CABIOS Vol.9 no 3 1993
Pages 361 – 370

Optimal alignment between groups of sequences and its application to multiple sequence alignment

Osamu Gotoh

Abstract

Four algorithms, A–D, were developed to align two groups of biological sequences. Algorithm A is equivalent to the conventional dynamic programming method widely used for aligning ordinary sequences, whereas algorithms B–D are designed to evaluate the cost for a deletion/insertion more accurately when internal gaps are present in either or both groups of sequences. Rigorous optimization of the 'sum of pairs' (SP) score is achieved by algorithm D, whose average performance is close to $O(MNL^2)$, where M and N are numbers of sequences included in the two groups and L is the mean length of the sequences. Algorithm B uses some approximations to cope with profile-based operations, whereas algorithm C is a simpler variant of algorithm D. These group-to-group alignment algorithms were applied to multiple sequence alignment with two iterative strategies: a progressive method based on a given binary tree and a randomized grouping–realignment method. The advantages and disadvantages of the four algorithms are discussed on the basis of the results of examinations of several protein families.

Introduction

Alignment of a family of protein or nucleotide sequences is a fundamental step in the study of function–structure relationships and evolution of families of macromolecules. Many methods using various strategies have been developed to obtain multiple sequence alignments (Sankoff and Cedergren, 1983; Sobel and Martinez, 1986; Santibánez and Rohde, 1987; Carrillo and Lipman, 1988; Vingron and Argos, 1989). Because of their relatively good cost-performance, the currently most popular methods rely on iterative use of pairwise or three-way dynamic programming algorithms (Hogeweg and Hesper, 1984; Waterman and Perlwitz, 1984; Bains, 1986; Feng and Doolittle, 1987; Barton and Sternberg, 1987; Higgins and Sharp, 1988; Corpet, 1988; Taylor, 1988; Gotoh and Fujii-Kuriyama, 1989; Subbiah and Harrison, 1989; Hein, 1989; Smith and Smith, 1990). In the course of iteration, such a procedure aligns two (or three) groups of sequences, each of which may contain gaps (insertions and deletions). Since gaps tend to occur outside secondary structures (Lesk *et al.*, 1986), their locations provide important information about the alignment of distantly related groups of proteins or nucleic acids. However, little attention has been paid to this point, and iterative alignment procedures have often been performed without a definitive measure that takes internal gaps into consideration.

With respect to simultaneous alignment of several sequences, Carrillo and Lipman (1988) suggested that the 'sum of pairs' score (SP-score) is the most practical for evaluating alternative multiple sequence alignments, especially when the gap penalty, $w(k)$, is not directly proportional to the length of the gap, k. It is now widely recognized that non-proportional gap-weighting functions, the simplest of which is a linear function of the form, $w(k) = uk + v$, produce much better alignments than a proportional function (Gotoh, 1990). Although scores other than the SP-score, such as the parsimony score (Sankoff and Cedergren, 1983), may be biologically more feasible, it is very complicated to incorporate non-proportional gap-weighting functions into such a generalized scoring scheme (Altschul and Lipman, 1989). Altschul (1989) also discussed the computational complexity of dynamic programming algorithms with the SP-score and a linear gap-weighting function. He showed that the computation time spent at each recursive step grows more rapidly than the factorial of N, the number of sequences aligned simultaneously. This implies that to obtain the exact solution for $N \geq 4$, the computation time can reach an impractical range even if one restricts the scan space by some means (Carrillo and Lipman, 1988; Spouge, 1989).

In this paper, I first discuss several algorithms for obtaining alignment of two groups of sequences with internal gaps. One of the algorithms gives an optimal solution in the sense of the best SP-score with a linear gap-weighting function. When one group consists of M primary sequences and the other consists of N sequences, this algorithm runs close to $O(MNL^2)$, where L is the average length of the two groups of sequences. The algorithm is then applied to an iterative procedure for producing multiple-sequence alignment in a similar way to that recently proposed by Berger and Munson (1991). The Berger and Munson method, which follows the gap-weighting scheme of Murata *et al.* (1985), does not take into account detailed alignment of the sites where any sequence in either of the prealigned groups contains a deletion. The present method explicitly evaluates all configurations, ensuring a better approach toward the best multiple-sequence alignment under given conditions.

Department of Biochemistry, Saitama Cancer Center, Research Institute, Ina-machi, Saitama 362 Japan

System and methods

The programs were written in standard C and run under both Unix (4.2 BSD on SONY NEWS 830 or Sun OS4.1.1. on SPARCstation2) and MS-DOS (NEC PC9801 or IBM PC) environments. Turbo C v. 2.0 (Boland, CA) was used for compilation with the large memory model under the MS-DOS systems.

Algorithm

Assume that the two groups of prealigned sequences, A and B, are represented by arrays of column vectors, $A = \{\mathbf{a}_i;\ i = 1, \ldots, I\}$ and $B = \{\mathbf{b}_j;\ j = 1, \ldots, J\}$, where the dimensions of $\mathbf{a}_i$ and $\mathbf{b}_j$, M and N, are the numbers of primary sequences contained in A and B respectively, and I and J are the lengths of sequence groups A and B respectively. In an alignment process, each column vector, $\mathbf{a}_i$ or $\mathbf{b}_j$, behaves as a unit, and hence we may refer to $\mathbf{a}_i$ or $\mathbf{b}_j$ as a unit of A or B. The mth element of $\mathbf{a}_i$, $a_{m,i}$ $(1 \le m \le M,\ 1 \le i \le I)$, is either a residue of the mth sequence in group A or a null indicating a site of deletion introduced by the prealignment. (A null is indicated by a 'Δ' in the following text.) Likewise, the nth element of $\mathbf{b}_j$, $b_{n,j}$ $(1 \le n \le N,\ 1 \le j \le J)$, is either a residue on the nth sequence in group B or a null.

Let $d(x,y)$ denote the measure of dissimilarity between elements x and y. When $x \ne \Delta$ and $y \ne \Delta$, $d(x,y)$ is a usual dissimilarity measure such as −MDM78 (Dayhoff *et al.*, 1978). On the other hand, $d(x, \Delta)$ or $d(\Delta, y)$ specifies the cost for elongation of a deletion opposite to the residue x or y, and $d(\Delta, \Delta) = 0$ because matching nulls are ignored. When $d(x, \Delta) = d(\Delta, y) = u = \text{const.}$, u equals the proportional coefficient of the linear gap-weighting function. The SP-score of group A is defined by

$$\mathrm{SP}(A) = \sum_{m=2}^{M} \sum_{k=1}^{m-1} S_{m,k} \qquad \text{(1A)}$$

$$S_{m,k} = \sum_{i=1}^{I} d(a_{m,i}, a_{k,i}) + v \cdot g_{m,k} \qquad \text{(1B)}$$

where $S_{m,k}$ is the score associated with the pairwise alignment between the mth and the kth sequences in A, $g_{m,k}$ is the number of gaps in that alignment, and v equals the constant term of the gap-weighting function (gap-opening penalty). The SP-score of group B is defined analogously.

