BIOINFORMATICS

Vol. 18 no. 2 2002
Pages 306–314

Predicting reliable regions in protein sequence alignments

Melissa Cline[1], Richard Hughey[2] and Kevin Karplus[2]

[1]Center for Biomolecular Science and Engineering and [2]Department of Computer Engineering, Jack Baskin School of Engineering, University of California, Santa Cruz, CA 95064, USA

Received on April 19, 2001; revised on August 1, 2001; accepted on August 21, 2001

ABSTRACT

Motivation: Protein sequence alignments have a myriad of applications in bioinformatics, including secondary and tertiary structure prediction, homology modeling, and phylogeny. Unfortunately, all alignment methods make mistakes, and mistakes in alignments often yield mistakes in their application. Thus, a method to identify and remove suspect alignment positions could benefit many areas in protein sequence analysis.

Results: We tested four predictors of alignment position reliability, including near-optimal alignment information, column score, and secondary structural information. We validated each predictor against a large library of alignments, removing positions predicted as unreliable. Near-optimal alignment information was the best predictor, removing 70% of the substantially-misaligned positions and 58% of the over-aligned positions, while retaining 86% of those aligned accurately.

Availability: The shift score alignment comparison algorithm is available online at http://www.soe.ucsc.edu/research/compbio/HMM-apps/compare-align.html and from the authors on request.

Contact: cline@soe.ucsc.edu

INTRODUCTION

In the words of Jones (1997), 'as the familiar joke goes, there are really only three things that govern the overall accuracy of comparative modeling: alignment quality, alignment quality, and . . . alignment quality.' Comparative modeling is one application of sequence alignment; others include tertiary and secondary structure prediction, prediction of functional residues, phylogenic analysis, and gene prediction. Sadly, all alignment methods make mistakes; sequence alignment is not a solved problem. As suggested by Dr Jones, mistakes in an alignment can compromise its application. Thus, before one uses an alignment, one would like to know which portions can be trusted. To this aim, we addressed the task of studying fold recognition alignments, and predicting which positions are reliable.

One approach to predicting alignment reliability involves *near-optimal alignment analysis* (Vingron, 1996): assessing alignment positions according to other alignments of the same sequences. Positions consistent among many alignments are considered more reliable while those that vary between different alignments are considered more suspect. Many have used such information for building alignments, and report greater accuracy (Holmes and Durbin, 1998; Miyazawa, 1994; Zhang and Marr, 1995). Vingron and Argos (1990) demonstrated a method for estimating the reliability of an alignment position by comparing the score of the alignment to that of the best-scoring alignment that omits the position. Mevissen and Vingron (1996) later validated it on a large library of moderate to difficult alignments.

Yu and Smith (1999) approached near-optimal alignment information from a Hidden Markov Model (HMM) framework. Given a pairwise alignment and an HMM trained on one sequence, they estimate the importance of each alignment position according to the HMM's *forward–backward* probabilities, also called *posterior decoding* probabilities. For any node in a model and residue in a sequence, these probabilities reflect the likelihood of aligning the residue to the node, given all possible alignments of the sequence to the model.

Another approach is to measure the score of segments of the alignment and omit low-scoring regions. Many local alignment methods use this approach, including BLAST (Altschul *et al.*, 1990) and FASTA (Pearson and Lipman, 1988). Zhang *et al.* (1999) proposed a system to improve alignments by identifying and removing low-scoring regions.

By conventional wisdom, there are variable regions within protein families, such as loop regions, and alignment to such regions is more suspect. Dopazo (1997) suggested a method to quantify variability within an alignment according to the PAM distances between the aligned sequences, and suggested that when a new sequence is aligned to the family, its alignment should be trusted less in the variable positions.

We explored four predictors of alignment position

reliability: (1) the posterior decoding cost of each position, derived from the HMM forward–backward information; (2) the distance in the template sequence to the nearest secondary structural element; (3) a log-likelihood ratio reflecting the score of each alignment position; and (4) a neural network combining these predictors with others.

To test these predictors, we developed an alignment quality measure called the *shift score*. This measure compares predicted and structural alignments; reflects many types of alignment error including misalignment, aligning too much, and aligning too little; and reports them in a single number. The number compares well to accepted measures. Where they disagree, arguments can be made for the shift score.

We tested the predictors on alignments of remote homologs by removing columns predicted as unreliable and measuring the change in shift score. The predictors removed as many as 73.5% of the unreliable positions, while preserving 81.8% of the accurate ones, yielding an improvement in overall alignment quality of about 15%.

