal alignments.

This approach for the construction of suboptimal alignment generates only a limited sample of suboptimal alignments. Thus one can easily miss suboptimal solutions that contribute significant to the evolutionary correct alignment.

An approach that takes into consideration the full set of suboptimal alignments is the calculation of the partition function of all alignments of two sequences. A modification of the dynamic programming algorithm commonly used to calculate optimal alignments can be applied to calculate the partition function of all solutions of a search space. Miyazawa [49] was the first one who used the partition function to compute the probability of each possible match between two sequences.

In our work we developed an improved variation of a dynamic programming algorithm for the calculation of the partition function: Dynamic programming algorithms show an inherent ambiguity that is caused by the non-uniqueness of an optimal solution and the particular recursion scheme by which the search space is evaluated [18]. The ambiguity that arises from the recursion scheme results in the generation of duplicate and non-canonical solutions. If redundant or non canonical alignments are not excluded the results of the partition function can be misleading. We used a variation of a dynamic programming algorithms that employs an asymmetric recursion scheme to avoid duplicate and non-canonical solutions. This is done by determining a fixed sequence of gaps if gaps in one sequence are adjacent to gaps in the other one (see page 37). Furthermore we used the match probabilities to implemented a stochastic backtracking. Stochastic backtracking provides an ensemble of possible alignments with a distribution corresponding to the probability of each individual alignment.

To evaluate the performance of our program we calculated the entropy of the stochastic ensemble. The entropy of the stochastic ensemble is a measure of the diversity of the alignments generated by stochastic backtracking. For closely related sequences one expects to obtain a stochastic ensemble that is dominated by the optimal alignment between the two sequences. The diversity of the stochastic ensemble as well as its entropy should be low. If two sequences are very different many possibilities to align them exist resulting in a high entropy of the stochastic ensemble. We computed the mean entropy of alignments of two random as well as two identical sequences for different sequence lengths (see Figure 17) and found the expected correlation between the entropy of the stochastic ensemble and sequence homology. Furthermore we showed that this relationship between sequences similarity and entropy is also true for biological sequences (see Figure 17).

To get on overview of the stochastic alignment space we analysed the alignment between leghaemoglobin from yellow lupin, *Lupinus luteus*, (1GDJ) and chain A of human deoxyhaemoglobin (2HHB_A). The pairwise identity between 1GDJ and 2HHB_A is low but they share a high structural similarity. For this analysis one million stochastic alignments were compared to a 3D structure alignment. The structure alignment was constructed by extracting reliably aligned regions from five different structure alignments generated by various Web-accessible alignment programs (see page 56ff).
The ensemble of stochastic alignments between lupin leghaemoglobin and chain A of human deoxyhaemoglobin is dominated by suboptimal alignments with a score well below the score of the optimal alignment. Only 2.48% of all alignments have a score within $1kT$ of the optimal alignments. The frequency of identical stochastic alignments increases exponentially with the score of the alignment. The vast majority of stochastic alignments has more than 60% identity to the structural alignment (see Figure 23). The optimal alignment, however, has only an identity of 87% to the structural alignment. The stochastic ensemble contains many alignment with 100% identity to the structure alignment, the highest scoring 100% identical alignment has a score of 50.35, that is close to the score of the optimal alignment ($S_{\text{opt}} = 52.97$), the lowest scoring alignment with 100% identity to the structure alignment has a score of -131.31. The number of alignments with different % identities to the structure alignment shows a notable increase at % identities that correspond to alignments which contain some of the the reliable aligned regions of the structure alignment. The preferential generation of alignments that contain reliable aligned parts of the structure alignments is a result of the higher match probabilities in this regions. This is clearly visible in Figure 24 which shows the match probability distribution in the 3D structure and the optimal alignment. Nearly all matches in the reliably aligned fragments of the 3D structure alignment have a match probability > 0.4, whereas the regions of the structure alignment for which no reliable alignment exists contain only one match with a probability > 0.4. Regarding the apparent

