An Overview of Multiple Sequence Alignment

Multiple sequence alignment (MSA) is the central technique for inferring biological information from a set of sequences. It involves the alignment of more than two sequences and aims to find equivalent positions across the aligned query set of sequences. An MSA can be viewed as a 2-dimensional table in which the sequences are the rows, and where the columns of equivalent amino acids have been arranged by placing gap characters in appropriate positions, such that the biological relationship of the sequences is best represented. An MSA can provide a wealth of information about structure-function relationships within a set of protein sequences, e.g., the evolutionary conservation of functionally or structurally important amino acids at certain sequence positions or conserved hydrophobicity patterns in particular regions. It is also often useful as a starting point for site-directed mutagenesis experiments. In addition to being an important means for gleaning the above biological clues by visual inspection, MSAs are an essential prerequisite to many computational modes of analysis of protein families, such as homology modeling, secondary structure prediction, and phylogenetic reconstruction. They may further be used to derive profiles (Gribskov et al., 1987) or hidden Markov models (Haussler et al., 1993; Bucher et al., 1996) that can be used to scour databases for distantly related members of the family.

An MSA is an attempt to represent evolutionarily related sequences in the most consistent way. Finding the maximal alignment score according to a given evolutionary model is equivalent to maximizing the probability that the sequences evolved as given by the MSA. Despite the considerable history of MSA (Needleman and Wunsch, 1970; Smith and Waterman, 1981; Hogeweg and Hesper, 1984; Barton and Sternberg, 1987; Corpet, 1988; Higgins and Sharp, 1988; Taylor, 1988; Altschul et al., 1989; Gotoh, 1993, 1996; Thompson et al., 1994a,b; Heringa, 1999, 2002; Notredame et al., 2000; Lee et al., 2002), the methodology is still under continuous development.

However, because a complicated relationship often exists between homologous sequences, combined with a lack of information about their true evolutionary history, absolute certainty about the correctness of MSAs is often hard to achieve. Therefore, it is instructive to bear in mind that a computational biologist is dealing with "truth" or "correctness" at two levels: the first level being the biological reality (current and past), and the second the investigator's model of this reality in terms of scoring schemes, graphs, or alignments. Thus, it is appropriate to quote: "All models are wrong but some are useful" (George E.P. Box). The fundamental model for all types of sequence comparisons is the (generalized) model of evolution. Such models have been derived from trusted sequence alignments by assuming a Markov model of evolution (Dayhoff et al., 1978; Müller and Vingron, 2000) or by empirical derivation (Benner et al., 1993; Henikoff and Henikoff, 1994). The information about sequence evolution derived from such models is usually stored in amino acid substitution matrices. In the following sections, the focus is on higher-level aspects of modeling, concerning the complex relationships within sequence families, such as tree construction and progressive alignment strategies. These strategies are discussed along with practical considerations about the construction of meaningful MSAs.

With the completion of the first draft of the human genome and well over one hundred genomes of other species, the accurate alignment of biological sequences has become more important than ever. This is due to the fact that the direct prediction of a protein's structure and function is still a major unsolved problem. To increase the knowledge of the function and interaction of protein sequences obtained by sequencing techniques, many initiatives are underway for large-scale proteomics and structure elucidation of novel genomic proteins. However, roughly 50% of the proteins in most sequenced species do not have an assigned function, and consequently an important target of bioinformatics method development is aimed at gathering the function of an increased fraction of translated proteins by enhancing comparative sequence techniques and threading protocols. In the quest for knowledge about the role of a certain unknown protein in the cellular molecular network, comparing the query sequence with the many sequences in annotated protein sequence databases often leads to useful suggestions regarding the protein's 3-dimensional (3-D) structure or molecular function. These suggestions are obtained by extrapolating the properties of sequences, residing in annotated public databases, which are identi-

Contributed by Victor Simossis, Jens Kleinjung, and Jaap Heringa

Current Protocols in Bioinformatics (2003) 3.7.1-3.7.26

Copyright © 2003 by John Wiley & Sons, Inc.

fied as "neighbors" of the query sequence by comparative sequence analysis techniques. During the last three decades, such homology search methods have arguably led to the putative characterization (annotation) of more sequences than any other single technology. Significant progress has been made in homology searching over the last few years by employing MSA techniques in iterative sequence database search strategies (Taylor, 1988; Altschul et al., 1997). Since the advent of the genome sequencing projects and resulting rapid expansion of sequence databases, the method of indirect inference by comparative sequence techniques has only gained in significance. Many current research projects aim to improve the sensitivity of (multiple) sequence alignment techniques, which require high-performance computing given the current and rapidly growing database sizes. Given the plethora of sequence data, the alignment engines also have to be extremely fast and fully automatic to be included in genomic pipelines.

Before embarking on a review of the large number of approaches to multiple sequence alignment, it is important to stress that each of the methods comes with individual strengths and weaknesses relating to accuracy, sensitivity, speed, consistency, versatility, and the like. There is no single best method for each individual query set of sequences. The user should therefore attempt a number of methods, preferably including different strategies, to approach the biological complexity of most MSA problems; the only exceptions are those cases where the sequences turn out to be highly similar.

Global or Local Methods

Many MSA techniques perform *global* alignment (Needleman and Wunsch, 1970) and match sequences over their full lengths. Problems with this approach can arise when sequences that are only homologous over local regions are compared. In such cases, global alignment techniques might fail to recognize highly similar internal regions because these may be overshadowed by dissimilar stretches and high gap penalties normally required to achieve proper global matching. Moreover, many biological sequences are modular and show shuffled domains (Heringa and Taylor, 1997), which can render a global alignment of two complete sequences meaningless. The occurrence of varying numbers of internal sequence repeats (Heringa, 1998) can also severely limit the applicability of global methods. In general, when there is a large difference in the lengths of two sequences to be compared, it is advisable to include local alignment techniques in the analysis. To address these problems, Smith and Waterman (1981) early on developed a so-called *local* alignment technique in which the most similar regions in two sequences are selected and aligned. The algorithm has been extended in various techniques to compute a list of top-scoring pairwise local alignments (Waterman and Eggert, 1987; Huang et al., 1990; Huang and Miller, 1991). Alignments produced by the latter techniques are nonintersecting, i.e., they have no matched pair of amino acids in common. For multiple sequences, the main automatic methods include the Gibbs sampler (Lawrence et al., 1993), MEME (Bailey and Elkan, 1994) and Dialign2 (Morgenstern, 1999). These local MSA programs often perform well when there is a clear block of ungapped alignment shared by all of the sequences, but perform poorly under moderate gap requirements and show inferior results over general sets of test cases when compared with global methods (Thompson et al., 1999b; Notredame et al., 2000).

Performing MSA

Carrying out an MSA of a given protein sequence set and extracting maximum information from the alignment involves a number of critical steps:

1. The selection of sequences.
2. The choice of the scoring function used to compare sequences or sequence blocks.
3. The application and optimization of this scoring function in compiling the alignment.

Selecting the Sequences

An MSA can be misleading when a sequence set contains sequences that are not homologous. Ideally, the sequences should all be orthologs, but in practice it is often difficult to ensure that this is the case. It should be stressed that most MSA routines will produce an alignment even in the case of biologically unrelated sequences, which can give rise to spurious suggestions regarding the proteins' structure or function ("garbage in, garbage out"). A widely used way to create a sequence set around a given query sequence of interest is to employ a homology searching technique (e.g., Altschul et al., 1997; Taylor, 1988; Karplus et al., 1998; Eddy, 1998; Karplus and Hu, 2001; *http://hmmer.wustl.edu*) to scour sequences in public sequence databases. Although the development of P- and E-values to estimate the statistical significance of putative homologs

found by these programs limits the chance of false positives, it is entirely possible that essentially nonhomologous sequences will enter the alignment set, which might confuse the alignment method used.

The Scoring Function

The scoring function is the formalization of the biological knowledge used in aligning the sequences. Ideally, it should contain all available knowledge about evolutionary, structural, and functional aspects of the compared sequences, so that the scoring function approximates the biological reality. In practice, however, this information is often not available or cannot be formalized mathematically. Although each cross-comparison of a residue between two sequences should in reality be evaluated individually based on its structural and functional context, the most widely used scheme to compare sequences is based on generalized averages for scoring each pair of residue types, given in the form of a symmetric 20×20 amino acid exchange matrix. The scheme models the alignment of two sequences as a Markov process, where the amino acid matches are considered independent, so that the product of the probabilities for each match within an alignment can be taken. Since many of the generally applied 20×20 scoring matrices contain propensities converted to logarithmic values (*log-odds*), the alignment score can normally be calculated by summing the log-odds values corresponding to matched residues minus appropriate gap penalties:

$$S_{a,b} = \sum_l s\left(a_i, b_j\right) - \sum_k N_k \cdot gp(k)$$

where the first summation is over the exchange values associated with l matched residues and the second over each group of gaps of length k, with N_k being the number of gaps of length k and $gp(k)$ the associated gap penalty. In case affine gap penalties are used, $gp(k) = pi + k \cdot pe$, where pi and pe are the penalties for gap initialization and extension, respectively. A consequence of the widely used affine gap penalty scheme is that long gaps that are required, for example, to span a domain B in aligning a two-domain sequence AC (where A and C represent domains) with a three-domain sequence ABC, are often too costly, so that such sequences become misaligned. A complication with gap penalties is that there exists no formal model to set their values according to the evolutionary distances suggested by the exchange values within scoring matrices, so that one has to resort to empirical tuning of the gap penalties.

Applying the Scoring Function

Apart from being a fundamental biological challenge, MSA is also a computationally intense problem, which means that for all but the smallest data sets of less than 10 sequences, an exact solution is not feasible. Algorithms that perform simultaneous alignment over a multidimensional search matrix, where each sequence in the MSA represents an extra dimension (Stoye et al., 1998; Lipman et al., 1989), come closest to an exact solution.

The most populated class of algorithms is that of progressive MSA methods. The progressive strategy implies that an algorithm for pairwise sequence alignment is repeatedly used in a stepwise fashion until all sequences are aligned. In the vast majority of progressive methods the Dynamic Programming (DP) strategy is adopted. The DP strategy guarantees that, given an amino acid exchange matrix and gap penalty values, the highest scoring or optimal pairwise alignment is calculated. The progressive alignment strategy reuses the pairwise DP algorithm in a "greedy" manner; i.e., alignments formed during the progression towards the final MSA cannot be changed anymore. The main difference between the available DP-based methods is the way in which the information of aligned blocks of sequences is represented. While early methods used consensus sequences to represent alignment blocks, current methods all use a profile formalism to represent the information in an MSA (Gribskov et al., 1987). Recent developments in multiple alignment techniques have mainly focused on sensitive and optimal models to represent MSA information.

A class of techniques that are able to revisit and optimize is that of iterative multiple alignment techniques. Pioneered by Hogeweg and Hesper (1984), iterative techniques attempt to enhance the alignment quality by gleaning information from a multiple alignment constructed in an earlier round, which is then applied in a next round to improve the alignment according to a given scoring scheme. Another classes of alignments covered in this unit is stochastic alignments, where probabilistic frameworks such as hidden Markov models and Bayesian networks have been attempted. Other techniques based on fast computational techniques such as suffix trees and fast Fourier transforms (FFT) will be discussed below.

In the remainder of this unit, general methodological issues will be covered (see next section, MSA Methodology), after which an overview of current state-of-the-art methods will be presented. The overview also includes some older methods that, although no longer widely used, have been important for developments in the field. Finally, the last section offers some considerations on evaluating MSAs.

MSA METHODOLOGY

In this section the focus will be on higher-level aspects of modeling, concerning the complex relationships within sequence families, such as tree construction and progressive alignment strategies. These strategies are discussed along with practical considerations about the construction of meaningful MSAs.

Progressive Alignment Strategies

The most populated class of algorithms is that of progressive MSA methods. The progressive strategy implies that an algorithm for pairwise sequence alignment is repeatedly used in a stepwise fashion until all sequences are aligned. In the vast majority of progressive methods the Dynamic Programming (DP) strategy is adopted. The main difference between the DP-based methods is the way the information of aligned blocks of sequences is represented (see Profiles, below). To increase the chance of correct alignment, many methods calculate an appropriate order in which the sequences should be progressively aligned (Fig. 3.7.1). In many cases, this order is derived from all-against-all pairwise alignment of the sequences and the calculation of a dendrogram, often referred to as the "guide tree," using the pairwise alignment scores (see discussion of Trees, below). The resulting branch order of the dendrogram is then followed to align the sequences, such that the most similar sequences are aligned first, and gradually the more distant sequences are included in the growing MSA. Although an efficient and stable strategy, the progressive alignment protocol suffers from its greediness and is not able to revise any of the alignments made earlier, such that any alignment errors made during the construction of the MSA cannot be repaired. This drawback is particularly significant for distant sequence sets because the alignments of sequences at early steps during progressive alignments cannot make use of information from other sequences, so that proper positional information required for correct matching is not available at early stages. Only later during the alignment progression does more information from other sequences (e.g., through profile representation) become employed in the alignment steps, but this happens quite possibly after misalignment has already taken place.

