Volume 16 Number 5 1988 Nucleic Acids Research

A flexible multiple sequence alignment program

Hugo M.Martinez

PO Box 0448, Department of Biochemistry and Biophysics, University of California, San Francisco, CA 94143, USA

Received August 17, 1987; Revised and Accepted December 5, 1987

ABSTRACT

The 'regions' method for multisequence alignment used in the previously reported program MALIGN [1] has been generalized to include recursive refinement so that unaligned portions between two regions at the current level of resolution can be handled with increased resolution. Additionally, there is incorporated a limiting of the number of regions to be used at any level of resolution from which to abstract an alignment. This provides a significant increase in speed over the unlimited version. The program GENALIGN uses this improved regions method to execute fast pairwise alignments in the framework of Taylor's multisequence alignment procedure using clustered pairwise alignments. Pairwise alignments by dynamic programming are also provided in the program.

INTRODUCTION

The program GENALIGN is a multisequence alignment program which has evolved from the previously reported program MALIGN as an attempt to deal with the problem of fragmentary commonality and still maintain reasonable speed of computation. Fragmentary commonality refers to the cases in which less than all the sequences have a segment in common and this segment is to be regarded as significant.

Speed of computation in MALIGN is achieved by the 'regions' method of alignment. A region is a segment (subsequence of contiguous elements) which is common to all the sequences being aligned and the idea is to first find all the regions having a length above a specified minimum and then abstract from these a sequence of non-overlapping ones which maximizes a function that rewards matches and penalizes insertions/deletions.

There are two important limitations to the regions method. One is the failure to account for unaligned portions between successive regions, and the other is to allow for reduced commonality, that is, when all the sequences may not have a segment in common but some of them do.

The first of these limitations can be partially compensated by reducing the minimum region length used, but this soon leads to a very large set of regions from which to extract an optimal sequence and hence in greatly reduced speed. Instead, I

have adopted what may be referred to as 'telescoping' in that an alignment is first done at a specified minimum region length and the minimum length then recursively reduced to accommodate inter-region portions. Thus a previous alignment acts as the guide to a refined alignment.

An approach to the second limitation is to use regions of reduced commonality. In addition to minimum length, the region must then involve a minimum number of sequences. But when combined with telescoping, this proved to involve a number of *ad hoc* decisions regarding how the telescoping was to be done between successive regions which use different subsets of the sequences. Alternatively, I have adopted the multisequence alignment approach advocated by Taylor [2] in which all pairs of sequences are first aligned and a full alignment is then displayed in accordance with a clustering based on the pairwise alignment scores.

The clustered ordering of the sequences affects the overall alignment. This is a deficiency of any multi-sequence alignment procedure which uses pair-wise alignment as the basic comparison method. But I feel that the clustering technique which helps solve the reduced commonality problem far outweighs the disadvantages accruing from ad hoc definitions of regions of reduced commonality.

The basic algorithm used by GENALIGN is thus one of clustered, pairwise alignment in which the pairwise alignment is done by a telescoped regions method. The option is provided to do the pairwise alignments by the Needleman-Wunsch [3] dynamic programming algorithm which is roughly ten times slower than the telescoped regions method but provides, as detailed below, unbiased full resolution and a different emphasis on what is regarded as significant. The factor of ten in speed is based on a wide variety of examples. The theoretical improvement in speed is difficult to estimate because of the dependence on similarity. The greater the similarity, the faster the regions method because the dynamic programming approach does not capitalize on segments of similarity to limit searches.

THE TELESCOPING REGIONS METHOD ALGORITHM

Automatic specification of the initial minimum region length as well as the subsequent minimum region lengths for unaligned portions between successive regions is done with the 'expected longest repeat' formula of Karlin, et al., [4]. This formula gives the expected length of the longest repeat (common segment) to be found in the sequences. There is also provided a standard deviation estimate relative to this expected longest length, and hence a means of estimating the significance of a region. If E is the expected longest length and SD the corresponding standard deviation for the given sequences (or subsequences for inter-region alignment), then the minimum region length is taken to be E + (length_factor)*SD. The value of the parameter

length_factor is user specified and has a default value of 0. For fixed alphabet length, the expected longest length decreases as the lengths of the concerned sequences decrease. Thus, the minimum region length that will be used for aligning an unaligned portion between two successive regions corresponding to a previous alignment of minimum region length M, will generally be less than M. The smallest that minimum region length is allowed to get is specified by the parameters 'amino_res_length' and 'nucleic_res_length' for proteins and nucleic acids, respectively. The default values are 2 and 4.

