CABIOS

Vol. 10 no. 3 1994
Pages 301–307

A Workbench for large-scale sequence homology analysis

Erik L.L.Sonnhammer and Richard Durbin

Abstract

When routinely analysing very long stretches of DNA sequences produced by genome sequencing projects, detailed analysis of database search results becomes exceedingly time consuming. To reduce the tedious browsing of large quantities of protein similarities, two programs, MSPcrunch *and* Blixem, *were developed, which assist in processing the results from the database search programs in the BLAST suite.* MSPcrunch *removes biased composition and redundant matches while keeping weak matches that are consistent with a larger gapped alignment. This makes BLAST searching in practice more sensitive and reduces the risk of overlooking distant similarities.* Blixem *is a multiple sequence alignment viewer for X-windows which makes it significantly easier to scan and evaluate the matches ratified by* MSPcrunch. *In* Blixem, *matches to the translated DNA query sequence are simultaneously aligned in three frames. Also, the distribution of matches over the whole DNA query is displayed. Examples of usage are drawn from 36* C.elegans *cosmid clones totalling 1.2 megabases, to which these tools were applied.*

Introduction

With the arrival of large-scale genome sequencing projects (Oliver *et al.*, 1992; Sulston *et al.*, 1992), where highly automated laboratory techniques produce DNA sequences at an ever-increasing rate, the need for equally powerful sequence analysis tools has become obvious. Characterizing genes found in 'blindly' sequenced DNA by searching for homologous proteins is presently the only means of predicting their function. Thanks to recent developments of high-speed database searching programs like BLAST (Altschul *et al.*, 1991), BLAZE (Brutlag *et al.*, 1993) and FLASH (Rigoutsos and Califano, 1993), searching time is of little concern. Instead, the bottleneck lies in the manual evaluation of the matches reported by the search programs, which often form a list of many thousands of potential homologies. Common obstacles are spurious matches from regions of biased composition and large protein families that tend to overshadow a few weak but relevant matches. Restricting the amount of results by using a high score cutoff only makes the problem of missing distant similarities worse. This is very undesirable, since the distant matches generate just as much scientific interest as the close ones. However, manual reading of exceedingly long search result lists becomes an inhuman task for sequencing projects of several megabases. What is needed is a workbench which automatically performs the routine actions of a sequence analyst as well as presents the cases where manual inspection is necessary in an interactive user-friendly environment (Bernstein, 1987; Schuler *et al.*, 1991).

This paper describes such a workbench which was specially developed for the *C.elegans* genome sequencing project. It is currently based on output from the database search programs BLASTX, BLASTP, BLASTN and TBLASTN, which produce a list of ungapped alignments, or locally maximal segment pairs (MSPs) (called HSPs in BLAST). First, a set of rules are applied by a program *MSPcrunch* to filter out as many unwanted matches as possible, by compensating the score for compositional bias and by limiting the number of matches in congested regions. Weak matches that are potentially distant similarities are kept, however, if they support each other as being conserved regions of a gapped alignment. After filtering, the accepted matches can be viewed as a multiple sequence alignment in the graphical tool *Blixem* running under X-windows, which aligns all matches in a scrollable window. *Blixem* relies on a program *Fetch* for information retrieval from any sequence database equipped with EMBL index files (Fuchs and Stoehr, 1993).

Alternatively, the accepted matches can be imported into the genomic database ACEDB (R.Durbin and J.Mieg, unpublished), from which *Blixem* can be called up. ACEDB also contains a gene prediction package based on GeneFinder (P.Green and L.Hillier, unpublished), which together with *Blixem* forms an interactive system for making gene predictions where homology to other proteins can be analysed in detail.

