doi:10.1016/j.jmb.2004.03.023 J. Mol. Biol. (2004) 338, 847–854

JMB

Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

# Improving Profile HMM Discrimination by Adapting Transition Probabilities

**Markus Wistrand and Erik L. L. Sonnhammer***

*Center for Genomics and Bioinformatics, Karolinska Institutet, S-17177 Stockholm Sweden*

Profile hidden Markov models (HMMs) are used to model protein families and for detecting evolutionary relationships between proteins. Such a profile HMM is typically constructed from a multiple alignment of a set of related sequences. Transition probability parameters in an HMM are used to model insertions and deletions in the alignment. We show here that taking into account unrelated sequences when estimating the transition probability parameters helps to construct more discriminative models for the global/local alignment mode. After normal HMM training, a simple heuristic is employed that adjusts the transition probabilities between match and delete states according to observed transitions in the training set relative to the unrelated (noise) set. The method is called adaptive transition probabilities (ATP) and is based on the HMMER package implementation. It was benchmarked in two remote homology tests based on the Pfam and the SCOP classifications. Compared to the HMMER default procedure, the rate of misclassification was reduced significantly in both tests and across all levels of error rate.



*Corresponding author

*Keywords:* profile hidden Markov models; adaptive transition probabilities

## Introduction

Detecting protein relationships is central in analysis of biological sequences. Often, characteristics known to be true for one class of proteins can be transferred to others that have similar structure or sequence as detected by computational methods. The more sequences have diverged, the harder it becomes to recognize true homology, and at some point it becomes impossible. A lot of effort has therefore been focused on improving detection of remote homologs. Experiments have shown that profile-based methods that draw on the characteristics of many homologs in the search for new ones do better than methods that look only for pairwise homology.[1] Among the profile-based methods, profile hidden Markov models (profile HMMs)[2–4] have been shown to perform best.[5,6]

Since profile HMMs were first introduced for modeling multiple sequence alignments, considerable refinements have been carried out on different aspects of building the HMMs. Improvements have, for example, been done to weighting schemes,[7] null models,[8] prior emission probabilities[9,10] and modeling strategies.[5,6] Database libraries of HMMs[6,11–13] have been developed to aid annotation of genomes and to study relationships between sequences and protein families. These databases all use either of two widespread profile HMM software packages: SAM[3,5] and HMMER.[4]

An HMM, as used in these packages, is a probabilistic model of a protein family. Figure 1 outlines the profile HMM architecture used by HMMER. The structure is repetitive and a set of match, insert and delete states (a node in HMMER jargon) is used to model the columns of a multiple alignment of the sequences. Match and insert states are associated with probabilities for emitting each amino acid, and transitions between states are associated with transition probabilities. These are posterior probabilities that are estimated by combining data from the alignment with prior probabilities. A consensus column (where all or most sequences have a residue) in the alignment forms a match/delete state while non-consensus columns are assigned to insert states in the model. Prior probabilities basically add extra counts to the observations in the alignment, which helps to

Abbreviations used: HMM, hidden Markov model; ATP, adaptive transition probabilities; SVM, support vector machine; MER, minimum error rate; OTN, over top noise; ML, maximum-likelihood.

E-mail address of the corresponding author: erik.sonnhammer@cgb.ki.se

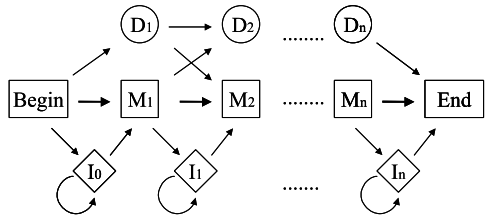


**Figure 1**. Overview of the "plan 7" architecture of HMMER 2.3.1. D stands for delete state, M for match state and I for insert state. Arrows indicate the allowed transitions between states.

avoid zero probabilities caused by unobserved amino acid residues, and increases model generality and sensitivity by learning about unseen events.

The standard HMM building procedure is to estimate the model from a set of related sequences (positives) so as to maximize the probability of those sequences given the model. Such a procedure focuses entirely on sensitivity, i.e. the ability to detect all related sequences and thereby avoid false negatives. However, it is also desirable not to detect sequences that are unrelated to the model, so-called false positives. In practice, this normally entails specifying a score cut-off that separates members from non-members. One could argue that HMM parameter estimation should not focus only on modeling the positive training sequences optimally, but rather on maximizing the separation of members from noise. This requires unrelated (negative) training examples in addition to the positive ones, to teach the HMM to discriminate between them. It also requires a training algorithm that is bound to be slower than the standard profile HMM estimation, because it is using more training data.

