BIOINFORMATICS

Vol. 19 no. 11 2003, pages 1404–1411
DOI: 10.1093/bioinformatics/btg158

SATCHMO: sequence alignment and tree construction using hidden Markov models

Robert C. Edgar[1,]* and Kimmen Sjölander[2]

[1] 195 Roque Moraes Drive, Mill Valley, CA 94941 and [2] Department of Bioengineering, University of California, Berkeley, CA 94720, USA

Received on November 17. 2002; revised on February 4, 2003; accepted on February 7, 2003

ABSTRACT

Motivation: Aligning multiple proteins based on sequence information alone is challenging if sequence identity is low or there is a significant degree of structural divergence. We present a novel algorithm (SATCHMO) that is designed to address this challenge. SATCHMO simultaneously constructs a tree and a set of multiple sequence alignments, one for each internal node of the tree. The alignment at a given node contains all sequences within its sub-tree, and predicts which positions in those sequences are alignable and which are not. Aligned regions therefore typically get shorter on a path from a leaf to the root as sequences diverge in structure. Current methods either regard all positions as alignable (e.g. ClustalW), or align only those positions believed to be homologous across all sequences (e.g. profile HMM methods); by contrast SATCHMO makes different predictions of alignable regions in different subgroups. SATCHMO generates profile hidden Markov models at each node; these are used to determine branching order, to align sequences and to predict structurally alignable regions.

Results: In experiments on the BAliBASE benchmark alignment database, SATCHMO is shown to perform comparably to ClustalW and the UCSC SAM HMM software. Results using SATCHMO to identify protein domains are demonstrated on potassium channels, with implications for the mechanism by which tumor necrosis factor alpha affects potassium current.

Availability: The software is available for download from http://www.drive5.com/lobster/index.htm.

Contact: bob@drive5.com

1 INTRODUCTION

The construction of multiple sequence alignments is a focus of the computational biology community due to its importance in a wide range of applications, including homology modeling, phylogenetic tree reconstruction, sub-family classification, and identification of critical residues. When sequences are similar, many alignment methods produce good results. However, evolutionary divergence in multi-gene families can result in family members with very low pairwise similarity. Even when sequence similarity is detectable, local changes in structure between members can be significant and represent a great challenge to alignment algorithms. Loop regions, and other positions exposed to solvent, are known to be far more variable than the hydrophobic core elements or key catalytic amino acids. This is reflected in a multiple sequence alignment of proteins in a diverse family: some columns are found in conserved motifs, while others are in regions with many gaps and varying residue types.

Current multiple sequence alignment methods either treat all columns as alignable across all sequences, e.g. ClustalW (Thompson *et al.*, 1994), or single out only those columns believed to be alignable across all sequences, e.g. profile HMM methods (Krogh *et al.*, 1994; Eddy, 1996; Karplus *et al.*, 1997). Both approaches have essential limitations when applied to highly variable protein sequences. In our experience, HMM methods tend to be successful at detecting and aligning critical motifs and conserved core structure of protein families, but may not correctly align positions outside these conserved regions. Other methods are often superior to HMMs at correctly aligning sequences within similar subgroups; however, subgroups with significant divergence may not be correctly aligned to the consensus structure, causing misalignment of family-defining conserved motifs.

Here, we present a novel multiple sequence alignment method that attempts to present a more nuanced and informative view of the relationships and divergence within a set of sequences by making different predictions of alignable regions in different subgroups.

2 THE SATCHMO ALGORITHM

We call our method SATCHMO, for Simultaneous Alignment and Tree Construction using Hidden Markov mOdels. Like ClustalW, SATCHMO is a progressive method, meaning that alignments are built iteratively in pairs. First a pair of sequences is aligned. In each

*To whom correspondence should be addressed.

subsequent step, a new pair is chosen and aligned to each other. The pair could be two sequences, a sequence and an alignment, or two alignments. When the pair includes an alignment, columns in this alignment are kept intact in the combined alignment; thus a column is ‘frozen’ once it has been created. Many progressive alignment methods have been described (see for example Chan *et al.*, 1992; Feng and Doolittle, 1996; Durbin *et al.*, 1998).

SATCHMO differs from previous progressive methods in its use of profile HMMs (Krogh *et al.*, 1994) to create pairwise alignments and to determine the clustering hierarchy. It also differs in that alignments of the same sequences are typically not the same at each tree node. Alignment columns remain frozen; however alignments vary in their predictions of which columns are structurally alignable. On a path from a leaf to the root, the number of alignable columns predicted typically gets smaller as structures diverge. Thus, in contrast to other multiple sequence alignment algorithms, the output of SATCHMO cannot be presented in a single matrix. A graphical interface is therefore provided, enabling the user to browse the tree and examine the alignment produced at each node.

