Homology Detection via Family Pairwise Search

William Noble Grundy
Department of Computer Science and Engineering
University of California, San Diego (619) 453-4364
La Jolla, California 92093-0114 FAX 534-7029
bgrundy@cs.ucsd.edu

Journal of Computational Biology, 5(3):479-492, 1998.

Abstract

The function of an unknown biological sequence can often be accurately inferred by identifying sequences homologous to the original sequence. Given a query set of known homologs, there exist at least three general classes of techniques for finding additional homologs: pairwise sequence comparisons, motif analysis, and hidden Markov modeling. Pairwise sequence comparisons are typically employed when only a single query sequence is known. Hidden Markov models (HMMs), on the other hand, are usually trained with sets of more than 100 sequences. Motif-based methods fall in between these two extremes.

The current work introduces a straightforward generalization of pairwise sequence comparison algorithms to the case when when multiple query sequences are available. This algorithm, called Family Pairwise Search (FPS), combines pairwise sequence comparison scores from each query sequence. A BLAST implementation of FPS is compared to representative examples of hidden Markov modeling (HMMER) and motif modeling (MEME). The three techniques are compared across a wide range of protein families, using query sets of varying sizes. BLAST FPS significantly outperforms motif-based and HMM methods. Furthermore, FPS is much more efficient than the training algorithms for statistical models.

Running head: Family Pairwise Search

Keywords: Homology detection, proteins, pairwise sequence comparison, motif analysis, statistical modeling.

1 Introduction

The Human Genome Project and similar work on other species are producing biological sequence data at an accelerating rate. However, this data represents only a first step toward the goal of understanding the functions of these genetic sequences. Computational methods, although they will probably never replace the wet lab techniques of molecular biology, provide an important set of tools for inferring function.

One of the most effective means of inferring the function of an unidentified protein is to ask what functions are performed by homologous proteins. Two proteins are homologous if they share a common ancestor. Since the actual sequence of the common ancestor is unavailable, sequence homology can only be inferred by statistical means.

The most widely used means of inferring homology involves performing pairwise comparisons between a single query sequence and a sequence in a protein database. Dynamic programming algorithms, such as the Needleman-Wunsch [28] and Smith-Waterman algorithms [39], or related heuristic algorithms, such as BLAST [2, 3] and FASTA [33], can be used to assign to each sequence in the database a score indicating the likelihood that this sequence is homologous to the query sequence.

Because homology inferences are based upon statistical measures, they become increasingly uncertain when the evidence for homology is weak. The *twilight zone* of sequence similarity sets the boundary of confidence levels for detecting evolutionary relatedness of proteins [15]. For most pairwise alignment programs, the twilight zone falls between 20-25% sequence identity [14].

In order to push back the twilight zone and thereby discover more remote homologs, additional information is needed. Family-based methods of homology detection leverage the information contained in a set of proteins that are known to be homologous. In a diverse family of proteins, individual members may have very low pairwise sequence similarity and hence might be missed by a pairwise analysis. An algorithm that uses a representative set of sequences from the family, however, can uncover these missed relationships because homology is transitive [35, 3].

The simplest means of detecting homologs using a set of related query proteins is to perform multiple pairwise comparisons. In the FPS algorithm, each sequence in the database is compared with each sequence in the query set and the resulting scores are combined into an overall score for that sequence, either by taking the average or the best score from the set. This approach may be augmented by adding the newly discovered homologs to the query set and iterating until a transitive closure of the homology relationship is computed [40, 30].

More sophisticated homology detection methods involve two steps: first building a statistical model of the family and then comparing that model to each sequence in the database. For example, hidden Markov models (HMMs) have been used extensively to model protein families [26, 10, 16]. These statistical models have a strong theoretical basis in probability and are supported by efficient algorithms for training, database searching, and multiple sequence alignment. The model parameters are learned via expectation-maximization, and the homology detection algorithm is a form of dynamic programming. One drawback to modeling proteins using HMMs is that they contain many free parameters and therefore require a large amount of training data. A typical, 200-state HMM may contain on the order of 5000 trainable parameters. Adequate training of such a model can require on the order of 200 homologous sequences [13]. The use of empirically derived Dirichlet mixture priors [13, 38] can partially offset the need for larger training sets.

The size of the model may be greatly reduced by focusing only upon regions that are highly conserved across family members. Usually these regions, called motifs, have been conserved by evolution for important structural or functional reasons. As such, the motifs constitute a summary of the essential details of the family of proteins. Motifs may be represented as regular expressions [8, 31], or more generally as profiles [18] or position-specific scoring matrices, in which each column in the matrix represents a distribution across the amino acids at that position in the motif. These matrix models are formally equivalent to a class of HMMs and hence may be learned via expectation-maximization [4] or Gibbs sampling [27] from a set of unaligned protein sequences. Because motif-based methods ignore the poorly conserved spacer regions between motifs, they can be trained using smaller sets of related sequences.

This work compares the performance of each of these three homology detection methods and examines the extent to which each is dependent upon the size of the query set. A BLAST-based implementation of FPS performs better than both model-based methods. All three methods have difficulty recognizing larger protein families. Also, larger query sets uniformly lead to improved homology detection. Pairwise comparisons and hidden Markov models have difficulty recognizing families containing repeated elements. A version of FPS that selects the best BLAST E-value score performs better than versions of FPS that use the best bit score or the average E-value or bit scores. Overall, BLAST Family Pairwise Search provides excellent performance and is much more computationally efficient than competing, model-based methods of homology detection.