We can calculate $g_{m,k}$ and hence $S_{m,k}$ by simple induction. Define $q_{m,i}$ as a Boolean value indicating $a_{m,i} = \Delta$. We assume that any logical expression takes the value of 1 if true and 0 if false. Let $Q_{m,i}$ denote the number of consecutive nulls in the sequence m including and immediately preceding the element $a_{m,i}$. With the initial condition of $Q_{m,0} = 0$, $Q_{m,i}$ is obtained by the recursion relation (Taylor, 1984):

$$\text{if } q_{m,i} \text{ then } Q_{m,i} \leftarrow Q_{m,i-1} + 1 \text{ else } Q_{m,i} \leftarrow 0 \qquad (2)$$

A

	q_i	0	0	1	1
	r_i	0	1	0	1
q_{i-1}	r_{i-1}				
0	0	0	+	+	0
0	1	0	0	+	0
1	0	0	+	0	0
1	1	0	?	?	0

q_i	0	0	1	1
r_i	0	1	0	1
$Q_{i-1}=R_{i-1}$	0	+	+	0
$Q_{i-1}<R_{i-1}$	0	0	+	0
$Q_{i-1}>R_{i-1}$	0	+	0	0

B

```
$Q_0 \leftarrow R_0 \leftarrow g \leftarrow 0$;
for $i \leftarrow 1$ to $I$ do
    begin
        if $(Q_{i-1} \le R_{i-1})$ and $q_i$ and $(1-r_i)$ or
           $(Q_{i-1} \ge R_{i-1})$ and $(1-q_i)$ and $r_i$
          then  $g \leftarrow g + 1$;
        if $q_i$ then $Q_i \leftarrow Q_{i-1} + 1$ else $Q_i \leftarrow 0$;
        if $r_i$ then $R_i \leftarrow R_{i-1} + 1$ else $R_i \leftarrow 0$;
    end
```

C

```
a = a a t g Δ Δ Δ Δ c Δ Δ t
b = a t Δ Δ c Δ Δ a c Δ t t

q = 0 0 0 0 1 1 1 1 0 1 1 0
r = 0 0 1 1 0 1 1 0 0 1 0 0

Q = 0 0 0 0 1 2 3 4 0 1 2 0
R = 0 0 1 2 0 1 2 0 0 1 0 0
        ↑   ↑           ↑
```

Fig. 1. Rules for detecting gap formation (**A**) and an algorithm for counting the number of gaps, g, in a pair of sequences within a multiple-sequence alignment (**B**). The Boolean value, q_i or r_j, denotes that the element of a_i or b_j is a null. Q_i and R_j indicate the numbers of consecutive nulls including and immediately preceding the elements a_i and b_j respectively. The situations under which a new gap opens at the position i are indicated by plus signs in the table of (A), whereas the situations without formation of a new gap are indicated by zeros. A question mark indicates that information is insufficient for the detection of gap formation. A test run of the algorithm in (B) is shown in (C). The arrows indicate the positions where new gaps open.

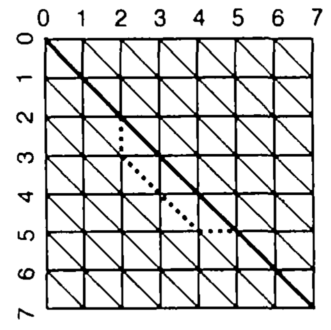
Figure 1(A) lists the cases for a new gap between the sequences m and k to open at the column position i. The notations in this table are simplified as: $Q_i = Q_{m,i}$, $R_i = Q_{k,i}$, $q_i = (a_{m,i} = \Delta)$, and $r_i = (a_{k,i} = \Delta)$. Figure 1(A) shows that column vectors $\mathbf{q}_i = \{q_{m,i};\ 1 \le m \le M\}$ and $\mathbf{q}_{i-1}$ do not, but $\mathbf{q}_i$ and $\mathbf{Q}_{i-1}$ do hold sufficient information to detect all gaps that open at the position i. Hence, $\mathbf{Q}_i$ may be referred to as the 'gap state' at position i. Summarizing the above considerations, we obtain $g_{m,k}$ by the algorithm shown in Figure 1(B). Figure 1(C) exemplifies the results of a run of the algorithm applied to a test pair of sequences. Given appropriate initial gap states, the same algorithm as shown in Figure 1(B) is applicable for calculation of the score associated with a limited range of alignment. Use of this method in group-to-group alignment is described later.

Let C be an alignment between groups A and B. Since the gaps newly inserted into C do not affect $S_{k,l}$ for intra-group

(A)

	0	1	2	3	4	5	6	7
		S N A	– A A	– L –	– D G	– G –	– V –	G – K
0	0 (3)	72 (2)	108 (2)	144 (2)	162 (2)	180 (2)	198 (2)	216 (2)
1	AS 72 (1)	-7 (3)	65 (2)	101 (2)	119 (2)	137 (2)	155 (2)	173 (2)
2	CA 96 (1)	41 (1)	17 (3)	77 (2)	95 (2)	113 (2)	131 (2)	149 (2)
3	DM 108 (1)	53 (1)	41 (1)	**53 (3)**	77 (2)	95 (2)	113 (2)	131 (2)
4	G– 132 (1)	77 (1)	77 (1)	54 (3)	**77 (3)** 84 (2)	107 (2) 102 (2)	125 (2) 120 (2)	143 (2) 138 (2)
5	F– 144 (1)	89 (1)	89 (1)	78 (1)	79 (3)	**106 (3)** 103 (2)	126 (3) 130 (2) 121 (2)	139 (2)
6	V– 156 (1)	101 (1)	101 (1)	90 (1)	97 (1)	98 (3)	**120 (3)** 117 (3) 122 (2)	144 (2) 140 (2)
7	H– 180 (1)	137 (1)	133 (3) 137 (1)	127 (3) 138 (1)	123 (3)	143 (3) 153 (2) 146 (1)	137 (3)	**152 (3)** 154 (3)

(B)



(C)

```
A C – D G F V H
S A – M – – – –
. . . . . . . . . . . . . .
S – – – – – – – G
N A L D G – V –
A A – G – – – K
```

(D)

```
A C D G F V H
S A M – – – –
. . . . . . . . . . . .
S – – – – – – G
N A L D G V –
A A – G – – K
```

Fig. 2. An example of execution of algorithms C and D. (A) The matrix is filled with the retained candidates represented by the values for $D_{i,j}$ and the direction codes in parentheses computed with algorithm D to align $A = \frac{\text{ACDGFVH}}{\text{SAM - - - -}}$ and $B = \begin{smallmatrix}\text{S - - - - - G}\\ \text{NALDGV -}\\ \text{AA - G - - K}\end{smallmatrix}$. The bold-face candidates contribute to the optimal alignment shown in (D). Underlined are the candidates chosen by algorithm C, which yields the alignment shown in (C). Only some of the candidates generated by algorithm C are shown. (B) Path graph in which the broken line and the thick line correspond to alignments (C) and (D) respectively. The parameter values used are: $d(x,y) = -$MDM78 (Dayhoff *et al.*, 1978), $u = 6$, and $v = 6$. The same set of parameters is used throughout this paper.

sequences, i.e. the *k*th and the *l*th sequences both belonging to either group *A* or group *B*, the SP-score of *C* can be written as

$$\mathrm{SP}(C) = \mathrm{SP}(A) + \mathrm{SP}(B) + \mathrm{SP}(A{\cdot}B) \qquad (3A)$$

where

$$\mathrm{SP}(A{\cdot}B) = \sum_{m=1}^{M} \sum_{n=1}^{N} S_{m,n} \qquad (3B)$$

Hence the problem of seeking for the optimal SP(C) is equivalent to that for finding the optimal SP($A{\cdot}B$).

