BIOINFORMATICS

Vol. 17 no. 0 2001
Pages 1–8

Evaluation of protein multiple alignments by SAM-T99 using the BAliBASE multiple alignment test set

Kevin Karplus and Birong Hu

[1]Author: Please provide address., , , , ,

Received on February 2, 2001; revised on April 11, 2001; accepted on May 30, 2001

ABSTRACT

Motivation: SAM-T99 is an iterative hidden Markov model-based method for finding proteins similar to a single target sequence and aligning them. One of its main uses is to produce multiple alignments of homologs of the target sequence. Previous tests of SAM-T99 and it's predecessors have concentrated on the quality of the searches performed, not on the quality of the multiple alignment. In this paper we report on tests of multiple alignment quality, comparing SAM-T99 to the standard multiple aligner, CLUSTALW.

Results: The paper evaluates the multiple-alignment aspect of the SAM-T99 protocol, using the BAliBASE benchmark alignment database. On these benchmarks, SAM-T99 is comparable in accuracy with ClustalW.

Availability: The SAM-T99 protocol can be run on the web at http://www.cse.ucsc.edu/research/compbio/HMM-apps/T99-query.html and the alignment tune-up option described here can be run at http://www.cse.ucsc.edu/research/compbio/HMM-apps/T99-tuneup.html. The protocol is also part of the standard SAM suite of tools. http://www.cse.ucsc.edu/research/compbio/sam/

1 INTRODUCTION

Multiple protein sequence alignment has been widely used in finding conserved regions in protein families and in predicting protein structures (Dayhoff *et al.*, 1978; Karplus *et al.*, 1999; Hannenhalli and Russel, 2000; Holm and Sander, 1999; Pascarella *et al.*, 1998; Levin *et al.*, 1993).

The quality of the predictions depends critically on the quality of the multiple alignments and the diversity of the sequences aligned, but few of the current multiple aligners are capable of aligning hundreds or thousands of homologous sequences.

The SAM-T99 search protocol finds and aligns protein sequences and can easily generate huge multiple alignments. We needed to know whether the multiple alignments it produces are good ones, or whether more computationally intensive techniques are needed to create multiple alignments of adequate quality. In this paper we evaluate SAM-T99 as a multiple aligner (ignoring its main purpose as a search tool), comparing it to the well-known ClustalW program (Thompson *et al.*, 1994).

Many programs have been developed for multiple protein sequence alignment, and they fall into two classes: progressive and iterative. The classic progressive approach is to build up the alignment gradually by aligning the closest sequences first and then successively adding in more distant ones. Examples include ClustalW (Thompson *et al.*, 1994; Higgins *et al.*, 1996; Jeanmougin *et al.*, 1998), PILEUP (Group, 1991), and PIMA (Smith and Smith, 1992). Another choice is to use an iterative strategy to refine and improve an initial multiple alignment. Programs in the category include PRRP (Gotoh, 1999), DIALIGN (Morgenstern *et al.*, 1998), and SAGA (Notredame and Higgins, 1996).

The SAM-T99 method is an iterative hidden Markov model-based search technique, which aligns sequences to a hidden Markov model (HMM) and improves the alignment by retraining the HMM on the sequences. It is close in spirit to the iterative multiple aligners, though rather different in internal implementation.

SAM-T99 is a protocol for using the SAM collection of HMM tools (Hughey and Krogh, 1996; Hughey *et al.*, 1999). The SAM-T99 protocol is included in version 3 of SAM and is an improved version of SAM-T98, which was used for protein structure prediction in CASP3 (Karplus *et al.*, 1999). SAM-T98 has been shown to be more effective in finding remote homologs than competing sequence-based methods (Park *et al.*, 1998a; Karplus *et al.*, 1998).

One problem for evaluating various alignment programs has been the lack of a standard benchmark for comparison. Different programs do well on different examples, so author-selected examples often distort the performance, making comparison difficult. Recently, a benchmark alignment database (BAliBASE) has been constructed to serve such a purpose (Thompson *et al.*, 1999a). Since BAliBASE was not deliberately designed in favor of any specific alignment program, it is relatively objective, and

has been used as a standard multiple alignment test set to evaluate many protein alignment programs (Thompson *et al.*, 1999b; Notredame *et al.*, 2000). Nevertheless, there are some biases in the test set. Most notably, it favors global alignment methods, because sequences to align are trimmed at the boundaries of the alignment. SAM-T99, however, uses local alignment, so the test set is somewhat biased against SAM-T99. Despite that, our results still show that SAM-T99 is not significantly different in accuracy from ClustalW.