correlation between higher match probabilities and regional alignment reliability in the structure alignment it is tempting to use match probability as a measure of regional reliability. However this obvious correlation between match probability and reliability vanishes at closer inspection of the optimal alignment. The optimal alignment, which has an identity of 87% to the structure alignments, contains many matches with probabilities between 0.4 to 0.7 that are not included in the structure alignment. Most of this matches reside in regions for which no consensus exits in the structure alignment. However a run of matches with probabilities between 0.5 to 0.6 overlaps with a run of matches of lower probability that are included in the structure alignment. Our example indicates that contiguous series of matches with match probabilities > 0.7 are good candidates for reliable aligned regions. To determine the correlation between match probabilities < 0.7 and the reliability of aligned fragments containing matches in this probability range further studies are necessary.
We also examined the correlation between the score of an alignment and its % identity to the structure alignemt (see Figure 25). In our example no correlation between alignment score and % identity to the structure alignment was found. The scores for alignments with 100% identity to the structure alignment varied between 50.35 and -131.32.
In our work indicates a correlation between regions of high match probability and reliably aligned regions. A correlation between match probability and alignment quality was also found by Miyazawa [49], who used a match probability threshold of > 0.5 for the prediction of reliable aligned regions. This threshold did not work for the example we calculated. However the scoring system, in particular the scores for gaps, are different in Miyazawa's work, which could cause different match probability values. To study the extent of correlations between reliable aligned regions and match probability more examples alignments of sequences of known structure are necessary. It would also be interesting to examine if there is any correlation between regions of high match probability and functionally conserved elements. Furthermore

our program can be expanded to the calculation of multiple alignments. Multiple alignments of different suboptimal alignments of the same two sequences as well as multiple alignments of related sequences can help to determine reliably aligned regions.

References

[1] N.N. Alexandrov and D. Fischer. Analysis of topological and nontopological structural similarities in the PDB: new examples with old structures. *Proteins: Struct. Funct. Genet.*, 25:354–65, 1996.

[2] S. F. Altschul. Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.*, 219(3):555–65, 1991.

[3] S. F. Altschul and B. W. Erickson. Locally optimal subalignments using nonlinear similarity functions. *Bull. Math. Biol.*, 48(5-6):633–60, 1986.

[4] S. F. Altschul and B. W. Erickson. Optimal sequence alignment using affine gap costs. *Bull. Math. Biol.*, 48(5-6):603–616, 1986.

[5] S.F. Altschul. A protein alignment scoring system sensitive at all evolutionary distances. *J. Mol. Evol.*, 36(3):290–300, 1993.

[6] Winona C. Barker, John S. Garavelli, Zhenglin Hou, Hongzhan Huang, Robert S. Ledley, Peter B. McGarvey, Hans-Werner Mewes, Bruce C. Orcutt, Friedhelm Pfeiffer, Akira Tsugita, C. R. Vinayaka, Chunlin Xiao, Lai-Su L. Yeh, and Cathy Wu. Protein Information Resource: a community resource for expert annotation of protein data. *Nucleic Acids Research*, 29(1):29–32, 2001.

[7] R. Bellman and D. Stuart. *Applied Dynamic Programming.* Princeton University Press, Princeton, New Jersey, 1962.

[8] S. A. Benner, M. A. Cohen, and G. H. Gonnet. Amino acid substitution during functionally constrained divergent evolution of protein sequences. *Protein Eng.*, 7(11):1323–32, 1994.

[9] Humberto Carillo and David Lipman. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, 48(5):1073–82, 1988.

[10] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model for evolutionary change in proteins. *In M.O. Dayhoff (ed), Atlas of Protein Sequence and Structure. National Biomedical Research Foundation, Washington, DC*, 5:345–352, 1978.

[11] R. Durbin, S. R. Eddy, A. Krogh, and Mitchison G. *Biological sequence analysis*, chapter 2. Cambridge University Press, 1998.

[12] D. F. Feng, M. S. Johnson, and R. F. Doolittle. Aligning amino acid sequences: comparison of commonly used methods. *J. Mol. Evol.*, 21(2):112–25, 1984-85.

[13] G. Fermi, M.F. Perutz, B. Shaanan, and R. Fourme. The crystal structure of human deoxyhaemoglobin at 1.74 a resolution. *J. Mol. Biol.*, 175:159, 1984.

[14] W. Fitch and T. Smith. Optimal sequence alignments. *Proc. Natl. Acad. Sci. USA*, pages 1382–86, 1983.

[15] W.M. Fitch. An improved method of testing for evolutionary homology. *J. Mol. Biol.*, 16(1):9–16, 1966.

[16] S. J. Freeland and L. D. Hurst. The genetic code is one in a million. *J. Mol. Evol.*, 47(3):238–48, 1998.

[17] Robert Giegerich. A Systematic Approach to Dynamic Programming in Bioinformatics. Part 1 and 2: Sequence Comparison and RNA Folding. Technical report, Faculty of Technology, Bielefeld University, 33594 Bielefeld, Germany, 1999.

[18] Robert Giegerich. Explaining and Controlling Ambiguity in Dynamic Programming. Technical report, Faculty of Technology, Bielefeld University 33501 Bielefeld,Germany, Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, 2000. robert@techfak.uni-bielefeld.de.