There are two principal, time consuming operations in any regions-based method of alignment: generating the set of regions, and finding the optimal sequence of regions from this set that will constitute the alignment. The first, generating the set of regions, is L*log(L) where L is the sum of the lengths of the sequences, while the second is proportional to the square of the size of the region set. It is therefore advantageous to limit the size of the region set at each level of alignment refinement, including the initial alignment. This size limit, SL, is taken to be 5 times the length of the smallest of the sequences (subsequences). When the number of found regions at the current alignment level reaches SL, the found regions are ordered according to two ranking factors, length and registration. Registration here refers to the position of a region in each of the two sequences being aligned relative to the beginning and ends of these sequences. If SP1, SP2 are the positions relative to the start and EP1, EP2 are the positions relative to the ends, then the condition SP1 = SP2 and EP1 = EP2 would correspond to perfect registration. Otherwise, as a measure of registration I use the sum |SP1-SP2|/SP + |EP1-EP2|/EP in which SP is the mean of SP1, SP2 and EP is the mean of EP1, EP2. Region R1 is better than region R2 if its length is larger than that of R2. If the lengths are equal, the better of the two is the one with the best registration. Subsequently found regions are saved only if they are better than the lowest ranked one of those already saved.

Choosing an optimal set of non-overlapping regions at a given level of refinement is a dynamic programming problem. Having ordered the regions at a given level of refinement according to their starting position in the first sequence, then given any region R, a region path issuing from it is an increasing sequence of pair-wise, nonoverlapping regions having R as its first member. Such a path is assigned a similarity score S defined as follows. Let L be the sum of the lengths of its regions and let D be the sum of the deletions between successive regions of the path. Then S = aL-bD in which a and b are positive, user-defined weighting factors (default value = 1). A region path beginning at R is defined to be optimal if its similarity score is maximal relative to all paths beginning at R. Having found an optimal path for the region R, it is assigned the corresponding similarity value and a pointer to the next region in

this path. Proceeding backwards in the initially ordered set of regions, an optimal path is found for each region based on the paths already found for the regions succeeding it in the ordering.

The justification for limiting the region set based on a ranking is the heuristic that it is these which should guide the alignment at the current refinement level. Of those which are discarded, some may appear at the next level of refinement.

Still another heuristic relative to refining an alignment is that when the lengths of the subsequences corresponding to the portion between two regions are the same, then the alignment is stopped and the matches at corresponding positions are recorded.

The set of regions at any level (refinement and initial) is found by the repeats finding algorithm reported on in Martinez [5], which is based on successive sorting, and is otherwise equivalent to associating a unique sequence with each position and comparing positions based on this association.

THE CLUSTERING ALGORITHM

Once all pairs of sequences from the given set of sequences have been aligned, then the sequences are ordered according to the following simple scheme which is best explained by an illustrative example. Consider the alignment of sequences A, B, C and D for which the pairwise alignment scores of

pwscore(A,B) = 35
pwscore(A,C) = 70
pwscore(A,D) = 100
pwscore(B,C) = 60
pwscore(B,D) = 25
pwscore(C,D) = 125

have been obtained. Because the pair (C,D) has the highest score, the ordering of the sequences would start with list {C,D}. Now A has the combined score of 170 with C and D, while B's combined score with C and D is 85. A is therefore added to the list and at the end to which it is most similar, namely D, to produce the list {C,D,A}. B is now added to the list by putting it at the end to which it is most similar to give the final list {B,C,D,A}. This is the order in which the sequences will be displayed, and the display will be such as to preserve the matching of B with C, the matching of C with D, and the matching of D with A as were found by the pairwise alignments.

OUTPUTTING THE ALIGNMENT

The pairwise alignments are saved in a 'matching matrix' *mm* such that *mm* [s1][i+1][s2] is j+1 if position i of sequence s1 has been matched with position j of

sequence s2 and is otherwise 0 (position i of sequence s1 has not been matched with a position of sequence s2). Given that the sequences are to be output in the order dictated by the clustering, there is constructed a 'number of blanks' array *nb* such that *nb* [s][j] is the number of blanks to be inserted after element j in sequence s. Determining the array *nb* is the key to displaying the multisequence alignment. Thus, the clustering has dictated which pairwise alignments are to be displayed and the *nb* array serves the purpose of displaying these and linking them together in a manner which preserves the individual pairwise matches.