It is well known that there is a one-to-one correspondence between any alignment between *A* and *B* and a path through an $(I + 1) \times (J + 1)$ rectangular matrix as exemplified in Figure 2(B). An edge in the path connects a node of the lattice to one of the three (lower, right or lower-right diagonal) adjacent nodes. A segment in a path consists of consecutive edges of the same direction. A diagonal segment from $(i - k, j - k)$ to (i,j) indicates that the units of *A* $\{\mathbf{a}_{i-k+1} \ldots \mathbf{a}_i\}$ match those of *B* $\{\mathbf{b}_{j-k+1} \ldots \mathbf{b}_j\}$, whereas a horizontal segment from $(i,j - k)$ to (i,j) or a vertical segment from $(i - k,j)$ to (i,j) indicates a gap of length *k*. We assume here that diagonal segments and non-diagonal segments alternate in a path, implying that a run of matches such as

$$\begin{matrix} a_i \ldots a_{i+k} \\ b_j \ldots b_{j+k} \end{matrix}$$

is always better than consecutive gaps in the opposite (groups of) sequences,

$$\begin{matrix} a_i \ldots a_{i+k}\Delta \ldots \Delta \\ \Delta \ldots \Delta\, b_j \ldots b_{j+k} \end{matrix}$$

or

$$\begin{matrix} \Delta \ldots \Delta\, a_i \ldots a_{i+k} \\ b_j \ldots b_{j+k}\,\Delta \ldots \Delta \end{matrix}$$

Let $D_{i,j}{}^{\alpha}$ denote the score associated to a subalignment between $\{\mathbf{a}_1 \ldots \mathbf{a}_i\}$ and $\{\mathbf{b}_1 \ldots \mathbf{b}_j\}$. The optimal alignment we are seeking is that associated with $\min_{\alpha}\{D^{\alpha}_{I,J}\}$. The superscript α indicates the direction from which the final segment of the alignment path came. More specifically, α = 1, 2 and 3 indicate that the path reaches from an upper node, a left node and an upper-left diagonal node respectively. We will use an asterisk to distingush an element(s) in the subalignment from that in the original group of sequences. Thus, $\mathbf{a}_{ij}{}^{*} = \mathbf{a}_i$ when $\alpha = 1$ or 3, and $\mathbf{a}_{ij}{}^{*} = \Delta$ when $\alpha = 2$, where Δ indicates a vector whose elements are all nulls. Similarly, $\mathbf{b}_{ij}{}^{*} = \mathbf{b}_j$ when $\alpha = 2$ or 3, and $\mathbf{b}_{ij}{}^{*} = \Delta$ when $\alpha = 1$.

I discuss below four group-to-group sequence alignment algorithms, algorithms A–D. All the algorithms follow the dynamic programming paradigm (Needleman and Wunsch, 1970) formulated by Waterman *et al.* (1976) and improved in efficiency by Gotoh (1982). However, several generalizations and modifications are made to deal with pre-existing gaps. Algorithm A is the simplest and least accurate, algorithm D, the final version, gives an exact solution in terms of optimizing SP(*A*·*B*), and algorithms B and C have intermediate features.

Algorithm A

The simplest way to calculate $D_{ij}{}^{\alpha}$ is to use the conventional $O(L^2)$ dynamic programming algorithm (Gotoh, 1982) regarding each internal null as an ordinary sequence element:

$$D^1_{ij} = \min\{D^3_{i-1j} + V,\ D^1_{i-1j}\} + d(\mathbf{a}_i,\Delta) \qquad (4A)$$

$$D^2_{ij} = \min\{D^3_{ij-1} + V,\ D^2_{ij-1}\} + d(\Delta,\mathbf{b}_j) \qquad (4B)$$

$$D^3_{ij} = \min_{\beta=1}^{3} \{D^{\beta}_{i-1j-1}\} + d(\mathbf{a}_i,\mathbf{b}_j) \qquad (4C)$$

where $V \equiv MNv$ and $d(\mathbf{a}^*_{ij}, \mathbf{b}^*_{ij}) = \sum_{m=1}^{M} \sum_{n=1}^{N} d(a^*_{m,ij}, b^*_{n,ij})$.

Equations (4A)–(4C) or their minor variants appear to be adopted by some existing iterative multiple-sequence alignment methods (e.g. Barton and Sternberg, 1987; Higgins and Sharp, 1988). The opening of a gap is equally penalized irrespective of the locations of the pre-existing gaps, and hence on practical application this algorithm fails to minimize the SP-score in most cases.

Algorithm B

Without great complexity we can more accurately assess opening of gaps accompanying the edge that connects an immediately preceding node to the current node. Consider a short alignment

$$\begin{matrix} a_{i-1} & a_i \\ b_{j-1} & b_j \end{matrix}$$

which corresponds to the path segment from $(i-2, j-2)$ to (i,j). As before, let $q_{m,i}$ be the Boolean value indicating $a_{m,i} = \Delta$. Similarly $r_{n,j} = (b_{n,j} = \Delta)$. As indicated in Figure 1(A), we detect opening of a new gap if $(1 - r_{n,j-1})\cdot r_{n,j}\cdot(1 - q_{m,i})$ or $(1 - q_{m,i-1})\cdot q_{m,i}\cdot(1 - r_{n,j})$. When $q_{m,i-1}\cdot r_{n,j-1}$, additional upstream information is needed to detect a new gap explicitly. This is considered later in connection with algorithms (C) and (D). Here we statistically estimate the number of new gaps accompanying the edge, $g_{ij}{}^{33}$, with a parameter γ^{33} $(0.0 \le \gamma^{33} \le 1.0)$:

$$g^{33}_{ij} = \sum_{m=1}^{M} \sum_{n=1}^{N} [(1 - r_{n,j-1} + \gamma^{33} q_{m,i-1} r_{n,j-1})(1 - q_{m,i}) r_{n,j} + (1 - q_{m,i-1} + \gamma^{33} q_{m,i-1} r_{n,j-1}) q_{m,i} (1 - r_{n,j})] \qquad (5)$$

The superscript '33' indicates the directions ('3' means diagonal) of the last two edges in the alignment path. By replacing *q*s and *r*s in Equation (5) with their asterisked forms, we get $g_{ij}{}^{\beta\alpha}$ for the other six combinatory directions of the last two edges. Now equations (4A)–(4C) may be modified as:

$$D^1_{ij} = \min_{\beta=1,3} \{D^{\beta}_{i-1j} + v\cdot g^{\beta 1}_{ij}\} + d(\mathbf{a}_i,\Delta) \qquad (6A)$$

$$D^2_{ij} = \min_{\beta=2,3} \{D^{\beta}_{ij-1} + v\cdot g^{\beta 2}_{ij}\} + d(\Delta,\mathbf{b}_j) \qquad (6B)$$

$$D^3_{ij} = \min_{\beta=1}^{3} \{D^{\beta}_{i-1j-1} + v\cdot g^{\beta 3}_{ij}\} + d(\mathbf{a}_i,\mathbf{b}_j) \qquad (6C)$$

If each group of sequences has been transformed into 'profile' (Gribskov *et al.*, 1987) by pretreatment, $d(\mathbf{a}_{ij}{}^{*}, \mathbf{b}_{ij}{}^{*})$ and $g_{ij}{}^{\beta\alpha}$ can be computed with fixed steps. Hence the computational complexity of algorithm B is $O(L^2)$, which is a great advantage of this algorithm when either group of sequences contains many members.

