realistic models of gap penalization treat the initiation of a gap differently from its continuation: increasing the length of a gap by one amino acid is penalized less than inserting one amino acid in a different position.

In the linear penalty model, each gap has a penalty w_o augmented by a penalty w_e lower than w_o for each increase in the length of the indel. Alternatively, each inserted and deleted amino acid in an indel can be assigned a penalty lower than the previous one; for example, $\log_e w_o$, in which w_o is the penalty assigned to the first inserted or deleted amino acid, and the index e runs from 2 for the adjacent inserted or deleted amino acid to l for the last amino acid in the gap. If $w_o = 5$, the penalty for inserting four amino acids is $5 + \log_2 5 + \log_3 5 + \log_4 5 = 5 + 2.32 + 1.46 + 1.16 = 9.95$. Compare this result with the value $5 \times 4 = 20$ that would be obtained by application of a linear penalty scheme.

Local versus Global Alignment

The alignment between two protein sequences is global when it is aimed at finding the optimal correspondence between all amino acids of both sequences and local when it attempts to find local regions of similarity between the two sequences. The latter is biologically relevant because it might allow us to detect evolutionarily related domains present in proteins whose remaining sequences have no evolutionary relationship or allow us to highlight regions that contain functional units subject to stronger evolutionary pressures. Local alignments are also useful for the detection of proteins homologous to a target protein in a large data set of unrelated proteins.

How Do We Align Sequences?

Global Alignment of Two Protein Sequences: The Needleman and Wunsch Algorithm

An alignment of two protein sequences is a correspondence between the amino acids or appropriately inserted gaps of the first sequence and amino acids or gaps of the second sequence. Clearly, a gap in the first sequence cannot correspond to a gap in the second sequence, and the alignment can be seen as a matrix with two rows, one corresponding to each sequence, and each column corresponds to a pair of aligned amino acids or to an amino acid and a gap.

We use a maximum-parsimony approach and assume that the best alignment (i.e., the one that best reflects the evolutionary relationship between the two protein sequences) requires the minimum number of substitutions, insertions, and deletions. If we use our substitution matrices to assign a score to each pair of amino acids (a score that reflects the probability that the amino

acids replaced each other during evolution) and a penalty value for indels, we are looking for the alignment for which the sum of the scores of each pair of aligned amino acids, diminished by the penalty values for the indels, is maximum.

Thus, the problem of aligning two protein sequences is reduced to the problem of finding the alignment between the two strings that represent their sequences such that the global score is maximum, *given a score function and penalty values for indels*.

The optimal alignment between two strings can be exactly computed, but the optimal alignment is biologically correct only insofar as the score function and the indel penalty values are biologically reasonable.

The Needleman and Wunsch algorithm finds the optimal alignment between two sequences by calculating the optimal alignment between subsequences of increasing length. It is based on two assumptions:

- Mutations in different sites of a sequence occur independently.
- The length of a gap does not depend on the elements aligned to the gap.

Both hypotheses are approximations of biological reality because different positions in sequences are subject to different evolutionary pressure, and not all sequences of amino acids can be accommodated in inserted regions. These sequences are usually solvent exposed and, therefore, are more likely composed of hydrophilic or flexible amino acids.

However, under these assumptions and given a scoring matrix and a gap penalty scheme, the alignment problem can be exactly solved by dynamic programming methods.

Let us write the two sequences in the first row and the first column of a matrix (Figure 17). Each cell corresponds to the alignment of the amino acid in the row with that in the column and contains a score reflecting the similarity between the two amino acids. Our alignment problem can now be reformulated as follows: what is the set of cells (that is, pairs of aligned amino acids) that we should use to go from the upper left corner to the lower right corner so that we "collect" the larger score?

We must build a new matrix (the so-called cumulative matrix) in which each cell contains the maximum score achievable by any alignment that ends in that cell. This score is not difficult to compute.