Suggestions of this kind can be found in the literature. Mamitsuka[14] proposed a method for HMM estimation using both positive and negative training sequences. In classification experiments, this algorithm does better than similar algorithms trained on only positive sequences. However, the results are limited to modeling of only the motif region of one protein family and the method cannot compete with standard profile HMM estimation in speed. Yet, it is interesting to see how the negative sequences add information to the HMM.

Another approach was taken by Eddy *et al*., who proposed a method for better discrimination, although training is done only on positive sequences.[15] This algorithm iteratively updates sequence weights so that most weight is given to highly divergent training sequences, and does not explicitly tune internal state HMM parameters to optimize the discrimination. HMMs have been employed with support vector machines (SVMs), a machine learning technique for classification. Jaakkola *et al*. presented an SVM algorithm for protein sequence classification where the so-called kernel function used was derived from HMMs of protein families.[16] Both positive and negative training sequences belonging to appropriately chosen families were used in the training process. In a large-scale test on G-protein coupled receptors,[17] the authors concluded that SVMs are superior at detecting subtle differences between related sequence families, but perform worse than HMMs in remote homology detection.

Here, we propose a way to include negative training examples for estimating profile HMMs. Our algorithm uses a heuristic derived for the transition probabilities of the model and adjusts these for better discrimination. The heuristic was designed to be well-behaved and robust. It has one weight parameter that was tested over a range of values. We compared the method to the default HMMER 2.3.1 package using two different remote homology detection test sets (based on Pfam or SCOP) and obtained consistently better discrimination at the cost of a moderate decrease in speed.

Our algorithm (called ATP for adaptive transition probabilities) was developed as an extension of the freely available code from the HMMER software package. When we say ATP, we thus mean HMMER + ATP, where ATP is a kind of "post-processing" step. Briefly described, a standard profile HMM is first estimated from a multiple sequence alignment of training sequences. Then a number of noise sequences are aligned to the HMM, and those scoring highest are considered further. On the basis of differences in the paths that the positive and negative sequences follow through the model, the main transition probability parameters are adjusted. The idea is that if positive and negative sequences use the same paths at a particular node in the model, then the transition probabilities are left unchanged. If not, the transition probabilities are adjusted to better fit the positive sequences (and the negative sequences less well).

## Data

Two tests based on detection of remote homologs were used to evaluate the performance of ATP in comparison to default HMMER: one derived from the Pfam database[11] and one from the SCOP database.[18] SCOP is a database classifying all protein sequences having a known structure based on structure, function and sequence. The unit of classification is the domain; multi-domain proteins are split into their domain components. The classification is hierarchical and in four levels: class, fold, superfamily and family. Sequences belonging to the same family are clearly similar by sequence, suggesting a common evolutionary origin and similar function. Families related to each other with a low level of sequence similarity but with structural evidence for a common origin are grouped into superfamilies. At the next level, fold, superfamilies having the same overall secondary structure topology are grouped together, although

no firm evidence of being evolutionarily related exists. In this study we are interested in homology detection at the superfamily level.

The ASTRAL database[19] provides sequences corresponding to the SCOP domain classification filtered to different levels of sequence similarity. We took the ASTRAL 1.63 release filtered to 90% residue identity. All instances of a family that had at least ten sequences and for which the remaining families in the superfamily had at least five but no more than 50 sequences were considered. An HMM was constructed from the family sequences, and the remaining sequences in the superfamily were used as positive test sequences. Sequences belonging to any fold other than the training and test sequences were used as negative test examples. This procedure produced 70 training families. The number of training sequences ranged from ten to 62, the number of positive test sequences from five to 48, and the number of negative test sequences from 7129 to 8025. Altogether there were 1365 positive and 555,509 negative test sequences.