2 Algorithm

The Family Pairwise Search algorithm, summarized in Figure 1, is a straightforward extension of pairwise sequence comparison that allows for multiple multiple-sequence queries. FPS takes as input a query set of sequences known to be homologous to one another, as well as a target database to be searched. In the first phase of the algorithm, the database is searched separately with each sequence in the query set. The result, for a query set of n sequences, is a set of n similarity scores for each sequence in the database. The n scores are then

```
procedure FPS (query_set, database, compare_fn, combine_fn)
  for i ← 1 to size(database)
    target ← database[i]
    score_set = {}
    for j ← 1 to size(query_set)
      score ← compare_fn(query_set[j], target)
      score_set ← score_set ∪ {score}
    end
    scores[i] = combine_fn(score_set)
  end
  return scores
```

Figure 1: **The Family Pairwise Search algorithm.**

combined to yield a final score for that database sequence. The output of FPS is a scored version of the given database.

FPS is parameterized via two functions: a function that compares pairs of sequences and a function that combines multiple query scores. In this work, all pairwise sequence comparisons are carried out using BLAST. Presumably, FPS's performance could be improved by using the Smith-Waterman algorithm instead. Multiple sequence scores are combined either by averaging or by taking the best score. The choice of score combination function depends in part upon the type of scores computed by the pairwise similarity algorithm. For example, averaging E-values is unlikely to be effective, since the E-value does not scale linearly with the degree of sequence similarity. Taking the maximum score has the further advantage that it obviates the need for sequence weighting.

3 Methods

Protein families

A collection of 73 protein families [6] is used in the homology detection experiments. These families were selected from the Prosite database [8] for their difficulty, based upon the number of false positives reported in the Prosite annotations. The Prosite IDs and sizes of these families are listed in Appendix A. The families range in size from 5 to 109 sequences, and from 949 to 58 015 amino acids. The associated release of SWISS-PROT [9] contains 36 000 sequences and nearly 12.5 million amino acids.

Attempts to model families of related proteins are necessarily biased because the set of known protein sequences does not uniformly sample from the space of existing family members. Sequence weighting schemes attempt to compensate for this bias by assigning weights to individual sequences, usually based upon the level of sequence similarity. Such schemes may significantly improve the performance of homology detection algorithms [1, 37, 41]; however, Henikoff and Henikoff [22] have shown that many weighting schemes perform almost as well as one another. Accordingly, all the experiments reported here employ a simple, binary weighting scheme based upon BLAST similarity scores [29]. This approach is simple, since the highly similar sequences can be removed at once before any analysis is performed, and leads to faster training, since the sizes of the weighted training sets are reduced. For these experiments, a BLAST similarity threshold of 200 is used. The sizes of the weighted Prosite families, given in Appendix A, range from 1 to 73 sequences with an average of 10.7 sequences, and from 394 to 18 702 amino acids with an average of 4202.

For each family, nested query sets of sizes 2, 4, 8, 16 and 32 sequences are randomly selected from the set of weighted sequences. This results in 73 query sets of size 2, 57 sets of size 4, 35 of size 8, 16 of size 16 and 3 query sets of size 32. In addition, for each family a single, independent test set is constructed, consisting of all family members not contained in the query sets.

Pairwise sequence comparison

For homology detection using Family Pairwise Search, gapped BLAST version 2.0 is used [2, 3]. BLAST is a heuristic approximation of a dynamic programming optimization of maximal segment pair scores. The program is run with its default parameters, including the filtering of low-complexity regions and the use of the BLOSUM62 scoring matrix. For each database sequence, BLAST computes two scores, a bit score and an E-value. Initial FPS experiments are carried out using the best (i.e., minimum) E-value score. However, the performance of FPS using maximum bit scores and average E-value and bit scores is also examined. For each database search, an expectation threshold of 1000 is used, and any database sequence that would have received an E-value larger than 1000 is assigned an E-value of 1000 and a bit score of 0 (the lowest possible bit score).

Motif analysis

Ungapped motifs are discovered using MEME version 2.1 [4] with the default parameter settings from the web interface [20]. These defaults include empirical Dirichlet mixture priors weighted according to the megaprior heuristic [5], a minimum motif width of 12 and a maximum of 55, and a motif model biased toward zero or one motif occurrence per sequence. A total of ten motifs is discovered from each query set, and motif significance is judged using the majority occurrence heuristic [21]: motifs that do not appear in more than half of the query sequences are discarded. This heuristic excludes motifs that are specific to subfamilies of the given query set. For eight-sequence queries, the heuristic selects an average of 5.2 motifs.

Homology detection is performed using MAST [7, 6]. For each sequence in the database, MAST computes a p-value for each given motif and combines these values assuming that motif occurrences are statistically independent. The resulting sequence-level E-value scores are used to rank the sequences in the database.

Hidden Markov model analysis

Hidden Markov models of each query set are built using the HMMER software package version 1.8 [16]. Models are trained using expectation-maximization coupled with simulated annealing. The default geometric annealing schedule is used, and Dirichlet mixture priors are used in order to allow the models to be trained with smaller training sets.

Preliminary experiments showed that, of the four search programs provided in HMMER, `hmmsw` consistently provides the best results when searching protein databases. That program uses a modified form of the Smith-Waterman algorithm to search for sequence-to-model matches, allowing partial matches to either the sequence or the model. Thus, `hmmsw` performs a semi-local search. For each database sequence, the program returns a log-odds scores in bits.