2 BAliBASE TEST SET

2.1 BAliBASE alignments

BAliBASE (Thompson *et al.*, 1999b) is a database of multiple protein sequence alignments. It contains reasonably high-quality, well-documented alignments that have been confirmed using a variety of programs and by manual verification. Because the alignments are given in a format that requires all positions to be aligned, even for regions that have no sensible correspondence, BAliBASE annotates the alignments with the *core blocks* that include only the regions that are believed to be reliably aligned.

The current version of BAliBASE contains 143 reference alignments, with a total of more than 1000 sequences. The number of distinct sequences is much smaller, as many of the reference alignments include the same sequences. Thus the database only covers a very small portion of protein space, and may have biases due to the repeated use of the same protein families.

BAliBASE is divided into five different categories by the length and similarity of the sequences in the core blocks and by the presence of insertions and N/C-terminal extensions.

- Category 1 contains more than 80 alignments of sequences of similar length. Each alignment has a small number (3–7) of sequences.
- Category 2 consists of 23 alignments with at least 15 closely related sequences and one 'orphan' (<25% identical) sequence.
- Category 3 has more divergent sequence alignments that contain multiple subgroups with <25% residue identity between groups.
- Category 4 contains sequences with N/C-terminal extensions (up to 400 residues). This is the only subset that favors local over global alignment.
- Category 5 contains sequences with long internal insertions.

2.2 Assessing alignments with alignment scores

To determine the similarity of the alignment obtained by a program to the reference alignment in BAliBASE, we calculated two alignment scores: Sum-of-Pair Score (SPS) and Column Score (CS). Our score calculation program was modified from the bali_score program that came with the BAliBASE distribution.

The SPS counts how many pairs of residues are correctly aligned. For two sequences x and y, we score 2 points for each pair of residues x_i and y_j that are aligned with each other in both the tested alignment and a core block of the reference alignment. We also score 1 point for each residue that is aligned with a gap in both the tested alignment and a core block. The total score is normalized by the maximum possible score, so that the range of possible values is from 0 to 1, with 1 indicating a multiple alignment that is identical on the core blocks. The original bali_score program used a somewhat different way of handling gaps in core blocks, which resulted in incorrect normalizations, so that even perfect multiple alignments could have a score less than one.

More formally, in an alignment with N sequences of length of M, at every alignment position i, if sequence x and sequence y align the same way as in a core block of the reference alignment, then the pair value P_{ixy} is positive. If, in column i, both x and y have residues aligned in a core block of the reference alignment, then $P_{ixy} = 2$; if one of the sequences has a gap (in both alignments), then $P_{ixy} = 1$, otherwise $P_{ixy} = 0$. The score S_i for the ith column is

$$S_i = \sum_{j=1}^{N} \sum_{k \neq j} P_{ijk}.$$

Editor: As usual SPS and CS set as roman. Is it OK?

The SPS is then:

$$\mathrm{SPS} = \sum_{i=1}^{M} S_i \Big/ \sum_{i=1}^{M_r} S_{ri}.$$

Where M_r is the number of columns in the core blocks of the reference alignment and S_{ri} is the score S_i for the ith column in the reference alignment.

The CS counts the number of columns of the core blocks that are aligned correctly in all sequences, normalized by the number of alignment columns. More formally, at each position i of an alignment, $C_i = 1$ if all the sequences are aligned the same way as in a core block of the reference alignment, otherwise $C_i = 0$. Thus CS of the whole alignment is

$$\mathrm{CS} = \sum_{i=1}^{M_c} C_i / M_c$$

where M_c is the total number of columns in core blocks of the reference alignment. No column is ignored in our calculation (except for columns containing all gaps, which contain no information about the alignment of the sequences).

In essence, the higher the SPSs and CSs are, the more accurate are the alignments generated by the programs. It is noticeable however, that one badly misaligned sequence reduces SPS from 1 to $1-2/N$, but reduces CS from 1 to 0. Thus the CS measurement is more useful for alignments that are nearly perfect. Indeed we have observed that CS tends to be almost a binary value—with each alignment either being very good or scoring 0.

In our report, we calculated the alignment scores for the annotated core blocks only, because these are sometimes the only regions that are reliably aligned in BAliBASE.