In Pfam, protein domains are classified into families on the basis of sequence similarity and evidence of evolutionary origin and function. While SCOP is biased to globular proteins for which the structure is known, Pfam should have a composition that better represents biological reality. Pfam also provides manually edited alignments of the seed sequences in each family, which is an advantage when building high-quality HMMs. The construction of the Pfam-based test set for our benchmark has been described.[20] In brief, we used the seed sequences from the Pfam 7.0 release. Families that could be divided into two subfamilies with less than 20% sequence identity between them were divided in this fashion. The larger subfamily was used as training set and the remaining subfamily as positive test sequences. For Pfam, it is not safe to use sequences from other families as negative test sequences, since many of the Pfam families are related and cannot be separated by a criterion such as different fold in SCOP. We instead used reversed real sequences as negative test examples. These were fetched randomly from Swiss-Prot and are of similar length as the training sequences.

The Pfam test set consisted of 373 families. The number of training sequences per family ranged from three to 437 (mean of 34) and the number of positive test sequences from one to 34 (mean of 3.7). The number of negative test sequences was 5000 for each family, except for families with sequences of length close to 1000 amino acid residues, where 2874 negative examples were used. In total, this produced 1378 positive and 1,860,748 negative test sequences.

### Assessment procedure

The comparison of the ATP algorithm to default HMMER employed the same methodology for both the SCOP and the Pfam benchmark test. A profile HMM was constructed for each family from the training sequences. The corresponding positive and negative test sequences were matched to the HMM and sorted in a list from best scoring match to worst. Ideally, such a list should be divided into two parts with all positive test sequences first, followed by the negative test sequences. However, for highly divergent families this is often not the case. From the rank list, a cut-off can be defined to minimize the number of misclassifications, i.e. to get as few false positives and false negatives as possible. We add up these numbers for all families in the test to get the minimum error rate (MER). This is a measure of the lowest number of misclassifications one would get if allowed to choose an optimal cut-off for each family. As a complementary measure, we give the number of false positives that are detected above the highest-scoring negative sequence (over top noise, OTN).

In large-scale annotation efforts it is generally not possible to employ an MER-optimized cut-off for each sequence family being modeled. The MER value is therefore a somewhat artificial measure. A more revealing way of comparing methods is by plotting the number of true matches *versus* false matches, similar to a coverage *versus* error plot described by Brenner *et al*.[21] All family rank lists (corresponding to all HMMs) are merged and the resulting total list is sorted by *E*-value. A sequence's *E*-value is the number of sequences that would be expected by chance to score as high as this sequence in a database of a specified size. From the sorted list, the number of false positives is plotted *versus* the number of false negatives. Compared to the MER, this gives a better overall picture of how one method fares in comparison to another.

Two sources of irreproducibility arise in our analysis. First, HMMER calculates *E*-values empirically from randomly generated sequences. The rank list of sequences for a particular HMM will not change from one run to another but, when all lists are merged and sorted on *E*-values, the sorted list will not be completely reproducible. Second, ATP generates negative training sequences randomly and therefore the training will be slightly different from time to time. We handled the problem of reproducibility by plotting three coverage *versus* error curves corresponding to three independent benchmark runs. This illustrates the variability in the results that can be expected. For the MER values, we calculated the mean value from the three runs.

## Results

When comparing the MER values for ATP and default HMMER, we found that ATP overall produced a lower MER value (Table 1). The difference is clear for both the Pfam and the SCOP test. ATP

**Table 1.** Comparing ATP to default HMMER

| | Pfam test (373 families) | | SCOP test (70 families) | |
|---|---|---|---|---|
| | ATP | HMMER Default | ATP | HMMER default |
| MER | | | | |
| All | 346 | 389 | 1131 | 1146 |
| Number of families better modeled | 25 | 9 | 12 | 3 |
| OTN | | | | |
| All | 1000 | 966 | 220 | 203 |
| Number of families better modeled | 23 | 9 | 11 | 4 |

The aggregated result for all families and the number of families that one method models better than the other is shown. A low minimum error rate (MER) and a high over top noise (OTN) are desired. Mean values from three runs are shown.

detects more sequences above the highest-scoring noise sequence (OTN).

This overall discrimination performance could possibly be caused by a big improvement in a large family at the expense of worsening the performance for other, smaller families. To investigate whether this was the case, we compared the MER and OTN values of individual families, and counted how many were better modeled by one or the other method. Although we found that some families did achieve better discrimination by default HMMER, ATP performed better for most families (Table 1). This shows that the ATP improvement is not the result of overfitting to one or a few families with many test sequences.