Progressive alignment can be viewed as an agglomerative clustering procedure for building a binary tree in which the leaves represent sequences and internal nodes represent clusters of two or more sequences. For each cluster, SATCHMO constructs an alignment of its sequences and a profile HMM. At the start of each iteration, HMM scoring is used to identify the two most closely related clusters. These two clusters are then combined by aligning the alignment of sequences in one cluster to the HMM of the other. A prediction is made of the positions that are structurally alignable in the combined alignment using a function derived from match state scores. A new HMM is then built from the combined alignment in which match states correspond to those positions, completing the construction of a new cluster. This procedure exploits proven HMM techniques that: (a) rank homologs through scoring (Barrett *et al.*, 1997); (b) build accurate profiles from small numbers of sequences (Sjölander *et al.*, 1996); and (c) accurately combine two alignments having low sequence similarity (Edgar, submitted).

The implementation described here is SATCHMO version 2, which differs in a few significant details from SATCHMO version 1 (Edgar and Sjölander, 2003); those differences will be noted as the algorithm is discussed.

2.1 HMM architecture

We use the profile HMM architecture described by Krogh *et al.* (1994) as shown in Figure 1. In SATCHMO version 1 we used a different architecture as employed by the HMMER package (Eddy, 1996). The Krogh architecture is more general in that it permits transitions between delete and insert states; these are forbidden in HMMER, preventing a rigorous accounting for transition costs in our alignment and scoring methods.

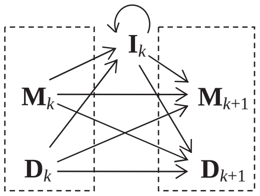


Fig. 1. Two consecutive nodes k and $k+1$ in a profile HMM. Letters represent states, arrows represent transitions. Match (M) and insert (I) states emit residues; delete (D) states are silent. Insert state I_{k+1} and transitions out of M_k and D_k are not shown.

2.2 Sequence weighting

Following standard practice, we employ relative weights to compensate for correlation among the sequences. We chose the method described by Gerstein *et al.* (1994). Sequence weights are re-calculated in each alignment using only those positions predicted to be in alignable columns. The total sequence weight used in the Dirichlet mixture computation is determined using the method employed by the UCSC SAM HMM software suite (Hughey and Krogh, 1996). We define the number of bits saved relative to the background (Kevin Karplus, personal communication) as:

$$b = 1/M \sum_k \sum_a P_k(a) \log_2(P_k(a)/P_0(a)). \qquad (1)$$

Here, $k = 1 \ldots M$ is the HMM node number, a is the amino acid type, $P_k(a)$ is the emission probability of a in the kth match state, and $P_0(a)$ is the approximation to the background probability of a obtained by applying the Dirichlet regularizer to a vector of zero counts. We iteratively adjust the total sequence weight until b converges on a desired value, which, following SAM, is set to 0.5 by default. In SATCHMO version 1, we set the total weight to be an estimate of the number of independent sequences using a heuristic based on the average degree of residue conservation, subject to a ceiling which was a parameter of the algorithm. Preliminary results (unpublished) suggest that the bits saved method gives better alignments.

2.3 HMM construction

A profile HMM is constructed from a multiple sequence alignment. Each column in the alignment is tagged to indicate whether or not it is predicted to be structurally alignable. Following the convention commonly used in HMM software, we use upper-case letters to indicate alignable columns and lower-case letters otherwise. One node is created for each alignable column. Probability distributions for state transitions and for match state emissions are constructed by combining observed counts with Dirichlet mixture priors (Sjölander *et al.*, 1996). The probability distribution for insert state emissions is set to the background probabilities of amino acids observed in nature.

2.4 Aligning an alignment to an HMM

A central step in SATCHMO is to score and align an alignment to an HMM, keeping columns of the input alignment intact. In SATCHMO version 1, this was done using an approximate method that treated gaps as pseudo-residues and required all sequences to take the same path through the HMM. In version 2 we use an extended Viterbi algorithm that correctly accounts for transition scores of sequences that must take different paths through the HMM due to gaps (Edgar, submitted). This method guarantees to find the optimal alignment with only a small increase in computational complexity over the method used in version 1.

Consider a multiple sequence alignment A and a profile HMM H. An alignment of A to H may be specified by assigning an emitter state in H to each column of A. We call such an assignment a *route*, which can be viewed as a generalization of a path (if delete states are added from all nodes for which the match state is not assigned, then a route is exactly the path that a sequence containing no gapped positions must take). Specifying a route π uniquely determines the path π_s that a given sequence s in A must take through H. Let $P(s|\pi_s)$ be the probability that π_s generates s. The probability of A given route π is:

$$P(A|\pi) = \prod_s P(s|\pi_s). \tag{2}$$

A most probable route π^+ (there may be more than one) is then:

$$\pi^+ = \mathrm{argmax}_\pi P(A|\pi). \tag{3}$$

The extended Viterbi algorithm is used to determine π^+ and $P(A|\ \pi^+)$.