ALIGNMENT TRIMMING METHODS

Our goal was to identify and remove suspect positions from fold recognition alignments, where some target sequence is aligned to some template sequence family. Figure 1 describes our experimental design. First, we selected a number of pairs of remote homologs of known structure. Following a fold recognition framework, we labeled one sequence as the *template sequence* and one as the *target sequence*, where all knowledge of the target sequence's structure is set aside until the end of the experiment. We selected a number of alignments of the template and target sequences, where the alignments could be built using information on the template sequence structure. We then trimmed the alignments by identifying and removing suspect positions, experimenting with various predictors of alignment quality. Finally, we evaluated the trimming methods by scoring the trimmed alignments against structural alignments of the template and target sequences.

Selection of the sequence pairs

For this work, we chose 200 pairs of remotely-related sequences. The 'twilight zone,' now described as below about 20% sequence identity (Rost, 1999), is where sequence analysis becomes most challenging: where contemporary methods become less reliable and where the next generation of improvements must be made. We selected 200 pairs of structures with high structural similarity and low sequence similarity. We ensured low sequence similarity by discarding any pair of sequences that a simple pairwise method such as FASTA (Pearson and Lipman, 1988) could align well. We ensured high structural similarity by selecting sequences that were superimposed well by each of three structural aligners:

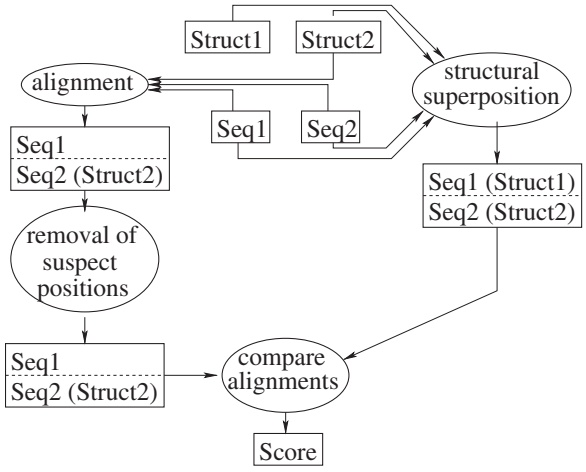

Fig. 1. This figure illustrates the experimental process performed for a single pair of sequences, *Seq1* and *2*. The process follows a fold recognition framework in which *Seq2* is the template sequence; its structure is known and can be used to build or refine the alignment of the two sequences. *Seq1* is the target sequence; its structure would not be known in a fold recognition scenario, and its structural information is not used until the final scoring phase. First, an alignment for *Seq1* and *2* is obtained. Next, the alignment is trimmed by removal of suspect positions. Then, this trimmed alignment is compared to a structural alignment of *Seq1* and *2*, generating an alignment score. This score is compared to the score of the untrimmed alignment, measuring the effectiveness of the trimming process.

DALI (Holm and Sander, 1997), VAST (Gilbrat *et al.*, 1996) and the Yale aligner (Gerstein and Levitt, 1998). We used standards of structural superposition proposed by each author or accepted within the scientific community: a VAST p-value $\leqslant 0.0001$, a Yale RMSD $\leqslant 4.0$, and a DALI score $\geqslant 7.0$. The 200 sequence pairs ranged from 3 to 24% sequence identity, with an average identity of 12%. This approximates the difficulty of a CASP3 fold recognition target (Murzin, 1999).

We divided this set of 200 remote homology pairs into an optimization set of 130 pairs and a validation set of 70 pairs. The optimization pairs were used to develop the predictors, with 65 assigned for neural network training and 65 for threshold selection. All validation pairs were reserved for final performance testing. Note that each sequence pair corresponds to two assignments of template and target sequences. Thus, we had 130 template-target assignments for neural network training, 130 for threshold selection, and 140 for final validation.

Description of the alignments

The alignments used in this investigation were built by the SAM (Hughey and Krogh, 1996) HMM software suite. We used two pools of alignments, built with two

slightly different methods. Both methods involve training an HMM on a *training alignment*, and alignment of the template sequence and its homologs, and aligning the target sequence to the model. The methods differ in the choices of training alignment and alignment algorithm.

We used two sets of training alignments, referred to as FSSP and SAM-T99. For the FSSP alignments, the DALI aligner (Holm and Sander, 1997) selected template sequence homologs and aligned the template sequence according to structural similarity. The target sequence was excluded. The SAM-T99 training alignments were built with the SAM-T99 method (Karplus *et al.*, 1998). This method is based on the SAM HMM suite (Hughey and Krogh, 1996), and is available on the world wide web at http://www.soe.ucsc.edu/research/compbio/ under `HMM applications`. Sequences were selected and aligned according to sequence similarity to the template sequence, and include close and moderate homologs of the template. Because the target and template sequences have low sequence similarity, the target sequences did not need to be removed from these alignments.

