BIOINFORMATICS

Vol. 20 no. 6 2004, pages 847–853
DOI: 10.1093/bioinformatics/btg492

Fold-specific substitution matrices for protein classification

R. B. Vilim[1,], R. M. Cunningham[2], B. Lu[3], P. Kheradpour[3] and F. J. Stevens[1]*

[1]*Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439, USA,* [2]*Computer Engineering Department, University of Cincinnati, Cincinnati, OH 45221, USA and* [3]*Computer Science Department, University of Illinois, Urbana, IL 61801, USA*

Received on December 4, 2002; revised on October 15, 2003; accepted on October 16, 2003
Advance Access publication February 5, 2004

ABSTRACT

Motivation: Methods that focus on secondary structures, such as Position Specific Scoring Matrices and Hidden Markov Models, have proved useful for assigning proteins to families. However, for assigning proteins to an attribute class within a family these methods may introduce more free parameters than are needed. There are fewer members and there is less variability among sequences within a family. We describe a method for organizing proteins in a family that exhibits up to an order of magnitude reduction in the number of parameters. The basis is the log odds ratio commonly used to measure similarity. We adapt this to characterize the sequence dissimilarities that give rise to attribute differentiation. This leads to the definition of Class Attribute Substitution Matrices (CLASSUM), a dual of the BLOSUM.

Results: The method was applied to classify sequences hierarchically in the lambda and kappa subgroups of the immunoglobulin superfamily. Positions conferring class were identified based on the degree of amino acid variability at a position. The CLASSUM computed for these positions classified better than 90% of test data correctly compared with 35–50% for BLOSUM-62. The expected value for a random matrix is 14%. The results suggest that family-specific data-derived substitution matrices can improve the resolution of automated methods that use generic substitution matrices for searching for and classifying proteins.

Contact : rvilim@anl.gov

INTRODUCTION

In recent years, orders of magnitude increases in available sequence data and computer power have transformed the way biologists approach sequence analysis. Ten years ago, compiling a substitution matrix required admitting sequence data from many families to assure a sufficient number of sequences. Since then, with the explosion of sequence data, there is reason to re-think the definition of a substitution matrix. Not only do many families now have a sufficient number of well-annotated sequences that one can compute a family-specific substitution matrix for similarity searches, but other family-specific tasks can also be addressed. In this regard, we define substitution matrices that encode those amino acid interchanges important to determining protein attribute (e.g. function, dysfunction and stability) for use in classifying protein sequences. Development of methods for discovery of primary and secondary structure features shared by proteins related by attribute (e.g. function, dysfunction and stability) is an important problem in computational biology (Chou, 2002).

A goal of our work is, given the identity of the family to which a new protein belongs, to place the protein in the hierarchy for that family. The placement of a protein in the proper family is addressed elsewhere (Gracy and Patrick, 1998; Linial *et al*., 1997; Sonnhammer *et al*., 1997; May, 2001; Silverstein *et al*., 2001; Heger and Holm, 2000; Enright *et al*., 2002). Given that some substitutions of amino acids within a family are associated with change in attribute value, we expect a type of substitution matrix might be all that is needed to represent hierarchical relationships within a family.

In this paper, we extend the substitution matrix, through the log odds ratio that underlies it, for use beyond the identification of similarity among sequences. We define a dissimilarity matrix as a dual to the similarity matrix. The former encodes substitutions that give rise to differentiation of protein attribute. We show how this matrix is computed from protein sequence data labeled by attribute.

METHODS

Substitution matrices

The substitution matrix made its first appearance more than 20 years ago (Schwartz and Dayhoff, 1978) and now lies at the core of many sequence analysis methods. Its definition is statistical (Karlin and Altschul, 1990; Henikoff and Henikoff, 1992; Altschul, 1991). Each element is the logarithm of the ratio of two probabilities associated with a pair of amino acids

*To whom correspondence should be addressed.

at a particular position among a set of sequences. The ratio is the observed probability of occurrence of the pair among all sequences divided by the probability expected by chance. Karlin and Altschul (1990) developed a theory that suggests that a substitution matrix of a given value implies a frequency of occurrence of amino acid pair i and j in high-scoring sequence segments. The substitution matrix element for pair ij in these high-score segments is $S_{ij} = \log_a q_{ij}/(p_i p_j)$. Here q_{ij} is referred to as a target frequency and is the probability of finding pair ij in the high score sequence segments and p_i is the chance occurrence of amino acid i in a typical sequence. This result is the basis of the method used to compute BLOSUM-62 (Henikoff and Henikoff, 1992).