2.3 Problems with BAliBASE

Although BAliBASE is a useful test set, we noticed three limitations:

- BAliBASE is biased towards global alignment programs, since the majority of the BAliBASE alignments have been trimmed down to core blocks only. We would like to see a test set having full length sequences, since such a test set more closely resembles real alignment problems.
- Some sequence sets are used more than once in different categories. For example, 1ajsA appears in Category 1–3. Although different categories emphasize different aspects of the alignments, the repetitive use of the same protein family may cause bias in assessing alignment programs. If a program does particularly well (or poorly) on one family, then reusing the family in the test set amplifies the effect.
- Some of the BAliBASE alignments are incorrect.

  One of them is kinase3 in Category 5. The conserved lysine in the first block of kgp2_drome was one amino acid off in the BAliBASE alignment. The creators of balibase have accepted this correction to the balibase data set.

  Another case is 1ped in Category 2. The BAliBASE alignment of 2ohxA is clearly wrong starting at the 5th core block. Both SAM-T99 and ClustalW aligned these blocks correctly (in agreement with the 1ped alignment in Category 3).

  We reported these incorrect alignments to the creators of the BAliBASE alignment, and they have corrected them in the newer release of the bechmark set.

  We did not use these corrected alignments in evaluating SAM-T99 and ClustalW, but did the analysis with the original unmodified test set. We felt it inappropriate to report results of modifications to either the benchmark or the method being tested made as a result of the benchmark tests. The scores would be slightly better for both SAM-T99 and ClustalW if the corrected alignments were used.

3 THE SAM-T99 PROTOCOL

The Sequence Alignment and Modeling system (SAM) is a collection of software tools for multiple protein sequence alignment and profiling using hidden Markov models (HMMs). SAM-T99 is an iterative search protocol released with version 3.0 of the SAM suite.

With the SAM-T99 protocol, a multiple alignment (or even a single seed sequence) can be used to build an HMM, which can then be used for searching for new members of the family. When new members are found, the HMM can be retrained to include them, new multiple alignments can be made, and the process iterated. This technique is the essence of the SAM-T98 protocol, which has proven more effective in finding remote homologs than competing sequence-based methods such as PSI-BLAST and ISS (Park *et al.*, 1998a; Karplus *et al.*, 1998; Altschul *et al.*, 1997; Park *et al.*, 1997). The SAM-T99 protocol that we now use is an evolutionary improvement over SAM-T98, with slightly better parameter settings, more efficient implementation, better estimates of statistical significance, and better performance (Hughey *et al.*, 1999). The construction, training, and application of the HMMs is all done with programs from the SAM package (Hughey and Krogh, 1996).

We will describe the standard SAM-T99 protocol first, then the -tuneup option used for the test in this paper.

3.1 The default SAM-T99 protocol

SAM-T99 starts with a query sequence (or seed alignment) and searches the non-redundant protein database (NR) using WU-BLASTP (Altschul *et al.*, 1990) to produce two sets of potential homologs: one of very similar sequences ($E < 0.0005$) and one of possibly similar sequences ($E < 300$). The initial WU-BLASTP cull of NR is used for two reasons: we do not expect an HMM built from a single sequence to do any better at finding close homologs than WU-BLASTP, and an HMM database search of all of NR is too slow for building large numbers of alignments.

The SAM-T99 method then uses four iterations of a selection, training, and alignment procedure. For each iteration it needs an initial alignment, a set of sequences to search, a threshold value, and a transition regularizer. From the alignment and regularizer, an HMM is constructed and used to score the set of sequences. All sequences that score better than the threshold value are used to estimate a new HMM. Alignment of the training sequences to that HMM produces the alignment that is the input for the next iteration.

The SAM-Txx methods use sequence weighting for building models from alignments, both internally and when the final alignments are used to create the models for scoring a set of sequences.

In SAM-T99, the relative weights are set with our own

weighting scheme which gives more weight to outliers, and the absolute weight is set to get a specific level of entropy averaged over all columns after a Dirichlet mixture regularizer (Sjölander *et al.*, 1996) is applied to the weighted counts. The desired entropy is specified as the number of bits saved relative to the entropy of the background distribution. This relative entropy measure has been used previously to characterize substitution matrices (Altschul, 1991), and the popular BLOSUM50 and BLOSUM62 matrices corresponds to saving about 0.5 and 0.7 bits per column. The SAM-T99 method uses 0.8 bits per column as the target, but preliminary fold-recognition tests indicate that this may be too high a value.