2.5 HMM scoring

Following HMMER, we define a simple null model consisting of a single insert state that emits letters according to the background distribution. The self-loop probability of the state is tuned so that the average length of the emitted sequence is the average length of a protein. We denote by $P(s|Null)$ the probability of sequence s being emitted by this model, and define the score of A against H as:

$$S(A,\ H) = 1/(MN)\log_2(P(A|\pi^+)/P(A|Null)). \tag{4}$$

Here, M is the number of nodes in H, and N is the number of sequences in A (strictly, the total sequence weight). We divide by M as an approximate correction for the well-known length bias of profile HMM scores, and by N to give a per-sequence score. This score is designed so that values for alignments of different sizes are comparable.

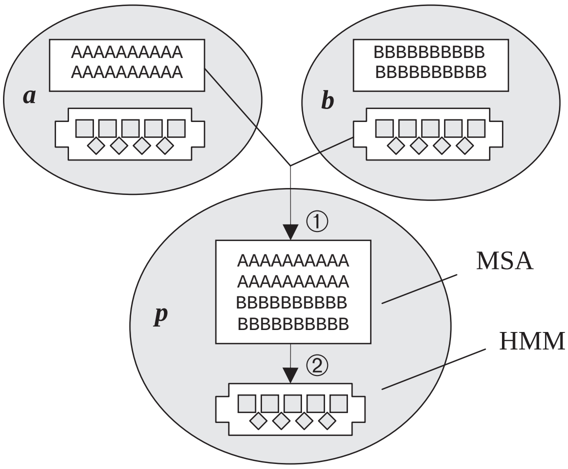


Fig. 2. This diagram shows schematically how SATCHMO combines two clusters a and b to create a new cluster p. Each cluster contains a multiple sequence alignment (MSA) and a profile hidden Markov model (HMM) built from that MSA. Here, cluster b is selected as the template, meaning that its HMM is used in the process, and cluster a is selected as the target. The MSA from a is aligned ① to the HMM from b, creating a combined alignment of the sequences in the two clusters. Then a new HMM is created ② from this new MSA.

2.6 Similarity measure

Given two alignments A_i and A_j, we construct a model H_i from A_i and a model H_j from A_j. We define a symmetrical similarity measure as follows:

$$s_{ij} = (S(A_i,\ H_j) + S(A_j,\ H_i))/2. \tag{5}$$

2.7 Prediction of alignable positions

Given two alignments, we generate a combined alignment by constructing an HMM from one (the *template*), and aligning the other (the *target*) to that HMM, as shown in Figure 2. We expect that if in this alignment there is a high probability (relative to the null model) of a match state emitting a given template column, then the template and target columns are alignable. A histogram of the relative match state scores (*affinities*) of all model nodes typically shows regions of generally high and low values; the regions of high values are assumed to be alignable (see Fig. 3). Regions of sufficiently low values are not believed to be alignable. The minimum affinity value for predicting alignability is a parameter of the algorithm.

Affinity proves to be a noisy signal; we therefore smooth over a window, the length of which is another parameter

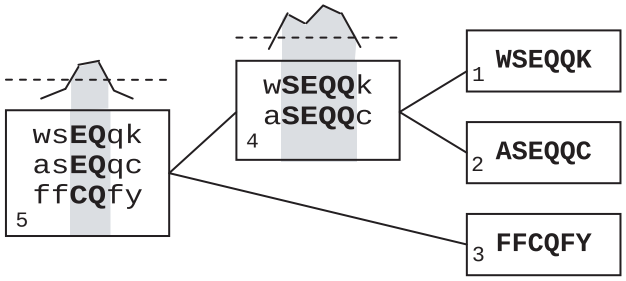


Fig. 3. A tree built by SATCHMO. Three input sequences are found in the leaf nodes numbered 1, 2 and 3. At each interior node, there is an alignment of the sequences in the sub-tree below that node. Graphs show the smoothed affinity per column. The minimum affinity threshold is indicated as a dashed line (- - -). Columns for which the smoothed affinity exceeds this threshold are tagged as aligned (upper-case letters, shaded background); other columns are tagged as not aligned (lower case). In the first interior node (4), four columns are predicted to be alignable. This is reduced to two columns in the root node (5) where a third, more diverged sequence (3) has been aligned to the alignment of the first two sequences (1 and 2).

of the algorithm. In detail, this is implemented as follows.