MSA programs like ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*), T-Coffee (Notredame et al., 2000), and PRALINE (Heringa, 1999; Heringa, 2002) are based upon the progressive alignment strategy (Feng and Doolittle, 1987) and are all able to produce high-quality align-

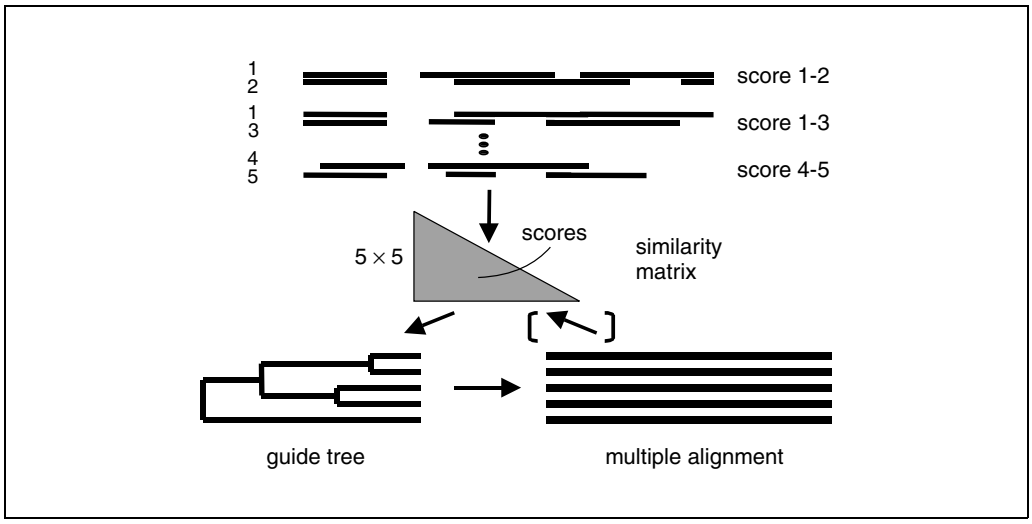

Figure 3.7.1 Representation of the progressive alignment strategy comprising compilation and scoring of all pairwise alignments, yielding a similarity matrix, which is used to construct a guide tree. The resulting MSA is constructed in the order as given by the guide tree. The arrow in brackets represents alignment iteration.

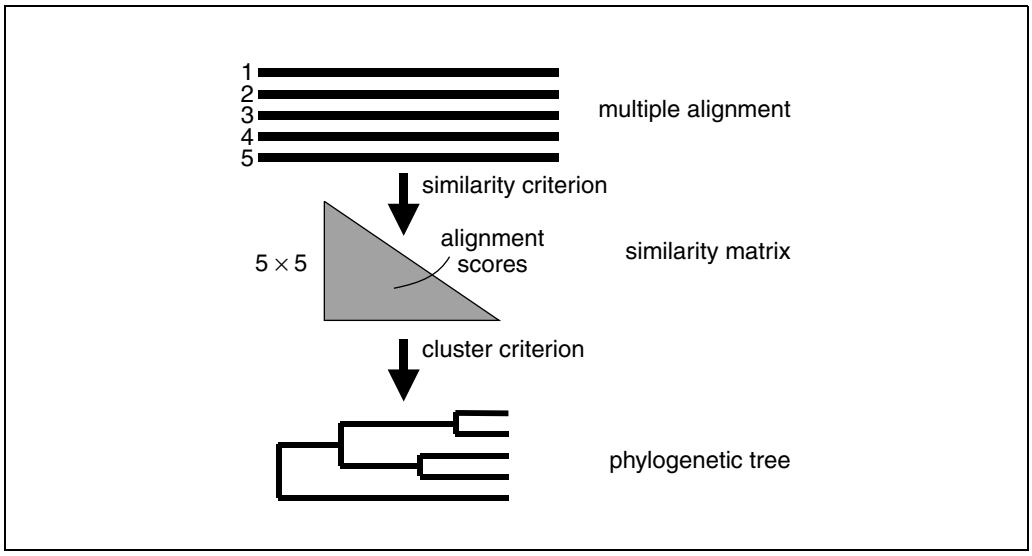

Figure 3.7.2 An MSA gives rise to a similarity matrix containing all pairwise distances, which can be clustered and represented by a phylogenetic guide tree.

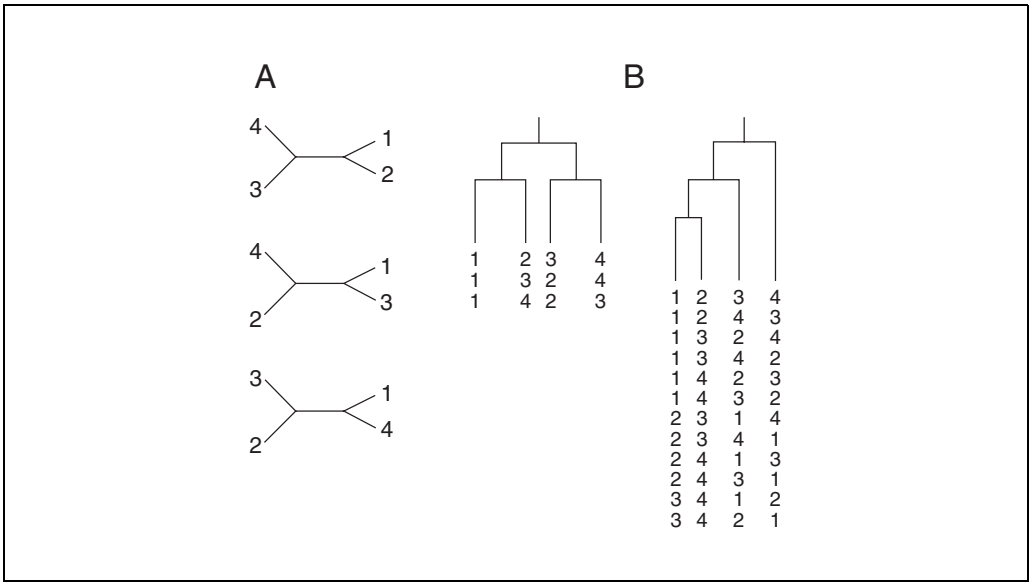

Figure 3.7.3 Illustration of all phylogenetic trees for a set of four sequences. (**A**) There are three different possible unrooted trees. (**B**) There are two different tree topologies and a total of 15 different rooted trees.

ments. This was demonstrated in a recent benchmark (Heringa, 2002) with over 144 alignments in the BAliBASE repository (Thompson et al., 1999a), although the results produced by the various programs are not necessarily identical, particularly with more divergent sequence sets.

Trees

Reconstruction of the evolutionary history of proteins (Chapter 6) is one of the central aims of sequence comparison. An evolutionary model of a particular sequence family consists of an evolutionary tree depicting the sequence relationships within the family, and an MSA (Fig. 3.7.2), which shows the detailed local relation between the individual sequences. The following conventions and terminology are adopted: (a) trees consist of edges (lines) and nodes (crossings), where edge lengths define the distance between sequences and nodes represent the actual sequences; (b) trees are binary, with one incoming edge and two outgoing edges; (c) the ultimate ancestor is called the "root" and the terminal nodes are called "leaves." Some tree construction algorithms (like parsimony; *UNIT 6.4*) give no indication about the position of the root, leading to "un-

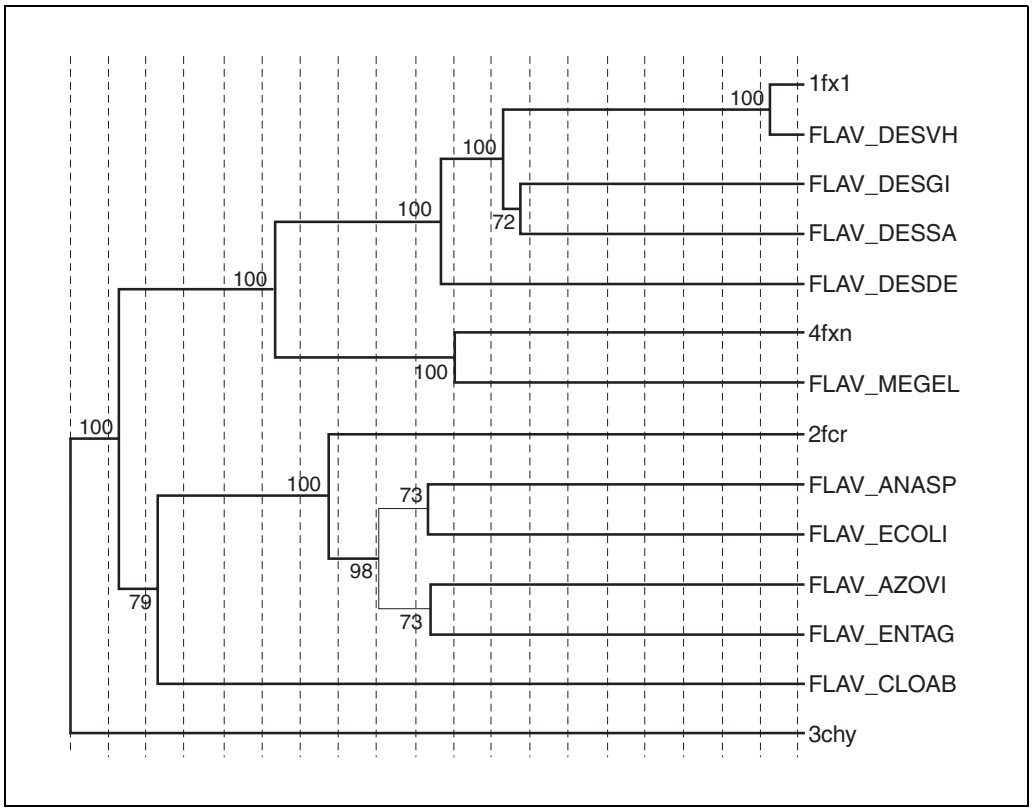

Figure 3.7.4 The phylogenetic tree of the flavodoxin family. The numbers at the ancestral nodes are bootstrap values.

rooted" trees. For the sake of completeness (and to correct some erroneous formulas in the standard literature), the equations for the number of possible unrooted and rooted trees for n sequences (Fig. 3.7.3) are presented here:

$$\text{number of unrooted trees} = \frac{(2n-5)!}{2^{n-3}(n-3)!}$$

$$\text{number of rooted trees} = \frac{(2n-3)!}{2^{n-2}(n-2)!}$$

Rooting of a tree can be achieved by adding a very distant member, also called an out-group, to the family, which defines the root as the point where its branch meets the tree.

The construction of "real" phylogenetic trees is a computer-intensive procedure that requires probabilistic modeling. A widely used approach is the application of maximum-likelihood methods (*UNIT 6.6*), which calculate the most probable phylogenetic tree associated with a given MSA and evolutionary model (Saitou, 1990). For the purpose of constructing a guide tree for an MSA (see above), more amenable ad hoc strategies are often adopted to reduce the computations. A frequently used approach is to estimate sequence distances from the pairwise alignment scores. Using these heuristic distances, a phylogenetic tree can be constructed to guide the construction of the MSA (see Progressive Alignment Strategies, above).

Regardless of whether the phylogenetic tree is calculated using maximum-likelihood (*UNIT 6.6*) or distance methods (*UNIT 6.3*), the significance of the branching of the tree can be estimated using the bootstrap (Felsenstein, 1985; *UNIT 6.1*). For bootstrapping, the columns in the multiple alignment are re-sampled a significant number of times (100 to 1000) with replacement, such that a single alignment column can occur multiple times in a resampled MSA, after which the significance for each branching in the original tree is taken from the frequency of the occurrence of the branch over all resampled trees (Fig. 3.7.4).

The transformation from pairwise alignments to a phylogenetic tree is performed by clustering algorithms, which fall into two conceptually different categories: distance-based (UPGMA, Neighbor-Joining; *UNIT 6.3*) and parsimony (*UNIT 6.4*). Distances can be obtained from sequence identities (Fitch and Margoliash, 1967) or pairwise sequence alignment scores (Hogeweg and Hesper, 1984).

UPGMA (Unweighted Pair Group Method using arithmetic Averages; Sokal and Michener, 1958) joins sequences to clusters and progressively to larger clusters until a single cluster (tree) is accomplished. The order in which sequences or clusters are joined is simple: always the closest pair of sequence/sequence, sequence/cluster, or cluster/cluster is joined. Each joining operation creates an ancestral node, and the distance between the joined sequences (which are the leaves of the tree) is expressed as:

$$\text{distance between joined sequences} = \frac{1}{n_i \cdot n_j} \sum d_{ij}$$

where n_i, n_j are the number of sequences in the two joined clusters, and d_{ij} are the distances of all possible pairwise sequence combinations between the joined clusters. For example, if cluster "A" consists of sequences (1,2,3) and cluster "B" consists of sequences (4,5):

$$\text{distance between joined sequences} = \frac{1}{3 \cdot 2}\left(d_{14} + d_{15} + d_{24} + d_{25} + d_{34} + d_{35}\right)$$

The ancestral node is exactly in the middle between the joined sequences, so that the edge length between the joined sequences and their ancestral node is 1/2 of the above distance.

