Identifying structure fragments by sequence alignments to predict protein secondary structures

Haitao Cheng

Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Recommended Citation
Cheng, Haitao, "Identifying structure fragments by sequence alignments to predict protein secondary structures" (2004). Retrospective Theses and Dissertations. 20378.
https://lib.dr.iastate.edu/rtd/20378

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Identifying structure fragments by sequence alignments to predict protein secondary structures

by

Haitao Cheng

A thesis submitted to the graduate faculty in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Robert Jernigan, Co-major Professor
Dimitris Margaritis, Co-major Professor
Vasant Honavar

Iowa State University
Ames, Iowa
2004
Graduate College
Iowa State University

This is to certify that the master's thesis of

Haitao Cheng

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy
# TABLE OF CONTENTS

## CHAPTER 1 INTRODUCTION
- Why predict secondary structure .......................................................... 1
- History, current approaches and achievements ........................................... 2
  - Background ......................................................................................... 2
  - Methods summary ................................................................................ 4
- Motivation of current study ...................................................................... 6

## CHAPTER 2 METHODS
- Brief review of the methods .................................................................... 9
- Dataset selection .................................................................................... 9
- Local sequence alignment using BLAST .................................................. 10
- CB513 prediction and evaluation of prediction .......................................... 11
  - Secondary structure elements interpretation ......................................... 11
  - Weight assignment for matches of segments ........................................ 11
  - Calculation of normalized scores for each residue ............................... 12
  - Parameters adjusted for weight assignments ........................................ 13
  - Prediction based on normalized scores of residues ............................... 17
- Working on the KS267 dataset ................................................................ 18
- Some cases (prediction application) .......................................................... 18
- PDB and SCOP ...................................................................................... 19
  - PDB resource ..................................................................................... 19
  - SCOP resource ................................................................................... 19

## CHAPTER 3 RESULTS
- Q₃ definition .......................................................................................... 20
- Flow chart of prediction and results ......................................................... 20
- Weight assigning method one ................................................................. 20
  - Basic method ..................................................................................... 22
  - Different similarity cutoffs ................................................................ 22
  - Different similarity level zones/regions ............................................... 23
  - Surface area of exposure to solvent of residues .................................... 25
  - The occurrences of matches ................................................................ 25
  - Subfamilies of the matches .................................................................. 26
  - Using SCOP classification to see the prediction for each subclass ....... 27
- Weight assigning method two ................................................................. 28
  - Basic method ..................................................................................... 28
  - Different identity score cutoffs ............................................................. 28
  - Different identity score zones ............................................................... 31
- Normal and reverse direction prediction .................................................. 32
CHAPTER 4 DISCUSSION
The best results obtained
Comparison with other methods
Some influence factors
Weight assigning methods
Substitution matrix
Nonlinear weight assignment scheme for matching occurrences
Solvent accessibility of residues
Sequence direction
Proteins of different types and different lengths
Subfamilies of matching segments
Different ways of translation
Using AI approaches
PSIBLAST and BLAST
Way to improve with an example for analysis
Reliability assessment
Others
Brief conclusion and problems
Lessons learned

APPENDICES
Appendix A. Case Applications
Weight assigning method 1
Weight assigning method 2
Appendix B. Substitution Matrices – PAMs and BLOSUMs
Background
PAMs (Percent Accepted Mutation)
BLOSUMs (Blocks Substitution Matrix)
Appendix C. Artificial Intelligence Approaches
Basic ideas
Running programs
Appendix D. Datasets CB513 and KS267
CB513
CHAPTER 1. INTRODUCTION

Why predict secondary structure

As a result of genome and other sequencing projects, the protein sequence-structure gap is widening rapidly. Prediction of protein structure and function from amino acid sequences is one of the most important problems in molecular biology. One of the ultimate goals of protein science is reliable prediction of tertiary (three-dimensional, 3D) structure of proteins from sequences instead of performing expensive experiments. However, the 3D structure prediction from amino acid sequences is still not as accurate as required. An intermediate but useful step is to predict the secondary structure, which might simplify the complicated 3D structure prediction problem.

Understanding the functions of proteins is based upon an understanding of their structures. Therefore, the study of the relationships between structure and sequence is of great importance. The relationship between sequence and structure makes it possible to construct structure models for newly defined sequences from structure prediction.

For bioinformatists, the challenge is trying to predict the secondary structure of sequences, which is the fundamental first step in predicting the higher-level structure of a protein. This is actually a process of reducing the difficult 3D structure determination to a one-dimensional problem. This reduction is possible since proteins can form local conformational patterns like helices and sheets. Many scientists have shown that predicting secondary structure is the first step toward predicting 3D structure. (Ortiz et al., 1999; Eyrich et al., 1999; Lomize et al., 1999; Chen et al., 1999). Therefore, it is helpful to narrow the sequence-structure gap by successfully predicting secondary structure.

Sometimes, secondary structure prediction can even be useful when a protein structure is already known. Young (Young et al, 1999; Kirshenbaum et al., 1999) reports the correlation between secondary structure prediction and global conditions. They use secondary structure prediction to monitor regions, trying to identify putative allosteric switches, and apply their finding to the myosin family.
Secondary structure prediction can be used to predict some aspects of protein functions. During the prediction, some by-products can be retrieved serving as inputs for some other important bioinformatic procedures. In genome analysis, it is used to help classify proteins, separate domains, do genome analysis, and even recognize functional motifs. (Rost, 2001)

**History, current approaches and achievements**

**Background**

In 1960s and 1970s, secondary structure prediction was based mainly on single amino acid propensities. Until the early 1990s, the main methods still focused on local information of the sequences. That leads to low prediction accuracy, at a level slightly above 60% of \( Q_3 \) (the percentage of correctly predicted residues in one of the three states: helix, strand, and other, see definition in Chapter 3 - \( Q_3 \) definition).

Dating back to late 1970s, scholars began thinking about introducing global information to this procedure (Dickerson *et al.*, 1976; Zvelebil *et al.*, 1987). They are pioneers trying to combine the evolutionary propensities to the prediction, and started a new generation of prediction. Thinking about the fact that during the evolution process, "structure is more conserved than the sequence" (Kloczkowski *et al.*, 2002), it is not unusual to try to combine evolutionary information to the prediction to obtain higher accuracy if the relationship between sequence identity and structure similarity has been fully considered.

Enlarged databases, new searching models and algorithms make it feasible to extend family divergence, i.e. larger and more divergent families are available even when the structures of some of the family members are unknown. It is possible to do large-scale searches with the help of PSI-BLAST (Altschul, 1997) and Hidden Markov Models (Karplus *et al.*, 1998; Eddy, 1998). The PSI-BLAST search tool uses iteration, starting from the safe zone of comparisons, extending to the twilight zone evolutionarily. This way it identifies fairly divergent family members. By doing the multiple alignments, conserved information of long-ranged segments along sequences is likely to be found and thus conserved structure prediction improves. There are alternatives to increase the search divergence to improve the prediction accuracy. Jennings et al. (Rost, 2001) claimed to have improved the prediction
accuracy of ClustalW/HMMer alignment. They similarly started with a safe zone alignment of ClustalW/HMMer, and iteratively refined the prediction through DSC (Discrimination of protein secondary structure class) (King & Sternberg, 1996).

By using divergent profiles, the prediction accuracy reaches 76%. It is commonly believed that the improvement of prediction is due to the growth of the databases, the extended search and other reasons. A study (Przybylski, 2000) on the reasons of improvement was done to separate the contributions from enlarged database and extended searching. They first used standard BLAST against SWISS-PROT, and against SWISS-PROT + TrEMBL + PDB (larger database). Then they compared the different searching methods, standard BLAST and PSI-BLAST against the same large database. Both factors improved the accuracy by approximately 2 percentage points.

The accuracy is also influenced by the secondary structure assignment. It makes the prediction easier to predict only the key elements of secondary structures, namely α-helix, β-sheet and coils. The assignment of DSSP (Database of Secondary Structure in Proteins) (Kabsch & Sander, 1983) is now widely accepted. According to DSSP classification, there are eight elements of secondary structure assignment: H (α-helix), E (extended β-strand), G (3_10 helix), I (π-helix), B (bridge, a single residue β-strand), T (β-turn), S (bend), and C (coil). It is based mainly on hydrogen bonds between the backbone carbonyl and NH groups. But even on the single DSSP assignment, there are different interpretations (Kloczkowski, 2002). Converting 3_10 helices and β-bulges to non-regular structure makes the prediction accuracy higher (Rost, 2001).

Besides the factors mentioned above, good methods or algorithms help to increase the accuracy of prediction. SSpro is an example (Baldi et al., 1999). It applied the idea of recursive neural networks without using more divergent profiles. The comparison of prediction accuracy for different methods (Rost, 2001) proves its success. NNSSP (Salamov, 1997) applied the idea of nearest neighbor. The procedure is, first, construct the library of local stretches of residues, and second, assemble the local structure motifs according to Hidden Markov Model.
Methods summary

There exist 4 main methods for secondary structure prediction: empirical statistical methods, nearest neighbor methods, methods that use hidden Markov models, and neural network methods.

**Empirical statistical methods**

There are two representative algorithms, Chou-Fasman and GOR methods.

The Chou-Fastman method is the first popular algorithm used to predict the secondary structure of globular proteins (Chou & Fasman, 1974). It first used the very small dataset available at that time. Based on the frequencies of appearance of each residue in α-helix and β-strand of the dataset, each amino acid was assigned a weight value. Accordingly, the sequences were classified into six categories. A cutoff value was set. For a query sequence, the algorithm scanned the sequence, and found the regions with values lower than the cutoff values. These regions were regarded as regions with high probability of being in α-helix or β-strand. Other regions were coils. Larger test datasets in later research proved its prediction accuracy to be around 50%. More complex algorithms were then developed such as the one by Lim (Lim, 1974). Its difference in comparison with Chou-Fasman is that it classified the sequences into groups according to size, hydrophobicity, and conformational flexibility instead of single value assignments.

GOR (Garnier, Osguthorpe and Robson, 1978) is another representative. This algorithm has been updated several times. It uses the principles of information theory. The newer versions improve the prediction accuracy to above 71% using pair-wise statistics of residues beyond nearest neighbors instead of single residue statistics in older versions. Version V even reaches 73.5% by introducing multiple sequence alignments to the algorithm (Kloczkowski, et al., 2002).

**Nearest neighbor methods**

These methods are based on the existence of a large data bank of known structure folds. Theoretically, similar sequence segments share similar structures. The main idea is doing alignment for the query sequence against some database (for example PDB) and calculating the structural element for each residue of the query sequence according to the matching information. This method gives ideal results if there are many similar matches to the query
sequence. There are, however, many shortcomings. First, novel sequences may not always have many ideal matches in the data bank; long sequences may not have enough matches to achieve reliable prediction, since the matches are usually short local alignment segments; the ends of the query sequence may not get enough matches, etc. Therefore, selection of a satisfactory set of matching segment must be cautiously considered (Salamov & Solovyev, 1997). Also it is based on local alignments and therefore, the accuracy of aligning algorithm may affect the prediction result dramatically. The representatives are NNSSP (Salamov & Solovyev, 1997) and PREDATOR, which use the eight elements assignment of DSSP for the prediction.

**Methods based on hidden Markov models**

The idea is quite similar to nearest neighbor method. The different approach is that it constructs the prediction model based on hidden Markov models. It has disadvantages similar to those of nearest neighbor methods. HMMSTR (Bystroff, et al., 2000) is reported to reach an accuracy level of 74.3%.

**Neural network methods and other artificial intelligence approaches**

This approach gains its name in some sense that it simulates the operations of synaptic connections among neurons. As the neurons are layered to process different levels of signals, the neural networks accept the original input (which might be a sequence segment, having been modified by a weighting factor), combine more information, integrate all to an output, send the output to the next level of processor, and so on. During the process, the network adjusts the weight and bias continuously, according to the known information, i.e. the network is trained by a training set of homologous sequences of known structure in the prediction problem. The more layers, the more information it distinguishes. The PHD (Rost & Sander, 1994) is a representative method, reporting 72% accuracy of prediction. This method depends on homologous proteins of known structure. If the query protein is not homologous to the training proteins, the results may be poor.

Last is the combination of various methods. There are usually two kinds of errors existing in prediction methods, one is systematic error, which is from the nonlocal effects, and the other is from the method itself, for example in AI approaches, the training error from the training sets. Theoretically, if the errors from different methods are independent and are not
systematic only, then the combination may help improve the prediction accuracy. But it seems that there is not a criterion for when to combine and what methods are to be included.

**Motivation of current study**

Even a few years ago, according to David Baker, it was quite unlikely to predict how proteins assume their intricate three-dimensional forms if no related proteins of known structure were available. The unknown protein structure for those proteins whose sequence resembles a protein of known structure is obtained through the templates of the known three-dimensional structures. However, at present about 40 percent of protein sequences arising from the genome sequencing projects have no homologues of known structure.

In the field of *ab initio* protein structure prediction, Rosetta algorithm, computational technique developed by Baker and his colleagues at the University of Washington, was quite successful in predicting the three-dimensional structure of a folded protein from its linear sequence of amino acids during the fourth Critical Assessment of Techniques for Protein Structure Prediction (CASP4).

Under the assumption that the distribution of structures sampled by an isolated chain segment can be approximated by the distribution of conformations of that sequence segment, Rosetta predicts local structures by searching all possible conformations. "Folding to the native structure occurs when conformations and relative orientations of the segments allow burial of the hydrophobic residues and pairing of β-strands without steric clashes" (Bonneau et al., 2001).

It is commonly believed that similarities between the sequences of two proteins infer similarities between their structures, especially when the sequence similarity is greater than 50%. A protein sequence folds into a unique three-dimensional structure. Interestingly, there is not always a one-to-one correspondence. The structure space is smaller than the sequence space. Or we can say, the structure of a fold is more representative. It is possible that different sequences may share similar structures. Even from the evolutionary point of view, the structure is more conserved than sequence.
Usually, sequences derived from the same ancestor (homologues) are more similar. Yet if the relationship is very distant (distant homologues), the sequence similarity is hard to detect. But the conservation of some special motifs can possibly be detected using special methods. Those special motifs are considered even more important than general matches found in alignments of close homologues. We can say that the local conservation is more prominent than the sequence as a whole.

Based on the fact that a similar sequence implies a similar structure, and conserved local motifs contribute to a similar structure, we try to find a local alignment method to obtain structure information which can be used to predict the query sequences. Basically, we try to apply Rosetta’s segment assembly idea of tertiary structure analysis to our secondary structure prediction, by combining the information of segments obtained through local alignment and use this information to make the prediction. BLAST provides such an alignment method. If we do BLAST on query sequences against some databases in which all the information of proteins is available (for example PDB), then it is possible to use BLAST results to predict the secondary structure from sequence \((de \ novo/ab \ initio\) prediction). BLAST can be regarded as a shortcut by which the evolutionary information can be obtained.

We get this information and assign different weights to the matches according to the similarity scores. Based on the matching information, we calculate the normalized scores of each position to be some secondary structure. We choose the highest score as the prediction. Sometimes, the scores of being E, H or C are close or even equal, so that choosing the highest is difficult or unreasonable. Sometimes, the predicted secondary structure element is not physically meaningful in real life, for example, helices having only two residues, or mixtures of strand (E) and helix (H) residues. Therefore, we try to use artificial intelligence (AI) approaches to learn from a training set for some cases of weight assigning methods. The idea is to choose the most likely score and make the prediction accordingly.

More concretely, the procedure is as follows:

Choose a dataset of sequences with known structures as query sequences. Do local alignment against some database of known structures. Retrieve the segments from the alignment result. Assign weights to the segments. Calculate normalized scores for each of the
residues of the query sequences to be some specific secondary structure elements. Make final prediction according to the normalized scores. Finally, evaluate the prediction accuracy.
CHAPTER 2. METHODS

Brief review of the methods

Methods section includes dataset selection, dataset sequence alignment against the PDB (Protein Data Bank) using BLAST (Basic Local Alignment Search Tool), prediction and evaluation.

In the prediction and evaluation part, for each query sequence from the dataset, we will assign weights to the matching segments obtained from BLAST, calculate normalized scores for each residue, predict the secondary structure for that residue according to the normalized scores, and finally, calculate the Q₃ (an accuracy measure, see definition in Chapter 3 – Q₃ definition) to evaluate.

In the weight assignment part, several parameters are considered, including substitution matrices, similarity/identity score regions, similarity/identity cutoffs, degree of exposure to solvent of residues, nonlinear function of match occurrences, subfamilies of matches, different sequence directions.

Two strategies are applied to predict the secondary structure according to normalized scores of residues. One is to choose the highest-score-corresponding element, the other is to use AI (artificial intelligence) approaches to choose based on a learning result.

As part of the evaluation, the prediction accuracies for sequences of different classes and lengths calculated and compared.

Figure 2-1 shows the procedure of prediction and evaluation, including weight assignment strategies and parameters, strategies of prediction based on normalized scores, and evaluation methods.

