MSPcrunch: a BLAST enhancement tool for large-scale sequence similarity analysis

By Erik L.L. Sonnhammer and Richard Durbin.

Summary

For high-throughput genomic sequence similarity analysis, most database search programs generate prohibitively large amounts of information. To allow an annotator to concentrate on essential tasks, an automatic system was developed that makes many of the standard decisions a trained sequence analyst would, and only presents the most informative matches to the annotator.

The system is currently based on the database search programs in the BLAST suite, which have been primarily designed for short query sequences. A number of algorithmic additions were made that are especially valuable for multi-domain queries, which is the norm for genomic cosmid-size sequences. It is implemented by changing some input parameters to BLAST, and applying a number of filtering rules to the output by a post-processing program, called MSPcrunch.

The main advantages compared to default BLAST searching are: 1. Domains with weak but significant hits will not be missed due to other higher-scoring domains. 2. 'Junk' matches with biased composition are eliminated. 3. Higher sensitivity and selectivity are achieved for multiple matching segments in the 'twilight zone', thanks to strict consistency criteria. 4. A range of output formats is provided, including gapped alignment, graphical schematic and tabular data.

The default mode of operation has been calibrated empirically to suit the needs of efficient and sensitive genome annotation. This mode will remove redundant matches even if they are significant, but will report the highest scoring of the statistically insignificant matches, since these may prove to be biologically relevant after further analysis.

Introduction

Large scale genome sequencing projects [Wilson *et al.*, 1994; Dujon, 1996], generate new sequences at such a high rate that homology analysis has become a serious bottleneck. The availability of fast database searching programs such as BLAST [Altschul *et al.*, 1991], Fasta [Pearson, 1990],and Flash [Rigoutsos and Califano, 1993], and fast parallel computing hardware such as MasPar [Brutlag *et al.*, 1993], DAP [Collins *et al.*, 1988] and Bioccelerator [Esterman, 1995] ensure that the actual computation is a more or less solved problem for at least the foreseeable future. Analysing the search results generated from a hundred thousand newly sequenced basepairs per day, however, presents a major challenge. Available search programs produce results that are too time-consuming to digest on a large scale. For genome projects, there is a demand for automated analysis systems [Scharf *et al.*, 1994]. Our philosophy is that human evaluation is still a required for high-quality sequence analysis, but many monotonous time-consuming tasks can be automated by computational methods. Presented here is a program, MSPcrunch, which applies a number of rules to evaluate matches reported in a database search, and concisely presents only the most relevant information for further consideration.

Much of the recent developments of database search programs has been refinement of the statistical significance of a match [Karlin and Altschul, 1990], i.e. finding the probability that a match was caused by chance. For most search algorithms, the *extreme value distribution* [Gumbel, 1958] has proven the best model so far for describing the distribution of optimal scores to database sequences [Altschul *et al.*, 1994]. Statistical significance of a match calculated under this model has been shown to generally agree well with biological relevance.

However, sequences that are unusually rich in a few amino acid types may give rise to spurious matches that under the random model are very significant. A solution to this is to detect low complexity regions and remove them from the query [Wootton and Federhen, 1993; Claverie and States, 1993], but this has the drawback of artificially disrupting the query sequence, which may lead to missed true similarities [Koonin *et al.*, 1996b]. The risk of this is reduced if only the abnormally frequent residues are removed [Casari, personal communication]. In this chapter a method for avoiding matches caused by biased composition is described, that instead of modifying the query, compares the observed score to the expected score, given the residue composition. If the observed score was deemed to be the result of compositional bias, the match is rejected.

When the amino acid composition is not biased, the statistical significance of single matches reported by BLAST [Altschul *et al.*, 1990] is normally reliable. The significance of multiple matching segments is however difficult to calculate properly using the extreme value distribution, and BLAST uses a heuristic to decide which segments are consistently ordered with respect to each other [Karlin and Altschul, 1993]. This heuristic does not take the distance between two segments into account, and often overestimates the combined significance of what are really independent segments [Koonin *et al.*, 1996b]. In this chapter, a method is described that uses empirically derived consistency rules for multiple matches between two sequences, which explicitly takes the distance between segments into account. This has a flavour of gap penalties, but the method is very different from dynamic programming.

MSPcrunch is implemented as an add-on tool for the BLAST programs, which were chosen because of their robustness, speed and the underlying philosophy of only looking for ungapped matching segments, which allows finding of matches to multiple independent domains. For genomic analysis, which involves finding protein matches to short exons interspersed between introns, this is a big advantage. The main problem with using BLAST for large multi-domain queries is that it by default only reports the highest scores. This can cause weakly conserved domains to be missed if other domains generate too many high-scoring matches. The problem can be alleviated by changing a parameter so that BLAST reports all matches. This can cause severe over-reporting, but MSPcrunch then removes redundancy in congested regions so that only the high-scoring matches of any given region are kept.

Statistical significance tends to work well to support clear similarities. Weakly significant matches may or may not infer homology. Matches in this so-called ‘twilight zone’ are a mixture of true and false similarities, and the problem is to separate the signal from the noise. For ungapped matches from BLAST, the problem of spurious matches is bigger than for dynamic programming methods that find the single best match.

After filtering out redundant and ‘junk’ matches, the accepted matches can be viewed either as a graphical ``Big Picture" schematic display with one database sequence per line, as a gapped alignment, or can be exported to other programs such as ACEDB or Blixem in a tabular form.

Methods and materials

MSPcrunch is in itself not a database search program, but relies on the programs Blastp, Blastn, Blastx, Tblastn and Tblastx from the BLAST suite [Altschul *et al.,* 1991]. This has the advantage that end-users can use any BLAST service provided on the internet and process the output with MSPcrunch. Another reason for implementing MSPcrunch as a post-processing filter was to keep it flexible to adaptation to output from other database search programs that may become popular in the future.

Version 1.4.6 of BLAST was used. BLAST searches for ungapped segments in two sequences and extends them until the maximum score is achieved. All such maximal segment pairs (MSPs) scoring above a certain threshold are reported. This threshold is normally set to report 10 spurious hits, but here we lower it to 25 for Blastp and 35 for Blastx, Tblastn and Tblastx, using the BLOSUM62 score matrix. A lower threshold causes BLAST to report more spurious and true MSPs, but MSPcrunch removes most of the spurious ones by consistency checks described below.

The BLAST B parameter was set to a high enough value so that it does not limit the number of MSPs reported (10^6). For Blastx on cosmids, we use the -span1 option to avid a too voluminous output, and the score matrix BLOSUM62-12, which is a slightly modified version of the BLOSUM62 matrix [Henikoff and Henikoff, 1992]. The 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. Note that for older versions than 1.4 of BLAST one needs to set the S (first pass cutoff) parameter set to a low value, since only database sequences that have a match above this threshold will be searched in the second pass, which looks for matches above S2.

Running Blastx on long DNA query sequences (more than 10^5 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 BLAST on the smaller chunks, another program Blastunsplit combines all the output files into one and reconstructs the positions in the original query.

The protein sequence database searched, Swir, is a low-redundancy collection of sequences from Wormpep, Swissprot and TREMBL [Bairoch and Apweiler, 1996]. Redundancy was removed by a program nuswir [P. Rice, personal communication], which rejects any sequence from TREMBL that is more than 95% identical to any Wormpep or Swissprot entry. Wormpep entries in Swissprot were also removed. 118182 Release 11 of swir consisted of 118182 sequences. 7299 of these sequences came from Wormpep release 11, 51474 from Swissprot release 33 and 59409 from TREMBL. See chapter 9 for more information on Wormpep.

MSPcrunch rules

The post-processing of MSPs from Blastx or Blastp in MSPcrunch is outlined in figure 1. An MSP consists of an ungapped alignment between a segment in the query sequence, simply called 'the query' hereafter, and a segment of a database sequence, called 'the subject'.