Substitution matrices for classification of sequences by attribute

The classification-by-attribute problem is, to a degree, related to the grouping-by-similarity problem addressed by Henikoff. In each, one seeks to identify a particular, albeit different, set of amino acid substitutions. In the case of similarity, it is those substitutions that occur among similar proteins. In the case of classification, it is those substitutions that give rise to a change in attribute. In the former, the substitutions can be identified by computing a log odds matrix from protein blocks (Henikoff and Henikoff, 1992). In the latter, the substitutions can be identified by an analogous procedure we describe here. We focus on a class of positions just as Henikoff and Henikoff (1992) did in the development of BLOSUM. However, rather than those positions whose amino acid type tends to be conserved across the fold, we identify those positions whose amino acids vary in a manner that gives rise to different sequence attributes within a family.

Suppose, we know *a priori* for a set of proteins which positions are important to conference of attribute. In those positions, there will be a high frequency of certain substitution pairs for which each of the amino acids in the pair tends to be associated with a protein class different from that of the other amino acid. There will be a low frequency of those substitution pairs where one amino acid over the other does not affect the class to which the protein belongs. We quantify this through a log odds ratio. Assume we have a collection of proteins, each protein belonging to one of n classes, and that the proteins have been grouped by class. Only some positions across the proteins are important to class discrimination; delete the rest. Count the number of occurrences, v_{ij}^p, of amino acid pair ij at a position p over all possible protein pairs subject to no pair being in the same class. Sum the v_{ij}^p over all positions to obtain v_{ij}. Compute q_{ij}, the observed probability of occurrence of pair ij, $v_{ij}/\sum\sum v_{ij}$. Next compute the chance probability of pair ij. The probability of occurrence of the i-th amino acid in pair ij is

$$p_i = q_{ii} + \sum_{j \neq i} q_{ij}/2,$$

while the chance probability of occurrence e_{ij} for ij is $p_i p_j$ for $i = j$ and $p_i p_j + p_j p_i = 2 p_i p_j$ for $i \neq j$ (Henikoff and Henikoff, 1992). The log odds ratio for amino acid pair ij is $S_{ij} = \log_2 (q_{ij}/e_{ij})$, $i \neq j$ and $S_{ij} = 0$, $i = j$. This log odds ratio serves as the basis for measuring dissimilarity of attribute among sequences. It has a large value for those substitution pairs where one of the amino acids is found in one class to the exclusion of the other and vice versa. It will have a smaller value if the two amino acids are found more uniformly among classes.

We develop a distance measure consistent with the notion that attributes are differentiated in terms of preferred substitutions at a subset of positions in the sequence. Let the distance between two amino acid sequences u and v be given by

$$d(u, v) = \sum_{p=1}^{n} S[\mathrm{aa}(p, u),\ \mathrm{aa}(p, v)]w(p), \qquad (1)$$

where $\mathrm{aa}(p, u)$ is the amino acid in position p of sequence u of total length n; $S[i, j]$, the element ij of matrix $S(= S_{ij})$; and $w(p)$, the position weight.

A position weight is unity if the position is the site of a substitution pair that confers class and near zero otherwise. Note S is a matrix whose elements are constants and the row and column indexes identify one of 20 amino acids. S depends only on amino acid pair and not on position or on sequence identity. This substitution matrix is to be derived for a specific family and, within a family, for a specific attribute (e.g. function). We refer to these as Class Attribute Substitution Matrices (CLASSUM). One sees that $d(u, v)$ will be large if the sequences u and v belong to different classes and small if they belong to the same class. Thus, we have a basis for differentiating among proteins of different attributes.

Classification algorithm

The above measure of sequence dissimilarity is used to sort new protein sequences according to attribute. These sequences are assumed to belong to a family for which we have previously computed the CLASSUM from sequences of known attribute.

The single linkage or nearest neighbor rule (Arabie, 1996) is used to agglomerate sequences into clusters. Initially, a cluster contains a single sequence so that the number of clusters equals the number of sequences. The distance between all pairs of sequences is calculated from Equation (1) once and is stored in a table. For each pair of clusters we find the pair of sequences, one sequence in one cluster and the other in the other cluster, that are nearest and define this as the distance between the clusters. The pair of clusters with the shortest distance between them is merged. The agglomeration process continues until all data are clustered into a single cluster. The tree of linkages generated in this process is stored as a linked list. The linked list is used to retrieve the members of a cluster. The single linkage rule has the advantage that distances need

be calculated only once, before the first agglomeration. Other agglomeration methods require an average or representative member to be identified after each agglomeration followed by updating the distances. With single linkage we have achieved classification rates better than 90% suggesting that skewness is not adversely affecting performance.