The same outcome was observed when the number of true positives was plotted against the number of false positives. We have chosen to restrict these plots to less than 1000 false positives found. ATP performs better than default HMMER across all error rates; clearly so in the Pfam test (Figure 2) and it is slightly better in the SCOP test (Figure 3).

The ATP procedure requires some extra computer time: estimating the models for the 70 HMMs in the SCOP test took 56 seconds using default HMMER and 337 seconds using the ATP procedure. Including the time it takes to calibrate the models, necessary to get proper *E*-values, ATP is about 1.55 times slower than HMMER (801 compared to 516 seconds).

To illustrate the effect of ATP, we use an example where ATP performs particularly well in comparison to default HMMER: the Pfam family pyridoxal phosphate-dependent enzyme (PF00291). The seed alignment contained 98 sequences, which were split into a training set of 91 sequences and a positive test set of seven sequences: 5000 reversed real sequences acted as noise sequences. Figure 4 shows that an HMM built by HMMER from the 91 training sequences discriminates the positive test sequences poorly from the noise: only three test sequences get higher scores (lower *E*-values) than the top-scoring noise sequence. However, if the HMM is processed by ATP, all seven homologs get higher scores (lower *E*-values) than the best scoring noise sequence.

## Discussion

We have described results using a heuristic to improve profile HMM models for homology detection. The heuristic adjusts the transition probabilities of a profile HMM taking top-scoring random sequences into account. Apparently, these can reveal weaknesses in the model that allow sequences unrelated to the sequence family to get through the model with relatively high scores. The improved performance of ATP was verified in two different remote homology detection tests based

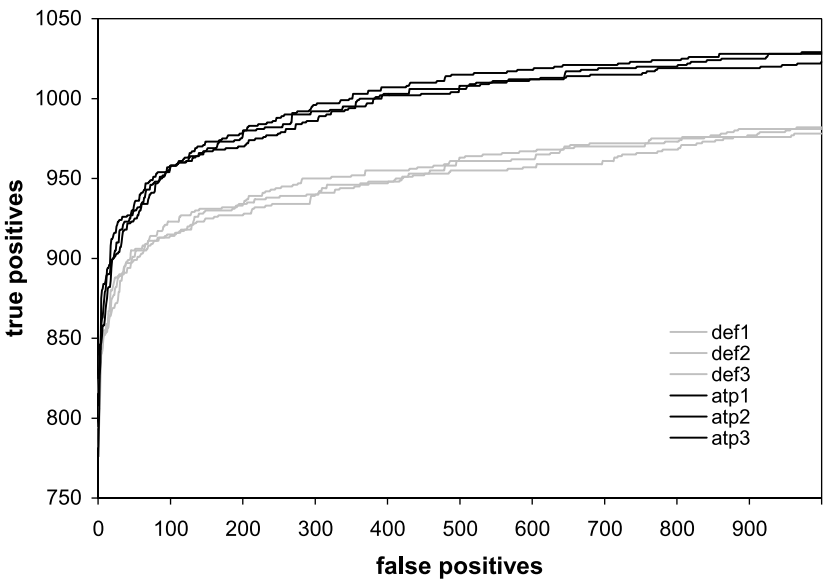


**Figure 2**. Coverage *versus* error plots for the Pfam test sampling. atp stands for ATP and def for the HMMER 2.3.1 default procedure. Curves from three separate runs are shown because the plots are not completely reproducible. The total number of positive and negative sequences were 1378 and 1,860,748, respectively.