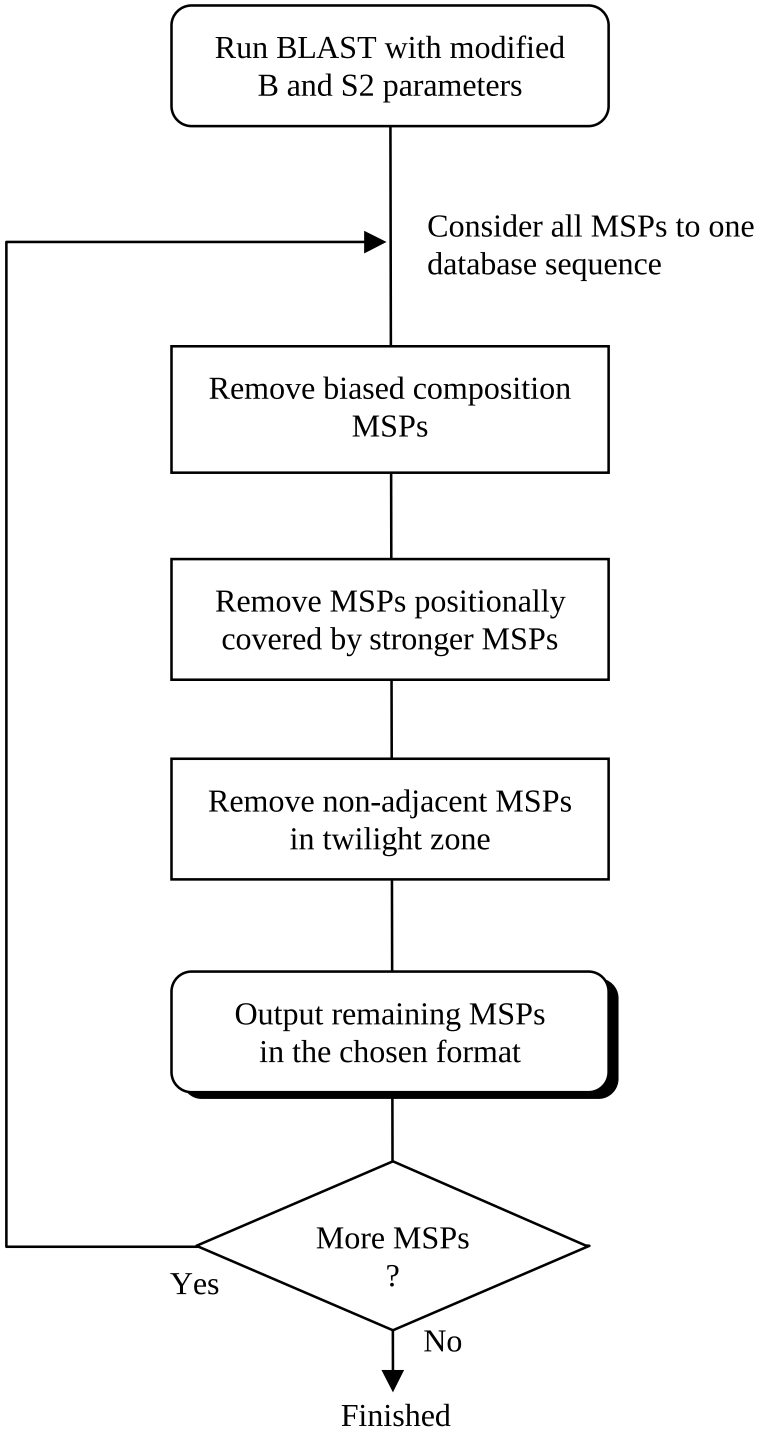

Run BLAST with modified B and S2 parameters
Consider all MSPs to one database sequence
Remove biased composition MSPs
Remove MSPs positionally covered by stronger MSPs
Remove non-adjacent MSPs in twilight zone
Output remaining MSPs in the chosen format
More MSPs ?
Yes
No
Finished

Figure 1. Overview of the different modules in MSPcrunch.

Biased composition matches

Biased composition MSPs are detected by a rule that compares the score of the MSP with the score of an MSP with no composition bias, in relation to the amino acid composition of the MSP in question.

The expected score of an MSP, S_{exp}, is the average score two random sequences of that particular length and amino acid composition would have. For a typical MSP the expected score is negative, but if the composition is biased the expected score may be positive. The expected score is calculated the following way: Two vectors *Q* and *D* with the observed frequencies of the amino acids in the query and database segments making up the MSP are constructed. The vectors are then scored against each other so that

$$S_{\mathrm{exp}} = L \sum_{i=1}^{20} \sum_{j=1}^{20} Q_i D_j M_{ij}$$

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. To avoid unjustified high values of S_{exp} due to small sample sizes in short MSPs, the frequencies Q_i and D_i are given pseudocounts according to

$$Q_i = \frac{Qc_i + \alpha \cdot p_i}{L + \alpha}, \quad D_i = \frac{Dc_i + \alpha \cdot p_i}{L + \alpha}$$

where Qc_i and Dc_i. are the counts of residue i in the query and database segments in the MSP. A good value for the pseudocount weight α was found to be 5 (cf. [Henikoff and Henikoff, 1996]). Using a lower weight tends to reject too many short true matches, while a higher weight may cause acceptance of too many biased composition matches.

To evaluate whether the score *S* of the MSP is the result of biased composition, we calculate the bias-ratio β :

$$\beta = \frac{S - S_{\mathrm{exp}}}{S - LM_{\mathrm{exp}}}$$

where M_{exp} is the frequency-weighted expected score of random (unbiased) sequences according to the scoring matrix used. For BLOSUM62, $M_{exp} = -0.945$. β can be used as an index of how biased the composition of the MSP is. As a rule, $\beta < 0.8$ is a clear sign that the MSP has a biased composition and should be rejected. Table 1 shows to what degree the unwanted biased composition MSPs are removed for different values of β . For values of β above 0.8, loss of good matches with slight bias becomes a problem.

A simpler and less effective version of this algorithm has previously been described. The method described here has been implemented since version 1.2 of MSPcrunch.

	YMH5_CAEEL		B0284.1		CA14_CAEEL		GRP_ARATH	
β	biased	good	biased	good	biased	good	biased	self
0.1	331	132	292	3	2503	27	1625	1
0.2	298	132	288	3	2485	27	847	1
0.3	221	132	277	3	2443	27	326	1
0.4	133	132	51	3	2418	27	66	1
0.5	23	132	191	3	2378	27	15	1
0.6	7	132	67	3	2255	27	3	0
0.7	0	132	15	3	1867	27	0	0
0.8	0	132	3	3	623	25	0	0
0.9	0	132	0	0	22	22	0	0

Table 1. Separation of biased composition matches from good ones by MSPcrunch as a function of the bias-ratio β . The numbers refer to counts of MSPs that passed the MSPcrunch adjacency criteria. No coverage limit was used (see below). YMH5_CAEEL (Swissprot P34472) has a stretch of biased composition (acid-rich) in the N-terminus as well as a reverse transcriptase domain and 3 C-type lectin domains (see figure 7. B0284.1 (Wormpep CE00650) has a charged-residue biased region. CA14_CAEEL (Swissprot P17139) is a collagen, containing mainly [Gxy] repeats. Although these matches have biased composition, they are to other collagens, and it is therefore useful that MSPcrunch does not reject all of them. GRP_ARATH is the most biased composition protein in Swissprot 28 (72% Glycine, relative entropy 2.0 bits). In this extreme case even the match to itself does not pass the biased composition test when β > 0.6.

Positional coverage limitation

For genomic cosmid-size analysis, this is perhaps the most important feature of MSPcrunch. BLAST has a settable limit for the number of highest scoring database sequences to report, which is by default set to 250. If matches to one domain fill this quota entirely, other weaker scoring domains will not be reported. To prevent this from happening, we set the limit in BLAST to a sufficiently high number that all matches are reported (B=1000000). MSPcrunch then limits the number of matches by taking the position in the query into account. If the query segment of an MSP is already covered by many other MSPs that score higher and are accepted by MSPcrunch, the MSP is rejected. Figure 2 shows the MSP coverage on a cosmid sequence of 40 kbases. The two main causes of very high number of MSPs covering certain regions are strong amino acid frequency bias and large protein families. We limit the coverage by default to 10-fold on each strand. An MSP is only rejected if every residue in the query segment is covered. In practise, due to staggering of matches, this leads to a coverage up to 20-fold.