Let $m_k(a)$ be the match state emission probability of amino acid a in node k of the template HMM, $q(a)$ be the background (null model) probability of a, $A_s(i)$be the amino acid in column i of sequence s in the target alignment A, and $c_k(\pi^+)$ be the column in A to which node k is assigned in π^+. We then define the affinity of node k to the target to be:

$$f_k = 1/N \log_2 \prod_s m_k(A_s(c_k(\pi^+)))/q(A_s(c_k(\pi^+))). \tag{6}$$

Sequences that contain a gap in $c(k, \pi^+)$ are excluded from the product. If π^+ passes through the delete state of node k, f_k is defined to be zero. We next define $W(k, w)$ to be the set of nodes $k-\lfloor w/2 \rfloor, k-\lfloor w/2 \rfloor+1 \ldots k+\lfloor w/2 \rfloor$, excluding from this list node numbers < 1 (before the beginning of the model) or $> M$ (past the end). The value w is the *window length*, and is required to be an odd integer $\geqslant 1$. We denote the number of nodes in this set by $|W(k, w)|$. In the case of a typical node, $|W(k, w)| = w$; close to the beginning or end of a model the window contains fewer nodes. The smoothed affinity F_k of node k is defined to be the average over the nodes in the window centered on k:

$$F_k = |W(k, w)|^{-1} \sum_l f_l, \tag{7}$$

where $l \in W(k, w)$. The kth position is predicted to be alignable if and only if the smoothed affinity is not less than a given value Z:

$$F_k \geqslant Z. \tag{8}$$

Z is the *minimum smoothed affinity threshold.*

2.8 Algorithm

The SATCHMO algorithm may be summarized as follows.

Input: A set of unaligned protein sequences.

Step 1 Create a cluster for each input sequence and construct an HMM from the sequence (Section 2.3), tagging each position in the sequence as alignable. This results in a set of clusters, each having one sequence and one HMM built from that sequence. For each cluster, create a tree node with no edges.

Step 2 Calculate the similarity (Equation 5) of all pairs of clusters and identify a pair *ab* with highest similarity. (If more than one pair has the highest similarity, choose one arbitrarily.) Choose *a* or *b* to be the template according to which gives the highest score $S(A, H)$ (Equation 4). The other cluster then becomes the target. Align the target to the template (Section 2.4), creating a combined alignment of the sequences in the two clusters. Predict alignable regions in this combined alignment (Section 2.7) and tag the columns accordingly. Construct a profile HMM from this combined alignment (Section 2.3). Create a tree node *p* corresponding to the new cluster, and add edges *pa* and *pb*.

Repeat Step 2 until: (a) all sequences are assigned to one cluster, (b) the highest similarity between clusters is below a user-defined threshold, or (c) no alignable positions are predicted, in which case the algorithm terminates without creating a new cluster.

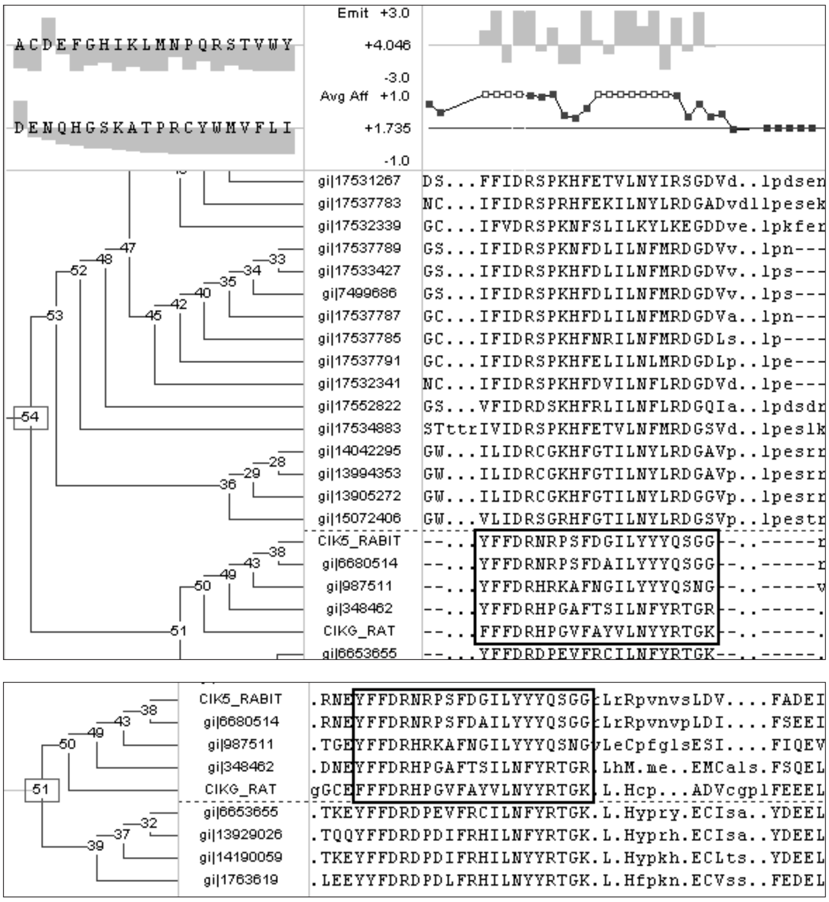
Output: A set of one or more binary trees in which each leaf contains an input sequence and each node contains an HMM and an alignment of the sequences in its sub-tree.