An HMM is a statistical model with probabilities for transitions between alignment columns and for the matching of residues to a specific column (Krogh *et al.*, 1994). An alignment of a sequence to the HMM reflects a choice of transitions and residue matches evaluated with a dynamic programming algorithm. In the *Viterbi* algorithm, at each decision point, the single most-likely option is chosen. This yields the single most-likely path. This algorithm does not use the full power of the HMM (and dynamic programming) to sum over all possible alignments of the sequence to the model. The *forward–backward* algorithm calculates for each residue and for each alignment column, the probability that that residue is in that alignment column, given all possible alignments. Forward–backward requires substantially more computation than the Viterbi algorithm. In the *posterior decoding* algorithm (Holmes and Durbin, 1998), the Viterbi algorithm is applied to these posterior probabilities to find a most probable alignment to the HMM. Note that this is not necessarily the single most likely alignment. For example, any single (non-optimal) alignment may pick between a number of equally-likely paths in an area of high variation. The posterior decoded alignment will effectively sum this choice over the exponential number of possible alignments and determine which choice is the most favorable overall. While posterior decoding has a higher cost (Wheeler and Hughey, 2000), it often generates more accurate alignments than Viterbi (Holmes and Durbin, 1998). Posterior decoding is a form of near-optimal alignment (Vingron, 1996).

The following methods were used to align the target sequences to the template families.

FSSP-Posdecoding. This method involves models trained on FSSP alignments and posterior-decoded alignment of the target sequences to the models.

SAM-T99-Viterbi. This method involves models trained on SAM-T99 alignments and Viterbi alignment of the target sequences to the models.

In both cases, we used global alignment rather than local alignment. In our experience, the choice between global and local alignment is often a trade-off between over-alignment and misalignment, with local alignment more prone to misalignment and global alignment more prone to over-alignment. We chose to investigate global alignment because the alignments generally contain a greater number of accurate positions, suggesting that the trimmed alignments might be of better quality.

Predictors of column reliability

We explored four predictors of column reliability, described below.

Non-loop distance is the distance in template sequence residues from the column in question to the nearest secondary structural element. When a template sequence residue is in a strand or helix region, the non-loop distance is zero. When a template residue is in a loop, the non-loop distance is the number of template residues to the nearest end of the loop.

Column log odds is the log-likelihood ratio $\log\left(\frac{P(\mathrm{aa}_t|\mathrm{col}_k)}{P(\mathrm{aa}_t)}\right)$, where $P(\mathrm{aa}_t|\mathrm{col}_k)$ is the probability of seeing amino acid type aa_t in alignment column col_k, given the other residues already aligned. $P(\mathrm{aa}_t)$ represents the background probability of amino acid type aa_t. The log likelihood ratio describes how much more likely aa_t is in column col_k than anywhere in any alignment. Both probabilities were computed using Dirichlet mixtures (Sjölander *et al.*, 1996).

Posterior-decoding cost (Yu and Smith, 1999) is a near-optimal alignment measure (Vingron, 1996) reflecting the likelihood that each target residue aligns to each template family column, given all possible alignments of target sequence and template family. Positions kept consistent across many different alignments have lower costs, while those that vary between different alignments have higher costs. The cost is the minus log probability of the alignment position, where the probability is computed from the HMM forward–backward information.

The fourth predictor was a neural network combining the predictors above with other factors influencing column reliability. These factors include location of gaps near the column, alignment column entropy, a log likelihood ratio reflecting the agreement between the template secondary structure and the predicted secondary structure of the target sequence, and the distance in template residues from where the target residue is aligned to where its

posdecoding cost is minimized. The neural network was given this information not merely for the column in question, but for a window centered on the column. This yielded a large set of inputs, which was later pruned down to those found most relevant.

Reducing neural network complexity

Neural networks perform best when presented with a small number of relevant inputs. Thus, when working with neural networks, one should always try to simplify the input set. We used *sensitivity analysis* (Koda, 1997) to identify extraneous inputs. This technique measures the change in training error relative to the change in each input, indicating which inputs might be removed without much impact on network performance.

We performed numerous iterations of sensitivity analysis. In each iteration, we trained a neural network and estimated the sensitivity of the training error to each input. We then selected a feature with low estimated sensitivity, removed that feature for all columns in the window, and retrained the neural network. If performance suffered, the feature was reinstated. Otherwise, we continued with another feature. This process continued until no feature could be removed without compromising network performance.

We then optimized the window size empirically, reducing the width of the window until network performance suffered. Finally, we optimized the network architecture empirically, reducing the number of hidden units.

Note that there are two schools of thought in optimization. They are *top–down*, progressively removing complexity from a complex system; and *bottom–up*, progressively adding complexity to a simple system. We followed a top–down approach in hopes of capturing any indirect relation between the features.