Evaluating search results

Each homology detection experiment returns a score-labeled version of the database. For BLAST FPS, sequences are labeled with the combined pairwise score, as described above. For MAST, sequences are labeled with E-values, and for HMMER, sequences are labeled with log-odds scores. The database is then sorted according to these scores, and each sequence in the sorted database is marked with a "1" or a "0," indicating whether that sequence appears in the Prosite listing for the current family. In order to test the ability of the homology detection algorithms to generalize from the query set, all family members that do not appear in the independent test set are eliminated from the sorted list. The resulting, purged sequence of bits represents the homology detection algorithm's ability to separate novel family members from non-family members. Perfect performance corresponds to a series of 1s followed by a series of 0s.

This bit sequence is subjected to two forms of analysis. The first is a modified version of the Receiver Operating Characteristic, called ROC_{50} [19]. The ROC score is the area under a curve that plots true positives versus true negatives for varying score thresholds. ROC analysis combines measures of a search's sensitivity and selectivity. The ROC_{50} score is the area under the ROC curve, up to the first 50 false positives. This value has the advantages of yielding a wider spread of values, of requiring less storage space, and of corresponding to the typical biologist's willingness to sift through only approximately fifty false positives. ROC_{50} scores are normalized to range from 0 to 1, with 1.0 corresponding to the most sensitive and selective search.

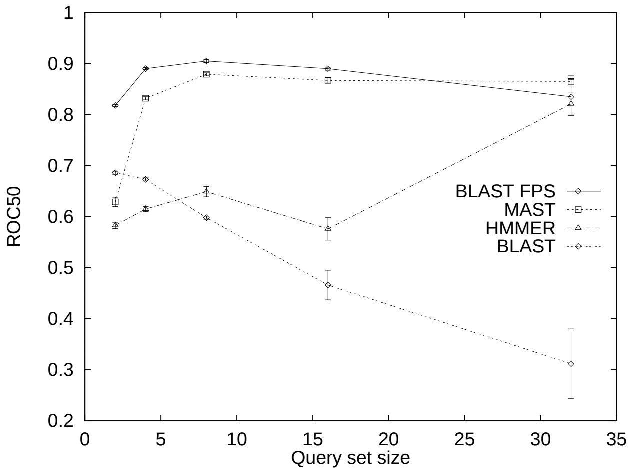
In addition to ROC_{50} analysis, each homology detection method is evaluated using the equivalence number [34]. The equivalence number is the number of false positives given by a database search when the threshold is set so that the number of false positives equals the number of false negatives. To compute the equivalence number from the sequence of bits described above, a mark is moved along the sequence until the number of 0s to the left of the mark equals the number of 1s to the right. Perfect separation corresponds to an equivalence number of 0, and the maximum possible equivalence number is the size of the family. In the results reported here, equivalence numbers are scaled to range from 0 to 1 by dividing by the size of the family. This allows equivalence numbers from homology searches for variously sized families to be combined.

4 Results

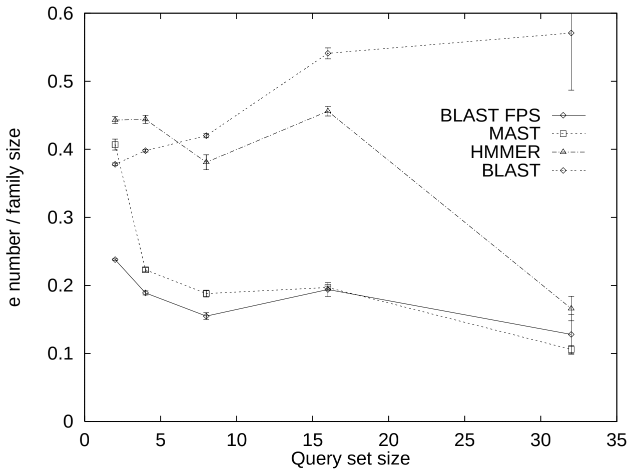
Figure 2(a) shows the average ROC_{50} scores for each homology detection method on all 73 protein families in the study. Also included are ROC_{50} scores from a BLAST search using a single sequence randomly selected from the smallest query set for the family. In comparing distributions of scores, a paired t test is used, and differences are deemed significant at a 1% confidence level. In Figure 2(a), BLAST FPS significantly outperforms all other methods for query set sizes of 2, 4 and 8 sequences. At a query set size of 16, BLAST FPS and MAST perform approximately as well as one another. Only three families contain more than 32 weighted members, so the differences between techniques at that query set size are not significant. Considering ROC_{50} scores from all 183 query sets together, BLAST FPS performs significantly better than MAST and HMMER, and MAST performs better than HMMER.

In Figure 2(b), the same homology detection results are analyzed using equivalence numbers. Recall that, unlike ROC_{50} scores, a lower equivalence number is better. Hence, the two methods of analysis closely agree as to the relative performance of the three homology detection methods. However, the differences between normalized equivalence numbers are not as significant. Overall, BLAST FPS still performs better than MAST, which in turn performs better than HMMER. However, BLAST FPS is only significantly better than MAST for query sets of size 2 (1% confidence) and 8 (5% confidence). HMMER performs significantly worse than BLAST FPS and MAST for all query set sizes except 32.