Clusters must be scored with respect to purity of composition to establish their identity. The clusters are labeled by class in a way that yields the highest percentage of proteins of class i in that cluster with that label. To do so, cut the dendogram at the elevation where there are n clusters (for n classes). The set of clusters so obtained is labeled $\mathrm{C}^0 = \{\mathrm{C}l_1, \mathrm{C}l_2, \ldots, \mathrm{Cl}_n\}$. Define $f_i(\mathrm{Cl}_j)$ as the fraction of proteins in cluster j that are of type i and x_i as the fraction of proteins in the full set that are of type i.

The score for labeling as such is taken as

$$F = \prod_{i=1}^{n} f_i(\mathrm{Cl}_i)x_i. \qquad (2)$$

The labeling of the clusters must reflect the contents of the clusters. The following rules provide for this:

Rule 1: The number of clusters must equal the number of distinct classes. Outliers identified in the initial clustering are removed from the dataset for the fundamental reasons that deem them outliers.

Rule 2: Each class label can be assigned to at most one cluster.

Rule 3: If several clusters are pure and of the same type, then the cluster with the most proteins is assigned the type. The other clusters take left over unused labels.

As a consequence of this rule choice, we have:

> The splitting of a cluster of one class of proteins into more than one cluster is penalized. This discourages differentiating among proteins of one class.
>
> There is no need to put a tentative class label on a cluster other than those that occur amongst the proteins in the cluster (e.g. the tentative label λ1a need not be considered for a cluster if there are no proteins of type λ1a in the cluster). This eases the combinatorial blowup as the number of clusters grows.

The 'best' labeling of the clusters is the one that gives the maximum score in Equation (2), subject to the rules above. Unity score or perfect score occurs when each cluster has only proteins from one class. This is the classic assignment problem which we presently solve by brute force. Note that, in general, the scheme above will lead as intended to a labeling of clusters that reflects the predominate class in each cluster.

This labeling process is used in the classification of a previously unseen protein. Classification is performed as follows. A trained system is defined as consisting of the sequences used to compute the substitution matrix (exemplars), their class tags and the exemplar-derived substitution matrix and position weight values. A new dataset composed of the unseen protein and the exemplar protein dataset is clustered using the substitution matrix and position weights. If there were n classes, then this dendogram is cut at n clusters. These clusters are then labeled according to the most predominant class of exemplars in each cluster using the scheme described above. The identity of the unseen protein is then taken from the label assigned to the cluster it appears in.

Attributes as constraints

The method during training attempts to find the values for the substitution matrix and position weights that classify the training set sequences according to the values of their attribute tags. Assume that there are n different tag values. The object is to find the similarity matrix and position weights in Equation (1) that give the best assignment of training exemplar sequences among n clusters where each cluster takes one of the attribute labels according to the cluster labeling scheme. In a perfect assignment, all sequences of type i would reside in a single cluster with label type i with this true for all i. The corresponding values of the similarity matrix and position weights would maximize the fitness function given by Equation (2). During training, we cluster repeatedly, each time with new values for the similarity matrix and position weights. The iteration process stops when those values are found that place all proteins of type i into a cluster whose label is i (as determined by the rules above) or until the assignment score reaches a maximum value, something less than unity. The attribute tags act as constraints that imply a value for the substitution matrix and weight vector. The number of free parameters is roughly 500.

Genetic algorithm

Two idealizations inherent in the log odds method above are relaxed to obtain improved classification performance. In the above, a binary weight vector identifies those positions that confer class while a substitution matrix identifies the amino acid pairs that determine class. However, in practice the importance of a position to class conference lies along a continuum while sequences within the same class exhibit preferred substitutions different from those related to attribute differentiation. We include a treatment of these refinements by replacing the former log odds criterion with a constraint criterion that when satisfied partitions the sequences according to their attribute tags. By iterating over values for the weights and substitution matrix using a genetic search algorithm we find values for these parameters that satisfy the constraint.

The problem is cast as a genetic search which is less prone to converging to a local minimum and is well suited for multi-processor computation. The fitness function is given by Equation (2) and its associated cluster labeling scheme. The similarity matrix and position weights are encoded as binary

strings. The similarity matrix is a symmetric square matrix with 20 amino acids along the column and row dimension defining all possible pairs of substitutions. A 21st position represents the gap/amino acid substitution pair. A single parameter is used to represent an unknown amino acid.