On the first iteration the single sequence passed to the method is used as the initial (trivial) alignment and the close homologs found by WU-BLASTP are used as the search set. The threshold is set strictly (E-value $<$ 0.000 01), so only strong matches to the sequence are considered. The transition regularizer allows gaps, but favors long matches and long gaps over frequent short gaps. Requiring both WU-BLASTP and the initial HMM to score a sequence well ensures that only very similar sequences are included at this stage of the process.

On subsequent iterations the input alignment is the output from the previous iteration and the search set is the larger set of possible homologs found by WU-BLASTP. The thresholds are gradually loosened (E-value $<$ 0.0001, $<$ 0.001, $<$ 0.01).

For the second and third iteration, we again use the regularizer that encourages long sequences of match states, and for the final iteration we use a transition regularizer trained on FSSP structural alignments.

The above selection, training, and alignment procedures consists of several calls to SAM programs. Models are created with SAMs modelfromalign program which uses the alignment, sequence weighting, transition regularizer, and Dirichlet mixture to build an HMM. Scoring the sequence set with an HMM uses SAMs multiple domain scoring procedure, now part of hmmscore, which selects only the portion of a sequence matching the HMM [local scoring (Smith and Waterman, 1981) as applied to SAM models (Tarnas and Hughey, 1998)]. From the sequences selected using this procedure, a new model is estimated using SAMs buildmodel HMM training program. The alignment of the training sequences back to the resulting HMM is accomplished with SAMs hmmscore program. To ensure that the initial sequence to the whole process is not lost, it is added to the training set at this point, and any duplicate sequences in the training set are eliminated.

3.2 The tuneup option of SAM-T99

SAM-T99 was originally designed to be used as a protein search program. As a search program, finding reliable homologs and removing extraneous sequences are desirable features. SAM-T99 can also be used for protein sequence alignment, without search, using the -tuneup option. Even with this option, only the sequences that achieve a certain similarity threshold will be aligned. Thus very diverged sequences will be dropped out of the final alignment. Since one of the applications of the -tuneup option is to improve the alignments created by other search tools, the ability to reject sequences as insufficiently similar was regarded as a feature, not a bug. (The newer SAM-T2K protocol, currently under development, will allow the user the option of forcing all the provided sequences into the final alignment.)

The -tuneup option to the SAM-T99 script changes the method in several important ways. First, no search of NR is done—only the provided sequences are used as the potential homolog set on each iteration. Second, the seed alignment is created by buildmodel from the set of unaligned sequences, and the seed is not forced into the output alignment on each iteration. Third, the weighting is set to save only 0.5 bits per position relative to the background. Fourth the number of iteration is changed to 3 and the significance thresholds are changed to (E-value $<$ 0.0001, $<$ 0.1, $<$ 10.0).

SAM-T99 generates a multiple alignment in SAMs standard a2m format. The alignment score calculation program, however, takes MSF format alignments as input. To use the program, the SAM-T99 results were converted into MSF format using the command

```
prettyalign foo.a2m -f | readseq -fMSF -p >foo.out
```

The prettyalign command adds informationless dots to the a2m format, so that readseq can read the alignment correctly.

4 RESULTS

4.1 SAM-T99 tuneup results

Multiple alignments created by SAM-T99 -tuneup were scored as described in Section 2.2 as were multiple alignments created by the ClustalW program, version 1.8 (Thompson *et al.*, 1994; Higgins *et al.*, 1996). The default parameters were used to generate alignments for all test data.

Using the default parameters, SAM-T99 -tuneup aligned all input sequences in 123 out of 143 cases. In the other 20 cases, the more diverged sequences were dropped, because the final significance thresholds were set fairly tight. This behavior is desirable in a search program looking for possible homologs, but not in an alignment program.

As expected, SAM-T99 dropped sequences most frequently in Category 2, which has orphan sequences that are very diverged from the main group (see Table 1). Within Category 1, three of the alignments with dropped sequences were short alignments with less

Table 1. The number of cases that SAM-T99 failed to align all sequences, rejecting some sequences as too dissimilar. Note that Category 2, which contains 'orphan' sequences, has the highest rate of rejection

	Alignments		
Category	Total number	Dropped number	%dropped
1 V1	23	4	17.4
1 V2	30	0	0
1 V3	28	0	0
2	23	9	39.1
3	11	3	27.3
4	16	4	25
5	12	0	0

than 25% residue identity, and the remaining one was a medium-length alignment with less than 25% identity.