Neighbor-Joining (NJ; Saitou and Nei, 1987; *UNIT 6.3*) should be performed instead of UPGMA if the distances between sequences are not additive, which is equivalent to an unscaled evolutionary clock resulting from unequal evolutionary speeds in the various branches. The algorithm starts from a distance table containing all pairwise sequence distances by joining the closest pair of sequences and placing their ancestral node at half distance. The distance table is updated: the two columns/rows of the joined sequences are fused together and the average distance of the two joined sequences to every other sequence is computed. For example, for a sequence set A, B, C, D, after joining the closest sequence pair A and B to AB:

$$\text{average distance of AB to C} = \frac{1}{2}\left(d_{AC} + d_{BC}\right)$$

and

$$\text{average distance of AB to D} = \frac{1}{2}\left(d_{AD} + d_{BD}\right)$$

The closest pair, assuming (AB) and C, is joined to (ABC) and their ancestral node is placed at half distance. The procedure of average distance computation and joining is repeated until all sequences are included. The resulting tree is unrooted.

The Maximum Parsimony (MP; *UNIT 6.4*) method is an algorithm to find the tree that minimizes residue substitutions summed up over all sites of the whole tree (Eck and Dayhoff, 1966; Kluge and Farris, 1969). Therefore, a sufficient number of different tree topologies is generated in a first phase, and a cost for residue substitutions is assigned to each tree in a second phase. This cost function can be simply the total number of residue substitutions or the sum of weights W_{AB} for each substitution from residue A to residue B (weighted parsimony). The algorithm of Fitch (1971) is usually used for counting the number of residue changes. Sequences on ancestral nodes can be inferred if pointers between residues on ancestral and daughter nodes are stored. The number of possible tree topologies increases drastically with the number of sequences (see above), but stochastic approaches have been developed (Felsenstein, 1981).

Profiles

An MSA profile is a comprehensive representation of an MSA, stressing the composition of the alignment positions (columns) rather than the composition of the constituting sequences. In general form, a profile is a vector composed of 20 components (amino acids) at each MSA position. The vector components describe the contribution (score, weight, probability etc.) of each amino acid at this position to the MSA. Profiles are used to align an MSA to a single sequence or to another MSA by means of dynamic programming. Scoring of such multiple-to-single or multiple-to-multiple alignments is, in a theoretical sense, far from trivial, because the substitution matrices are strictly derived from probabilities of pairwise residue alignments.

However, classical pairwise substitution scores are commonly averaged or condensed by some other linear transformation over the representing amino acids at each alignment position, yielding a "position-specific scoring matrix" (PSSM). The equation for the average profile score s at alignment position x reads:

$$s_x = \frac{\sum_{i=1}^{20} f_i S_{ij}}{N}$$

where i denotes amino acids, f_i is the frequency of amino acid i in alignment position x, S_{ij} is the substitution score of amino acids i and j, and N is the number of sequences in the alignment (Gribskov et al., 1987). The average profile score is appropriate for alignments with large N, but gives poor results for small N when f_i/N deviates from the expected probability p_i of finding residue i at position x. Therefore, it is advantageous to add a quantity proportional to the background probability of each amino acid to the real frequency f_i, yielding:

$$s_x = \frac{\sum_{i=1}^{20} (f_i + Aq_i) S_{ij}}{N + \sum_{i=1}^{20} Aq_i}$$

where the term Aq_i is called "pseudo-counts," with constant A as weight of the pseudo-counts (relative to f_i) and q_i as background frequency of residue i. There is a good theoretical justification for the use of pseudo-counts within the framework of Bayesian statistics, where they represent the prior information about the data (Durbin et al., 1998).

It must be stressed, however, that in the context of progressive multiple sequence alignment the application of pseudo-counts, and thus the incorporation of background amino acids frequencies, can well decrease proper alignment, notably during early steps of progressive alignment when sequence blocks to be aligned only contain a single or few sequences (J. Heringa, unpub. observ.). Strict Bayesian modeling treats model parameters for prior information as distributions rather than single values. Such distributions can be described by Dirichlet densities or mixtures. A Dirichlet mixture is a probability density over a set of probability vectors, in this case, vectors containing the probabilities of 20 amino acids as components, so that each vector describes a different probability distribution of the amino acids (Sjölander et al., 1996).

Another flexible motif search technique introduced by Bucher et al. (1996) uses "generalized profiles," which are similar to hidden Markov Models (HMMs). A profile is represented by the sequence alphabet and the possible states of an alignment that are defined as *begin, match, insert, deletion*, and *end*.

A consensus sequence represents the most reduced form of a profile, with each position having one component set to one (the consensus amino acid) and all others to zero. A straightforward way to construct a consensus sequence is to choose the most frequent or most likely residue at each alignment position. Although appealing because of its simplicity, a consensus sequence carries less information than an MSA (thus it is a degenerated representation), which may lead to misinterpretations in comparisons of the consensus sequence with related sequences (Schneider, 2002). A related but more sensitive way to compress the consensus information given by an MSA is through "partial order graphs" (Lee et al., 2002), which can be viewed as a formalism that provides multiple alternative consensus sequences for nonconserved MSA regions. A partial order graph of similar sequences contains a main "consensus" branch for conserved MSA segments and loops where sequences diverge from each other. Despite this condensed representation, the entire information of the MSA is retained. A new sequence to be aligned against the MSA will then be aligned to such nonconserved regions through the most similar sequence within the alignment.

Positional Conservation

Positional conservation is an important measure for detecting homology. Conserved alignment blocks are often described as "motifs," implicating structurally and/or functionally important parts of proteins. Frequently, even highly divergent sequence families share common motifs; sometimes such motifs are the only indication for sequence relatedness. Some databases are derived from grouping sequences with common alignment blocks or motifs into families, e.g., Blocks (*UNIT 2.2*) and FSSP.

A problematic aspect is the relation between positional conservation and sequence conservation. When sequences of high pairwise sequence identity are aligned, positional conservation scores are accordingly high, but mostly due to redundancy rather than true evolutionary conservation. In other words, sequences that are close in evolutionary time yield little information about true conservation patterns. This consideration has led to the usage of various weighting schemes: tree-based (Altschul et al., 1989; Thompson et al., 1994a,b), pairwise distance–based (Vingron and Argos, 1989; Sibbald and Argos, 1990; Vingron and Sibbald, 1993), and position-based (Henikoff and Henikoff, 1994).

Simultaneous Alignment

The dynamic programming algorithm for pairwise sequence alignment can be extended to multiple sequences (Murata et al., 1985; Gotoh, 1986) by using a multidimensional search matrix. However, the dimensionality of the search matrix is equal to the number of sequences and the search space equals the product of all sequence lengths, $O(L^N)$, where L is the average sequence length and N the number of sequences, rendering the search unfeasible even for moderately sized alignments. Approaches to reduce the computational load comprise reduction of the search space to near-diagonal paths (Carillo and Lipman, 1988; Wang and Jiang, 1994; Stoye et al., 1997), preselecting similar segments (Johnson and Doolittle, 1986), or word matching (Sobel and Martinez, 1986; Waterman, 1986; Vingron and Argos, 1989; Waterman and Jones, 1990).

A more recent development is mimicking simultaneous alignment by using precompiled profiles or libraries in progressive alignment. For each sequence, such a library contains information from other sequences and thus extends the sequence information used for each pairwise alignment. The idea behind multisequence profiles or libraries is to accumulate information from global and local alignments as well as from various sequence groupings. The accumulated information for each sequence is considered to be more reliable than single sequence alignments alone, so that match errors during progressive alignment are reduced.

The information for each sequence can be gathered by using pairwise alignments to construct a master-slave alignment. In the program PRALINE (Heringa, 1999, 2002), N master-slave alignments are constructed for N sequences, where each sequence in turn is the master sequence. The inclusion of slave sequences can be adjusted by a score threshold, so that sequences deemed too divergent are excluded and perturbation of the conservation pattern is avoided. The master-slave alignments are then converted into pre-profiles and used for the progressive construction of a final alignment. The advantage of this method is the ability to combine local and global alignment information. Moreover, sequences contained in multiple pre-profiles can be used to derive position-specific consistency scores, which effectively measure the agreement between the multiple alignment and pairwise alignments (see below).

A combination of local and global alignment is also achieved by the program T-Coffee (Notredame et al., 2000). Information from local and global pairwise alignments is complemented with information from triplet alignments that provide an alignment for each considered pair of sequences through each possible third sequence. For each pair of sequences, the contributions from the direct pairwise alignment and the triplet alignments are combined in a position-specific weight library (library extension), yielding a weight for each aligned residue pair. This library is then used to construct a final alignment by dynamic programming following the progressive alignment strategy.

Alignment Iteration

As mentioned in the above sections, the central aim of MSA methodology is to capture the complex evolutionary relationship between sequences and to convert the biological reality into a sensible scoring scheme. The ultimate step is to find the optimal mutual sequence arrangement by maximizing the alignment score. While strict (multidimensional) dynamic programming guarantees that the optimal (multiple) alignment score will be found, heuristic procedures such as progressive alignment do not necessarily yield the optimal score.

A class of techniques that are able to revisit and optimize an MSA is that of iterative multiple alignment techniques. Pioneered by Hogeweg and Hesper (1984), iterative techniques attempt to enhance the alignment quality by gleaning information from a multiple alignment constructed in an earlier round, which is then applied in a next round to improve the alignment according to a given scoring scheme. Iteration can be employed to further increase the alignment score, the incentive being to reach the optimum by "hill climbing," i.e., stepwise increase of the target function (alignment score) until convergence is reached. During iteration, the order at which the sequences are progressively aligned can be altered (Hogeweg and Hesper, 1984; Gotoh, 1996), or other criteria derived from a multiple alignment produced in the preceding round can be applied as an iterative scoring scheme (Heringa, 2002). This means that the target function of the iteration process can be different from the alignment score. In some cases it is desirable to maximize consistency, conservation, or some other function specific to the alignment problem.

Iteration is a reasonably efficient and robust technique that alleviates the greediness of the

progressive strategy. Results are critically dependent on the scoring scheme used, and often there is no certainty that convergence will be reached, and if so, that the converged multiple alignment will be biologically more optimal than earlier ones. The other two possible scenarios in addition to convergence are divergence, in which the program enters a route through a virtually infinite number of states, and limit cycle, in which the program recursively visits a finite number of states. In cases where a different target function than the alignment score has been used to guide iteration, a decision has to be made whether the last scoring alignment (with the maximal target function value) or the highest scoring alignment will be taken as the result after convergence has been reached. A choice between several solutions also exists in the outcome of the limit cycle and divergence scenarios. It is the task of the investigator to perform alignment iteration with intuition and knowledge, in order to choose the right combination of target functions, alignment strategies, and iterations, to gain information about the sequence set under consideration.

Probabilistic MSA

Generation of an MSA can be performed entirely in a probabilistic framework. The probability of observing a certain sequence can be inferred using a hidden Markov model (HMM). An HMM consists of character-emitting states and transitions between these states. Character emissions and state transitions are connected to probabilities (the probabilistic model) that determine the behavior of the HMM. To create a sequence, an HMM generates a Markovian path through states, i.e., each step, including character emission and transition, is independent of the previous step. Given a sequence, the probability of observing this specific sequence can be derived using the Viterbi algorithm (Viterbi, 1967). Pairwise alignment algorithms for HMMs have been described (Durbin et al., 1998). For progressive MSA, the pairwise approach is extended to include the phylogenetic inter-dependence of sequences (*http://evol-linux1.ulb.ac.be/ueg/ProAlign/Loytynoja_and_Milinkovitch_2003.pdf*). Probabilistic modeling of sequence alignments is theoretically and computationally involved, and was in the past largely restricted to specialists. However, recent improvements in program development and computer speed have made this approach more accessible and the quality of probabilistic alignments for nucleotide sequences is now comparable to that of standard alignment methods.

Another probabilistic modelling approach is Bayesian inference (Liu and Lawrence, 1999), where all unknown variables are treated as probability distributions. The advantage of Bayesian modeling is that it allows for inclusion of prior knowledge about the system. However, the computational burden can be prohibitive, and specific sampling techniques such as "general Markov chain Monte Carlo" may be required (Tanner and Wong, 1987). A collection of algorithms for Bayesian alignment has been described by Zhu et al. (1998).

Parallelization of MSA

With the increasing availability of computer clusters in computationally oriented laboratories, parallelization of the most time-consuming computational tasks is worth considering. Highly repetitive procedures, such as the pairwise sequence or profile alignment phase in progressive alignment, are favorable targets for parallelized (or distributed) computing. Parallelized programs are designed to split the total computational task into subtasks that are processed on separate CPUs (nodes). Implementation of parallelized code typically requires one to identify the most CPU-intensive task (frequently a loop structure) and to split it into subtasks for independent execution. For example, an MSA of 4 sequences requires 6 pairwise alignments (subtasks), which can be performed, for example, in blocks of 3 on 2 nodes (CPUs), or in blocks of 2 on 3 nodes.