Algorithm C

Algorithm C is similar to algorithm B, but we now hold the current gap staes so that new gaps accompanying any edge should be detected explicitly. Consider again a subalignment between $\{\mathbf{a}_1 \ldots \mathbf{a}_i\}$ and $\{\mathbf{b}_1 \ldots \mathbf{b}_j\}$. Let us define $Q_{m,ij}{}^{\alpha}$ as the number of consecutive nulls including and immediately preceding the element of $a_{m,ij}{}^{*}$ within the subalignment. $R_{n,ij}{}^{\alpha}$ is defined analogously to $b_{n,ij}{}^{*}$. We can obtain $Q_{m,ij}{}^{\alpha}$ and $R_{n,ij}{}^{\alpha}$ by recursion relations in which $q_{m,i}$ in equation (2) is replaced by $q_{m,ij}{}^{*}$ and $r_{n,ij}{}^{*}$ respectively. The number of new gaps accompanying the last edge is calculated by the rules listed in Figure 1(A). Upon implementation we need to memorize only a single column (or row) of $D_{ij}{}^{\alpha}$, $Q_{m,ij}{}^{\alpha}$ and $R_{n,ij}{}^{\alpha}$ matrices, and hence the $O[(M + N)\cdot\min(I,J)]$ space, apart from that possibly used for traceback, is sufficient to run this algorithm. The computational complexity of this algorithm is $O(MNL^2)$.

Algorithm D

Although algorithm C runs well for most applications, it still fails to optimize the SP-score under some subtle conditions. An example is shown in Figure 2, where algorithm C produces an alignment (alignment C) whose SP-score is worse by two than that of the optimal alignment (alignment D). This kind of error can occur when an internal gap in one group matches a new or pre-existing gap in the other group so that assignment of the gap opening penalty is postponed until the end of either of the gaps. Because algorithm C judges the most proper path at each node based on the local information retained by the last two edges, the correct path can be discarded during this 'moratorium' period. In the example shown in Figure 2, algorithm C chooses the path (5,4) → (5,5) → (6,6) at the node (6,6) because $D_{6,6}{}^3 = 117$ is better than 120. Consequently, the path (4,4) → (5,5) → (6,6) that should finally reach the optimal score is closed at this node; the relative scores associated to the progenies of these paths should be reversed at the last corner (7,7) by the retarded gap cost assigned to the progeny of the former path. To achieve the global optimization, we must retain additional paths that do not contribute to the locally optimal subalignments. Obviously it is impractical to keep all the alternative paths because of the large computation time and space required. However, we can design nearly quadratic algorithms with the help of the 'candidate list paradigm' first introduced into alignment algorithms by Waterman (1984) and later refined by Miller and Myers (1988) and Galil and Giancarlo (1989) to utilize concave gap-weighting functions.

In algorithms B and C, we examined a fixed number of candidates at each node; three from the preceding diagonal node, two from the preceding horizontal node, and two from the preceding vertical node. In algorithm D the candidates are also nominated from those in the three preceding nodes, and their $D_{i,j}{}^\alpha$ values are evaluated just as in algorithm C. However, the number of candidates may vary, so we generalise the notion of the superscript in equations (4) and (6) to indicate alternative candidates at the node (i,j). Thus the alignment paths giving rise to $D_{i,j}{}^\alpha$ and $D_{i,j}{}^\beta$ may or may not end with the same direction. Then we compare the nominated candidates to each other, and retain only the minimum number of members that are enough to contribute to the optimal path leading to any future nodes. We eliminate a candidate, α, if its progenies' scores are predicted never to be better than the corresponding values of another nominee, β, since we intend to get not all but only a single optimal alignment. This process is equivalent to pruning in the backtracking technique.

The candidates to be retained are selected as follows. If two candidates, α and β, take the same path in the future, $\Delta D(K) = D^\alpha(K) - D^\beta(K)$ for the Kth node along the path is predicted to be:

$$\Delta D(K) = \Delta D_{i,j} + v\cdot\sum_{m=1}^{M}\sum_{n=1}^{N} X_{m,n}\{(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j}) - (Q^\beta_{m,i,j} \le R^\beta_{n,i,j})\} + v\cdot\sum_{m=1}^{M}\sum_{n=1}^{N} Y_{m,n}\{(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j}) - (Q^\beta_{m,i,j} \ge R^\beta_{n,i,j})\} \quad (7A)$$

where

$$X_{m,n} \equiv \sum_{k=1}^{K}\left(\prod_{l=1}^{k-1} q^*_{m,l}\, r^*_{n,l}\right)(1 - q^*_{m,k})\, r^*_{n,k} \quad (7B)$$

and

$$Y_{m,n} \equiv \sum_{k=1}^{K}\left(\prod_{l=1}^{k-1} q^*_{m,l}\, r^*_{n,l}\right) q^*_{m,k}(1 - r^*_{m,k}) \quad (7C)$$

are Boolean values (k or l indicates a future node rather than a coordinate position). With the help of equation 7, we can eliminate many candidates from the list. I show below four necessary conditions (criteria) for a candidate α to survive.

[1] Let λ be the candidate for which $D_{i,j}{}^\lambda$ is minimum among all candidates at node (i,j). Candidate λ passes the test unconditionally. For another candidate α to survive, at least one of the inequalities of

$$(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j}) < (Q^\lambda_{m,i,j} \le R^\lambda_{n,i,j}) \quad (8A)$$

and

$$(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j}) < (Q^\lambda_{m,i,j} \ge R^\lambda_{n,i,j}) \quad (8B)$$

$(1 \le m \le M,\ 1 \le n \le N)$ must be satisfied, since otherwise all the terms in the right member of equation (7A) are non-negative, so $D_{h,k}{}^\alpha \ge D_{h,k}{}^\lambda$ for any future nodes.

[2] Define

$$E^\alpha_{i,j} \equiv D^\alpha_{i,j} + v\sum_{m=1}^{M}\sum_{n=1}^{N}(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j})$$

and let μ be the candidate with which $E_{i,j}{}^\alpha \ge E_{i,j}{}^\mu$ for any candidates α that passed criterion [1]. Then,

$$\Delta D(K) = E^\alpha_{i,j} - E^\mu_{i,j} - v\sum_{m=1}^{M}\sum_{n=1}^{N}(1 - X_{m,n})\{(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j}) - (Q^\mu_{m,i,j} \le R^\mu_{n,i,j})\} + v\sum_{m=1}^{M}\sum_{n=1}^{N} Y_{m,n}\{(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j}) - (Q^\mu_{m,i,j} \ge R^\mu_{n,i,j})\} \quad (9A)$$

By the same reasoning as above, at least one of the inequalities of

$$(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j}) > (Q^\mu_{m,i,j} \le R^\mu_{n,i,j}) \quad (9B)$$

and

$$(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j}) < (Q^\mu_{m,i,j} \ge R^\mu_{n,i,j}) \quad (9C)$$

must be satisified for $\alpha \ne \mu$ to survive.