System and methods

For generating the MSPs, BLASTX v. 1.3.7 was used with the following parameters:

blastx swir3 ⟨query⟩ B = 1 000 000 S = 50
M = BLOSUM62-12 V = 0

Sanger Centre, Hinxton Hall, Cambridge CB10 1RQ, UK and MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 2QH, UK

where swir3 is the database, B the maximum number of MSPs, S the score cutoff, M the scoring matrix and V the size of the high-scoring list. The protein sequence database searched, Swir3, consisted of 56 280 sequences. It was constructed by combining 470 sequences from Wormpep release 3, 31 783 sequences from SwissProt release 26 (Bairoch and Boeckman, 1991) that were not derived from Wormpep, and 24 027 sequences from PIR release 37 (Barker *et al.*, 1992) that were not included in either SwissProt or Wormpep entries. Wormpep is a Sanger Centre in-house database containing all predicted proteins so far from the *C.elegans* genome sequencing project. The large value of the B parameter is somewhat arbitrary, provided it is big enough not to limit the number of MSPs reported. BLASTP also has a second-pass score cutoff S2 which is applied in a second search, performed only on database sequences with a match scoring above S in the first scan. Using a low S2 instead of S reduces the amount of spurious hits reported by BLAST, but at present S2 is not officially supported by BLASTX. BLOSUM62-12 is a modified version of the BLOSUM62 matrix supplied by the NCBI. Our modification was to lower the score for stop codons from −4 to −12. Such a high penalty for stop codons is preferable for DNA sequences with very low error rates.

Running BLASTX on very long DNA query sequences (> 100 000 bases) may prove impossible due to memory limitations. For such cases, we have developed a program *Seqsplit* which splits up the query into smaller chunks with overlaps. After running BLASTX on the smaller chunks, another program, *Blastunsplit*, combines all the output files into one and reconstructs the positions in the original query.

BLASTP, BLASTX, *MSPcrunch*, *Blixem*, *Fetch*, *Seqsplit* and *Blastunsplit* were run on Unix workstations from Silicon Graphics running Irix 4.0.5, and Sun running SunOS 4.1.3. All programs were written in ANSI C. The graphics routines used in *Blixem* are part of the ACEDB graphics library (R.Durbin, unpublished) and require X-windows. *Blixem* also requires *Fetch* and external protein databases with EMBL index files, which can be created by programs in the Staden package (Staden and Dear, 1992).

Algorithm

The post-processing of MSPs from BLAST in *MSPcrunch* is outlined in Figure 1. An MSP consists of an ungapped alignment between a region in the query sequence, simply called 'the query' hereafter, and a region of a database sequence, called 'the subject'.

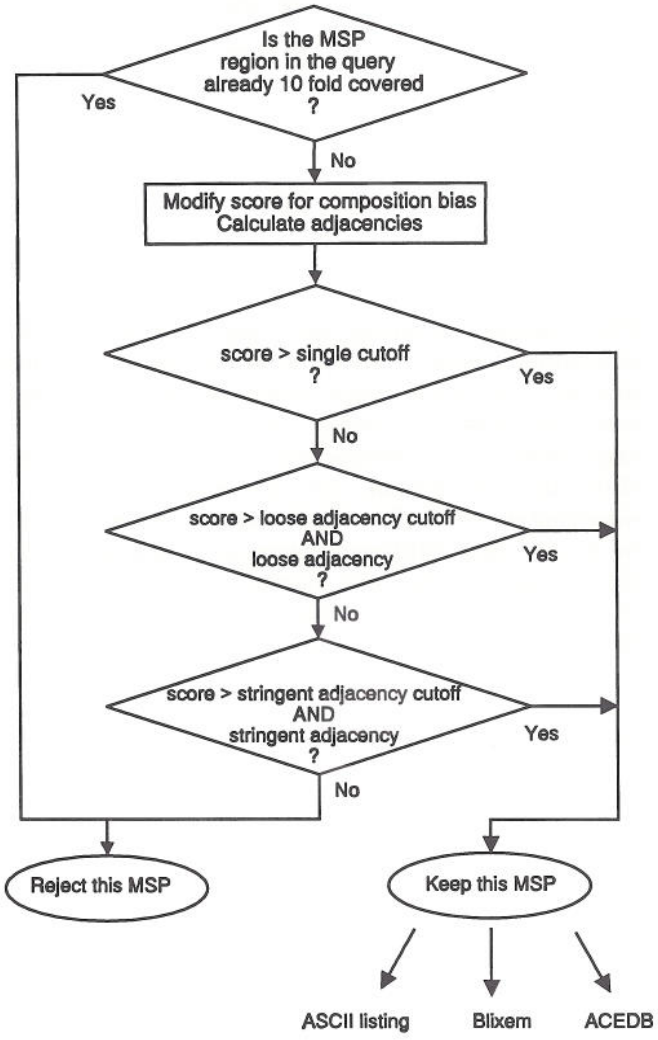

Fig. 1. Flow diagram of actions taken by *MSPcrunch* to filter out unwanted matches from BLAST. The methods for calculating the modified score and adjacencies as well as the values used for the cutoffs are described in the Algorithm section of the text.