In our example the pairs would be B with C, C with D and with A. Using the matching matrix, the *nb* entries for B and C are first made so that if B and C were displayed as an alignment pair the corresponding matched elements would line up. Given the *nb* entries for C, those for D are determined so that C and D would line up but in such a way that the previously determined *nb* entries for C are at most augmented. That is, we consider the previously determined alignment of C and D. The blanks in the sequence C may not all correspond to its alignment with B. Where there are fewer than those required for the alignment with D, the deficient number are restored; and where there are more, then those of D are increased. The *nb* entries for D and A are finally determined and, again, in such a way that those for D are at most augmented. One now works backward. The *nb* entries for C are adjusted to those of D and the entries for B are adjusted to those of C. One pass forward and one pass backward are all that are required.

AN ACTUAL EXAMPLE

This example is concerned with the alignment of thymidylate synthase from four sources. The pairwise alignments are done in two ways: by the regions method explained above and by the Needleman-Wunsch dynamic programming algorithm. The scoring function for the pairwise alignments is the same for both methods: matches count +1 and spaces (insertion/deletion) count -1. The multisequence alignment scores shown are the sums of column scores. A column score is equal to the number of matches (capitalized letters - 1) minus the number of spaces in the column. The large difference in the two alignment scores (130 for the regions method and 208 for the Needleman-Wunsch method) is deceptive. It is primarily due to the large number of spaces used in the regions method to accommodate the T4 Phage protein which does not bear a strong homology to the other three in the sense of long runs. The regions method of pairwise alignments favors runs of matches since it is these which guide the alignment. On the other hand, the Needleman-Wunsch pairwise alignment will break up runs in order to achieve a higher overall score. The fifth row in each of the alignments constitutes the corresponding consensus sequence.

Clustered Pair-Wise 'Region' Alignment

in 'identity (no translation)' alphabet of:

1. Human (1-313)
2. T4 Phage (1-286)
3. E.coli (1-264)
4. Lactobacillus casei (1-316)

listed in clustered order.

```
  1 mpvagselprrplppaaqerdaeprpphgelQYlgqIqhIlrcGvrkDDRTGTGTlsvFGmqaRysLrdeFP
                                  ||   |  |   |   ||||||||   ||   |  |   ||
  1                             MKQYqdLIkdIfenGyetDDRTGTGTialFGsklRwdLtkGFP
                                ||||  |       |    |||||||   ||   |  |  |||
  1                             MKQYLeLmqKVLDEGtqKnDRTGTGTlSIFGHQMRFnLqdGFP
                                   || |  ||||||  | |||  ||| |||||||||| |  |||
  1                           mleqpYLdLakKVLDEGhfKpDRThTGTySIFGHQMRFdLskGFP

    -----------------------------mkqYldlikkildeG--kdDRTgTGTlsiFGhqmRfdL-dgFP



 73 llTTKrvfWKgvleELlWFikGSTN       akelsskGvkiWD
      |||   ||    || ||  ||||             |   ||
 44 aVTTKklaWKacIaELiWFLsGSTNvndlrliqhdsliqGkTvWDE
     ||||       | || ||| | ||                | |||
 44 LVTTKrchlrsIIhELLWFLqGDTN       IayLhennvTIWDEWA
    | |||      |  |||||| ||||       |  |      ||||||
 46 LlTTKkvpfglIksELLWFLhGDTN       IrfLlqhrnhIWDEWAfekwvksdeyhgpdmtdfghrsqk

    lvTTKkv-wk-ii-ELlWFl-GdTN-------i--ll--gvtiWDewa-----------------------



111            angsrdfldslgfstreegDLGpvyGfqwrhfGaeyRDmesdysgqGVDQlqrVIDtIKtn
                                  |||   |      |   ||        ||||   ||| ||
 90                         nyENqakDLGyhsGelgpiyGkqwRD      fgGVDQIieVIDrIKkL
                              ||   |||                            |||         |
 85                           dEN  GDLG        pVYGkQWRAWpTpdGrhIDQI   ttVlnQL
                                  ||||         ||| ||||| |  |  |||      |  |
111 dpefaavyheemakfddrvlhddafaakyGDLG        lVYGsQWRAWhTskGdtIDQ   lgdVieQi

    ----------------d--------en--gDLG---g-----vgg-qrdaw-t--g-giDQi--vidvikql



172 PdD   RRiImcAWNPrdLplMALPPCHalcQFyVvNseLscQlYQRSgDngLGvPFNIASYAlLtymiAhi
    | |   || |  ||||  |  |||||||   || | |  |  | |||| |  || ||||||||  |    |
133 PND   RRqIVSAWNPaELkyMALPPCHmFyQFnVrNGyLdlQwYQRSvDVFLGLPFNIASYAtLVHivAkm
     ||   || ||||||  ||  ||| ||| | || |  | |  | |||| ||||||||||||||| |||  |
120 KNDPdSRRiIVSAWNvgELdkMALaPCHaFfQFYVaDGKLScQLYQRScDVFLGLPFNIASYALLVHmmAqq
    |  | ||| ||||||       ||| |||   |||| ||||| |||||| | ||| ||||||||||  |  |
172 KthPySRRlIVSAWNpedvptMALpPCHtlyQFYVnDGKLSlQLYQRSaDiFLGvPFNIASYALLtHlvAhe

    kndp-sRRiIvsAWNp-elp-MALpPCHafyQFyV-dgkLscQlYQRS-DvfLGlPFNIASYAlLvhmvAh-
```

```
241 tgLkPGDfIhtlGdaHIYlNHiEplKiqLqREP            rpFpkLrilrkvekiddfkaeDFqieg
      | ||| |   |  ||| || |  |  | |||              |  |               ||
202 CnLiPGDlIfsGGnTHIYmNHvEQcKeiLrREPkelcelvisgLPykFryLstKeqlkyvlklRpkDFvlnn
    | |  ||    || || | ||  |    | |||          ||     |  |         |  ||
192 CdLEVGDFvwTGGDTHLYsNHmDQthlQLSREP        RPLP    kLiiKrkpesIFDyRfeDFeieg
    | |||| |  | || ||| || ||   |||| |        || |     |        |||    |
244 CgLEVGeFihTfGDaHLYvNHlDQikeQLSRtP        RPaP    tLqlnpdkhdIFDfdmkDiklln

    cgLepGdfihtgGdtHiY-NH-dq-keqLsReP--------rplp--f-kL-ikrk-e-ifdfr-eDf-ieg


301 YnpHPtIKmeMAV
    |  || ||  |||
274 YvsHPpIKgkMAV
    |  || ||   |
252 YDPHPgIKAPVAi
    ||| | ||||||
304 YDPyPaIKAPVAv

    YdphP-IKapmAv
```

```
ALIGNMENT SCORE = 132.00
Scoring matrix:
     1     2     3     4
 1        52.0  93.0  12.0
 2              87.0  36.0
 3                   106.0
```