Experimental design

We developed our system on 400 pairs of distantly-related template and target sequences. For each sequence pair, the target sequence was aligned to the template family with each of the alignment methods described previously.

First, for each alignment method, we trained neural networks to predict alignment position reliability using the 130 training alignments. These 130 alignments contained approximately 20 000 target residue alignment positions, or 20 000 data points. These 20 000 data points were divided at random into training and cross-training sets, with the cross-training set used to adjust algorithmic parameters during training. All neural network optimization involved these same 20 000 data points.

Next, we used the 130 threshold selection alignments to empirically derive reliability thresholds for each of the four predictors. We applied each predictor to each alignment, and experimented with a range of reliability thresholds, removing positions that fell beyond the threshold value. We then scored these trimmed alignments against structural alignments, and selected one threshold value according to the shift score. Table 1 shows these thresholds, and compares the shift scores of the trimmed global alignments to shift scores of untrimmed global alignments and local alignments built with the same alignment method.

Finally, we applied each predictor and threshold value to the 140 validation alignments. We measured trimming performance by removing alignment positions predicted as unreliable, and scoring the trimmed alignments against structural alignments.

ALIGNMENT SCORING

Before one can evaluate an alignment trimming method, one needs a scoring system to indicate if trimming improved an alignment. This work required an alignment quality measure with certain characteristics. First, it should be meaningful, with a strong score implying a strong alignment. Second, it should incorporate penalties for the many types of errors seen in alignments: aligning too much (*over-aligning*), aligning too little (*under-aligning*), and misaligning. Third, it should be optimizable, such that removing any positions that detract from the score improves the alignment. Most alignment quality measures do not meet these criteria.

Many alignment quality measures must be paired with a second measure to be meaningful. For example, *RMS Deviation (RMSD)* or *Mean Shift Error (MSE)* are useful only with a measure of alignment length, as both reflect alignment accuracy but not extent. An alignment that aligns only one residue, but aligns it accurately, would have an excellent RMSD and MSE, but would probably not be useful.

Another example is *alignment specificity* and *alignment sensitivity*, the fraction of residues aligned correctly as a proportion of the lengths of the predicted and structural alignments, respectively. *Specificity* does not reflect under-alignment, and would give a great score to an alignment that aligns only one residue but aligns it correctly. *Sensitivity* does not reflect over-alignment, and would give a great score to an alignment that correctly aligns one domain of two multi-domain proteins but incorrectly aligns additional domains. In practice (Marchler-Bauer *et al.*, 1997; Briffeuil *et al.*, 1998; Sauder *et al.*, 2000), *specificity* and *sensitivity* are reported together, and overall assessment involves a subjective weighting of the two. This becomes awkward for large-scale experiments.

Furthermore, most measures do not lend themselves to *alignment optimization*, identifying the portion of the alignment which yields the best score. Most measures are optimized by removing any position not aligned correctly. Consider an alignment that correctly aligns a

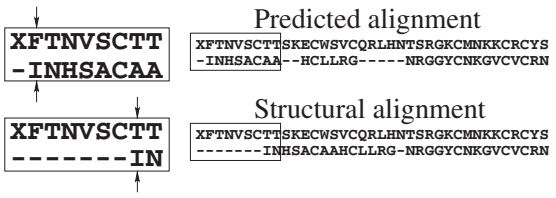

Fig. 2. Illustration of alignment shift. Here, the Isoleucine is shifted by six positions. The shift of the Phenylalanine is undefined.

short motif and misaligns everything else by one position. If optimized according to most measures, all of the alignment would be discarded except for the short motif, losing a lot of arguably useful information.

Alignment shift scores

Definition. Because no accepted alignment quality measure met our criteria, we invented our own. This measure, referred to as the *shift score*, is based on alignment *shift* information. Shift is a measure of misalignment or disagreement between two alignments: if some residue a_i is aligned to residue b_j in alignment X and residue b_k in alignment Y, $shift(a_i)$ is the number of residues from b_j to b_k. If residue a_i is not aligned to another residue in either case, $shift(a_i)$ is undefined (Figure 2).

For some pair of sequences A and B aligned in predicted alignment X and in structural alignment Y, the *shift score* $SS_Y(X)$ is

$$SS_Y(X) = \frac{\sum_{i=1}^{|X|} cs_Y(X_i)}{|X| + |Y|}, \text{ where}$$

$$|X| = \text{Number of aligned residue pairs in } X$$

$$cs_Y(X_i) = \text{Score for column } X_i \text{ in alignment } X$$

$$= \begin{cases} s(a_j) + s(b_k) & \text{if } X_i \text{ aligns some residues } a_j \text{ and } b_k \\ 0 & \text{otherwise} \end{cases}$$

$$s_Y(r_i) = \text{Score for residue } r_i \text{ with respect to structural alignment } Y$$