2.9 Complexity

Given Nsequences of length L, the space complexity of SATCHMO is dominated by the dynamic programming matrix used by the Viterbi algorithm, which is O(L^2). The total time complexity is O($L^2N^2 + LN^3$).

2.10 Graphical interface

Given N sequences, SATCHMO produces N–1 multiple sequence alignments and a binary tree. Conventional tree and alignment viewers cannot show this information in an easily assimilated fashion; our implementation therefore



```
gi|17531267   DS...FFIDRSPKHFETVLNYIRSGDVd..lpdsen
gi|17537783   NC...IFIDRSPRHFEKILNYLRDGADvdllpesek
gi|17532339   GC...IFVDRSPKNFSLILKYLKEGDDve.lpkfer
gi|17537789   GS...IFIDRSPKNFDLILNFMRDGDVv..lpn---
gi|17533427   GS...IFIDRSPKHFDLILNFMRDGDVv..lps---
gi|7499686    GS...IFIDRSPKHFDLILNFMRDGDVv..lps---
gi|17537787   GC...IFIDRSPKHFDLILNFMRDGDVa..lpn---
gi|17537785   GC...IFIDRSPKHFNRILNFMRDGDLs..lp----
gi|17537791   GC...IFIDRSPKHFELILNLMRDGDLp..lpe---
gi|17532341   NC...IFIDRSPKHFDVILNFLRDGDVd..lpe---
gi|17552822   GS...VFIDRDSKHFRLILNFLRDGQIa..lpdsdr
gi|17534883   STttrIVIDRSPKHFETVLNFMRDGSVd..lpeslk
gi|14042295   GW...ILIDRCGKHFGTILNYLRDGAVp..lpesrr
gi|13994353   GW...ILIDRCGKHFGTILNYLRDGAVp..lpesrr
gi|13905272   GW...ILIDRCGKHFGTILNYLRDGGVp..lpesrr
gi|15072406   GW...VLIDRSGRHFGTILNYLRDGSVp..lpestr
CIK5_RABIT    --...YFFDRNRPSFDGILYYYQSGG--..-----r
gi|6680514    --...YFFDRNRPSFDAILYYYQSGG--..-----r
gi|987511     --...YFFDRHRKAFNGILYYYQSNG--..-----v
gi|348462     --...YFFDRHPGAFTSILNFYRTGR--..-----.
CIKG_RAT      --...FFFDRHPGVFAYVLNYYRTGK--..-----.
gi|6653655    --...YFFDRDPEVFRCILNFYRTGK--..-----.
```

```
CIK5_RABIT    .RNEYFFDRNRPSFDGILYYYQSGGtLrRpvnvsLDV....FADEI
gi|6680514    .RNEYFFDRNRPSFDAILYYYQSGGtLrRpvnvpLDI....FSEEI
gi|987511     .TGEYFFDRHRKAFNGILYYYQSNGvLeCpfglsESI....FIQEV
gi|348462     .DNEYFFDRHPGAFTSILNFYRTGR.LhM.me..EMCals.FSQEL
CIKG_RAT      gGCEFFFDRHPGVFAYVLNYYRTGK.L.Hcp...ADVcgplFEEEL
gi|6653655    .TKEYFFDRDPEVFRCILNFYRTGK.L.Hypry.ECIsa..YDEEL
gi|13929026   .TQQYFFDRDPDIFRHILNFYRTGK.L.Hyprh.ECIsa..YDEEL
gi|14190059   .TKEYFFDRDPDIFRHILNYYRTGK.L.Hypkh.ECLts..YDEEL
gi|1763619    .LEEYYFDRDPDLFRHILNYYRTGK.L.Hfpkn.ECVss..FEDEL
```

Fig. 4. Two views of the SATCHMO graphical interface showing an alignment of two groups of sequences that share a common domain: voltage gated potassium channels and TNF-alpha induced protein B12 homologs (Section 3.6). For readability, these views have been cropped: neither view shows the whole tree or a complete alignment; also, widgets such as scrollbars have been eliminated. In the upper view, the root node (54) of the tree (lower-left panel) is selected and the root alignment is shown (lower-right panel). The upper-right panel displays a histogram of the affinity at each node (Equation 6), below it is a graph of the smoothed affinity (Equation 7). The upper-left panel shows the affinity contribution from each amino acid type in a column selected by the user, sorted alphabetically (upper) and by score (lower). SATCHMO successfully separated the two groups at the root node: the K^+ channels are all found under node 53 and the B12 homologs under node 51. In the lower view, node 51 is selected to show the alignment of the B12 homologs only. For comparison, a core block of the common domain has been highlighted in both views (rectangle). Upper case letters indicate positions predicted to be alignable. The dashed horizontal line in the lower-right panel separates the template alignment (above line) from the target alignment (below line). Edge lengths in the tree are chosen for readability in the display and are uninformative.

includes a graphical interface that shows the tree and alignments in an integrated display (Fig. 4). The user selects an alignment by clicking on a node of the tree.