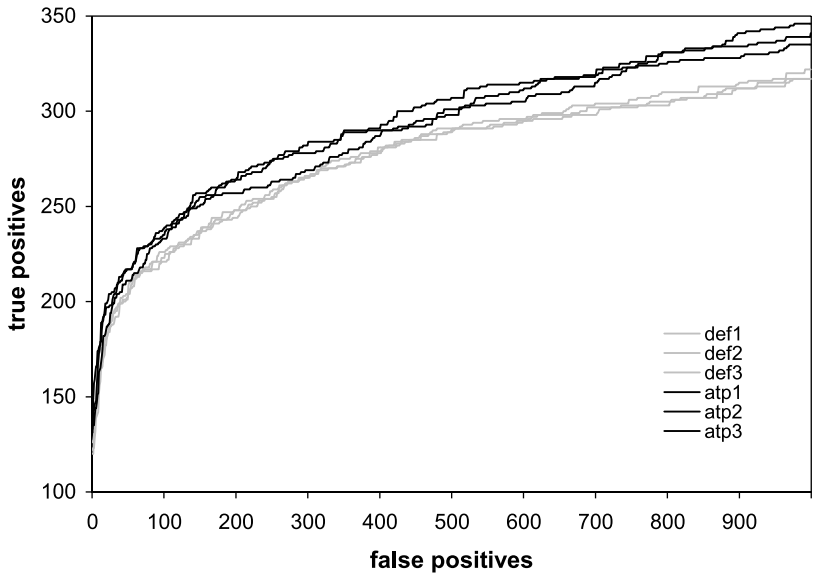


**Figure 3**. Coverage *versus* error plots for the SCOP test. atp stands for ATP and def for the HMMER 2.3.1 default procedure. Curves from three separate runs are shown because the plots are not completely reproducible. The total number of positive and negative sequences were 1365 and 555,509, respectively.

on the Pfam and the SCOP classifications, respectively.

Profile HMMs are statistical models of a multiple sequence alignment. Emission and transition probability parameters are normally estimated by combining the observed data in an alignment with prior probabilities. The results here suggest that this may not be the optimal way to estimate the transition probabilities if we want optimally discriminating models. ATP does not try to optimize the likelihood of all positive sequences but instead tries to improve discrimination. As this is done by reducing the probability of transitions that are exploited by noise sequences, normally the scores become somewhat reduced. However, the absolute value of the score is not relevant for measuring discrimination. Furthermore, the reduction in absolute score value is normalized when using *E*-values, which leaves the negative sequences essentially unchanged, while the positive sequences get lower *E*-values, as illustrated in Figure 4.

ATP uses a heuristic that is only one out of many that are plausible. It may be of interest to the reader that we first implemented a more rigorous method, using the Forward algorithm. The

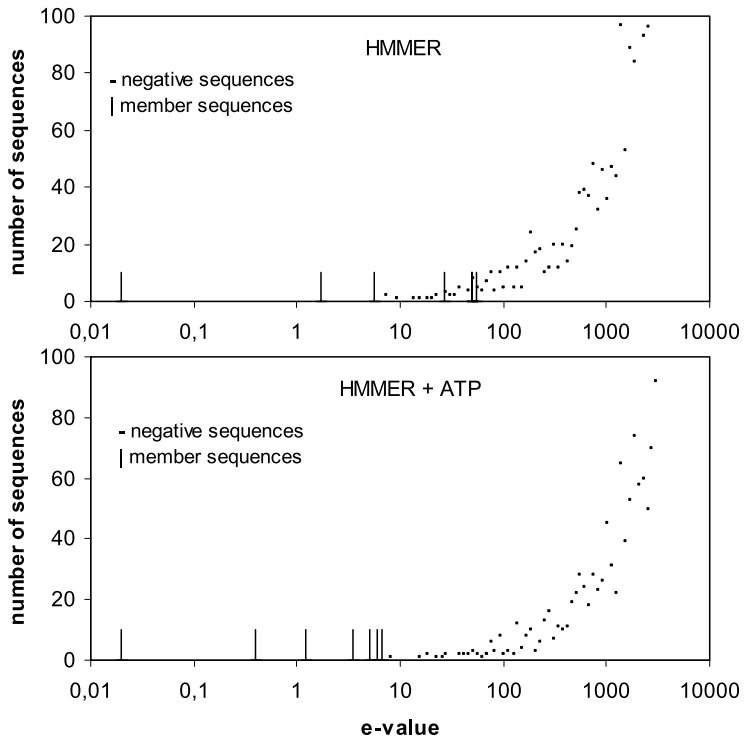


**Figure 4**. Example of how ATP can improve discrimination. The seed alignment for Pfam family PF00291 was split into training and test sequences, and an HMM was built from the training sequences using HMMER (results in top panel) or HMMER followed by ATP (results in bottom panel). A total of 5000 negative test sequences (−) and the seven positive test sequences (|) were scored to the model. The *E*-value is the expected number of matches, hence lower *E*-values are more significant matches. The ATP algorithm results in an improved separation of positive and negative test sequences.