A binary string or 'chromosome' is formed by concatenating the binary string equivalent of each of the values of the elements of the substitution matrix and position weight vector. Eight-bit encoding is used giving a resolution of one part in 256. A 'population' consisting of a set of 100 chromosomes is constructed. The 'fitness' of each chromosome is evaluated for its ability to classify the sequences as measured by the value of the fitness function. Each generation we select 0.1 of the population using tournament selection and apply two-point crossover with a crossover rate of 0.85. The mutation rate among the new population is 0.01. If this population does not contain an individual with at least the fitness of the elite member, then a member is replaced at random with the elite member.

Table 1. Distribution of antibody light chain amino acid sequence data

κλ (288)	κ (156)	κ1 (85)	κ1a (25)
			κ1b (38)
			κ1c (22)
		κ2 (15)	κ2b (15)
		κ3 (28)	κ3a (13)
			κ3b (15)
		κ4 (28)	κ4 (28)
	λ (132)	λ1 (36)	λ1a (9)
			λ1b (20)
			λ1c (7)
		λ2 (36)	λ2a (12)
			λ2b (20)
			λ2c (4)
		λ3 (49)	λ3a (17)
			λ3b (12)
			λ3c (20)
		λ4 (11)	λ4 (11)

Sub-subgroups and numbers of sequences in each from which our 10-fold cross-validation datasets were assembled.

DATA

The antibody light chain variable domain (Stevens *et al*., 1995; Stevens, 1999), a subset of the immunoglobulin superfamily of proteins (Halaby *et al*., 1998,1999) was adopted for our preliminary studies. The immunoglobulin superfamily is perhaps the most studied and understood as evidenced by the human genome project (Lander, 2001; Venter, 2001). Our model system, the light chain variable domain (V_L), is grouped in two different classes, κ and λ, which are encoded by clusters of related germline genes on two chromosomes. Humans have approximately 15 functional κ and 25λ functional V_L domains all of which have varying degrees of genetic difference. To an extent, these germline genes can be thought of as corresponding to a family of homologous proteins with similar, but distinct functions. In addition to inherited variations, the normal process of immune system diversification results in the accumulation of numerous somatic mutations.

As for the choice of the light chain variable domain as our model system, we consider this system a good proxy for families of evolutionary related proteins that exhibit diversified function, families for which are method is intended. Depending upon the relationships between the germline genes of origin, comparisons of amino acid sequences of protein from two patients would usually find 40–70 amino acid differences out of the approximately 120 amino acids in the V_L domain. (i.e. 45–65% sequence identity). As a rule-of-thumb sequences in families other than the immunoglobulins exhibit about 40% sequence identity. So the light chain variable domain is representative of the families we seek to classify. Our interest in κ and λ functional domains stems from earlier work on the role of amino acid mutations in disease (Stevens *et al*., 1995; Stevens, 1999) and the applicability of our method to the detection of function impairing single nucleotide polymorphisms (SNPs). The appearance of an amino acid differing from the ones associated with attribute differentiation at a position with a large weight signals an increased probability of such an SNP.

We compiled a database of sequences for the V_L domain. We began with a raw dataset consisting of some 740κ and λ entries taken from earlier studies of patients with plasma cell diseases. These sequences had been previously aligned manually in a generalized template 120 amino acids long to accommodate a variety of loop inserts found in some germlines. This set was reduced to a smaller final working dataset by eliminating sequences with more than 13 unknowns and sub–subgroups with fewer than four sequences. The distribution of resulting sequences is shown in Table 1.

Training and test datasets

Statistical analysis of the prediction accuracy of the classification algorithm is evaluated using training-set and test-set data (Chou and Zhang, 1995). We use the M-fold cross-validation method of testing, widely used for gauging prediction accuracy (Ambroise and McLachlan, 2002). The data are divided into M non-overlapping subsets of the same size. The classifier is trained on all but one with this last one used for prediction. This training–testing process then cycles through the remaining non-overlapping subsets. The average prediction accuracy is calculated from all M training–testing sessions.