3 VALIDATION

3.1 Reference alignments

We used version 1 of the BAliBASE benchmark alignment database (Thompson *et al.*, 1999a) as a source of reference alignments. BAliBASE is divided into five reference sets. Ref1 contains alignments of a small number (<6) of equidistant sequences, meaning that the percent identity of all pairs is within a specified range. These Ref1 alignments contain sequences of similar length, with no large insertions or extensions. Alignments in Ref2 combine up to three distantly related sequences ($<25\%$ identical) from Ref1 with a family of at least 15 closely related sequences. Ref3 contains alignments of up to four subgroups, with $<25\%$ identity between sequences from different groups. Ref4 contains alignments with long N/C-terminal extensions of up to 400 residues. Ref5 has long insertions of up to 100 residues. Ref1, 2 and 3 are divided into groups with short, medium and long sequences. Ref1 is further subdivided by percent identity.

3.2 Alignment quality scoring

Given an alignment produced by an algorithm (a *test* alignment), we need a score that compares this with a trusted alignment of the same sequences (the *reference* alignment). BAliBASE provides a module (*baliscore*) that defines two scores, SP (sum of pairs) and TC (total columns). SP is the ratio of the number of correctly aligned pairs of core block positions in the test alignment to the number of aligned pairs in the reference alignment. TC is the ratio of the number of correctly aligned core block columns in the test alignment to the number of core block columns in the reference alignment. Both SP and TC range from 1.0 for perfect agreement to 0.0 for no agreement. The designers of BAliBASE recommend SP as the best quality score for Refs 1, 2 and 3, TC as the best score for Refs 4 and 5 (Thompson *et al.*, 1999b). We wrote our own module to compute SP and TC as the published *baliscore* software module produces incorrect results on some inputs: specifically we found that *baliscore* would report scores that were less than the correct value for alignments with gapped positions. Using the published *baliscore* did not change the overall rankings of the tested methods versus our own scoring module, but reduced median scores.

3.3 Algorithm parameters

In addition to Dirichlet mixture priors, SATCHMO has two parameters: Z, the minimum smoothed affinity, and w, the window length for affinity smoothing. For the BAliBASE reference alignments, we found that the SP and TC scores were optimized by setting Z less than the smallest observed smoothed affinity value, which is equivalent to disabling the prediction of alignable columns described in Section 2.7. Thus, all alignment columns were used to create HMM match states. We believe that represents an artifact of the BAliBASE data and scoring function (see **Discussion**). Dirichlet mixture parameters were set to the defaults used in SAM. We configured the Viterbi algorithm to produce global alignments as this proved to give better results on BAliBASE than local alignments, following the trend found in Thompson *et al.* (1999b) analysis of several algorithms.

3.4 Comparison with ClustalW and SAM

We chose to compare the performance of SATCHMO with two other methods: ClustalW, and the *tuneup* script found in the UCSC SAM package. We consider these tools to be high-quality representatives of the non-probabilistic and HMM approaches to sequence alignment respectively. ClustalW has been shown to have excellent performance against BAliBASE (Thompson *et al.*, 1999b). We used ClustalW version 1.81 with default parameters. Karplus and Hu (2001) found that *tuneup* has comparable performance to ClustalW against BAliBASE; a conclusion that is supported by our own results. Following Karplus and Hu, we assigned zero scores to the 18 reference sets where *tuneup* failed to produce a complete alignment of the test sequences owing to rejection of one or more sequences deemed to be too distantly related.

Table 1. Here we show the median score for the three methods in each reference category and for the complete set of BAliBASE alignments. The value shown is the median of SP for Refs 1, 2 and 3 and of TC for Refs 4 and 5

	ClustalW	SAM *tuneup*	SATCHMO
Ref1 <25% id short	0.72	0.40	0.50
Ref1 <25% id medium	0.68	0.61	0.58
Ref1 <25% id long	0.64	0.60	0.60
Ref1 20–40% id short	0.92	0.97	0.94
Ref1 20–40% id medium	0.96	0.96	0.95
Ref1 20–40% id long	0.96	0.99	0.93
Ref1 >35% id short	0.99	0.99	0.98
Ref1 >35% id medium	0.98	0.99	0.97
Ref1 >35% id long	0.99	0.99	0.99
All Ref1	*0.94*	*0.97*	*0.94*
Ref2 short	0.88	0.00	0.83
Ref2 medium	0.86	0.89	0.87
Ref2 long	0.86	0.87	0.78
All Ref2	*0.86*	*0.82*	*0.83*
Ref3 short	0.72	0.00	0.82
Ref3 medium	0.74	0.76	0.71
Ref3 long	0.83	0.74	0.85
All Ref3	*0.81*	*0.74*	*0.85*
Ref4	0.52	0.13	0.70
Ref5	0.58	0.75	0.58
All BAliBASE	**0.88**	**0.89**	**0.88**

