CABIOS

Vol.6. no.4. 1990
Pages 373 – 381

A space-efficient algorithm for local similarities

Xiaoqiu Huang, Ross C.Hardison[1] and Webb Miller[2]

Abstract

Existing dynamic-programming algorithms for identifying similar regions of two sequences require time and space proportional to the product of the sequence lengths. Often this space requirement is more limiting than the time requirement. We describe a dynamic-programming local-similarity algorithm that needs only space proportional to the sum of the sequence lengths. The method can also find repeats within a single long sequence. To illustrate the algorithm's potential, we discuss comparison of a 73 360 nucleotide sequence containing the human β-like globin gene cluster and a corresponding 44 594 nucleotide sequence for rabbit, a problem well beyond the capabilities of other dynamic-programming software.

Introduction

Local sequence alignment algorithms seek only conserved regions, whereas global methods align entire sequences including unconserved regions. A number of dynamic-programming algorithms for local sequence alignment have been developed (Smith and Waterman, 1981; Goad and Kanehisa, 1982; Sellers, 1984; Gotoh, 1987; Waterman and Eggert, 1987; Hall and Myers, 1988). Indeed Waterman (1988) states that such a method is 'probably the most useful dynamic programming algorithm for current problems in biology'.

Alternatively, far faster heuristic methods are known (Karlin *et al.*, 1988; Pearson and Lipman, 1988; Pearson, 1990). Indeed, there exist programs that produce useful, though limited, information and that run 10^4 times faster than the program reported here (Altschul *et al.*, 1990). Although they are very slow by comparison, dynamic-programming algorithms have the advantage of producing high-resolution alignments that are guaranteed to optimize a well-understood alignment score. The program presented in this paper first computes a highest-scoring alignment, then an alignment that is highest-scoring among all alignments containing no aligned pair from the first alignment, then a highest-scoring alignment that is disjoint from the first two, and so on. Alignment scores depend on only three parameters, which can be freely adjusted to achieve some desirable effect.

Department of Computer Science, Michigan Technological University, Houghton, MI 49931, [1]Department of Molecular and Cell Biology and [2]Department of Computer Science, The Pennsylvania State University, University Park, PA 16802, USA

Frequently, execution efficiency is less important than having a program that is trivial to operate and that produces easily interpreted results. For example, we describe below the comparison of a 73 kb DNA sequence with a 44 kb sequence to find the 100 best non-intersecting local similarities. Execution time was not an important constraint. Rather, we wanted the comparison to proceed without requiring intervention from us and to produce results that optimize some explicit measure of similarity. In this case, the difficulty with dynamic programming approaches is that existing software retains a huge 'traceback' matrix. For our two sequences, billions of bytes of storage would be required, more than all the disk space available to us.

The software described in this paper easily solved the problem on our workstation. The program ran unobtrusively (i.e. 'in the background' and at low priority), cranking out four or five alignments per day while the workstation performed its regular duties. Recent improvements (X.Huang and W.Miller *Advances in Applied Mathematics* in press) allow all 100 alignments to be computed overnight. In addition, the program and all its data fit handily in memory; the only use of disk space was to hold computed alignments.

Our linear-space local similarity algorithm combines the linear-space global alignment algorithm of Myers and Miller (1988) with techniques of Waterman and Eggert (1987). The algorithm and a variant for finding repeated regions in a single sequence are described in detail below. We then discuss the application of our method to the alignment of long DNA sequences.

System and methods

The program described in this paper is written in C and developed on a Sun4/260 running SunOS Unix. The code is portable and is designed to run on a workstation or a large personal computer.

Algorithm

Before launching into a precise description of the linear-space algorithm, we give an informal sketch. Conceptually, aligning a M-element sequence with a N-element sequence amounts to determining the entries of a $(M + 1) \times (N + 1)$ matrix of alignment scores, where values are computed systematically from the upper left corner of the matrix to the lower right. To find just the score of an optimal alignment it is sufficient to save a few rows (or columns) of the matrix; traditional techniques to explicitly produce an optimal alignment require, in essence, that the entire matrix be retained.

Smith and Waterman (1981) arrange the computation so that the largest value in the matrix corresponds to the right end of an optimal local alignment. Starting at the position of this maximum value and inverting the process, i.e. computing from lower right to upper left, we find the left end of our optimal local alignment. Once these endpoints are known, the problem is reduced to finding an optimal global alignment on a submatrix, and the procedure of Myers and Miller (1988) can be invoked.

With an optimal local alignment in hand, we can apply a technique of Waterman and Eggert (1987) and repeat the above steps to compute the second best local alignment, where we do not allow a pair of sequence positions that correspond under the first alignment to correspond under the second alignment. Continuing, we can find the k best non-intersecting local alignments for any $k > 0$.

Finding a best local alignment

When scoring an alignment between a region (i.e. a consecutive substring) of $a_1a_2 \cdots a_M$ and a region of $b_1b_2 \cdots b_N$, a bonus $v(a_i,b_j)$ is added for every aligned pair (a_i,b_j) and a penalty $q + rx$ is subtracted for each gap of length x. Here q can be thought of as the cost of opening up a gap, while each symbol in the gap costs r. The following development parallels that in Myers and Miller (1988).

Define

$H(i,j)$ = larger of 0 and maximum score of any local alignment that ends at (i,j)
$E(i,j)$ = maximum score of any local alignment that ends at (i,j) with a_i in a gap
$F(i,j)$ = maximum score of any local alignment that ends at (i,j) with b_j in a gap

Since $E(0,j)$ and $F(i,0)$ can be defined freely, they are chosen to make the recurrences correct on the boundary. This gives the following equations.

$$H(i,j) = \begin{cases} \max\{0,E(i,j),F(i,j),H(i-1,j-1)+v(a_i,b_j) & \text{if } i>0 \text{ and } j>0 \\ 0 & \text{if } i=0 \text{ or } j=0 \end{cases}$$

$$E(i,j) = \begin{cases} \max\{E(i-1,j),H(i-1,j) - q\} - r & \text{if } i>0 \text{ and } j>0 \\ -q & \text{if } i=0 \text{ and } j>0 \end{cases}$$

$$F(i,j) = \begin{cases} \max\{F(i,j-1),H(i,j-1) - q\} - r & \text{if } i>0 \text{ and } j>0 \\ -q & \text{if } i>0 \text{ and } j=0 \end{cases}$$

