Discovery of Diagnostic Patterns from Protein Sequence Databases

Björn Olsson, Kim Laurio

Dept. of Computer Science, University of Skövde, Box 408, 54128, Skövde, Sweden
bjorne@ida.his.se, kim@ida.his.se

Abstract. We show how prior domain knowledge can be used in a system for mining databases of biological data. Our system performs automated discovery of diagnostic patterns from a database of protein sequences. Such patterns are used for classification of new sequences, and identification of biologically interesting positions in the proteins. The patterns have a simple syntax and can be translated into regular expressions, which can be used for rapid scanning of databases. Current pattern libraries are built semi-manually, since the correctness of the pattern depends on the incorporation of domain knowledge. Due to the dramatic growth of the databases it is desirable to automate this process. Our results show that the patterns derived by our fully automated system compete well with the semi-manually constructed patterns.

1 Introduction

In the past few years, there has been a dramatic increase in the amount of biological sequence data in public-domain databases. Since 1987, the number of protein sequences in the SWISSPROT database [2] has doubled every year, and the November 1997 release contains 68,830 entries. The December 1997 release of the EMBL database [7] of RNA and DNA sequences contains 1,917,868 sequence entries, and this database also doubles in size every year. This explosive growth of the amount of available sequence data will continue well into the 21st century. The Human Genome Project alone will contribute another estimated 60,000 to 80,000 protein sequences before the target date of completion in 2005. It can also be noted that the August 1995 release of the GenBank database [4] of DNA sequences, although containing 492,483 sequence records, only had sequences from 15,511 species [11]. Since the estimates of the number of extant species range from 5 to 50 million [8], the GenBank database only had sequences from at most 0.3% of the species. An average of 10 new species are added per day.

The fields of molecular biology and bioinformatics, although undergoing rapid development, still have problems in keeping up with the new data. An example is the problem of determining the 3D structure of proteins. The structure is known for less than 3% of all sequenced proteins [12], and the gap between the number of known sequences and solved structures is rapidly increasing [12] [13].

This paper addresses the construction and maintenance of a library of patterns for classification of protein sequences. Current pattern libraries, such as PROSITE [3], are built using semi-manual approaches, making it time-consuming to construct the initial patterns. Also, since new sequences are continuously added to the database, the patterns degrade over time. To maintain the discriminatory power of the patterns, they must frequently be inspected and updated.

Our work aims at a system for automatic discovery of biologically significant patterns, and automatic updates of the pattern library as new sequences are added to the database. Our method for pattern construction is based on information theory, but also makes extensive use of biological information to guide the construction process. Thus, we show an example of how knowledge discovery techniques can be amended by incorporating domain knowledge - in this case knowledge of typical amino acid frequencies found in protein sequence data.

The next chapter is a brief introduction to protein sequence analysis, and can be skipped by readers familiar with the area. The third chapter is an overview of our system, while chapter four discusses algorithms for constructing diagnostic patterns and shows how we extend and improve on current algorithms by using prior knowledge. The fifth chapter presents our results and conclusions.

2 Protein Sequences and Sequence Analysis

Genes are blueprints for the myriad of proteins which perform the important tasks within organisms. Through many generations, proteins have evolved which perform an impressive variety of tasks: acting as enzymes in biochemical reactions, performing transportation of nutrients, being sensors for taste and smell, being detectors of invading viruses in the immune system, and acting as switches turning genes on or off. Understanding the details of how proteins perform their function is one of the most important issues in molecular biology.

The building blocks of proteins are 20 different types of amino acids. The basic chemical structure of an amino acid is to contain a carboxyl group and an amino group, both attached to a central carbon atom. In addition, each type of amino acid has a unique side chain, which determines the specifics of its chemical properties. A protein is made up of a sequence of amino acids, joined together by peptide bonds. When an amino acid occurs in the sequence, it is called a residue. Many proteins consist of several hundred residues, and in some cases several thousand. The residue sequence is termed the primary structure of the protein. Before the protein becomes biologically functional in the cell, it undergoes a folding process, where the primary structure folds into a specific three-dimensional, tertiary structure (an example is shown in figure 4.