$$= \begin{cases} \dfrac{1+\epsilon}{1+|\text{shift}(r_i)|} - \epsilon & \text{if shift}(r_i) \text{ is defined} \\ 0 & \text{otherwise} \end{cases}$$

$$\epsilon = \text{scoring parameter, normally set to 0.2}$$

$s_Y(r_i)$ ranges from $-\epsilon$ to 1.0. For accurate positions, they are 1.0. For small shifts, they are positive, dropping to zero with shifts of five $(1/\epsilon)$. For large shifts, they approach $-\epsilon$. The $cs_Y(X_i)$ terms are 2.0 for columns aligned correctly, between 2.0 and -2ϵ for columns containing misaligned residues, and 0 for *over-aligned* columns: those that were not aligned in the structural alignment. The shift score ranges between $-\epsilon$ and 1.0. If the two alignments are identical, their shift score is 1.0.

If the predicted alignment X is largely misaligned, its shift score will be low because the $s_Y(r_i)$ terms in the numerator will be small. If X aligns too much, the shift score will be small because of the large $|X|$ in the denominator with few nonzero $s_Y(r_i)$ terms in the numerator. If X aligns too little, then the $|Y|$ term in the denominator will be large, with few nonzero terms in the numerator. Thus, the shift score incorporates penalties for misalignment, aligning too much, and aligning too little.

Validation

We validated the shift score by comparing it with the standard measures of alignment quality, *alignment specificity* and *alignment sensitivity*. In a study of 325 pairs of predicted and structural alignments, the shift score compared very well to alignment specificity and alignment sensitivity (Figure 3). In cases where the scores disagreed, arguments could be made for the shift score. In Figure 3, the upper circled outlier has a high shift score of 0.447 but low sensitivity and specificity of 11.3 and 19.1 respectively. It aligned a small number of residues correctly, yielding its low specificity and sensitivity. However, it misaligned many residues by only one position, accounting for its high shift score. The other circled outlier has a high alignment sensitivity of 72.6, an alignment specificity of 30.6 and a low shift score of 0.144. It aligned many residues correctly but included too much: a global alignment where a domain hit was correct. In addition, it included a number of substantially-misaligned positions. The misalignment, plus the over-aligned positions, account for the low shift score. Other outliers with lower shift scores than expected from alignment sensitivity are also a result of overaligning.

Optimization

The shift score can be optimized by the following simple greedy algorithm:

(1) Compute the shift score of predicted alignment X.

(2) Remove from X column X_m, the column with the worst column score $cs_Y(X_m)$.

(3) Compute the shift score for revised alignment X.

(4) If the score has not worsened, return to step 2.

(5) Reinstate X_m, the last column removed.

This algorithm produces a version of predicted alignment X with the most inaccurate regions removed. It will retain all accurate positions and positions with small shifts, and will remove all over-aligned columns. With an ϵ of 0.2, it will remove all positions shifted by 5 or more residues. Other positions might be removed or retained according to the overall alignment accuracy. In an accurate alignment, positions shifted by 3 or 4 residues will usually be