A best local alignment ends at (I,J) if

$$H(I,J) = \max\{H(i,j) : 1 \le i \le M \text{ and } 1 \le j \le N\}$$

$H(I,J)$ is the score of that best local alignment. Figure 1(A) gives an algorithm that uses the above formulas to locate such a (I,J) in linear space. Since only the maximum similarity score is required, it is sufficient to save only the most recently computed rows of the H and E. The vectors HH and EE are

```
vectors HH[0 . . N],EE[0 . . N]
scalars f, h, p

score ← I ← J ← 0
for j ← 1 to N do
  { HH(j) ← 0
    EE(j) ← − q
  }
for i ← 1 to M do
  { h ← p ← 0
    f ← − q
    for j ← 1 to N do
      { f ← max {f,h − q} − r
        EE(j) ← max {EE(j),HH(j) − q} − r
        h ← max {0,EE(j),f,p + v(a_i,b_j)}
        p ← HH(j)
        HH(j) ← h
        if h > score then
          { score ← h
            I ← i
            J ← j
          }
      }
  }
write 'a best alignment ends at' (I,J)
```

Fig. 1(A) Forward pass to find the end-point.

```
vectors HH[0 . . N],EE[0 . . N]
scalars f, h, p

cost ← K ← L ← 0
for j ← J downto 1 do
  { HH(j) ← − 1
    EE(j) ← − 1
  }
for i ← I downto 1 do
  { h ← f ← − 1
    p ← if i = I then 0 else − 1
    for j ← J downto 1 do
      { f ← max {f,h − q} − r
        EE(j) ← max {EE(j),HH(j) − q} − r
        h ← max {EE(j),f,p + v(a_i,b_j)}
        p ← HH(j)
        HH(j) ← h
        if h > cost then
          { cost ← h
            K ← i
            L ← j
            if cost ≥ score then
              goto found
          }
      }
  }
found: write 'the best alignment starts at' (K,L)
```

Fig. 1(B) Reverse pass to find the start-point.

used for this purpose. If rows $i - 1$ of H and E are stored in HH and EE respectively, then rows i are computed by overwriting values for rows $i - 1$ in a left-to-right order using three scalars, f, h and p. Specifically, if $i,j > 0$, then before the start of the jth iteration of the inner **for** loop, we have

$$HH(k) = \begin{cases} H(i,k) & \text{if } k < j \\ H(i - 1,k) & \text{if } k \ge j \end{cases}$$

$$EE(k) = \begin{cases} E(i,k) & \text{if } k < j \\ E(i-1,k) & \text{if } k \geq j \end{cases}$$

$$f = F(i,j-1)$$
$$h = H(i,j-1)$$
$$p = H(i-1,j-1)$$

With this loop-invariant condition in mind, Figure 1(A) is readily understood. Note that there is no need to have a vector for the F array since the F values on row i do not depend on those on row $i - 1$. In Figure 1, the scalar f is used to save the last F value computed on the current row.

Once we have determined that a best local alignment ends at (I,J), the next step is to locate the start-point of that alignment. That is, we want to find $K \in [1,I]$ and $L \in [1,J]$ such that some (global) alignment of $a_K a_{K+1} \cdots a_I$ and $b_L b_{L+1} \cdots b_J$ achieves a score $H(I,J)$. The algorithm in Figure 1(B) finds such a (K,L). Notice that the algorithms in Figure 1(A) and (B) are not symmetric. Because several best alignments may exist, a point with the maximum value in a reverse pass may be the start-point of an alignment not ending at (I,J). To guarantee that the correct start-point is found, we set the initial values of H, E and F to -1, except for the value of H on the origin (i.e. $p = 0$ when $i = I$). In addition, 0 is removed from the maximum expression in computing H. It is not difficult to show that the (K,L) computed by the algorithm in Figure 1(B) is indeed the desired point. Observe that in the reverse pass, the computation can stop when $cost \geq score$.

Alternatively, start-points of optimal alignments can be determined during the initial forward pass. The idea is that when a matrix value $X(i,j)$ is computed (X is H, E or F), the left endpoint of an alignment ending at (i,j) with score $X(i,j)$ is computed accordingly. Indeed, it is possible to keep track of the numbers of insertions and deletions in an optimal alignment (Gotoh, 1987). Since this method spends extra time on each matrix entry, and since an optimal local alignment is normally much shorter than the orginal sequences, the one-pass method is slower than our two-pass method for computing a single optimal alignment.

Once the start-point (K,L) and end-point (I,J) of a best alignment are determined, the alignment can be recovered by applying the method of Myers and Miller (1988) to substrings $a_K a_{K+1} \cdots a_I$ and $b_L b_{L+1} \cdots b_J$. Since that procedure minimizes a distance measure instead of maximizing a similarity score, a conversion is required. Parameters w, g and h for the distance measure are calculated by the transformations

$$w(a,b) = v_{\max} - v(a,b) \text{ for all pairs } (a,b)$$
$$g = q$$
$$h = r + \tfrac{1}{2} v_{\max}$$

where $v_{\max} = \max_{(a,b)} v(a,b)$ (Smith *et al.*, 1981). The

```
vectors $HH[0 .. M], EE[0 .. M]$
scalars $f, h, p$

$score \leftarrow I \leftarrow J \leftarrow 0$
for $j \leftarrow 1$ to $N$ do
  { $HH(j) \leftarrow 0$
    $EE(j) \leftarrow -q$
  }
for $i \leftarrow 1$ to $M - 1$ do
  { $h \leftarrow 0$
    $p \leftarrow HH(i)$
    $f \leftarrow -q$
    for $j \leftarrow i + 1$ to $M$ do
      { $f \leftarrow \max\{f, h - q\} - r$
        $EE(j) \leftarrow \max\{EE(j), HH(j) - q\} - r$
        $h \leftarrow \max\{0, EE(j), f, p + v(a_i, b_j)\}$
        $p \leftarrow HH(j)$
        $HH(j) \leftarrow h$
        if $h > score$ then
          { $score \leftarrow h$
            $I \leftarrow i$
            $J \leftarrow j$
          }
      }
  }
write 'a best alignment ends at' (I,J)
```