The technical details of parallelization are dealt with by high-level routines provided by a parallelization interface such as the "message passing interface" package MPICH, available at *http://www-unix.mcs.anl.gov/mpi/mpich/* (Gropp et al., 1996; Pacheco, 1997). If all nodes execute the same operations but perform these on different subsets of distributed data, the parallelization technology is called single-instruction multiple-data (SIMD). Parallel code is most efficient with a minimum amount of communication between the nodes and with optimal balancing of the computational load over the CPUs.

The MSA program PRALINE (Heringa, 1999, 2002) has been parallelized in the form of SIMD technology (Kleinjung et al., 2002). The scaling of computational times versus the number of employed nodes is plotted in Figure 3.7.5 for three differently sized sets of sequences. Parallelized PRALINE generated an MSA up to ten times more quickly than the

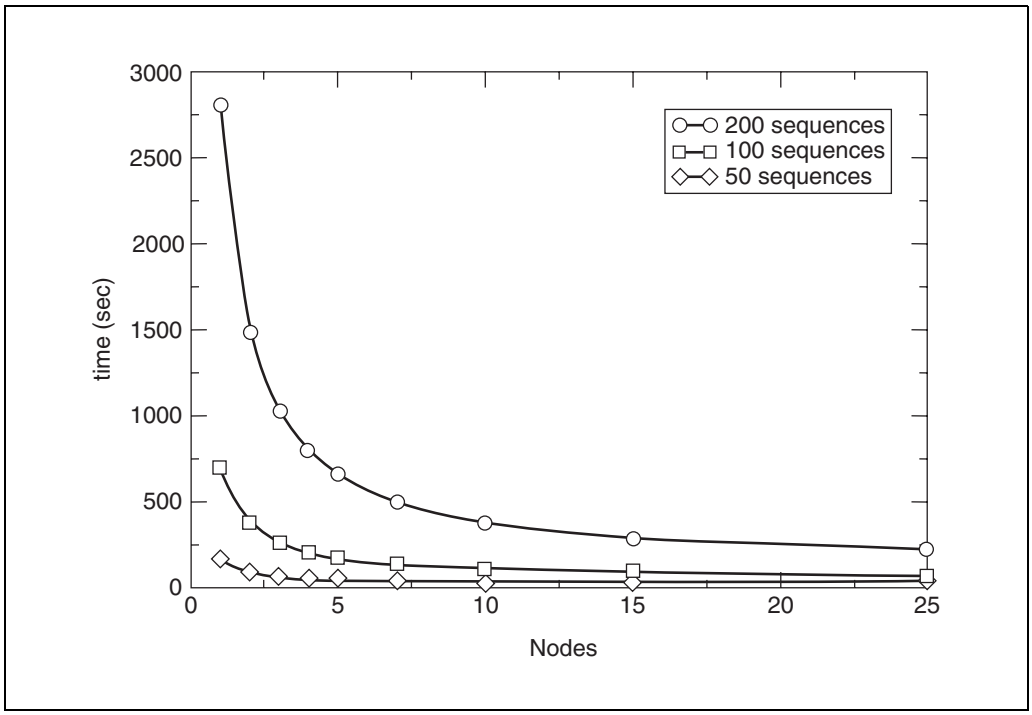

Figure 3.7.5 Computational times of parallelized PRALINE on different numbers of nodes for three sets of 200, 100 and 50 sequences, each 200 residues long.

single-processor version, when tested on a set of 200 random sequences of 200 residues length.

MSA METHODS

MSA is an intricate problem and over the past 30 years an increasing number of approaches have been developed that try to solve it, each with their own strengths and weaknesses. Unfortunately, the diversity of the methodologies makes it difficult for nonspecialists to know which method is the best to use for a particular problem. A sensible decision can be made with a clear and thorough understanding of how the methods work, where they perform well, and what their limitations are. The methods described below are a compilation of the best characterized methods to date, and useful guidelines are proposed based on published assessments, where available. A summary of the properties and availability of the various methods, along with literature references, can be found in Table 3.7.1.

BioPat

BioPat is a mathematical package that includes the first-ever integrated MSA method, which is a global progressive algorithm with iteration capabilities (Hogeweg and Hesper, 1984). Initially, a coalescence tree (dendrogram) is constructed based on all pairwise similarities of the sequences to be aligned, matched by dynamic programming (Needleman and Wunsch, 1970). The method provides choices among many of the commonly used clustering techniques to build the dendrogram, such as Unweighted Pair-Group Mean Average (UPGMA; Sneath and Sokal, 1973), the present-day ancestor method (Blanken et al., 1982), or the Neighbor-Joining (NJ) method (Saitou and Nei, 1987; *UNIT 6.3*). Once the dendrogram is completed, the sequences are progressively aligned following the branch order of the dendrogram. The resulting alignment is then used to infer the associated new pairwise similarities and the initial dendrogram is updated to produce a new alignment. A new dendrogram is then constructed iteratively, from which a succeeding alignment is created based on the increased information.

MultAlin

The method MultAlin (Corpet, 1988) follows the Hogeweg and Hesper (1984) approach in that it also uses hierarchical clustering for constructing a guide dendrogram and iteration. It is different in that for the alignment of two sets of sequences it uses the average similarity score between a pair of alignment columns i and j, one from each set, which is the average over the amino acid exchange values associated with all pairwise intercolumn residue compari-

Table 3.7.1 MSA Methods

Method	Algorithm	Online availability	Reference
BioPat	Iterative/progressive	BioPat software package: *http://www.biopat.de*	Hogeweg and Hesper (1984)
Clustal	Progressive	*http://www.ebi.ac.uk/clustalw/*	Thompson et al. (1994a, 1997); *UNIT 2.3*
DCA	Exact (MSA based)	*http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/dca_submit*	Stoye (1998)
Dialign2	Consistency-based	*http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/dialign_submit*	Morgenstern et al. (1999)
ITERALIGN	Iterative	*http://giotto.stanford.edu/~luciano/iteralign.html*	Brocchieri and Karlin (1998)
MACAW	Iterative/Progressive	*http://www.bris.ac.uk/Depts/PathAndMicro/services/CGR/Common%20Files/MACAWinstructions.htm*	Schuler et al. (1991)
MAFFT	Consistency-based	*http://www.biophys.kyoto-u.ac.jp/~katoh/programs/align/mafft/*	Katoh et al. (2002)
Match-Box	Progressive	*http://www.sciences.fundp.ac.be/biologie/bms/matchbox_submit.shtml*	Deperieux and Feytmans (1992)
MEME	Consistency-based	*http://meme.sdsc.edu/meme/website/meme.html*	Bailey and Elkan (1994); *UNIT 2.4*
MSA	Exact	*http://xylian.igh.cnrs.fr/msa/msa.html*	Lipman et al. (1989)
MULTAL	Progressive	*http://mathbio.nimr.mrc.ac.uk/ftp/wtaylor/multal/*	Taylor (1988)
MultAlign	Progressive	*http://cbrg.inf.ethz.ch/Server/MultAlign.html*	Barton and Sternberg (1987)
MultAlin	Progressive	*http://prodes.toulouse.inra.fr/multalin/multalin.html*	Corpet (1988)
MUMmer	Suffix Tree	*http://www.tigr.org/software/mummer/*	Delcher et al. (2002); *UNIT 10.3*
OMA	Iterative DCA	*http://bibiserv.techfak.uni-bielefeld.de/oma/*	Reinert et al. (2000)
PileUp	Progressive	GCG software package: *http://www.accelrys.com*	GCG (1993); *UNIT 3.6*
POA	Partial order	*http://www.bioinformatics.ucla.edu/poa/*	Lee et al. (2002)
PRALINE	Iterative/progressive	*http://www.cs.vu.nl/~ibivu/programs/pralinewww/*	Heringa (1999, 2002)
Prrp	Iterative/Stochastic	*ftp://ftp.genome.ad.jp/pub/db/hgc/software/saitama-cc/*	Gotoh (1996)
SAGA	Iterative/Stochastic/GA	*http://igs-server.cnrs-mrs.fr/~cnotred/*	Notredame and Higgins (1996)
T-Coffee	Consistency-based/ Progressive	*http://igs-server.cnrs-mrs.fr/Tcoffee/*	Notredame et al. (2000)
TNB	Consistency-based	*http://globin.cse.psu.edu/ftp/dist/TNB/*	Boguski et al. (1992)

sons. This way of scoring alignment positions is effectively follows the profile comparison technique (Gribskov et al., 1987).

MULTAL

The early method MULTAL (Taylor, 1988) is very fast and constructs a dendrogram during the progressive alignment, as in the method of Feng and Doolittle (1987). It uses a fast sequential branching method to align the closest pairs of sequences first and then subsequently align the next closest sequences to those already aligned. The order in which the sequences are aligned is largely based on the global amino acid composition of the sequences, which saves the fixed cost of performing all-against-all pairwise alignments. Blocks of aligned sequences are scored by dynamic programming similar to the method MultAlin (see above), but the similarity of two alignment columns is additionally normalized by the minimum number of sequences in either of two compared alignment blocks.

MultAlign

The global progressive method MultAlign (Barton and Sternberg, 1987) establishes a simple chain order in which the individual sequences are aligned one by one. Initially, all pairwise alignment scores are determined and the two most similar sequences are matched first. During further iterations, the sequence showing the highest alignment score when matched with the prealigned sequence block is added to it. To determine the alignment score, each sequence position *i*, of the *k*th sequence matched with position *j* of a pre-aligned block of $k-1$ sequences receives a score per matched position averaged over the corresponding residue substitution values. The PAM-250 substitution matrix (Dayhoff et al., 1983) is utilized, with a constant of 8 added to remove all negative matrix elements. Matched gaps are evaluated by the lowest exchange weight of zero.

The MultAlign method incorporates iteration capabilities in that the resulting MSA can be progressively refined by realigning each sequence with the previous alignment from which that sequence is deleted: i.e., sequence A1 is matched with aligned sequences A2...A*N*; sequence A2 is then realigned with the alignment of A1, A3...A*N*, and so forth. This process is repeated until all *N* sequences are realigned. Barton and Sternberg (1987) recommend two such complete refinement cycles.

The quality of an MSA is assessed by comparisons with alignment scores over randomized sequences if the sequence groups are not too large; otherwise, normalized alignment scores (NAS) are used, which is the alignment score divided by either the length of the shorter matched sequence or the number of residues not aligned with gaps.

ClustalW, ClustalX

ClustalW (Thompson et al., 1994a; *UNIT 2.3*) and the later window graphic user interface (GUI) version ClustalX (Thompson et al., 1997) are the newest versions of the global progressive alignment algorithm Clustal (Higgins and Sharp, 1988), and are generally considered as the standard method for MSA. The progressive strategy used is a simplification of the original Feng and Doolittle (1987) scheme. The alignment is constructed by first building a guide dendrogram using Neighbor-Joining (NJ; *UNIT 6.3*; previous versions used the UPGMA strategy), based on sequence similarity, which is subsequently used to order successive pairwise alignments. The already aligned sequences are reduced to a profile for the subsequent pairwise alignment (previous versions used position consistencies). However, during the progressive alignment process, highly specialized heuristics are applied to try and optimize how the sequence information is processed. When the sequences are ordered for alignment according to the precomputed dendrogram, the alignment of distantly related sequences is delayed, thus overriding the dendrogram. This is implemented to correct for the limitation of progressive alignment, which does not allow alterations of the alignment once a sequence has been aligned even if later-added sequences may require it—"once a gap always a gap" (Feng and Doolittle, 1987). Also the pairwise alignments are performed using local gap penalties and there is automatic selection and adjustment of the residue substitution matrix and gap penalties, respectively.

ClustalW and ClustalX (*UNIT 2.3*) perform best when the sequences to be aligned are global cases and have no obvious outlier. The evolutionary distance between the sequences must be relatively low, thus producing a dense dendrogram. Also, the penalty scheme used by ClustalW/X discriminates against long insertions and deletions (indels) and will, therefore, exhibit reduced accuracy in such cases. In multidomain-alignment cases its accuracy is greatly reduced compared to more recent global or local algorithms (Lassmann and Sonnhammer, 2002) such as POA (Lee et al., 2002), Dialign2 (Morgenstern, 1999), and T-Coffee (Notredame et al., 2000).

The algorithm is reasonably fast and can handle sizeable sets of sequences. However, its speed decreases when it is given very large sets of data, such as genomic data, and the overall performance is less accurate when compared to the other available methods (Lassmann and Sonnhammer, 2002) such as POA (Lee et al., 2002).