Limit the coverage of MSPs

If the segment in the query of an MSP is already covered by many other MSPs that score higher, the MSP is rejected to avoid redundant data. Figure 2 shows the MSP coverage on a cosmid sequence of 40 kb. The major causes of the very high number of MSPs covering certain regions are strong amino acid frequency bias and very large protein families. We limit the coverage by default to 10-fold on each strand. If every residue in the query has this coverage already, the MSP is rejected.

Calculate a modified score for biased comparison MSPs

The *expected score* of an MSP is the average score that two random sequences of that length and amino acid composition would have. For a typical MSP the expected

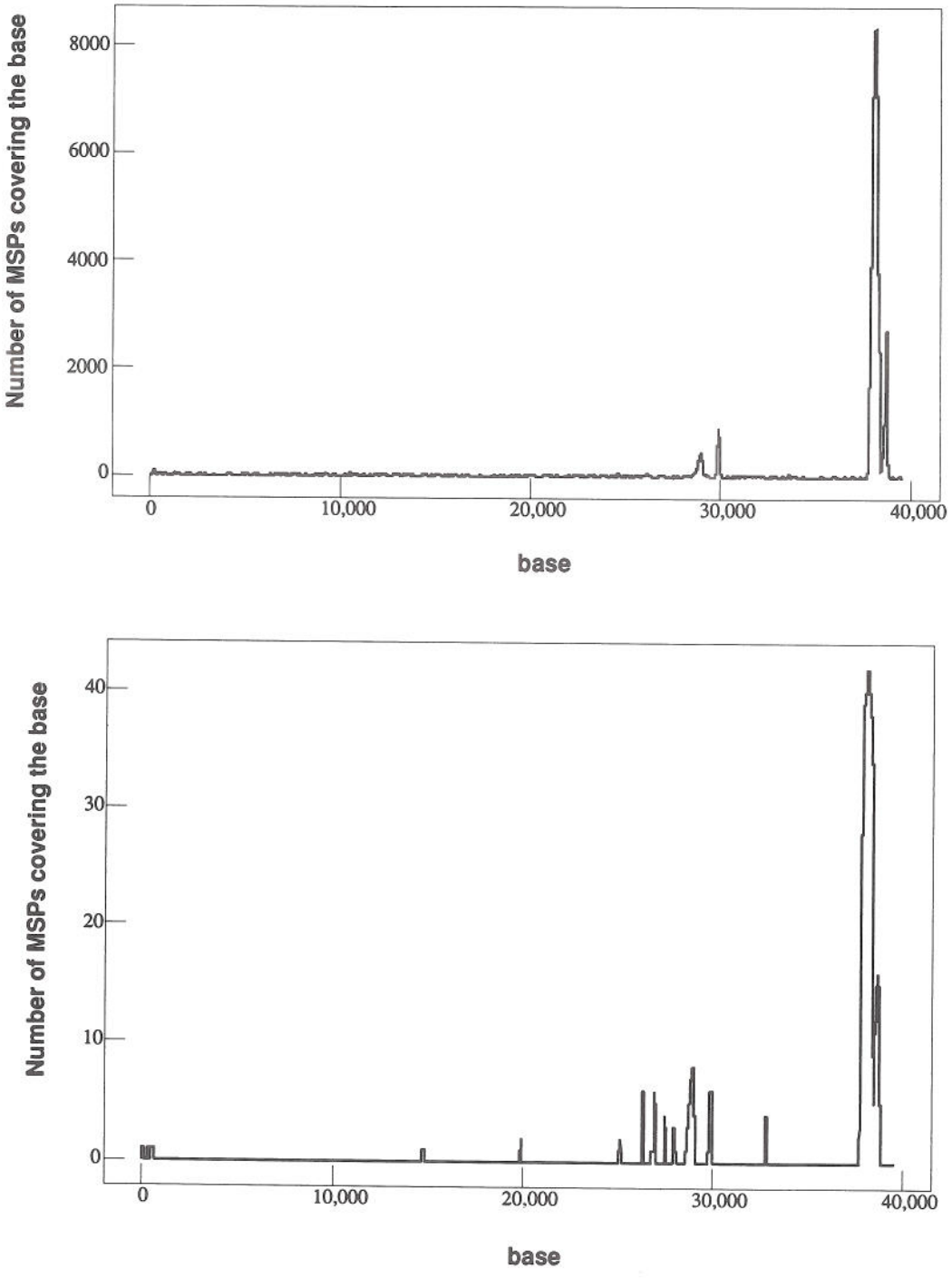

Fig. 2. Histograms of MSP coverage on both strands of the 40 kb DNA query sequence from the *C.elegans* cosmid ZK643 (**a**) before and (**b**) after *MSPcrunch*. MSPs were generated by BLASTX as described in System and methods. The number of MSPs was reduced from 28 450 to 164 by *MSPcrunch*. The peak at 38 000 is the result of a repetitive region which gives rise to very biased amino acid sequences [poly(G) in the positive strand and poly(P) in the negative] and the peaks at 29 000 and 30 000 have a strong bias for charged residues. Most matches to the charged amino acid region were removed by the biased composition compensation mechanism. The G- and P-rich regions contain a great deal of high scoring matches, however, so the coverage limitation mechanism is needed to reduce the number of these matches. Instead of stopping at the coverage limit 10 the peak reaches 40, due to matches on both strands and staggering of matches that do not cover each other entirely. The main significant homology in this cosmid is to a G-protein coupled receptor (ZK643.3), located between bases 25 000 and 28 000. Only very weak matches are found, but because they are adjacent, they are kept by *MSPcrunch*. There is also a motif conserved with DCMP deaminases at 32 700–32 800. Other peaks in (b) correspond to predicted proteins from this cosmid.

score is negative, but if the composition is very biased the expected score may become positive. This positive contribution to the score of the MSP does not infer relatedness between the sequences and should therefore be removed.

A simple algorithm is used to calculate the expected score for a given MSP. Two vectors **Q** and **S** with the observed frequencies of the amino acids in the query and subject are constructed. The vectors are then scored against each other so that the expected score **E** equals:

$$\mathbf{E} = L \sum_{i=1}^{20} \sum_{j=1}^{20} \mathbf{Q}_i \mathbf{S}_j \mathbf{M}_{ij}$$

where L is the length of the MSP and **M** is the scoring matrix. This method yields the same result as random shuffling methods would asymptotically, but is faster. If the expected score is positive, it is subtracted from the original score. This modified score is used for all further cutoffs. If the expected score is negative, the original score will be used.

Test MSP adjacency

Low-scoring MSPs may be due to fragmentation caused by gaps in a larger alignment. Since these gapped alignments are potentially real, a lower score threshold should be used for adjacent MSPs that can be concatenated within some limits of allowed overlaps and gaps in the query and subject sequences. Adjacency can be explained by this example. Two sequences A and B have two MSPs in different regions A1 and B1, and A2 and B2 (region 1 is N-terminal to region 2). If they line up like this:

A1–A2
B1–B2

they are called adjacent, given that the gaps at A1–A2 and B1–B2 are reasonably small. In this case, is possible that there exists a larger alignment that spans both MSPs, but was split in two because of a gap. If, however, the MSPs line up like this:

A1–A2
B2–B1

the MSPs are *inconsistent* with there being a gap between them, and they cannot be called adjacent. The definition of adjacency completely depends on the chosen parameters for how big the gaps and overlaps between MSP may be. If the query was DNA there may also be an intron between A1 and A2, producing a large gap. As a general rule, the lower the score of the MSPs, the more stringent one ought to be with adjacency criteria. By empirical testing, we found a three-level test the most efficient. With no adjacency at all the score is required to be at least 75. With loose adjacency (defined below), the score cutoff is 60, while with stringent adjacency we accept MSPs scoring 50. Using even lower score cutoffs may improve sensitivity, but the amount of poor MSPs reported by BLASTX for long DNA queries becomes prohibitively large. If the query is a protein sequence, however, the stringent adjacency cutoff can readily be lowered to 40. The following parameters concern only BLASTX post-processing. To test adjacency between two MSPs, MSP1 and MSP2, where MSP2 is C-terminal of MSP1 in the subject sequence, we define the following variables:

Query_gap = (MSP2_QueryStart − MSP1_QueryEnd −1)/3
Subject_gap = MSP2_SubjectStart − MSP1_SubjectEnd −1
MSP_dist = minimum (Query_gap, Subject_gap)
MSP_gap = |Query_gap − Subject_gap|

The following criteria were used for the MSPs adjacency tests:

Loose adjacency

score > 60
−20 < MSP_dist < 300 a.a. (amino acids)
MSP_gap < 25 a.a.
Intron < 5000 bases

Stringent adjacency

score > 50
−20 < MSP_dist < 50 a.a.
MSP_gap < 25 a.a.
Intron < 500 bases

The introns are essentially Query_gaps. If the Query_gap is larger than the Subject_gap but smaller than the intron limit, the intron is excised by setting Query_gap equal to Subject_gap before calculating MSP_dist and MSP_gap.

Displaying the accepted MSPs

There are currently three ways to view the output of *MSPcrunch*:

As a listing of accepted MSPs in N- to C-terminal order. This verbose ASCII output of *MSPcrunch* is shown in Figure 3. The layout has been designed to be easy to read as well as easy to parse by other programs. Instead of sorting the MSPs in score order, like BLASTX and

```
> RTJK_DROME  P21328 RNA-DIRECTED DNA POLYMERASE (EC 2.7.7.49) (REVERSE TRANSCRIPTASE) (MOBILE ELEMENT JOCKEY).
--------------------------------------------------------------------------------------------------------------

Score= 41, Expected score= -14, Adjacency= 1
Query:   F58A4.5       662 - 677     PNRWKHAVIIPIPKKG
                                     P  WK A II I K G
Subject: RTJK_DROME    491 - 506     PKAWKSASIIMIHKTG

Score= 64, Expected score= -22, Adjacency= 1
Query:   F58A4.5       679 - 706     PSSPSNYRPISLTDPFARIMERIICSRI
                                     P+   +YRP SL     +IMER+I +R+
Subject: RTJK_DROME    509 - 536     PTDVDSYRPTSLLPSLGKIMERLILNRL

Score= 69, Expected score= -51, Adjacency= 1
Query:   F58A4.5       718 - 770     QHGFLNFRSCPSSLVRSISLYHSILKNEKSLDILFFDFAKAFDKVSHPILLKK
                                     Q GF      P  L R ++     ++N++     F D  +AFD+V HP LL K
Subject: RTJK_DROME    550 - 602     QFGFRLQHGTPEQLHRVVNFALEAMENKEYAVGAFLDIQQAFDRVWHPGLLYK

Score= 42, Expected score= -11, Adjacency= 1
Query:   F58A4.5       785 - 803     KEFLHLRTFSVKINKFVSS
                                     K FL  RTF V ++ + SS
Subject: RTJK_DROME    616 - 634     KSFLEERTFHVSVDGYKSS

Score= 81, Expected score= -20, Adjacency= 1
Query:   F58A4.5       803 - 830     SNAYPISSGVPQGSVSGPLLFILFINDL
                                     S+  PI++GVPQGSV GP L+ +F +D+
Subject: RTJK_DROME    633 - 660     SSIKPIAAGVPQGSVLGPTLYSVFASDM

Score= 42, Expected score= -14, Adjacency= 1
Query:   F58A4.5       910 - 927     LGLITDLKLNFEPHIIKI
                                     LG+  D KL F  HI  I
Subject: RTJK_DROME    753 - 770     LGITLDRKLTFSRHITNI

Score= 45, Expected score= -22, Adjacency= 1
Query:   F58A4.5       952 - 980     HIFKTYVAPIINYCSEIYSPSPNSSLSAI
                                     +I+K+ +AP + Y  ++Y  +  S L+ I
Subject: RTJK_DROME    798 - 826     NIYKSILAPCLFYGLQVYGIAAKSHLNKI
```

Fig. 3. Output generated by *MSPcrunch* on BLASTP results. These low-scoring MSPs, involving predicted protein no. 5 in the *C.elegans* cosmid F58A4 and the *D.melanogaster* reverse transcriptase RTJK_DROME (SwissProt accession no. P21328) contained in the *jockey* mobile element, were accepted by *MSPcrunch* given the criteria described in Algorithm. *Expected score*, the score expected by an MSP of two random sequences with the same length and composition as the MSP in question. If the expected score is positive, the difference between the original score and the expected score is used for the acceptance criteria. *Adjacency*, 1 if the MSP is supported by a neighbour MSP, 0 if not. The MSPs are sorted by position to show clearly how the fragments fit together in a larger gapped alignment.