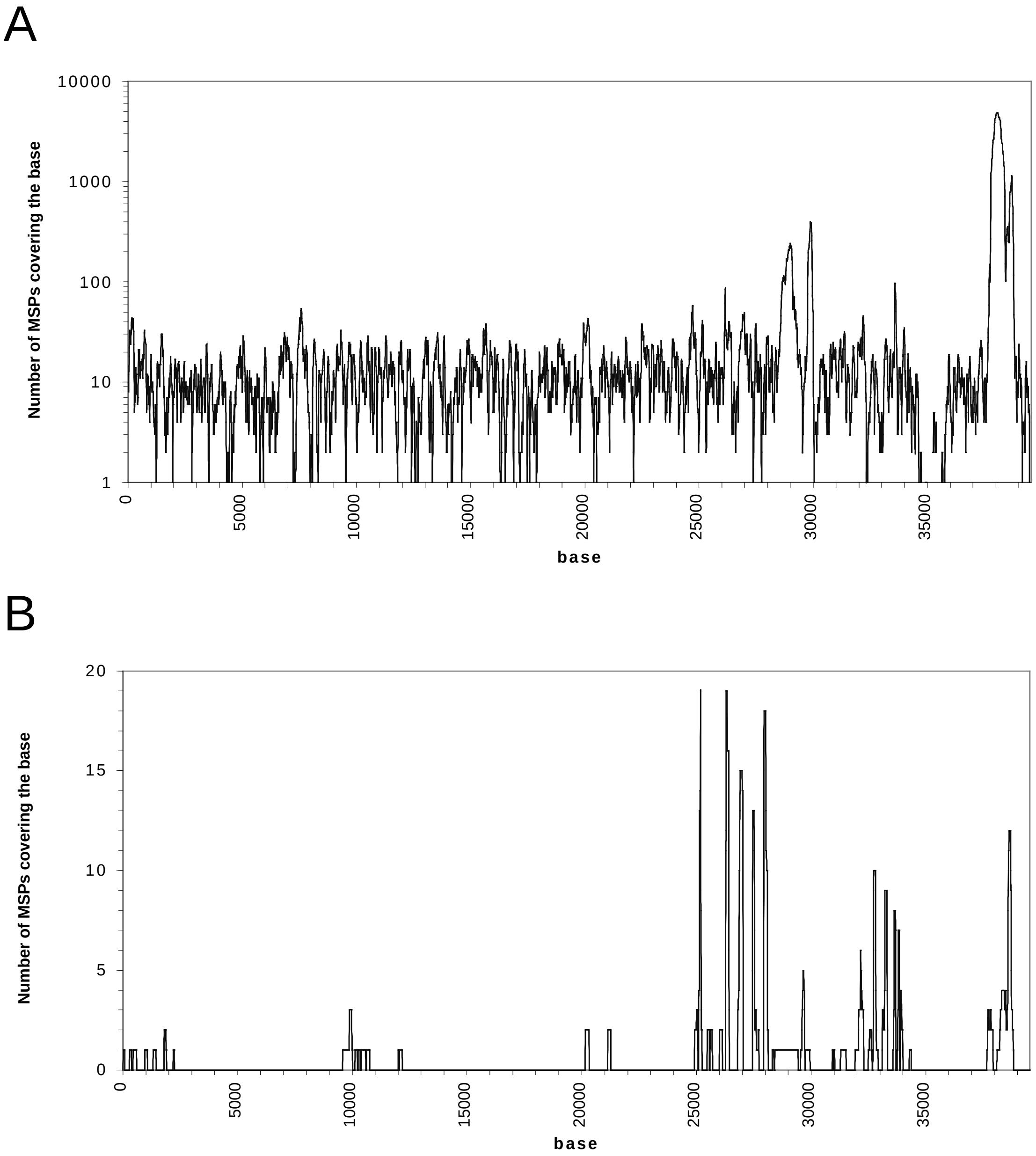


Figure 2. Histograms of MSP coverage on both strands of the 40 kbase DNA query sequence from the *C. elegans* cosmid ZK643 (A) before and (B) after MSPcrunch. MSPs were generated by Blastx as described in Methods. The number of MSPs was reduced from 119168 to 200 by MSPcrunch. The peak at 38000 is the 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 29000 and 30000 have a strong bias for charged residues. Most matches to these regions were removed by the biased composition detection mechanism. The main significant homology in this cosmid is to a G-protein coupled receptor (ZK643.3), located between bases 25000 and 28000. Most matches are insignificant alone, but satisfy the adjacency criteria in MSPcrunch with neighbouring matches. There is also a motif conserved with DCMP deaminases at 32700-32800. Most other accepted matches are to predicted proteins from this cosmid.

Adjacency tests for multiple MSPs

MSPs that have an unbiased composition and score above a certain threshold are clear indicators of homology. This threshold is normally considered to be approximately 80-90 score units, using the BLOSUM62 score matrix. MSPs with scores in a region below this threshold, i.e. in the twilight zone, may have such low scores due to fragmentation caused by gaps in one of the sequences relative to the other. 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.

Figure 3 illustrates two cases of pairs of MSPs; the MSPs in the first pair A1/A2 is clearly consistent with a normal gapped alignment. The MSPs of the second pair B1/B2 could be caused by a gap in a true alignment, but this is less likely due to the long overlap. The wider the gaps and the more the MSPs overlap, the less likely are they to combine into a true gapped alignment. If B2 would overlap B1 with more than the length of B1, no gapped alignment could possibly join them together and the MSPs should be treated separately.

The definition of adjacency completely depends on the chosen parameters for how big the gaps and overlaps between MSP may be. BLAST itself has a consistent ordering check for multiple matches [Karlin and Altschul, 1993], which is used for calculating the combined probability. It is very conservative however, and only dismisses consistency if joining a pair of MSPs is impossible, and does not take the length of the gap into account. Still, this simple rule does reduce the noise level a fair amount.

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 (see figure 3):

Query_gap = MSP2_QueryStart - MSP1_QueryEnd -1
Subject_gap = MSP2_SubjectStart - MSP1_SubjectEnd -1

MSP_dist = minimum (Query_gap, Subject_gap)
MSP_shift = | Query_gap - Subject_gap |

For Blastx, the query coordinates are converted to amino acid coordinates. Introns are essentially Query_gaps. If the Query_gap is larger than the Subject_gap but smaller than the intron limit, MSP_intron, potential introns are accommodated by setting Query_gap equal to Subject_gap before calculating MSP_dist and MSP_shift.

Consistency checking algorithm. For a truly consistent pair of MSPs, the values of MSP_dist and MSP_shift are located in a band that in order to distinguish true from false adjacency must become more narrow for lower scoring MSPs. The consistency checking algorithm is performed as follows:

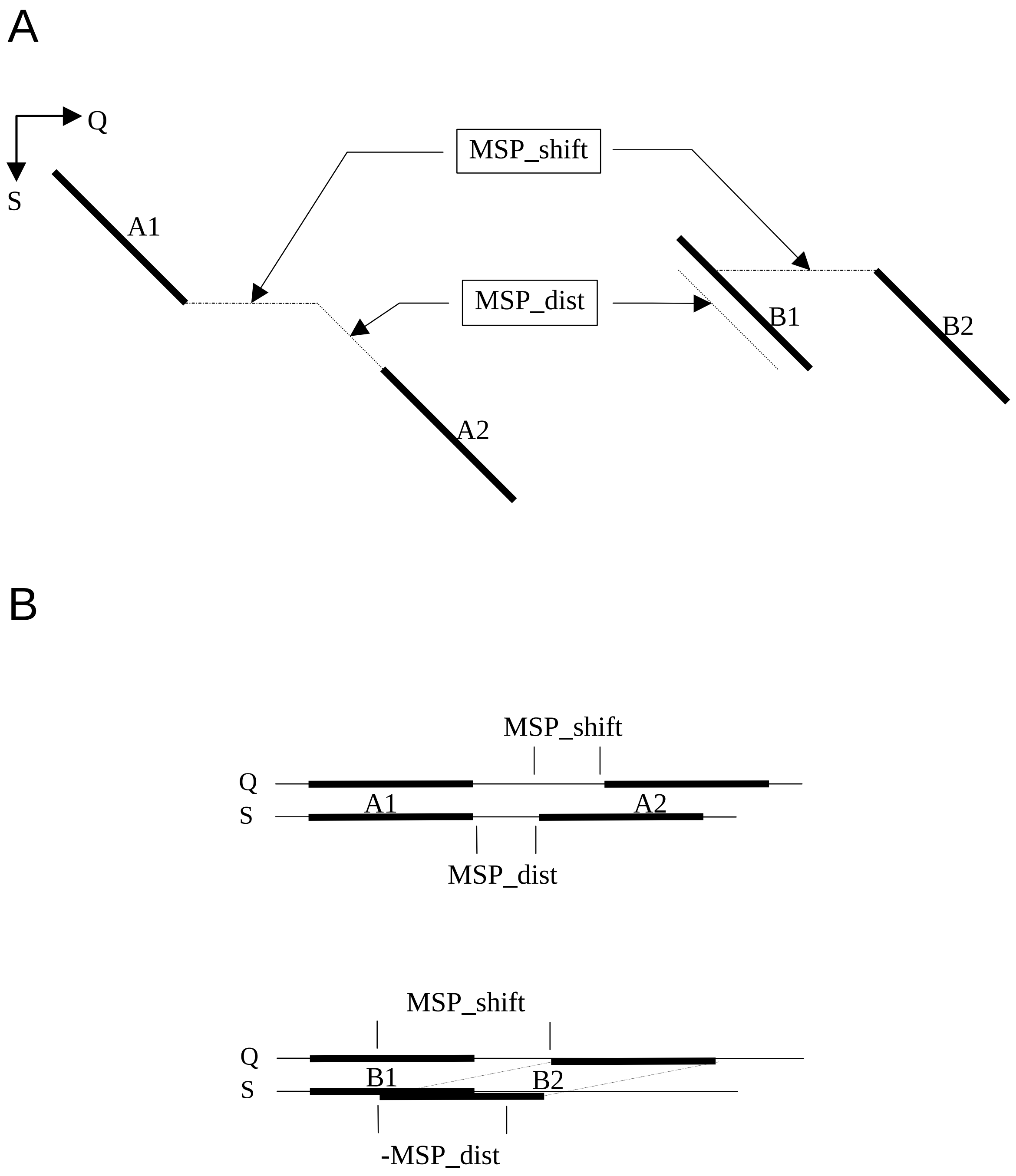


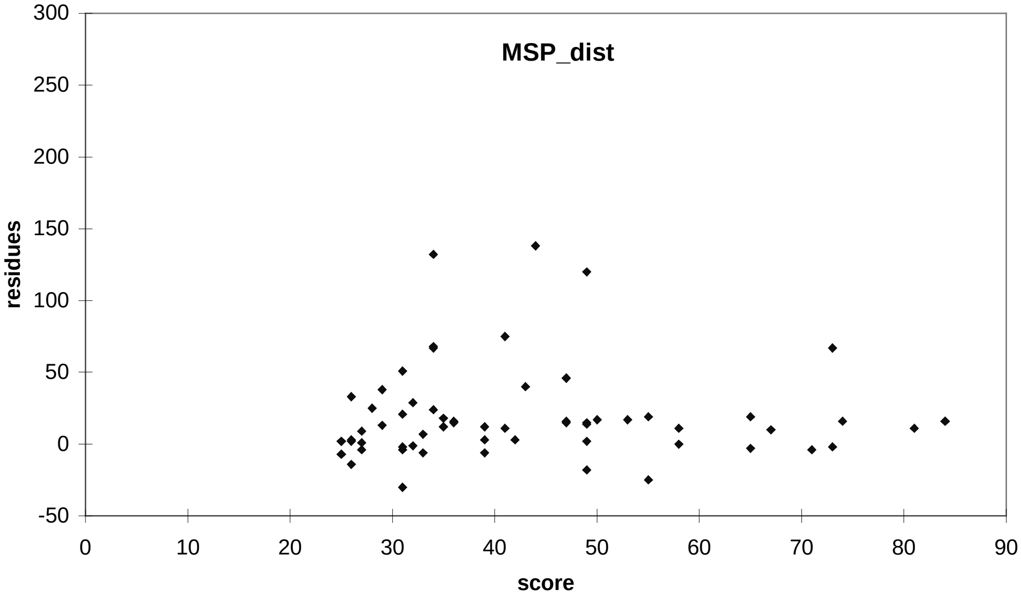
Figure 3. Diagrams of two cases of neighbouring MSPs in the sequences Q and S, for illustration of the MSPcrunch parameters MSP_dist and MSP_shift. The MSPs A1 and A2 do not overlap, while B1 and B2 do, yielding a negative MSP_dist value.

```
For all pairs of MSPs between two sequences {
    Calculate MSP_dist and MSP_shift for the pair.
    Calculate the acceptance boundaries for MSP_dist and MSP_shift based on the lowest scoring MSP of the pair.
    If MSP_dist and MSP_shift are within the limits, mark the two MSPs as adjacent to each other.
}
For all MSPs between two sequences {
    If it scores above the twilight zone upper limit, accept it.
    Otherwise, reject unless it was found to be adjacent to another MSP.
}
```

Parameter estimation. The basis for adjacency analysis is that random spurious matches may occasionally end up adjacent to each other purely by chance, whereas real matches will do so very frequently. To investigate the range of the adjacency parameters in real homologies, matches to a G-protein coupled receptor and a carboxyl esterase were analysed manually to verify if they were true or false, based on the overall dotplot as reference. The MSPcrunch adjacency parameters for 78 MSPs that were verified to be correct are shown in figure 4. Most of the values are clustered near zero, but some low-scoring true neighbours are also present. In the lower region, there is overlap between noise and signal, but not in the upper region. A similar plot of randomly generated spurious matches would have a flat distribution along y axes in both plots, but tend to be strongly concentrated in the lower region on the x axes.

Allowed adjacency bands. We are now faced with the problem of devising a set of rules that will include as many as possible of the true MSPs, while as few as possible of the false MSPs. Usually it is more desirable to include some spurious matches, since removing all of them may reduce sensitivity to true matches.

The simplest rule for confirming adjacency would be constant distance and shift cutoffs for matches in the twilight zone. This would not work well, however, since it would be too permissive for low scoring MSPs and too restrictive for high-scoring ones. To accommodate for this, a gradual tightening of the permissive distance and shift cutoffs is needed. This could be done either linearly, or according to some function. For MSP_shift and the lower bound of MSP_dist (maximum allowed overlap), we found that the rule needs to be quite strict, even for MSPs in the upper twilight zone, so a linear curve was found adequate. For the upper bound of MSP_dist and MSP_intron, however, strictness is much more required in the lower twilight zone than in the upper. Attempts to use a linear allowance function were unsatisfactory because they were either too strict in the upper zone, or would go to zero too suddenly in the lower zone. A quadratic function proved to gave a significant improvement over a linear. We did not pursue a more complex curve form, as we would only expect marginal improvements from this.



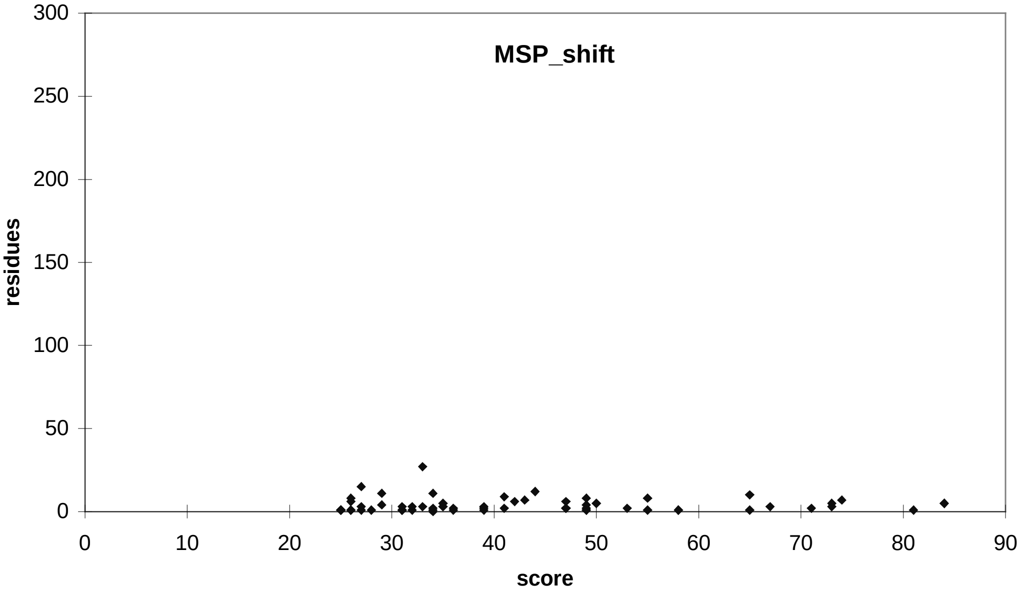


Figure 4 A sample of true MSP_dist and MSP_shift values from 78 MSPs that were manually verified. The score of the weakest MSP in each pair was used. Data from matches to the G-protein coupled receptor ZK643.3 (Swissprot YOW3_CAEEL P30650) and the carboxylesterase K07C11.4 (Wormpep CE07347).