Dataset selection

The 513 non-redundant domains collected by Cuff and Barton (Cuff & Barton, 1999; Cuff & Barton, 2000) are selected as query sequences. (Also see Appendix D)

KS267 dataset is the set used by GOR IV (Garnier, 1996).
PDB sequences and known secondary structures are obtained from ftp://ftp.rcsb.org/pub/ released in July 2003. The secondary structure file is “ss.txt”.

Dataset selection

Alignment against PDB

Prediction

Second. struct. translation

Weight assignment

Normalized scores

Prediction based on scores

Evaluation

Reciprocal of similarity powers

Identity score powers

Substitution matrices

Similarity or identity score regions

Similarity or identity score cutoffs

Surface area of exposure to solvent

Nonlinear weight assignment scheme

Subfamilies of matches

Sequence directions

Strategies

Parameters

Choose highest score

AI learn and choose

O₁ for dataset sequences

O₁ for different classes in set

O₁ for seq. of different

Figure 2-1. Procedure of prediction and evaluation

Local sequence alignment using BLAST

Download the Blastcl3 program from BLAST ftp server.

Run blastcl3 on CB513 sequences using different parameters. Here is an example:

blastcl3 -p blastp -d pdb -e 10000 -i <query sequence input file name> -o <output file name>.

Following are some parameters:

-p indicates we are using Protein-protein BLAST (blastp).
-d defines the database which is PDB in our case.
-i is followed by the input file in FASTA format which contains query sequences.
-o is followed by the output file which contains all the alignment matching information by doing BLAST against corresponding database.
-e indicates the expect threshold, value of matches expected to be found merely by chance, according to the stochastic model of Karlin and Altschul (1990). If the statistical significance ascribed to a match is greater than the expect threshold, the match will not be reported.

CB513 prediction and evaluation of prediction

Secondary structure elements interpretation

We follow three-element notations for the secondary structures, namely, helix (H), extended (β-sheet) (E), and coil (C).

We thus need a translation from Dictionary of Secondary Structures of Proteins (DSSP) classification (Kabsch & Sander, 1983) to the three-element notation. According to DSSP classification, there are eight elements of secondary structure assignment: H (α-helix), E (extended β-strand), G (3_10 helix), I (π-helix), B (bridge, a single residue β-strand), T (β-turn), S (bend), and C (coil). For the translation, we follow the strategy of CASP (Critical Assessment of Techniques for Protein Structure Prediction) (Moult et al., 1995). Helices (H, G, and I) in the DSSP code are assigned the letter H in the three-letter secondary structure code, whereas strands (E) and bridges (B) in the DSSP code are translated into sheets (E). Other elements of the DSSP structure (T, S, C) are translated into coil (C).

Weight assignment for matches of segments

Basically, we apply two weight-assigning methods. One is using the reciprocal of similarity or identity scores and their powers \(1/\text{sim}_d^d\) or \(1/\text{id}_c^c\); the other is using similarity or identity scores and their powers \(\text{sim}_c^c\) or \(\text{id}_c^c\) directly (\(d\) and \(c\) are positive real numbers).
The *id* mentioned above is the identities in fraction, representing the ratio of number of exact matches of residues and the total number of residues of the matching segment. The *sim* is the positives in fraction, representing the ratio of the number of matches of residues, including both exact matches and similar matches, and the total number of residues of the matching segment.

Weights are adjusted when different parameters are considered. This will be illustrated in the parameter section.

**Calculation of normalized scores for each residue**

The prediction is position-based (residue by residue on the query sequences). At each position, the predicted secondary structure is determined by the actual secondary structures of the matches at that position. Each match is assigned a weight according to the similarity or identity score of the alignment of BLAST. At each position, the weights are normalized, and the normalized scores for the position being some specific secondary structure element are calculated.

The procedure of normalized score calculation is illustrated by showing the prediction of a fictitious example (see Figure 2-2).

<table>
<thead>
<tr>
<th>Query sequence</th>
<th>Matches</th>
<th>Weight (w)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0.1</td>
<td>E</td>
<td>E</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.2</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td>E</td>
<td>C</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.3</td>
<td>C</td>
<td>C</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.4</td>
<td>C</td>
<td>H</td>
<td>E</td>
<td>H</td>
<td>H</td>
<td>E</td>
<td>C</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2-2. A fictitious example showing the query sequence and matching segments. The query sequence residues are represented with their sequence position numbers. The matching segments are expressed as secondary structure elements. Suppose we have got the weights for the segments.

Define \( s(H, i) \) to be the normalized score of position \( i \) to be \( H \). There are two ways for score calculation:

1) Consider residue by residue
\[ s(H, i) = \frac{\sum w(H, i)}{\sum w(H, i) + \sum w(E, i) + \sum w(C, i)} \]

\( w(H, i) \) is the weight of the matching segment whose residue at \( i \)th position belongs to a helix. \( w(E, i) \) and \( w(C, i) \) are similarly defined.

In above example:

\[ s(H, 2) = 0.2/(0.1 + 0.2 + 0.4) \]
\[ s(H, 4) = 0.1/(0.1 + 0.2 + 0.3 + 0.4) \]
\[ s(E, 4) = (0.2 + 0.4)/(0.1 + 0.2 + 0.3 + 0.4) \]

2) Consider "doublets" instead, both the \( i \)th and \((i+1)\)th positions are considered:

\[ s(H,i) = \frac{\sum w(HH, i)}{\sum w(HH, i) + \sum w(EH, i) + \sum w(CH, i)} + \frac{\sum w(HE, i)}{\sum w(HE, i) + \sum w(EE, i) + \sum w(CE, i)} + \frac{\sum w(HC, i)}{\sum w(HC, i) + \sum w(EC, i) + \sum w(CC, i)} \]

\( w(HH, i) \) stands for the weight of the matching segment whose \( i \)th residue belongs to a helix and the \((i+1)\)th residue also belongs to a helix.

Then normalize it. \( s(E, i) \) and \( s(C, i) \) are similarly defined.

In our application, we did not show results from definition 2), since with this method, the prediction accuracy was much lower than that from the first method.

There could be other definitions of normalized scores, which may lead to better results.

In addition, we give a prediction example (prediction of 1C75_A) in appendix F.

**Parameters adjusted for weight assignments**

**Different substitution matrices**

PAM (Percent Accepted Mutation) matrix was introduced by Dayhoff (Dayhoff *et al.*, 1978) to quantify the amount of evolutionary change in a protein sequence, i.e. how often different amino acids replace other amino acids in evolution. It is a way to assign weights to
measure the amino acid differences. This was based on a database of 1572 changes in 71
groups of closely related proteins.

BLOSUM (Blocks Substitution Matrix) was introduced by Henikoff (Henikoff &
Henikoff, 1992) to get a better measure of differences between two proteins specifically for
more distantly related proteins. It is a substitution matrix in which scores for each position
are derived from observations of the frequencies of substitutions in blocks of local
alignments in related proteins (according to National Center for Biotechnology Information,
NCBI). The blocks were constructed by PROTOMAT from 504 non-redundant groups of
proteins catalogued in Prosite 8.0 (Bairoch, 1991) keyed to Swiss-Prot 20 (Bairoch and

More detailed calculations and the procedures of matrix construction can be found in
appendix B.

We used BLOSUM45, 62, 80 and PAM30, 70 in BLAST alignments in our experiments.
The numbers are percentage values of sequence identity for clustering (see Appendix B).

**Similarity or identity score regions** (Rost, 1997)

Based on the basic weight assignment methods, either the inverse of similarity scores and
their powers, or the identity scores and their powers, the weights are modified according to
the zones in which the identity or similarity scores are located. Usually, when the
identity/similarity score of a match is greater than some cutoff, say 50%, its weight should be
larger. (Koehl & Levitt, 2002) If the identity/similarity score is within 20~30%, the weight
should be lower.

**Similarity or identity score cutoffs**

To avoid overestimating prediction accuracies, different cutoffs of similarity or identity
scores of matches are set. Cutoffs include 99%, 90%, 80%, 70% and 60%. The matches with
similarity or identity scores higher than a cutoff are eliminated from matching lists of
segments, which are used to calculate normalized scores of residues. Results are also
observed when some reasonable high-score matches (similarity or identity scores higher than
cutoff) are kept for calculation.

**Residue surface area (degree of exposure to solvent)**
The degrees of exposure to solvent of the residues of all the sequences in PDB are calculated. Following are the Procedures of calculation.

1) Download all the PDB files. A perl script is downloaded and applied. Its input is a file containing all the protein names in PDB.

2) Calculate the solvent accessibility of residues for each protein. Download Naccess (Hubbard & Thornton, 1992) that calculates the accessible area of a molecule from a PDB (Protein Data Bank) format file. Write a script to do the calculation in batch.

3) Construct a residue status file `resStatus.txt` that gives the solvent accessibility status of all residues of all proteins in PDB. Develop a program which reads all the `.rsa` files (produced by Naccess, see an example in appendix E for its format). We retrieve the All-atoms-REL column, which gives the relative solvent exposure area of residues. Then construct this status file according to NUM column, which gives the residue index of the amino acid sequence. Because of the variability of NUM, this time we chose the files in which the NUM starts from 1. So the `resStatus.txt` is not complete.

4) Interpret `.rsa` files. In our application, we define the accessibility of residues to be buried if the REL value in a `.rsa` file is less than or equal to 5.0, exposed if the number is greater or equal to 40.0, and intermediate if in-between.

5) Add the status information to the prediction. Develop a prediction program that modifies the previous weight calculation method. Originally, the best result is obtained from the prediction if the match weight is defined as $1/\sqrt{\text{sim}}$. In this program, instead of assigning the same weight for the same match, each residue in one match is considered separately, according to the buried and exposed information of that residue. Buried residues gain more weight.

**Nonlinear weighting scheme for match occurrences**

We used a sigmoid function (see Figure 2-3) $f(x) = \frac{1}{1 + e^{-x}}$ to define the weight of a match at some particular sequence position of the query sequence. Basically, we want to define a non-linear function to reflect the influence of the match occurrences on the weights. $x$ corresponds to the number of matching residues and is adjusted in several ways:
a) $x$ is the occurrence of some secondary structure element at position $i$ of query sequence. b) $x$ is the occurrence of some secondary structure element at position $i$ of query sequence, but $x$ is multiplied by 100 to make the graph sharper ($f(x) = \frac{1}{1 + e^{-100x}}$).

c) $x$ is the occurrence of some secondary structure element at position $i$ minus the average occurrence of three elements (H, E and C) at that position, trying to move the number of occurrences to the steep part of the graph. d) $x$ is the occurrence of some secondary structure element at position $i$ minus the average occurrence of the three elements at that position. In the sigmoid function, $x$ is multiplied by a coefficient 100 to make the graph sharper.

![Sigmoid function](image)

Subfamilies of the matches

Trial 1. After running BLAST against PDB, we collected all the matches and did complete alignment using Clustalx (1.83). Based on the alignment results, we tried to create a phylogenetic tree using Mega2. However, the matches are too divergent to create such a tree. We finally took the BLAST result as the alignment result, and tried to create trees corresponding to some specific-position-centered residues. To increase the similarity of the matches, we tried to define windows of different sizes, to keep the sizes of overlaps of matches centered at that residue to be greater than the window sizes. However, that did not help in constructing the phylogeny trees.

Trial 2. Use SCOP classification database. See SCOP resource part for classification of proteins. After the matches have been classified into several subfamilies, the weights of matches are modified accordingly. The weight assigning is position-based, i.e. residue by
residue. The intuitive way is to "divide" the total weight of the matches into \( m \) parts, where \( m \) is the number of subfamilies. The weight of a subfamily is divided evenly into \( n \) parts, where \( n \) is the number of matches in this subfamily. The weight of some specific match in an \( n \)-numbered subfamily thus has \( 1/n \) of its original weight.

**Prediction in different sequence directions and their combination** (Park *et al.*, 2000)

Develop programs to produce the sequence in reverse direction and secondary structure files of CB513.

Do the prediction as normal, and get the accuracy.

Combine the normal and reverse order prediction. In the program, first do the predictions separately. While the normalized score matrices are formed, compare the matrix entries. If the corresponding entries are inconsistent, choose the higher-normalized-score-corresponding secondary structure element to be the predicted secondary structure at that position. Then report the final accuracy.

**Prediction based on normalized scores of residues**

**Choosing the highest score**

The secondary structure element having the highest score is chosen as the final predicted result for the residue. For a specific position of some query sequence, we have three normalized scores for the residue to be some secondary structure element \((s(H, i), s(E, i), \text { and } s(C, i))\). We always choose the highest score among these three to determine the prediction for that residue.

**Artificial intelligence (AI) approaches**

We used AI techniques to modify the final secondary structure decision step. Instead of assigning the secondary structure for some specific position according to the highest normalized score of some secondary structure at that position, we applied artificial intelligence approaches to choose the most appropriate normalized score according to the learning result from some training sets. We used naïve Bayes (NB), decision trees (DT), neural networks (NN) and support vector machines (SVM) and performed comparisons among them. We tried to learn from the amino acid sequences to predict secondary structures.
The main idea of these AI approaches is to gather information from the training set and use it to predict the test set. The ratio of the numbers of training and test sequences is 4:1.

**Amino acid sequence training.** We randomly partitioned the files into training and test sequence files and training and test secondary structure files. We defined different window sizes from 3 to 13, scanned the sequence training file of some window sizes, moved forward one residue at a time, listed the secondary structure letter of the middle residue of the window as the target.

**Normalized-score-based training.** We formed a file that contains all the normalized scores for all the query sequences from the benchmark dataset, randomly partitioned these scores into training and test sets, and applied AI approaches to make the prediction.

The principles of these approaches and running the programs can be found in Appendix C.

**Working on the KS267 dataset**

The KS267 dataset was tested by the GOR IV method (Garnier, Gibrat & Robson, 1996). We also predicted and evaluated this dataset for comparison, using the reciprocal of similarity scores and their powers \((1/sim^d)\) as the weight assignment method for the local alignment matches. Here no optimization steps are taken, i.e. only the basic methods are applied.

**Some cases (prediction application)**

On the basis of the SCOP classification schemes, several proteins from all-\(\alpha\), all-\(\beta\), \(\alpha+\beta\), and \(\alpha/\beta\) classes were freely chosen to test the prediction accuracy and to illustrate how the predictions behave. In this case, two methods of weight assigning were used (see Appendix A).
PDB and SCOP

PDB resource

The PDB is updated frequently. For comparison purposes, we used the PDB release file obtained in July 2003. Two files were obtained from PDB derived data ftp site (ftp://ftp.rcsb.org/pub/pdb/derived_data/), “ss.txt” and “pdb_seqres.txt” in FASTA format. “ss.txt” contains all the protein names and chains following the secondary structures for the proteins, “pdb_seqres.txt” contains all the protein names and chains following the amino acid sequences. Both of them are in same order.

SCOP resource

While using the SCOP classification, the parseable files were downloaded. The classification was made based on the PDB code and the chain name. Take the file dir.cla.scop.txt 1.63 as an example, the sunid—SCOP Unique Identifier follows the “cl” entry for each protein record. More concretely, the following are the correspondence relationships among the number codes and classification codes: 46456 – all-α, 48724 – all-β, 51349 - α/β, 53931 - α+β, 56572 - Multi-domain proteins (alpha and beta), 56835 - Membrane and cell surface proteins and peptides, 56992 - Small proteins, 57942 - Coiled coil proteins, 58117 - Low resolution protein structures, 58231 - Peptides and 58788 - Designed proteins.
CHAPTER 3. RESULTS

$Q_3$ definition

The main parameter used to measure prediction accuracy is $Q_3$ (Kloczkowski et al., 2002), the percentage of all correctly predicted residues within the three state (H, E, C) classes. An accuracy matrix $[A_{ij}]$ of the size $3 \times 3$ ($i$ and $j$ stand for the three states H, E, C) was introduced. The $ij$-th element $A_{ij}$ of the accuracy matrix is the number of residues predicted to be in state $j$, which according to the PDB data is actually in state $i$. Obviously, the diagonal entries of $[A_{ij}]$ represent the number of correctly predicted residues. $Q_3$ is therefore defined as:

$$Q_3 = \frac{\sum_{i=1}^{3} A_{ii}}{N} \times 100$$

where $N$ is the total number of residues in the query sequence, and defined as the total number of all the entries of $[A_{ij}]$:

$$N = \sum_{i=1}^{3} \sum_{j=1}^{3} A_{ij}$$

Flow chart of prediction and results

Figure 3-1 gives the main prediction procedure and results. There are two main methods of weight assignment. Starting from the basic methods, different parameters are used to adjust the weights of matching segments.