If we know the maximum score that can be achieved by any alignment that ends in the cell $\{s_{i-1},t_{j-1}\}$, $\{s_{i-1},t_j\}$, and $\{s_i,t_{j-1}\}$, then the maximum score $F(i,j)$ that can be achieved by any alignment that includes the pair $\{s_i,t_j\}$ can be easily calculated. Only three "moves" allow extension of the alignment to the i,j position: aligning s_i and t_j, aligning s_i with a gap, and aligning t_j with a gap. We select the path that provides the maximum score:

$$F(i,j) = \max \begin{bmatrix} F(i-1,j-1) + \sigma(s_i, t_j) \\ F(i-1,j) - w_i \\ F(i,j-1) - w_j \end{bmatrix} \quad (1)$$

where (s_i,t_j) is the score value for the pair of amino acids x_i and y_j, and w_i and w_j are the appropriate penalties for inserting a single amino acid gap in the i and j positions, respectively.

We must now compute $F(0,0)$, $F(1,0)$, $F(0,1)$. $F(0,0)$ is equal to zero, as it does not correspond to any pair of aligned residues. $F(1,0)$, $F(0,1)$ are the maximum scores that can be achieved by aligning the first amino acid of each sequence with a gap (Figure 17), and, therefore, we can set their values to $-w_i$ and $-w_j$ or to 0, if we do not want to penalize gaps at the beginning of either sequence. Similar considerations allow us to fill the complete first row and first column. They correspond to further insertions in one of the two sequences, so we can either assign an extra indel penalty for each of them or set them to 0. Now we can completely fill the matrix, and the maximum score of the global alignment is, by definition, the one reported in the lower right cell (n,m). We can achieve the maximum score if we include in the alignment the (n,m) pair and one of the pairs $(n-1,m)$, $(n,m-1)$, or $(n-1,m-1)$, namely, the one that we used to obtain the maximum value in (n,m) that we recorded during the matrix construction procedure. This step can be iterated until a cell of the first column or of the first row is reached, which generates a path though the matrix including the cells that contribute to the alignment with maximum score.

Local Alignment of Two Protein Sequences: The Smith and Waterman Algorithm

A modification of the Needleman and Wunsch algorithm can be used to obtain local sequence alignments; that is, alignments that do not necessarily include the whole length of the two sequences.The difference consists essentially of not allowing negative scores, because a good local alignment is unlikely to include gaps or rarely observed substitutions. From the algorithmic point of view, this assumption implies that equation (1) is replaced by

$$F(i,j) = \max \begin{bmatrix} 0 \\ F(i-1,j-1) - \sigma(s_i, t_j) \\ F(i-1,j) - w_i \\ F(i,j-1) - w_j \end{bmatrix} \quad (2)$$

We assign a value of 0 to $F(i,0)$ and $F(j,0)$ for each i and j and construct the matrix as in the previous case. The reconstruction of the alignment now

starts from the maximum value in the matrix and proceeds with the same strategy as in global alignments, but it stops when a score of 0 is encountered.

Multiple-Sequence Alignments

A multiple-sequence alignment (MSA) between N protein sequences is again a matrix, this time with N rows such that each row contains a protein sequence, possibly with gaps inserted. Clearly, each column should contain at least one element that is not a gap.

As for pairwise alignment, the problem is to find the MSA that maximizes a predefined score. Unfortunately, the extension of the scoring scheme and of the algorithms described for two sequences to multiple-sequence alignments presents a number of problems.

One possible scoring scheme, called SP-score (sum of pairs score), is calculated by addition of all possible pairwise scores for each column and then summation of the scores for all columns. The underlying assumptions for the SP-score is that both the columns and the rows of the alignment matrix are statistically independent. This assumption is clearly not valid, because the protein sequences in the alignment are supposedly evolutionarily related.

Another problem of SP-scores can be better illustrated by the following example. Let us calculate an SP-score for the alignment in Figure 18a. For simplicity, we assign a score of 1 to identical amino acids and 0 for all other amino acids. Each column has three amino acids. In the first column, they are all different, and the score is 0. In the second and third columns, all three pairs are formed by identical amino acids, and the score is $3 \times 1 = 3$. The fourth column has again a score of 0. The last column contains one match (*F*-*F*) and two mismatches (*F*-*Y*), and, therefore, the score is $1 \times 1 + 2 \times 0 = 1$.