MSA

The simultaneous alignment algorithm MSA (Lipman et al., 1989) employs multidimensional dynamic programming. Note that MSA here denotes the name of the Lipman et al. algorithm rather than the abbreviation for multiple sequence alignment used throughout this unit. To reduce computations, the MSA method employs the Carillo and Lipman (1988) approach, which estimates, using pairwise alignments, how much around the *N*-dimensional search matrix diagonal needs to be searched, where *N* is the number of sequences to be aligned. The Carillo and Lipman method generalizes the earlier pairwise diagonal strip method of Ficket (1984) to *N* dimensions. Although this approximation of simultaneous alignment optimizes the sum-of-pairs score, which in principle is much more accurate and error-free than progressive methods using the same optimization, it has huge limitations in how many sequences it can simultaneously align due to its excessive memory and computational requirements. Up to 10 sequences of 200 to 300 residues in length can be aligned with the MSA method. The method addresses an additional problem in the comparison of multiple sequences, which is the weighting of the aligned sequences, as similar sequences should not dominate the final alignment. Lipman et al. (1989) used the weighting scheme suggested by Altschul et al. (1989) based on phylogenetic trees. More recently, the MSA method was extended to larger data sets using a divide-and-conquer strategy (Stoye et al., 1997) implemented in the method DCA (see below).

DCA, OMA

The DCA (Divide-and-Conquer MSA) method (Stoye, 1998) is an exact divide-and-conquer (Stoye et al., 1997) alignment algorithm. DCA follows the same strategy as the MSA algorithm by Lipman et al. (1989) and performs simultaneous MSA instead of the progressive approach (Feng and Doolittle, 1987). The DCA approach is an attempt to overcome the computational complexity of the MSA method (Lipman et al., 1989). The divide-and-conquer strategy first selects the longest sequence in the set to be aligned and cuts it near its midpoint. The rest of the sequences are also cut at suitable positions, which are calculated through a heuristic method to reduce computational time, and consequently two new subsequence sets arise. This can then be repeated on the subsequence sets until a certain predefined minimum threshold for subsequence length is reached. The smaller the threshold value setting, the faster, but less optimal, the alignment becomes. The now shorter sets of subsequences can then be separately aligned using the MSA algorithm, thus decreasing the time and memory requirements. At the end, all the subalignments are concatenated to produce the full final alignment.

DCA represents an improvement in accuracy and speed with respect to MSA, but computational time is still extremely sensitive to sequence distance and length (Stoye, 1998) so that the number of sequences that can be aligned still remains very low. The DCA algorithm has also been implemented as an iterative scheme called OMA (Optimal MSA; Reinert et al., 2000). The OMA method represents an improvement with respect to speed and accuracy by adding face bounding, gray code enumeration, sequence weighting, realignment of cut positions, and parallelization techniques. The OMA protocol initiates a DCA alignment using a very low sequence length threshold that is only calculated once. Using this calculated threshold, a new larger threshold is produced at each iteration. This means that the alignment becomes slower at each subsequent iteration but also more optimal. The user can set the number of iterations to get a compromise between alignment speed and quality. Although OMA shows many improvements in memory usage and accuracy (Reinert et al., 2000), it is still very computationally demanding for average systems and cannot handle large data sets.

Dialign

Dialign (Morgenstern et al., 1996) is a local consistency-based alignment algorithm, which, instead of aligning single residues, aligns whole sequence segments. These segments can be envisaged as diagonals, as they would appear on a dot plot of a dot matrix analysis (Fitch, 1966; Gibbs and McIntyre, 1970). The most recent version, Dialign2 (Morgenstern, 1999) initially performs all pairwise

alignments of the sequences to be aligned, after which all ungapped segments (diagonals) are identified. Consistent sets of diagonals are then determined and added sequentially to the alignment using an iterative mathematical procedure that determines the optimal order of addition. Only sequence fragments for which matched segments are found are aligned; regions in-between blocks of similar segments are left unaligned. The improvement of Dialign2 compared to Dialign1 is the alteration of the original weighting of diagonals, which was previously based on Altschul and Erickson (1986). The Dialign2 algorithm is both an accuracy and computational time improvement over the original method.

Morgenstern (1999) reported that Dialign2 outperforms many local and global algorithms in identifying related motifs, such as ITERALIGN (Brocchieri and Karlin, 1998), ClustalW (Thompson et al., 1994a,b), MultAlin (Corpet, 1988), DCA (Stoye, 1998) and Match-Box (Deperieux and Feytmans, 1992). Dialign2 has also been shown to perform well in both local and global cases of varying evolutionary distance and, more recently, in multidomain cases (Lassmann and Sonnhammer, 2002) against T-Coffee (Notredame et al., 2000), POA (Lee et al., 2002) and ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*).

MEME

The program MEME (Bailey and Elkan, 1994, 1995; *UNIT 2.4*) is a tool for unsupervised motif searching within DNA and protein sequences, which operates using an expectation maximization (EM) algorithm. It finds occurrences of motifs by comparing the residue composition at each position of a putative motif against the general composition of background sequence regions that do not display the motif. Regions showing the most discriminating compositions are then selected as motifs. A limitation of the MEME motifs is that they are ungapped, but the program can find multiple occurrences in individual sequences, which on the other hand do not need to be encountered within each input sequence. Another useful feature of the MEME method (also see *UNIT 2.4*) is that it is geared towards finding DNA palindrome sequences, which are often implicated as DNA-binding sites for proteins. To increase the chance of finding palindrome sets, the nucleotide probabilities of corresponding motif columns (column 1 and W, 2 and $W - 1$, and so forth, with W the width of the motif) are constrained to be the same.

ITERALIGN

The program ITERALIGN (Brocchieri and Karlin, 1998) is a local iterative algorithm that optimizes the consistency between local pairwise alignments and their embedding in an MSA across all input sequences. It first aligns all ungapped regions of significant local similarity. The high-scoring regions are then iteratively replaced by a consensus, based on the distance between them. The existing consensus set is used as input to the next round, until convergence is reached. Core blocks are extracted and optimized using local dynamic programming to further enhance the result. Finally, these blocks are linked to produce the final alignment. Each of these aligned blocks can then be studied independently as a potential functional/structural unit. The authors went so far as to edit individual sequences by replacing amino acids with those that are preponderant at a corresponding position in an MSA, to achieve better recognition of crucial alignment regions.

MACAW

Schuler et al. (1991) introduced the iterative local progressive alignment algorithm MACAW (MSA Construction and Analysis Workbench), which allows the user to lock or shift regions in an alignment, while nonlocked subsequences are aligned automatically. The method is semiautomatic and produces blocks of alignments shared by all or a subset of the sequences. It is possible to iteratively define conserved regions such that the fraction of poorly defined segments which must be aligned automatically become fewer at each iteration cycle. The GIBBS method of Lawrence et al. (1993) has been incorporated in the MACAW procedure to detect the local fragments.

Match-Box

The method Match-Box (Deperieux and Feytmans, 1992) aims to find ungapped sequence regions with a high degree of similarity across a set of input sequences. This is achieved by comparing the frequency distribution of all pairwise aligned sequence fragments (which are gathered from global alignments), with that derived from shuffled sequences. Using a set of the most similar nine-residue fragments, local alignments are created for each fragment, if similarity beyond a threshold is found with segments across all other sequences. Boxes of ungapped regions are then delineated from these local alignments and assembled in a final alignment with unaligned amino acids and gaps in between the boxes. The method also gener-

ates a reliability index for the aligned positions within the boxes, which relies on statistics derived from analyzing a relatively small number of known family alignments.

PileUp

The GCG package alignment program PileUp (GCG, 1993; *UNIT 3.6*) is a global progressive alignment algorithm. It creates an MSA using a simplification of the progressive alignment method of Feng and Doolittle (1987). It generates an UPGMA-based dendrogram and, for the alignment of two sets of matched sequences, uses the average alignment similarity score of Corpet (1988).

PileUp is limited to 500 sequences, with any single sequence in the final alignment restricted to a maximum length of 7000 characters. If longer sequences are included in the alignment, the number of sequences PileUp can align decreases.

Prrp

Prrp (Gotoh, 1996) is a global iterative stochastic alignment algorithm. This algorithm is a double-nested strategy for MSA optimization. In the inner iteration, the sequences are divided into two groups and subsequently realigned using a global group-to-group alignment algorithm. When the inner iteration converges, new pairwise sequence weights are derived from a dendrogram constructed with the UPGMA cluster criterion and used to calculate the alignment scores when sequence blocks are matched. When these weights converge, the outer iteration stops.

Gotoh (1996) reported improved accuracy when compared to ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*). These results were confirmed using JOY, a database of structural alignment (Thompson et al., 1999b) and later on BAliBASE (Thompson et al., 1999a) in the assessment of T-Coffee (Notredame et al., 2000).

POA

The Partial Order Alignment (POA; Lee et al., 2002) is an extension of the conventional dynamic programming approach. Instead of performing pairwise alignments following a specific order (from a guide tree), sequences are aligned in the order in which they are given. The growing MSA is represented by a "partial order graph," in which identical residues within a column are fused and the information of the sequence origin is stored. Thus, despite the condensed representation, all of the information of the MSA is retained. A typical PO-MSA of similar sequences contains a main "consensus" branch and loops where sequences diverge from each other. The POA dynamic programming matrix reflects this structure by adopting the bifurcation points, so that the matrix consists of multiple two-dimensional layers that part and rejoin according to the PO-MSA graph. The best alignment is found by a conventional trace-back operation. The POA algorithm guarantees that each sequence is aligned to the closest sequence in the growing MSA.

POA is a novel local progressive algorithm. The novel feature of this method is that it employs partially ordered graphs to represent aligned sequences instead of profiles (see Profiles, above). The progressive strategy for this method does not follow a guide dendrogram to determine the order in which the sequences will be aligned, but aligns the input sequences in the order in which they are given. Each time a new sequence is added to the growing alignment, it is aligned with the most closely related hybrid sequence within the MSA as given by the partial order graph. Pairwise alignments are performed using the Smith-Waterman algorithm (Smith and Waterman, 1981), which is extended to accommodate the partial order graph representation. The partial order graph is constructed in two main steps: first, the sequences are converted to PO-MSA (Partial Order-MSA) data structures. Next, the most closely related pair of PO-MSAs are aligned and the identical residues are fused into nodes (like the knots on two ropes tied together at points along their length), while the remaining residue origins and positions are recorded and considered as incoming and outgoing (directed) edges from each node (the rope "bubble" before and after each knot; see Fig. 3.7.6). When the partial order graph is then aligned to the next PO-MSA, aligned identical residues are fused regardless of whether they are nodes or edges, and aligned nonidentical residues are recorded as aligned. Finally, any edges connecting the same pair of nodes are removed.

POA combines accuracy and speed (Lassmann and Sonnhammer, 2002). It performs marginally worse than the global method T-Coffee (Notredame et al., 2000) and local method Dialign2 (Morgenstern, 1999) in sequences of varying evolutionary distance. In multidomain alignment cases it performs comparably to Dialign2 and better than the global methods T-Coffee and ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*). Finally, it is by far the fastest method to date and can handle very large data sets, although its accuracy in these cases,

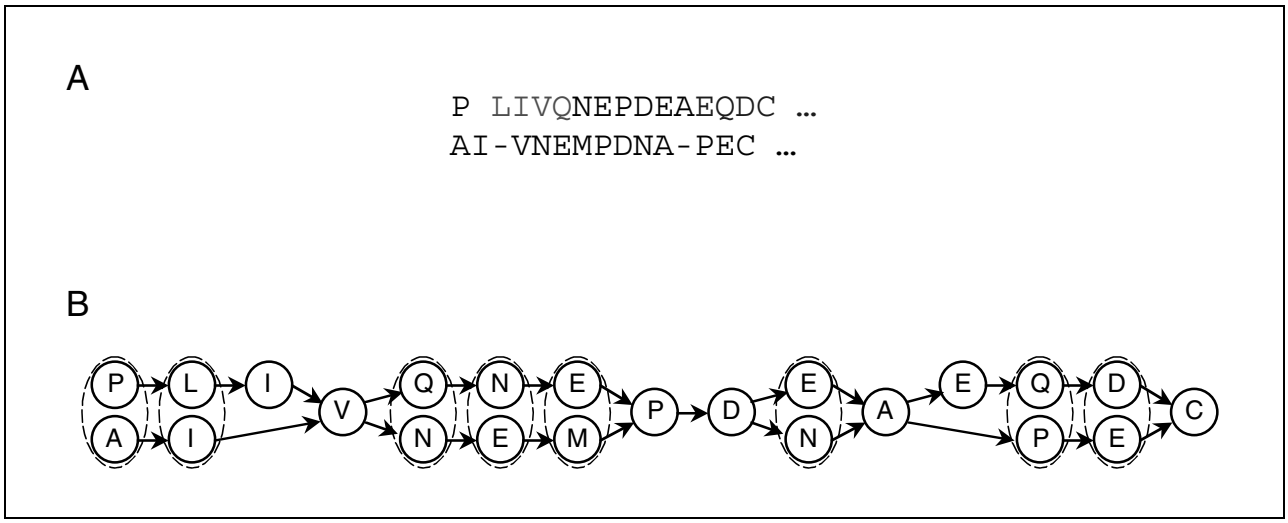

Figure 3.7.6 The Partial Order Graph (POA) alignment representation of the C-termini of a pair of flavodoxin proteins. (**A**) The alignment in standard format; (**B**) The alignment in POA representation (adapted from Lee et al., 2002).

particularly since the order in which the sequences are treated is not optimized, has not yet been determined.