[19] Robert Giegerich and David Wheeler. Pairwise Sequence Alignment. In *BioComputing Hypertext Coursebook*, version 2.01. http://www.techfak.uni-bielefeld.de/bcd/Curric/PrwAli/prwali.html, May 21 1996.

[20] G. Gonnet, M. Hallett, C. Korostensky, and L. Bernardin. Darwin version 2.0: An interpreted computer language for the biosciences. Dept. of Computer Science,ETH Zürich,Zürich,Switzerland, 2000.

[21] G. H. Gonnet, M. A. Cohen, and S. A. Benner. Exhaustive matching of the entire protein sequence database. *Science*, 256(5062):1443–5, 1992.

[22] G.H. Gonnet. *Handbook of Algorithms and Data Structures*. Addison Wesley, London, 1984.

[23] P. J. Goss and R. C. Lewontin. Detecting heterogeneity of substitution along DNA and protein sequences. *Genetics*, 143(1):589–602, 1996.

[24] O. Gotoh. An improved algorithm for matching biological sequences. *J. Mol. Biol.*, 162(3):705–8, 1982.

[25] O. Gotoh. Optimal sequence alignment allowing for long gaps. *Bull. Math. Biol.*, 52(3):359–73, 1990.

[26] R. Grantham. Amino acid difference formula to help explain protein evolution. *Science*, 185(154):862–4, 1974.

[27] E.H. Harutyunyan, T.N. Safonova, I.P. Kuranova, A.N. Popov, A.V. Teplyakov, G.V. Obmolova, A.A. Rusakov, B.K. Vainshtein, G.G. Dodson, J.C. Wilson, and M.F. Perutz. The structure of deoxy- and oxyleghaemoglobin from lupin. *J. Mol. Biol.*, 251:104–15, 1995.

[28] S. Henikoff. Scores for sequence searches and alignments. *Curr. Opin. Struct. Biol.*, 6(3):353–60, 1996.

[29] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, 89(22):10915–9, 1992.

[30] S. Henikoff, J.C. Wallace, and J.P. Brown. Finding protein similarities with nucleotide sequence databases. *Methods Enzymol.*, 183:111–32, 1990.

[31] D. G. Higgins and P. M. Sharp. Clustal: a package for performing multiple sequence alignments on a microcomputer. *Gene*, 73:237–244, 1988.

[32] I. L. Hofacker, W. Fontana, P. F. Stadler, L. S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of rna secondary structures. *Monatsh. Chem.*, 125:167–188, 1994.

[33] I. Holmes and Durbin R. Dynamic programming alignment accuracy. *J. Comput. Biol.*, 5(3):493–504, 1998.

[34] R.W.W. Hooft, G. Vriend, C. Sander, and E.E. Abola. Errors in protein structures. *Nature*, 381:272–272, 1996.

[35] W. Humphrey, A. Dalke, and K. Schulten. ”VMD - visual molecular dynamics”. *J. Molec. Graphics*, 14:33–38, 1996.

[36] M. S. Johnson and J. P. Overington. A structural basis for sequence comparisons. An evaluation of scoring methodologies. *J. Mol. Biol.*, 233(4):716–38, 1993.

[37] S. Karlin and S. F. Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, 87(6):2264–8, 1990.

[38] S. Karlin and G. Ghandour. Comparative statistics for dna and protein sequences: multiple sequence analysis. *Proc. Natl. Acad. Sci. USA*, 82(18):6186–90, 1985.

[39] T. Kawabata and K. Nishikawa. Protein structure comparison using the markov transition model of evolution. *Protein structure*, 41(1):108–22, 2000.

[40] M. Kimura. The neutral theory of molecular evolution. *Camb. Univ. Press*, 1983.

[41] M. Kschischo and M. Lassig. Finite-temperature sequence alignment. *Pac. Symp. Biocomput.*, 1:624–35, 2000.

[42] S. Kurtz and G. W. Myers. Estimating the probability of approximate matches. In *Proceedings of Combinatorial Pattern Matching*, pages 52–64. Springer Lecture Notes in Computer Science 1246, 1997.

[43] R.H. Lathrop. The protein threading problem with sequence amino acid interaction preferences is np-complete. *Protein Eng.*, 7(9):1059–68, 1994.

[44] G. Lu. A WWW service system for automatic comparison of protein structures. *Protein Data Bank Quarterly Newsletter*, (78):10–11, 1996.

[45] G. Lu. A new method for Protein Structure Comparisons and Similarity Searches. *Journal of Appl. Cryst.*, 33:176–183, 2000.

[46] J.S. McCaskill. The equilibrium partition function and base pair binding probabilities for rna secondary structure. *Biopolymers*, 29(6-7):1105–19, 1990.