The alignment in Figure 18b is biologically less convincing than the alignment in Figure 18a, because the third position contains one mismatch, and the SP-score is indeed lower. The ratio between the score of the two alignments is $7/6 = 1.17$.

Let us now assume that we add one more sequence to both alignments as shown in Figure 18c and Figure 18d. From a biological point of view, the addition of a fourth sequence where again a C appears in the third position makes the first of the two alignments even more convincing, whereas it decreases the relative likelihood of the second. The presence of the G (in bold in Figure 18) in an otherwise very conserved position is more unlikely here than in the alignment shown in Figure 18a and Figure 18b. We would like our scoring system to recognize that the addition of the fourth sequence shifts our confidence toward the first alignment, yet the ratio of the scores of the two alignments is now 1.08 and erroneously points to a smaller difference between the quality of the two alignments shown in Figure 18c and Figure 18d with respect to those shown in Figure 18a and Figure 18b. This problem and the fact that there is no statistical justification for its usage notwithstanding the SP-score is often used because of its simplicity.

Let us assume that we need to align two sequences:

ACFFTGHILPRG and ADYTGHLMPKA

We first build a substitution matrix: the two sequences are in the first row and in the first column. In each cell there is the score for the pair of amino acids corresponding to the column and the row derived from the PAM250 matrix shown in Table 2:

	A	C	F	F	T	G	H	I	L	P	R	G
A	2	-2	-3	-3	1	1	-1	-1	-2	1	-2	1
D	0	-5	-6	-6	0	1	1	-2	-4	-1	-1	1
Y	-3	0	7	7	-3	-5	0	-1	-1	-5	-4	-5
T	1	-2	-3	-3	3	0	-1	0	-2	0	-1	0
G	1	-3	-5	-5	0	5	-2	-3	-4	0	-3	5
H	-1	-3	-2	-2	-1	-2	6	-2	-2	0	2	-2
L	-2	-6	2	2	-2	-4	-2	2	6	-3	-3	-4
M	-1	-5	0	0	-1	-3	-2	2	4	-2	0	-3
P	1	-3	-5	-5	0	0	0	-2	-3	6	0	0
K	-1	-5	-5	-5	0	-2	0	-2	-3	-1	3	-2
A	2	-2	-3	-3	1	1	-1	-1	-2	1	-2	1

Let's decide that we select a constant penalty value of 5 for indels. Now we need to build the cumulative matrix. We will set the penalty for inserting at the beginning of each sequence to zero. Therefore,

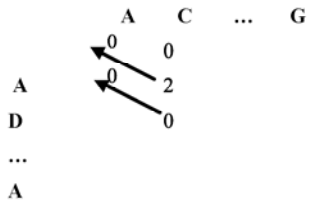
		A	C	…	G
	0	0			
A	0				
D					
…					
A					

We need to fill the shaded cell, and we can get there either from the cell on top of it, from the one at its left or from the cell diagonally up. If we came from the cell on the top, we would effectively insert one residue: the A of the horizontal sequence would not correspond to any amino acid of the vertical sequence. In this case the score in the cell should be – 5, i.e. 0 (the content of the cell we are coming from) minus the penalty for an indel, which is 5. The second case is similar (we would be inserting in the other sequence) and the score would again be – 5. In the third case there would be no insertion and the score would be 0 (the content of the cell we are coming from) plus 2 (the score of the A, A pair) =2. The maximum value between (-5, -5, 2) is 2, therefore we write 2 in the cell and remember that we obtained it using the cell diagonally up:

		A	C	…	G
	0	0			
A	0	2			
D	0				
…					
A					

FIGURE 17
The Needleman and Wunsch algorithm for pairwise sequence alignment. *(continues)*

We can now fill another cell, shaded in the scheme. If we came from the cell above it, the score would be 2 (the starting value) –5 (the insertion penalty) = -3. If we came from the cell on the left, we would have a score of 0 –5 = –5, for the diagonal cell the score would be 0 +0 = 0. Therefore:



And so on, until we complete the cumulative matrix:

		A	C	F	F	T	G	H	I	L	P	R	G
	0	**0**	**0**	0	0	0	0	0	0	0	0	0	0
A	0	**2**	**-2**	**-3**	-3	1	1	-1	-1	-2	1	-2	1
D	0	0	**-3**	**-8**	-8	-3	2	2	-3	-5	-3	0	-1
Y	0	-3	0	**4**	**-1**	-6	-3	2	1	-4	-8	-5	-5
T	0	1	-4	-1	1	**2**	-3	-3	2	-1	-4	-9	-5
G	0	1	-2	-6	-4	1	**7**	2	-3	-2	-1	-6	-4
H	0	-1	-2	-4	-8	-4	-1	**13**	0	-5	-2	1	-4
L	0	-2	7	0	-2	-7	-6	8	**15**	10	5	0	-3
M	0	-1	2	7	0	-3	-8	3	10	**19**	14	9	4
P	0	1	-3	2	2	0	-3	-2	5	14	**25**	20	15
K	0	-1	-4	-3	-3	2	-2	-3	0	9	20	**28**	23
A	0	2	-3	-7	-6	-2	3	-2	-4	4	15	23	**29**

The optimal alignments, given our substitution matrix and our indel penalty scheme, are:

A D - Y T G H L M P K A
A C F F T G H I L P R G

A D Y - T G H L M P K A
A C F F T G H I L P R G

- A D Y T G H L M P K A
A C F F T G H I L P R G

- - A D Y T G H L M P K A
A C F - F T G H I L P R G

FIGURE 17 (CONTINUED)

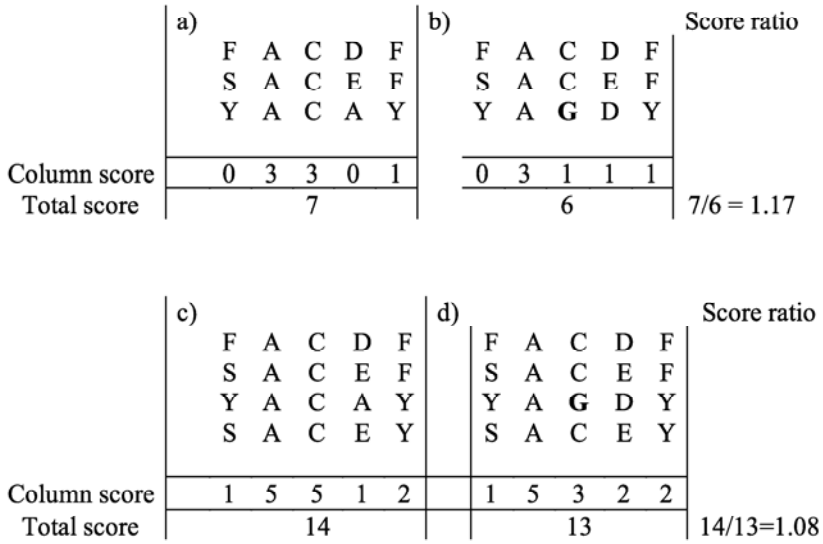


FIGURE 18
Illustration of the problems connected with the SP-score in multiple sequence alignments.

Several alternative scoring schemes have been proposed, but they all assume that each sequence is unrelated to any other sequence in the alignment; this assumption is not valid for sequences that belong to an evolutionary family.

Another nontrivial problem is connected with multiple-sequence alignments. A simple calculation shows that the extension of the Needleman and Wunsch algorithm to N sequences is technically unfeasible.

If we want to extend the algorithm to a multiple-sequence alignment of N sequences of length L, each cell has $2^{(n-1)}$ neighboring cells, and the total number of cells is L^N. Therefore, the calculation of the optimal alignment requires a number of steps of the order of $L^N \times 2^N$, a number that becomes quickly untreatable. An alignment of 50 sequences of length 150 amino acids requires more than 10^{150} operations.

Interesting ideas have been presented to solve this problem. For example, one can calculate the minimum of the alignment score for each pairwise alignment and assume that this score is a lower bound for the overall score. When we use our algorithm, we can reduce the number of cells that we have to consider by discarding those that would make our score lower than the lower bound. Even so, the problem can be solved in a reasonable time only if the number of sequences is not more than a few, and their length is only 100 to 200 amino acids.