The bounds for the allowed bands that performed best were produced with these parameters. If the twilight zone is defined between scores *lower* and *upper*, we define the bounds for a parameter at these endpoints as distmax.lower, distmax.upper, distmin.lower, etc.. The shape of a curve is controlled by the .power exponents. A power of 1 gives a linear curve, while a higher power gives a curve which is relatively more stringent in the lower region. For scores in the twilight zone, this gives the following bounds:

MSP_dist < distmax.lower + (score - lower)^distmax.power/distmax.scale

MSP_dist > distmin.lower - (score - lower)^distmin.power/distmin.scale

MSP_shift < shiftmax.lower + (score - lower)^shiftmax.power/shiftmax.scale

MSP_intron < intronmax.lower + (score - lower)^intronmax.power/intronmax.scale

where

distmax.scale = (upper-lower)^distmax.power/(distmax.upper - distmax.lower)

distmin.scale = (upper-lower)^distmin.power/(distmin.lower - distmin.upper)

shiftmax.scale = (upper-lower)^shiftmax.power/(shiftmax,upper - shiftmax.lower)

intronmax.scale=(upper-lower)^intronmax.power/(intronmax.upper-intronmax.lower)

(MSP_shift and MSP_intron are always positive so the lower bound is zero).

For Blastp (protein-protein comparison) we found the best definition of the twilight zone to be 25-75 (12.5-37.5 bits), while for Blastx, Tblastx and Tblastn (DNA-protein comparison) 35-75 (17.5-37.5 bits) due to the higher background noise levels. A default score of 75 was chosen as the upper limit of the twilight zone since only few spurious matches score above this value. For Blastn (DNA-DNA comparison) we set the zone by default to 90-140 (25 - 39 bits). We found that a linear behaviour in the distmin parameter was adequate, while a squared function for the other parameters gave better performance. The allowance bands that these functions yield are plotted graphically in figure 5.

To illustrate how these rules work in practice, an example is shown in figure 6. BLAST reports some false MSP together with the true MSPs. The false MSPs are not the lowest scoring ones, but since they lack adjacency, they can be ruled out as spurious twilight zone matches. The rule is stringent enough to allow confident use of lower scores than BLAST normally does. By default, BLAST sets the cutoff so that we expect 10 spurious matches to be reported, on a purely statistical basis. For this particular query sequence and database (ZK643.3 and swir10), this gives a lowest accepted MSP score (S2) of 33. As shown in figure 6c and d, lowering the BLAST score cutoff to 25 results in more true matches, but also more noise. However, thanks to the adjacency rules in MSPcrunch, the MSPs that are low-scoring due to gaps in the alignment can be separated from the spurious ones. Of course, all true MSPs are not always found. For instance, of the 78 manually verified MSPs in figure 4, 9 could not meet the adjacency criteria and were thus incorrectly missed by MSPcrunch.

Our results are generally applicable to any scoring scheme, but since the most popular scoring schemes, BLOSUM [Henikoff and Henikoff, 1992] and PAM [Dayhoff *et al.,* 1978], are in half bit units, we have chosen to express all scores in this unit. To convert them to other scoring schemes, they have to rescaled appropriately. Since the parameters have been estimated empirically to best suit the needs of interactive genome analysis, they may require adjustment for other purposes. All the twilight zone parameters can be changed on the command line. The above described algorithm was implemented in MSPcrunch version 2.0. Simpler and less effective algorithms have previously been described [Sonnhammer and Durbin, 1994a; Sonnhammer and Durbin, 1994b].

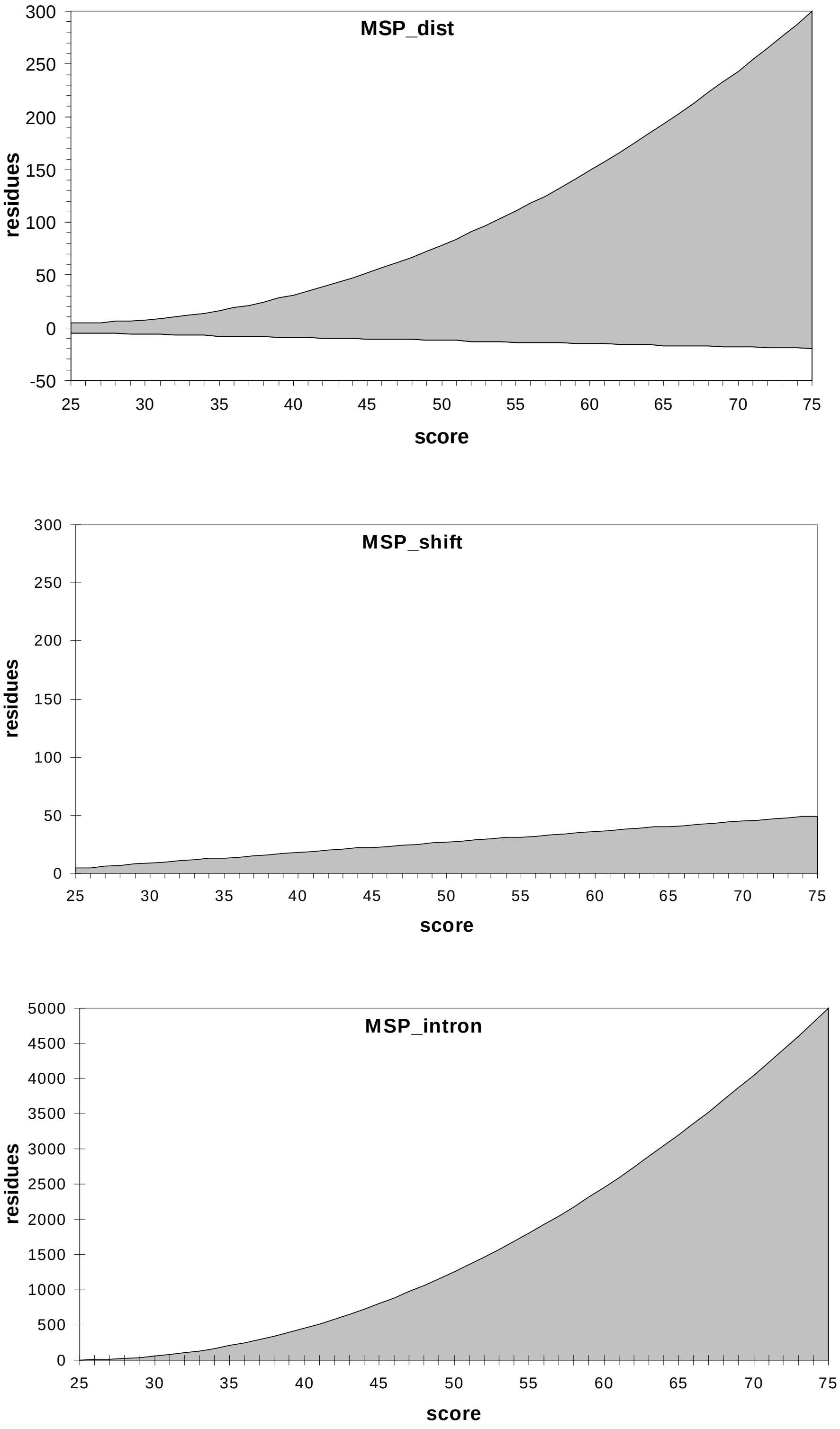


Figure 5. Bands of allowed MSPcrunch twilight zone adjacency parameters. If the parameters MSP_dist, MSP_shift and MSP_intron (Blastx only) between two MSPs fall within the shaded area, they are considered adjacent and will be accepted by MSPcrunch. The lower score of two MSPs is used.