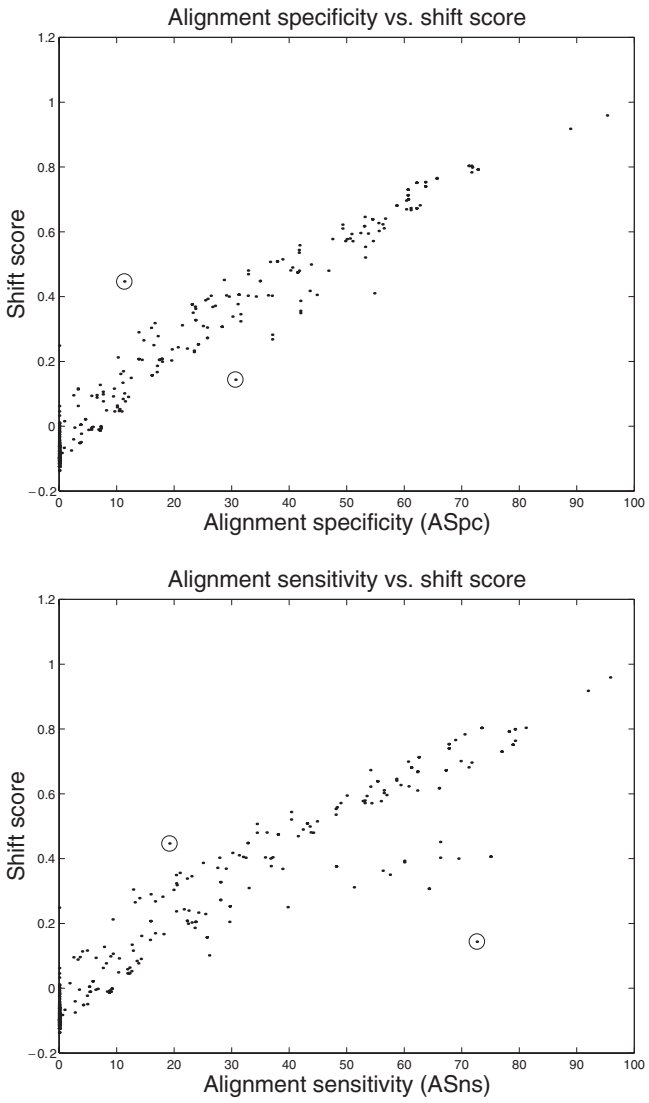

Fig. 3. Comparison of the shift score to alignment specificity (top) and alignment sensitivity (bottom). Circles indicate the two outliers discussed in the text.

removed. In a mostly inaccurate alignment, such positions will be retained. These optimized alignments are referred to as *optimal subalignments*. An alignment's optimal subalignment score indicates how much the alignment could be improved by removing only the worst positions, assuming prior knowledge of exactly which positions to remove.

Finally, we scored the predicted alignments against a panel of three structural alignments: DALI (Holm and Sander, 1997), VAST (Gilbrat *et al.*, 1996), and the Yale aligner (Gerstein and Levitt, 1998). While comparisons to one set of structural alignments might favor a method somehow related to the one structural aligner, comparisons to three structural aligners reduce this risk.

RESULTS

The underlying goal of this work was to identify factors that suggest accuracy in sequence alignments. Toward this end, we experimented with a set of methods to predict the unreliable regions in alignments built with two different HMM methods: FSSP-posdecoding and Target99-Viterbi. We conducted two sets of experiments in parallel, one for each of the two alignment methods, and in this section we compare and contrast the results.

Neural network optimization

As described in the Section Methods, we optimized the neural networks in a top–down fashion, beginning with a complex architecture and gradually removing complexity. We optimized the input feature set first, the window size second, and the network architecture last.

For FSSP-posdecoding, the only features that proved worthwhile were posterior decoding cost, column entropy, locations of gaps within the window, and predicted secondary structure similarity. For Target99-Viterbi, the features that proved worthwhile were the same features plus the distance from where each target residue was aligned to where its posterior decoding cost was minimized. Features that did not prove worthwhile in either case include the column log odds, the non-loop distance, and the probability that the target sequence was in a loop region. This does not necessarily mean these features are of no value. More likely, it indicates that other features conveyed similar information, but were slightly more informative.

Selecting prediction thresholds

For each predictor, we selected a trimming threshold empirically by experimenting with a range of possible threshold values, removing positions that fell outside each threshold, and scoring these trimmed alignments against structural alignments of the same sequences.

Table 1 shows that all predictors improved on the FSSP-posdecoding alignments. Improvements were realized by removing positions with a column log odds of less than -2.7, a posterior decoding column cost of greater than 1.9, or a neural network reliability prediction of less than 20%. All of these represent removing only those positions of very low confidence. For the non-loop distance, we found our best results by removing alignment positions where the number of template residues to the nearest secondary structural element was more than five: removing the middle of very long loops.

For Target99-Viterbi alignments, trimming by column log odds was not effective. Compared to the FSSP-posdecoding alignments, these are noisier, and contain so many low-probability positions that any threshold value removed large portions of the alignment. While the remaining positions might be accurate, the alignments were

Table 1. For each predictor, this table describes the threshold values that yielded best results for the threshold selection alignments. Local alignment scores are shown for comparison. On FSSP-posdecoded alignments (top), we realized minor improvements with non-loop distance and column log odds, and greater improvements with the posterior decoding (*posdecoding*) cost or the neural network prediction. On Target99-Viterbi alignments (bottom), we realized improvement by trimming according to the non-loop distance, the posdecoding cost, or the neural network prediction. For column log odds, we did not find a threshold value that improved the Target99-Viterbi alignments

Predictor	Shift score	Threshold
Positions removed: FSSP-posdecoding		
None	0.402	–
Local alignment	0.366	–
Non-loop distance	0.402	>5
Column log odds	0.405	<−2.7
Posdecoding cost	0.417	>1.9
Neural network	0.418	<0.20
Optimal	0.533	–
Positions removed: Target99-Viterbi		
None	0.259	–
Local alignment	0.251	–
Non-loop distance	0.262	>3
Column log odds	–	None
Posdecoding cost	0.287	>0.7
Neural network	0.294	<0.15
Optimal	0.408	–

too short to yield good shift scores. For the other three predictors, we see in Table 1 that the best thresholds found were lower than those for FSSP-posdecoding alignments. For non-loop distance and posterior decoding cost, this represented removing a greater number of positions. For the neural network, approximately the same proportion of positions were removed, and the lower threshold merely indicates a greater bias by the neural network to predict everything as unreliable.