Forward algorithm considers the likelihood of each and every path through the model. Calculating these likelihoods for all positive and negative sequences, we could iteratively adjust all transition probability parameters for better discrimination. This algorithm was very slow and yet the results were no better than those obtained by ATP, and seemed prone to overfit training data.

Earlier, we analyzed how to empirically estimate optimal transition prior probabilities and found that a maximum-likelihood (ML) estimated prior using a wealth of data was unable to produce optimal discrimination.[20] The probabilities into and from deletions were critical. In multiple alignments, most deletions are longer than just one residue, which is reflected in the relationship between the parameters $\alpha_{\text{delete-to-delete}} > \alpha_{\text{delete-to-match}}$ in the ML-estimated transition prior. However, this relationship is inverted in the empirically optimized prior. In other words, reducing the *a priori* probability for deletions gives much better discrimination in homolog searches. However, employing ATP gives considerably better results than standard profile HMM estimation using this best transition prior. Our interpretation is that ATP, by taking non-related sequences into account, can obtain a performance that is otherwise not achievable.

Both the results presented here and from the previous study on transition priors were obtained running HMMER in the global/local mode, where a sequence is aligned to the whole model. Global/local mode is HMMER default and more sensitive than the optional local/local mode†.[22] Our test set is not suited to test the local/local mode, as all positive test sequences are full-length domain sequences, hence we cannot draw valid conclusions on local/local performance of ATP. One could argue, however, that ATP should have less effect in local/local mode because here a sequence does not have to get through the whole HMM but can transit to the end state from any node. Unrelated sequences will therefore possibly gain less from parameters that are liberal with deletions and thus benefit less from ATP.

Madera & Gough[22] compared the HMMER and the SAM packages, and showed that the search algorithms perform similarly but that the SAM algorithm for estimating HMMs from a sequence alignment is superior to the HMMER algorithm. Nevertheless, we have chosen to use the HMMER package as a basis for our algorithm, one reason being that the source code of HMMER is freely available. We think, however, that the results could be of relevance also to other packages. ATP can be downloaded and be used by anyone who is already running the HMMER package‡.

† http://hmmer.wustl.edu/
‡ ftp://ftp.cgb.ki.se/pub/prog/ATP_HMM

## Methods

### Algorithm

ATP starts with estimating a profile HMM from a multiple sequence alignment using *hmmbuild* (the HMM construction program in HMMER). A number of sequences are sampled randomly from the negative set and are aligned to the HMM; those scoring highest are used as negative training sequences. We now have a set of positive and a set of negative sequences. Based on the paths these sets of sequences follow through the model, the main transition probability parameters are adjusted. All other probabilities (emission probabilities, special state probabilities and transitions between those) are locked.

HMMER uses sequence weighting to compensate for the fact that training sequences seldom constitute a diverse and evenly spread sample. Two identical sequences are thus assigned a relatively low weight compared to more unique sequences. The more similar the sequences, the lower the total weight, also called the effective sequence number. In ATP, positive sequence counts are based on these weights according to HMMER's default weighting scheme, while all negative sequences are given an equal weight of 1.

ATP employs the Viterbi algorithm, i.e. considers only the best path through the model. Note that all sequences have to pass either the match or the delete state of each node (Figure 1). The idea in ATP is to (1) look at the distribution of positive and negative sequences on match/delete states at each node and (2) adjust the transition probabilities in proportion to the difference in the distributions. Transition probabilities related to insertions are left unchanged.

First the Viterbi path is calculated for all training sequences, and for each node the weighted number of sequences that passes through match or delete state is counted ($c_{\text{pos}}^{\text{match}}$ and $c_{\text{pos}}^{\text{delete}}$ for the positives and $c_{\text{neg}}^{\text{match}}$ and $c_{\text{neg}}^{\text{delete}}$ for the negative training examples). These counts are used to estimate two simple probability distributions of positive and negative sequences ($D_{\text{pos}}$ and $D_{\text{neg}}$) on match/delete states. To avoid zero probabilities an *ad hoc* pseudocount of 0.5 is added to each state (i.e. 1 in total). For example, at a particular node, $D_{\text{pos}}$ is estimated as:

$$D_{\text{pos}} = (D_{\text{pos}}^{\text{match}}, D_{\text{pos}}^{\text{delete}}) = \left( \frac{c_{\text{pos}}^{\text{match}} + 0.5}{1 + c_{\text{pos}}^{\text{match}} + c_{\text{pos}}^{\text{delete}}}, \frac{c_{\text{pos}}^{\text{delete}} + 0.5}{1 + c_{\text{pos}}^{\text{match}} + c_{\text{pos}}^{\text{delete}}} \right) \quad (1)$$