One unexpected characteristic of Figure 2 is the downward trend of the scores as the query set size increases. However, this trend is an artifact of the presentation of the data: the 73 2-sequence queries contain many sequences from very small families. For these small families, the task of homology detection is relatively easy. The sixteen 16-sequence query sets, however, each correspond



(a)



(b)

Figure 2: **Superior performance of the BLAST FPS algorithm.** Figure (a) plots ROC_{50} score, and Figure (b) plots normalized equivalence number, both as a function of query set size. For each of the 73 Prosite families, nested query sets were randomly selected after binary weighting was carried out. The figure includes 73 query sets of size 2, 57 sets of size 4, 35 of size 8, 16 of size 16 and 3 sets of size 32. Error bars represent standard error.

Family	BLAST FPS		HMMER		MAST		Total
	ROC_{50}	Rank	ROC_{50}	Rank	ROC_{50}	Rank	rank
PS00190	0.585	2	0.016	3	0.530	3	8
PS00339	0.543	1	0.305	8	0.433	1	10
PS00211	0.659	3	0.277	6	0.781	6	15
PS00402	0.770	5	0.383	11	0.503	2	18
PS00092	0.891	11	0.053	4	0.732	4	19
PS00120	0.696	4	0.393	12	0.824	9	25
PS00038	0.842	7	0.335	9	0.906	15	31
PS00061	0.863	8	0.439	13	0.863	10	31
PS00198	0.824	6	0.824	17	0.892	13	36
PS00301	0.885	10	0.733	15	0.868	11	36
PS00030	0.927	14	0.278	7	0.927	18	39
PS00343	0.873	9	0.871	21	0.884	12	42
PS00338	0.936	15	0.725	14	0.936	19	48
PS00659	0.917	13	0.190	5	0.992	33	51
PS00716	0.894	12	0.923	25	0.894	14	51

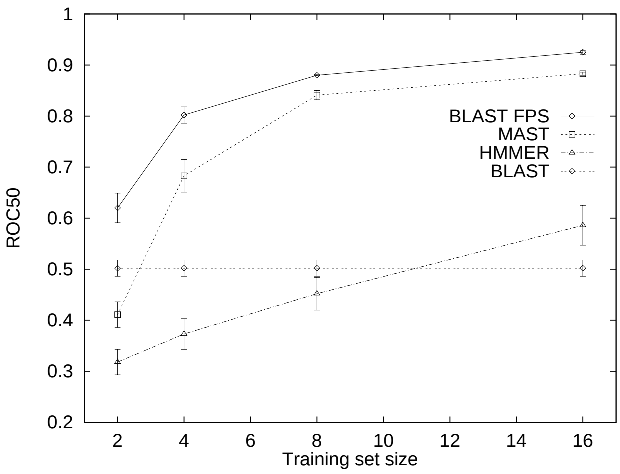
Table 1: **Difficult families.** Listed are the fifteen families that contain eight or more weighted sequences and that received the lowest ROC_{50} scores for 8-sequence queries. For each method, the families are ranked by increasing ROC_{50} score. They are listed in order of increasing total rank.

to a relatively large and hence difficult-to-recognize protein family. The effect of family size upon recognition difficulty is clearly illustrated by the downward trend of the single-sequence BLAST series in Figure 2. Since each point in this series represents data collected from single-sequence BLAST searches, the only difference between successive points in the series is the families over which the scores are averaged.

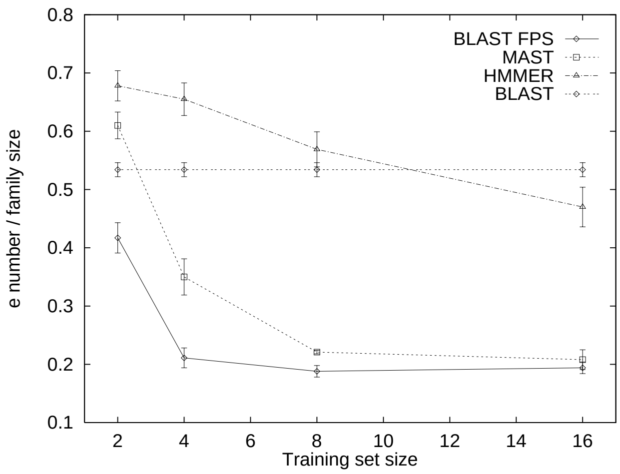
Figure 3 corrects for differences in family size by including only families containing between 16 and 31 members. Here, the trend toward better performance with more query sequences is clearer. As before, the overall difference between methods are significant at the 1% confidence level.

Some protein families are difficult to recognize regardless of the homology detection method employed. Table 1 shows the fifteen families that received the lowest ROC_{50} scores from all three methods. The data show a strong correlation between the families for which the three homology detection methods have difficulty: included in the fifteen most difficult families are all fifteen of the most difficult families for BLAST FPS, and twelve of the most difficult fifteen families for MAST and HMMER. This agreement indicates that, for these families, a low ROC_{50} score indicates a family that is difficult to recognize, rather than a problem with the homology detection method.

Any evaluation of homology detection methods can only be as accurate as the



(a)



(b)

Figure 3: **Improved performance of homology detection algorithms with larger query sets.** Figure (a) plots ROC_{50} score as a function of query set size; Figure (b) plots normalized equivalence number as a function of query set size. Each figure includes data from the 13 families containing more than fifteen and less than 32 members after binary sequence weighting. Error bars represent standard error.