Weight assigning method one

We define the weight for each of the BLAST matches to be the reciprocal of similarity or identity scores and their powers ($1/sim^d$ or $1/id^d$)
Figure 3-1. Main prediction procedure with best Q₃ results for parameters investigation. P30 represents the substitution matrix PAM30, B45 represents BLOSUM 45, B62 BLOSUM62. id and sim represent identity score and similarity score, respectively obtained from local alignments. The exponent d is a real positive number. AI stands for Artificial Intelligence. DT, NN, and SVM stand for Decision Trees, Neural Networks, and Support Vector Machines, respectively. This figure shows the prediction accuracies for the two main different weight assigning methods, after different parameters are applied to adjust the weights of matches.
Basic method

The weights of matches are defined to be the reciprocal of the similarity score or their power of a match. No optimization methods are applied. The prediction results are shown in Figure 3-2, for different substitution matrices are used while doing alignment, and for different powers of similarity/identity scores used to define the match weight. The best prediction accuracy obtained is 0.750 when the weight match function is \(1/\sqrt{\text{sim}}\) and the substitution matrix is PAM30.

![Figure 3-2](image)

Figure 3-2. Accuracies of basic prediction using weight assigning method one (reciprocal of similarity score powers) under different substitution matrices. Alignment with PAM30 yields consistently the best result.

Different similarity cutoffs

In this set of experiments, some high-similarity matches were filtered out using a number of identity score cutoffs, but some of the reasonable high-similarity matches were still kept.

Similarity cutoff 99%

The matches with similarity scores greater than 99% are eliminated from the matches used for prediction. Based on the basic results (Figure 3-2), we only consider the cases of using BLOSUM45, BLOSUM62, and PAM30 matrices while doing the alignment. Again,
the best result (0.706) comes from the prediction when the match weight is defined as $1/\sqrt[3]{\text{sim}}$ and PAM30 is used to do the alignment, shown in Table 3-1.

Table 3-1. Prediction accuracies (Q3) of weight assigning method one with filter cutoff 99% similarity

<table>
<thead>
<tr>
<th></th>
<th>$1/id$</th>
<th>$1/sim$</th>
<th>$1/sim^2$</th>
<th>$1/\sqrt{\text{sim}}$</th>
<th>$1/\sqrt[3]{\text{sim}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>0.634</td>
<td>0.646</td>
<td>0.618</td>
<td>0.661</td>
<td>0.665</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>0.660</td>
<td>0.670</td>
<td>0.651</td>
<td>0.679</td>
<td>0.682</td>
</tr>
<tr>
<td>PAM30</td>
<td>0.694</td>
<td>0.699</td>
<td>0.675</td>
<td>0.705</td>
<td>0.706</td>
</tr>
</tbody>
</table>

**Similarity cutoff 90%**

In this set of experiments, the matches with similarity scores greater than 90% are eliminated from the matches used for prediction. During this process, however, many good matches are kept. The good ones that are neither too short (less than 5 residues) nor too long (as long as close to the length of query sequence, longer than 95% of query sequence length) are kept. The best result is 0.668, shown in Table 3-2.

Table 3-2. Prediction accuracies (Q3) of weight assigning method one with filter cutoff 90% similarity

<table>
<thead>
<tr>
<th></th>
<th>$1/id$</th>
<th>$1/sim$</th>
<th>$1/sim^2$</th>
<th>$1/\sqrt{\text{sim}}$</th>
<th>$1/\sqrt[3]{\text{sim}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM62</td>
<td>0.652</td>
<td>0.658</td>
<td>0.645</td>
<td>0.663</td>
<td>0.664</td>
</tr>
<tr>
<td>PAM30</td>
<td>0.663</td>
<td>0.665</td>
<td>0.651</td>
<td><strong>0.668</strong></td>
<td><strong>0.668</strong></td>
</tr>
</tbody>
</table>

**Different similarity level zones/regions**

In these experiments, matches with similarity scores in different similarity level zones (Rost, 1997) are assigned different weights.

**Two zones considered**

The similarity zones considered are [0, 0.5) and [0.5, 1.0]. When the similarity is greater than 50%, multiply the weight of that match by 0.5, otherwise we multiply the weight of that match by 10. PAM30 is of particular interest, because it gave the best results in a previous case. We also consider the zones [0, 0.35) and [0.35, 1.0], with different adjustments of
weight assigning. The best result obtained is 0.756. Recall that the best result obtained in the basic method is 0.75. Detailed results are shown in Table 3-3.

Table 3-3. Prediction accuracies (Q3) of weight assigning method one with two zones, w stands for weight.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Adjust Weights</th>
<th>1/ id</th>
<th>1/ sim</th>
<th>1/ sim²</th>
<th>1/ √sim</th>
<th>1/ √√sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td></td>
<td>0.640</td>
<td>0.725</td>
<td>0.685</td>
<td>0.740</td>
<td>0.736</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>w*0.5 if sim ≥ 0.5</td>
<td>0.665</td>
<td>0.704</td>
<td>0.675</td>
<td>0.719</td>
<td>0.720</td>
</tr>
<tr>
<td>BLOSUM80</td>
<td>w*10 if sim &lt; 0.5</td>
<td>0.696</td>
<td>0.716</td>
<td>0.692</td>
<td>0.730</td>
<td>0.735</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.726</td>
<td>0.747</td>
<td>0.723</td>
<td>0.754</td>
<td><strong>0.756</strong></td>
</tr>
<tr>
<td>PAM70</td>
<td></td>
<td>0.708</td>
<td>0.726</td>
<td>0.701</td>
<td>0.739</td>
<td>0.743</td>
</tr>
<tr>
<td>PAM30</td>
<td>w*0.5 if sim ≥ 0.5</td>
<td>0.726</td>
<td>0.748</td>
<td>0.724</td>
<td>0.755</td>
<td><strong>0.756</strong></td>
</tr>
<tr>
<td></td>
<td>w*5 if sim &lt; 0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>w*0.5 if sim ≥ 0.5</td>
<td>0.726</td>
<td>0.748</td>
<td>0.725</td>
<td>0.755</td>
<td>0.755</td>
</tr>
<tr>
<td></td>
<td>w*2 if sim &lt; 0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>w*0.5 if sim ≥ 0.35</td>
<td>0.726</td>
<td>0.740</td>
<td>0.707</td>
<td>0.750</td>
<td>0.751</td>
</tr>
<tr>
<td></td>
<td>w*5 if sim &lt; 0.35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>w*0.5 if sim ≥ 0.35</td>
<td>0.726</td>
<td>0.739</td>
<td>0.707</td>
<td>0.750</td>
<td>0.751</td>
</tr>
<tr>
<td></td>
<td>w*2 if sim &lt; 0.35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Four zones considered

Similarity zones are separated by 20% and 30%. Namely, the zones are [0, 0.2), [0.2, 0.3], (0.3, 0.5), and [0.5, 1.0]. When the similarity is greater than 0.5, multiply the weight of that match by 0.5, when the similarity is between 0.2 and 0.3, multiply the weight of the match by 5, and for others keep the weight unchanged. This time the result is compared with a control for similarity cutoff 0.9. The best result obtained is 0.670 (Table 3-4) when the match weight is defined as 1/√√Sim and BLOSUM62 is used to do the alignment.

Table 3-4. Prediction accuracies (Q3) of weight assigning method one with four zones

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Adjust Weights</th>
<th>Similarity Cutoff</th>
<th>1/ id</th>
<th>1/ sim</th>
<th>1/ sim²</th>
<th>1/ √sim</th>
<th>1/ √√sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>w*0.5 if sim ≥ 0.5</td>
<td>0.9</td>
<td>0.625</td>
<td>0.657</td>
<td>0.653</td>
<td>0.656</td>
<td>0.656</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>w*5 if</td>
<td></td>
<td>0.652</td>
<td>0.669</td>
<td>0.665</td>
<td>0.670</td>
<td><strong>0.670</strong></td>
</tr>
<tr>
<td>PAM30</td>
<td>0.2 ≤ sim ≤ 0.3</td>
<td></td>
<td>0.663</td>
<td>0.669</td>
<td>0.663</td>
<td>0.669</td>
<td>0.669</td>
</tr>
</tbody>
</table>
Surface area of exposure to solvent of residues

The accessibility thresholds are defined to be 5.0 and 40.0, i.e. if the relative accessibility of a residue is less than 5.0, it is regarded as a buried residue; if the relative accessibility of a residue is greater than 40.0, it is regarded as an exposed residue; the rest are regarded as intermediate residues (See methods part for details). Here, we only see the results when the weights of matches are defined as $1/\sqrt{\text{sim}}$. Accordingly, we make simple linear changes of the weights of residues. If the residue is buried, its weight is multiplied by an integer (we tested values of 2, 3, and 20); if it is intermediate, we multiply the original weight by a relatively small number (1.5, 2, and 3, respectively); if exposed, the weight is unchanged. No other optimization methods have been applied. The best accuracy is 0.757, obtained when PAM30 is used for the alignment. (Table 3-5)

Table 3-5. Prediction accuracies ($Q_3$) of weight assigning method one with accessibility of residues considered. $w$ stands for weight.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Adjust Weights</th>
<th>$1/\sqrt{\text{sim}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>$w*2$ if buried</td>
<td>0.691</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>$w*1.5$ if intermediate</td>
<td>0.700</td>
</tr>
<tr>
<td>BLOSUM80</td>
<td></td>
<td>0.730</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td><strong>0.757</strong></td>
</tr>
<tr>
<td>PAM70</td>
<td></td>
<td>0.741</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>$w*3$ if buried</td>
<td>0.687</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>$w*2$ if intermediate</td>
<td>0.698</td>
</tr>
<tr>
<td>BLOSUM80</td>
<td></td>
<td>0.724</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.747</td>
</tr>
<tr>
<td>PAM70</td>
<td></td>
<td>0.733</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>$w*20$ if buried</td>
<td>0.597</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>$w*3$ if intermediate</td>
<td>0.613</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.695</td>
</tr>
</tbody>
</table>

The occurrences of matches

In these experiments, the weights of matches are adjusted according to the number of occurrences of matches. The premise is that, if a residue has higher frequency of matches for a particular secondary structure element, it ought be more reliable for predicting that element to be the secondary structure, and thus the weights of those matches should be greater than those of the matches with lower frequencies. Table 3-6 shows partial results of prediction using the sigmoid function as the weight adjustment function. $x$ is the exponent in the
sigmoid function mentioned previously in the method discussion of the parameters for weight adjustment. We only show the results for the predictions in which BLOSUM45 and PAM30 are used while doing local alignments. Unfortunately, no obvious differences are observed in comparison with the original basic method for any of these adjustments. The best accuracy obtained is 0.750, which is the same as that of basic method.

Table 3-6. Prediction accuracies (Q₃) of weight assigning method one with match occurrences considered

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Adjust Weights</th>
<th>1/id</th>
<th>1/sim</th>
<th>1/sim²</th>
<th>1/√sim</th>
<th>1/³sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>Basic (control)</td>
<td>0.640</td>
<td>0.658</td>
<td>0.658</td>
<td>0.677</td>
<td>0.682</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.726</td>
<td>0.738</td>
<td>0.738</td>
<td>0.749</td>
<td>0.750</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>x is occurrence of some ss element at position i</td>
<td>0.640</td>
<td>0.658</td>
<td>0.658</td>
<td>0.677</td>
<td>0.682</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.731</td>
<td>0.742</td>
<td>0.742</td>
<td>0.749</td>
<td>0.750</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>x is occurrence of some ss element at position i, exponent is now 100x in the sigmoid function</td>
<td>0.640</td>
<td>0.658</td>
<td>0.658</td>
<td>0.677</td>
<td>0.682</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.726</td>
<td>0.738</td>
<td>0.738</td>
<td>0.749</td>
<td>0.750</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>x is the occurrence of some ss element minus the average occurrence at that position</td>
<td>0.670</td>
<td>0.677</td>
<td>0.677</td>
<td>0.683</td>
<td>0.684</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.741</td>
<td>0.749</td>
<td>0.749</td>
<td>0.749</td>
<td>0.750</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>x is the occurrence of some ss element minus the average occurrence at that position, exponent is 100x now</td>
<td>0.655</td>
<td>0.664</td>
<td>0.664</td>
<td>0.678</td>
<td>0.682</td>
</tr>
<tr>
<td>PAM30</td>
<td></td>
<td>0.735</td>
<td>0.743</td>
<td>0.743</td>
<td>0.749</td>
<td>0.750</td>
</tr>
</tbody>
</table>

Subfamilies of the matches

In this set of experiments we used SCOP as a tool to classify the matches. The results are obtained from the prediction using the BLOSUM45 and PAM30 matrices. This is only one option of doing classification. The results shown in Table 3-7 indicates that this prediction method yields much worse accuracy.

Table 3-7. Prediction accuracies (Q₃) of weight assigning method one when subfamilies of matches are considered (SCOP classification)

<table>
<thead>
<tr>
<th></th>
<th>1/id</th>
<th>1/sim</th>
<th>1/sim²</th>
<th>1/√sim</th>
<th>1/³sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>No classification (B45)</td>
<td>0.640</td>
<td>0.658</td>
<td>0.658</td>
<td>0.677</td>
<td>0.682</td>
</tr>
<tr>
<td>5 subfamilies</td>
<td>0.604</td>
<td>0.619</td>
<td>0.590</td>
<td>0.639</td>
<td>0.646</td>
</tr>
<tr>
<td>12 subfamilies</td>
<td>0.549</td>
<td>0.561</td>
<td>0.539</td>
<td>0.575</td>
<td>0.581</td>
</tr>
<tr>
<td>No classification (P30)</td>
<td>0.726</td>
<td>0.738</td>
<td>0.738</td>
<td>0.749</td>
<td>0.750</td>
</tr>
<tr>
<td>5 subfamilies</td>
<td>0.693</td>
<td>0.703</td>
<td>0.675</td>
<td>0.717</td>
<td>0.721</td>
</tr>
</tbody>
</table>
Using SCOP classification to see the prediction for each subclass

The following results are based on the results obtained above (the weights of matches are adjusted in accordance with numbers of occurrences of matches.)

Classification of the CB513 dataset according to SCOP classification.

Five subclasses for the CB513 dataset are obtained. There are 99 all-α, 123 all-β, 145 α/β, and 64 α+β protein sequences in this dataset respectively. The remaining sequences go to the “others” subclass. The sequences in the “others” subclass cannot be classified according to current SCOP parcellable files.

Analysis of the prediction accuracies for each subclass

Table 3-8. Prediction accuracies (Q₃) of weight assigning method one for different kinds of proteins (SCOP classification)

<table>
<thead>
<tr>
<th>Matrix</th>
<th>Protein Classification</th>
<th>1/id</th>
<th>1/sim</th>
<th>1/sim²</th>
<th>1/sim</th>
<th>1/3sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>α</td>
<td>0.685</td>
<td>0.700</td>
<td>0.700</td>
<td>0.714</td>
<td>0.718</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>0.610</td>
<td>0.629</td>
<td>0.629</td>
<td>0.649</td>
<td>0.655</td>
</tr>
<tr>
<td></td>
<td>α/β</td>
<td>0.630</td>
<td>0.648</td>
<td>0.648</td>
<td>0.668</td>
<td>0.673</td>
</tr>
<tr>
<td></td>
<td>α+β</td>
<td>0.623</td>
<td>0.640</td>
<td>0.640</td>
<td>0.659</td>
<td>0.664</td>
</tr>
<tr>
<td>PAM30</td>
<td>α</td>
<td>0.731</td>
<td>0.744</td>
<td>0.744</td>
<td>0.751</td>
<td>0.751</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>0.715</td>
<td>0.727</td>
<td>0.727</td>
<td>0.735</td>
<td>0.736</td>
</tr>
<tr>
<td></td>
<td>α/β</td>
<td>0.731</td>
<td>0.740</td>
<td>0.740</td>
<td>0.748</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td>α+β</td>
<td>0.735</td>
<td>0.749</td>
<td>0.749</td>
<td>0.758</td>
<td>0.758</td>
</tr>
<tr>
<td>BLOSUM45</td>
<td>α</td>
<td>0.706</td>
<td>0.713</td>
<td>0.713</td>
<td>0.718</td>
<td>0.718</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>0.641</td>
<td>0.650</td>
<td>0.650</td>
<td>0.656</td>
<td>0.657</td>
</tr>
<tr>
<td></td>
<td>α/β</td>
<td>0.659</td>
<td>0.667</td>
<td>0.667</td>
<td>0.674</td>
<td>0.675</td>
</tr>
<tr>
<td></td>
<td>α+β</td>
<td>0.651</td>
<td>0.660</td>
<td>0.660</td>
<td>0.666</td>
<td>0.667</td>
</tr>
<tr>
<td>PAM30</td>
<td>α</td>
<td>0.743</td>
<td>0.750</td>
<td>0.750</td>
<td>0.752</td>
<td>0.751</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>0.726</td>
<td>0.735</td>
<td>0.735</td>
<td>0.736</td>
<td>0.736</td>
</tr>
<tr>
<td></td>
<td>α/β</td>
<td>0.739</td>
<td>0.746</td>
<td>0.746</td>
<td>0.749</td>
<td>0.749</td>
</tr>
<tr>
<td></td>
<td>α+β</td>
<td>0.749</td>
<td>0.756</td>
<td>0.756</td>
<td>0.758</td>
<td>0.758</td>
</tr>
</tbody>
</table>

Here we can evaluate the accuracy differences among different kinds of proteins. Table 3-8 shows the detailed results. We can make several observations. First, the average prediction accuracy for the all-α group is the highest. Second, the difference of prediction
accuracies between the all-α and all-β group is greater than 5% approximately, while using BLOSUM 45 as substitution matrix; but less than 2% if PAM 30 is applied while doing alignment. The results for α/β, α+β groups are comparable, and are between the results of the all-α and all-β groups. The α+β group has the highest prediction accuracy while PAM 30 is used, which is 75.8% as measured in Q₃.