The enormity of the computing task limited us to 10-fold ($M = 10$) cross-validation. A discussion on 10-fold versus leave-one-out is given in Ambroise and McLachlan (2002). Two training-test datasets were generated from data shown in Table 1, one with 132 sequences from the λ sub-subgroup data (i.e. λ1a, λ1b, . . . , λ2a, . . .) and one with 150 sequences from the κ sub-subgroup data. The sequences from each were

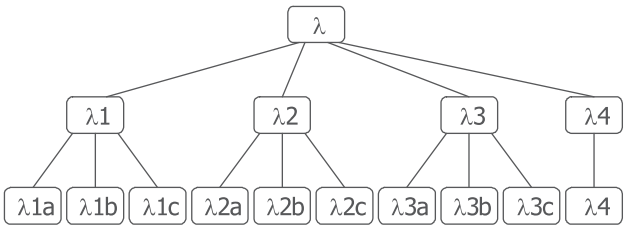


Fig. 1. Hierarchical organization by germline of λ antibody light chain sequences. The subgroup level contains the λ1, λ2, . . . while the sub-subgroup level contains the λ1a, λ1b, . . . , λ2a, ...

Table 2. Classification accuracy on sub-subgroup on 10-fold cross-validation datasets for three different substitution matrices

Dataset	BL-62	Log odds		Genetic algorithm	
		Train	Test	Train	Test
1	0.30	0.77	0.73	0.98	0.91
2	0.36	0.78	0.73	0.97	0.91
3	0.30	0.62	0.61	0.97	1.00
4	0.39	0.77	0.64	0.98	0.91
5	0.45	0.78	0.73	0.73	0.85
6	0.48	0.62	0.64	0.96	0.94
7	0.39	0.65	0.70	0.98	0.97
8	0.42	0.76	0.70	0.96	0.97
9	0.39	0.53	0.61	0.84	0.88
10	0.36	0.64	0.82	0.84	0.85
AVG	0.39	0.69	0.69	0.92	0.92

(a) BLOSUM-62 with unity weights, (b) log odds substitution matrix with weights nulled for non-class conferring positions and (c) attribute constrained substitution matrix and weights. Performance on test data improves as one progresses from (a) through (c).

first randomized and then divided into 10 non-overlapping sets by assigning the first one-tenth of the proteins of each sub-subgroup encountered to the first non-overlapping set. The next one-tenth went into the second non-overlapping set, and so on.

RESULTS

Reference case

The reference case involves the λ sub-subgroup training-test dataset described above. The hierarchical ordering of classes is shown in Figure 1. The task was to find a similarity matrix and position weight vector from the training data such that any sequence in the test data is correctly classified down to the sub-subgroup level (e.g. λ2b).

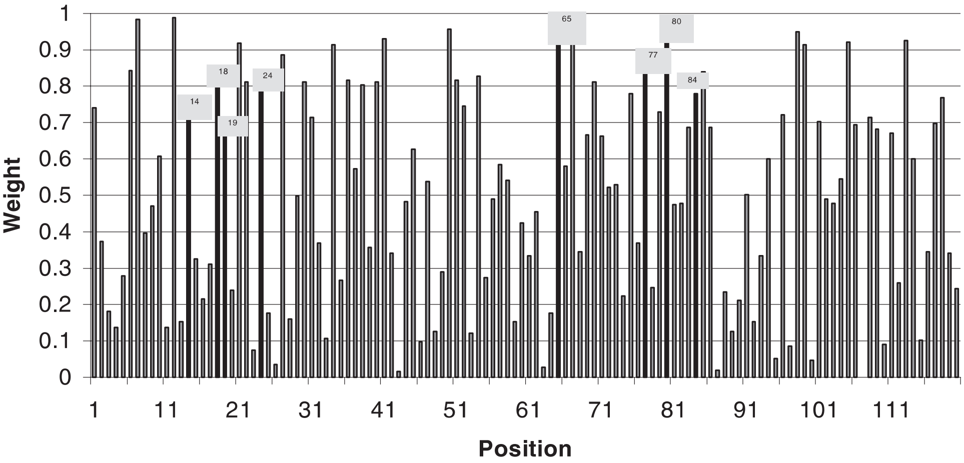
The performance results are summarized in Table 2 for a progression of increasingly refined substitution matrix and weights: BLOSUM-62 with unity weights; log odds substitution matrix with weights nulled for non-class conferring positions; and attribute-constrained genetic-algorithm derived substitution matrix and weights. The genetic-algorithm approach gives the best performance averaging 92% on both training and test data over the 10-fold cross-validation. The log odds approach in turn did better than BLOSUM-62. This suggests that at those positions that confer attribute, the substitution pair values from BLOSUM-62 are very unrepresentative of those that differentiate among attribute at the λ sub-subgroup level. Each of the 10 entries in Table 2 was obtained by averaging the performance of a series of classifiers based on the series of substitution matrices output during a training session. The high classification rate on test data suggests that each attribute tag correlates highly with a unique set of amino acid features in the sequence. The results also suggest that the assignment by germline (i.e. tagging the sequence as to sub-subgroup) is well established for V_L proteins.