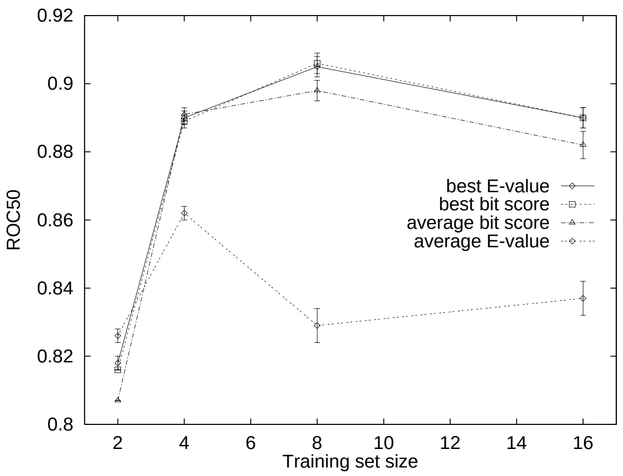
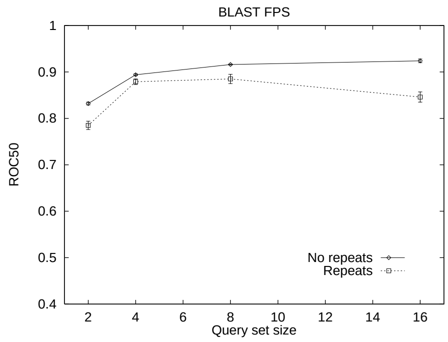


Figure 4: **Relative performance of variants of the BLAST FPS algorithm.** The figure plots ROC_{50} score as a function of query set size for all 73 families.

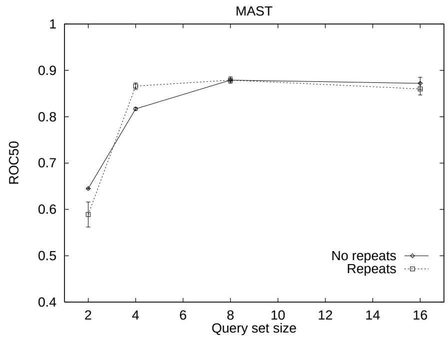
curated list of family members upon which the evaluations are based. Unannotated family members will cause all three methods to apparently perform poorly on that family. Thus, for example, the first 50 false positives that BLAST uncovered for the cytochrome c family contain six sequences for which the annotation includes the words `CYTOCHROME C`. Three of these false positive sequences are cytochrome C precursors; three more are listed as cytochrome c family members in a later version of SWISS-PROT (one as a potential member).

Numerous variants of the FPS algorithm are possible. Figure 4 compares four such variants: scoring sequences using using BLAST E-values or bit scores, and combining these scores by taking the best score or the average score. The results show that using the best score, rather than the average score, provides better homology detection performance. Either the best bit score or the best E-value performs approximately as well, although a paired t test comparing ROC_{50} scores from all 183 training sets indicates that the bit score gives slightly better performance, with 1% confidence. Not surprisingly, computing the average E-value score does not lead to good performance.

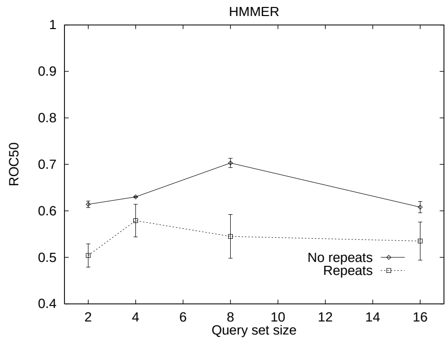
An important difference between BLAST and MAST on the one hand and HMMER on the other is that the former two algorithms employ local search techniques. The `hmmsw` program, in contrast, performs a semi-local search, in which a single subsequence of the HMM can match a subsequence of the database protein sequence. The three-row topology of the standard linear HMM implies a simple model of evolution, involving point mutations, insertions and deletions. Semi-local searching adds to this model the possibility of large-scale



(a)



(b)



(c)

Figure 5: **Relative performance of homology detection algorithms on families with and without repeated elements.** Each figure plots the average ROC_{50} score of one homology detection algorithm as a function of query set size for families with and without repeated elements. The figures contain data for 21 families containing repeats and 52 families without repeats. Error bars represent standard error.

Program	Query set size		
	2	4	8
BLAST FPS	18.2	39.4	82.3
MEME	67.2	170.4	548.0
MAST	65.5	39.7	33.9
hmmt	41.6	62.6	1716.7
hmmsw	8965.2	8772.3	8692.0

Table 2: **Typical execution times for the three homology detection methods.** Times reported are total CPU time in seconds on a 167 MHz Sparc Ultra for one protein family.

deletions and insertions at either end of the protein. Still, however, the linear topology cannot accurately model protein families in which motifs or domains are repeated or shuffled. Accordingly, one would expect HMMER to perform poorly on families known to contain repeated elements. Figure 5 illustrates this effect. Prosite annotations are used to separate out those families containing repeated elements. For all query set sizes, HMMER performs significantly worse on families containing repeated domains. For MAST, although some differences between ROC_{50} scores for families with and without repeats are significant, those differences are smaller, and no consistent trend appears. Surprisingly, however, BLAST performs better on families without repeated domains. This is unexpected because the gapped BLAST algorithm allows a single subsequence to participate in more than one maximal segment pair and therefore should be able to cope with repeated elements.