Clustered Pair-Wise 'Needleman-Wunsch' Alignment

in 'identity (no translation)' alphabet of:

1. T4 Phage (1-286)
2. E.coli (1-264)
3. Lactobacillus casei (1-316)
4. Human (1-313)

listed in clustered order.

```
 1 M KQ YqdLikdifenGyetdDRTGTGTialFGsklRwdLtkGFPaVTTKklawkacIaELiWFLsGsTnvN
   | || |  |       |    |||||||   ||   |  |  ||| ||||       | || ||| | |  |
 1 M KQ YLeLmqKVLDEGtqKnDRTGTGTlSIFGHQMRFnLqdGFPLVTTKrchlrsIIhELLWFLqGDT  N
   |  | || |  ||||||  | ||| ||| ||||||||| |  |||| |||      |  |||||| |||  |
 1 MleQpYLdLakKVLDEGhfKpDRThTGTySIFGHQMRFdLskGFPLlTTKkvpfGlIksELLWFLhGDT  N
   |       |    |       |                                |
 1 M pvagseLprrpLppaaqerDaeprpphgelqylgqiqhilrcgvrkddrtgtG  tlsvfgmqaryslrd

   M-kq-yldL-kkvldeg-qk-Drtgtgt-sifghqmrfdl-kgfplvttkk---g-ii-ellwfl-gdt--n


71  dlrL  iqH              dsliqgkT      vWD         E             nyenqakD
       |    |                     |       ||         |                    |
69 I ayL    H       EnnV          T      iWD         E wA         D    enGD
   |   |    |       |  |          |        |         |  |         |      ||
71 IrfLLqhrnHiWdewafEkwVksdeyhgpdmTdfghrsqkDpefaavyheE mAkFddRvlhdDafaakYGD
      ||      |     |                      |              |  |    |     ||
70 efpLLttkrvfWkgvleEllwfikgstnakelsskgvkiwD  angsrdfldslgFstR eegD lgpvYGf

   i--lL----h-w-----e--v----s-----t------iwD---------e--a-f--r----d-----ygd
```

```
 98 LGyhsGelgpiygkqwrDfggvDQIieVidriKklPndRRqIVSAWNpaELkyMALpPCHmFyQFnVrnGyL
    ||   |           |    |||  |    |  |  || ||||||  ||  ||| ||| | || |  | |
 90 LGpVYG kQWR AWpTpDGrhIDQIttVlnQlKndPdSRRiIVSAWNvgELdkMALaPCHaFfQFYVaDGKL
    || |||  ||| || |  |  |||   |  | |  | ||| ||||||      ||| |||   |||| ||||
142 LGlVYG sQWR AWhTskGdtIDQLgdVIeQIKThPySRRlIVSAWNPeDvPtMALPPCHtLyQFYVnDGKL
        |    |       |   |||  ||  ||| |  || |  |||| | | |||||||| | ||||    |
138 qwrhfG aeyRdmesdysGqgvDQLqrVIdtIKTnPddRRiImcAWNPrDlPlMALPPCHaLcQFYVvnseL

    lg-vyG--qwr-aw-t-dg-giDQi--VidqiKt-PdsRRiIvsAWNp-dlp-MALpPCHafyQFyV-dgkL


170 dlQwYQRSvDVFLGLPFNIASYAtLVHivAkmCnLipGDlifsGGnTHiYmNHveQckeiLrREPkeLceLv
      | |||| |||||||||||||| |||  |  | |  ||    || || | ||  |    | |||  |  |
160 ScQLYQRScDVFLGLPFNIASYALLVHmmAqqCdLEVGDFvwTGGDTHLYsNHmDQthlQLSREPRPLPkL
    | |||||| | ||| |||||||||| |  |  | |||| |  | || ||| || ||  |||| ||| | |
212 SlQLYQRSaDiFLGVPFNIASYALLTHlvAHeCGLEVGeFIHTfGDAHLYvNHlDQiKeQLSRtPRPaPtL
    | |||||| |  |||||||||||||||   ||  ||  | |||| |||| | ||    | || | ||| | |
209 ScQLYQRSgDngLGVPFNIASYALLTymiAHitGLkpGdFIHTlGDAHiYlNHieplKiQLqRePRPfPkL

    scQlYQRS-DvfLGlPFNIASYAlLthmvAh-cgLevGdfihtgGdaHlY-NH-dq-keqLsRePrplpkL-