**Weight assigning method two**

Define the weight for each of the BLAST matches to be the identity (few cases of similarity) scores and their powers ($id^p$).

**Basic method**

We measure the performance of two basic weight-assigning methods. For the first one, the weight is defined to be the powers of identity score of a match; for the second, it is the powers of the similarity scores of a match.

The weights of matches are defined to be the powers of identity scores of matches

We tried different combinations of matrices and identity powers. The best result comes from using BLOSUM 45 and $id^3$ as weight assigning method. Figure 3-3 shows the basic average prediction accuracies for weight assigning method two using different substitution matrices.

The weights of matches are defined to be the powers of similarity scores of matches

The best result is when we use the PAM30 matrix, and the weight of match is $sim^2$ (see Table 3-9). Comparing with the above weight assigning method (identity score powers), this one is much worse.

**Different identity score cutoffs**

All “good” matches are filtered out

In the following experiments, we use the basic prediction method, and observe the influence of matches at various identity levels. We define several identity cutoffs. If the
identity score of a match is greater than the cutoff, the match is eliminated from consideration. According to the basic result, we focus mainly on predictions using BLOSUM45 and PAM30 matrices. Table 3-10 shows the details. The combination of BLOSUM45 and ID gives better results at different identity cutoff levels. Note that when we use BLOSUM45 and the weight is assigned to be \( id^3 \) at cutoff levels 0.99 and 0.90, the accuracies are 0.825 and 0.735 respectively.

Our best results for different cutoffs come from the prediction using BLOSUM45 and \( id^3 \) as weight assigning method. Figure 3-4 summarizes the tendency of changes with the drops of cutoffs.

![Figure 3-3. Basic CB513 prediction accuracy using id powers (weight assignment method two)](image)

Table 3-9. Basic prediction accuracies \( (Q_3) \) of weight assigning method two (powers of similarity scores) with \( id \) weight shown for comparison.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>( id )</th>
<th>( id^2 )</th>
<th>( id^3 )</th>
<th>( \frac{1}{sim^2} )</th>
<th>( sim )</th>
<th>( \frac{1}{sim} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>0.810</td>
<td>0.729</td>
<td>0.736</td>
<td>0.769</td>
<td>0.769</td>
<td></td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>0.779</td>
<td>0.722</td>
<td>0.727</td>
<td>0.747</td>
<td>0.747</td>
<td></td>
</tr>
<tr>
<td>BLOSUM80</td>
<td>0.794</td>
<td>0.753</td>
<td>0.755</td>
<td>0.769</td>
<td>0.769</td>
<td></td>
</tr>
<tr>
<td>PAM30</td>
<td>0.801</td>
<td>0.781</td>
<td>0.781</td>
<td>0.785</td>
<td>0.785</td>
<td></td>
</tr>
<tr>
<td>PAM70</td>
<td>0.803</td>
<td>0.767</td>
<td>0.769</td>
<td>0.780</td>
<td>0.780</td>
<td></td>
</tr>
</tbody>
</table>
Table 3-10. Prediction accuracies ($Q_3$) of weight assigning method two with different identity levels of cutoffs.

<table>
<thead>
<tr>
<th>Matrix</th>
<th>$id$ Cutoffs</th>
<th>$id^3$</th>
<th>$id^2$</th>
<th>$id$</th>
<th>$id^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>1.0</td>
<td>0.738</td>
<td>0.754</td>
<td>0.81</td>
<td>0.902</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>0.708</td>
<td>0.719</td>
<td>0.755</td>
<td>0.809</td>
</tr>
<tr>
<td></td>
<td>0.90</td>
<td>0.675</td>
<td>0.680</td>
<td>0.697</td>
<td>0.725</td>
</tr>
<tr>
<td></td>
<td>0.80</td>
<td>0.663</td>
<td>0.667</td>
<td>0.681</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td>0.70</td>
<td>0.654</td>
<td>0.657</td>
<td>0.667</td>
<td>0.683</td>
</tr>
<tr>
<td></td>
<td>0.60</td>
<td>0.646</td>
<td>0.649</td>
<td>0.657</td>
<td>0.668</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
<td>0.631</td>
<td>0.632</td>
<td>0.638</td>
<td>0.643</td>
</tr>
<tr>
<td>BLOSUM62</td>
<td>1.0</td>
<td>0.729</td>
<td>0.739</td>
<td>0.779</td>
<td>0.860</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>0.710</td>
<td>0.717</td>
<td>0.741</td>
<td>0.787</td>
</tr>
<tr>
<td></td>
<td>0.90</td>
<td>0.682</td>
<td>0.684</td>
<td>0.694</td>
<td>0.710</td>
</tr>
<tr>
<td></td>
<td>0.80</td>
<td>0.673</td>
<td>0.675</td>
<td>0.681</td>
<td>0.691</td>
</tr>
<tr>
<td></td>
<td>0.70</td>
<td>0.665</td>
<td>0.667</td>
<td>0.672</td>
<td>0.678</td>
</tr>
<tr>
<td></td>
<td>0.60</td>
<td>0.660</td>
<td>0.662</td>
<td>0.665</td>
<td>0.669</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
<td>0.650</td>
<td>0.651</td>
<td>0.651</td>
<td>0.652</td>
</tr>
<tr>
<td>BLOSUM80</td>
<td>1.0</td>
<td>0.758</td>
<td>0.765</td>
<td>0.794</td>
<td>0.856</td>
</tr>
<tr>
<td>PAM30</td>
<td>1.0</td>
<td>0.789</td>
<td>0.79</td>
<td>0.801</td>
<td>0.825</td>
</tr>
<tr>
<td></td>
<td>0.99</td>
<td>0.735</td>
<td>0.735</td>
<td>0.739</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td>0.90</td>
<td>0.678</td>
<td>0.678</td>
<td>0.677</td>
<td>0.669</td>
</tr>
<tr>
<td></td>
<td>0.80</td>
<td>0.660</td>
<td>0.660</td>
<td>0.658</td>
<td>0.650</td>
</tr>
<tr>
<td></td>
<td>0.70</td>
<td>0.646</td>
<td>0.646</td>
<td>0.645</td>
<td>0.639</td>
</tr>
<tr>
<td></td>
<td>0.60</td>
<td>0.627</td>
<td>0.627</td>
<td>0.627</td>
<td>0.625</td>
</tr>
<tr>
<td>PAM70</td>
<td>1.0</td>
<td>0.773</td>
<td>0.779</td>
<td>0.803</td>
<td>0.846</td>
</tr>
</tbody>
</table>

Figure 3-4. Prediction accuracies of weight assigning method two under different identity cutoffs for weights of $id^3$ and BLOSUM45 substitution matrix.
Perfect matches filtered out, but reasonable matches kept

Matches with the highest identity scores (greater than \( id \) cutoff) are filtered out, but the "reasonable" high-\( id \) matches are kept. We define the "reasonable" high-\( id \) matches to be those matches that have relatively high identity scores (greater than \( id \) cutoff), are not too short (>5 residues), and are not as long as the query sequence (less than 90\% or 95\% the length of the query sequence).

The prediction accuracy \( Q_3 \)'s are compared in 3 following cases. All results are obtained using BLOSUM45 and \( id \) cutoff being set to 0.90.

In case 1, all high-\( id \) matches (matches with identity scores higher than identity cutoff) are filtered out. This case is used as a control.

In case 2, sequences with identity scores greater than \( id \) cutoff (0.90 here), lengths longer than 5 residues, and lengths less than 90\% of query sequence are kept.

In case 3, sequences with identity scores greater than 90\%, lengths longer than 5 residues, and lengths less than 95\% of query sequence are kept.

Table 3-11 gives the accuracies in above three cases.

<table>
<thead>
<tr>
<th></th>
<th>( id ) Cutoff</th>
<th>( id ) Matches Processing</th>
<th>( \frac{1}{id^3} )</th>
<th>( \frac{1}{id^2} )</th>
<th>( id )</th>
<th>( id^2 )</th>
<th>( id^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM45</td>
<td>0.90</td>
<td>Case 1 (all filtered)</td>
<td>0.675</td>
<td>0.680</td>
<td>0.697</td>
<td>0.725</td>
<td>0.735</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Case 2</td>
<td>0.677</td>
<td>0.683</td>
<td>0.701</td>
<td>0.730</td>
<td>0.740</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Case 3</td>
<td>0.678</td>
<td>0.683</td>
<td>0.702</td>
<td>0.731</td>
<td><strong>0.742</strong></td>
</tr>
</tbody>
</table>

Different identity score zones

The weights of matches within different identity score zones are treated differently. This is based on the sequence similarity level classification (Rost, 1997). It is commonly believed that higher sequence similarity implies higher structure similarity. Again, we predict the CB513 dataset using matrix BLOSUM45, \( id \) cutoff 90\%, and some high-\( id \) matches are kept as described above. Several cases are considered.

Case 1: the identity level is divided into 3 zones, namely above 30\%, between 20\% and 30\%, and below 20\% ([0,20), [20,30] and (30, 100]). The weights of matches with identity...
scores with in the 20% to 30% zone are kept as originally defined (id powers), while the weights of the matches above and below this region are made 5 or 2 times as large as original values, respectively.

Case 2: the identity level is divided into 2 zones, namely above and below 50. The weights of matches with identity scores greater than or equal to 50% get 10 or 5 times as large as original weights. Table 3-12 gives the details. It appears that this kind of mapping between the match identity score zones and weight assignments makes no improvement to predictions.

Table 3-12. Prediction accuracies (Q₃) of weight assigning method two with identity score zones considered receiving special weights, w stands for weight

<table>
<thead>
<tr>
<th>Adjust Weights</th>
<th>(\frac{1}{id^3})</th>
<th>(\frac{1}{id^2})</th>
<th>(id)</th>
<th>(id^2)</th>
<th>(id^3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No adjustment</td>
<td>0.678</td>
<td>0.683</td>
<td>0.702</td>
<td>0.731</td>
<td>0.742</td>
</tr>
<tr>
<td>(w*5) if (id &gt; 0.3) OR (id &lt; 0.2)</td>
<td>0.689</td>
<td>0.697</td>
<td>0.713</td>
<td>0.729</td>
<td>0.731</td>
</tr>
<tr>
<td>(w*2) if (id &gt; 0.3) OR (id &lt; 0.2)</td>
<td>0.683</td>
<td>0.692</td>
<td>0.713</td>
<td>0.733</td>
<td>0.734</td>
</tr>
<tr>
<td>(w*10) if (id &gt; 0.5)</td>
<td>0.697</td>
<td>0.712</td>
<td>0.722</td>
<td>0.720</td>
<td>0.720</td>
</tr>
<tr>
<td>(w*5) if (id &gt; 0.5)</td>
<td>0.691</td>
<td>0.704</td>
<td>0.726</td>
<td>0.722</td>
<td>0.720</td>
</tr>
<tr>
<td>(w*5) if (id &lt; 0.5)</td>
<td>0.697</td>
<td>0.712</td>
<td>0.722</td>
<td>0.720</td>
<td>0.720</td>
</tr>
</tbody>
</table>

Normal and reverse direction prediction

Here we use the result obtained from BLAST using BLOSUM62 and an identity cutoff 90%. We attempted normal sequence direction prediction, the reverse sequence direction prediction and a combination of normal and reverse sequence direction prediction. The reverse order gives much worse accuracy than normal order prediction. The combination of the two methods does not improve accuracy. Table 3-13 shows the details.

Subfamilies of matches

We also used SCOP for classifying the matches. Although this may not be the best way to classify the sequences into subfamilies, this is our choice. The results are obtained from the prediction using the BLOSUM45 matrix, with id cutoff 0.90. See table 3-14 for the
prediction accuracy figure when the matches are classified into 5 and 12 subfamilies respectively, according to SCOP. Note that the accuracy declines.

Table 3-13. Prediction accuracies \((Q_3)\) of weight assigning method two with sequence direction considered

<table>
<thead>
<tr>
<th>Strategy</th>
<th>(id) Cutoffs</th>
<th>(id^3)</th>
<th>(id^2)</th>
<th>(id)</th>
<th>(id^3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal direction</td>
<td>0.682</td>
<td>0.684</td>
<td>0.694</td>
<td>0.710</td>
<td>0.717</td>
</tr>
<tr>
<td>Reverse direction</td>
<td>0.90</td>
<td>0.541</td>
<td>0.541</td>
<td>0.540</td>
<td>0.536</td>
</tr>
<tr>
<td>Combined</td>
<td>0.648</td>
<td>0.651</td>
<td>0.658</td>
<td>0.669</td>
<td>0.677</td>
</tr>
</tbody>
</table>

Table 3-14. Prediction accuracies \((Q_3)\) of weight assigning method two when matches are classified into subfamilies

<table>
<thead>
<tr>
<th>SCOP Classification</th>
<th>(id^3)</th>
<th>(id^2)</th>
<th>(id)</th>
<th>(id^2)</th>
<th>(id^3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No classification (Cutoff = 0.9)</td>
<td>0.678</td>
<td>0.683</td>
<td>0.702</td>
<td>0.731</td>
<td>0.742</td>
</tr>
<tr>
<td>5 subfamilies</td>
<td>0.64</td>
<td>0.647</td>
<td>0.666</td>
<td>0.701</td>
<td>0.720</td>
</tr>
<tr>
<td>12 subfamilies</td>
<td>0.574</td>
<td>0.58</td>
<td>0.597</td>
<td>0.634</td>
<td>0.665</td>
</tr>
<tr>
<td>No classification (Cutoff = 1)</td>
<td>0.738</td>
<td>0.754</td>
<td>0.810</td>
<td>0.902</td>
<td>0.931</td>
</tr>
<tr>
<td>5 subfamilies</td>
<td>0.691</td>
<td>0.708</td>
<td>0.764</td>
<td>0.872</td>
<td>0.920</td>
</tr>
</tbody>
</table>

Accuracies for proteins of different lengths

For these experiments, the dataset is divided into four groups according to sequence lengths, namely, tiny (\(<100 \text{ residues, 154 sequences}\)), small (\(>100, \text{ but } \leq 200 \text{ residues, 216 sequences}\)), large (\(>200, \text{ but } \leq 300 \text{ residues, 84 sequences}\)), and giant (\(>300 \text{ residues, 58 sequences}\)). We only experimented with one case, namely the BLOSUM45 matrix, weight function \(id^3\). No optimization was attempted. The accuracies vary from 0.911 for “tiny” group to 0.948 for “large” group. Table 3-15 shows the prediction accuracies for proteins of different lengths.

Table 3-15 the accuracies of predictions for proteins of different lengths

<table>
<thead>
<tr>
<th>Groups</th>
<th>CB513</th>
<th>Tiny</th>
<th>Small</th>
<th>Large</th>
<th>Giant</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Q_3)</td>
<td>0.931</td>
<td>0.911</td>
<td>0.936</td>
<td>0.948</td>
<td>0.940</td>
</tr>
</tbody>
</table>
Application of artificial intelligence (AI) approaches

We use AI approaches to determine the final secondary structure prediction based on the amino acid sequences and normalized scores of some secondary structure elements. All the prediction accuracies are measured using $Q_3$. We consider some popular AI approaches, including the Naive Bayes Classifier (NB), Decision Tree (DT), Neural Networks (NN), and Support Vector Machines (SVM).

**AA sequence based training**

Figure 3-5 shows the accuracy of different AI approaches plotted against different window sizes. It shows the prediction accuracies of different AI approaches for amino acid sequences. Our results are similar to the results in the literature, i.e., the accuracy based only on sequence is around 60%.

![Figure 3-5. Prediction accuracies of different AI approaches from amino acid sequences](image)

**Normalized score-based prediction**

**Weight assignment method 1**

Here the weight function for the matches is $1/\sqrt{Sim}$ and the substitution matrix used in BLAST is PAM 30. We produce a normalized score file. We divided the dataset into training and test files in the same way as above. We only use SVM to classify the normalized scores instead of assigning secondary structure to residues only according to the highest normalized
scores of residues. The comparison is shown below in Table 3-16. In this case, the improvement is even more obvious, it is almost a 4% increase in average accuracy.

Table 3-16. Comparison of prediction accuracy between SVM at different window sizes and previous method on test set.