PRALINE

PRALINE (PRofile ALIgNmEnt; Heringa, 1999) is a global progressive algorithm. Its distinctive feature is that it integrates many strategies for alignment optimization, which will be described below. The progressive alignment strategy does not use a precalculated search dendrogram but performs, at each alignment step, a full profile search with the most recently aligned sequence block. It, therefore, re-evaluates, at each alignment step, which sequences or blocks of sequences should be aligned, and hence determines the alignment order during progressive alignment. The pairwise alignments are performed using dynamic programming. PRALINE offers a number of strategies to optimize the quality of MSA, such as local global alignment and global and local profile preprocessing, and has weighted iteration capabilities. In addition, it can integrate secondary structure to guide the resulting alignment.

PRALINE is more of a tool kit than a one-step MSA method. It allows the user to apply or combine different strategies to a given problem and find the best solution, rather than applying a single approach to everything. The *local global optimization* strategy performs a local alignment of the sequences first, and then, using that information, performs a final global alignment. The *profile preprocessing* strategy is aimed at incorporating into each sequence the most trusted information from other sequences, either through global or local alignments. For each sequence, a preprocessed alignment is created including only those sequences that score beyond a user-specified threshold when aligned pairwise with the sequence considered. A low threshold would result in a preprocessed alignment for each sequence comprising all other sequences, while higher thresholds would allow fewer and fewer sequences into the alignment. For each of the formed preprocessed alignments, a profile is constructed. PRALINE then performs progressive MSA using the preprocessed profiles to represent each of the original sequences. Because the preprocessed profiles for each of the sequences incorporate knowledge about other sequences (in particular similar sequences) and comprise position-specific gap penalties, they enable increased matching of distant sequences and likely placement of gaps outside ungapped core regions during progressive alignment (Fig. 3.7.7). The MSA of the preprocessed profiles can also be used to derive *consistency scores* for each amino acid in the alignment, which for each sequence reflect the consistency among the pairwise alignments used that include the sequence. The consistency of preprocessed profiles can also be used to optimize the alignment through iteration. Since each of the preprocessed profiles can contain information about all the sequences, each sequence in the final alignment can be assessed in terms of the degree of consistency reached across the profiles, which is translated into a consistency score for each amino acid in the MSA. Iteration is then guided by these obtained scores, which are used as weights in the construction of align-

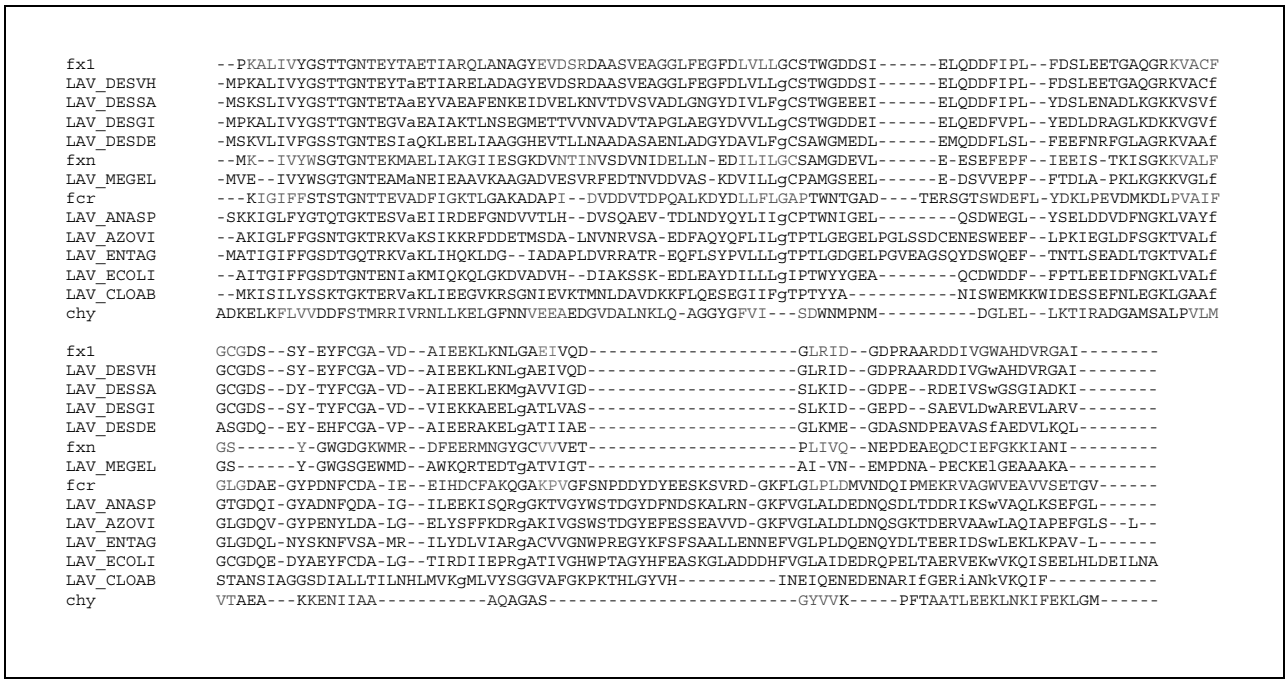

Figure 3.7.7 An MSA of the flavodoxin family members (13 proteins) created by PRALINE (Heringa, 1999) using local preprocessing with a threshold of 300. The bottom sequence is the cheY sequence (PDB code 3chy), which is an outlier (Fig. 3.7.4), with very low sequence similarity but the same basic flavodoxin fold. Note that the PRALINE alignment shows reasonably matched secondary structure elements, also for the cheY sequence.

ments (using the dynamic programming protocol) during the next MSA step (Heringa, 2002). From the resulting set of iterative alignments, the one with the highest cumulative score over all pairwise matched amino acids in the alignment (sum-of-pairs score) can be selected as a safeguard to prevent alignments from wandering away to less optimal areas in the alignment space (Heringa, 2002).

The secondary structure incorporation proceeds by initially constructing an MSA without information about the corresponding secondary structure. Then, for each sequence, the secondary structure is predicted by the PREDATOR (Frishman and Argos, 1996, 1997) or the PHD method (Rost and Sander, 1993), although in principle this could be done by any available method, and iteratively a new alignment is constructed, now using the predicted secondary structure. PRALINE employs dynamic programming to progressively construct an MSA for a query set of sequences, and therefore relies on an amino acid exchange weights matrix and a pair of gap penalties. The initial alignment is constructed using a default residue exchange matrix (e.g., the BLOSUM62 matrix; *UNIT 3.5*) and gap penalties. After secondary structure prediction, resulting in a tentative secondary structure for each sequence or in a single secondary structure when using a single sequence-based or an MSA-reliant method, respectively, PRALINE utilizes the obtained secondary-structure information. During the progressive alignment, pairs of sequences (and/or profiles representing already aligned sequence blocks) are matched using three secondary structure-specific residue exchange matrices (Lüthy et al., 1994) and associated gap penalties. The residue exchange weights for matched sequence positions with identical secondary structure states are taken from the corresponding residue exchange matrix; matched sequence positions with nonidentical secondary-structure states are assigned the corresponding value from the default exchange matrix—e.g., the BLOSUM62 matrix (*UNIT 3.5*). The secondary-structure information is used in a conservative manner, based upon the assumption that consistent secondary structure predictions are indicative of their reliability. Secondary-structure prediction can also be used to optimize the alignments in an iterative fashion. Most reliable secondary-structure prediction methods utilize sequence information in MSAs, and their prediction accuracy relies on the quality of the MSA used. In the PRALINE approach, the MSA is guided by predicted secondary structure, so that an iterative scheme arises that optimizes both the quality of the MSA and that of the secondary-structure prediction. In this iterative scenario, each iteration proceeds in the same way as described above,

producing an MSA that is passed on to the next iteration and guides secondary-structure prediction, which in turn guides alignment and so on.

A future development for incorporating the secondary structure into the creation of an MSA is the use of an optimally segmented secondary-structure consensus for each sequence to be aligned. The optimization step is performed using weighted dynamic programming and can involve the production of a consensus from a single or multiple prediction methods. The optimal segmentation works based on the fact that current state-of-the-art predictors process alignments prior to prediction, causing inaccurate predictions in regions where alignment blocks have been merged. Each consensus produces a structure-based score giving the prediction a quality measure, which can be used in combination with an equivalent residue-based score to give the whole alignment an overall quality measure. In the PRALINE iterative scenario, the structure-based score can be used to select the best-scoring secondary structure predictions for each sequence, from all iteration cycles, which can then be used to perform a separate final structure-guided alignment. Also, the alignment produced at each iteration can be assessed according to combined structure- and residue-based scores, enabling improved quality evaluation.

PRALINE is a very diverse package. When used as a tool kit it can overall outperform both of the currently popular methods ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*) and T-Coffee (Notredame et al., 2000) in varying evolutionary distance and multidomain cases. However, this requires time, because many strategies can be tried until the best solution is found. In addition, the use of these strategies makes PRALINE relatively slow. However, PRALINE has been parallelized (Kleinjung et al., 2002), yielding a ten-fold acceleration compared to single-processor execution (see Parallelization of MSA, above).

SAGA

The program SAGA (Sequence Alignment by Genetic Algorithm; Notredame and Higgins, 1996) is an iterative stochastic alignment method that uses a genetic algorithm (GA; Goldberg, 1989) to select the alignment from an evolving alignment population and which optimizes, as an Objective Function (OF), the weighted sum of pairs as used in the MSA program. The algorithm initially generates a random population of alignments of the sequences, called generation zero (G_0). Offspring alignments are then generated from the parent alignments in G_0 that are evaluated for fitness based on alignment quality. The better the alignment, the more offspring alignments it creates. The operators for offspring alignment creation can be either the mixing of the contents of the parent alignments (crossovers) or the alteration of a single parent (mutation). This process is iterated through successive generations, allowing only the fittest (best-quality) offspring alignments to proceed to the next generation and produce their own offspring alignments. The iteration process stops when no more improvement can be achieved.

SAGA was found to produce overall better scoring alignments when compared to MSA (Lipman et al., 1989) and ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*), and, although optimal alignments could be performed for sets of over 30 sequences, the processing time was extremely high. More recently, a measure of consistency between the considered MSA and a corresponding library of Clustal pairwise alignments was taken. This OF was developed for the Coffee algorithm (Notredame et al., 1998).

T-Coffee

T-Coffee (Tree-based Consistency Objective Function For alignmEnt Evaluation; Notredame et al., 2000) is a global progressive consistency-based algorithm. Initially, all pairwise alignments of the sequences are performed twice: once with the global alignment method ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*) where a single global alignment is generated, and once with the local alignment method Lalign (Huang and Miller, 1991) where 10 top-scoring nonintersecting local alignments are generated. The results are pooled into a primary library of combined weights for each nonredundant residue pair. The combined weight for each residue pair (x, y) corresponds to the sum (Σ) of scores (S) of the global and local alignments containing that residue pair. Each alignment score (S) is the percentage sequence identity of that alignment. A library extension step is then performed using a procedure called *matrix extension* (Notredame et al., 2000) to measure how residue pairs align with respect to other residues in the library, producing triplet weights. These triplets are then used to assess how well sequences are aligned compared to the other sequences in the data set, rather than looking at pairs of sequences in isolation. The final alignment is built by performing the library extension step to

produce a guide dendrogram, which then orders how the sequences are aligned.

The assessment of Lassmann and Sonnhammer (2002) show that T-Coffee is one of the most reliable MSA methods to date in cases of low to moderate evolutionary distance, with higher accuracy compared to ClustalW (Thompson et al., 1994a,b; *UNIT 2.3*), Prrp (Gotoh, 1996), Dialign2 (Morgenstern, 1999), and POA (Lee et al., 2002). In multidomain cases, it shows good performance as a global alignment method and is only outperformed by local alignment strategies.

However, T-Coffee has speed and computational demand limitations when alignments (>30 sequences) of large sequences (>10,000 residues) are performed and may even fail to complete them on average-powered systems.

TNB

The Boguski et al. (1992) semimanual program suite TNB combines the space-efficient local alignment routine SIM (Huang et al. 1990) and the method MSA (Lipman et al., 1989). The pairwise alignment strategy starts by identifying the highest-scoring gap-containing local alignments. From these, the nongapped regions occurring in each of the sequences are extracted. The adjacent blocks of such motifs are then aligned with the intervening sequence fragments using the MSA method, thus allowing gaps. This method also provides a user interface through which parts of the alignment can be manually edited.