The central tenet in protein sequence analysis is that the amino acid sequence determines the tertiary structure, and that the tertiary structure determines the function of the protein [12]. Mutations are changes at sequence level, which may or may not have any significant effect on the folded conformation. Proteins with similar sequences are evolutionary related and have similar folds [6], whereas unrelated sequences generally produce different folds. Computational approaches in bioinformatics address many difficult problems by analysing the sequence, which can easily be handled by computer algorithms. One example of such a problem is to derive phylogenetic trees, showing the evolutionary relationships between proteins. Another problem is the one addressed in this paper: to derive patterns which can be used to classify new sequences according to family membership.

One of the most important basic techniques in sequence analysis is to derive alignments. As a very simple example of a sequence alignment, consider the following alignment of four variants of the word “sequence”:

```
S E - Q U E N C E
S E - Q - E N C E
S E a Q U E N C E
S E - Q U E N S E
```

The types of changes occurring in the example also occur in protein sequences, i.e. deletion (U in SEQENCE), insertion (A in SEAQUENCE), and replacement (S in SEQUENSE). The alignment highlights the changes, and clearly shows the positions which are preserved. Algorithms for constructing alignments seek to minimize the mutational distance between the sequences. In the analysis of a protein family, a multiple sequence alignment can be used to discover the residue positions which are evolutionarily conserved in the family. Such positions often correspond to important biological functionality of the protein.

Sequence analysis algorithms rely on public-domain databases, such as SWISSPROT [2], which contains sequences and extensive documentation for over 70,000 proteins. PROSITE [3] is a database of patterns for 997 protein families. The patterns can be seen as signatures, distinguishing family from non-family sequences. The Pfam database [17] contains 527 protein families. Among other things, Pfam contains an alignment of a subset of the sequences for each family.

3 Overview of Method

We aim to achieve an automated system for discovering and maintaining diagnostic protein sequence patterns. We are currently building a library of patterns derived through a system which interfaces three public-domain databases. Figure 1 shows an overview of our method. Every step of the method can be fully automated, making it possible to achieve large scale updates, so that the patterns can be continuously refined to take new sequences into account.

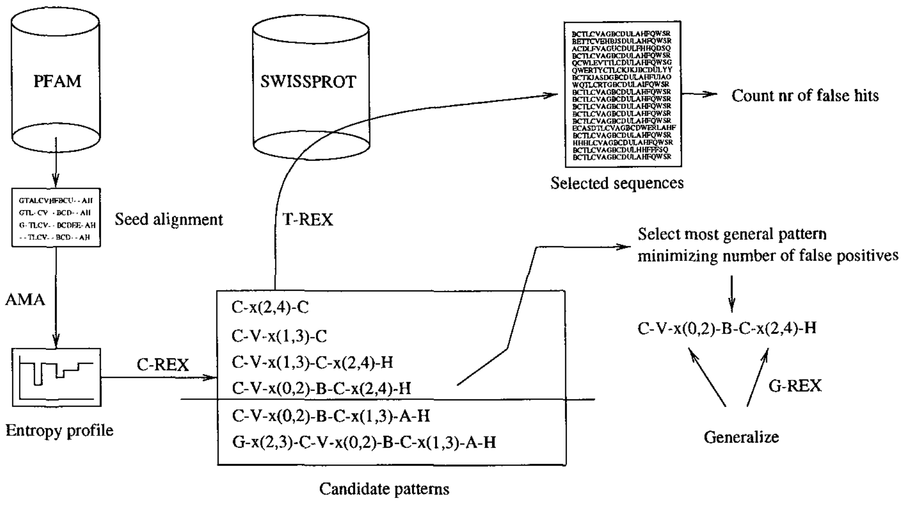
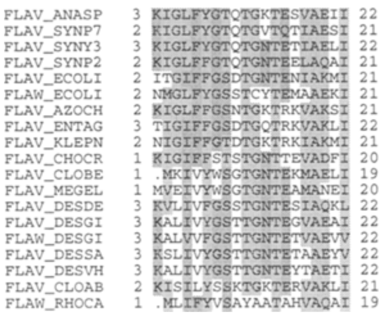