The value of the position weight vector can be interpreted to identify sequence positions that determine attribute. In general, three factors influence the values taken on by the weights. The first is the variability in the amino acid at a particular position as a consequence of evolutionary divergence between related proteins (Dokholyan and Shakhnovich, 2001). From the standpoint of class differentiation the variability is noise and the training procedure will assign a small value to this weight since such a position does not influence class differentiation. The second factor is the variability at a fold-conserved position. There the amino acid is constant across the whole family. The weight ought to be set to zero in advance (which we did) if one knows *a priori* that the residue at that position is conserved. The third factor involves a position that is important for class differentiation in the family. The magnitude of the weights is plotted as a histogram in Figure 2 with sequence position on the abscissa. Shown below the histogram are sequences taken at random from two of the sub-subgroups, λ1a and λ2a. Blanks in the sequences indicate the amino acid at that position is the same as in the first sequence that appears for the sub-subgroup. Positions with the same amino acid for the sequences in λ1a but a different amino acid for λ2a have been marked with stars in the sequences and appear as shaded bars in the histogram. One sees that large weights, as expected, are associated with positions that determine attribute.

Relative contribution of weights

A set of calculations that paralleled those described for the λs was performed for the κs with additional calculations intended to identify the contribution of weights to classification performance. The class hierarchy, as in the case of the λs, is down to the sub-subgroup.

The results are consistent with those obtained for the λs. Table 3 shows that the BLOSUM-62 matrix with unity weights again performed poorly, classifying only 47% of the test data correctly. The classifier performance improves when the log odds method is used to compute the substitution matrix and improves further when the genetic-algorithm method is used. The latter method results in 98% of the training data classified correctly and 95% of the test data. If the weights are held at unity and the matrix is calculated, the performance on test




```
123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890  label   class
             *   **    *                                        *           *  *   *
QSVLTQPPS.VSAAPGQKVTISCSGSSSDMG....NYAVSWYQQLPGTAPKLLIYENNKRPSGIPDRFSGSK..SGTSATLGITGLWPEDEADYYCLAWDTSPRA..-------------  scwl1a  lam1a
  V    P                 G TNI      NY   H H           EDN         I A           A    RTG       ATWDSSLNA..VVFGGGTKVTVLG  new     lam1a
  -    -                 S SNI      YF   Y Q           DSD         F G           D    QTG       GTWDN LRA..MVFGGGTTLTVL-  igl3    lam1a
  V    P                 S SDM      YA   Y Q           ENN         F G           G    WPE       LAWDT PRA..-------------  GL1A1   lam1a
  V    P                 S SNI      NY   Y Q           DNN         F G           G    QTG       GTWDS LSA..-------------  GL1A3   lam1a
-SALTQPAS.VSASPGQSITISCTGTTNDIGS...YSYVSWYQQYPGKAPKVLIFDVNSRPSGVSHRFSGSK..SGNTASLTISGLQAEDEAHYFCSSYRTSGT...IIFGGGTYVTVLR  tog     lam2a
-S          G             SSDV     YNF    Q HP  A  LI YDVTY    I S    SR..   T         A    D Y SS TSDST.  RLFGGGTKLTVLR  wh      lam2a
QS          G             SSDI     YDY    E HQ  A  LM YDVNN    V N    SK..   T         A    D Y SS TSSSTL  LVSAGGNMLTVLG  fks13   lam2a
QA          G             SNN.     LHF    Q HP  V  IL YEVSK    V N    FK..   A         S    D Y GL VGDRL.  WVFGGGTRLTVLQ  bou     lam2a
QS          G             SSDV     YNL    Q HP  A  LM YEGSK    V N    SK..   T         A    D Y CS AGSSTL  -------------  GL2A1   lam2a
```

Fig. 2. Histogram of position weights from λ sub-subgroup genetic algorithm training session. Positions that differentiate between λ1a and λ2a sequences identified by inspection are indicated by stars below sequence position numbers. Histogram of position weights shows large weights (numbered shaded vertical bars) coincide, as expected, with these same positions.