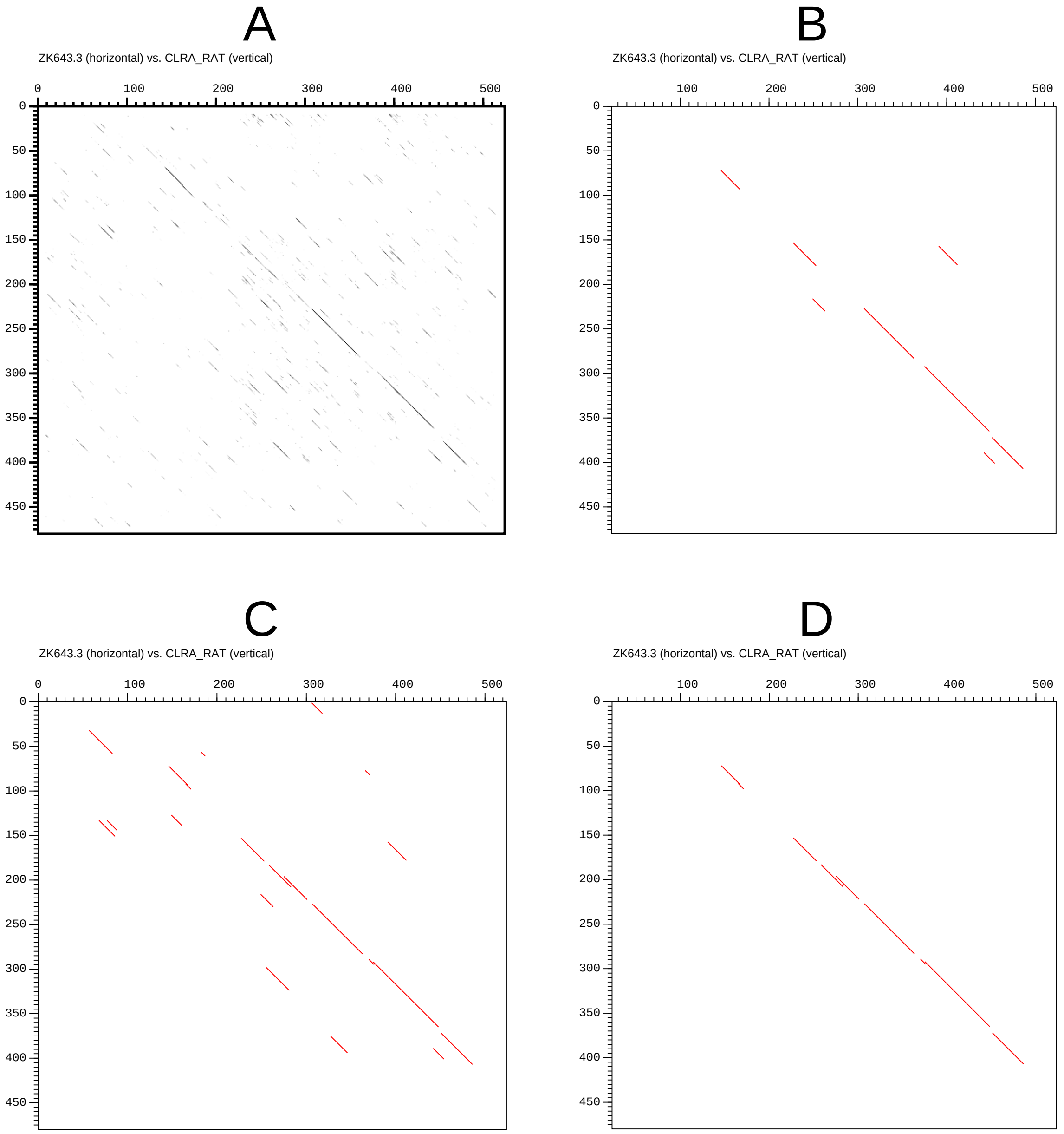


Figure 6. Dotplots illustrating the effect of MSPcrunch on the comparison of ZK643.3 (Swissprot YOW3_CAEEL P30650) with CLRA_RAT (Swissprot P32213). A. The full dotplot generated by Dotter (chapter 5) with a window size of 17. B. MSPs generated by Blastp using default parameters in a search against swir10 (S2=33). 5 true and 3 false MSPs are reported. C. MSPs generated by Blastp using an S2 cutoff of 25. 9 true and 11 false MSPs are reported. D. MSPs from C kept by MSPcrunch. All the false MSPs were rejected and all the true MSPs were kept, thus effectively enhancing both sensitivity and selectivity.

Displaying results

The recommended way to view MSPcrunch results is in Blixem. Nevertheless, MSPcrunch also supports a number of ASCII text output formats, that are useful for quick inspection of the results, and for exporting the data to other programs. Currently, the following output formats are available:

- A graphical ‘‘Big Picture” schematic of the relevant matches, with one database sequence per line as shown in figure 7. This way one rapidly gets a good picture of which proteins match a certain region of the query. It is not unlike the Big Picture display in Blixem, except that matches that are considered adjacent are combined onto one line, and the sum of their scores is given as the score. The number of adjacent segments is also shown. Non-adjacent MSPs of the same sequences are displayed on separate lines. If an MSP with a positive expected score passes the biased composition filter, its score will be marked by an asterisk.

- Gapped pairwise alignments, as shown in figure 8a. This is achieved by simply concatenating adjacent MSPs. Only if BLAST reports overlapping MSPs will gaps appear as dashes in one sequence and residues in the other. For non-overlapping MSPs, BLAST does not provide the residues of the one sequence that spans the gap, and no attempt is made to retrieve it separately, since little information would be added by this. Such gaps will therefore be represented by dashes in both sequences. In practice, short gaps of a few residues can usually be reconstructed from the overlap, while long gaps can not. For a more concise report, any gap longer than ten will not be shown at full length, but will be truncated to ten dashes. The layout has been designed to be easy to read by humans as well as easy to parse by other programs.

- A detailed listing of each MSP, as shown figure 8b. Instead of sorting the MSPs in score order, which BLAST does, 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.

- In tabular format, with one line per MSP, for parsing by other programs. A variety of one-line formats are supported, one of which is shown in figure 9a. This format, called ‘exblx’ can be parsed directly by Blixem (chapter 3), which will fetch all the matching sequences, using Efetch (chapter 6). Another format, ‘seqbl’, contains all the information Blixem needs, including sequence data (figure 9b), and thus eliminates the sometimes time-consuming sequence fetching.