3.5 Results

We created alignments using SATCHMO version 2, ClustalW and *tuneup*. Median scores for each reference category and for BAliBASE overall are shown in Table 1. The scores show high variability, even within a narrow category. For example, the SATCHMO SP scores in Ref1, sub-category short sequences with less than <25% identity, were: 1aboA = 0.400, 1idy = 0.223, 1r69 = 0.625, 1tvxA = 0.267, 1ubi = 0.500, 1wit = 0.815, 2trx = 0.614. We also found that individual scores from SATCHMO alignments varied significantly with different choices of Dirichlet mixtures, though the median scores tended to remain similar. Pearson rank sum tests showed no statistically significant difference between any pair of algorithms.

3.6 Domain identification

Our preliminary experiments with SATCHMO suggest that it is effective at identifying protein domains. In

Figure 4, we show the tree constructed by SATCHMO for two sets of proteins: TNF-alpha-induced protein B12 and homologs, and voltage-gated potassium channels. The surprising homology between these two groups was discovered by one of us (Sjölander, previously unpublished) while scoring the NR database with an HMM constructed for voltage-gated potassium channels, where these B12 proteins received weak but significant scores. SATCHMO assigns these two groups to separate subtrees, and identifies a common domain. Our analysis shows this region to be the tetramerization (T1) domain of potassium channels, for which several solved structures exist (e.g. PDB entry 3KVT). This allows us to predict the fold of TNF-alpha-induced protein B12 and homologs. Intriguingly, tumor necrosis factor alpha is known to affect potassium current, but the precise mechanism is unknown (Soliven *et al.*, 1991; McLarnon *et al.*, 1993). Since TNF-alpha induces the B12 protein (and, presumably, its homologs), and B12 appears to share a common fold with the tetramerization domain, we predict that a possible mechanism by which TNF-alpha affects potassium current is by inducing B12, which then tetramerizes with potassium channels, thereby affecting potassium current.

4 DISCUSSION

We have presented SATCHMO, a novel approach to multiple sequence alignment and tree construction that is designed to accommodate structurally divergent families of related protein sequences. The algorithm predicts different extents of alignable regions in subsets of variable structure.

Tests using the BAliBASE benchmark data set show that SATCHMO, on average, produces alignments of the same accuracy as ClustalW and the UCSC SAM *tuneup* module: mature tools that are respected in the scientific community. To the best of our knowledge, only three algorithms—PRRP (Goto, 1996), T-Coffee (Notredame *et al.*, 2000) and MAFFT (Katoh *et al.*, 2002)—have been shown to produce more accurate alignments on BAliBASE. Both PRRP and T-Coffee have significantly higher computational complexity than SATCHMO. MAFFT was brought to our attention after this work was completed; comparison with MAFFT is left to future work. We find it encouraging that SATCHMO has achieved a competitive level of accuracy relatively early in its development and believe that its performance can be improved by refining various aspects of the algorithm.

The BAliBASE methodology has a number of issues which makes it less than ideal for validating SATCHMO. Most alignments (except those in Ref5) are limited to core block regions, with positions before and after the core block deleted as needed. Such trimming cannot be performed in typical applications, as core blocks are not known *a priori*. BAliBASE is therefore biased towards algorithms that are designed for global alignment, such as ClustalW. (For these experiments, we configured SATCHMO for semi-global alignment, i.e. alignments global to the template HMM, partly for this reason.) Scoring is done only in core block regions, so does not measure the ability of an algorithm to distinguish alignable from unalignable regions, nor does it penalize over-aligning, i.e. aligning regions that are not structurally superposable. It is therefore not surprising that BAliBASE scores are optimized by forcing SATCHMO to align all columns assigned to match states (see Section 3.3). The BAliBASE score gives no credit for unaligning a region, even if this is structurally correct, so the score of a SATCHMO alignment can only be reduced by raising the value of the alignment threshold Z. Larger values of Z may be more informative to the user in real applications by successfully identifying superposable regions, however in BAliBASE the score can only decrease with larger Z due to a core block position falling below the given threshold and therefore being flagged as unaligned. (Strictly speaking, it is possible that the alignment could be improved via a shift in a later iteration, increasing the score enough to compensate for any unaligned core block positions, but this is presumably unlikely.)