Fig. 2(A) Forward pass to find the end-point.

```
vectors $HH[0 .. M], EE[0 .. M]$
scalars $f, h, p$

$cost \leftarrow K \leftarrow L \leftarrow 0$
for $j \leftarrow J$ downto 1 do
  { $HH(j) \leftarrow -1$
    $EE(j) \leftarrow -1$
  }
for $i \leftarrow I$ downto 1 do
  { $h \leftarrow f \leftarrow -1$
    $p \leftarrow$ if $i = I$ then 0 else $-1$
    for $j \leftarrow J$ downto $i + 1$ do
      { $f \leftarrow \max\{f, h - q\} - r$
        $EE(j) \leftarrow \max\{EE(j), HH(j) - q\} - r$
        $h \leftarrow \max\{EE(j), f, p + v(a_i, b_j)\}$
        $p \leftarrow HH(j)$
        $HH(j) \leftarrow h$
        if $h > cost$ then
          { $cost \leftarrow h$
            $K \leftarrow i$
            $L \leftarrow j$
            if $cost \geq score$ then
              goto found
          }
      }
  }
found: write 'the best alignment starts at' (K,L)
```

Fig. 2(B) Reverse pass to find the start-point.

Myers–Miller algorithm requires linear space and quadratic time, so a single best local alignment can be found in $O(M + N)$ space and $O(MN)$ time.

Finding k best non-intersecting alignments

To find several non-intersecting alignments, the algorithm of Figure 1 is applied repeatedly with the slight twist that aligned pairs (a_i,b_j) are excluded from subsequent alignments.

Specifically, if (a_i,b_j) is already used, then $H(i-1,j-1) + v(a_i,b_j)$ is removed from the maximum expression in computing $H(i,j)$ in all subsequent computations, i.e. the forward pass, the reverse pass and the Myers–Miller procedure.

For each row $i \in [1,M]$ we maintain a sorted list of all $j \in [1,N]$ such that (a_i,b_j) appears in a previous alignment. $O(MN)$ time is spent searching the lists when computing one alignment, so the total time to compute k best non-intersecting alignments is $O(kMN)$. In theory, the major storage requirement is for $O[k \times \min(M,N)]$ space to hold aligned pairs. However, in practice the space used may be $O[\min(M,N)]$ because most local alignments are much shorter than $\min(M,N)$.

Finding repeated regions within a sequence

When the two sequences being aligned are identical, the matrix H becomes a symmetric matrix with a trivial alignment on the main diagonal. To avoid this trivial alignment, we compute just the half of H above the main diagonal. The recurrences are adjusted as follows:

$$H(i,j) = \begin{cases} \max\{0, E(i,j), F(i,j), H(i-1,j-1)+v(a_i,b_j)\} & \text{if } i>0 \text{ and } i<j \\ 0 & \text{if } i=0 \text{ or } i=j \end{cases}$$

$$E(i,j) = \begin{cases} \max\{E(i-1,j), H(i-1,j) - q\} - r & \text{if } i>0 \text{ and } i<j \\ -q & \text{if } i=0 \text{ and } j>0 \end{cases}$$

$$F(i,j) = \begin{cases} \max\{F(i,j-1), H(i,j-1) - q\} - r & \text{if } i>0 \text{ and } i<j \\ -q & \text{if } i>0 \text{ and } i=j \end{cases}$$

Note that there is no need to compute $E(i,i)$ and $F(0,j)$.

The algorithm for finding the start-point and end-point of a repeat is given in Figure 2. The algorithms in Figures 1 and 2 differ only in the index limits of the inner **for** loops and in the initial value for p in the forward pass. Notice that in Figure 2(B) the first **for** loop initializes to -1 the values of H and E on the main diagonal from $(1,1)$ to (I,I).

After a best algorithm is located, the Myers–Miller procedure is again called to construct the actual alignment. Like the algorithm in Figure 2, the Myers–Miller algorithm must avoid the main diagonal. The lists introduced for storing previously aligned pairs are also useful here. Before the Myers–Miller algorithm is invoked, each $i \in [1,M]$ is entered into the list for row i. The algorithm in Figure 2 can also be easily generalizbd to find k best non-intersecting repeats, as in the two-sequence case.

Implementation

We have implemented the algorithms in Figures 1 and 2 as a portable C program called SIM. SIM finds similar segments or repeats according to user-supplied scoring parameters $v(a,b)$, q and r. For DNA sequences, users can simply provide numbers m, q and r, where m is a negative number used for scoring each mismatch [i.e. aligned pair (a,b) with $a \neq b$] and non-negative numbers q and r specify a gap penalty $qx + r$ to be subtracted for each gap of length x. Aligned pairs (a,b) with $a = b$ automatically receive the bonus 1.0. Incidentally, replacing real arithmetic by integer arithmetic in the Myers–Miller software resulted in the program running around five times faster on our machine (a Sun 4/260).

The SIM program was used to align the 73 360 nucleotide sequence of the human β-like globin gene cluster with the 44 594 nucleotide sequence of the same gene cluster in rabbit. After some experimenting, we settled on a mismatch penalty of $m = -1.5$, gap-open penalty of $q = 6$ and gap-extension penalty of $r = 0.2$, which seemed useful for producing long alignments while minimizing inappropriate matches, e.g. between non-homologous repeats in the two species. At these criteria, the longest aligned segment includes 9875 nucleotides from the human sequence and 10 724 nucleotides from the rabbit sequence, in both cases containing the δ- and β-globin genes along with the intergenic sequences. If repeats in the rabbit sequence are counted only once, and repeats not homologous between species are ignored (Margot *et al.*, 1989), only 31 800 of the 44 594 nucleotides are available for matching with the human sequence. Six alignments accounted for 21 464 of these nucleotides (67%).

Most of the sequence alignments are in regions known to be similar by dot-plot analysis (Margot *et al.*, 1989), but the alignments allow the sequences to be analyzed at high resolution (i.e. individual nucleotides). As expected, long regions of sequences within and flanking the orthologous genes are aligned. The previous dot-plots could only reveal matching sequences at a resolution of ~50–100 nucleotides, and one could not determine instantly which sequences were conserved. In the new alignments, one can see the most conserved regions easily, and very short, well-conserved sequences (even only 10–20 nucleotides long, the size of recognition sites for transcription factors) are readily apparent. The protein-coding sequences are well conserved, as expected, but these alignments also show extensive matches in the flanking and intron regions. These conserved sequences are candidates for regulatory regions, and they are the targets for on-going site-directed mutagenesis studies to ascertain possible functions.