One important reason for using homology detection to infer protein function is speed. Most wet lab experiments are slow relative to a protein database search. However, not all computational methods are equally fast. In this respect, BLAST FPS clearly outperforms both MEME/MAST and HMMER. For example, Table 2 shows typical timing data for one protein family. For an 8-member query set, BLAST FPS requires only 82 seconds; combined, MEME and MAST require 9.7 minutes, and HMMER training and searching require 2.9 hours. BLAST implements a linear algorithm, whereas the training algorithms for both MEME and HMMER are roughly $O(n^2)$ in the size of the training set. On the other hand, the MAST search algorithm is considerably faster than the corresponding HMMER search algorithm, `hmmsw`. A MAST query requires less than a minute, but with `hmmsw`, searching even a relatively small database like SWISS-PROT takes nearly 2.5 hours on a fast workstation.

5 Discussion

For family-based homology detection using query sets of the sizes investigated here, the Family Pairwise Search algorithm is clearly preferable to the more computationally expensive statistical modeling methods tested here. The reasons for FPS's excellent performance are four-fold.

First, the FPS score incorporates information from multiple sequence comparisons into a single score. The method thereby allows for the detection of remote homologs that lack significant similarity with one or more of the training set sequences. FPS is therefore similar to the intermediate sequence approach suggested by Pearson [36] and Park *et al.* [32].

Also, FPS may perform well relative to motif-based methods because BLAST allows for query-to-target matches along the entire length of the sequences, rather than only within the motif regions. These non-motif regions often contain important evidence of homology [36].

On the other hand, FPS avoids building models of the relatively noisy, inter-motif regions in the query set. The difficulty of properly aligning these regions, especially when the number of query sequences is small, most likely accounts for the relatively poor performance of the hidden Markov model method.

Finally, by avoiding a position-specific scoring matrix representation of the training sequences, FPS does not assume that the occurrences of amino acids at a particular site in the protein are independent of amino acid occurrences at other sites in the same protein. If, in fact, covariation between sites imposes a significant evolutionary constraint, then searching separately with each training set sequence will respect that constraint.

The results reported here appear to be in conflict with those of Tatusov *et al.* [40]. They compare four homology detection techniques based upon position-specific scoring matrices with a control method in which candidate sequences are scored according to their maximum match with any sequence in a set of known homologs. This control method is thus very similar to FPS. Tatusov *et al.* report that all four matrix-based techniques provide superior homology detection performance relative to the control method. However, they build matrices only of motif regions and compare their matrix-based techniques to sequence searches that use only the same motif regions. The improvement reported here of BLAST FPS over MAST is likely a result of the FPS algorithm's ability to exploit homology information from the non-motif regions of the sequence. Furthermore, HMMER's relatively poor performance indicates that Tatusov *et al.*'s results likely would not extend to complete sequence models.

For fairness of comparison, the experiments reported here employ the default settings of each technique. It may be the case, however, that selecting different parameter settings for the various homology detection methods may result in slightly different results. For example, although both MEME and HMMER employ Dirichlet mixture priors, MEME weights the prior more heavily by default. This heuristic may have given MEME an advantage for the smaller training sets. Furthermore, advances in hidden Markov modeling, such as improved scoring schemes [11], internal sequence weighting [24] and maximum discrimination training [17], may significantly improve the performance of these methods.

Currently, an important drawback to FPS is its lack of accompanying statistics. Computing an E-value for the minimum of a set of E-values is difficult without an adequate model of the query sequence dependencies. Empirical approximations of these statistics will be the subject of future research. Even in the absence of accurate E-values, however, FPS should be useful as a baseline for comparison with future homology detection algorithms.

The large difference in performance between single-sequence BLAST queries on the one hand and family-based homology detection methods on the other suggests a bootstrap approach when only a single query sequence is available. In such an approach, BLAST would be used initially to search for close homologs, which would then be given to a family-based homology detection algorithm.

Iterating this bootstrap procedure should provide even better homology information than the single pass reported here. Iterative applications of BLAST have been suggested [25, 40] and implemented in PSI-BLAST [3]. However, PSI-BLAST searches the database using position-specific scoring matrix representations. In order to test the usefulness of this representation, it would be interesting to compare the performance of PSI-BLAST with that of an iterative search that employs Family Pairwise Search.

Acknowledgments

The author would like to thank Timothy Bailey for helpful discussion and for providing the 75 protein families and the accompanying database.

References

[1] S. F. Altschul, R. J. Carroll, and D. J. Lipman. Weights for data related by a tree. *Journal of Molecular Biology*, 207(4):647–53, 1989.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. A basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[3] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Research*, 25:3389–3402, 1997.

[4] T. L. Bailey and C. P. Elkan. Fitting a mixture model by expectation-maximization to discover motifs in biopolymers. In R. Altman, D. Brutlag, P. Karp, R. Lathrop, and D. Searls, editors, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, 1994.

[5] T. L. Bailey and M. Gribskov. The megaprior heuristic for discovering protein sequence patterns. In D. J. States, P. Agarwal, T. Gaasterland, L. Hunter, and

R. Smith, editors, *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 15–24. AAAI Press, 1996.

[6] T. L. Bailey and M. Gribskov. Score distributions for simultaneous matching to multiple motifs. *Journal of Computational Biology*, 4(1):45–59, 1997.

[7] T. L. Bailey and M. Gribskov. Combining evidence using p-values: Application to sequence homology searches. *Bioinformatics*, 14(1):48–54, 1998.

[8] A. Bairoch. PROSITE: A dictionary of sites and patterns in proteins. *Nucleic Acids Research*, 20:2013–2018, 1992.