BLASTP do, *MSPcrunch* sorts them by position from N to C terminus in the database sequence. This way a much better appreciation of the global alignment with gaps is gained, if it exists.

View the MSPs as a multiple sequence alignment in Blixem. If a long stretch of DNA is analysed, like a whole cosmid sequence, it is usually preferable to walk along the query and inspect all the matches in each region. This way both strong and weak matches reinforce the confidence that the homology is real, something that is easily lost if the matches are inspected in high-scoring order. As seen in Figure 4, *Blixem* provides a user-friendly interface to get the big picture.

Another advantage of *Blixem* is that by clicking on a match, the entire annotation of the database entry is fetched from the database and is presented in a separate window. This way, the function of the region in question can be assessed by consulting the feature table of the database entries. The retrieval of data from the databases is based on a program *Fetch*, which uses the EMBL database indexing system.

The user can choose to sort the MSPs by score, name or position. Score sorting is practical when one wants to highlight the closest relative, name sorting if a certain protein has many MSPs on the screen that one wishes to see next to each other, while position sorting usually produces the cleanest looking alignment. The alignment currently on the screen can be printed on a Postscript printer.

Export the MSPs to ACEDB. This is particularly useful if the user wants to predict genes in a DNA query sequence, since ACEDB includes a semi-automatic gene prediction environment. *Blixem* can be called up from ACEDB's sequence window so that exon predictions can be validated in the light of homology to other proteins. When called from ACEDB, *Blixem* integrates the display of predicted exons into the multiple sequence alignment (see Figure 4).

Implementation

MSPcrunch was used to process the BLASTX results for 36 cosmid sequences from the *C.elegans* sequencing project, totalling 1.2 megabases of DNA. With the parameters described above, BLASTX produced 277 292 MSPs, of which 6463 were kept by *MSPcrunch*. Time