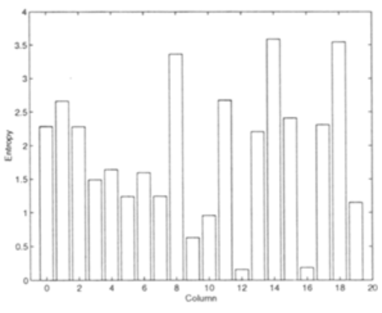
Fig. 1. Overview of the method.

When generating a pattern for a family, the system uses a multiple alignment from the Pfam database [18], aligning a subset of the sequences belonging to the family. The alignment is analysed by AMA (Analysis of Multiple Alignments [10]), which generates an entropy profile. The profile can be used to detect positions which are likely to be conserved in the family, and therefore good candidates as pattern elements. In estimating the entropy of columns, AMA takes into account biological domain knowledge by using a Dirichlet mixture. It was shown in [9] and [16] that use of this prior information improves the degree of generalisation in statistical models of small samples of protein sequence data. This is important since many Pfam alignments contain only a handful of sequences.

Using the entropy profile, C-REX (Creating REgular eXpressions) creates initial patterns by adding the most conserved columns, separated by wild-cards. More elements are added in the order of increasing entropy, and the gradually more specific patterns are tested by searching SWISSPROT. Since initial patterns match many false positives, C-REX adds elements until all false positives are excluded. In some cases, the pattern at this stage matches every family member, but in most cases it is too specific, and excludes some family members. Figure 2 illustrates the algorithm for extracting initial patterns. Low entropy columns correspond to promising pattern elements, and are used first. Column 12 is a conserved T-column with estimated entropy $\overrightarrow{p} \simeq 0.16$, and column 16 is a conserved A-column, having the second lowest estimated entropy ($\overrightarrow{p} \simeq 0.18$).



```
FLAV_ANASP   3 KIGLFYGTQTGKTESVAEII 22
FLAV_SYNP7   2 KIGLFYGTQTGVTQTIAESI 21
FLAV_SYNY3   3 KIGLFYGTQTGNTETIAELI 22
FLAV_SYNP2   2 KIGLFFGTQTGNTEELAQAI 21
FLAV_ECOLI   2 ITGIFFGSDTGNTENIAKMI 21
FLAW_ECOLI   2 NMGLFYGSSTCYTEMAAEKI 21
FLAV_AZOCH   2 KIGLFFGSNTGKTRKVAKSI 21
FLAV_ENTAG   3 TIGIFFGSDTGQTRKVAKLI 22
FLAV_KLEPN   2 NIGIFFGTDTGKTRKIAKMI 21
FLAV_CHOCR   1 KIGIFFSTSTGNTTEVADFI 20
FLAV_CLOBE   1 .MKIVYWSGTGNTEKMAELI 19
FLAV_MEGEL   1 MVEIVYWSGTGNTEAMANEI 20
FLAV_DESDE   3 KVLIVFGSSTGNTESIAQKL 22
FLAV_DESGI   3 KALIVYGSTTGNTEGVAEAI 22
FLAW_DESGI   3 KALVVFGSTTGNTETVAEVV 22
FLAV_DESSA   3 KSLIVYGSTTGNTETAAEYV 22
FLAV_DESVH   3 KALIVYGSTTGNTEYTAETI 22
FLAV_CLOAB   2 KISILYSSKTGKTERVAKLI 21
FLAW_RHOCA   1 .MLIFYVSAYAATAHVAQAI 19
```



0.16	T
0.34	T-x(3)-A
0.97	[TY]-x(2)-T-x(3)-A
1.93	[TY]-[ACG]-x(2)-T-x(3)-A
3.07	[TY]-[ACG]-x(2)-T-x(3)-A-x(2)-[ILV]

Fig. 2. Upper left: Small portion of multiple alignment. Upper right: The corresponding entropy profile. Lower: Subset of corresponding patterns, and for each pattern, the total entropy of the corresponding alignment columns.