[3] Define

$$F^\alpha_{i,j} \equiv D^\alpha_{i,j} + v\sum_{m=1}^{M}\sum_{n=1}^{N}(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j})$$

and let ν be the candidate with which $F^\alpha_{i,j} \ge F_{i,j}{}^\nu$ for any candidate α that passed criteria [1] and [2]. The same logic as above leads to the inequalities of

$$(Q^\alpha_{m,i,j} \le R^\alpha_{n,i,j}) < (Q^\nu_{m,i,j} \le R^\nu_{n,i,j}) \quad (10A)$$

and

$$(Q^\alpha_{m,i,j} \ge R^\alpha_{n,i,j}) > (Q^\nu_{m,i,j} \ge R^\nu_{n,i,j}) \quad (10B)$$

at least one of which must be satisfied for $\alpha \neq \nu$ to survive.
[4] Define

$$G^{\alpha}_{i,j} \equiv D^{\alpha}_{i,j} + \nu \sum_{m=1}^{M} \sum_{n=1}^{N} \{(Q^{\alpha}_{m,i,j} \leq R^{\alpha}_{n,i,j}) + (Q^{\alpha}_{m,i,j} \geq R^{\alpha}_{n,i,j})\}$$

and let x be the candidate with which ${G_{i,j}}^{\alpha} \geq {G_{i,j}}^{x}$ for any candidate α. Then,

$$\Delta D(K) \geq D^{\alpha}_{i,j} - D^{x}_{i,j} - \nu \sum_{m=1}^{M} \sum_{n=1}^{N} \{X_{m,n}(Q^{x}_{m,i,j} \leq R^{x}_{n,i,j}) + Y_{m,n}(Q^{x}_{m,i,j} \leq R^{x}_{n,i,j})\} \geq D^{\alpha}_{i,j} - G^{x}_{i,j} \qquad (11)$$

since $X_{m,n}$, $Y_{m,n}$, $({Q_{m,i,j}}^{\alpha} \leq {R_{n,i,j}}^{\alpha})$ and $({Q_{m,i,j}}^{\alpha} \geq {R_{n,i,j}}^{\alpha})$ are Boolean (0 or 1). Therefore, if ${D_{i,j}}^{\alpha} \geq {G_{i,j}}^{x}$, $\alpha \neq x$ can be eliminated from the list. Conversely, α may survive only when ${D_{i,j}}^{\alpha} < {G_{i,j}}^{x}$.

The order of the tests affects the performance considerably, and the above order is not the best. It is recommended that criterion [4] should be tested first, since it is the least demanding of the four criteria. Criteria [1]–[3] are then examined successively. A single application of the suite of tests to each candidate at a node is usually sufficient to narrow down the candidates to be retained. However, nested suites of tests, first for subsets of the candidates and then for the survivors of the first tests, are useful to keep the size of the candidate list compact, when the number of candidates nominated from one direction exceeds a certain threshold, T. In the present implementation, T is set to be $M + N$ by default.

The computational complexity of algorithm D is difficult to estimate. Although the worst case is not known, the examples shown later suggest that in typical cases the average number of candidates at each node is slightly less than half the fixed number of candidates examined by algorithm C. $O(MN)$ operations are necessary to calculate ${D_{i,j}}^{\alpha}$ and other related quantities. The same order of operations is used for examinations of criteria [1]–[4]. Thus the total arithmetic operations used by algorithm D is close to $O(MNL^2)$ in typical cases.

Multiple sequence alignment

Two strategies were adopted to align multiple sequences based on the group-to-group pairwise alignment algorithms. The first is the most popular progressive method (e.g. Waterman and Perlwitz, 1984; Feng and Doolittle, 1987), in which the order of pairwise alignments is predetermined by a given binary tree. The second is the randomized iterative strategy recently proposed by Berger and Munson (1991). This strategy successively refines an alignment of a set of N sequences. In each iterative step, the sequences in the preliminary alignment are divided into two groups, where one of the 2^{N-1}-1 ways of division is randomly selected, the columns composed of nulls alone are removed from each group, and then the groups are realigned by a pairwise method. Equation (3A) ensures that the SP-score of the resultant alignment is never worse than the previous score so long as the optimal group-to-group pairwise alignment is obtained. Hence the iteration certainly converges. At this stage, the same SP-score is obtained for all the $2^{N-1}-1$ ways of division. The convergence is easily monitored by the use of a series of pseudo-random numbers with the period of 2^{N-1} generated by the mixed congruence method (Jansson, 1966).

Implementation

Algorithms A–D were implemented as optional functions in the alignment programs alp/aln (Gotoh, 1990), and one of these or another algorithm (Gotoh, 1990) is selected at the run time. In addition to the progressive method, a simpler 'pile up' method (e.g. Bains, 1986) may be chosen as an option of these programs. This method is equivalent to a special case of the progessive method, in which a single member is added to the previously aligned subgroup at each iteration step. The pile up method is useful for rapid preparation of a crude multiple sequence alignment as a seed for further refinement. The randomized iterative method was implemented in another pair of programs, rrp/rrn, that call the same alignment functions as alp/aln.

The accuracies and efficiencies of the four algorithms were examined with several families of protein sequences: the five serine proteases examined by Berger and Munson (1991), ten non-vertebrate globins (PIR codes: ggewa3, ggwn2c, ggicea, ggice7, gggaa, gglmf, ggnwlb, ggzlb, gggab, and ggnkib), eight enzymes related to galactose epimerase (GalE) (SWISS-PROT codes: GALX$SACCA, GALE$ECOLI, GALE$SALTY, GALX$KLULA[1–347], GALE$RAT, GALE$STRLI, GALE$STRTR, and RFBJ$SALSP), and ten bacteria-type cytochrome P450s (CYP55, 101, 103, 104, 105A-C, 106, 107A and 107B). The results of progressive alignment (Table I) give us an idea of the relative efficiencies of the four algorithms. Namely, algorithm A is about five times faster than the other algorithms. Although the relative speeds of algorithms B, C and D depend slightly on the test sequences and possibly on details of the implementation, there seems to be a general trade-off between speed and accuracy. Algorithm B is ~10% faster than algorithm C, and the latter is 10~20% faster than algorithm D. The SP-scores obtained with algorithms C and D were identical for the families examined except for the GalE-related enzymes. These scores were consistently better than those obtained with algorithms A and B. With the single exception of bacterial cytochrome P450, algorithm B produced better alignment than algorithm A. The SP-scores for well-conserved families (e.g. serine proteases) are relatively insensitive, but those for diversified families (e.g. non-vertebrate globins) are sensitive to the algorithm used. Thus the accurate algorithm is preferred to align distantly related sequences, although it is a little more demanding for both time and space.

Algorithm D involves additional routines to examine criteria [1–4] that are absent in algorithm C. Nevertheless, algorithm D takes only 10~20% more CPU time than algorithm C. This is due to compensation between the overhead of the test routines and the consequent reduction in the number of candidates. With algorithm C the numbers of candidates examined and retained at each node are fixed at seven and three respectively. These numbers are variable with algorithm D. Figure 2(A) lists the candidates retained at each node for a small example. More extensive examinations of the four families mentioned above revealed that the average numbers of nominated and retained candidates are fairly constant, i.e. 3.2 ± 0.3 (SD) and 1.4 ± 0.1 (SD) per node respectively, and virtually independent of the number of sequences in each group and the degree of sequence divergence between the two groups. These numbers are a little less than half those used for algorithm C. Similar results were observed with a larger set of group-to-group alignments performed in the randomized iterative strategy. Hence, the criteria [1–4] used in algorithm D appear to be efficient enough to achieve rigorous optimization of the alignment score with little loss of overall performance.