An example of the sequence alignment in a known regulatory region is that of the γ-globin gene promoter, shown in Figure 3. As expected, the binding sites for known transcription factors, such as TFIID, the CAAT-binding protein CP1, CACCC-binding protein, the octanucleotide binding factor OTF1 and NFE1 (reviewed by Johnson and McKnight, 1989), are well conserved in both species. The CCAAT-boxes are duplicated in the γ-globin gene promoter, at positions 1214 and 1241 in the alignment, but the sequence matches around the distal CCAAT-box are more extensive than are those around the proximal CCAAT-box. It is notable that the upstream CCAAT-box is preceded by a binding site for the factor NFE3 (Mantovani *et al.*, 1989), which may account for the more extensive conservation. In this same region, the CCAAT

```
                               NFE1
  350     .    :    .    :     ++++++:   .    :    .    :    .    :    .    :    .    :    .    :
38508 GAAGTGAACCTAGCATTTATACAAGATAATTAATTCTAATCCACAGTACCTGCCAAAGAACATTCTACCATCATCTTTACTGAGCATAGAAGAGCTACGC human
        || |    ||| ||||||||    |||||||| | ||   | |||||--------------------------------|||||| | |||||||| || |
14554 AAATTTTCCCTGGCATTTATCAGAGATAATTTACTCAGAGCCACA
                              ++++++                                CTGAGTACACAGAAGAGCAACAC rabbbit

                                         NFE1  Enh core
  450     .    :    .    :    .    :     +++++--+++++++-   :    .    :    .    :    .    :    .    :
38608 CAAAACCCTGGGTCATCAGCCAGCACACACACTTATCCAGTGGTAAATACACATCATCTGGTGTATACATA      CATACCTGAATATGGAATCAAAT...~300 nts...
       ||| || |||||||| ||| ||||| |||||||||||||||||| ||  |||||| || |||||  |||| |------||| | ||    ||| ||--| ||
14620 CAATACTCTGGGTCACCAGTCAGCATACACACTTATCCAGTGGGAAGCACACATTATGTGGTGCCTACAGAGCAAGACATTCATGTGGATGAAA  ATAT...~300 nts...
                                         +++++--++++-+++

                                                                                   NFE1
  850     .    :    .    :    .    :    .    :    .    :    .    :    .    :    .+++-++   .    :    .    :    .    :
38966 AATCCTGGACCTATGCCTAAAACACATTTCACAATCCCTGAACTTTTCAAAAATTGGTACATGCTTTAGCTTTAAACTACAGGCCTCACTGGAGCTACAG human
      |||| |||| || |||| |||| ---| ||  |||------||||||| |  || |  ||||||||||| |||  | ||||| ||  ||| | || |
15012 AATCTTGGATCTGTGCCCAAAAT   TATCTGAAT       ACTTTTCATATGTTTGCTTATGCTTTAGCTATAATTTGCAGGCATCTATGGTGTTAGTG rabbit
                                                                  +++-++

  950     .    :    .    :    .    :    .    :    .    :    .    :    .    :  Purine-rich .    :    .    :
39066 ACAAGAAGGTAAAAAACGGCTGACAAAAGAAGTCCTGGTATCCTCTATGATGGG   AGAAGGAAACTAGCTAAAGGGAAGAATAAATTAGAGAA       human
      || | ||  ||  ||| |-||| ||||| ||||||| |  |  | ||| || |---| ||| ||| |||||||| ||| ||||||| |||||-----
15103 ACTAAAATATAGGAAATG CTGGCAAAAAAAGTCCTAGACTATTTAATGGTGAGGAAACAAGAAAAAATAGCTAAATAGAAAAAATAAATGGGAGAACTCTT rabbit

                                                                   -202 -198 -196  NFE1    OTF1
 1050     .    :    .    :    .    :    .    :    .    :    .    :    .    :  .|   | |  .  ++++++ .+++++
39158    AAACTGGAATGACTGAATCGGAACAAGGCAAAGGCTATAAAAAAAATTAAGCAGCAGTATCCTCTTGGGGGCCCCTTCCCCACACTATCTCAATGCA human
      ---|||  ||||---||||  || ||| | |--------||| |||| || || |||||  || |||| |||  |||  |   |||||||||||--|
15202 TAAAAAAATGAAT   TGAAAAGGGACAGGAC        TAACAAAACTTGAGGAGCAGAGCCCCCTTGAGGGGGCCTCTCTGCCACTATCTCAAT  A rabbit
                                                                                   ++++++
   -175
     |
    NFE1                                                         -117
    -++ +++                                              NFE3     CDP           CDP            CDP
                                       CAC-BP           ++++++++++++CP1         ++++           +++++
 1150 +++ .    :    .    :    .    :    +++++:    .    :    .    :  ++++++       NFE1             CP1
39255 AATA TCTGTCTGAAACGGTCCCTGGCTAAACTCCACCCATGGGTTGGCCAGCCTTGCCTTGACCAATAGCCTTGACAAGGCAAACTTGACCAATAGTCT human
                                                           : | +++++ :     +++-++:    .   :+++++      :
       |||-||||| |||| | -|| ||| ||||||  ||||| || | || ||||||||||||||||||||| | | |||    ||   |||||||||| ||
15289 TATAGTCTGTGTGAAGCT TCTCTGCCTAAACCTCACCCCTGCGCTGACCAGCCTTGCCTTGACCAATAGTCGTTACACAAAAACACTGACCAATAGCCT rabbit
       -++-+++                         +++++                       +++++                       +++++
                                                        ++++++++++++                       +++++
                                                                 +++++

                                                        TFIID
 1250     .    :    .    :    .    :    .    :    .    ++++++ .    :    .    :    .   cap  :    .    :
                                                                                      |:
39354 TAGAGTATCCAGTGAGGCCAGGGGCCGGCGGCTGGCTAGGGATGAAGAATAAAAGGAAGCACCCTTCAGCAGTTCCACACACTCGCTTCTGGAACGTCTG human
       |||| |  | | ||   | ||||||| |--- || | || || |||||||||||||| |  || |  ||||||| |||| ||| |||||||  || ||||
15388 CAGAGAACACGGCGAAACAAGGGGCCAG    ATGTCCAGCGAGGAAGAATAAAAGGACGAGCCTTAGAGCAGTTTCACATACTTGCTTCTGAGACATCTG rabbit
                                                        ++++++                      |
```