Estimating $D_{\text{neg}}$ is done similarly from the counts for the negative training sequences. The relative entropy $H$ is then used as a measure of difference $\delta$ between $D_{\text{pos}}$ and $D_{\text{neg}}$ at each node:

$$\delta = (H(D_{\text{pos}} \| D_{\text{neg}}) + H(D_{\text{neg}} \| D_{\text{pos}}))/2 \quad (2)$$

where, generally:

$$H(P \| Q) = \sum_i P(x_i) \log \frac{P(x_i)}{Q(x_i)} \quad (3)$$

The relative entropy can never take on negative values and equals 0 only if $P$ and $Q$ are identical distributions. This means that $\delta = 0$ if the paths of positive and negative sequences are identical through a node in the model, and increasingly higher the more divergent the paths are. We can bound this between 0 and 1 using $1 - e^{-\delta}$, which is 0 for $\delta = 0$ and approaches 1 when $\delta$

**Table 2.** Minimum error rate (MER)/Over top noise (OTN) scores when varying the number of sampled sequences and $k$ in the ATP algorithm

| | Number of sampled sequences | | | |
|---|---|---|---|---|
| $k$ | 100 | **200** | 500 | 1000 |
| 0.5 | 1137/213 | 1138/212 | 1137/213 | 1136/215 |
| 0.75 | 1132/220 | 1134/214 | 1133/218 | 1134/216 |
| **1** | 1134/216 | **1131/220** | 1131/219 | 1132/221 |
| 2 | 1138/212 | 1138/209 | 1133/212 | 1133/215 |
| 5 | 1158/195 | 1151/199 | 1154/198 | 1147/199 |
| Time (s) | 221 | 337 | 476 | 1256 |

A low MER and a high OTN are desired. Mean results from three runs on the SCOP test are shown, with the chosen default settings in bold. The time necessary to construct the HMMs from the 70 families in the test set is shown.

increases. The transition probabilities are now updated as:

$$\tau_{\text{new}} = \tau_{\text{old}}(1 + k(1 - e^{-\delta})) \quad (4a)$$

or:

$$\tau_{\text{new}} = \tau_{\text{old}}/(1 + k(1 - e^{-\delta})) \quad (4b)$$

for each transition changed, where $k$ is a parameter governing the magnitude of change. For example, if $D_{\text{neg}}^{\text{delete}} > D_{\text{pos}}^{\text{delete}}$ at node $l$, we make it more "expensive" to use this delete state. Transition probabilities into this delete state are therefore decreased (equation (4b)) and transition probabilities leading into the match state of the same node are increased (equation (4a)). No transition probability involving insert states is adjusted. Finally, the probability parameters are renormalized to sum to 1 for each node.

We ran a large number of tests to find out how ATP performs with different parameter settings. We varied the number of random sequences sampled, the number of best-scoring random sequences kept for model adjustment and the $k$ in equations (4a) and (4b). The number of best-scoring random sequences kept for model adjustment was set to 10, which in our experience is suitable. Table 2 shows how the performance varied with the other two parameters. Sampling more sequences takes longer, but adds little to discrimination improvement. More important is the magnitude of change and an optimal setting seems to be close to $k = 1$. We set the ATP parameters to 200 sampled sequences and $k = 1$, and used this in comparisons with default HMMER.

### Details and settings

The SCOP training families were aligned using CLUSTAL W 1.83.[23] HMMs were built from the training sequences using the default settings in HMMER. Default settings means the models were configured for global/local searches, i.e. a test sequence is aligned locally to the whole model. Models were calibrated using *hmmcalibrate*. Test sequences were scored to the model using *hmmsearch*.

## Acknowledgements

We thank Sean Eddy for providing benchmarking tools. This work was supported by grants from Pfizer Inc. and the Swedish Knowledge Foundation.