Another solution to the problem is the use of a progressive method; that is, first align two sequences, then align a third sequence to the first alignment, then align a fourth to the alignment, and so on, iteratively. To align a sequence to an alignment, we can align the new sequence to each sequence of the alignment and select the highest scoring pair. We can align two alignments by calculating each possible pairwise score between sequences of the first alignment and sequences of the second, again using the highest scoring pair to build the new alignment.

We can also use a simple modification of the pairwise alignment algorithm. Instead of writing a single sequence in the first row, we write a previously calculated pairwise alignment. The score of each cell in the substitution matrix is the average score of each amino acid of the alignment with the corresponding amino acid of the vertical sequence:

	A	**C**	**F**	**F**	**T**	**G**	**H**	**I**	**L**	**P**	**R**	**G**
	A	**D**	**Y**	**F**	**S**	**G**	**S**	**V**	**M**	**P**	**K**	**A**
A	2.0	–1.0	–3.0	–3.0	1.0	1.0	0.0	–0.5	–1.5	1.0	–1.5[a]	1.5
D	0.0	–0.5	–5.0	–6.0	0.0	1.0	0.5	–2.0	–3.5	–1.0	–0.5	0.5
Y	–3.0	–2.0	8.5	7.0	–3.0	–5.0	–1.5	–1.5	–1.5	–5.0	–4.0	–4.0
T	1.0	–1.0	–3.0	–3.0	2.0	0.0	0.0	0.0	–1.5	0.0	–0.5	0.5
G	1.0	–1.0	–5.0	–5.0	0.5	5.0	–0.5	–2.0	–3.5	0.0	–2.5	3.0
H	–1.0	–1.0	–1.0	–2.0	–1.0	–2.0	2.5	–2.0	–2.0	0.0	1.0	–1.5
L	–2.0	–5.0	0.5	2.0	–2.5	–4.0	–2.5	2.0	5.0	–3.0	–3.0	–3.0
M	–1.0	–4.0	–1.0	0.0	–1.5	–3.0	–2.0	2.0	5.0	–2.0	0.0	–2.0
P	1.0	–2.0	–5.0	–5.0	0.5	0.0	0.5	–1.5	–2.5	6.0	–0.5	0.5
K	–1.0	–2.5	–4.5	–5.0	0.0	–2.0	0.0	–2.0	–1.5	–1.0	4.0	–1.5
A	2.0	–1.0	–3.0	–3.0	1.0	1.0	0.0	–0.5	–1.5	1.0	–1.5	1.5

[a] $= \frac{1}{2}[score\ (R.A) + score\ (K.A)] = \frac{1}{2}(-2-1) = -1.5$

We must now decide the order in which to align the sequences. In other words, which pair do we align first, and in which order do we select the subsequent sequences? Aligning the most similar pair of sequences first seems sensible because their correct alignment is likely to be easier and might, therefore, aid in the more difficult subsequent alignments.

The strategy is as follows:

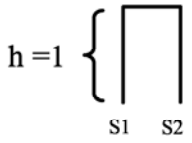
1. Calculate similarity values for each pair of sequences.
2. Select the pair with highest similarity and proceed to align them.
3. Recalculate the similarity between the aligned pair and each of the other sequences.
4. Repeat steps 2 and 3 until all sequences have been aligned.

The result of the procedure can be visually represented as a binary tree, in which each node is a sequence and each edge is proportional to the "distance" between the nodes. The distance is inversely related to the similarity between two sequences, between one sequence and one alignment, or between two alignments. The tree is usually called the guide tree of the alignment and approximates the evolutionary relationship between the sequences.

Phylogenetic trees (i.e., trees representing true evolutionary relationships) are rooted; they include one node that represents the original ancestor of all sequences. Several algorithms are available for constructing trees, provided

In the matrix, S1, S2, S3, S4 and S5 represent five sequences and the numbers in each cell are the "distance" between the sequences of the row and the column. This can be some function of the similarity or identity value between the sequences. We select the two "closest" sequences and draw them as shown on the left in such a way that the vertical line is half of their distance.