Validation results

Using the thresholds derived in the previous section, we applied the predictors to a library of 130 validation alignments. As described in the Section Experimental design, these alignments were not used during neural network training or threshold selection.

For Target99-Viterbi alignments, we found no effective threshold for column log odds, and therefore did not apply it to the validation alignments. The remaining predictors were applied to each alignment in the validation set, and positions with predictions beyond the threshold values were removed. We then scored each trimmed alignment against structural alignments from the three aligners: DALI, VAST, and Yale. Table 2 shows the average shift scores.

Table 2 shows that all methods improved on the FSSP-posdecoding alignments. The best performance was achieved by the neural network, with posterior decoding cost performing nearly as well. These methods yield an improvement of 1.9–3.5%. While this might sound modest, FSSP-posdecoding is our best alignment method, so any improvement is noteworthy. For Target99-Viterbi alignments, the non-loop threshold was not effective: the improvement found during threshold selection did not hold through validation. However, the other two methods yielded improvement, with the neural network improving the alignments by 13.5–20.1%. In both cases, the neural network achieved the best performance, with posterior decoding cost performing nearly as well. We selected these two predictors for further analysis.

Table 2. Results on trimming the 140 validation alignments according to the various predictors of alignment reliability. The best results in each category are shown in boldface, and local alignment results are shown for comparison

Trimming method	Shift score		
	DALI	VAST	Yale
FSSP-posdecoding			
None	0.369	0.365	0.334
Local alignment	0.311	0.309	0.289
Non-loop threshold	0.373	0.368	0.337
Column log odds	0.371	0.367	0.336
Posdecoding cost	0.377	0.371	0.340
Neural network	**0.382**	**0.372**	**0.344**
Optimal	0.512	0.464	0.437
Target99-Viterbi			
None	0.193	0.207	0.206
Local alignment	0.178	0.188	0.189
Non-loop threshold	0.193	0.206	0.206
Posdecoding cost	0.222	0.225	0.230
Neural network	**0.233**	**0.235**	**0.236**
Optimal	0.352	0.330	0.330

Table 3. Analysis of what positions are retained after the validation alignments are trimmed. Alignment positions were divided into the following categories according to DALI's structural alignments: *accurate*; *over-aligned*, aligning portions of the sequences that exhibit no structural similarity; and *badly misaligned*, shifted by five or more residues

Category	None	Posdecoding cost	Neural network
Retained by trimming method: FSSP-posdecoding			
Total	24 802	19 514	16 161
Accurate	8 987	8 270	7 700
Over-aligned	8 972	4 277	3 064
Badly misaligned	5 661	3 745	2 725
Retained by trimming method: Target99-Viterbi			
Total	27 101	13 317	13 955
Accurate	6 121	5 264	5 460
Over-aligned	8 522	3 566	3 418
Badly misaligned	9 701	2 941	3 400

To explore which positions were removed, we analyzed the validation alignments relative to DALI's structural alignments. We focused on three categories of alignment positions: aligned accurately; over-aligned, aligning regions of the template and target sequences with no structural similarity; and misaligned by five or more residues. We counted the number of positions in each category before and after trimming (Table 3).

Two points are evident from Table 3. First, both methods are doing something right. They remove as much as 70% of the badly misaligned positions and 50–65% of the over-aligned positions while retaining more than 85% of the accurate positions. Second, the neural network does not yield much performance gain over posterior decoding cost, which had been its strongest feature. In other words, the neural network gets most of its information from posdecoding cost, and the additional complexity buys little added performance in practice. Where the two methods appear to yield different results, much of the distinction might come from the differences in their threshold values. Because posterior decoding column cost is far simpler, it is the superior predictor.

DISCUSSION

Alignment methods make mistakes. However, some of the mistakes can be identified. We explored four methods for identifying suspect alignment positions, applying them to hard remote homologs. When we removed the suspect positions, we saw substantial improvement in overall alignment quality.

The stronger predictors both focus on posterior decoding information. Posterior decoding is a form of near-optimal alignment algorithm (Vingron, 1996), algorithms that estimate alignment information according to various alignments of the sequences, emphasizing the positions that are consistent across different alignments.

One predictor was the *posterior decoding (posdecoding) cost*, the negative log probability of each target residue aligning to each template family column, with this probability estimated according to all possible alignments of the target sequence and template family. A simple threshold on this cost yielded a 3% improvement over one of our best alignment methods and a 15% improvement over a more-general method. The second predictor was a neural network trained to predict alignment position reliability given this cost plus other information. It performed slightly better than the posdecoding cost. In analysis of the positions retained after trimming, we saw that these two predictors removed 65–70% of the columns misaligned by more than a few residues and 60% of the over-aligned positions while preserving 85 and 89% of the accurate positions.