[9] A. Bairoch. The SWISS-PROT protein sequence data bank: Current status. *Nucleic Acids Research*, 22(17):3578–3580, September 1994.

[10] P. Baldi, Y. Chauvin, T. Hunkapiller, and M. A. McClure. Hidden Markov models of biological primary sequence information. *Proceedings of the National Academy of Sciences of the United States of America*, 91(3):1059–1063, 1994.

[11] C. Barrett, R. Hughey, and K. Karplus. Scoring hidden Markov models. *Computer Applications in the Biosciences*, 13(2):191–199, 1997.

[12] NCBI BLAST search. `http://www.ncbi.nlm.nih.gov/BLAST`, 1997.

[13] M. Brown, R. Hughey, A. Krogh, I. Mian, K. Sjolander, and D. Haussler. Using Dirichlet mixture priors to derive hidden Markov models for protein families. In C. Rawlings et al., editor, *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 47–55. AAAI Press, 1995.

[14] S. Y. Chung and S. Subbiah. A structural explanation for the twilight zone of protein sequence homology. *Structure*, 4(10):1123–1127, 1996.

[15] R. F. Doolittle. *Of Urfs and Orfs: Primer on how to analyze derived amino acid sequences.* University Science Books, 1986.

[16] S. R. Eddy. Multiple alignment using hidden Markov models. In C. Rawlings et al., editor, *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pages 114–120. AAAI Press, 1995.

[17] S. R. Eddy, G. Mitchison, and R. Durbin. Maximum discrimination hidden Markov models of sequence consensus. *Journal of Computational Biology*, 2:9–23, 1995.

[18] M. Gribskov, R. Lüthy, and D. Eisenberg. Profile analysis. *Methods in Enzymology*, 183:146–159, 1990.

[19] M. Gribskov and N. L. Robinson. Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Computers and Chemistry*, 20(1):25–33, 1996.

[20] W. N. Grundy, T. L. Bailey, and C. P. Elkan. ParaMEME: A parallel implementation and a web interface for a DNA and protein motif discovery tool. *Computer Applications in the Biosciences*, 12(4):303–310, 1996.

[21] W. N. Grundy, T. L. Bailey, C. P. Elkan, and M. E. Baker. Meta-MEME: Motif-based hidden Markov models of protein families. *Computer Applications in the Biosciences*, 13(4):397–406, 1997.

[22] S. Henikoff and J. G. Henikoff. Position-based sequence weights. *Journal of Molecular Biology*, 243:574–578, 1994.

[23] S. R. Eddy group, Dept. of Genetics, Washington University. `http://genome.-wustl.edu/eddy/hmm.html`, 1997.

[24] R. Karchin and R. Hughey. Weighting hidden Markov models for maximum discrimination. *Bioinformatics*, 1998. To appear.

[25] E. V. Koonin and R. L. Tatusov. Computer analysis of bacterial haloacid dehalogenases defines a large superfamily of hydrolases with diverse specificity: application of an iterative approach to database search. *Journal of Molecular Biology*, 244(1):125–132, 1994.

[26] A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.

[27] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, 1993.

[28] S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology*, 48:444–453, 1970.

[29] A. F. Neuwald and P. Green. Detecting patterns in protein sequences. *Journal of Molecular Biology*, 239(5):698–712, 1994.

[30] A. F. Neuwald, J. Liu, D. Lipman, and C. Lawrence. Extracting protein alignment models from the sequence data database. *Nucleic Acids Research*, 25(9):1665–1677, 1997.

[31] C. G. Nevill-Manning, K. S. Sethi, T. D. Wu, and D. L. Brutlag. Enumerating and ranking discrete motifs. In *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology*, 1997.

[32] J. Park, S. A. Teichmann, T. Hubbard, and C. Chothia. Intermediate sequences increase the detection of homology between sequences. *Journal of Molecular Biology*, 273:1–6, 1997.

[33] W. R. Pearson. Rapid and sensitive sequence comparisions with FASTP and FASTA. *Methods in Enzymology*, 183:63–98, 1985.

[34] W. R. Pearson. Comparison of methods for searching protein sequence databases. *Protein Science*, 4:1145–1160, 1995.

[35] W. R. Pearson. Effective protein sequence comparison. *Methods in Enzymology*, 266:227–258, 1996.

[36] W. R. Pearson. Identifying distantly related protein sequences. *Computer Applications in the Biosciences*, 13(4):325–332, 1997.

[37] P. R. Sibbald and P. Argos. Weighting aligned protein or nucleic acid sequences to correct for unequal representation. *Journal of Molecular Biology*, 216(4):813–8, 1990.

[38] K. Sjolander, K. Karplus, M. Brown, R. Hughey, A. Krogh, I. S. Mian, and D. Haussler. Dirichlet mixtures: A method for improving detection of weak but significant protein sequence homology. *Computer Applications in the Biosciences*, 1996.

[39] T. Smith and M. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[40] R. L. Tatusov, S. F. Altschul, and E. V. Koonin. Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 91(25):12091–12095, 1994.

[41] J. D. Thompson, D. G. Higgins, and T. J. Gibson. Improved sensitivity of profile searches through the use of sequence weights and gap excision. *Computer Applications in the Biosciences*, 10(1):19–29, 1994.