242 IsglpyKfRylstKeqlkyvlklRpkDFvlnnYvsHPpIKgkmAv
    |     | |    |          |  ||    |  || ||   |
231 I    iK R    K  pesIFDyRfeDFeiegYDPHPgIKAPVAi
                      |||    |     ||| | ||||||
283             qlnpdkhdIFDFdmkDiklInYDPyPaIKAPVAV
                      | ||   |     | | | ||   ||
280             rilrkvekIdDFkaeDfqiegYnPhPtIKmemAV

    i-----k-r----k---e-ifdfr-eDf-iegYdphP-IKapvAv

ALIGNMENT SCORE = 280.00 Scoring matrix:

     1      2      3      4
 1         104.0   83.0   76.0
 2                110.0  100.0
 3                       102.0
```

It will be noticed in this example that individual residues have been aligned even though the resolution had been set to 2. This occurs because of the heuristic, previously mentioned, that when the lengths of the sequences between two successive regions of an alignment are the same, then refinement (telescoping) is terminated and matching is then done on the basis of position alone.

ADDITIONAL FEATURES

It is sometimes useful to change the alphabet in which an alignment is done. Thus, in aligning protein sequences, significant homologies might be revealed by grouping the amino acids into equivalence classes, such as hydrophobic and hydrophilic. Four standard alphabets are provided with the program as well as the feature of a user-specified alphabet. The purine-pyrimidine alphabet is provided for nucleic acids and, as for proteins, there is the user-specified alphabet feature.

Evaluating the significance of a multisequence alignment is, in general, a difficult task, and is perhaps best approached by redoing the alignment with the concerned sequences being randomized (composition preserved). This feature is provided.

TECHNICAL DESCRIPTION and AVAILABILITY

The GENALIGN program is written in the C language, provides menu driven interaction and is completely self-documented. Dynamic memory allocation is used throughout so that the number and size of the sequences which can be handled is limited only by hardware considerations. The resident size of the program is 90K bytes and consists of 20 modules.

Binary code for academic distribution is available from UCSFBCL. Intelligenetics, Inc. has been licensed to revise and distribute the program.

ACKNOWLEDGEMENT

I thank Prof. Daniel Santi for the use of his peptide sequences in the above example.

REFERENCES

1. Sobel, E. and Martinez, H.M. (1986) Nucl. Acids Res.14:363-374.
2. Taylor, W.R. (1987) CABIOS 3:81-88.
3. Needleman, S.B. and Wunsch, C.D. (1970) J. Mol. Biol. 48:444-453.
4. Karlin, S; Ghandour, G; Ost, F; Tavare; Korn, L.J. Proc. Natl. Acad. Sci. USA 80:5660-5664.
5. Martinez, H.M. (1983) Nucl. Acids Res. 11:4629-4634.