Table 3. Classification accuracy on sub-subgroup on 10-fold cross-validation datasets for three different substitution matrices

Dataset	BL-62	Log odds		Genetic algorithm			
				Weights = 1		Free weights	
		Train	Test	Train	Test	Train	Test
1	0.40	0.84	0.87	0.99	0.93	0.97	1.00
2	0.33	0.86	0.87	0.99	0.87	0.99	1.00
3	0.60	0.60	0.60	0.98	0.87	1.00	0.93
4	0.53	0.73	0.73	0.98	1.00	0.99	1.00
5	0.47	0.62	0.67	0.98	1.00	0.98	1.00
6	0.33	0.59	0.53	0.99	0.93	0.99	1.00
7	0.40	0.78	0.80	0.99	0.93	0.98	0.80
8	0.53	0.84	0.73	0.99	0.87	0.97	0.93
9	0.47	0.70	0.80	0.99	0.87	0.97	0.93
10	0.60	0.87	0.80	0.98	1.00	0.97	0.93
AVG	0.47	0.74	0.74	0.98	0.93	0.98	0.95

(a) BLOSUM-62 with unity weights, (b) log odds substitution matrix with weights nulled for non class conferring positions and (c) attribute constrained substitution matrix and weights. Performance on test data improves as one progresses from (a) through (c).

data declines to 93%. It appears then that position weights do play a role in improving classifier performance by highlighting positions that are involved in conferring attribute.

Performance with position in hierarchy

The family-specific classifier was then trained one level higher in the λ–κ hierarchy. The intent was to identify any trend in relative performance between the data-driven matrix and BLOSUM-62 with position in the hierarchy. The training task was to find a similarity matrix and position weight vector such that any sequence presented to the classifier is correctly classified as to subgroup (i.e. $\kappa 1, \kappa 2, \ldots \lambda 1, \ldots$).

The classification accuracy of the attribute-constrained classifier at the subgroup level was 94% for the training set data, 92% for the test set data. The BLOSUM-62 matrix performance improved to 69% from the 39% and 47% for the sub-subgroup cases. This is consistent with the notion that the relative performance of family-specific methods, compared against methods based on aggregated data improves with increasing specialization of attribute.

DISCUSSION

We have shown that generic substitution matrices such as BLOSUM are not well suited for classification of proteins within a family. They are encoded for substitutions that are known to preserve homology across many folds rather than substitutions that contribute to attribute differentiation within a family.

Our method requires several hundred free parameters, significantly less than other classification methods such as neural networks (NNs), Hidden Markov Models and Position Specific Scoring Matrices. These methods for a protein 150 positions long have at least several thousand free parameters (Baldi and Brunak, 2001; Krogh, 1994, 1999). That our

classifier with its smaller parameter set performed with >90% correct classification on previously unseen sequences suggests that these other methods may be vulnerable to over-fitting. A consequence is that their generalization performance would be poorer and, hence, classification performance on unseen data would not be as good.

Our agglomerative clustering method provides, as a natural feature, the hierarchical classification sought. The binary dendritic tree underlying the method ensures this. While agglomerative clustering algorithms may produce suboptimal clustering, this is relevant only to the extent it prevents us from achieving training set and test set members grouped such that each group contains only proteins with the same class label. We achieved classification rates on test proteins in excess of 90%. Thus, to the extent suboptimal clustering is occurring, it does not appear to be detrimental. Although cluster analysis methods can become computationally burdensome when datasets number several thousand exemplars (Jain, 1999), since we are computing family-specific matrices, the number of elements in each training set will likely remain below 1000.

One objective is a ready physical interpretation of the values acquired by the parameters of the classifier during training. NN classifiers suffer from difficulty in interpreting the significance of the network weights (Fine, 1999). Support vector machines (Burges, 1998) suffer similarly. For nonlinear kernel functions it is difficult to extract the small number of critical features from a large set that contribute to accurate classification. For a protein 150 positions long and any of 20 amino acids possible at each position, the number of features is 3000. While both NN and SVD methods perform accurate classification, the difficulty in interpreting the network parameters makes the methods less useful. Hidden Markov Models (Baldi and Chauvin, 1996) are more amenable to interpretation of the free parameters. However, with their underlying probabilistic definition, they are a departure from the data-driven non-statistical parameters we seek.

REFERENCES

Altschul,S.F. (1991) Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.*, **219**, 555–565.