Whenever SAM-T99 failed to align all the sequences, the alignment score cannot be calculated, because the scoring program requires the same number of sequences in the reference alignment and the test alignment. In these cases, the scores are treated as zero, which is a reasonable estimate of the column score, but a gross underestimate for the SPS.

For each BAliBASE category, the mean alignment scores for SAM-T99 were calculated and compared with ClustalW version 1.8. Table 2 shows the mean SPSs and CSs for both SAM-T99 and ClustalW.

The results in Table 2 show that, except where SAM-T99 dropped sequences in the alignments, SAM-T99 results are comparable to ClustalW. Whenever SAM-T99 dropped some sequences in the alignments, we counted its score to be 0, which dragged down the mean score. In the following section, we discuss how to use SAM-T99 to align all the input sequences, to provide a more reasonable comparison.

4.2 SAM-T99 tuneup, followed by forced alignment

SAM can be used to align all of the input sequences, regardless of their divergence using the *SAM-T99 forced alignment protocol*:

(i) use SAM-T99 -tuneup to generate an alignment;

(ii) use w0.5 to build an HMM using the SAM-T99 alignment;

(iii) use hmmscore to align all the input sequences against the above HMM.

The mean SPSs and CSs for this method are given in the fifth and eighth columns of Table 2. As we would expect, this method does as well or better than the direct use of SAM-T99.

The alignment scores for SAM-T99 forced alignment are summarized and compared with ClustalW in Table 3. Wilcoxon rank-sum tests of significance were done, and most of the differences between ClustalW and SAM-T99 forced alignment turn out not to be statistically significant. The most significant differences are for Category 1V2 (20–40% residue identity), for which SAM-T99 clearly outperforms ClustalW. The next most significant difference is for Category 1V1 (<25% residue identity), for which ClustalW performs better.

In the final step of the forced alignment, we have several choices for aligning the sequences to the HMM. The results in Tables 2 and 3 are for local alignment using the Viterbi algorithm. Because we believe that the test set is biased towards global aligners, we also tried doing global alignment on the final forced alignment to the HMM (though not on the iterations that created the SAM-T99-tuneup alignment from which the HMM was created).

In some earlier pairwise alignment tests, we found that SAMs posterior decoding (-adpstyle 5) was superior to Viterbi alignment for very dissimilar homologs (Cline, 2000), and so we also tried that option for the forced alignment. The posterior decoding technique uses a full forward–backward algorithm to create a matrix of the probabilities for each position of a sequence aligning to each state, then does a Viterbi path on this probability matrix, rather than on the standard dynamic programming matrix.

We tried four different settings for hmmscore:

- local alignment using the Viterbi path;
- local alignment using posterior decoding;
- global alignment using the Viterbi path;
- global alignment using posterior decoding;

The results of these forced alignments are summarized in Table 4.

As we expected, global alignment does somewhat better than local alignment on this test set, consistent with the findings presented by Thompson *et al.* (1999b). Among the 143 alignments compared, global alignment did better than local in 20 cases, while local did better in only two cases. Most of the improvement (10 of the 20) for global alignment is in Category 2, where trimming the alignment provides a strong clue to global aligners about how to align the orphan sequence. This is the only category for which the difference between global and local alignment is statistically significant ($P < 0.05$), and the difference is not significant over the entire test set.

Because real alignment problems do not usually have the information used to trim the sequences in the BAliBASE test set, we believe that using global alignment methods on Category 2 gives an inflated impression of the accuracy of sequence aligners. The local-alignment results

Table 2. Mean SPSs and CS comparison

Category	Missing/total	Mean SPS			Mean CS		
		ClustalW	SAM-T99 tuneup	SAM-T99 forced	ClustalW	SAM-T99 tuneup	SAM-T99 forced
1 V1S	3/7	0.720	0.393*	0.485	0.560	0.292*	0.292
1 V1M	1/8	0.688	0.551*	0.619	0.492	0.363*	0.395
1 V1L	0/8	0.664	0.595	0.595	0.501	0.390	0.390
1 V2S	0/11	0.913	0.920	0.920	0.820	0.838	0.838
1 V2M	0/9	0.939	0.959	0.959	0.882	0.926	0.926
1 V2L	0/10	0.954	0.964	0.971	0.888	0.903	0.903
1 V3S	0/8	0.980	0.977	0.977	0.951	0.948	0.948
1 V3M	0/10	0.979	0.982	0.982	0.954	0.960	0.960
1 V3L	0/10	0.984	0.990	0.990	0.961	0.978	0.978
1 V1	4/23	0.689	0.518*	0.570	0.516	0.351*	0.362
1 V2	0/30	0.935	0.948	0.950	0.864	0.888	0.888
1 V3	0/28	0.981	0.983	0.984	0.955	0.961	0.962
1	4/81	0.881	0.838*	0.854	0.796	0.761*	0.764
2	9/23	0.867	0.534*	0.843	0.389	0.251*	0.337
3	3/11	0.766	0.581*	0.708	0.449	0.410*	0.434
4	4/16	0.788	0.532*	0.742	0.515	0.216*	0.370
5	0/12	0.892	0.880	0.880	0.713	0.742	0.742
All	20/143	0.860	0.739*	0.831	0.666	0.589*	0.624