- In .ace format, for export to ACEDB, as shown in figure 9c. This is particularly useful for homology assisted gene prediction, since ACEDB includes an interactive gene prediction workbench coupled to Blixem, which integrates the display of predicted exons into the BLAST-based multiple sequence alignment (see chapter 3).

```
QUERY= YMH5_CAEEL P34472 HYPOTHETICAL 136.3 KD PROTEIN F58A4.5 IN CHROMOSOME III.

                  |==============================================|1222
F40F12.2    1643 1                           _______________     CE00617 REVERSE TRANSCRIPTASE
ZK1236.4     423 3                             ___ _ ____        CE00531 TRANSPOSON T1-2
B34751       453 5                          _______ _ _ _        B34751  MOSQUITO TRANSPOSON
PC1123       320 4                         ___ ____              PC1123  BLOODFLUKE PLANORB
PC1231       329 5                          _ __  _  _           PC1231  MOSQUITO TRANSPOSON
H44490       260 3                          _________            H44490  REVERSE TRANSCRIPTASE
S31175       309 4                          __ __ _              S31175  TRANSPOSON NLR1CTH
YTX2_XENLA   137 1                          _____                P14381  TRANSPOSON TX1
C06E8.4      220 3                          _____ _ _            CE00800 RNA-DIRECTED DNA POL
RTJK_DROME   343 6                           _ _ __   _ _        P21328  RNA-DIRECTED DNA POL
S20106       168 2                           __ __               S20106  HYPOTHETICAL PROTEIN
MANR_HUMAN    75 1                     __                        P22897  MANNOSE RECEPTOR
MANR_HUMAN    79 1                      __                       P22897  MANNOSE RECEPTOR
B26330       229 4                          _ _ ___              B26330  TRANSPOSON I FACTOR
A32713       358 7                          ____  _ _ ___        A32713  REVERSE TRANSCRIPTASE
POL2_MOUSE   210 3                           __ _ _              P11369  REVERSE TRANSCRIPTASE
S16783       233 4                          _ _ _ _              S16783  RETROPOSON L1 - RAT
B34087       274 5                          _ _ _ _ _            B34087  HYPOTHETICAL PROTEIN
A44490       147 2                           _    _              A44490  REVERSE TRANSCRIPTASE
S28721       304 5                          _ _ ___ _            S28721  HYPOTHETICAL PROTEIN
JU0033       226 4                           _ __ _ _            JU0033  HYPOTHETICAL L1 PROT
S27771       263 5                          __    _ _ _          S27771  RNA-DIRECTED DNA POL
Y2R2_DROME   202 3                           _  _ _              P16425  RETROTRANSPOSABLE ELEM
B27672       214 4                           _ _  _ _            B27672  RNA-DIRECTED DNA POLY
POLR_DROME   183 3                          __ __ _              P16423  POL POLYPROTEIN
LIN1_NYCCO   199 3                           __ _ _              P08548  REVERSE TRANSCRIPTASE
C07A9.1      114 2   _ _                                         CE00502
B36186       208 4                          _ _ ___              B36186  TRANSPOSON
E44255        75 1                     __                        E44255  MANNOSE RECEPTOR
G44255        77 1                      __                       G44255  MANNOSE RECEPTOR
TETN_CARSP    77 1 ___                                           P26258  TETRANECTIN-LIKE
TETN_HUMAN    76 1 ___                                           P05452  TETRANECTIN PRECURSOR
S23650       160 3                             __ _ _            S23650  HYPOTHETICAL PROTEIN
LECE_ANTCR    83 2                     _ _                       P06027  ECHINOIDIN.
IXA_TRIFL     85 2    _ _                                        P23806  FACTOR IX/X-BINDING
LECI_HUMAN    99 2                                          _ _  P07307  HEPATIC LECTIN H2
LECI_MOUSE    88 2                                          _ _  P24721  HEPATIC LECTIN 2
A42230        88 2                                          _ _  A42230  LECTIN M-ASGP-BP
LECH_RAT      96 2                                          _ _  P02706  HEPATIC LECTIN 1
ODP1_ECOLI    99 2                     __ _                      P06958  PYRUVATE DEHYDROGENASE
ANP_OSMMO     90 2                     _ _                       Q01758  ANTIFREEZE PROTEIN
JH0626        90 2                     _ _                       JH0626  ANTIFREEZE PROTEIN II
VP3_ROTS1     92 2                                        _  _   P15736  INNER CORE PROTEIN VP3
LECI_RAT      82 2                                          _ _  P08290  HEPATIC LECTIN
```

Figure 7. Example of the Big Picture display of MSPcrunched Blastp results. The sequence YMH5_CAEEL (Swissprot P34472) was searched against swir5. The domain organisation of this protein is C-type lectin (30-160), an acid-rich stretch (160-540), C-type lectin (540-620), Reverse Transcriptase (650-980) and C-type lectin (1080-1150). All matches to the acid-rich stretch were removed by the biased composition rule (β =0.8). In the original output from Blastp, the 62 highest-scoring MSPs were all biased composition matches, apart from the close relatives F40F12.2 and ZK1236.4 from the same chromosome. The columns are: Entry name, combined score, nr. of MSPs, schematic alignment, accession nr. and abbreviated description. Sequences from Wormpep include a dot and the ones from Swissprot an underscore. Other sequences are from PIR.

A

```
QUERY = ZK643.3  Length = 522
=================================

> CLRA_RAT  P32213 CALCITONIN RECEPTOR A PRECURSOR (CT-R-A) (C1A).
------------------------------------------------------------------

Score= 96 (Sum of 2 contiguous HSPs), Identity= 48%

Query: ZK643.3 146 - 171 CPPTWDGWNCFDSATPGVVFKQ-CPNY
                         C  TWDGW C+D    GV+  Q CP+Y
Sbjct: CLRA_RAT 72 - 98  CNRTWDGWMCWDDTPAGVMSYQHCPDY


Score= 61 (Sum of 2 contiguous HSPs), Identity= 25%

Query: ZK643.3 227 - 283  LLTYSASVIFLIPAVFLLTLLRPIRCQ----LHRHLLISCLLYGAFYLITVSLFVVN
                          L+ +S S+  LI ++ +    + + CQ    LH+++ ++ +L     +I +   V N
Sbjct: CLRA_RAT 153 - 208 LVGHSMSIAALIASMGIFLFFKNLSCQ----LHKNMFLTYILNSIIIIIHLVEVVPN


Score= 322 (Sum of 5 contiguous HSPs), Identity= 33%

Query: ZK643.3 275 - 486  ITVSLFVVNDAPLSSQVFQNHLFCRLL-----RYLRLTNFTWMLAEAVYLWRLLHTAQHS
                          I + + +V   P    V ++ + C++L     +Y+   N+ WML E +YL  L+  A  +
Sbjct: CLRA_RAT 196 - 407 IIIIHLVEVVPNGDLVRRDPISCKIL-----QYMMACNYFWMLCEGIYLHTLIVMAVFT

                          EGETLRSYKVICWGVPGVITVVYIFVRSL------CWIENSTVAWIEWMIITPSLLAMGV
                          E + LR Y ++ WG P V T+++   R++      CW+   T   + ++I  P + A+ V
                          EDQRLRWYYLLGWGFPIVPTIIHAITRAV------CWLSTET--HLLYIIHGPVMAALVV

                          NLLLLGLIVYILVKKLRCDPHLERIQYRKAVRGALMLIPVFGVQQLLTIYRFSN------
                          N   L  IV +LV K+R     E   Y KAV+  ++L+P+ G+Q ++  +R SN
                          NFFFLLNIVRVLVTKMRQTHEAEAYMYLKAVKATMVLVPLLGIQFVVFPWRPSN------

                          YQVTDQSLNGLQGMFVSFIVCYTNRSVVECVLKFWS
                          Y     SL   QG FV+ I C+ N  V   + + W+
                          YDYLMHSLIHFQGFFVATIYCFCNHEVQVTLKRQWA
```

Figure 8. MSPcrunch output of pairwise alignments. A (this page). Gapped alignments of the accepted MSPs in figure 6d. Note that only contigs of adjacent MSPs are aligned with gaps, separate contigs are not. The start and end coordinates of the entire contig is given at the start of each alignment, to make parsing easy. B (next page). Each MSP reported separately with MSP-specific information in N to C-terminal order. Matrix_expected and bias-ratio are referred to in the text as M_{exp} and β .