G-REX (Generalising REgular eXpressions) generalises the pattern, but only allows generalisations which still exclude all false positives. The result is a more sensitive pattern, with the specificity of the original pattern being preserved. We define sensitivity and specificity as

$$Sens = \frac{True_{pos}}{True_{pos} + False_{neg}} \qquad Spec = \frac{True_{pos}}{True_{pos} + False_{pos}} \tag{1}$$

where $True_{pos}$ is the number of family sequences matched by the pattern, and $False_{neg}$ the number of family sequences not matched by the pattern, whereas

$False_{pos}$ is the number of non-family sequences matched by the pattern. This sensitivity measure is equal to the fraction of the known family sequences which are matched by the pattern, whereas specificity equals the probability that a sequence which is matched by the pattern really belongs to the family.

4 Construction of Sequence Patterns

An overview of algorithms for the construction of patterns is given in [5]. Briefly, the algorithms can be divided into two groups: those using "bottom-up" and "top-down" approaches. The essence of the bottom-up approach is to enumerate candidate patterns and count the number of occurrences of the candidate patterns that can be found in the sequences. The obvious limitation of this approach is that the size of the search space is exponential in the length of the patterns. In contrast, top-down approaches look for local similarities between sequences, and extract candidate patterns based on these similarities. This can be done in several ways, for example by searching for sufficiently long common substrings, or by first aligning the sequences to minimize the mismatches, and extract patterns from the alignment. Both the problem of finding the longest common substring, and that of finding the optimal alignment are NP-complete, and top-down algorithms therefore incorporate the use of heuristics.

Our algorithm is a top-down approach, and uses an alignment as starting point. Currently, we use Pfam's so called 'seed alignments' [18], which are hand-designed or automatically generated alignments of a subset of the family. Our algorithm makes no assumptions regarding the origin of the alignment, so that later versions of the system may include construction of the alignment.

4.1 Deriving an Entropy Profile Using Prior Knowledge

Central to our approach is the use of both information theory and Dirichlet mixture priors [16] in the alignment analysis. The entropy of each column of the alignment is estimated, and low-entropy columns used for building initial candidate patterns. The assumption is that low-entropy columns generally correspond to conserved positions in the protein, and that a pattern built from these columns will represent the most characteristic properties of the family. To estimate the entropy of columns we use the method developed in [10], based on the concept of entropy from Shannon's classical work [14]. The entropy of an alignment column can be estimated by

$$ent(\vec{p}) = -\sum_{i} p_i \, log \, p_i \tag{2}$$

where p_i is the estimated probability of observing symbol i in the column, so that a "flat" distribution over the symbol alphabet gives maximum entropy. The simplest way of estimating p_i is to use the observed frequency, but such a naive approach does not take into account the number of observations. This is a serious problem, since for small samples, there is a high risk that the observed frequencies do not correspond to the true distribution [16]. This problem is relevant in our case, since the input to our system is often an alignment of only 10 sequences or less, and we therefore use Dirichlet mixtures in the estimation of p_i. Dirichlet mixtures were shown in [9] to be close to the theoretical optimum for a prior, and have previously been used in other biosequence applications, such as hidden Markov models of protein families [1] and phylogenetic trees [15].

A Dirichlet mixture contains a number of components, representing typical amino acid distributions. The observed amino acid frequencies are combined with all components, using a weighting scheme where the the component which best matches the frequencies is given highest weight. This is done using pseudo-counts, which gradually shifts the emphasis from the mixture to the observed frequencies when more observations are added.