The missing/total column gives the number of cases for which SAM-T99 failed to produce an alignment of all sequences and the total number of alignments in the category. In the category column, v1, v2, v3 indicate <25% identity, 20–40% identity and >35% identity respectively. S, L, and M stand for small (<100 residues), medium (200–300 residues) and long (>500 residues) respectively. The asterisk * indicates that SAM-T99 dropped some sequences. In the calculation we counted those alignments as score 0—a gross underestimate for SPSs, but a reasonable estimate for CSs.

Table 3. Mean SPSs and CS comparisons of SAM-T99 forced alignment results and ClustalW version 1.8

Category	SPSs				CSs			
	ClustalW 1.8	SAM-T99 forced	Clustal better?		ClustalW 1.8	SAM-T99 forced	Clustal better?	
1 V1 (23)	0.689	0.570	17 (73.9%)	$+(P < 0.05)$	0.516	0.362	14 (60.8%)	$+(P < 0.005)$
1 V2 (30)	0.935	0.950	6 (20.0%)	$-(P < 0.005)$	0.864	0.888	6 (20.0%)	$-(P < 0.001)$
1 V3 (28)	0.981	0.984	12 (42.8%)	Same	0.955	0.962	11 (39.3%)	Same
1 (81)	0.881	0.854	35 (43.2%)	Same	0.796	0.764	31 (38.2%)	Same
2 (23)	0.867	0.843	12 (52.1%)	$+(P < 0.1)$	0.389	0.337	12 (52.2%)	$+(P < 0.05)$
3 (11)	0.766	0.708	6 (54.5%)	Same	0.449	0.434	3 (27.2%)	$-(P < 0.14)$
4 (16)	0.788	0.742	7 (43.7%)	Same	0.515	0.370	6 (37.5%)	Same
5 (12)	0.892	0.880	7 (58.3%)	Same	0.713	0.742	5 (41.7%)	Same
All (143)	0.860	0.831	67 (46.9%)	Same	0.666	0.624	57 (39.9%)	Same

In the category column, v1, v2, v3 indicate <25% identity, 20–40% identity and >35% identity respectively. The ClustalW1.8 and SAM-T99 columns show the mean scores for the methods. The 'Clustal better?' columns show the number and percentage of cases that ClustalW did better than SAM-T99. A plus sign indicates that ClustalW is significantly better, and a minus sign indicates that SAM-T99 is significantly better. The P-value is estimated based on the Wilcoxon rank sum test (Hollander and Wolfe, 1999).

are more indicative of how the aligners will work on real problems.

Similarly, when comparing Viterbi paths with posterior decoding the difference is again not striking. Out of 143 tests, global alignment with posterior decoding did better than global with Viterbi decoding in eight cases and worse in one case. Local alignment with posterior decoding did better than local with Viterbi in four cases and worse in 11 cases. None of these differences are statistically significant.

5 DISCUSSION AND FUTURE WORK

The SAM-T99 method was developed as a way to find similar proteins given a single sequence or a small seed alignment. It is an evolutionary improvement over SAM-T98, which has done very well in superfamily

Table 4. Mean SPSs and CSs comparison for SAM-T99 forced alignments with different alignment options on the final step. Although global posterior decoding does as well or better on all categories (except Category 4 for CSs), the differences are not statistically significant, except for Category 2, where global posterior decoding is significantly better than local on SPSs, with $P < 0.05$