Fig. 3. Alignment of the promoter region for the γ-globin genes in humans and rabbits. A small subset of a SIM alignment (match = 1, mismatch = −1.5, gap-open penalty = 6.0, gap-extension penalty = 0.2) that begins at position 38196 in the human sequence (5′ to the A γ-globin gene) and at position 14241 in the rabbit sequence (5′ to the γ-globin gene) is shown. Vertical lines indicate matching nucleotides, and dashes are gaps introduced to optimize the alignment. The cap site, which encodes the nucleotide at the 5′ end of the mRNA, is indicated at position 1329. A match to the consensus sequence for a known binding protein is indicated by a +, and a mismatch is indicated by a −. The sites of HPFH mutations are indicated by the conventional negative numbers, which are the distances in nucleotides from the cap site in the human sequence. The conserved sequences mentioned in the text that have not yet been shown to be important binding sites are underlined. The sequences between 550 and 850 in the alignment are not shown to save space, although many matches are in this region. The following consensus sequences were used to identify the binding sites for specific factors: WATAAA for TFIID, CCAAT for CP1, TGACC for CDP, WGATAR for NFE1, GCCTTG for NFE3, CACCC for CAC-BP, ATGGCAAAT for OTF1, GGGCGG for Sp1, and TGTGGWWWG for the enhancer core; W = A or T, R = G or A. The first letter of the protein name is over the nucleotide beginning the binding site.

displacement protein, CDP, competes with CP1 for binding to the CCAAT sequences (Superti-Furga *et al.*, 1988). However, the middle binding site for CDP and the binding site for NFE1 located between the two CCAAT sequences (position 1224 in Figure 3) are not conserved in the rabbit sequence, indicating either that these latter binding sites are not required for promoter function or are used for a species-specific function. The G at position −117 (two nucleotides 5′ to the distal CCAAT sequence) is mutated to an A in many individuals with the Greek-type hereditary persistence of fetal hemoglobin (HPFH), in which the γ-globin gene continues to be expressed in adult life. This G to A transition affects the binding of all three factors in this region (NFE3, CP1 and CP) and may account for the HPFH phenotype (Superti-Furga *et al.*, 1988; Mantovani *et al.*, 1989).

Two molecules of NFE1 and one of OTF1 bind to the human sequence between 1136 and 1158 in the alignment (Tsai *et al.*, 1989). The upstream NFE1 site is very well conserved between rabbit and humans (Figure 3), indicating that it is involved in promoter function. However, the OTF1 site in humans is not conserved in rabbits, indicating that it may not be as important. Although the downstream NFE1 site has a one-nucleotide insertion in the rabbit sequence, the human sequence is the site of another HPFH mutation (−175 T to C), thus it has been implicated in promoter function, at least in humans. Functional analysis shows that the −175 T to C transition decreases OTF1 binding but increases NFE1 binding (Mantovani *et al.*, 1988), and the increase in γ-globin gene expression with the -175 mutation requires NFE1 but not OTF1 (Nicolis *et al.*, 1989). This lack of involvement of OTF1 fits with the absence of conservation of this sequence between rabbit and human. Interestingly, several other HPFH mutations (reviewed in Stamatoyannopoulos and Nienhuis, 1987) are in a region that is not conserved in the rabbit sequence (-196 and -198 in Figure 3).

Many sequences whose functions have not been adequately tested are also well conserved between the two species. Some notable examples are the AGGGGCC at position 1270 in the alignment (between the CCAAT and AATAAA sequences), a purine-rich sequence between 1010 and 1038, and the sequence between 911 and 921 that contains a good match to an NFE1 site (all underlined in Figure 3). Several hundred nucleotides further upstream, the sequences between 477 and 496 have matches to NFE1 sites and enhancer core sequences, and the sequence 374–381 has an exact match to the consensus NFE1 site in both species. Although these upstream sequences are not required for tissue- and stage-specific expression in transgenic mice (Perez-Stable and Constantini, 1990), their potential function in efficiency of expression has not been tested, and the patterns of sequence conservation shown in Figure 3 make them candidates for some functional role. Another clue that the AGGGGCC sequence at 1270 may be important is that the very similar sequence GGGGGCC is the site of the HPFH mutation at -202. The GGGGGCC sequence is also present in rabbit, but it is offset by 2 nucleotides in this particular alignment (position 1122 in Figure 3). The C to G transversion at -202 has been proposed to increase the binding of an Sp1-like protein to this sequence (Collins *et al.*, 1984) and thus increase the expression of the γ-globin gene. A similar protein may bind at position 1270.