B

```
QUERY = ZK643.3  Length = 522
=================================

> CLRA_RAT  P32213 CALCITONIN RECEPTOR A PRECURSOR (CT-R-A) (C1A).
-----------------------------------------------------------------

Score= 68, Identity= 50%, Matrix_Expected= -20.8, bias-ratio= 1.04, Adjacency= Right

Query: ZK643.3 146 - 167 CPPTWDGWNCFDSATPGVVFKQ
                         C  TWDGW C+D    GV+  Q
Sbjct: CLRA_RAT 72 - 93  CNRTWDGWMCWDDTPAGVMSYQ


Score= 28, Identity= 43%, Matrix_Expected= -6.6, bias-ratio= 1.00, Adjacency= Left

Query: ZK643.3 165 - 171 FKQCPNY
                         ++ CP+Y
Sbjct: CLRA_RAT 92 - 98  YQHCPDY


Score= 34, Identity= 26%, Matrix_Expected= -25.5, bias-ratio= 0.89, Adjacency= Right

Query: ZK643.3 227 - 253  LLTYSASVIFLIPAVFLLTLLRPIRCQ
                          L+ +S S+  LI ++ +    + + CQ
Sbjct: CLRA_RAT 153 - 179 LVGHSMSIAALIASMGIFLFFKNLSCQ


Score= 27, Identity= 23%, Matrix_Expected= -24.6, bias-ratio= 0.83, Adjacency= Left

Query: ZK643.3 258 - 283  LHRHLLISCLLYGAFYLITVSLFVVN
                          LH+++ ++ +L     +I +   V N
Sbjct: CLRA_RAT 183 - 208 LHKNMFLTYILNSIIIIIHLVEVVPN


Score= 27, Identity= 22%, Matrix_Expected= -25.5, bias-ratio= 0.97, Adjacency= Right

Query: ZK643.3 275 - 301  ITVSLFVVNDAPLSSQVFQNHLFCRLL
                          I + + +V   P    V ++ + C++L
Sbjct: CLRA_RAT 196 - 222 IIIIIHLVEVVPNGDLVRRDPISCKIL


Score= 109, Identity= 35%, Matrix_Expected= -53.9, bias-ratio= 0.98, Adjacency= LeftRight

Query: ZK643.3 307 - 363  RYLRLTNFTWMLAEAVYLWRLLHTAQHSEGETLRSYKVICWGVPGVITVVYIFVRSL
                          +Y+   N+ WML E +YL  L+  A  +E + LR Y ++ WG P V T+++   R++
Sbjct: CLRA_RAT 227 - 283 QYMMACNYFWMLCEGIYLHTLIVMAVFTEDQRLRWYYLLGWGFPIVPTIIHAITRAV


Score= 27, Identity= 43%, Matrix_Expected= -6.6, bias-ratio= 0.98, Adjacency= Right

Query: ZK643.3 370 - 376  CWIENST
                          CW+   T
Sbjct: CLRA_RAT 289 - 295 CWLSTET


Score= 105, Identity= 35%, Matrix_Expected= -69.9, bias-ratio= 0.93, Adjacency= LeftRight

Query: ZK643.3 375 - 448  STVAWIEWMIITPSLLAMGVNLLLLGLIVYILVKKLRCDPHLERIQYRKAVRGALMLIPVFGVQQLLTIYRFSN
                          ST   + ++I  P + A+ VN    L  IV +LV K+R      E   Y KAV+  ++L+P+ G+Q ++  +R SN
Sbjct: CLRA_RAT 292 - 365 STETHLLYIIHGPVMAALVVNFFFLLNIVRVLVTKMRQTHEAEAYMYLKAVKATMVLVPLLGIQFVVFPWRPSN


Score= 54, Identity= 33%, Matrix_Expected= -34.0, bias-ratio= 0.99, Adjacency= Left

Query: ZK643.3 451 - 486  YQVTDQSLNGLQGMFVSFIVCYTNRSVVECVLKFWS
                          Y     SL   QG FV+ I C+ N  V   + + W+
Sbjct: CLRA_RAT 372 - 407 YDYLMHSLIHFQGFFVATIYCFCNHEVQVTLKRQWA
```

Figure 8b.

A

```
  68 (+1)     146     167    72    93 CLRA_RAT P32213 CALCITONIN RECEPTOR
  28 (+1)     165     171    92    98 CLRA_RAT P32213 CALCITONIN RECEPTOR
  34 (+1)     227     253   153   179 CLRA_RAT P32213 CALCITONIN RECEPTOR
  27 (+1)     258     283   183   208 CLRA_RAT P32213 CALCITONIN RECEPTOR
  27 (+1)     275     301   196   222 CLRA_RAT P32213 CALCITONIN RECEPTOR
```

B

```
# seqbl
# BLASTP
  68 (+1)     146     167    72    93 CLRA_RAT CNRTWDGWMCWDDTPAGVMSYQ
  28 (+1)     165     171    92    98 CLRA_RAT YQHCPDY
  34 (+1)     227     253   153   179 CLRA_RAT LVGHSMSIAALIASMGIFLFFKNLSCQ
  27 (+1)     258     283   183   208 CLRA_RAT LHKNMFLTYILNSIIIIIHLVEVVPN
  27 (+1)     275     301   196   222 CLRA_RAT IIIIIHLVEVVPNGDLVRRDPISCKIL
```

C

```
Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 68 146 167 72 93

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 68 72 93 146 167

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 28 165 171 92 98

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 28 92 98 165 171

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 34 227 253 153 179

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 34 153 179 227 253

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 27 258 283 183 208

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 27 183 208 258 283

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 27 275 301 196 222

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 27 196 222 275 301
```

Figure 9. Examples of tabular output from MSPcrunch. A. The ‘exblx’ format, which contains score, frame, start and end coordinates and subject name and description of each MSP on one line. B. The ‘seqbl’ format, which is the same as exblx, except that it contains the sequence of the database entry instead of its description. Both formats are parsed by Blixem, but for exblx data, Efetch (chapter 6) must be installed to retrieve the sequences. C. The same data in .ace format, which is used to export the MSPs to ACEDB.

Discussion

An often heard criticism of using ungapped alignments is that distantly related proteins as a rule can only be aligned by inserting gaps. However, the regions which require gaps usually correspond to loops between secondary structure elements in the 3-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 results of algorithms that produce gapped alignment depend strongly on a somewhat arbitrary gap penalty. A further advantage of ungapped alignments is that repeated and shuffled domains in one sequence can be detected, something which is often compromised by programs that produce a gapped alignment.

One drawback of ungapped alignments is the difficulty of calculating an appropriate composite score for all MSPs with the same protein. Here we put the emphasis on making sure that a series of MSPs are truly consistent with a single gapped alignment. We then simply sum up the individual scores. The BLAST programs also calculate the probability of multiple matches by summing the individual scores of consistently ordered MSPs and correcting for the number of MSPs. However, their consistency criterion [Karlin and Altschul, 1993] is much weaker than our adjacency criteria and falsely high significance may arise from spurious hits that are not truly adjacent, especially those involving biased composition matches.

An additional practical problem with the probabilities calculated in BLAST is that they 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 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 (which are log odds ratios).

The parameters used to deem two MSPs adjacent or not, MSP_dist and MSP_shift, were here used independently of each other. It might be worth considering treating them in a combined way, so that a large MSP_dist is more readily accepted if MSP_shift is small. We have not found any such combinatorial rule that works satisfactorily in practice, and that make sense both biologically and statistically. Biologically one would expect a larger MSP_shift for a larger MSP_dist, but allowing this would increase the levels of accepted noise. There does not seem to exist a strong correlation between the parameters, and since it is also important for rules to be simple enough to understand, we have not pursued this any further.

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. This is demonstrated in Part II, using the Pfam collection of protein families based on hidden Markov models. Whether searching a collection of aligned families always is more sensitive than pairwise comparison to all sequences is however not entirely clear. Sensitivity may also decrease if the family is not well defined, or if the query is much closer to one of the members than to the average of the family. Therefore, we have here pursued a higher quality of traditional single-sequence database searching, which most likely will remain an important tool complementary to family-based searching techniques.

The system described here is similar to GeneQuiz [Scharf *et al.*, 1994] in that it performs many sequence analysis tasks automatically. MSPcrunch however is primarily concerned with the proper treatment of similarities along large DNA queries, for which a solution of the multi-domain problem is needed. Together with Blixem and Dotter, integrated in ACEDB, MSPcrunch is part of a sequence analysis workbench. This workbench also has different goals than GeneQuiz, since it was designed to analyse the DNA sequence and improve the quality of exon/intron predictions using sequence similarity, as well as being an annotation tool. Given the complexity of the gene prediction process in higher eukaryotic genomes, we don't envisage a fully automatic system for this in the near future.

Many of the features in MSPcrunch have been made partly obsolete by subsequent improvements in the BLAST software. Especially the sensitivity for multiple weak matches was improved significantly in BLAST 1.4. Our biased composition check is currently being tested as an option in BLAST, and we hope that other features will be included in the future too. Performing the filtering process during the search phase would reduce the computational load.

MSPcrunch, Seqsplit, Blastunsplit are available by anonymous FTP from ncbi.nlm.nih.gov in the directory /pub/MSPcrunch+Blixem.