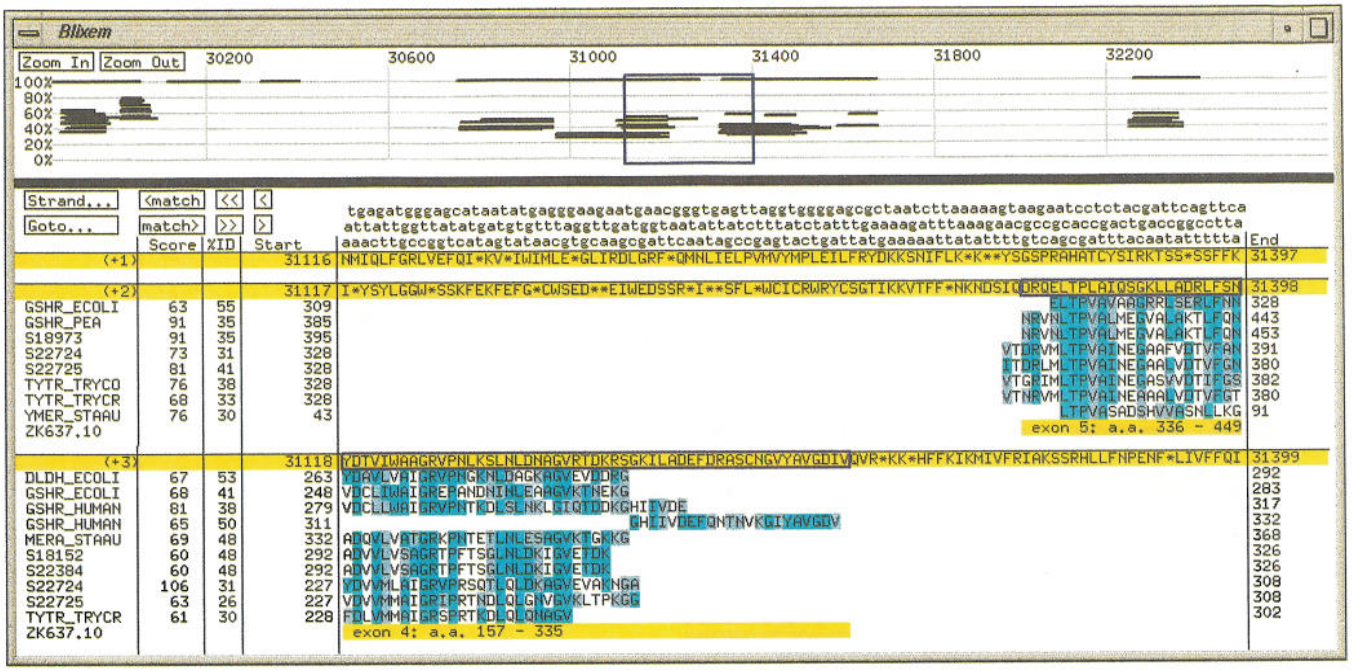

Fig. 4. Multiple alignment of MSPs in *Blixem* when called from ACEDB. Example taken from the *C.elegans* cosmid ZK637, showing matches between the predicted gene ZK637.10 and glutathione reductases. The top display shows a global overview of the MSPs in the vicinity of the multiple alignment window in the bottom display. Every MSP becomes a line on the corresponding position and percentage identity level in the overview display. The box in the overview marks the position of the current window in the multiple alignment below. The lines on 100% identity are the predicted exons of ZK637.10, of which all exons (1–6) are seen at this zoom level. The lower display shows the DNA sequence on top, three-frame translation on the yellow lines and the aligned MSPs in blue; dark blue for identical residues and light blue for conserved substitutions. Predicted exons are marked in the lower display as boxes on the translated genomic sequences on lines (+2) and (+3) and as the lines ZK637.10 which show what amino acid residues the exon corresponds to in the encoded protein.

consumption for BLASTX was 100 h on a Silicon Graphics Indigo RS4000, while *MSPcrunch* took 4 min. *MSPcrunch*, *Blixem*, *Fetch*, *Seqsplit*, *Blastunsplit* and *Wormpep* are available by anonymous FTP from ftp.sanger.ac.uk in the directory /pub/MSPcrunch or by sending e-mail to esr@sanger.ac.uk.

Discussion

Biased composition sequences are a problem in sequence analysis since most programs are not designed to handle them correctly. The most common solution is to locate stretches of biased composition and erase them *before* searching the database (Claverie and States, 1993). The drawback of such approaches is that highly significant matches may be lost if they occur in biased regions. Here we have circumvented that drawback by limiting the amount of matches to such regions *after* the searching.

The reduction of redundant results due to large protein families was achieved here by rejecting excess matches to a given region. A more subtle way of accomplishing this is to search a pre-clustered database. Instead of finding similarities to every member of the family, a single match would be found to the entire family, thus giving the relations to all other members of the family, not only the closest relatives. Presently available collections of protein families such as BLOCKS (Henikoff and Henikoff, 1992) and ProDom (E.Sonnhammer and D.Kahn, in press) could be used for this. Given that the number of protein sequences grows faster than the number of families (Green *et al.*, 1993), a pre-clustering approach seems very desirable.

A major criticism of using ungapped alignments is that distantly related proteins generally can only be aligned by inserting gaps. However, the regions that require gaps usually correspond to loops between secondary structure elements in the three-dimensional structure, where the length of the loop may vary. The loop residues can often not be aligned structurally, which makes sequence alignments of these regions rather meaningless. Also, the result of algorithms that produce gapped alignment depend strongly on a somewhat arbitrary gap penalty. We have therefore not attempted to concatenate adjacent MSPs into gapped alignments, but feel that most information is already present in the ungapped MSPs.

A further advantage of ungapped alignments is that repeated and shuffled domains in one sequence can be detected, something that is often compromised by programs that produce a gapped alignment.