The intergenic regions contain a large number of repeated DNA sequences in both rabbit and human. The repeated DNAs are probably transposable elements that have dispersed throughout the genome. In all cases, the repeats are in different locations in the rabbit and human β-like globin gene clusters (Margot *et al.*, 1989). This indicates that the repeats have inserted separately into these gene clusters, and presumably throughout the genomes, of rabbits and humans. By examining the alignments in the intergenic regions, we see several examples of clean insertions of a repeat in the rabbit sequence

```
 2100          .         :         .         :         .         :         .         :         .         :
56779 TATGTTGTCTTTCTTCCTCCTTCCTTTGCCTGCACATTGTAGCCCATAATACTA TACCCCATCAAGTGTTCCTGCTCCAAGAAATAGCTTCCTCCTCTT Human
        ||||  ||||||| ||| ||||  |   |  ||| |||| | | || |||||-|  || | || || ||||||||| ||||---| |||||| |-|||
23999 GGTGTTTCCTTTCTTTCTCTTTCCAATATTTATACACTGTAACTCGTATTACTAATTGCCAAACATGTATTCCTGCTCTAAGA   ACCTTCCTGC CTT Rabbit

 2200          .         :         .         :         .         :         .         :         .         :
56878 ACTTGCCCCAGAACATCTCTGTAAAGAATTTCCTCTT ATC   TTCCCATATTTCAGTCAAGATTCATTGCTCACGTATTACTTGTGACCTCTCTTGAC Human
       ||   |||| |||--||||  | |  | | | ||||-|||---| | ||  || ||||  ||-|| | | ||| |  || |||  |||  ||| | || 
24095 CCTGCTCCCAAAAC  CTCTTCAGAATAATGCTTCTTTATCCAGTCCTCAATTTCCAGTACAG TTTAGTACTCTCTCATCACTGCTGAGGTCTGTGGAG Rabbit

 2300          .         :         .         :         .         :         .         :         .         :
56974 CCCAGCC                                                                                              Human
      | |||||---------------------------------------------------------------------------------------------
24192 CTCAGCCTTGGGGAATTTTATTTTTAATTTTTAAGATTTATTTATTTAATTTAAAATTGGTATTATAGAGAAAGGGAGAGAGAGAGAGGGAGAGAGAGA Rabbit

 2400          .         :         .         :         .         :         .         :         .         :
56981                                                                                                      Human
      ----------------------------------------------------------------------------------------------------
24292 GAGAGAGAGAGATCTTCCATTCACTGGTTTACTCTACAAATGGCCATATTGGCTAGGCTGAAGCTAGGAGCTTTATCCAGATCCCCTATGTGGGTGGCAA Rabbit

 2500          .         :         .         :         .         :         .         :         .         :
56981                                                                                                      Human
      ----------------------------------------------------------------------------------------------------
24392 GGACCCACACATTAGGCCATCTTCTGCTGCTTTTCCTGTGCCATTAGCAGGGAGTTGGGTCAGAAGCGGAGCAGCTGGGACACAAACTAGTGCTCATATG Rabbit

 2600          .         :         .         :         .         :         .         :         .         :
56981                                         ACAATA               AACTTCTCTATACTACCCAAAAAAT CTTTCCAAACCCT Human
      ----------------------------------------||||||---------------||||| |  ||  | |  |  ||||-|||||||||||||
24492 AGATGTCAGCATCAGAGGTGGCAGCTTTACCTCGTACGCCACAATACTGGCTCCCTCAGTGAACTTTTTAATGTTGCTGAGGAAATTCTTTCCAAACCCT Rabbit

 2700          .         :         .         :         .         :         .         :         .         :
57025 CCCCGACACCATATTTTTATATTT                TTCTTATTTATTTCATGCACACACACACACTCCG                           Human
       || |------||| || || |||----------------||||||| |    ||  ||||||||||||| |  --------------------------
24592 TCCTG      TATATTCATTTTTCACAACTTTACAGCTCTTCTTATGTCACACACACACACACACACACACGAGCAAATACATACATACACCCCACA Rabbit

 2800          .         :         .         :         .         :         .         :         .         :
57083        TGCTTTATAAGCAATTCTGCCTATTCTCTACCTTCTTACAATGCCTACTGTGCCTCATATTAAATTCATCAATGGGCAGAAAGAAAATATTTA Human
      -------||||| ||||  | ||||| |||||||  ||-|  ||   | ||||||||||||   ||| |||||| || || |  |  |||   | |||| 
24686 TATATTATGCTTGATAAATATTTCTGTCTATTCTGCAC TAATTTGCAGGCCTACTGTGCCCTGTATCAAATTCTTCTATAGATAATAAGTGGAAATTTT Rabbit
```

Fig. 4. Insertion of a C repeat into rabbit DNA. The nucleotide sequence of the DNA containing the δ- and β-globin genes was aligned between human and rabbit, and the portion including and immediately surrounding the C13 repeat is shown. The short direct repeats flanking C13 in the rabbit DNA are underlined. Virtually all of the rabbit C repeat sequence does not align with the human sequence, whereas the regions flanking it are quite similar (match = 1, mismatch = -1.5, gap-open penalty = 6.0, gap-extension = 0.2). The positions in the human and rabbit sequences are indicated at the beginning of the corresponding lines. C13 begins 507 nucleotides 3′ to the poly(A) addition site of the rabbit δ-globin gene.

that is absent from the human gene cluster. Figure 4 shows the alignment surrounding a short interspersed repeat in the rabbit gene cluster, C13, with the human sequence. Although most of the rabbit C13 repeat does not match with the human sequence, as expected for an insertion, short regions close to the ends do match. This is observed for several insertions, and may reflect some requirement for very short regions of homology at the insertion site. Not only can we see the insertion sites at nucleotide resolution, but we also can observe proposed gene conversion events, such as those between the δ- and β-globin genes, and determine their end-points.

Detailed analysis of the alignments revealed some new and unexpected aspects of the β-like globin gene clusters. While analyzing the sequences of the whole gene clusters, a previously undetected L1Oc repeat was found in the rabbit gene cluster between the ϵ- and γ-globin genes (L1Oc10 in Figure 5). The alignments were then repeated, focusing on the region between these two genes. These secondary runs were considerably faster than those using the whole gene clusters, taking only 24 min to compute 20 alignments of the 13 500 $\times$ 8000 nucleotide sequences. Several new insights were obtained into the evolution of the intergenic region between the ϵ- and γ-globin genes. L1Oc10 is a shortened example of a class of repeats that can be up to 8000 nucleotides long. This particular L1Oc is only 150 nucleotides long, and is shortened at both its 5′ end (as usual) and its 3′ end. This is the only known example in rabbits of an L1Oc repeat that does not extend through the 3′ untranslated region. It is located in a cluster of inserts of short C repeats (Figure 5). L1Oc10 was observed because of the sequence similarity to an L1 repeat in the human gene cluster (L1Hs3) and its similarity to other rabbit L1Oc repeats. In both cases the sequence similarity was seen by the alignments given by this program; the dot-plot analysis was not at a sufficiently high resolution to observe it in this complex region.

Additional sequence matches in the $\epsilon-\gamma$ intergenic regions allow us better to localize the positions for insertion of the

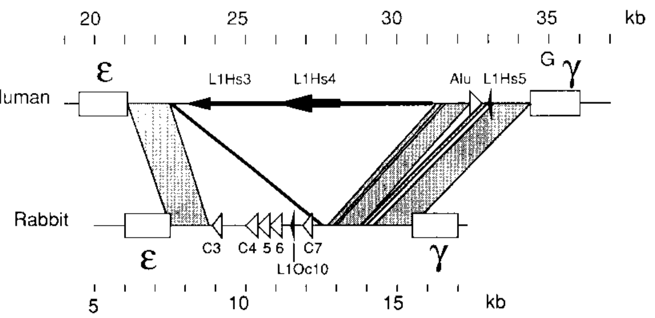

Fig. 5. Map of similarities between rabbit and human sequences in the $\epsilon-\gamma$ intergenic region. The sequences between the ϵ- and γ-globin genes were searched for similar sequences, and the local alignments found between the rabbit and human sequences are mapped diagrammatically as shaded parallelograms. Genes are shown as open boxes, short repeats are open triangles, and L1 repeats are filled arrows (some are severely truncated). The genes are transcribed 5′ to 3′ left to right. The symbols for the repeats are pointing toward the A-rich tracts at their 3′ ends.