A Prosite families

ID	Family	n	n_w	R
PS00030	Eukaryotic putative RNA-binding region RNP-1	59	24	Y
PS00037	Myb DNA-binding domain 1	18	4	Y
PS00038	Myc-type, 'helix-loop-helix' dimerization domain	90	26	N
PS00043	Bacterial regulatory proteins, gntR	10	8	N
PS00060	Iron-containing alcohol dehydrogenases 2	7	3	N
PS00061	Short-chain alcohol dehydrogenase	82	24	Y
PS00070	Aldehyde dehydrogenases cysteine active site	34	8	N
PS00075	Dihydrofolate reductase	33	14	N
PS00077	Cytochrome c oxidase subunit I, copper B binding region	53	2	N
PS00079	Multicopper oxidases 1	12	7	Y
PS00092	N-6 Adenine-specific DNA methylases	35	28	Y
PS00095	C-5 cytosine-specific DNA methylases C-terminal	33	16	N
PS00099	Thiolases active site	14	3	N
PS00118	Phospholipase A2 histidine active site	110	9	N
PS00120	Lipases, serine active site	36	14	N
PS00133	Zinc carboxypeptidases, zinc-binding region 2	19	5	N
PS00141	Eukaryotic and viral aspartyl proteases active site	50	13	Y
PS00144	Asparaginase / glutaminase active site 1	8	3	N
PS00180	Glutamine synthetase 1	55	7	N
PS00185	Isopenicillin N synthetase 1	10	2	N
PS00188	Biotin-requiring enzymes attachment site	15	8	N
PS00190	Cytochrome c family heme-binding site	223	73	Y
PS00194	Thioredoxin family active site	48	15	Y
PS00198	4Fe-4S ferredoxins, iron-sulfur binding region	109	53	Y
PS00209	Arthropod hemocyanins / insect LSPs 1	14	4	N
PS00211	ABC transporters	119	38	Y
PS00215	Mitochondrial energy transfer proteins	39	12	Y
PS00217	Sugar transport proteins 2	46	14	N
PS00225	Crystallins beta and gamma 'Greek key' motif	47	6	Y
PS00281	Bowman-Birk serine protease inhibitors	22	9	Y
PS00283	Soybean trypsin inhibitor (Kunitz) protease inhibitors	30	13	N
PS00287	Cysteine proteases inhibitors	32	11	Y
PS00301	GTP-binding elongation factors	110	8	N
PS00338	Somatotropin, prolactin and related hormones 2	86	12	N
PS00339	Aminoacyl-transfer RNA synthetases class-II 2	38	19	N
PS00340	Growth factor and cytokines receptors 2	37	16	N
PS00343	Gram-positive cocci surface proteins 'anchoring' hexapeptide	25	16	N
PS00372	PTS EIIA domains phosphorylation site 2	7	4	N
PS00399	ATP-citrate lyase and succinyl-CoA ligases active site	4	2	N
PS00401	Prokaryotic sulfate-binding proteins 1	5	2	N
PS00402	Binding-protein-dependent transport systems inner membrane component sign	39	19	N

ID	Family	n	n_w	R
PS00422	Granins 1	12	3	N
PS00435	Peroxidases proximal heme-ligand	41	8	N
PS00436	Peroxidases active site	40	8	N
PS00490	Prokaryotic molybdopterin oxidoreductases 2	9	6	N
PS00548	Ribosomal protein S3 1	18	3	N
PS00589	PTS HPR component serine phosphorylation site	10	5	Y
PS00599	Aminotransferases class-II pyridoxal-phosphate attachment site	21	8	N
PS00606	Beta-ketoacyl synthases active site	17	4	Y
PS00624	GMC oxidoreductases 2	9	5	N
PS00626	Regulator of chromosome condensation (RCC1) 2	6	2	Y
PS00637	CXXCXGXG dnaJ domain	9	5	N
PS00639	Eukaryotic thiol (cysteine) proteases histidine active site	62	19	N
PS00640	Eukaryotic thiol (cysteine) proteases asparagine active site	62	19	N
PS00643	Respiratory-chain NADH dehydrogenase 75 Kd subunit 3	5	2	N
PS00656	Glycosyl hydrolases family 6 2	5	4	N
PS00659	Glycosyl hydrolases family 5	40	19	N
PS00675	Sigma-54 interaction domain ATP-binding region A	36	6	N
PS00676	Sigma-54 interaction domain ATP-binding region B	36	6	N
PS00678	Beta-transducin family Trp-Asp repeats	26	17	Y
PS00687	Aldehyde dehydrogenases glutamic acid active site	33	7	N
PS00697	ATP-dependent DNA ligase AMP-binding site	11	6	N
PS00700	Ribosomal protein L6 2	13	4	N
PS00716	Sigma-70 factors 2	36	8	N
PS00741	Guanine-nucleotide dissociation stimulators CDC24	6	5	N
PS00760	Signal peptidases I lysine active site	8	5	N
PS00761	Signal peptidases I 3	8	5	N
PS00831	Ribosomal protein L27	6	3	N
PS00850	Glycine radical	4	3	N
PS00867	Carbamoyl-phosphate synthase subdomain 2	20	3	Y
PS00881	Protein splicing	3	3	Y
PS00904	Protein prenyltransferases alpha subunit	4	3	Y
PS00933	FGGY family of carbohydrate kinases 1	11	5	N

Prosite IDs of the 73 families included in this study. n is the total number of sequences in the family, and n_w is the number of sequences remaining after binary sequence weighting. The final column (R) indicates whether the family contains repeated elements. Two families from the original set of 75 [6] were discarded because they contained a single sequence after binary sequence weighting.