Category	SPS				CSs			
	Local		Global		Local		Global	
	Viterbi	Posterior	Viterbi	Posterior	Viterbi	Posterior	Viterbi	Posterior
1 V1 (23)	0.570	0.570	0.597	0.601	0.362	0.387	0.387	0.387
1 V2 (30)	0.950	0.950	0.950	0.950	0.888	0.888	0.888	0.888
1 V3 (28)	0.983	0.983	0.984	0.984	0.961	0.962	0.962	0.962
1 (81)	0.854	0.853	0.862	0.863	0.764	0.772	0.772	0.772
2 (23)	0.843	0.841	0.857	0.864	0.337	0.322	0.353	0.399
3 (11)	0.708	0.721	0.760	0.772	0.434	0.418	0.436	0.492
4 (16)	0.742	0.742	0.742	0.742	0.370	0.370	0.372	0.371
5 (12)	0.880	0.880	0.881	0.881	0.740	0.744	0.744	0.744
All (143)	0.831	0.831	0.842	0.844	0.624	0.625	0.632	0.643

classification tests (Park *et al.*, 1998b; Karplus *et al.*, 1998).

In this paper, we evaluated the method as a multiple aligner, using the BAliBASE multiple-alignment test suite (Thompson *et al.*, 1999a,b). We wanted to use an established test method, since other research has questioned the quality of alignments done by hidden Markov models (Gotoh, 1999).

Our results show SAM-T99 as an aligner is comparable to ClustalW version 1.8. In some cases, the ClustalW results obtained in our test are much better than the results reported earlier (Thompson *et al.*, 1999b). We reason that the difference may be that the ClustalW version 1.8 has been tuned using BAliBASE, while the previous version of ClustalW was not tuned. SAM-T99, on the other hand, has never been tuned for BAliBASE.

Recently, Notredame *et al.* (2000) reported that T-Coffee outperforms ClustalW on BAliBASE, but direct comparison of results is difficult as Notredame made different modifications to bali_score to fix its bugs, and discarded some of the alignments from the test set. The magnitude of the improvement makes it extremely likely that T-Coffee outperforms SAM-T99 significantly on this benchmark. T-Coffee has a higher computational cost than ClustalW, so its use will probably be limited to alignments with relatively few sequences.

It is clear that we now need a new, better multiple-alignment benchmark than BAliBASE, especially as some programs have been tuned to the mark. The latest version of the DALI Domain Dictionary (Dietmann *et al.*, 2001), which uses T-Coffee to build multiple alignments from pairwise structural alignments, might be a good starting point. One might wish to add other structural alignments besides Dali's into the T-Coffee library to improve the multiple alignments further.

Being an iterative method, the complexity for SAM-T99 is $O(L^2Nk)$ where L is the length of the sequence, and N is the number of sequences to be aligned, and k is the number of iterations. This compares favorably with the $O(L^2N^2)$ algorithm used in progressive alignment methods such as ClustalW. Because SAM-T99 is linear in the number of sequences to be aligned, it has been successfully used to align over 10 000 sequences. This advantage for SAM-T99 does not show up in the results for this paper, because BAliBASE does not contain alignments of large number of sequences. In preliminary timing tests (not reported here), we have found ClustalW faster for up to about 500 sequences, and SAM-T99 faster for larger sets. For small sets (say 25 sequences), ClustalW is about 20 times faster than SAM-T99. For 10,000 sequences, SAM-T99 is about 20 times faster than ClustalW. Speed improvements for SAM-T99 are clearly possible, and the crossover point can probably be brought to fewer than 100 sequences.

We conclude that the quality of the SAM-T99 multiple alignments are high enough that little benefit would be obtained from realigning the output of a SAM-T99 search using ClustalW. On the other hand, there is also little point to using the SAM-T99 script for aligning small sets of known homologs, since ClustalW does as well with less computation. Once the set of sequences to align gets large, then SAM-T99 is preferable.

REFERENCES

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

Altschul,S., Madden,T., Schaffer,A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3899–3402.

Altschul,S.F. (1991) Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.*, **219**, 555–565.

Cline,M. (2000) *Protein Sequence Alignment Reliablity: Prediction and Measurement*, PhD thesis, University of California, Computer Science, UC Santa Cruz, CA 95064.

Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C. (1978) A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, chapter 22, National Biomedical Research Foundation, Washington, DC, pp. 345–358.

Dietmann,S., Park,J., Notredame,C., Heger,A., Lappe,M. and Holm,L. (2001) A fully automatic evolutionary classification of protein folds: dali domain dictionary version 3. *Nucleic Acids Res.*, **29**, 55–57.

Gotoh,O. (1999) Multiple sequence alignment: algorithms and applications. *Adv. Biophys.*, **36**, 159–206.