[47] H. T. Mevissen and M. Vingron. Quantifying the local reliability of a sequence alignment. *Protein Eng.*, 9(2):127–32, 1996.

[48] W. Miller and E. W. Myers. Sequence comparison with concave weighting functions. *Bull. Math. Biol.*, 50(2):97–120, 1988.

[49] S. Miyazawa. A reliable sequence alignment method based on probabilities of residue correspondences. *Protein Eng.*, 8(10):999–1009, 1994.

[50] David Mount. *Bioinformatics: Sequence and Genome Analysis.* Cold Spring Harbor Laboratory Pr., 2001.

[51] D. Naor and D. L. Brutlag. On near-optimal alignments of biological sequences. *J. Comput. Biol.*, 1(4):349–66, 1994.

[52] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48(3):443–53, 1970.

[53] A. Sali and T.L. Blundell. Definition of general topological equivalence in protein structures. A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J. Mol. Biol.*, 212(2):403–28, 1990.

[54] D. Sankoff and J. Kruskal, editors. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison.* Addison-Wesley. London, 1983.

[55] M. A. Saqi and M. J. Sternberg. A simple method to generate non-trivial alternate alignments of protein sequences. *J. Mol. Biol.*, 219(4):727–32, 1991.

[56] P. H. Sellers. An algorithm for the distance between two sequences. *J. of Combinatorics Theory*, 16:253–58, 1974.

[57] I.N. Shindyalov and P.E. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Eng.*, 11:739–47, 1998.

[58] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147(1):195–7, 1981.

[59] T. F. Smith, M. S. Waterman, and W. M. Fitch. Comparative biosequence metrics. *J. Mol. Evol.*, 18(1):38–46, 1981.

[60] R. Spang and M. Vingron. Limits of homology detection by pairwise sequence comparison. *Bioinformatics*, 17(4):338–42, 2001.

[61] H. Tang and R. C. Lewontin. Locating regions of differential variability in DNA and protein sequences. *Genetics*, 153(1):485–95, 1999.

[62] J. D. Thompson, D. G. Higgins, and T. J. Gibson. Clustal W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, 22(22):4673–80, 1994.

[63] S.M. Ulam. Some ideas and prospects in biomathematics. *Annu. Rev. Biophys. Bioeng.*, 1:277–92, 1972. Review.

[64] R. Unger, D. Harel, S. Wherland, and Sussman J.L. A 3D building blocks approach to analyzing and predicting structure of proteins. *Proteins*, 5(4):355–73, 1989.

[65] M. Vingron. Near-optimal sequence alignment. *Curr. Opin. Struct. Biol.*, 6(3):346–52, 1996.

[66] M. Vingron and P. Argos. Determination of reliable regions in protein sequence alignments. *Protein Eng.*, 3(7):565–9, 1990.

[67] M. Vingron and M. S. Waterman. Sequence alignment and penalty choice. review of concepts, case studies and implications. *J. Mol. Biol.*, 235(1):1–12, 1994.

[68] G. Vogt, T. Etzold, and P. Argos. An assessment of amino acid exchange matrices in aligning protein sequences: the twilight zone revisited. *J. Mol. Biol.*, 249(4):816–31, 1995.

[69] Michael S. Waterman. *Introduction to Computational Biology: Maps, Sequences and Genomes.* Chapman & Hall/CRC, 1 edition, 1995.

[70] M. Zuker. On finding all suboptimal foldings of an RNA molecule. *Science*, 244:48–52, 1989.

[71] M. Zuker. Suboptimal sequence alignment in molecular biology. Alignment with error analysis. *J. Mol. Biol.*, 221(2):403–20, 1991.

List of Figures

List of Tables

Curriculum vitae

Name:	Ulrike Mückstein
Geburt:	2. November 1969 in Wien
Staatsbürgerschaft:	Österreich
Familienstand:	ledig

Schulbildung:	BG und BRG in Baden, Frauengasse 3 - 5, Matura 1988
Studium:	Beginn des Studiums der Biologie an der Universität Wien im Oktober 1988
	Beginn des Studium Irregulare der Molekulargenetik im März 1990
	1.Diplomprüfung im September 1996
	Diplomarbeit Dezember 1999 - Oktober 2001
Familie:	14. Oktober 1991 Geburt meines Sohnes Raphael
Praxis:	1988/1989 freie Mitarbeiterin bei Firma Vitroplant, Klosterneuburg
	1997/98 drei Monate bei Novartis, Wien

Publikationen:

[1] Fekete M., Flamm C., Hofacker I.L., Mückstein U., Rauscher S., Stadler P.F., Stocsits R., Thurner C. and Witwer C. Automatic Detection of Conserved Secondary Structure Elements in Viral Genomes. Poster.