## References

1. Park, J., Karplus, K., Barrett, C., Hughey, R., Haussler, D., Hubbard, T. & Chothia, C. (1998). Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J. Mol. Biol.* **284**, 1201–1210.
2. Krogh, A., Brown, M., Mian, I. S., Sjolander, K. & Haussler, D. (1994). Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.* **235**, 1501–1531.
3. Hughey, R. & Krogh, A. (1996). Hidden Markov models for sequence analysis: extension and analysis of the basic method. *Comput. Appl. Biosci.* **12**, 95–107.
4. Eddy, S. R. (1998). Profile hidden Markov models. *Bioinformatics*, **14**, 755–763.
5. Karplus, K., Barrett, C. & Hughey, R. (1998). Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, **14**, 846–856.
6. Gough, J., Karplus, K., Hughey, R. & Chothia, C. (2001). Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure. *J. Mol. Biol.* **313**, 903–919.
7. Karchin, R. & Hughey, R. (1998). Weighting hidden Markov models for maximum discrimination. *Bioinformatics*, **14**, 772–782.
8. Barrett, C., Hughey, R. & Karplus, K. (1997). Scoring hidden Markov models. *Comput. Appl. Biosci.* **13**, 191–199.
9. Brown, M., Hughey, R., Krogh, A., Mian, I. S., Sjolander, K. & Haussler, D. (1993). Using Dirichlet mixture priors to derive hidden Markov models for protein families. *Proc. Int. Conf. Intell. Syst. Mol. Biol.* **1**, 47–55.
10. Sjolander, K., Karplus, K., Brown, M., Hughey, R., Krogh, A., Mian, I. S. & Haussler, D. (1996). Dirichlet mixtures: a method for improved detection of weak but significant protein sequence homology. *Comput. Appl. Biosci.* **12**, 327–345.
11. Bateman, A., Birney, E., Cerruti, L., Durbin, R., Etwiller, L., Eddy, S. R. *et al.* (2002). The Pfam protein families database. *Nucl. Acids Res.* **30**, 276–280.
12. Letunic, I., Goodstadt, L., Dickens, N. J., Doerks, T., Schultz, J., Mott, R. *et al.* (2002). Recent improvements to the SMART domain-based sequence annotation resource. *Nucl. Acids Res.* **30**, 242–244.
13. Haft, D. H., Selengut, J. D. & White, O. (2003). The TIGRFAMs database of protein families. *Nucl. Acids Res.* **31**, 371–373.
14. Mamitsuka, H. (1996). A learning method of hidden Markov models for sequence discrimination. *J. Comput. Biol.* **3**, 361–373.
15. Eddy, S. R., Mitchison, G. & Durbin, R. (1995). Maximum discrimination hidden Markov models of sequence consensus. *J. Comput. Biol.* **2**, 9–23.
16. Jaakkola, T., Diekhans, M. & Haussler, D. (2000). A discriminative framework for detecting remote protein homologies. *J. Comput. Biol.* **7**, 95–114.
17. Karchin, R., Karplus, K. & Haussler, D. (2002). Classifying G-protein coupled receptors with support vector machines. *Bioinformatics*, **18**, 147–159.
18. Murzin, A. G., Brenner, S. E., Hubbard, T. & Chothia, C. (1995). SCOP: a structural classification of proteins

database for the investigation of sequences and structures. *J. Mol. Biol.* **247**, 536–540.
19. Chandonia, J. M., Walker, N. S., Lo Conte, L., Koehl, P., Levitt, M. & Brenner, S. E. (2002). ASTRAL compendium enhancements. *Nucl. Acids Res.* **30**, 260–263.
20. Wistrand, M. & Sonnhammer, E. L. L. (2004). Transition priors for protein hidden Markov models: an empirical study towards maximum discrimination. *J. Comput. Biol.* **11**, 181–194.
21. Brenner, S. E., Chothia, C. & Hubbard, T. J. (1998). Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl Acad. Sci. USA*, **95**, 6073–6078.
22. Madera, M. & Gough, J. (2002). A comparison of profile hidden Markov model procedures for remote homology detection. *Nucl. Acids Res.* **30**, 4321–4328.
23. Thompson, J. D., Higgins, D. G. & 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. *Nucl. Acids Res.* **22**, 4673–4680.

***Edited by J. Thornton***

*(Received 20 November 2003; received in revised form 25 February 2004; accepted 4 March 2004)*