MUMmer

Genome-wide sequence alignments require very fast algorithms that can handle millions of nucleotides. The alignment system MUMmer (Delcher et al., 1999, 2002; *UNIT 10.3*) uses "suffix trees," which allow for an alignment of two entire genomes in linear time and space. The program finds "maximal unique matches" (MUMs) between two input sequences. A suffix tree is a unique character string where the sequences are identical. A new branch is created where they differ. MUMmer creates a suffix tree based on one (reference) sequence and streams the second (query) sequence against it. MUMmer has been used to assemble contigs from shotgun-sequencing to construct the complete genome.

MAFFT

The MAFFT program (Katoh et al., 2002) is based on the fast Fourier transform (FFT) for rapid detection of homologous segments. Amino acids are represented by volume and polarity values, yielding signal peaks if homologous segments are aligned. The recorded segments are joined to a final alignment by dynamic programming.

ASSESSMENT OF MSA

In this section, MSA benchmarking issues, as well as the scoring schemes currently in use to evaluate MSA quality using reference alignments, are discussed. As high alignment scores do not necessarily entail a good biological quality, MSA score optimization is also briefly discussed.

Scoring MSAs

The score of an MSA is the target function that is optimized by the (progressive) alignment algorithm. Most programs use a Dayhoff-type substitution matrix as evolutionary model and compute the sum-of-pair (SP) score of all $n(n-1)/2$ combinations of aligned residue pairs (Lipman et al., 1989). Since pairwise alignment algorithms optimize the alignment score based on residue exchange scores and gap penalties, an obvious way of scoring MSAs is to extend the pairwise sequence scores to MSAs. This is referred to as the SP score for alignment: for each amino acid $a_{i,j}$ in sequence i and at position j in the MSA, the SP score is $S(j) = \Sigma_{k<l}\, s(a_{k,j}, a_{l,j})$, where $s(a_{k,j}, a_{l,j})$ is the amino acid exchange value. Using the SP alignment column scores, alignments are scored by taking the total sum of the SP scores: $S = \Sigma_{1\leq j\leq N} S(j)$, where N is the number of aligned positions. In some applications, the occurrence of each gap in the MSA is penalized using typical gap penalty values. However, the SP score is problematic in several regards. It overweighs the evolutionary events close to the root of the alignment tree relative to the leaves. Therefore, a "circular sum" (CS) score derived from the "Traveling Salesman Problem" has been proposed to substitute the SP score (Gonnet et al., 2000).

Evaluating Alignment Methods

Evaluating MSA programs is a complex issue. First of all, there is no general agreement as to what the standard of truth should be. For instance, should an alignment be evaluated using evolutionary, structural, or functional criteria? Although in closely related familial sequences these criteria are expected to lead to the same alignment, in more distant cases they can result in very different answers. Moreover, benchmarks are usually carried out using a set of reference alignments, so that the evaluation

becomes crucially dependent on the quality of such a reference alignment database. A few recent attempts to alleviate this database problem are based on using protein 3-D structures directly in assessing the alignments (Notredame et al., 2000). Furthermore, different ways have been proposed to quantify the agreement between a proposed and a reference alignment, such as unweighted or weighted SP scores, or the column score. The unweighted SP score implies checking, for each aligned amino acid pair in the reference MSA, whether this pair has also been aligned in the alignment produced by the method considered. The final score usually is the percentage of the total number of aligned pairs in the reference alignment that have also been matched in the query alignment. The weighted SP score follows basically the same protocol but weights each pair with the corresponding value from an amino acid exchange matrix (e.g., Blosum62). Finally, the column score checks, for each column in the reference alignment, whether the amino acids found aligned here have been reproduced exactly in the query alignment. If only one sequence is misaligned at the column considered, the whole column is taken to be incorrectly reproduced. Compared to the unweighted and weighted SP scores, the column score is a more stringent measure for alignment evaluation. For example, an outlier sequence that is distant from all other sequences in the query set has a relatively high chance of becoming misaligned, and this will be reflected much more dramatically in the column score than in either SP scores.

Optimizing Alignment Scores

The SP scores of "incorrect" sequence alignments are often higher than those of "true" reference alignments derived from structural superpositioning of proteins. This is caused by a lack of structural information in the substitution models. Heringa (2002) calculated SP scores (for single alignments) for each of the BAliBASE benchmark alignments (Thompson et al., 1999a). These scores were then compared to corresponding SP scores of the alignments calculated using nonoptimized PRALINE conditions. More than a quarter of the PRALINE alignments turned out to have higher SP scores than the corresponding reference alignments, while for the largest BAliBASE alignments, more than half attain larger SP scores for the default PRALINE alignments than those of the corresponding reference alignments. This might be referred to as the "Charlie Chaplin" problem—at the peak of his fame, Charlie Chaplin allegedly entered a Charlie Chaplin contest incognito and came in second (Heringa, 2002). It is clear that trying to optimize the SP score for alignments that already score higher than their corresponding reference alignments is not likely to lead to convergence to the latter alignments.

New avenues to novel scoring schemes, as well as benchmarking methods, are being actively researched. Lin et al. (2003) introduced a new alignment-scoring scheme CAO (Contact Accepted mutatiOn) based on the amino acid interactions in tertiary structures. The scheme is based on a new evolutionary model expressed in 400×400 residue contact mutation matrices, and can be used to evaluate alignments whenever there is a tertiary structure at hand for one or more of the sequences, from which the pairwise residue contacts can be derived. Since the contact-based evolutionary model combines sequence and structure information, it yields biologically more meaningful alignment scores (Lin et al., 2003).

CONCLUSION

An MSA can be viewed as an inflexible representation that provides a unified picture of sequence similarity by averaging out matched residues that possibly cannot be consistently matched over the entire lengths of the sequences. This is because evolution, through mutations, insertions, and deletions of sequence fragments, works on spatially and temporally decoupled molecules, so that sequence alignment incompatibilities can well arise under divergent evolution.

Given these complications, building a reliable MSA for a query set of sequences is a daunting task. In this unit it has been made clear that the increased attention to multiple sequence alignment methodology has resulted in recent developments regarding most of its facets. Computational issues have been addressed by both adapting methods to high-throughput computing by code parallelization and by new speed-optimized alignment formalisms, such as the recent method POA (Lee et al., 2002). Sensitivity has been increased by the development of enhanced techniques for carrying out simultaneous alignment, by devising new profile formalisms, by combining local and global alignment, by new iterative schemes, and by the emergence of new schemes to score and exploit the consistency of alignments.

The increased focus has also led to the construction of new benchmark databases and

novel evaluation protocols. Further developments will be crucially dependent on the integration and representation of biological knowledge in new quality criteria. There are now a multitude of high-quality MSA techniques, each which with particular strengths and weaknesses. Increased sensitivity could abound as a result of new consensus protocols to utilize the combined power of the techniques, or new techniques to determine the kind of alignment problem at hand and then invoke the most appropriate method or combination of methods available. In the meantime, however, it remains important for the end user to run a combination of different MSA methods to optimize the biological information derived from a set of sequences, either through visual inspection of the resulting MSAs or by the application of other bioinformatics techniques that use these MSAs as input.

LITERATURE CITED

Altschul, S.F. and Erickson, B.W. 1986. A nonlinear measure of subalignment similarity and its significance levels. *Bull. Math. Biol.* 48:617-632.

Altschul, S.F., Carrol, R.J., and Lipman, D.J. 1989. Weights for data related by a tree. *J. Mol. Biol.* 207:647-653.

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. 1997. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.* 25:3389-3402.

Bailey, T.L. and Elkan, C. 1994. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *In* Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology. pp. 28-36. AAAI Press, Menlo Park, Calif.

Bailey, T.L. and Elkan, C. 1995. The value of prior knowledge in discovering motifs with MEME. *In* Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology. pp. 21-29. AAAI Press, Menlo Park, Calif.

Barton, G.J. and Sternberg, M.J.E. 1987. A strategy for the rapid multiple sequence alignment of protein sequences: Confidence levels from tertiary structure comparisons. *J. Mol. Biol.* 198:327-337.

Benner, S.A., Cohen, M.A., and Gonnet, G.H. 1993. Empirical and structural models for insertions and deletions in the divergent evolution of proteins. *J. Mol. Biol.* 229:1065-1082.

Blanken, R.L., Klotz, L.C., and Hinnebusch, A.G. 1982. Computer comparison of new and existing criteria for constructing evolutionary trees from sequence data. *J. Mol. Evol.* 19:9-19.

Boguski, M.S., Hardison, R.C., Schwartz, S., and Miller, W. 1992. Analysis of conserved domains and sequence motifs on cellular regulatory proteins and locus control regions using new software tools for multiple sequence alignment and visualization. *New Biol.* 4:247-260.

Brocchieri, L. and Karlin, S. 1998. A symmetric-iterated multiple sequence alignment of protein sequences. *J. Mol. Biol.* 276:249-264.

Bucher, P., Karplus, K., Moeri, N., and Hofmann, K. 1996. A flexible motif search technique based on generalized profiles. *Comput. Chem.* 20:3-24.

Carillo, H. and Lipman, D.J. 1988. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.* 48:1073-1082.

Corpet, F. 1988. Multiple sequence alignment with hierarchical clustering. *Nucl. Acids Res.* 16:10881-10890.

Dayhoff, M.O., Schwart, R.M., and Orcutt, B.C. 1978. A model of evolutionary change in proteins. *In* Atlas of Protein Sequence and Structure (M. Dayhoff, ed.) pp. 345-352. National Biomedical Research Foundation, Washington D.C.

Dayhoff, M.O., Barker, W.C., and Hunt, L.T. 1983. Establishing homologies in protein sequences. *Methods Enzymol.* 91:524-545.

Delcher, A., Phillippy, A., Carlton, J., and Salzberg, S.L. 2002. Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Res.* 30:2478-2483.

Delcher, A.L., Kasif, S., Fleischmann, R.D., Peterson, J., White, O., and Salzberg, S.L. 1999. Alignment of whole genomes. *Nucleic Acids Res.* 27:2369-2376.

Deperieux, E. and Feytmans, E. 1992. Match-Box: A fundamentally new algorithm for the simultaneous alignment of several protein sequences. *Comp. Appl. Biosci.* 8:501-509.

Durbin, R., Eddy, S.R., Krogh, A., and Mitchison, G. (eds.). 1998. Biological sequence analysis, Ch. 5.6, pp. 115-119. Cambridge University Press, Cambridge.

Eck, R.V. and Dayhoff, M.O. 1966. Atlas of Protein Sequence and Structure. National Biomedical Research Foundation, Silver Springs, Maryland.

Eddy, S.R. 1998. Profile hidden Markov models. *Bioinformatics* 14:755-763.

Felsenstein, J. 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. *J. Mol. Evol.* 17:368-376.

Felsenstein, J. 1985. Confidence limits on phylogenies: An approach using the bootstrap. *Evolution* 39:783-791.

Feng, D.F. and Doolittle, R.F. 1987. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.* 25:351-360.

Ficket, J.W. 1984. Fast optimal alignment. *Nucl. Acids Res.* 12:175-180.

Fitch, W. 1966. An improved method of testing for evolutionary homology. *J. Mol. Biol.* 16:9-16.

Fitch, W.M. and Margoliash, E. 1967. Construction of phylogenetic trees. *Science* 155:279-284.

Fitch, W.M. 1971. Toward defining the course of evolution: Minimum change for a specific tree topology. *Syst. Zool.* 20:406-416.

Frishman, D. and Argos, P. 1996. Incorporation of long-distance interactions in a secondary structure prediction method. *Protein Eng.* 9:133-142.

Frishman, D. and Argos, P. 1997. Seventy-five percent accuracy in protein secondary structure prediction. *Proteins*. 27:329-335.

GCG. 1993. Program manual for the GCG Package, v. 8. Genetics Computer Group, Madison, Wis.

Gibbs, A.J. and McIntyre, G.A. 1970. The diagram: A method for comparing sequences. Its use with amino acid and nucleotide sequences. *Eur. J. Biochem.* 16:1-11.

Goldberg, D.E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, New York.

Gonnet, G.H., Korostensky, C., and Benner, S. 2000. Evaluation measures of multiple sequence alignments. *J. Comput. Biol.* 7:261-276.

Gotoh, O. 1986. Alignment of three biological sequences with an efficient traceback procedure. *J. Theor. Biol.* 121:327-337.

Gotoh, O. 1993. Optimal alignment between groups of sequences and its application to multiple sequence alignment. *Comput. Appl. Biosci.* 9:361-370.

Gotoh, O. 1996. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *J. Mol. Biol.* 264:823-838.

Gribskov, M., McLachlan, A.D., and Eisenberg, D. 1987. Profile analysis: Detection of distantly related proteins. *Proc. Natl. Acad. Sci. U.S.A.* 84:4355-4358.

Gropp, W., Lusk, E., Doss, N., and Skjellum, A. 1996. A high-performance, portable implementation of the MPI Message-Passing Interface standard. *Parallel Computing* 22:789-828.