Given the count vector $\vec{n}$ of observations of the amino acids in a column, the estimated (posterior) probability of observing amino acid i in the column is

$$p_i = \sum_{j=1}^{l} Prob(\alpha_j | \vec{n}, \Theta) \times ((n_i + \alpha_{j,i})/(|\vec{n}| + |\alpha_j|)) \tag{3}$$

and is a sum over the components α_j of some Dirichlet mixture Θ. In the product, the first term implements the weighting scheme over the mixture components $\vec{\alpha}$, where the component which best matches the observed counts returns the highest weight. The second term of the product adds the pseudo-counts to the observed counts and normalises the sum. In our work we have used the nine-component Dirichlet mixture from [16].

4.2 Using an Entropy Profile to Build a Pattern

Given a multiple alignment of sequences from a protein family, AMA creates a profile of estimated entropy values. Each value is determined by equation 2, with p_i estimated according to equation 3. To build patterns from the profile, the low entropy columns are chosen, and corresponding pattern elements generated. The individual elements are combined into a complete pattern, which is meant to reflect the conserved positions in the sequences.

A *pattern* is a description of common syntactic features of a set of sequences [5]. A sequence is matched by the pattern if it contains the features described by the pattern. A pattern is diagnostic for a family if it matches every sequence in the family, and no other known sequences [5]. Many of the patterns in PROSITE are diagnostic, but the majority gives some false positives or negatives. It is potentially possible to improve on their diagnostic power by using an automated algorithm for their construction, since such an algorithm may be able to explore a much larger set of candidate patterns.

Patterns in the PROSITE syntax can be expressed on the form

$$E_1 - x(i_1, j_1) - E_2 - x(i_2, j_2) - ... - E_n$$

where each E_k is an element, and each $x(i_k, j_k)$ is a wild-card. An element specifies a single amino acid (e.g. L) or a set of amino acids (such as $[LIV]$). A wild-card specifies an arbitrary stretch of amino acids, the length of which must be at least i_k and at most j_k. For $i_k = j_k$ the shorter notation is $x(i_k)$, and $x(1)$ can be written x. Example patterns are shown in figure 4.

C-REX incrementally creates patterns of increasing complexity by adding elements from the alignment, in order of increasing entropy. An individual element is created simply by enumerating the observed symbols from the corresponding alignment column. To create a complete pattern from n elements, $n - 1$ wild-cards must be added to connect the elements. For a wild-card $x(i, j)$, the values of i and j are determined by counting the minimum and maximum number of residues occurring between the two columns in the alignment.

By testing the patterns generated by C-REX against SWISSPROT, we find the most general pattern with maximum specificity. Testing starts with the most general pattern, and stops if a pattern excludes all false positives. If no such pattern is found, the most general pattern among those with the minimal number of false positives is used during the subsequent generalisation phase.

4.3 Generalising a Pattern

A pattern minimizing the number of false positives is often too specific to match all family members. This because C-REX builds patterns using an alignment in which only a subset of the family is represented. Therefore, the generalisation program G-REX is used to find a more general pattern, matching all members of the family. G-REX currently uses three classes of operations for generalisation, where two operate on elements and one on wild-cards:

- An element can be generalised by adding one new symbol.
- An element can be excluded from the pattern.
- A wild-card region can be expanded by decreasing the lower bound on the length (or increasing the upper bound).

The generalisation phase currently involves only testing of randomly applied generalisations. A generalisation is tested by updating the pattern and testing the specificity and sensitivity. Unless both measures improve or stay the same, the generalisation is undone. This is repeated for a given number of trials, resulting in a gradually more general pattern. The current version, in other words, performs a crude form of search in the space of possible generalisations.