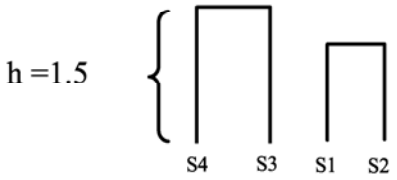
	S1	S2	S3	S4	S5
S1	0	2	5	4	7
S2		0	4	3	5
S3			0	3	6
S4				0	4
S5					0



We now calculate the distance of this group from each other sequence as the average of their distances from S1 and S2:

	S12	S3	S4	S5
S12	0	4.5	3.5	6
S3		0	3	6
S4			0	4
S5				0

= (5+4)/2



The closest pair is now S3 and S4:

	S12	S34	S5
S12	0	4	6
S34		0	5
S5			0

And finally s12 and S34:

	S1234	S5
S1234	0	5.5
S5		0

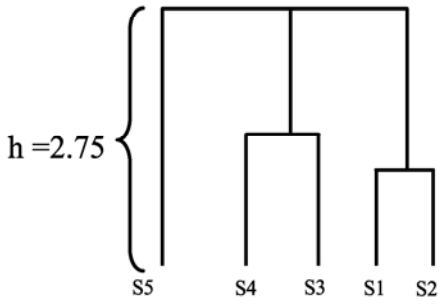


FIGURE 19
The UPMGA (unweighted pair-group average) method for building a tree.

that metrics between sequences have been defined. Some of the algorithms provide hypotheses about the location of the roots, while others do not and can be used to construct unrooted trees. The latter, although unable to give information about the evolutionary events that have generated the family of homologous proteins, are still very useful for simplifying the problem of multiple-sequence alignments.

Several methods exist for reconstructing the most likely evolutionary tree that has generated the observed differences between a given set of sequences. The methods take into account several factors, such as the possibility that different branches of the tree have different evolutionary rates. For the purpose of multiple-sequence alignment, we do not need a very accurate evolutionary reconstruction, and we can use less sophisticated methods, such as the one described in Figure 19.

The multiple alignment protocol originally devised by Feng and Doolittle consists of the following two steps:

1. Calculate the distances between each pair of sequences and use them to construct an approximate tree that will only be used to decide the order of alignments and, therefore, does not need to be especially accurate.
2. Align the pair of sequences, the pair of alignments, or one sequence and one alignment, starting from the child nodes of the tree, so that the most similar sequences are aligned first and the most dissimilar ones are aligned last.

In the Feng and Doolittle implementation of this procedure, gaps are penalized the first time they appear in an alignment but not when they are inserted in the same position in the subsequent alignment step (once a gap, always a gap). This practice is biologically sensible because all proteins in the tree are assumed to be homologous, which implies that their three-dimensional structures are topologically similar. Therefore, the position of a gap is structurally equivalent in all proteins, and if allowed in one pair, it also falls into an allowed region in the others.

Another reconstruction approach is to use the score of a column of the alignment (e.g., the SP-score) as the score in the alignment procedure. We only need to define the score for a gap-to-gap alignment (e.g., 0).

The widely used ClustalW method is essentially based on this last algorithm, with some clever additions. The alignment of each pair of sequences or alignments is built by use of a matrix appropriate for their evolutionary distance. Gap penalties depend on the amino acids observed in the column, so the presence of many hydrophilic or flexible residues in a column lowers the gap penalty in that position. Furthermore, the gap penalty is increased for columns that do not contain gaps, if gaps are present nearby in the alignment. Finally, the guide tree can be adjusted at the alignment stage on the basis of the scores of the alignments.

One problem with iterative multiple-sequence alignment is that the addition of a new sequence cannot modify the preexisting alignment. One solution, proposed by Barton and Sternberg, consists of taking out one sequence at a time and realigning it to the multiple alignment.

Profiles

Given a multiple-sequence alignment, we can derive the probability that each given amino acid is found in one of the aligned positions. We simply count the number of times each of the 20 amino acids appears in each column and divide the number of appearances by the number of aligned sequences. If the number of sequences is sufficiently high, these frequency values approximate the probability of finding any given amino acid in any position of the alignment.

When a new sequence is available, we can align it to the multiple alignment, calculate the probability that each of its amino acids is found in each