Haussler, D., Krogh, A., Mian, I.S., and Sjölander, K. 1993. Protein modeling using hidden Markov models: Analysis of globins. *In* Proceedings of the Hawaii International Conference on System Sciences. pp. 792-802. IEEE Computer Society Press, Los Alamitos, Calif.

Henikoff, S. and Henikoff, J.G. 1994. Position-based sequence weights. *J. Mol. Biol.* 243:574-578.

Henikoff, S. and Henikoff, J.G. 1999. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U.S.A.* 89:10915-10919.

Heringa, J. 1998. Detection of internal repeats: How common are they? *Curr. Opin. Struct. Biol.* 8:338-345.

Heringa, J. 1999. Two strategies for sequence comparison: Profile-preprocessed and secondary structure-induced multiple sequence alignment. *Comput. Chem.* 23: 341-364.

Heringa, J. 2002. Local weighting schemes for protein multiple sequence alignment. *Comput. Chem.* 26:459-477.

Heringa, J. and Taylor, W.R. 1997. Three-dimensional domain duplication, swapping and stealing. *Curr. Opin. Struct. Biol.* 7:416-21.

Higgins, D.G. and Sharp, P.M. 1988. CLUSTAL: A package for performing multiple sequence alignment on a microcomputer. *Gene* 73:237-244.

Hogeweg, P. and Hesper, B. 1984. The alignment of sets of sequences and the construction of phyletic trees: An integrated method. *J. Mol. Evol.* 20:175-186.

Huang, X. and Miller, W. 1991. A time-efficient, linear-space local similarity algorithm. *Adv. Appl. Math.* 12:337-357.

Huang, X., Hardison, R.C., and Miller, W. 1990. A space-efficient algorithm for local similarities. *CABIOS* 6:373-381.

Johnson, M.S. and Doolittle, R.F. 1986. A method for the simultaneous alignment of three or more amino acid sequences. *J. Mol. Evol.* 23:267-278.

Karplus, K. and Hu, B. 2001. Evaluation of protein multiple alignments by SAM-T99 using the BAliBASE multiple alignment test set. *Bioinformatics* 17:713-720.

Karplus, K., Barrett, C., and Hughey, R. 1998. Hidden Markov models for detecting remote protein homologies. *Bioinformatics* 14:846-856.

Katoh, K., Misawa, K., Kuma, K., and Miyata, T. 2002. MAFFT: A novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Res*. 30:3059-3066.

Kleinjung, J., Douglas, N., and Heringa, J. 2002. Parallelized multiple sequence alignment. *Bioinformatics* 18:1270-1271.

Kluge, A.G. and Farris, J.S. 1969. Quantitative phyletics and the evolution of anurans. *Syst. Zool.* 18:1-32.

Lassmann, T. and Sonnhammer, E.L.L. 2002. Quality assessment of multiple sequence alignment programs. *FEBS Lett.* 529:126-130.

Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., and Wootton, J.C. 1993. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple sequence alignment. *Science* 262:208-214.

Lee, C., Grasso, C., and Sharlow, M.F. 2002. Multiple sequence alignment using partial order graphs. *Bioinformatics* 18:452-464.

Lin, K.X., Kleinjung, J., Taylor, W.R, and Heringa, J. 2003. Testing homology with cao: A contact-based markov model of protein evolution. *Comp. Biol. Chem.* 27:93-102.

Lipman, D.J., Altschul, S.F., and Kececioglu, J.D. 1989. A tool for multiple sequence alignment. *Proc. Natl. Acad. Sci. U.S.A.* 86:4412-4415.

Liu, J.S. and Lawrence, C.E. 1999. Bayesian inference on biopolymer models. *Bioinformatics* 15:38-52.

Lüthy, R., Xenarios, I., and Bucher, P. 1994. Improving the sensitivity of the sequence profile method. *Protein Science* 3:139-146.

Morgenstern, B. 1999. Dialign 2: Improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics* 15:211-218.

Morgenstern B., Dress, A., and Werner, T. 1996. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. *Proc. Natl. Acad. Sci. U.S.A.* 93:12098-12103.

Müller, T. and Vingron, M. 2000. Modelling amino acid replacement. *J. Comput. Biol.* 7:761-776.

Murata, M., Richardson, J.S., and Sussman, J.L. 1985. Simultaneous comparison of three protein sequences. *Proc. Natl. Acad. Sci. U.S.A.* 82:3073-3077.

Needleman, S.B. and Wunsch, C.D. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* 48: 443-453.

Notredame, C. and Higgins, D.G. 1996. SAGA: Sequence alignment by genetic algorithm. *Nucl. Acids Res.* 24: 1515-24.

Notredame, C., Holm, L., and Higgins, D.G. 1998. COFFEE: An objective function for multiple sequence alignments. *Bioinformatics* 14:407-422.

Notredame, C., Higgins, D.G., and Heringa, J. 2000. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.* 302:205-217.

Pacheco, P.S. 1997. Parallel Programming with MPI. Morgan Kaufman Publishers, Inc., San Francisco.

Reinert, K., Stoye, J., and Will, T. 2000. An iterative method for faster sum-of-pair multiple sequence alignment. *Bioinformatics* 16:808-814.

Rost, B. and Sander, C. 1993. Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* 232:584-599.

Saitou, N. and Nei, M. 1987. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.* 4:406-425.

Saitou, N. 1990. Maximum likelihood methods. *Meth. Enzymol.* 183:584-598.

Schneider, T.D. 2002. Consensus sequence zen. *Appl. Bioinformatics* 1:111-119.

Schuler, G.D., Altschul, S.F., and Lipman, D.J. 1991. A workbench for multiple sequence alignment construction and analysis. *Proteins* 9:180-190.

Sibbald, P.R. and Argos, P. 1990. Weighting aligned protein or nucleic acid sequences to correct for unequal representation. *J. Mol. Biol.* 216:813-818.

Sjölander, K., Karplus, K., Brown, M., Hughly, R., Krogh, A., Mian, I., and Haussler, D. 1996. Dirichlet mixtures: A method for improved detection of weak but significant protein sequence homology. *CABIOS* 12:327-345.

Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *J. Mol. Biol.* 147:195-197.

Sneath, P.H. and Sokal, R.R. 1973. Numerical Taxonomy. Freeman, San Francisco.

Sobel, E. and Martinez, H.M. 1986. A multiple sequence alignment program. *Nucl. Acids Res.* 14:363-374.

Sokal, R.R. and Michener, C.D. 1958. A statistical method for evaluating systematic relationships. *Univ. Kansas Sci. Bull.* 28:1409-1438.

Stoye, J. 1998. Multiple sequence alignment with the divide-and-conquer method. *Gene* 211:GC45-GC56.

Stoye, J., Moulton, V., and Dress, A.W.M. 1997. DCA: An efficient implementation of the divide-and-conquer approach to simultaneous multiple sequence alignment. *Comput. Appl. Biosci.* 13:625-626.

Tanner, M.A., and Wong, W.H. 1987. The calculation of posterior distributions by data augmentation. *J. Am. Stat. Assoc*. 82:528-550.

Taylor, W.R. 1988. A flexible method to align large numbers of biological sequences. *J. Mol. Evol.* 28:161-169.

Thompson, J.D., Higgins, D.G., and Gibson, T.J. 1994a. CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucl. Acids Res.* 22:4673-80.

Thompson J.D., Higgins, D.G., and Gibson, T.J. 1994b. Improved sensitivity of profile searched through the use of sequence weights and gap excision. *CABIOS* 10: 19-29.

Thompson, J.D., Gibson, T.J., Plewniak, F., Jeanmougin, F., and Higgins, D.G. 1997. The CLUSTAL X windows interface: Flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Res.* 25:4876-4882.

Thompson, J.D., Plewniak, F., and Poch, O. 1999a. BAliBASE: A benchmark alignments database for the evaluation of multiple sequence alignment programs. *Bioinformatics* 15:87-88.

Thompson, J.D., Plewniak, F., and Poch, O. 1999b. A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Res.* 27:2682-2690.

Vingron, M. and Argos, P. 1989. A fast and sensitive multiple sequence alignment program. *CABIOS* 5:115-121.

Vingron, M. and Sibbald, P.R. 1993. Weighting in sequence space: A comparison of methods in terms of generalized sequences. *Proc. Natl. Acad. Sci. U.S.A.* 90:8777-8781.

Viterbi, A.J. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Informat. Theory* IT-13:260-269.

Wang, L. and Jiang, T. 1994. On the complexity of multiple sequence alignment. *J. Comput. Biol.* 1:723-728.

Waterman, M.S. 1986. Multiple sequence alignment by consensus. *Nucl. Acids Res.* 14:9095-9102.

Waterman, M.S. and Eggert, M. 1987. A new algorithm for best subsequence alignments with application to tRNA-rRNA comparisons. *J. Mol. Biol.* 197:723-728.

Waterman, M.S. and Jones, R. 1990. Consensus methods for DNA and protein sequence alignment. *Methods Enzymol.* 183:221-237.

Zhu J., Liu, J.S., and Lawrence, C. 1998. Bayesian adaptive sequence alignment algorithms. *Bioinformatics* 14:25-31.

KEY REFERENCES

Dayhoff et al., 1978. See above.

This atlas represents a seminal approach to sequence alignment. The evolutionary model that is still used today in most methods is introduced here, together with the now classical PAM series of amino acid exchange matrices. The evolutionary model is often referred to as the Dayhoff model, while the most widely used early matrix in the PAM series, the PAM250 matrix, is commonly known as the Dayhoff matrix.

Felsenstein, 1981. See above.

In this paper, the important evolutionary method of maximum likelihood is introduced. The method, which attempts to find the tree that maximizes the probability that the observed data will fit the tree under a given evolutionary model, is now generally accepted as the most accurate strategy.

Hogeweg and Hesper, 1984. See above.

This paper introduces the progressive multiple alignment strategy, which is still the most widely used multiple alignment technique. In this early paper, alignment iteration is already addressed. Another interesting feature of the paper is the use of so-called internode sequences, which are additionally inferred sequences ancestral to subgroups of sequences in the phylogenetic tree calculated for the query sequence set.

Needleman and Wunsch, 1970. See above.

This is one of the most quoted early papers on sequence alignment. In this paper, the global dynamic programming algorithm is introduced to the biological community and applied to pairwise amino acid sequences. The basic dynamic programming algorithm had been conceived before by the physicist Richard Belman, who published a large series of papers and books on the topic during the 1950s and 60s.

Smith and Waterman, 1981. See above.

Following the approach by Needleman and Wunsch (1970), Smith and Waterman derived the now classical algorithm for local pair-wise sequence alignment. Most widely used homology search engines such as BLAST are fast approximations of the Smith and Waterman algorithm and perform local alignment.

INTERNET RESOURCES

BioPat software package
http://www.biopat.de

Clustal
http://www.ebi.ac.uk/clustalw/

DCA
http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/dca_submit

Dialign2
http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/dialign_submit

ITERALIGN
http://giotto.stanford.edu/~luciano/iteralign.html

MACAW
http://www.bris.ac.uk/Depts/PathAndMicro/services/CGR/Common%20Files/MACAWinstructions.htm

MAFFT
http://www.biophys.kyoto-u.ac.jp/~katoh/programs/align/mafft/

Match-Box
http://www.sciences.fundp.ac.be/biologie/bms/matchbox_submit.shtml

MEME
http://meme.sdsc.edu/meme/website/meme.html

MSA
http://xylian.igh.cnrs.fr/msa/msa.html

MULTAL
http://mathbio.nimr.mrc.ac.uk/ftp/wtaylor/multal/

MultAlign
http://cbrg.inf.ethz.ch/Server/MultAlign.html

MultAlin
http://prodes.toulouse.inra.fr/multalin/multalin.html

MUMmer
http://www.tigr.org/software/mummer/

OMA
http://bibiserv.techfak.uni-bielefeld.de/oma/

PileUp (GCG software package)
http://www.accelrys.com

POA
http://www.bioinformatics.ucla.edu/poa/

PRALINE
http://www.cs.vu.nl/~ibivu/programs/pralinewww/

Prrp
ftp://ftp.genome.ad.jp/pub/db/hgc/software/saitama-cc/

SAGA
http://igs-server.cnrs-mrs.fr/~cnotred

T-Coffee

http://igs-server.cnrs-mrs.fr/Tcoffee/

TNB

http://globin.cse.psu.edu/ftp/dist/TNB/

Literature on the Web

http://evol-linux1.ulb.ac.be/ueg/ProAlign/Loytynoja_and_Milinkovitch_2003.pdf

Loytynoja and Milinkovitch, 2003. A hidden Markov model for progressive multiple alignment.

http://hmmer.wustl.edu

Eddy, S.R. 2001. HMMER: Profile hidden Markov models for biological sequence analysis.

Contributed by Victor Simossis, Jens Kleinjung, and Jaap Heringa
Integrative Bioinformatics Institute (IBIVU)
Free University
Amsterdam, The Netherlands