When comparing these last two predictors, the extra complexity in the neural network yielded modest performance gain. Posdecoding cost performed nearly as well, and is far simpler. Therefore, we consider posdecoding cost to be the best of the four predictors. When developing the neural network, we followed an optimization scheme, reducing the number of inputs and simplifying the architecture so long as these modifications yielded at least a slight performance gain. Ironically, a slight performance gain was exactly what we got! The lesson here is to justify complexity according to not merely what yields more information, but also according to when that information merits the additional software, late nights, and caffeine that its creation and use might require.

ACKNOWLEDGEMENTS

This paper represents the contributions of a number of people, including Spencer Tu, Mark Diekhans, and David Haussler. This work was supported in part by NSF grant DBI-9808007, DOE grant DE-FG03-99ER62849, and the GAANN fellowship program.

REFERENCES

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

Briffeuil,P., Baudoux,G., Lambert,C., Bolle,X.D., Vinals,C., Feytmans,E. and Depiereux,E. (1998) Comparative analysis of seven multiple protein sequence alignment servers: clues to enhance reliability of predictions. *Bioinformatics*, **14**, 357–366.

Dopazo,J. (1997) A new index to find regions showing an unexpected variability or conservation in sequence alignments. *CABIOS*, **13**, 313–317.

Gerstein,M. and Levitt,M. (1998) Comprehensive assessment of automatic structural alignment against a manual standard, the SCOP classification of proteins. *Protein Sci.*, **7**, 445–456.

Gilbrat,J., Madej,T. and Bryant,S. (1996) Surprising similarities in structure comparison. *Curr. Opin. Struct. Biol.*, **6**, 377–385.

Holm,L. and Sander,C. (1997) Dali/FSSP classification of three-dimensional protein folds. *NAR*, **25**, 231–234.

Holmes,I. and Durbin,R. (1998) Dynamic programming alignment accuracy. *J. Comput. Biol.*, **5**, 493–504.

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.

Jones,D.T. (1997) Progress in protein structure prediction. *Curr. Opin. Struct. Biol.*, **7**, 377–387.

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

Koda,M. (1997) Neural network learning based on stochastic sensitivity analysis. *IEEE Trans. Syst. Man Cybern.* B *(Cybernetics)*, **27**, 132–135.

Krogh,A., Brown,M., Mian,I.S., Sjölander,K. and Haussler,D. (1994) Hidden Markov models in computational biology: applications to protein modeling. *JMB*, **235**, 1501–1531.

Marchler-Bauer,A., Levitt,M. and Bryant,S. (1997) A retrospective analysis of CASP2 threading predictions. *Proteins Struct. Funct. Genet.*, **1** (Suppl.), 83–91.

Mevissen,H.T. and Vingron,M. (1996) Quantifying the local reliability of a sequence alignment. *Protein Eng.*, **9**, 127–132.

Miyazawa,S. (1994) A reliable sequence alignment method based on probabilities of residue correspondences. *Protein Eng.*, **8**, 999–1009.

Murzin,A.G. (1999) Structure classification-based assessment of CASP3 predictions for the fold recognition targets. *Proteins Struct. Funct. Genet.*, **3** (Suppl.), 88–103.

Pearson,W. and Lipman,D. (1988) Improved tools for biological sequence comparison. *PNAS*, **85**, 2444–2448.

Rost,B. (1999) Twilight zone of protein sequence alignments. *Protein Eng.*, **12**, 85–94.

Sauder,S.M., Arthur,J.W. and Dunbrack,R.L.D. Jr (2000) Large-scale comparison of protein sequence alignment algorithms with structural alignments. *Proteins Struct. Funct. Genet.*, **40**, 6–22.

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.

Vingron,M. (1996) Near-optimal sequence alignment. *Curr. Opin. Struct. Biol.*, **6**, 346–352.

Vingron,M. and Argos,P. (1990) Determination of reliable regions in protein sequence alignments. *Protein Eng.*, **3**, 565–569.

Wheeler,R. and Hughey,R. (2000) Optimizing reduced space sequence alignment. *Bioinformatics*, **16**, 1082–1090.

Yu,L. and Smith,T. (1999) Positional statistical significance in sequence alignment. *J. Comput. Biol.*, **6**, 253–259.

Zhang,Z., Berman,P., Wiehe,T. and Miller,W. (1999) Post-processing long pairwise alignments. *Bioinformatics*, **15**, 1012–1019.

Zhang,M.Q. and Marr,T.G. (1995) Alignment of molecular sequences seen as random path analysis. *J. Theor. Biol.*, **174**, 119–129.