<table>
<thead>
<tr>
<th>Window Size</th>
<th>SVM</th>
<th>Previous Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>76.44</td>
<td>74.59</td>
</tr>
<tr>
<td>3</td>
<td>77.16</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>77.75</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>78.02</td>
<td></td>
</tr>
</tbody>
</table>

Weight assignment method 2

In these experiments, the weight function is the identity score cube \( (id^3) \), and the substitution matrix used is BLOSUM45. All the matches that are better than 90% are discarded. The prediction process consists of two procedures, training and testing. We randomly partitioned the query dataset into training and testing sets at a proportion of 4:1. In this case, when manually assign secondary structure to a residue according to highest element normalized score of that residue, we obtained the best result of prediction accuracy of \( Q_3 = 71.99\% \) for the test set sequences. We note that there is a little improvement in prediction accuracy among decision trees, neural networks and support vector machines. Fig. 3-6 gives the comparison among AI approaches at different window sizes (note that the prediction accuracy using previous method for test set is 71.99%).

Fig. 3-6. Normalized score based prediction using AI approaches at different window sizes
Working on the KS267

We used the weight assigning method 1 (the reciprocal of similarity or identity scores and their powers \(1/sim^d\) or \(1/id^d\) as the weights assigned to matches) to predict KS267 data sequences. No optimization steps are taken, i.e. only the basic methods are applied. Table 3-17 shows the prediction accuracies when BLOSUM62 and PAM30 are applied at alignment step.

<table>
<thead>
<tr>
<th></th>
<th>1/id</th>
<th>1/sim</th>
<th>1/sim^2</th>
<th>1/\sqrt{sim}</th>
<th>1/\sqrt[3]{sim}</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM62</td>
<td>0.721</td>
<td>0.735</td>
<td>0.735</td>
<td>0.75</td>
<td>0.754</td>
</tr>
<tr>
<td>PAM30</td>
<td>0.773</td>
<td>0.783</td>
<td>0.783</td>
<td>0.793</td>
<td>0.793</td>
</tr>
</tbody>
</table>

Case applications

We displayed a number of representative sequences according to SCOP classification in order to demonstrate the prediction results. Those sequences include sequences of all-\(\alpha\), all-\(\beta\), \(\alpha+\beta\), \(\alpha/\beta\) (details see appendix A).
CHAPTER 4. DISCUSSION

The best results obtained

We only focus on some representatives with some of the best results. Following Table 4-1 summarizes our prediction accuracy.

Table 4-1. The best prediction results of different combinations of influence factors

<table>
<thead>
<tr>
<th>Weight Assigning Methods</th>
<th>Matrix</th>
<th>Basic</th>
<th>sim or id Cutoffs</th>
<th>Different Zone (no cutoff)</th>
<th>Accessibility</th>
<th>Occurrences</th>
<th>Subfamily</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>99%</td>
<td>90%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1/\sqrt[3]{sim}$</td>
<td>PAM30</td>
<td>0.750</td>
<td>0.706</td>
<td>0.668</td>
<td>0.756</td>
<td>0.757</td>
<td>0.750</td>
</tr>
<tr>
<td>$id^3$</td>
<td>BLOSUM45</td>
<td>0.931</td>
<td>0.825</td>
<td>0.735</td>
<td>0.932</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If the weights of the matches are defined as the reciprocal of the power of the similarity scores, the best prediction accuracy obtained is 0.757. When the weights are defined as the identity score powers, the best accuracy is 0.932. It is obvious that 0.932 is over-estimated, since in real application, a new query sequence cannot have a perfect match while doing alignment against any databases. So we filter out some good matches with similarity/identity scores higher than some cutoffs. The best prediction accuracy for cutoff 99% is 0.825, and that for 90% is 0.735. If some optimization methods (either separate or combined) are applied, we can expect some improvement of the accuracies. Actually we have seen such tendency from these optimization methods.

We may notice that the perfect matches play very important roles in the accuracy of the prediction. Even 1% cutoff decrease leads to a sharp drop in the prediction accuracy. It is not unusual to see the drop of accuracy in the second weight assigning method prediction. In the first weight assigning method, the weight of match is defined to be the inverse of similarity score. When the similarity is relatively high, the weight and the contribution of that match should be small. But in our result, it is not necessarily this case. There is still space to define more appropriate functions to implement this intention.
Application of AI approaches

Using AI approaches to determine the final prediction according to previously obtained normalized scores of being some specific secondary structures of residues.

Based on composition of sequences.

Here we only show the best result obtained by some specific AI approaches. Table 4-2 summarizes the prediction accuracies based on residues of query sequences only.

Table 4-2 Prediction accuracies based on residues of query sequences

<table>
<thead>
<tr>
<th>Naïve Bayes</th>
<th>Decision Tree</th>
<th>Neural Network</th>
<th>Support Vector Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>45.05</td>
<td>53.15</td>
<td>61.40</td>
<td>60.04</td>
</tr>
</tbody>
</table>

The highest accuracy is around 60%. This result is consistent with the highest levels of prediction methods focusing on local information.

Based on normalized scores

Again, we just show the best results for each approach. Table 4-3 summarizes the prediction accuracies based normalized score file.

Table 4-3. Prediction accuracy and improvement While AI approaches introduced

<table>
<thead>
<tr>
<th>Weight Assigning</th>
<th>Matrix &amp; Cutoff</th>
<th>Test Set control</th>
<th>DT</th>
<th>NN</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$id^2$</td>
<td>B45 (90%)</td>
<td>71.99</td>
<td>72.10</td>
<td>72.30</td>
<td>73.58</td>
</tr>
<tr>
<td>$1/\sqrt{sim}$</td>
<td>PAM30 (Basic)</td>
<td>74.59</td>
<td>N/A</td>
<td>78.02</td>
<td></td>
</tr>
</tbody>
</table>

We just use these AI approaches at the last step of prediction--- to determine the secondary structures of residues based on the normalized scores of residues in one of the three states: helix, strand and other). It is very likely that using these approaches may improve the accuracies. This tendency is well confirmed by the primitive applications shown above by the consistent increments from these approaches. So far, we cannot draw a conclusion for how much the increments might be, because of the limit of the calculation ability of our PC (we only test these methods with window size from 1 to 7). It seems that SVM shows the best classification capability during the learning process.
Comparison with Other Methods

Rost comments, "There is no value in comparing methods evaluated on different
datasets". However, we feel that looking at the prediction levels from different methods can
still give us some sense to compare. Table 4-4 (Rost, 2001) gives the comparison between
current study and some popular prediction algorithms.

<table>
<thead>
<tr>
<th>Method</th>
<th>Q_3 (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROF</td>
<td>77.0</td>
</tr>
<tr>
<td>PSIPRED</td>
<td>76.6</td>
</tr>
<tr>
<td>SSpro</td>
<td>76.3</td>
</tr>
<tr>
<td>Jpred2</td>
<td>75.2</td>
</tr>
<tr>
<td>PHDPsi</td>
<td>75.1</td>
</tr>
<tr>
<td>PHD</td>
<td>71.9</td>
</tr>
</tbody>
</table>

Table 4-4. Accuracy of Secondary Structure Prediction Methods (1)

<table>
<thead>
<tr>
<th>Method</th>
<th>1/3\sqrt{sim} (3)</th>
<th>1/3\sqrt{id} (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic</td>
<td>Cutoff 99%</td>
</tr>
<tr>
<td>Current</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>75.7</td>
<td>70.6</td>
</tr>
<tr>
<td></td>
<td>93.2</td>
<td>82.5</td>
</tr>
</tbody>
</table>

(1) Dataset and sorting: The results except current method are compiled by EVA (Evaluation of
automatic structure prediction servers, Rost & Eyrich, 2001). All methods for which details are
listed have been tested on 195 different new protein structures (EVA version February 2001).
None of these proteins was similar to any protein used to develop the respective method.

(2) Three-state per-residue accuracy, i.e., number of residues predicted correctly in one of the
three states, helix, strand, or other (conversion of DSSP states (HG) → helix, (EB) → strand; note
that the per-residue accuracy tends to favor methods overpredicting nonregular structure).

(3) Weight assigning methods

But what does a prediction accuracy being 76% mean? In real application, it does not
mean that you can get 76% of the residues of an unknown query sequence predicted
correctly. Actually, the actual accuracy can vary from 50% to 90% levels. There is a dilemma
that you are always worrying about the performance of the prediction for an unknown query
sequence. Nowadays, most of the methods provide an index measuring the reliability of the
prediction for each residue. (Rost, B. 1994) It has been shown that the prediction accuracy is correlated with reliability. In other words, if most of the residues are predicted with high reliability, then the prediction is probably more accurate than average.

We can also define our own accuracy index for our prediction. For each residue in the query sequence, we can get some matches with known structure (details see Ways to improve part). According to the occurrences and the similarity scores, we can get a reliability index after normalization.

Another way to evaluate the confidence is to look at the results from many other different methods.

**Some influence factors**

**Weight assigning methods**

It seems that there are mainly two sources for prediction accuracy improvement. One is enlarged database, the other is more divergent profiles. During evolution, some important motifs are kept to maintain the specific structures and functions. It is possible that structurally similar proteins can have sequence identities below 10%. (Rost, 1997) If this kind of conservation can be grasped by some methods, then it must be helpful to do the structure prediction base on these motifs with known structures. In some sense, the more divergent the motifs, the more important their contributions might be. Position specific iterated BLAST (PSI-BLAST) is pretty sensitive to weak but biological relevant sequence similarities (Altschul,1997). Theoretically, it can be used to get these conserves. Based on this idea, we want to put more weight more on the more divergent matches while doing the BLAST and the prediction. This is our first method of weight assigning, the reciprocal of the power of similarity scores.

It is commonly believed that similarities between the sequences of two proteins infer similarities between their structures, especially when the sequence similarity is greater than 50%. Yet this inference is not confident while the similarity level of sequences lies between 20%–30%. Therefore we can try a different weight assigning method using the identity scores and their exponents as the weights of the matches. Considering different
similarity/identity zones, we can make some adjustments to the assignments when the identity levels are in some specific zones, either raise the contribution of the corresponding matches or decrease them.

Our results show that the second method consistently yields better results in various cases. That means that the method based on the assumption that similar sequences imply similar structures gives more reliable prediction. If the assumption that more diverged profiles lead to more accurate prediction, this might be because the matches we obtained while doing the BLAST are not diverged enough. By changing parameters of BLAST, modifying substitution matrices or even algorithms, we can hopefully get better results.

**Substitution matrix**

We may notice that the matrices used make much difference in the prediction accuracies. Different ways of combining substitution matrices and weight assignment methods yield remarkably different accuracies. While using the identity score and its power as the match weight (weight assigning method 2 in current research), BLOSUM45 BLAST result leads to higher prediction accuracy, using reciprocal of similarity score as the match weight, PAM 30 gives better result. As for BLOSUMs and method 2, if more related segments are clustered, the accuracy increases. That gives us the hint for developing more “extreme” matrices to improve, e.g. “BLOSUM30”.

**Nonlinear weight assignment scheme for matching occurrences**

Intuitively, the more matches we get, the more reliable the prediction could be. For example, usually, while doing BLAST, the middle part of the query sequence has more matches than the terminal parts. It is reasonable to infer that the prediction for the middle part should be more reliable. Therefore, it is also reasonable to give more weight to the position with relatively more occurrences of matches. We can imagine the non-linear relationship between match occurrence and reliability of prediction. This is the reason why the sigmoid function is used here as a way to modify original weight definition, which is just one of the choices. It is only used in one case, the first weight assignment method. Although we cannot
see any improvement of the prediction, we might see some positive (or negative) result if we carefully choose some other functions (instead of the sigmoid).

**Solvent accessibility of residues**

Some special groups of residues in a protein sequence play the crucial role in protein folding and structure. It is hypothesized that some kinds of folding nucleus are the start of protein folding (Ptitsyn, 1998; Ptitsyn & Ting, 1999). We thus assume that the buried residues are more important to the structure than the exposed ones. To see the influence of the accessibility of the residues on structures, we simply define discrete functions to modify the weights of the matches. For example, if a residue from a match is calculated to be buried, its weight is twice as the original match weight; if exposed, its weight is the same as the match weight; etc. Because of the restriction of the calculation ability of NACCESS, we cannot get the complete information for all the matches. Using the incomplete information, we get a little improvement of prediction accuracy. Probably, this improvement is not significant, but we guess this modification is a reasonable way to improve. By defining appropriate criteria (percentages) for buried, exposed, and intermediate states, by defining new weight assigning functions (possibly exponential, or others, instead of very simple discrete functions), by enhancing the calculation ability to obtain residue exposure percentage, we may expect much more influence from this information on prediction accuracy.

**Sequence direction**

Park (Park, 2000) etc. reported their findings on the significance of sequence direction in protein secondary structure and prediction. They found a high conservation of secondary structure propensity in the reverse direction. They also reported comparable high prediction accuracy while using reverse protein sequences. A by-product of their research is providing one reliability index for the prediction, i.e. the agreement of forward and backward combined prediction. Trying to testify their findings, we also made some efforts. When we found a mismatch of forward and backward prediction, we would assign some secondary structure element to that position according to the higher normalized score. However, we could not
draw similar conclusion as the authors did. We got really low prediction accuracy while using backward and combined methods.

**Proteins of different types and different lengths**

Many methods are relatively mature for predicting specific kinds of proteins. Some methods are developed specially for coiled-coil regions, some are designed specially focus on β-turns (Shepherd AJ, 1999; Kolinski, 1997). It is believed that helices are relatively easier to be predicted than sheets. The problem of predicting β-sheet regions has been considered difficult because β-sheets typically range over several discontinuous sections in an amino acid sequence, and their sequences exhibit long distance dependency. Therefore we try to evaluate the prediction accuracy of our method on the all beta proteins in the dataset. According to the database from SCOP, we classify the 513 sequences into five subfamilies, namely, all alpha, all beta, α+β, α/β, and others (the ones that cannot found in SCOP database). The classification result is α-99, β-123, α/β-145, α+β-64. We evaluate the accuracies for all the different types. We use the first weight assignment method (reciprocal of sim powers) and consider the occurrence influence to see the prediction. We notice that the accuracy for all -β proteins is within 2% difference (a little bit lower) comparing to that of all α groups, which means the two accuracies are comparable.

Many protein-structure-related procedures, e.g., protein folding, consider proteins of different lengths. Some focus on small proteins (Enoch et al., 1999), some on proteins of larger sizes. We take a look at the accuracies of our method on proteins of different sizes. We simply divide the 512 sequences with matches founded by BLAST into 4 groups, namely, tiny (≤100 residues, 154 sequences), small (>100, but ≤ 200 residues, 216 sequences), large (>200, but ≤ 300 residues, 84 sequences), and giant (>300 residues, 58 sequences). We just look at the accuracies using the basic second weight assigning method (weight defined as \( id^3 \) without optimization). The accuracies of prediction for proteins of different sizes are quite similar (from 91.2% for “tiny” group to 94.8% for “large” group). If this prediction is representative enough, it seems that our method works quite well for proteins of various lengths.
Subfamilies of matching segments

Several papers (Ting & Jernigan, 2002; Ptitsyn & Ting, 1999; Ptitsyn, 1998) focused on finding possible folding nucleus from which the authors assumed the folding is started. In these papers, the authors used phylogenic method to classify families of interest into subfamilies according to the biological origins of the family members instead of being based on the structure and function information. During this process, they raised a so called "consensus principle", which states that each subfamily is equally important in defining the common folding nucleus. A residue cannot be regarded as part of the nucleus even when it is not consistent in the alignment in only one of the subfamilies.

This gives us some hint in the secondary structure prediction. We can imagine that our matches for some query sequence can also be classified into subfamilies (if we can find an appropriate way to classify them). Ideally, we can classify them into evolution-related subfamilies. According to the divergence degree, some subfamilies may include fewer members than others. While assigning weights to these members, we have to consider these numbers of family members. If a subfamily contains a lot of members, each of the members should weigh less than those from subfamilies including fewer members, since each subfamily of matches is equally important to the structure of the query sequence.

The problem is that we cannot find such an ideal way to classify the family of matches, since the BLAST gives only segments of "good" matches. The matches are so divergent that it is almost impossible to classify phylogenically. We thus follow another approach to classify them according to the structures. Again, we use SCOP to classify all proteins in PDB into 5 or 12 subfamilies, and thus classify all matches into 5 or 12 subfamilies. Define a simple strategy to assign the weight of a subfamily to its members.

The prediction result is not improved. As we mentioned above, we cannot find a reasonable way to classify the matches. But once this problem is solved, we can expect better results to some extent.

Different ways of translation

As mentioned above, DSSP assigns eight elements of secondary structure to residues. People usually use a three-state prediction. Therefore there are ways to translate the DSSP
alphabet into the three-letter code. In present study, we just follow the CASP (Critical Assessment of Techniques for Protein Structure Prediction) (Mount, 1995) assignment method. By just converting G (3_{10} helix) and B (bridge, a single residue β-strand) to nonregular structure, the results will be much better (Rost, 2001). There are still other ways to do the translation. Some of them also consider the neighbor compositions of secondary structure elements. We do not try other ways of translation, instead, we use some AI approaches in the hope to avoid some unreasonable assignments.