L1Hs3 and 4 repeats in humans and the set of C repeats in rabbits. In both cases, the multiple repeats are thought to represent recursive insertions into the same site (Rogan *et al.*, 1987; D.E.Krane and R.C.Hardison, unpublished). As diagrammed in Figure 5, C3–C7, along with L1Oc10, are present in a segment of rabbit DNA that has no homolog in human. Hence, we conclude that the multiple rabbit repeats inserted at the homolog of human positions 22 450–22 460. L1Hs3 and 4 interrupt a series of matching segments with the rabbit DNA, again indicating an insertion within the homologs to nucleotides 12 630–12 660 in the rabbit sequence. The L1Hs3 sequence is unusual in that it does not extend into the 3′ untranslated region of this class of repeats. However, its 3′ end is interrupted by a 55 nucleotide sequence that is also found in its 5′ flank. This was originally assigned as the flanking direct repeat indicating a duplication of the insertion site (Rogan *et al.*, 1987). However, we have previously argued that this is not the flanking direct repeat, because the copy of it closest to the γ-globin gene is embedded in a long sequence of homology to rabbit (Margot *et al.*, 1989). Thus this must represent some kind of recombinational event beyond that associated with insertion of a repetitive element.

All these data from the intergenic regions between the ϵ- and γ-globin genes show that this region has sustained many insertions and recombinations separately in each species. Clearly, this is a hot-spot for recombinations in the gene cluster, and the complexity of this pattern can only be seen with alignments such as those provided by this program.

Comparison with other methods

There exist several useful local similarity programs. SIM, the program presented here, is the Rolls Royce of them all; it costs the most to run but has definite advantages in certain situations.

Probably the fastest existing local similarity program is a variant of BLAST (Altschul *et al.*, 1990). BLAST aligns regions without allowing the introduction of gaps. Though primarily designed for database searches, it can be applied to comparing two long sequences. In just 3 s on our Sun4, it discerns the large-scale features of local similarities in the two β-globin clusters, revealing such features as corresponding exons of orthologous and paralogous genes and long matching regions within L1 repeats. Altschul *et al.* discuss this application of BLAST in more detail.

Gotoh (1987) presents a program with the same goals as SIM. It operates in roughly the same time as our current version of SIM, though it requires use of secondary storage. Its main drawback is that it does not detect all high-scoring alignments. For example, comparing the two $\delta-\beta$-regions ($m = -1.5$, $q = 6$, $r = 0.2$), it finds the three highest scoring alignments but misses the next two (i.e. the third exon of δ versus the third exon of β, and vice versa).

Probably the most useful current alternative to SIM is LFASTA (Pearson and Lipman, 1988; Pearson, 1990).

LFASTA uses a heuristic method to determine likely locations for significant local alignments and applies a dynamic programming algorithm to these regions. When LFASTA was applied to the β-globin clusters with its default parameters, it produced 127 alignments in 45 min on our Sun4. By comparison, our current version of SIM required ~15 h to produce 100 alignments. The SIM alignments were generally longer than the LFASTA alignments; the longest SIM alignment covered 9875 nucleotides of the human sequence, while the longest LFASTA alignment covered 2989 nucleotides. In a number of cases, several LFASTA alignments overlapped to cover the same region as a single SIM alignment. For example, LFASTA generated ten alignments of portions of the two ϵ genes. While the higher-scoring of these ten were typically comparable to a portion of the single 3490 nucleotide ϵ alignment from SIM, lower-scoring LFASTA alignments were sometimes clearly inappropriate. In particular, only one LFASTA alignment covered all three exons of ϵ, and it did not correctly align splice junctions and exons.

The SIM alignment covering the γ-globin gene promoter, discussed in detail above, was part of an alignment that encompassed 2981 nucleotides of the human gene cluster, extending 1236 nucleotides upstream of the cap site to 260 nucleotides downstream of the coding region. LFASTA covered this region with three alignments. The highest-score LFASTA alignment of the region began in precisely the same position as the SIM alignment and ended in the second intron. The second alignment picked up at 280 nucleotides downstream of the first alignment and ended 300 nucleotides downstream of the last pair in the SIM alignment. These differences between SIM and LFASTA probably arise primarily because, with the default settings, SIM penalizes mismatches more and long gaps less than does LFASTA. Unfortunately, it is difficult to adjust the two program's parameters to make them equivalent since LFASTA places hidden restrictions on the alignments that it allows, such as limiting gaps to 30 nucleotides, which cannot be changed by simply modifying the scoring parameters. (Indeed, the gaps permitted in LFASTA alignments can be 30 nucleotides long only under unusual circumstances that are difficult to describe.)

The third highest scoring LFASTA alignment of the γ-globin gene encompassed the regions aligned by the highest-scoring alignment, extending 180 nucleotides further upstream and 110 nucleotides further downstream. The two alignments typically paired a given position in the human sequence with different positions in rabbit. However, the two LFASTA alignments were identical to each other and to the SIM alignment for a stretch of ~450 nucleotides from position 892 in Figure 3 to a position lying between the cap site and the start of the coding sequence. Downstream of that position, the third LFASTA alignment seemed completely inappropriate. The first LFASTA alignment matched the NFE1 sites at positions 374–381 in Figure 3, but missed the interesting matches at positions 477–496. Conversely, LFASTA's third alignment detected the latter region but missed the former. Because LFASTA limits gaps to 30 nucleotides, there is no way to set its parameters to simultaneously align both regions (note the 34 nucleotide gap in rabbit at 395–429).

In summary, letting SIM run overnight on a moderately priced piece of equipment saves the user from the need to spend time splicing together overlapping alignments of the same region and resolving alignment conflicts. Moreover, arbitrarily long gaps are permitted in alignments. Finally, all factors controlling potential alignments and their scores are explicit and can be modified as appropriate.

Discussion

Previous dynamic-programming algorithms for local similarity require space proportional to the product of the sequence lengths. This paper presents an algorithm that uses only space proportional to the longer sequence's length. Recent algorithmic improvements (X.Huang and W.Miller, *Advances in Applied Mathematics* in press) have brought the time requirements for this approach in line with those for space-intensive dynamic-programming algorithms. This allows our software to align long sequences on a small computer. In the following, we mention a couple of points about use of the software.