The $\gamma^{\beta\alpha}$ parameter values used in algorithm B were estimated from several test runs of algorithm C so that $g^{\beta\alpha}$ in equation (5) should fit the exact value best assuming $\gamma^{11} = \gamma^{22}$, $\gamma^{13} = \gamma^{23}$, and $\gamma^{31} = \gamma^{32}$. The estimated $\gamma^{\beta\alpha}$ values were similar, ranging from 0.6 to 0.7. Hence the calculations mentioned above were done with default values of $\gamma^{\beta\alpha} = 0.65$ for all seven combinatory directions. Similar examinations as those shown in Table I with several sets of $\gamma^{\beta\alpha}$ values indicated that the default values gave slightly, but significantly better results than the two extreme sets of $\gamma^{\beta\alpha} = 0$ for all $\beta\alpha$ and $\gamma^{\beta\alpha} = 1$ for all $\beta\alpha$. The results with intermediate $\gamma^{\beta\alpha}$ values were similar to those with the default values.

As shown in Figure 3, the randomized iteration in its original scheme fails to converge when algorithm A or B is used. Upward trajectories are frequently seen with algorithms A and B, and sometimes with algorithm C. (A smaller SP-score indicates better alignment in our scoring scheme). On the other hand, no upward trajectory is present when algorithm D is used. These observations are fully consistent with the expected accuracy of the four algorithms. To ensure convergence with algorithms A, B or C, the iteration scheme was slightly modified so that any iterative step with a larger score than the minimum of the previous steps was discarded. The results obtained with this modified scheme are summarized in Table II. Two families, i.e. the same serine proteases as examined previously and a subset of insect globins consisting of the first six members in the previous list, were subjected to the randomized iteration. Each iteration was initiated with either a gapless alignment or an alignment obtained by the progressive method. It is obvious that a randomized iteration converges fater and generally yields better alignments when initiated with a prealigned state, and when the more rigorous algorithms are used.

Discussion

There have been many papers describing methods for multiple sequence alignment by means of repeated use of pairwise alignment. Sometimes pairwise methods have been used to

Table I. Results of progressive alignment

Family	serine protease	bacterial P450	galE-related	insect globin
No. of members	5	10	8	10
Length	231.0 ± 7.8	406.5 ± 8.4	345.5 ± 43.2	148.2 ± 3.6
Divergence (%)	67.7 ± 5.8	70.7 ± 7.3	64.6 ± 15.0	79.1 ± 5.1
CPU time (s)				
A	6.1 (1.0)	31.3 (1.0)	29.9 (1.0)	10.9 (1.0)
B	31.1 (5.1)	151.6 (4.8)	141.2 (4.7)	47.4 (4.3)
C	26.8 (4.4)	175.2 (5.6)	162.4 (5.4)	51.0 (4.7)
D	30.4 (5.0)	201.1 (6.4)	185.8 (6.2)	62.4 (5.7)
SP-score				
A	−3254 (142)	−11822 (2037)	−3338 (766)	4187 (3491)
B	−3360 (36)	−11645 (2214)	−3703 (401)	2699 (2003)
C	−3360 (36)	−11865 (1994)	−3714 (390)	2637 (1941)
D	−3360 (36)	−11865 (1994)	−3742 (362)	2637 (1941)
Best score	−3396	−13859	−4104	696
Best score/site/pair	−138.6	−69.4	−32.5	8.9

Progressive aligment of members in each family was performed in a given topological order with the group-to-group alignment algorithm (A–D) specified. CPU time was measured on a SONY NEWS 830 which runs ten times slower than a SPARCstation2. The values in parentheses are the ratio of the time taken with the specified algorithm to that with algorithm A. The same set of parameter values as shown in the legend to Figure 2 were used to calculate the SP-scores. The best score for each family is that obtained by several series of iterative refinements (Table II). The differences between the best scores and the SP-scores obtained with the progressive method are indicated in parentheses. Best score/site/pair indicates overall relatedness among the members in the family.

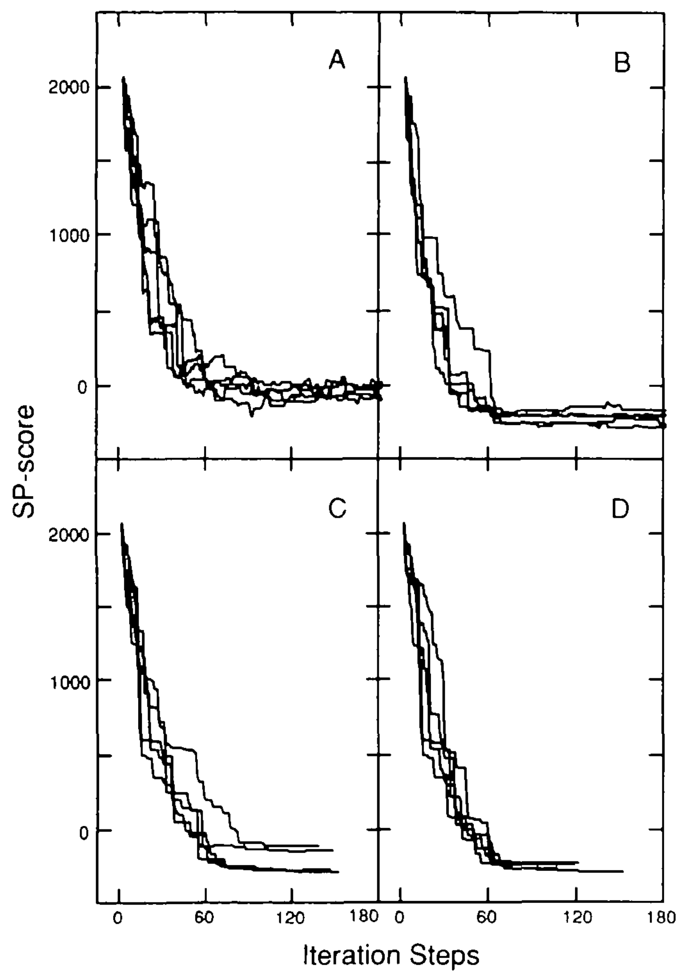


Fig. 3. Trajectories of SP-score in the course of randomized iteration applied to six insect globins. The symbol (A – D) given to each panel indicates the group-to-group alignment algorithm used. Five series of iterations were performed with each alignment algorithm initiated from the same gapless alignment. Iteration was broken off unless it converged up to the 180th step.

refine a multiple sequence alignment constructed by some means (Waterman and Perlwitz, 1984; Bains, 1986; Barton and Sternberg, 1987). In these methods, the refinement is made by pairwise alignment between a single member versus the consensus or the rest of the sequences of the preliminary multiple sequence alignment. Subbiah and Harrison (1989) extended the refinement strategy so that a series of pairwise alignments between various groups were performed. Recently Berger and Munson (1991) proposed another strategy, in which the members are divided into two groups in a random fashion and then the two groups are realigned by a pairwise method. The additive nature of the alignment score exhibited by equation (3) provides the key background to ensure the propriety of this strategy. However, the scoring scheme adopted by Berger and Munson does not have this additive nature, so continual improvement (more accurately non-deterioration) in the total score cannot be expected with their method. This is also true for all previously published methods in which no rigorous algorithm for optimizing objective score of single-to-group or group-to-group alignment was presented. The algorithm D shown above is the first rigorous algorithm suitable for any refinement strategies including the randomized iterative method.