Using AI approaches

A lot of artificial intelligence approaches, like Neural Networks (NN), Support Vector Machines (SVM), have been used extensively to predict secondary structure. (Qian, et al. 1988; Rost et al., 1993; S. Hua et al., 2001; Cai et al., 2003; Guo, 2004) Based on the matching information, we calculate the normalized scores of each position to be some secondary structure element. We choose the highest normalized score to be the prediction. Sometimes, the normalized scores of being E, H or C are so close or even equal, so choosing the highest is difficult or unreasonable. In some other cases, using our original method may lead to some impossible predictions. For example, CCCECCC is an impossible composition in real life. We try to use AI algorithms to learn from training set. Let the computer to choose the most possible normalized score and do the prediction accordingly.

We now take a look at our partial results.

Based on the composition of query sequences, we obtain prediction accuracy comparable with the results in literature, which is around 50~60% level.

As expected, we are interested in the finding that using AI approaches, we get consistently higher accuracies than our previous method (choosing the highest normalized score as the final prediction). More impressively, the SVM approach gives us a 3~4% improvement of accuracy. We cannot get a complete observation of what is the optimal window size for the improvement because of the computational limits. Although these approaches work on different weight assigning methods, on different parameters, we can
draw some conclusion based on the consistency of improvement from our partial result. This is a new application of AI approaches in secondary structure prediction area.

**PSI-BLAST and BLAST**

BLAST (Altschul *et al*., 1990) is a popular tool for searching protein and DNA databases for sequence similarities. It is a heuristic that tries to optimize the similarity measure. It has been refined several times (Altschul, *et al*., 1997). Based on the prototype, the ‘two-hit’ method only extends the non-overlapping word pairs on the same diagonal, and within a specific distance. Therefore, the algorithm saves time in the extension. By introducing possible gaps in the alignments, the gapped BLAST algorithm can increase the threshold T and thus yields greater speed. Usually profile and motif search method is more sensitive to detect distant relationships than pairwise comparison methods. BLAST searches are iterated with the position-specific score matrix generated by the previous run, which is Position-Specific Iterated BLAST search. It should get more distant homologues of the queries from database than regular BLAST methods.

In current research, we are using BLASTc13. Our main purpose is to test the structure relationships between local alignment segments and the target sequence. A paper (Przybylski, D. 2000) reported a comparison of the results obtained from BLAST and PSI-BLAST. The prediction accuracy difference is less than two percent. We may also download the PSI-BLAST executables, to verify the expected difference of the accuracies obtained using these two alignment strategies. We guess that the situation will be similar except for little difference in the accuracies.

**Way to improve with an example for analysis**

**Reliability assessment**

How reliable is a prediction? Many methods provide reliability indices to measure the performances (Rost, 1993). Several papers (Rost, 2000) showed the relationship between the reliability and the prediction accuracy.
We define our reliability index by considering the occurrences and similarity scores of the matches of BLAST. The maximum possible number of matches that a residue of query sequence may have is the total number of sequence matches of BLAST. For each residue of the query sequence, we can get the weight sums of the matches, we can also get the separate weight sums of different matches (different secondary structure elements, H, E or C). We then find the maximum sum from the three values. We therefore define the reliability of the residue to be the maximum value divided by the total number of match sequences.

We are not showing the total accumulated appropriate reliability which can make the high-reliable residues yield more than 88% of average accuracy which is not a tough work to do. Instead, we just show a method to evaluate the reliability of some special test cases. This is just one way to define the reliability of prediction.

One of our ultimate goals is to go from 2D prediction to 3D applications. As we can see in our current work, there are so many factors that may influence the prediction results. Also we have databases with known structures available. Our next trial is trying to answer how we can combine the information of the known structures, of other factors, to define more accurate reliability measures.

This is an example with reliability 0.01.

Protein Name: 1C75:A.
General information: all alpha, length is 71.
Number of matches: 128 (BLOSUM45).
Prediction Accuracy Q₃ = 0.788732 (weight assigning $id^3$, >90% perfect matches filtered out).

There are 66.2% high reliability residues (residues with reliability>0.01, indicated with *). The correctly predicted high-reliability residues are denoted with “+”. The average accuracy for high-reliability residues is 0.93617.

<table>
<thead>
<tr>
<th>Actual SS</th>
<th>Prediction</th>
<th>Reliability</th>
<th>Correctly</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CCHHHHHHHTHHHHCTTSSSCSSCTTTHHHHTH H HH H H H H H H H H H H C B T T B</td>
<td>50</td>
<td>-+++++++++++++++++++++++++++++++++++++++++++++++++++---------- 50</td>
</tr>
<tr>
<td>1</td>
<td>CCHHHHHHHCCHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCHHHHHHHHHHHH H H H H H H H H H H H H H C C E C E C E</td>
<td>50</td>
<td>++++++++++++++++++++++++++++++++++++++++++++++++++++---------- 50</td>
</tr>
</tbody>
</table>
Actual SS  51  CSCSSGHHHIIIIIIIIIIIIIIIHTCC  71
Prediction  51  CCCCCCCCCCCCCCCCCCC  71
Reliability  51  *----------------------  71
Correctly  51  +----------------------  71

If we could define proper reliability measures, we may expect to find the parts with high confidential predictions. For above example, we can thus say, the prediction of part 2~41 is more-likely-to-be-correct, but the prediction for part 52~71 is less reliable. Actually it is the fact in this case.

Another choice might be the difference between the highest normalized score and the second highest normalized score. We are not showing the application of this definition here.

However, we are not providing a satisfactory reliability index that works very well for all the sequences, which is definitely very important for a prediction method. That will be part of our future work.

Others

We have discussed the influences of some factors on prediction accuracy, including weight assigning methods, substitution matrix, solvent accessibility, occurrences, sequence order, types and lengths of query sequences, classification, etc. If we could design more expressive weight assigning method to represent the conservation of structure motifs, if we could design more extreme matrices, if we could complete the calculation of residue solvent accessibility or choose ideal correlated accessibility values and define more accurate criterion to determine residue status, if we could find appropriate non-linear relationship functions between match occurrence and reliability, if we could successfully combine all the factors together, we believe the prediction result will be better. But we are not hoping a drastic increase in the accuracy, since all the information is obtained through BLAST, which is impossible to cover everything that can reveals the nature of protein structures. Ideally, we may find “complementary” methods, whose prediction errors are independent with that of current method, to increase the performance of prediction.
Brief conclusion and problems

The present study shows comparable prediction accuracy with currently popular methods. We could draw some conclusions:

1) It verifies to some extent the roles of some factors that might influence prediction accuracy, such as, the more occurrences of the matches, the more reliable the prediction; hydrophobic residues might be more important in the formation and maintaining protein structures, therefore, considering the hydrophobicity of the residues of matching segments does no harm in secondary structure prediction; matches of different similarity levels play different roles in prediction; considering normal and reverse order of query sequence and combine both of them in prediction will not help in our method; etc.

2) We find that our method works almost equally well for sequences of different sizes in CB513.

3) Our method yields comparable prediction accuracies for different kinds (all alpha, all beta, $\alpha+\beta$ and $\alpha/\beta$) of proteins in the query dataset. The accuracy for all beta sequences is 1.5% less than all alpha sequences.

4) We use some AI approaches in the last step of our prediction, to determine the final secondary structure element according to normalized score file, and gain 3~4% improvement in accuracy based on the method of choosing highest-normalized score-corresponding secondary structure element.

It is possible that we cannot find matches for some query sequences using some specific parameters to do BLAST. For those with very few matches, the prediction accuracies are not stable (either very low or very high). The starting and ending parts of sequences are predicted with relatively lower reliability because of fewer matches. There is still a lot to do for real application.

Lessons learned

1) During the process, I have now a relatively clear comprehension in the secondary structure prediction field, including the significance and applications of prediction, current
methods and accuracy levels, possible influence factors of prediction, evaluations of prediction methods, some background theories, and scientists with their research interests. I realize that secondary structure prediction is indeed a basic step for many other procedures, like protein folding, protein interaction, and 3D structure study. Also it raises some new topics of interests.

2) BLAST and PSI-BLAST are popular tools in searching protein and DNA databases for sequence similarities. From the iteration of BLAST search, PSI-BLAST is more sensitive to weak but biologically related similarities and thus can identify fairly divergent evolutionary family members.

3) The information of segments from local alignment does help in structure prediction. But it is impossible to cover all information required by the complicated protein structures. We defined some factors that maybe important for protein structure, and the accuracies improve a little by considering these factors. But these improvements are within the scope of information included in the segments. We cannot expect a drastic improvement beyond these segments by just defining better functions for the factors. More “extra-segment” efforts are needed to improve prediction accuracy.

4) In the process of eliminating “unreasonable” predictions, we tried to come up with a “rule of reality” for the program to follow. For example, if a helix or a sheet is predicted with too few consecutive residues, then something must be done to adapt it. But there are too many restrictions, and there are various special cases that it is almost impossible to grasp or the “rules”. Sometimes the seemed rules even conflict with each other. Under this situation, I begin to seek help from artificial intelligence. Learning from training set, and using the information gained from learning in the prediction of test set seems to be a more appropriate way. But AI is a really new thing to me at the beginning. After this project, I have a basic understanding in its main idea, its popular approaches, its implementation, its power of classification and its application in other areas.

5) AI approaches are pretty successful in this area. In our study, we use them in last step as classifiers. The remarkable improvement shows their values in various researches. We are thinking about ways to combine all the factors mentioned above. The flexibility of AI approaches makes them good candidates of choices.
APPENDICES

Appendix A. Case Applications

The identity score \((id)\) or similarity \((sim)\) cutoff 90% is applied to filter out the perfect matches. We choose some representatives from SCOP database to show our prediction results. In the protein name line, we give out the classification, length, and PDB code for the sequence.

**Weight assigning method 1**

The reciprocal of similarity or identity scores and their powers \((1/sim^d \text{ or } 1/id^d)\) are used as the weight for the local alignment matches.

Substitution matrix used: PAM30

**Protein Name: mailnly-Alpha_length_146_3SDH:A**
Length= 146 #matches= 86 Prediction Accuracy Q3_sim_3rt= 0.767123
Actual SS 1 CCHHHHHHTCCHHHHNNHHHHNNHHHNNHHHHHCGGTTGG 50
Prediction 1 CCHHHHHHNNHHCHHHHHHHHHHHHHHCCB HHHH 50
Actual SS 51 GGGGTTGGGGGCHHHHNNNNHHHHHHNNHHHHHHNNHHTTCHHHHNNHNNHHHHHH 100
Prediction 51 HHHCCHHHHCHHHHHHCHHHHHHNNNNNNNNHHCHCCCCCCCCEHHB CCHCC 100
Actual SS 101 HHTTCCCHHHNNHTTNNHHHHHHHHNNHHHTTCCCHHHHHHHHHHHHHHHHTT 146
Prediction 101 CCCCCCCCCCCCHHHHHHHHHHHHCCCCCCCCCCHHHHHHHHHHHHHHCC 146

**Protein Name: mailnly-Alpha_length_116_1DLW:A**
Length= 116 #matches= 51 Prediction Accuracy Q3_sim_3rt= 0.767241
Actual SS 1 CTTTTTTTSIHHHHHHHHHHHHHHHHTTTTTGGGTTCCCHHHHNNHHHHHHHH 50
Prediction 1 CCCCCCCCCCCCCCCCCCCCCCHHHHHHHHHCCCHHHHHHCCCHHHHHHCC 50
Actual SS 51 HHTTCCSSCCCSCCHHHHHTTSSCCCHHHHHHHHHHHHHHHHTTCCCHHHHH 100
Prediction 51 HHCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHCCCCCCCCCH 100
Actual SS 101 HHHHHHHHTTHHHHCCC 116
Prediction 101 HHHHHHHHCCCCCCEEEC 116
Protein Name: mailnly-Alpha_length_71_1c75:A
Length= 71 #matches= 54 Prediction Accuracy Q3_sim_3rt= 0.647887
Actual SS 1  CCNHHNNHHTNNHNCTTSSCSSCCCTTNNHHTTNNNNNNNNHNBTTB 50
Prediction 1  ECHNNHHNNHNCCHCCSSCCCCCCCCCCCCCCCCCCHHNNNNHHHNBTTB 50
Actual SS 51  CSHSSCHNHNNNNNNNTTCC 71
Prediction 51  CCCCCCCCCCCCCCCCCC 71

Protein Name: mailnly-Alpha_length_63_1qoj:A
Length= 63 #matches= 46 Prediction Accuracy Q3_sim_3rt= 0.904762
Actual SS 1  CCCCCCCCHHHHHHHHHTCHHHHHHHCCTTNNNNNN 50
Prediction 1  HHHHHHHHHHHHHHTTCHHHHHHCCCTTNNNNNN 50
Actual SS 51  HNNNNNNNNNNNNTT 63
Prediction 51  HNNNNNNNNNNNNTT 63

Protein Name: mailnly-Alpha_length_37_1eci:A
Length= 37 #matches= 20 Prediction Accuracy Q3_sim_3rt= 0.567568
Actual SS 1  CCCCCHHHHHTNNHNHTCCHHHHHHTTTTCCCHHNNNNNNN 37
Prediction 1  CCCCCECCCECCHCCCHHCCCCCHHNNNNNNN 37

Protein Name: all Beta_length_124_1dqi:A
Length= 124 #matches= 67 Prediction Accuracy Q3_sim_3rt= 0.629032
Actual SS 1  CCGGGEECSBTTBCCSCEEEEEETTHEEEEEESSSSCCSSSBSBCC 50
Prediction 1  HEECCCCCCCCCECCCCHECCCCCCCCCCCCCCCCCCCCCCCBBB 50
Actual SS 51  EEEEEEEEETCCSCCECCEEEEECCCBBCCBTBTTCCSBCBCCSBBCEEEEEECS 100
Prediction 51  EEEEEECCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCBBCCBBCEEEEEECS 100
Actual SS 101  CEEEEEEEEEEETTEEEEEEEEE 124
Prediction 101  CCCCCCCCCCCEEEEEECCCCCCCHC 124

Protein Name: all Beta_length_109_1bww:A
Length= 109 #matches= 19 Prediction Accuracy Q3_sim_3rt= 1
Actual SS 1  CCCCCEEEECSSSEEEECTTCCEEEEEEESSSSCCTTCEEEEEEECTTSCCCCEEE 50
Prediction 1  CCCCCEEEECSSSEEEECTTCCEEEEEEESSSSCCTTCEEEEEEECTTSCCCCEEE 50
<table>
<thead>
<tr>
<th>Protein Name: all Beta_length_80_1B4R:A</th>
<th>Length= 80 #matches= 30 Prediction Accuracy Q3_sim_3rt= 0.55</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual SS 1</td>
<td>CEEEEECCCSSCCBSSEEEEEEESCSSCSEEEREEECSCSCCEEEETEEEEE 50</td>
</tr>
<tr>
<td>Prediction 1</td>
<td>CEEEEECCCSSCCBSSEEEEEEESCSSCSEEEREEECSCSCCEEEETEEEEE 50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protein Name: Alpha-Plus-Beta_length_150_1FMO:E</th>
<th>Length= 150 #matches= 89 Prediction Accuracy Q3_sim_3rt= 0.54</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual SS 1</td>
<td>CCCEECCCCCCCSSCCSCHHHHHHHHHHHTTTTCCEEEEEEESCCCCCCCCCEE 50</td>
</tr>
<tr>
<td>Prediction 1</td>
<td>CCCEECCCCCCCSSCCSCHHHHHHHHHHHTTTTCCEEEEEEESCCCCCCCCCEE 50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protein Name: Alpha-Plus-Beta_length_110_1A2P:A</th>
<th>Length= 110 #matches= 61 Prediction Accuracy Q3_sim_3rt= 0.763636</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual SS 1</td>
<td>CCCEEECCCSHCHHHHHHHHHHSSCCTEECHHHHHHHHHTTTGCTHHHHSTTC 50</td>
</tr>
<tr>
<td>Prediction 1</td>
<td>CCCEEECCCSHCHHHHHHHHHHSSCCTEECHHHHHHHHHTTTGCTHHHHSTTC 50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protein Name: Alpha-Plus-Beta_length_68_1HYW:A</th>
<th>Length= 68 #matches= 53 Prediction Accuracy Q3_sim_3rt= 0.823529</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual SS 1</td>
<td>EECEEECCCTTCCSCCCTTCCCCEEEESCCSSSCSCEEEETTECEEES 100</td>
</tr>
<tr>
<td>Prediction 1</td>
<td>EECEEECCCTTCCSCCCTTCCCCEEEESCCSSSCSCEEEETTECEEES 100</td>
</tr>
</tbody>
</table>
Protein Name: Alpha-Slash-Beta_length_355_1LUC:A
Length= 355 #matches= 96 Prediction Accuracy Q3_sim_3rt= 0.71831
Actual SS 1 CCCCCHHHHHHHHHHHTSCCEEEEECCCCCCTTTTTHHHHHHHH 50
Prediction 1 HHHHHHHHHHHHHHHHCHHHCCCCCCCCCCCCCCCCCCCCCCCCHHHHHHHH 50