A little care is needed when choosing scoring parameters. Experience shows that if excessively low penalties are used, for example, $m = -0.5$, $q = 1.0$ and $r = 0.3$, then SIM generates many long alignments of weak similarity. If mismatch weights $v(a,b)$ are positive on average, then the program always produces global alignments, so it is important that non-conservative substitutions be given negative scores.

It is difficult to determine *a priori* how large k should be. Our current implementation of SIM computes later alignments very quickly, so k can be set very large without significantly decreasing the efficiency of the program. For example, the program takes 10 h to find the best alignment of the β-globin clusters, and an additional 5 h to find the next 99 best. In practice, we simply pick a comfortably large k and discard alignments once we start getting, for example, paired $(AT)^n$ regions. One referee suggested that the program should stop when alignment scores reach the level that would be expected in comparing random sequences, but we do not know how to guarantee this. Indeed, the issue of statistical significance is an important topic not addressed here; it would be useful to extend SIM to produce an estimation of the statistical significance of each alignment, as done by Pearson and Lipman (1988).

Acknowledgements

The authors thank William Pearson for providing the LFASTA program and helping us understand it. Both referees made suggestions that improved the paper. X.H. and W.M. were supported by grant LM05110 from the National Library of Medicine; R.C.H. is supported by PHS grant DK27635 and an RCDA DK01589.

References

Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D. (1990) A basic local alignment search tool. *J. Mol. Biol.*, **215**.

Collins,F.S., Stoeckert,C.J., Serjeant,G.R., Forget,B.G. and Weissman,S.M. (1984) G γ β + hereditary persistence of getal hemoglobin: cosmid cloning and identification of a specific mutation 5′ to the G γ. *Proc. Natl. Acad. Sci. USA*, **81**, 4894–4898.

Goad,W.B. and Kanehisa,M.I. (1982) Pattern recognition in nucleic acid sequences I: a general method for finding local homologies and symmetries. *Nucleic Acids Res.*, **10**, 247–263.

Gotoh,O. (1987) Pattern matching of biological sequences with limited storage. *Comput. Applic. Biosci.*, **3**, 17–20.

Hall,J.D. and Myers,E.W. (1988) A software tool for finding locally optimal alignments in protein and nucleic acid sequence. *Comput. Applic. Biosci.*, **4**, 35–40.

Johnson,P.F. and McKnight,S.L. (1989) Eukaryotic transcriptional regulatory proteins. *Annu. Rev. Biochem.*, **58**, 799–839.

Karlin,S., Morris,M., Ghandour,G. and Leung,M.Y. (1988) Efficient algorithms for molecular sequence analysis. *Proc. Natl. Acad. Sci. USA*, **85**, 841–845.

Mantovani,R., Malgaretti,N., Nicolis,S., Ronchi,A., Giglioni,B. and Ottolenghi,S. (1988) The effects of HPFH mutations in the human γ-globin promoter on binding of ubiquitous and erythroid specific nuclear factors. *Nucleic Acids Res.*, **16**, 7783–7797.

Mantovani,R., Superti-Furga,G., Gilman,J. and Ottolenghi,S. (1989) The deletion of the distal CCAAT box region of the A γ-globin gene in black HPFH abolishes the binding of the erythroid specific protein NFE3 and of the CCAAT displacement protein. *Nucleic Acids Res.*, **17**, 6681–6691.

Margot,J.B., Demers,G.W. and Hardison,R.C. (1989) Complete nucleotide sequence of the rabbit β-like globin gene cluster. *J. Mol. Biol.*, **205**, 15–40.

Myers,E.W. and Miller,W. (1988) Optimal alignments in linear space. *Comput. Applic. Biosci.*, **4**, 11–17.

Nicolis,S., Ronchi,A., Malgaretti,N., Mantovani,R., Giglioni,B. and Ottolenghi,S. (1989) Increased erythroid specific expression of a mutated HPFH γ-globin gene promoter requires the erythroid factor NFE1. *Nucleic Acids Res.*, **17**, 5509–5516.

Pearson,W.R. (1990) Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol.*, **183**, 63–98.

Pearson,W.R. and Lipman,D. (1988) Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, **85**, 2444–2448.

Perez-Stable,C. and Costantini,F. (1990) Roles of fetal G γ-globin promoter elements and the adult β-globin 3′ enhancer in the stage-specific expression of globin genes. *Mol. Cell. Biol.*, **10**, 1116–1125.

Rogan,P.K., Pan,J. and Weissman,S.M. (1987) L1 repeat elements in the human ϵ-$^{G}\gamma$-globin gene intergenic regions. *Mol. Biol. Evol.*, **4**, 327–342.

Sellers,P.H. (1984) Pattern recognition in genetic sequences by mismatch density. *Bull. Math. Biol.*, **46**, 501–514.

Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular sequences. *J. Mol. Biol.*, **147**, 195–197.

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

Stamatoyannopoulos,G. and Nienhuis,A. (1987) Hemoglobin switching. In Stamatoyannopoulos,G., Nienhuis,A., Leder,P. and Majerus,P. (eds), *The Molecular Basis of Blood Diseases.* W.B.Saunders, Philadelphia, pp. 66–105.

Superti-Furta,G., Barberis,A., Schaffner,G. and Busslinger,M. (1988) The −117 mutation in Greek HPFH affects the binding of three nuclear factors to the CCAAT region of the γ-globin gene. *EMBO J.*, **7**, 3099–3107.

Tsai,S.-F., Martin,D.I.K., Zon,L.I., D'Andrea,A.D., Wong,G.G. and Orkin,S.H. (1989) Cloning of cDNA for the major DNA-binding protein of the erythroid lineage through expression in mammalian cells. *Nature*, **339**, 446–451.

Waterman,M.S. and Eggert,M. (1987) A new algorithm for best subsequences alignment with application to the tRNA–rRNA comparisons. *J. Mol. Biol.*, **197**, 723–728.

Waterman,M.S. (1988) Sequence alignments. In Waterman,M.S. (ed.), *Mathematical Methods for DNA Sequences.* CRC Press, Boca Raton, FL.

Received on January 29, 1990; accepted on May 30, 1990

Circle No. 10 on Reader Enquiry Card