Group,G.C. (1991) *Program Manual for the GCG Package, Version 7*. Genetics Computer Group, 575 Science Drive, Madison, WI 53711.

Hannenhalli,S.S. and Russel,R.B. (2000) Analysis and prediction of functional sub-types from protein sequence alignments. *J. Mol. Biol.*, **303**, 62–76.

Higgins,D.G., Thompson,J.D. and Gibson,T.J. (1996) Using CLUSTAL for multiple sequence alignments. *Meth. Enzymol.*, **266**, 383–402.

Hollander,M. and Wolfe,D.A. (1999) *Nonparametric Statistical Methods*, 2nd edn, Wiley, New York.

Holm,L. and Sander,C. (1999) Protein folds and families: sequence and structure alignments. *Nucleic Acids Res.*, **27**, 244–247.

Hughey,R. and Krogh,A. (1996) Hidden Markov models for sequence analysis: extension and analysis of the basic method. *CABIOS*, **12**, 95–107. Information on obtaining SAM is available at http://www.cse.ucsc.edu/research/compbio/sam.html.

Hughey,R., Karplus,K. and Krogh,A. (1999) SAM: Sequence Alignment and Modeling software system, version 3. *Technical Report UCSC-CRL-99-11*, University of California, Santa Cruz, Computer Engineering, UC Santa Cruz, CA 95064. Available from http://www.cse.ucsc.edu/research/compbio/sam.html.

Jeanmougin,F., Thompson,J., Gouy,M., Higgins,D. and Gibson,T. (1998) Multiple sequence alignment with Clustal X. *Trends Biochem. Sci.*, **23**, 403–405.

Karplus,K., Barrett,C. and Hughey,R. (1998) Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, **14**, 846–856.

Karplus,K., Barrett,C., Cline,M., Diekhans,M., Grate,L. and Hughey,R. (1999) Predicting protein structure using only sequence information. *Protein Struct. Funct. Genet.*, **3** (Suppl), 121–125.

Levin,J.M., Pascarella,S., Argos,P. and Garnier,J. (1993) Quantification of secondary structure prediction improvement using multiple alignments. *Protein Eng.*, **6**, 849–854.

Morgenstern,B., French,K., Dress,A. and Werner,T. (1998) Dialign: finding local similarities by multiple sequence alignment. *Bioinformatics*, **14**, 290–294.

Editor: Is it ok? Notredame,C. and Higgins,D.G. (1996) SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.*, **24**.

Notredame,C., Higgins,D.G. and Heringa,J. (2000) T-Coffee: a novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, **302**, 205–217.

Park,J., Teichmann,S., Hubbard,T. and Chothia,C. (1997) Intermediate sequences increase the detection of homology between sequences. *J. Mol. Biol.*, **273**, 349–354.

Park,J., Karplus,K., Barrett,C., Hughey,R., Haussler,D., Hubbard,T. and Chothia,C. (1998a) Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J. Mol. Biol.*, **284**, 1201–1210.

Park,J., Karplus,K., Barrett,C., Hughey,R., Haussler,D., Hubbard,T. and Chothia,C. (1998b) Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J. Mol. Biol.*, **284**, 1201–1210. Paper available at http://www.mrc-lmb.cam.ac.uk/genomes/jong/assess_paper/assess_paperNov.html.

Pascarella,S., De Persio,R., Bossa,F. and Argos,P. (1998) Easy method to predict solvent accessibility from multiple protein sequence alignments. *Protein Struct. Funct. Genet.*, **32**, 190–199.

Sjölander,K., Karplus,K., Brown,M.P., Hughey,R., Krogh,A., Mian,I.S. and Haussler,D. (1996) Dirichlet mixtures: a method for improving detection of weak but significant protein sequence homology. *CABIOS*, **12**, 327–345.

Smith,R.F. and Smith,T.F. (1992) Pattern-Induced Multi-sequence Alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Eng.*, **5**, 35–41.

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

Tarnas,C. and Hughey,R. (1998) Reduced space hidden Markov model training. *Bioinformatics*, **14**, 401–406.

Thompson,J.D., Higgins,D.G. and Gibson,T.J. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties, and weight matrix choice. *Nucleic Acids Res.*, **22**, 4673–4680.

Thompson,J.D., Plewniak,F. and Poch,O. (1999a) BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*, **15**, 87–88.

Thompson,J.D., Plewniak,F. and Poch,O. (1999b) A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Res.*, **15**. Editor: Is it OK?

To be balanced at final stage