Trees produced by SATCHMO are designed to model the structural similarity among a group of related proteins. This is in contrast with traditional phylogenetic tree estimation algorithms which explicitly attempt to model, or reconstruct, the evolutionary history among taxa. Since protein function is mediated by protein structure, and evolution conserves protein function, we expect that a tree topology that clusters proteins that are similar structurally (and functionally) to be more consistent with the true evolutionary history than one that does not. But SATCHMO is not designed with any explicit model of evolution, and we do not assert that trees produced by SATCHMO should be interpreted as modeling the evolutionary history. We tested SATCHMO trees by comparing with the structural and functional classifications of the SCOP database (Murzin *et al.*, 1995), which are based on expert analysis of solved structures and experimental evidence of function. We measured the ability of trees constructed by SATCHMO to reproduce SCOP family classifications, and found them to be superior to trees produced by neighbor joining and heuristic search using the parsimony score using the PAUP* package (Swofford, 2002), based on multiple sequence alignments generated by ClustalW or T-Coffee (Edgar and Sjölander, submitted).

In its current implementation, SATCHMO aligns 100 sequences of length 100 in 118 s on a 2.5 GHz Pentium 4 desktop PC: fast enough to make high-throughput applications tractable.

5 FUTURE WORK

We plan to review several aspects of the algorithm with a view to possible refinements. Tests (Edgar, submitted) suggest that Equation(4) does not predict the best choice of target and template once a closest pair of clusters has been identified (Section 2.8, Step 2); we therefore hope to find a score that is better suited for this purpose (see e.g. Barrett *et al.*, 1997). We regard the use of a smoothed log-odds score (Equation 7) and threshold (Equation 8) for predicting alignable regions as a simple, preliminary heuristic and plan to try alternative approaches. We also plan to implement alternatives to BAliBASE for validating alignment accuracy and the prediction of superposable regions.

ACKNOWLEDGEMENTS

The authors thank Melissa Cline, Kevin Karplus and Sean Eddy for helpful discussions, and Wayne Christopher for providing the source code of a speed-optimized function for calculating the log-gamma function.

REFERENCES

Barrett,C., Hughey,R. and Karplus,K. (1997) Scoring hidden Markov models. *CABIOS*, **13**, 191–199.

Chan,S.C., Wong,A.K.C. and Chiu,D.K.Y. (1992) A survey of multiple sequence comparison methods. *Bull. Math. Biol.*, **54**, 563.

Durbin,R., Eddy,S., Krogh,A. and Mitchison,G. (1998) *Biological Sequence Analysis*. Cambridge University Press.

Eddy,S.R. (1996) Hidden Markov models. *Curr. Opin. Struct. Biol.*, **6**, 361–365.

Edgar,R.C. and Sjölander,K. (2003) Simultaneous sequence alignment and tree construction using hidden Markov models. *Pac. Symp. Biocomput.*, 180–191.

Feng,D.-F. and Doolittle,R.F. (1996) Progressive alignment of amino acid sequences and construction of phylogenetic trees from them. *Methods Enzymol.*, **266**, 351–360.

Gerstein,M., Sonnhammer,E.L. and Chothia,C. (1994) Volume changes in protein evolution. *J. Mol. Biol.*, **236**, 1067–1078.

Goto,O. (1996) Significant improvement in accuracy of multiple protein alignments by iterative refinement as assessed by reference to structural alignments. *J. Mol. Biol.*, **264**, 823–838.

Hughey,R. and Krogh,A. (1996) Hidden Markov models for sequence analysis: extension and analysis of the basic method. *CABIOS*, **12**, 95–107.

Karplus,K. and Hu,B. (2001) Evaluation of protein multiple alignments by SAMT99 using the BAliBASE multiple alignment test set. *Bioinformatics*, **17**, 713–720.

Karplus,K., Sjölander,K., Barrett,C., Cline,M., Haussler,D., Hughey,R., Holm,L. and Sander,C. (1997) Predicting protein structure using hidden Markov models. *Proteins*, (Suppl. 1), 134–139.

Katoh,K., Kazuharu,M., Kuma,K. and Miyata,M. (2002) MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Res.*, **30**, 2059–3066.

Krogh,A., Brown,M., Mian,I.S., Sjolander,K. and Haussler,D. (1994) Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.*, **235**, 1501–1531.

McLarnon,J.G., Michikawa,M. and Kim,S.U. (1993) Effects of tumor necrosis factor on inward potassium current and cell morphology in cultured human oligodendrocytes. *Glia*, **9**, 120–126.

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

Murzin,A.G., Brenner,S.E., Hubbard,T. and Chothia,C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**, 536–540.

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

Soliven,B., Szuchet,S. and Nelson,D.K. (1991) Tumor necrosis factor inhibits K^+ current expression in cultured oligodendrocytes. *J. Membr. Biol.*, **124**, 127–137.

Swofford,D.L (2002) *PAUP*, Phylogenetic Analysis Using Parsimony (*and Other Methods)*, Version 4, Sinauer Associates, Sunderland, 1A.

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.*, **27**, 2682–2690.