For each pattern there is in fact a very large number of possible generalisations. Consider a pattern of n elements, $E_1, .., E_n$. Let s_k denote the number of symbols in the element E_k. Since the cardinality of the alphabet is 20, there are 2^{20-s_k} possible generalisations of E_k. In addition, every wild-card $x(i_k, j_k)$ has $i_k + (j_{max} - j_k)$ possible generalisations, where j_{max} denotes an arbitrary upper bound on the j-values. For a pattern of n elements, there are

$$\sum_{k=1}^{n} 2^{20-s_k} * \sum_{k=1}^{n-1} i_k * (j_{max} - j_k) \tag{4}$$

possible generalisations. Our restricted operators for generalisation (e.g. only adding one symbol at a time) impose a partial ordering, and thus constrains the search. Also, since G-REX keeps any generalisation which does not degrade the results, the algorithm is a simple hill-climber. Although G-REX often finds patterns with improved sensitivity, it is clear from our results that the search performed by G-REX in the space of possible generalisations is sub-optimal, and that there is potential for improvement of the generalisation phase.

5 Results and Conclusions

We tested our method on 439 families represented in both Pfam and PROSITE. For each family, we derived one pattern using our method, and compared it with the corresponding PROSITE pattern. For families where PROSITE reports more than one pattern, we chose only the 'best' pattern. Figure 3 shows a comparison between our patterns and those in PROSITE. For 87% of families our pattern had equal or better specificity than the PROSITE pattern, which is certainly a satisfactory result. However, our patterns had equal or better sensitivity for only 25% of the families. Further work is therefore needed to improve our results on the sensitivity measure. In analysing the worst-performing patterns, we have discovered some errors in the PROSITE documentation, resulting in family

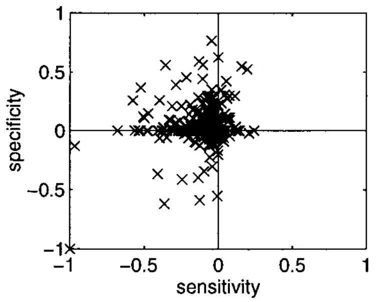


Fig. 3. Comparison of sensitivity and selectivity between our patterns and those in PROSITE, for 439 protein families. Positive values denote that our pattern was better than the PROSITE pattern.

members being treated as false positives by our system. However, no definitive conclusions can be drawn about the effect of the errors on the sensitivity.

Given the size of the search space, the current algorithm for searching generalisations is clearly inadequate. Currently, the sensitivity only improves from 0.883 to 0.906 (the average sensitivity of the Prosite patterns is 0.957). The hill-climbing done by G-REX will easily get trapped in local optima. We are currently investigating improved search algorithms for this purpose.

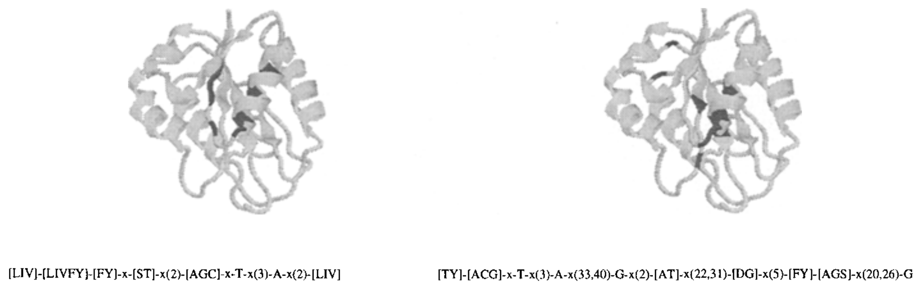

[LIV]-[LIVFY]-[FY]-x-[ST]-x(2)-[AGC]-x-T-x(3)-A-x(2)-[LIV]
[TY]-[ACG]-x-T-x(3)-A-x(33,40)-G-x(2)-[AT]-x(22,31)-[DG]-x(5)-[FY]-[AGS]-x(20,26)-G

Fig. 4. 3D structure of FLAV_ANASP, and comparison of the location of the residues corresponding to the PROSITE pattern (left) and our pattern. Although our pattern spans a longer subsequence, the residues matched by the pattern are located closely in 3D space, in the region involved in phosphate binding.