It is not easy to compare the advantages and disadvantages of the iterative multiple sequence alignment methods and the space-limiting methods (Carrillo and Lipman, 1988; Altschul and Lipman, 1989; Spouge, 1989). When the sequences to be aligned are not extensively divergent and the final alignment contains a relatively small number of gaps, the space-limiting methods might be superior to the iterative methods in both speed and accuracy. On the contrary, the iterative methods are more practical in more difficult cases (i.e. when a larger number of more distantly related sequences containing many gaps are aligned), because the efficiency of the iterative methods is much less sensitive than the space-limiting methods to these unfavorable conditions. As exemplified in Table I, the merits

Table II. Results of randomized iterative method

Algorithm	Without prealignment			After progressive alignment		
	SP-score	Best score	No. of steps	SP-score	Best score	No. of steps
Serine proteases						
A	−3347.7 ± 15.3	−3376	51.1 ± 9.0	−3361.4 ± 2.1	−3364	37.6 ± 9.9
B	−3372.8 ± 14.1	−3390	52.3 ± 10.2	−3378.6 ± 6.0	−3391	34.9 ± 7.6
C	−3382.0 ± 11.4	−3396	46.0 ± 6.2	−3385.6 ± 5.2	−3392	31.1 ± 4.1
D	−3386.4 ± 8.5	−3396	50.3 ± 9.9	−3385.9 ± 5.1	−3396	31.1 ± 4.8
Insect globins						
A	− 68.5 ± 96.6	− 215	155.3 ± 53.3	− 129.7 ± 38.1	− 193	93.9 ± 27.1
B	− 175.1 ± 35.1	− 238	144.1 ± 39.4	− 208.2 ± 21 5	− 233	109.4 ± 37.5
C	− 188.1 ± 41.1	− 245	121.8 ± 18.9	− 224.7 ± 19.5	− 240	94.5 ± 19.6
D	− 193.6 ± 33.2	− 241	116.0 ± 22.7	− 228.1 ± 19.1	− 246	87.3 ± 18.7

Five serine proteases and six insect globins were subjected to a modified randomized iterative refinement method (Berger and Munson, 1991) with and without prior progressive alignment. The SP-score and the number of steps up to the convergence shown are the means ± SD of those obtained by 32 series of iteration in each case.

of these elaborate methods over the conventional progressive alignment methods are clearest in these difficult conditions. About an hour of CPU time was spent on SUN SPARCstation 2 to get the alignment of the ten non-vertebrate globins shown in Figure 4. It is unlikely that this quality of alignment could be obtained by the space-limiting methods with similar computer resources.

Algiorithm D is superior to algorithm C in its theoretical strictness. Moreover, algorithm D is more flexible to generalization, for instance, to incorporate non-linear gap-weighting functions (Miller and Myers, 1988; Galil and Giancarlo, 1989; Gotoh, 1990). An advantage of algorithm C is its stability; the number of operations and necessary memory space for a given input can be explicitly determined before the execution. Although the average performance of algorithm D is similar to that of algorithm C, its performance in bad cases is not known.

The major drawback of algorithms C and D is that the computation time grows in proportion to the product of the numbers of sequences in the two groups. Conversion of the character-based sequences into a numerical 'profile' vector (Gribskov *et al.*, 1987) helps to suppress the computational augmentation. Unfortunately, it seems very complicated, if not impossible, to tailor profile-based operations to the gap-weighting rule presented in Figure 1. The best solution now available is probably to use algorithm B, which is fully compatible with profile-based operations. Another merit of its use is that not only the primary sequence but also additional

```
  1 ---------SLSAAEADLAGKSWAPVFANKNA-----NGADFLVALFEKFPDSANFFADF  gggaa
  1 PIVDSGSVAPLSAAEKTKIRSAWAPVYSNYET-----SGVDILVKFFTSTPAAQEFFPKF  gglmf
  1 --------APLSADQASLVKSTWAQVRNS---------EVEILAAVFTAYPDIQARFPQF  ggice7
  1 ---------VATPAMPSNTDAQVAAVKGDWEK--IKGSGVEILYFFLNKFPGNFPMFKKL  ggicea
  1 ---------GLSAAQRQVIAATWKDIAGNDNG---AGVGKDCLIKHLSAHPQMAAVFGFS  ggnwlb
  1 ---------MLDQQTINIIKATVPVLKEHGVT-----ITTTFYKNLFAKHPEVRPLFDMG  ggzlb
  1 K-------KQCGVLEGLKVKSEWGRAYGSGHD--REAFSQAIWRATFAQVPESRSLFKRV  ggewa3
  1 D-------TCCSIEDRREVQALWRSIWSAEDTGRRTLIGRLLFEELFEIDGATKGLFKRV  ggwn2c
  1 ---------GLDGAQKTALKESWKVLGADGPT--MMKNGSLLFGLLFKTYPDTKKHFKHF  gggab
  1 PSVQ-GAAAQLTADVKKDLRDSWKVIGSDKKG-----NGVALMTTLFADNQETIGYFKRL  ggnklb
                          o                  oo   o         F

 47 KG-KSVADIKASPKLRDVSSRIFTRLNEFVNDAA---NAGKMSAMLSQFAKEHVGF-GVG  gggaa
 56 KGMTSADQLKKSADVRWHAERIINAVNDAVASMD---DTEKMSMKLRDLSGKHAKSFQVD  gglmf
 44 AG-KDVASIKDTGAFATHAGRIVGFVSEIIALIGNESNAPAVQTLVGQLAASHKAR-GIS  ggice7
 50 -G-NDLAAAKGTAEFKDQADKIIAFLQGVIEKLG--SDMGGAKALLNQLGTSHKAM-GIT  ggicea
 49 GA-SDPAVADLGAKVLAQIGVAVSHLGDEGKMVA---QMKAVGVRHKGYGNKHIKG-QYF  ggnwlb
 47 RQ-ESLEQPKALAMTVLAAAQNIENLPAILPAV----KKIAVKHCQAGVAAAHYPI--VG  ggzlb
 52 -----HGDHTSDPAFIAHAERVLGGLDIAISTLD---QPATLKEELDHLQVQHEGR-KIP  ggewa3
 54 ----NVDDTHSPEEF-AHVLRVVNGLDTLIGVLG---DSDTLNSLIDHLAEQHKARAGFK  ggwn2c
 50 DD-ATFAAMDTTGVGKAHGVAVFSGLGSMICSID---DDDCVEGLAKKLSRNHLAR-GVS  gggab
 55 ---GNVSQGMANDKLRGHSITLMYALQNFIDQLD---NTDDLVCVVEKFAVNHITR-KIS  ggnklb
                          o  @           j          o   H     o

102 SAQFENVRSMFPGFVASVAA---PPAGADAWT----KLFGLII-DALKAAGK----      gggaa
113 PQYFKVLAAVIADTVAA---------GDAGFE----KLMSMIC-ILLRSAY-----      gglmf
102 QAQFNEFRAGLVSYVSSNVA-W-NAAAESAWT----AGLDNIF-GLLFAAL-----      ggice7
105 KDQFDQFRQALTELLGNLG----FGGNIGAWN----ATVDLMF-HVIFNALDGTPV      ggicea
104 EPLGASLLSAMEHRIGGKM----NAAAKDAWA----AAYADIS-GALISGLQS---      ggnwlb
100 QELLGAIKEVLGDAATDDILDA-WGKAYGVIA----DVFIQVE-ADLYAQAVE---      ggzlb
103 DNYFDAFKTAILHVVAAQLGERCYSNNEEIHDAIACDGFARVLPQVLERGIKGHH-      ggewa3
106 TVYFKEFGKALNHVLPEVA----SCFNPEAWN----HCFDGLV-DVISHRIDG---      ggwn2c
105 AADFKLLEAVFKEFLDEAT----QRKATDAQK----DADGALL-TMLIKAHV----      gggab
108 AAEFGKINGPIKKVLASK-------NFGDKYA----NAWAKLV-AVVQAAL-----      ggnklb
          o   o                              @    @
```