Ambroise,C. and McLachlan,G.J. (2002) Selection bis in gene extraction on the basis of microarray gene-expression data. *Proc. Natl Acad. Sci. USA*, **99**, 6562–6566.

Arabie,P. (1996) *Clustering and Classification*. World Scientific.

Baldi,P. and Brunak,S. (2001) *Bioinformatics*. MIT Press.

Baldi,P. and Chauvin,Y. (1996) Hybrid modeling, HMM/NN architectures, and protein applications. *Neural Comput.*, **8**, 1541–1565.

Burges,C.J.C. (1998) Support vector machines for pattern recognition. *Data Min. Knowl. Disc.*, **2**, 121–167.

Chou,K. (2002) A new branch of proteomics: prediction of protein cellular attributes. In Michael, P. Weiner and Quinn Lu (eds), *Gene Cloning and Expression Technologies*. Eaton Publishing.

Chou,K. and Zhang,C. (1995) Prediction of protein structural classes. *Crit. Rev. Biochem. Mol. Biol.*, **30**, 275–349.

Dokholyan,N.V. and Shakhnovich,E.I. (2001) Understanding hierarchical protein evolution from first principles. *J. Mol. Biol.*, **312**, 289–307.

Enright,A.J., Van Dongen,S. and Ouzounis,C.A. (2002) An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res.*, **30**, 1575–1584.

Fine,T.L. (1999) *Feedforward Neural Network Methodology*. Springer.

Gracy,J. and Patrick,A. (1998) Automated protein sequence database classification. I. Integration of compositional similarity search, local similarity search, and multiple sequence alignment. *Bioinformatics*, **14**, 164–173.

Halaby,D.M. and Mornon,J.P.E. (1998) The immunoglobulin superfamily: an insight on its tissular, species, and functional diversity. *J. Mol. Evol.*, **46**, 389–400.

Halaby,D.M., Poupon,A. and Mornon,J.P. (1999) The immunoglobulin fold family: sequence analysis and 3D structure comparisons. *Protein Eng.*, **12**, 563–571.

Heger,A. and Holm,L. (2000) Towards a covering set of protein family profiles. *Prog. Biophys. Mol. Biol.*, **73**, 321–337.

Henikoff,S. and Henikoff,J. (1992) Amino acid substitution matrices from protein blocks. *Biochemistry*, **89**, 10915–10919.

Jain,A.K. (1999) Data clustering: a review. *ACM Comput. Surv.*, **31**, 264–323.

Karlin,S. and Altschul,S.F. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl Acad. Sci. USA*, **87**, 2264–2268.

Krogh,A. (1994) Hidden Markov models in computational biology: application to protein modeling. *J. Mol. Biol.*, **235**, 1501–1531.

Krogh,A. (1999) Hidden neural networks. *Neural comput.*, **11**, 541–563.

Lander,E.S. (2001) Initial sequencing and analysis of the human genome. *Nature*, **409**, 860–921.

Linial,M., Linial,N., Tishby,N. and Yona,G. (1997) Global self-organization of all known protein sequences reveals inherent biological signatures. *J. Mol. Biol.*, **268**, 539–556.

May,A.C.W. (2001) Optimal classification of protein sequences and selection of representative sets from multiple alignments: application to homologous families and lessons for structural genomics. *Protein Eng.*, **14**, 209–217.

Schwartz,R.M. and Dayhoff,M.O. (1978) Atlas of protein sequence and structure. *Nat. Biomed. Res. Found.*, **5**, 353–358.

Silverstein,K.A., Shoop,E., Johnson,J.E. and Retzel,E.F. (2001) MetaFam: a unified classification of protein families. I. Overview and statistics. *Bioinformatics*, **17**, 249–261.

Sonnhammer,E.L.L., Eddy,S.R. and Durbin,R. (1997) Pfam: a comprehensive database of protein domain families based on seed alignments. *Prot. Struct. Funct. Genet.*, **28**, 405–420.

Stevens,F.J. (1999) Protein structure, stability, and conformational disease: human antibody light chains. Technical Report, Argonne National Laboratory. ANL/BIO/99-1.

Stevens,F.J., Myatt,E.A., Chang,C.H., Westholm,F.A., Eulitz,M., Weiss,D.T., Murphy,C., Solomon,A. and Schiffer,M. (1995) A molecular model for self-assembly of amyloid fibrils: immunoglobulin light chains. *Biochemistry*, **34**, 10697–10702.

Venter,J.C. (2001) The sequence of the human genome. *Science*, **291**, 11304–11351.