In examining our patterns, we noted that they are typically longer than those in PROSITE - modeling conserved alignment columns distributed over a larger part of the alignment (an example is shown in figure 4). By examining example 3D conformations from the PDB database, we hypothesise that our patterns include more of the structural constraints of the conformation. This is a possible reason that some of our patterns discriminate better than PROSITE patterns.

While local properties of an active site, modeled by PROSITE, may occur in an unrelated protein with another principal function, these properties would occur within another conformational context in the unrelated protein. Thus, our patterns model local details as well as global, conformational constraints.

As part of our future work, we aim to do a detailed analysis of patterns for a small number of families, to determine the exact differences between our patterns and PROSITE's. Such an analysis may provide further insight into what properties of the sequences our patterns model, which may result in improvements of the algorithm for pattern discovery and refinement. In addition, it may provide biologically interesting insight into the functionality of the proteins. It is conceivable that the patterns discovered by our system correspond to new discoveries of the proteins' functionality. This part of the analysis, may require laboratory experiments as a complement to statistical and computational modeling.

References

1. Krogh A., Brown B., Mian I.S., Sjölander K., and Haussler D. Hidden markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–31, 1994.
2. A. Bairoch and R. Apweiler. The SWISS-PROT protein sequence data bank and its supplement TREMBL. *Nucleic Acids Research*, 25:31–6, 1997.
3. A. Bairoch, P. Bucher, and K. Hofmann. The PROSITE database, its status in 1997. *Nucleic Acids Research*, 25:217–221, 1997.
4. D.A. Benson, M.S. Boguski, D.J. Lipman, J. Ostell, and B. Ouellette. GenBank. *Nucleic Acids Research*, 26(1):1–7, 1998.
5. A. Brāzma, I. Jonassen, I. Eidhammer, and D. Gilbert. Approaches to the automatic discovery of patterns in biosequences. Technical Report 113, Dept. of Informatics, Univ. of Bergen, 1993.
6. T.E. Creighton. Protein folding. In R.A. Meyers, editor, *Molecular Biology and Biotechnology: A Comprehensive Desk Reference*. VCH Publishers, 1995.
7. EMBL nucleotide sequence database: Release notes, release 53, December 1997.
8. L. Hunter. Molecular biology for computer scientists. In L. Hunter, editor, *Artificial Intelligence and Molecular Biology*. AAAI Press/MIT Press, 1993.
9. K. Karplus. Evaluating regularizers for estimating distributions of amino acids. In C. Rawlings, D. Clark, R. Altman, L. Hunter T. Lengauer, and S. Wodak, editors, *Proc. of ISMB95*. AAAI Press, 1995.
10. K. Laurio. Probabilistic modeling of protein families. Master's thesis, University of Skövde, Sweden, 1997.
11. NCBI News. NIH Publication No. 95-3272, September 1995.
12. B. Rost. Learning from evolution to predict protein structure. In *Biocomputing and Emergent Computation - Proceedings of BCEC97*. World Scientific, 1997.
13. B. Rost and R. Schneider. Pedestrian guide to analysing sequence databases. In K. Ashman, editor, *Core Technologies in Biochemistry*. Springer, 1997.
14. C. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27, 1948.
15. K. Sjölander. Bayesian evolutionary tree estimation. In *Proceedings of the Computing in the Genome Era conference*, Washington DC, March 1997.
16. K. Sjölander, K. Karplus, M. Brown, R. Hughey, A. Krogh, I.S. Mian, and D. Haussler. Dirichlet mixtures: A method for improved detection of weak but significant protein sequence homology. *CABIOS*, 12(4):327–45, 1996.
17. E.L.L. Sonnhammer, S.R. Eddy, E. Birney, A. Bateman, and R. Durbin. Pfam: Multiple sequence alignments and hmm-profiles of protein domains. *Nucleic Acids Research*, in press, 1998.
18. E.L.L. Sonnhammer, S.R. Eddy, and R. Durbin. Pfam: A comprehensive database of protein domain families based on seed alignments. *Proteins*, 28:405–20, 1997.