If several MSPs are found adjacent, a further step could be to calculate a composite score for them, which would increase the confidence in the similarity. Such a scheme reintroduces the problem of gap penalties and was beyond the scope of this work. A different approach to calculate a composite score, which avoids the gap problem, is taken by the BLAST programs. For n MSPs, they calculate a combinatorial Poisson probability $P(n)$. This way a larger alignment which was fragmented into several low-scoring MSPs due to gaps can be ranked high. This works well when a larger gapped alignment does exist, but falsely high Poisson rankings may arise from spurious hits that are not adjacent, especially those involving biased composition matches. In addition, the Poisson probabilities calculated in BLAST increase with the size of the database, because the expected number of spurious matches increases slowly as the database grows. However, the true match scores do not change, and because many of the new sequences are homologous to existing ones, the Poisson correction often overestimates the drop in significance. In any case it is more convenient to work with a measure of similarity that remains stable for a particular match. For these reasons we designed *MSPcrunch* to work only with the raw scores.

Acknowledgements

We thank Dr Sean Eddy for helpful discussions on the biased composition problem. E.S. was supported by a Wellcome Trust fellowship. The Sanger Centre is supported by the Wellcome Trust and the MRC.

References

Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Bairoch,A. and Boeckmann,B. (1991) The SWISS-PROT protein sequence data bank. *Nucleic Acids Res.*, **19**, 2247–2249.

Barker,W.C., George,D.G., Mewes,H.W. and Tsugita,A. (1992) The PIR-International protein sequence database. *Nucleic Acids Res.*, **20**, 2023–2026.

Bernstein,M. (1987) Reducing the man–machine barrier: the sequence analysis workbench. *Comput. Applic. Biosci.*, **3**, 229–232.

Brutlag,D.L., Dautricourt,J.P., Diaz,R., Fier,J. and Stamm,R. (1993) BLAZE^{tm}: an implementation of the Smith–Waterman sequence comparison algorithm on a massively parallel computer. *Comput. Chem.*, **17**, 203–207.

Claverie,J.M. and States,D.J. (1993) Information enhancement methods for large scale sequence analysis. *Comput. Chem.*, **17**, 191–201.

Fuchs,R. and Stoehr,P.J. (1993) EMBL-Search—a CD-ROM based database query system. *Comput. Applic. Biosci.*, **9**, 71–77.

Green,P., Lipman,D.J., Hillier,L., Waterson,R., States,D. and Claverie,J.M. (1993) Ancient conserved regions in new gene sequences and the protein databases. *Science*, **259**, 1711–1716.

Henikoff,S. and Henikoff,J.G. (1991) Automatic assembly of protein blocks for database searching. *Nucleic Acids Res.*, **19**, 6565–6572.

Oliver,S.G., Van der Aart,Q.J.M., Agostini-Carbone,M.L., Aigle,M., Alberghina,L., Alexandraki,D., Antoine,G., Anwar,R., Ballesta,J.P.G., Benit,P., Berben,G., Bergantino,E., Biteau,N., Bolle,P.A., Bolotin-Fukuhara,M., Brown,A., Brown,A.J.P. *et al.* (1992) The complete DNA sequence of yeast chromosome III. *Nature*, **357**, 38–46.

Rigoutsos,I. and Califano,A. (1993) dFLASH: a distributed fast look-up algorithm for string homology. *IEEE Comput. Sci. Engng.*, in press.

Schuler,G.D., Altschul,S.F. and Lipman,D.J. (1991) A workbench for multiple alignment construction and analysis. *Proteins*, **9**, 180–191.

Sonnhammer, E.L.L. and Kahn, D. (1994) Modular structure of proteins as inferred from analysis of homology. *Prot Sci*, in press.

Staden,R. and Dear,S. (1992) Indexing the sequence libraries: software providing a common indexing system for all the standard sequence libraries. *DNA Seq.*, **3**, 99–105.

Sulston,J., Du,Z., Thomas,K., Wilson,R., Hillier,L., Staden,R., Halloran,N., Green,P., Thierry-Mieg,J., Qiu,L., Dear,S., Coulson,A., Craxton,M., Durbin,R.K., Berks,M., Metzstein,M., Hawkins,T., Ainscough,R. and Waterston,R. (1992) The *C.elegans* genome sequencing project: a beginning. *Nature*, **356**, 37–41.