Actual SS 51 HHHHTTCCCCCCCCCCCCC 68
Prediction 51 HHHHCCCCCCCCCCCCC 68

Protein Name: Alpha-Slash-Beta_length_260_1JCJ:B
Length= 260 #matches= 85 Prediction Accuracy Q3_sim_3rt= 0.896154
Actual SS 1 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCTTTTGGGSS 50
Prediction 1 HHHHHHHHHHHHHHHTTTTTTEEEEEECCTTTTCHHHHHHHHHHHHHH 50

Actual SS 51 HHHHCCCCCCCCCCCCC 68
Prediction 51 HHHHCCCCCCCCCCCCC 68
Protein Name: Alpha-Slash-Beta_length_247_1TPH:1
Length= 247 #matches= 94 Prediction Accuracy Q3_sim_3rt= 0.955466
Actual SS 101 EEEEECCHHHHTTCCCHHHHHHHHHHHHHHTCEEEEEECCHHHHTTTHH 150
Prediction 101 EEEEECCHHHHCCCCCHHHCHHHHHHHHHHHHHHSEEEEECCHHHCCCHH 150

Actual SS 151 HHHHHHHHHHHHTCSEEECCCCSCSSCCCCHHHHHHHHHHHHHHTTTTTCE 200
Prediction 151 HHHHHHHHHHHHCCCCCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHCCCE 200

Actual SS 201 EEEESSCSHHHHHHHHHHHHHHHHHHTTTTTTTTEEEEESSCCHHHHHHHHTT 250
Prediction 201 EEEEECCCCCHHHHHHHHHHHHHHHCCCCCCEEEEECCCCCCCCCCCCC 250

Actual SS 251 CCCCCCCCCC 260
Prediction 251 CCCCCCCCCC 260

Protein Name: mailnly-Alpha_length_l46_3SDH:A
Length= 146 #matches= 79 Prediction Accuracy Q3_id3= 0.945205
Actual SS 1 CCCCCEEEEEEECSSCCCCHHHHHHHHHHHHHSSCCCTTEEEEECECGGGHHHH 50
Prediction 1 CCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHCCCCCCCCCCCCCCCCCHHHHHHH 50

Actual SS 51 HHHHCCCEEEEEECCCCCCCCCCCHHHHHHHHHHHCCCCCCCCCCCCCCCCCHHHHHHHCCC 100
Prediction 51 HHHHCCCEEEEEECCCCCCCCCCCHHHHHHHHHHHCCCCCCCCCCCCCCCCCHHHHHHHCCC 100

Actual SS 101 SCCCHHHHHHHHHHHHHHTTCEEEEEECCHHHHTTCHHHHHHHHHHHHHH 150
Prediction 101 CCCCHHHHHHHHHHHHHCCCEEEEEECCCCCHHHHHHCCCHHHHHHHHHHHHHH 150

Actual SS 151 TTCSGCCGREEEEEECCGGGTTTSCCCCCHHHHHHHHHHHHHHHTTCHHHHHHHHHHHHHH 200
Prediction 151 HCCCCCHHHHEEEEEECCHHHCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHHHHHHH 200

Actual SS 201 HHSSEEEESSCCCTTTHHHHTTTTCCCEEECESGGGTTTCHHHHHHHHTTTT 247
Prediction 201 HHCCEEECCCCCCCCCHHHHCCCCCCCCCCCCCHHHHHHCCCHHHHHHHHHHCC 247

Weight assigning method 2

Identity scores and their powers ($id^k$) were used as the weight of the local alignment matches. The mispredicted residues are in red.

Substitution matrix used: BLOSUM45

Protein Name: mailnly-Alpha_length_146_3SDH:A
Length= 146 #matches= 79 Prediction Accuracy Q3_id3= 0.945205
Actual SS 1 CCHHHHHHTCTCCHHHHHHHHHHHHHHTTCHHHHHHHHHHHHHCGGGTGGG 50
Prediction 1 CCHHHHHHCCCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHCHHHHHHH 50

Actual SS 51 GGGGTTGGGGGCHHHHHHHHHHHHHHHHHHHHTTCHHHHHHHHHHHHHH 100
Prediction 51 CCCCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH 100
Actual SS 101 HHTTCCCCHHHHTTBHHHHTTBHHHHTTBHHHHTTCCCHHHHHHHHHHHHHHHHHHTT 146
Prediction 101 HHCSCCCCHHHCCCHHHHHHHHHHHHHHHSSCHHHHHHHHHHHHHHHHHHCC 146

Protein Name: mailnly-Alpha_length_116_1DLW:A
Length= 116 #matches= 141 Prediction Accuracy Q3_id3= 0.956897
Actual SS 1 CTTTTTSHHHHHHHTTTHTTTTGGGTTCCCHHHHHHHHHHHHHHHH 50
Prediction 1 CCCCCCCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHCCCCCCHHHHHHHHHHHH 50

Actual SS 51 HHTTCCSSCCSCCHHHHHHTTBSCCHHHHHHHHHHHHHHHHHHHHHHTSCCHHHHH 100
Prediction 51 HHCSCCSCCCCCCCHHHHHCSCCSCCCCHHHHHHHHHHHHHHHHHHHHHHHHCCCHHHHH 100

Actual SS 101 HHHHTTHTHHHHC 116
Prediction 101 HHHHHHHCCCCHHHHHHCC 116

Protein Name: mailnly-Alpha_length_71_1C75:A
Length= 71 #matches= 128 Prediction Accuracy Q3_id3= 0.788732
Actual SS 1 CCHHHHHHHHTHHHHCTTSSCSSCHHHHHHHHHHHHHHHHHCBTTB 50
Prediction 1 CCHHHHHHHHCCHHHCCCCCCCCCCCCCHHHHHHHHHHCCCECCE 50

Actual SS 51 CSCSSCHHHHHHHHHHHHHTCC 71
Prediction 51 CCCCCCCCCCCCCCCCCCCC 71

Protein Name: mailnly-Alpha_length_63_1QOJ:A
Length= 63 #matches= 119 Prediction Accuracy Q3_id3= 0.936508
Actual SS 1 CCCCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHHHHHTTCTHTHHHHH 50
Prediction 1 CCCCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHHHHHCCCHHHHHHHHHHHH 50

Actual SS 51 HHHHHHHHHHHTT 63
Prediction 51 HHHHHHHHHHCC 63

Protein Name: mailnly-Alpha_length_37_1ECI:A
Length= 37 #matches= 116 Prediction Accuracy Q3_id3= 0.513514
Actual SS 1 CCCCHHHHHHHHTHHHHHHHTCCHHHHHHHHHHHHHHHHHHHHH 37
Prediction 1 CEEEECHHHCCCCCCCCCCCCCCCCCHHHHHHHHHHSH 37
Protein Name: all Beta_length_124_1DQI:A
Length= 124 #matches= 162 Prediction Accuracy q3_id3= 0.637097
Actual SS 1 CCGGGEESCBBTTBCCSCEESEEEEEETEEEEEESESSSSCCSSSSSSCBC 50
Prediction 1 CCHHECCCCCECCCCCEEECCECC CEEEEECCCCCCCCCCCCCHC 50
Actual SS 51 EEEEEEEEETCSCCECEEEEEECCCCBSCBTTBTTCCSBCCEEEEEEESCS 100
Prediction 51 HEEEECCCCCCCCCCCCCCCEEEEECCCCCCCCCCCCCCCCCCSCCC 100
Actual SS 101 CCEEEEEEETTTEEEEEEEEEECC 124
Prediction 101 CCEEEEEECCCCCCCCCCECCCCCH 124

Protein Name: all Beta_length_109_1BWW:A
Length= 109 #matches= 16 Prediction Accuracy q3_id3= 1
Actual SS 1 CCCCCCEEECEEEEEETCCEEEEEESEECCCCCCTTCEEEEEEECTTSCCEEE 50
Prediction 1 CCCCCCEEECEEEEEEECCCCCCCCCEEECCCCEEEEEECCCCCEEECE 50
Actual SS 51 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEESCCGGGCEEEEEEEESSSSSCCE 100
Prediction 51 ECCCCcccccccccccccccccccccccccccccccccccccccccccccccc 100
Actual SS 101 CCEEEEEECC 109
Prediction 101 CCEEEEEECC 109

Protein Name: all Beta_length_80_1B4R:A
Length= 80 #matches= 125 Prediction Accuracy q3_id3= 0.8125
Actual SS 1 CEEECCECCCCBBSEEEEEEEESESCSSCSEEEEEEECCSCEEEEEEETEEEE 50
Prediction 1 EEECCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCE 50
Actual SS 51 ECCCCcccccccccccccccccccccccccccccccccccccccccccccccc 80
Prediction 51 ECECCE EHHHEEECCCCEEEEEECCCEEE 80

Protein Name: Alpha-Plus-Beta_length_150_1FMO:E
Length= 150 #matches= 259 Prediction Accuracy q3_id3= 0.873333
Actual SS 1 CCCCCcccccccccccccccccccccccccccccccccccccccccccccccc 50
Prediction 1 CCCCCcccccccccccccccccccccccccccccccccccccccccccccccc 50
Actual SS 51 EEEEEEEEEECCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC 100
Prediction 51 EEEEEEEEEECCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC 100
Protein Name: Alpha-Plus-Beta_length_110_1A2P:A
Length= 110 #matches= 140 Prediction Accuracy Q3_id3= 0.945455
Actual SS 1  CCCCCSCHHHHHHHHHHHHHHHHSSCTTTECHHHHHHHHTTGGTTHHHHSTTCE 50
Prediction 1  CCCCCCHHHHHHHHHHHHHCCCEEEECCCCCCCCCCCCCCCCCCCCCHHHHHHCCE 50
Actual SS 51  ECEECECCCTTCCSCCTTCCCEEECSCSSESSCSCSEEETTCCCEEES 100
Prediction 51  ECEECCCCCCCCCCCCCCCCEEEECCCCCCCCCCCCCCCCCCCCCCCCCCC 100
Actual SS 101  STTSCCCEEEC 110
Prediction 101  CCCCCCCECC 110

Protein Name: Alpha-Plus-Beta_length_68_1HYW:A
Length= 68 #matches= 118 Prediction Accuracy Q3_id3= 0.602941
Actual SS 1  CCCCCHNNNNNNNNNNNNTSSCTTCEEEECEEECECSCEEEETTTHHHHHHHHH 50
Prediction 1  CCCCCHHNNNNNNNNNNNNCCCEEEECCCCCCCCCCCCCCCCCHHHHHH 50
Actual SS 51  HHHHTTTTCCCCCCCCCCC 68
Prediction 51  CCCCCCCCCCCCCCCCCCHHE 68

Protein Name: Alpha-Slash-Beta_length_355_1LUC:A
Length= 355 #matches= 169 Prediction Accuracy Q3_id3= 0.6
Actual SS 1  CEEEEECCCCCCTTCCCHHHHHHHHHHHHHHTTTCSEEEEECCSCBTTB 50
Prediction 1  CEEE EEEECCCCCEHHHHHHHHHHHHHHHHCCCCCCCCCHHHHHHHHH 50
Actual SS 51  CCCCCHHHHHHHHHHHHTCSCSEEEESCEEEECCGGSCHHHHHHHHHHHHHTTSC 100
Prediction 51  HCCCCCCECCHHHHHHHHHCCCEEECCCCCCCCCCCCCCCCCHHHHHHHHH 100
Actual SS 101  CEEEEECCCCCHHHHHHHHTCCGGGHHHHHHHHHHHHHHHHSEECEESSS 150
Prediction 101  ECCCCCCCCCHHHEHHHHCCCCCCHHHHHHHHHHHHHHHHHHHCCCCCCC 150
Actual SS 151  CEECECECSSSCSTTSSEESEECCSSHHHHHHHHHHHTTCEEEECSSCHHH 200
Prediction 151  CCCCCCHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC 200
Actual SS 201  HHHHHHHHHHHHHHHHTCSCGGGCCCCEEEEEEEECSCSSHHHHHHHHHHHHHHHH 250
Prediction 201  CCCCCCECCEECCHHHCCCCHHCHHHCHHHCCCCCCCCCCCCCCCCCCC 250
Protein Name: Alpha-Slash-Beta_length_260_1JCJ:B
Length= 260 #matches= 165 Prediction Accuracy Q3_id3= 0.996154

Actual SS 251 HHHHHHHHCCCCCCCCCTTTGGGGSS 300
Prediction 251 HCCCCCCCCCCCCCHCCCCCCCEEECHHCCCCCCCCCCEECCC 300

Actual SS 301 EESHHHHHHHNNHNNHHCCCCEEEEEGCGCGSHTGGG 350
Prediction 301 CCCCCCEECEEECHHHNNHHCCCCEEEEHHHCCCCCCCCCEEEH 350

Actual SS 351 CCCCC 355
Prediction 351 CCCCC 355

Protein Name: Alpha-Slash-Beta_length_247_1TPH:l
Length= 247 #matches= 190 Prediction Accuracy Q3_id3= 0.931174

Actual SS 1 CCCCHHHHHHHHHHHTEEEEECCTCCCHHHHHHHHHHEETTECSEEHE 50
Prediction 1 CCCCHHHHHHHHHHHCEEEEECCCCCCHHHHHHHHHHEECCECEEEEC 50

Actual SS 51 GGGHHHHHHHHHHHTTTTTTSEESEEESTTTCCHHHHHHHHHHHHHTCS 100
Prediction 51 HHHHHHHHHHHCCEEEEECCCEEEEEECCCCECCCCCCHHHHHHHHHHCCC 100

Actual SS 101 EEEECCCHHHHHTTCCCHHHHHHHHHHHHHHTTCEEEEEECHHHHHHTTHHH 150
Prediction 101 EEEECCCHHHHHCCEEEEECCCEEEEEECCCHHHHHHCCHHHH 150

Actual SS 151 HHHHHHHHHHHHTCSEEECCCSHCCCCCHHHHHHHHHHHHHHTTTTTEEEEEECSHHHHHHHTT 200
Prediction 151 HHHHHHHHHHHHCCEEEEECCCEEEEEECCCCCCCCCHHHHHHHHHHCCCCE 200

Actual SS 201 EEEESSCCSHHHHHHNNHNNNNHHTTTTTTTTTTEEEEEEESCHHHHHHHHTT 250
Prediction 201 EEEESSCCCHHHHHHNNHNNNNHHTTTTTTTTEEEEEEESCHHHHHHHHT 250

Actual SS 251 CCCCCCCCC 260
Prediction 251 CCCCCCCCC 260

Protein Name: Alpha-Slash-Beta_length_247_1TPH:l
Length= 247 #matches= 190 Prediction Accuracy Q3_id3= 0.931174

Actual SS 1 CCCCHHHHHHHHHHHTEEEEECCTCCCHHHHHHHHHHEETTECSEEHE 50
Prediction 1 CCCCHHHHHHHHHHHCEEEEECCCCCCHHHHHHHHHHEECCECEEEEC 50

Actual SS 51 HHHHCCTTSSSEEESCCSSSSSSSCCTTCCHHHHHHHHTCSEEEEEECSHHHHHTT 100
Prediction 51 HHHHCCTTSSSEEESCCSSSSSSSCCTTCCHHHHHHHHTCSEEEEEECSHHHHHHTT 100

Actual SS 101 SCCCCHHHHHHHHHHHHTTCEEEEECOCHHHHHHHTTCTHHHHHHHHHHHHH 150
Prediction 101 SCCCCHHHHHHHHHHHHTTCEEEEECOCHHHHHHHTTCTHHHHHHHHHHHHH 150
Appendix B. Substitution Matrices – PAMs and BLOSUMs

Background

It is always important to define the distances among amino acids in the structure study. The rates of substitution from one amino acid to some other amino acids vary and are not independent of the residues. Amino acids like cysteine and proline are very important for the structure and function of proteins. Amino acids such as tryptophan have bulky side groups and cannot be inserted easily into any site in a peptide. Because of this most amino acid distances use empirical weighting schemes. The most popular of these empirical measures is the PAM family of matrices.

PAMs (Percent Accepted Mutation)

Basics

A unit introduced by Dayhoff et al. to quantify the amount of evolutionary change in a protein sequence, derived from estimation of mutation rates, i.e. how often different amino acids replace other amino acids in evolution. It is a way to assign weights to measure the amino acid differences.

This was based on a database of 1572 changes in 71 groups of closely related proteins.

Construction of mutation probability matrix

This is PAM Markov transition matrix, the table of estimated transition probabilities for the underlying evolutionary model.