Fig. 4. Multiple sequence alignment of ten non-vertebrate globins obtained by the randomized iterative method in conjunction with the group-to-group alignment algorithm D. The iteration was initiated with an alignment obtained by a progressive method. The heme-binding histidines were not correctly aligned at this initial stage but reasonably aligned after the refinement. The SP-score was improved by 1941 in this refinement process (Table I). Completely conserved sites are indicated by the amino-acid codes in capital letters. Conserved hydrophobic (o), hydrophilic (j) and large aliphatic @ sites are also indicated beneath the alignment.

information, such as propensities for secondary structure formation and hydropathy indices, can easily be incorporated. I have recently shown (Gotoh, 1992) that structure-related information plays an essential role in reliable alignment of very distantly related pairs of protein families.

Algorithm A is obviously the best choice when there is no internal gap in any group of sequences. Hence the four algorithms described above have their own advantages and disadvantages. I am planning to make a system in which the most suitable algorithm is automatically selected at run time depending on the features of the input sequences.

Acknowledgement

This work was partly supported by a Grant-in-Aid for Scientific Research on Priority Areas, 'Genome Informatics', from the Ministry of Education, Science and Culture of Japan.

References

Altschul,S.F. (1989) Gap costs for multiple sequence alignment. *J. Theor. Biol.*, **138**, 297–309.

Altschul,S.F. and Lipman,D.J. (1989) Trees, stars, and multiple biological sequence alignment. *SIAM J. Appl. Math.*, **49**, 197–209.

Bains,W. (1986) Multan: a program to align multiple DNA sequences. *Nucleic Acids Res.*, **14**, 159–177.

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

Berger,M.P. and Munson,P.J. (1991) A novel randomized iterative strategy for aligning multiple protein sequences. *Comput. Applic. Biosci.*, **7**, 479–484.

Carrillo,H. and Lipman,D. (1988) The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, **48**, 1073–1082.

Corpet,F. (1988) Multiple sequence alignment with hierarchical clustering. *Nucleic Acids Res.*, **6**, 10881–10890.

Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C. (1978) A model of evolutionary change in proteins. In Dayhoff,M.O. (ed.), *Atlas of Protein Sequence and Structure*. National Biomedical Research Foundation, Washington, DC, Vol. 5, Suppl. 3, pp. 345–352.

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

Galil,Z. and Giancarlo,R. (1989) Speeding up dynamic programming with applications to molecular biology. *Theor. Comput. Sci.*, **64**, 107–118.

Gotoh,O. (1982) An improved algorithm for matching biological sequences. *J. Mol. Biol.*, **162**, 705–708.

Gotoh,O. (1990) Optimal sequence alignment allowing for long gaps. *Bull. Math. Biol.*, **52**, 359–373.

Gotoh,O. (1992) Substrate recognition sites in cytochrome P450 family 2 (CYP2) proteins inferred from comparative analyses of amino acid and coding nucleotide sequences. *J. Biol. Chem.*, **267**, 83–90.

Gotoh,O. and Fujii-Kuriyama (1989) Evolution, structure, and gene regulation of cytochrome P-450. In Ruckpaul,K. and Rein,H. (eds), *Frontiers in Biotransformation*. Akademie Verlag, Berlin, Vol. 1, pp. 195–243.

Gribskov,M., McLachlan,A.D. and Eisenberg,D. (1987) Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. USA*, **84**, 4355–4358.

Hein,J. (1989) A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given. *Mol. Biol. Evol.*, **6**, 649–668.

Higgins,D.G. and Sharp,P.M. (1989) Fast and sensitive multiple sequence alignments on a microcomputer. *Comput. Applic. Biosci.*, **5**, 151–153.

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.

Jansson,B. (1966) *Random Number Generators*. Almqvist-Wiksell, Stockholm.

Lesk,A.M., Levitt,M. and Chothia,C. (1986) Alignment of the amino acid

sequences of distantly related proteins using variable gap penalties. *Protein Engng.*, **1**, 77–78.
Murata,M., Richardson,J.S. and Sussman,J.L. (1985) Simultaneous comparison of three protein sequences. *Proc. Natl. Acad. Sci. USA*, **85**, 3073–3077.
Miller,W. and Myers,E.W. (1988) Sequence comparison with concave weighting functions. *Bull. Math. Biol.*, **50**, 97–120.
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.
Sankoff,D. and Cedergren,R.J. (1983) Simultaneous comparison of three or more sequences related by a tree. In Sankoff,D. and Kruskal,J.B. (eds), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, MA, pp. 253–263.
Santibánez,M. and Rohde,K. (1987) A multiple alignment program for protein sequences. *Comput. Applic. Biosci.*, **3**, 111–114.
Smith,R.F. and Smith,T.F. (1990) Automatic generation of primary sequence patterns from sets of related protein sequences. *Proc. Natl. Acad. Sci. USA*, **87**, 118–122.
Sobel,E. and Martinez,H.M. (1986) A multiple sequence alignment program. *Nucleic Acids Res.*, **14**, 363–374.
Spouge,J.L. (1989) Speeding up dynamic programming algorithms for finding optimal lattice paths. *SIAM J. Appl. Math.*, **49**, 1552–1566.
Subbiah,S. and Harrison,S.C. (1989) A method for multiple sequence alignment with gaps. *J. Mol. Biol*, **209**, 539–548.
Taylor,P. (1984) A fast homology program for aligning biological sequences. *Nucleic Acids Res.*, **12**, 447–455.
Taylor,W.R. (1988) A flexible method to align large numbers of biological sequences. *J. Mol. Evol.*, **28**, 161–169.
Vingron,M. and Argos,P. (1989) A fast and sensitive multiple sequence alignment algorithm. *Comput. Applic. Biosci.*, **5**, 115–121.
Waterman,M.S. (1984) Efficient sequence alignment algorithms. *J. Theor. Biol.*, **108**, 333–337.
Waterman,M.S. (1986) Multiple sequence alignment by consensus. *Nucleic Acids Res.*, **14**, 9095–9102.
Waterman,M.S. and Perlwitz,M.D. (1984) Line geometries for sequence comparisons. *Bull. Math. Biol.*, **46**, 567–577.
Waterman,M.S., Smith,T.F. and Beyer,W.A. (1976) Some biological sequence metrics *Adv. Math.*, **20**, 367–387.

Received on August 5, 1992, accepted on October 15, 1992

Circle No. 16 on Reader Enquiry Card