1. Align protein sequences that are at least 85 % identical.
2. Reconstruct phylogenetic trees and infer ancestral sequences. Here a most parsimonious tree is constructed. Example:
3. Count the amino acid replacements that occurred along the trees (i.e. count mutations accepted by natural selection). So a 20×20 count matrix is formed. Example:

\[ A = \begin{pmatrix}
A_{1,1} & A_{1,2} & \cdots & A_{1,19} & A_{1,20} \\
A_{2,1} & A_{2,2} & \cdots & A_{2,19} & A_{2,20} \\
& \cdots & \cdots & \cdots & \cdots \\
A_{19,1} & A_{19,2} & \cdots & A_{19,19} & A_{19,20} \\
A_{20,1} & A_{20,2} & \cdots & A_{20,19} & A_{20,20}
\end{pmatrix} \]

4. Use these counts to estimate probabilities for the replacements.

Define \( a_{j,k} = \frac{A_{j,k}}{\sum_{m=1}^{20} A_{j,m}} \). This is the observed relative frequency for the substitution \( j \rightarrow k \).

The \( a_{j,k} \)'s are estimated probabilities.

Now choose a scaling factor \( c \), such that 1% of the amino acids are expected to undergo accepted point mutations during one time unit.

\[ c = \frac{0.01}{\sum_{j=1}^{20} \sum_{k \neq j} q_j \cdot a_{j,k}}. \]

Where the \( q_j \) is the observed frequency of the amino acid no. \( j \) in the original blocks of aligned proteins.

**Construction of PAM substitution matrix (scoring matrix)**
This is the table of scores for all possible pairs of amino acids.

Consider two given protein sequences. Define two hypothesis $H_0$ (two sequences are not evolutionarily related, chance alignment) and $H_A$ (they are evolutionarily related, one depends on the other via the Markov model). Calculate the probabilities under $H_0$ and $H_A$. The alignment score is defined as the likelihood ratio:

$$\text{alignment score} = \frac{P_{H_A}(\text{the alignment})}{P_{H_0}(\text{the alignment})} = \prod_{i=1}^{n} \frac{p_{a_i,b_i}}{q_{b_i}}.$$ 

Or it is defined as log likelihood ratio (the log odds ratio):

$$\text{alignment score} = \log\left(\frac{P_{H_A}(\text{the alignment})}{P_{H_0}(\text{the alignment})}\right) = \sum_{i=1}^{n} \log \left( \frac{p_{a_i,b_i}}{q_{b_i}} \right).$$

The entry $(a,b)$ in the PAM substitution matrix is then of the form

$$S_{a,b} = \log \left( \frac{P_{a,b}}{q_{b}} \right)$$

Above is the construction of 1 PAM substitution matrix.

A PAMn substitution matrix is constructed with entries

$$S_{a,b}^{(n)} = \log \left( \frac{P_{a,b}^{(n)}}{q_{b}} \right).$$

Where the $n$-step transition probabilities $p_{a,b}^{(n)}$ are given as the entries in $P^n$, if $P$ is the 1 PAM transition matrix.

**Observations:**

Residue pairs with scores above 1 replace each other more often as alternatives in related sequences than in random sequences. This is an indication that both residues can carry out similar functions. A score exactly equal to one indicates amino acid pairs that are found as alternatives at exactly the frequency predicted by chance. Residue pairs with scores less than 1 replace each other less often than in random sequences and would be an indication that these residues are not functionally equivalent.
Table 1 - The log odds matrix for 250 PAMs (multiplied by 10)

|   | A | C | D | E | F | G | H | I | K | L | M | N | P | Q | R | S | T | V | W | Y |
| A | 2 | -2 | 0 | -4 | 1 | -1 | -1 | -1 | -2 | -1 | 0 | 1 | 0 | -2 | 1 | 1 | 0 | 1 | 0 | -5 | -3 |
| C | 12 | -5 | -5 | -4 | -3 | -3 | -2 | -5 | -6 | -5 | -4 | -3 | -5 | -4 | 0 | -2 | -2 | -2 | -8 | 0 |
| D | 4 | 3 | -6 | 1 | 1 | 2 | 0 | -4 | -3 | 2 | -1 | 2 | -1 | 0 | 0 | -2 | -7 | -4 |
| E | 4 | -5 | 0 | 1 | -2 | 0 | -3 | -2 | 1 | -1 | 2 | -1 | 0 | 0 | -2 | -7 | -4 |
| F | 9 | -5 | -2 | 1 | -5 | 2 | 0 | -4 | -5 | -5 | -4 | -3 | -3 | -1 | 0 | 7 |
| G | 5 | -2 | -3 | -2 | -4 | -3 | 0 | -1 | -1 | -3 | 1 | 0 | -1 | -7 | -5 |
| H | 6 | -2 | 0 | -2 | 2 | 2 | 0 | 3 | 2 | -1 | -1 | -2 | -3 | 0 |
| I | 5 | -2 | 2 | 2 | 2 | -2 | -2 | -2 | -2 | -1 | 0 | 4 | 5 | 1 |
| J | 5 | -3 | 0 | 1 | 1 | 1 | 0 | 0 | -2 | -3 | 4 |
| K | 6 | 4 | -3 | -3 | -2 | -3 | -3 | -2 | 2 | -2 | -1 |
| L | 6 | -2 | -2 | -1 | 0 | -2 | -1 | 2 | -4 | -2 |
| M | 2 | 1 | 0 | 1 | 0 | -2 | -4 | -2 |
| N | 6 | 0 | 0 | 1 | 0 | -1 | -6 | -5 |
| O | 4 | 1 | -1 | -1 | -2 | -5 | -4 |
| P | 6 | 0 | -1 | -2 | 2 | -4 |
| Q | 2 | 1 | -1 | -2 | -3 |
| R | 3 | 0 | -5 | -3 |
| S | 4 | -6 | -2 |
| T | 17 | 0 |
| U | 10 |

BLOSUMs (Blocks Substitution Matrix)

Definition (NCBI)

Blocks Substitution Matrix. A substitution matrix in which scores for each position are derived from observations of the frequencies of substitutions in blocks of local alignments in related proteins.

Main idea

To get a better measure of differences between two proteins specifically for more distantly related proteins.

Constructing Blocks Data Bases:

Constructed by PROTOMAT from 504 non-redundant groups of proteins catalogued in Prosite 8.0 (Bairoch, 1991) keyed to Swiss-Prot 20 (Bairoch and Boeckmann, 1991).

Alignment and homology searches:

MULTALIN V3.0

BLASTP of BLAST 3/18/91 (Using Smith-Waterman algorithm, a means of searching protein databases to find those with the best alignment using dynamic programming)
Deriving a frequency table from a data base of blocks:

Look at following example of block:

```
*** A ********
*** A ********
*** S ********
*** A ********
*** A ********
*** A ********
*** A ********
*** A ********
*** A ********
*** A ********
```

We now observe the specific column.

The new segment of the block has 8 AA matches and 1 AS mismatch.

After the new member is added to the block, we have:

- **AA pairs:** $8 + 7 + \ldots + 1 = \binom{9}{2} = 36$
- **AS** or **SA** pairs: 9
- **Total pairs:** $36 + 9 = \binom{10}{2} = 45$

Block width $w = 12$

Block depth (number of segment in the block) $s = 10$

Total amino acid pairs $w_s(s-1)/2 = \frac{w}{2} \cdot \binom{s}{2} = 12 \cdot 45 = 540$

There are possibly 210 different amino acid pairs. The frequency table lists the numbers of times each of these pairs occur among the blocks.

It is used to calculate a matrix representing the odds ratio between these observed frequencies and those expected by chance.

**Computing a logarithm of odds (Lod) matrix**

$f_{ij}$: the total number of amino acid $i, j$ pairs for each entry of the frequency table.

For the column of 9 $A$ residues and 1 $S$ residue in the example: $f_{AA} = 36$, $f_{AS} = 9$
1. The observed probability of occurrence for each $i, j$, pair is

$$q_{ij} = \frac{f_{ij}}{\sum_{i=1}^{20} \sum_{j=1}^{20} f_{ij}}$$

(The denominator is the number of total pairs.)

For example: $q_{AA} = 36/45 = 0.8, q_{AS} = 9/45 = 0.2$

2. The probability of occurrence of the $i$th amino acid in an $i, j$ pair

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

For example, 36 pairs have $A$ in both positions of the pair and 9 pairs have $A$ at only one of the two positions. So the expected probability of $A$ in a pair is $[36+9/2]/45=0.9$, and that of $S$ is $(9/2)/45=0.1$.

3. The expected probability of occurrence $e_{ij}$ for each $i, j$, pair

$$e_{ij} = \begin{cases} p_ip_j & i=j \\ p_ip_j + p_jp_i = 2p_ip_j & i \neq j \end{cases}$$

For example

$$e_{AA} = 0.9 \times 0.9 = 0.81$$
$$e_{AS+SA} = 2 \times (0.9 \times 0.1) = 0.18$$
$$e_{SS} = 0.1 \times 0.1 = 0.01$$

4. The odds ratio matrix is calculated where each entry is

$$q_{ij} / e_{ij}$$

5. The lod ratio is calculated

$$s_{ij} = \log_2 \left( \frac{q_{ij}}{e_{ij}} \right)$$

6. The BLOSUM (blocks substitution matrix) entries are obtained by multiplying scaling factor or 2 and rounded to the nearest integers.

$$s_{ij} = 2 \log_2 \left( \frac{q_{ij}}{e_{ij}} \right)$$

Calculate the average mutual information per amino acid pair $H$ (relative entropy) and the expected score $E$ in bit units
\[ H = \sum_{i=1}^{20} \sum_{j=1}^{i} q_{ij} \times s_{ij} \]

\[ E = \sum_{i=1}^{20} \sum_{j=1}^{i} p_{i} \times p_{j} \times s_{ij} \]

\(H\): is 0 when the target (observed) distribution of pair frequencies is the same as the background (expected) distribution.

**Clustering segments within blocks (BLOSUM80, 62, 45 etc)**

To reduce multiple contributions to amino acid pair frequencies from the most closely related members of a family, sequences are clustered within block, and each cluster is weighted as a single sequence in counting pairs. The matrix derived from a database of blocks in which sequence segment that are identical at ≥80% of aligned residues are clustered is referred to as BLOSUM 80.

**Observations**

1. If the observed number of differences between a pair of amino acids is equal to the expected number then \( S_y = 0 \); if the observed is less than expected then \( S_y < 0 \); and if the observed is greater than expected, then \( S_y > 0 \).

2. A consequence of clustering is that the contribution of closely related segments to the frequency table is reduced (or eliminated when an entire block is clustered, since this is equivalent to a single sequence in which no substitutions appear) (Henikoff & Henikoff, 1992).

3. If the BLOSUM62 matrix is compared to PAM160 (it is closest equivalent) then it is found that the BLOSUM matrix is less tolerant of substitutions to or from hydrophilic amino acids, while more tolerant of hydrophobic changes and of cysteine and tryptophan mismatches.

**An example BLOSUM62**


Table 2 - The log odds matrix for BLOSUM 62

|   | A | C | D | E | F | G | H | I | K | L | M | N | P | Q | R | S | T | V | W | Y |
| A | 4 | 0 | -2 | -1 | -2 | 0 | -2 | -1 | -1 | -1 | -2 | -2 | -1 | 1 | 0 | 0 | -3 | -2 |
| C | 9 | -3 | -4 | -2 | -3 | -3 | -1 | -3 | -1 | -3 | -3 | -3 | -1 | -1 | -1 | -1 | -2 | -2 |
| D | 6 | 2 | -3 | -1 | -1 | -3 | -1 | -4 | -3 | 1 | -1 | 0 | -2 | 0 | -1 | -3 | -4 | -3 |
| E | 5 | 3 | -2 | 0 | -3 | 1 | -3 | -2 | 0 | -1 | 2 | 0 | 0 | -1 | -2 | -3 | -2 |
| F | 6 | 3 | -1 | 0 | -3 | 0 | 0 | -3 | -4 | -3 | -3 | -2 | -2 | -1 | 1 | 3 |
| G | 6 | -2 | -4 | -2 | -4 | -3 | 0 | -2 | -2 | -2 | 0 | 0 | -1 | -2 | -3 | -2 | -3 |
| H | 8 | -3 | -1 | -3 | -2 | 1 | -2 | 0 | 0 | -1 | -2 | -3 | -2 | -2 |
| I | 4 | -3 | 2 | 1 | -3 | -3 | -3 | -2 | -1 | 1 | 3 | 3 | -1 |
| J | 5 | 2 | -1 | 0 | -1 | 1 | 2 | 0 | -1 | -2 | -3 | -2 |
| K | 4 | 2 | -3 | -1 | -3 | -2 | -2 | -2 | -1 | 1 | 2 | 1 |
| L | 5 | 2 | -2 | 0 | -1 | -1 | 1 | 1 | 1 | -1 |
| M | 6 | -2 | 0 | 0 | 1 | 0 | -3 | -4 | -2 |
| N | 5 | 1 | 0 | 0 | -1 | -2 | -2 | -1 |
| O | 5 | -1 | -1 | -3 | -3 | -2 |
| P | 4 | 1 | -2 | -3 | -2 |
| Q | 5 | 0 | -2 | -2 |
| R | 4 | -3 | -1 |
| S | 11 | 2 |
| T | Y | 7 |

Appendix C. Artificial Intelligence Approaches

Basic ideas

Two training sources

Amino acid sequence training. We randomly cut these files into training and test sequence files and training and test secondary structure files. Define different window sizes from 3 to 13, scan the sequence training file of some window sizes, move forward one residue at a time, list the secondary structure letter of the middle residue of the window as the target, and the training file is ready.

Normalized score-based training. Previously, for a specific position of some query sequence, we have three possibilities for the residue to be some secondary structure element (prob(H), prob(E), and prob(C)). We always choose the highest normalized score among these three to determine the prediction for that residue. Now, before we make this decision, we form a file that contains all the normalized scores for all the query sequences from the
benchmark dataset, randomly cut these normalized scores into training and test sets, and apply AI approaches to finish the prediction.

**Basic principles**

**The Naive Bayes Classifier.** This method is applicable to the task of concept learning. It reads a set of examples in attribute-value representation and uses Bayes theorem to estimate the posterior probabilities of all classifications. For each instance of the example language a classification with highest posterior normalized score is chosen as the prediction. For the Naive Bayes method the following assumption is essential: Let $x = <x_1, ..., x_n>$ be an instance of the example language and $c \in C$ a possible classification. Then

$$\text{Prob}(x | c) = \prod_{i \in \{1, ..., n\}} \text{Prob}(x_i | c)$$

The Bayes Rule is used here. If we have a hypothesis $h$ (our target class) and a data $D$ which bears on the hypothesis, then

$$p(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)}$$

where $P(h)$ is the independent probability of $h$ (prior probability); $P(D)$ is the independent probability of $D$; $P(D|h)$ is the conditional probability of $D$ given $h$ (likelihood); $P(h|D)$ is the conditional probability of $h$ given $D$ (posterior probability, our goal).

Here we would take the normalized score training and test as an example to illustrate the process of deciding the predicted secondary structure. The key point is it is supposed that all attributes are equally important to the target result. All their contributions of the attributes are independent. So it is called naïve Bayes model. The detailed process is as following (if the window size is 5, the attributes are L2, L1, M, R1, R2): 1) Turn the linear values to discrete ones. Here we may call them tiny, small, medium, large and giant. 2) For each attribute, calculate the frequencies and the corresponding conditional probabilities of the data. For example, we can easily get the frequency for L2 to be E or H or C when the value is tiny or small and so on. The total frequencies of E and H and C ($P(E)$, $P(H)$, $P(C)$) are also calculated. Accordingly, the corresponding conditional probs are calculated (for example $P(\text{tiny}|E)$ at position L2). 3) Calculating the final probs ($P(D|h)$) of given example (to be predicted). Take into account the probability of each attribute (attribute probs). Treat all attributes equally important. Multiply all their probabilities found in training data. 4) The
posterior probs of E, H and C are obtained by multiplying attribute probs with overall probs from the training frequency table. 5) Choose the class (H, E or C) so that it maximizes this posterior probability. The class is the ss prediction for the central residue. The heading and tailing residues (cannot be the center of some window size) are assigned C as their ss. There is a concrete example to show the procedure provided by F. Keller. (F Keller)

**Decision Tree Induction.** Decision tree induction is another machine learning method to predict the protein secondary structures. ID3 (Quinlan, 1986) is the most common algorithm, which grows tree by the information gain. If there is any noise in the training dataset or it does not completely cover the decision space, then overfitting occurs and the decision tree needs to be pruned in order to generalize well to the test set.

For AA sequence based method, input into the decision tree is a small window of amino acids. Each position is one attribute with 20 values (20 different amino acids). The goal of the decision tree is to predict the secondary structure of the amino acid in the middle position of the input window, which is the target attribute with 3 values.

For the normalized score based method, input into the decision tree is normalized scores of a small window of the secondary sequences. Each position has three attributes (H, E, C), and each attribute is a continuous normalized score value (between 0 and 1). The goal of the decision tree is to predict the secondary structure in the middle position of the input window, which is the target attribute with 3 values.

**Neural Networks.** Neural networks are first used to predict protein secondary structure by Qian and Sejnowski in around 1998. Amino acids are encoded in what is known as an orthogonal encoding. Each amino acid is assigned a binary vector with 20 variables, 1 is set to the position to which the current amino acid is corresponding, and the rest are set to 0. Thus the encoding of an amino acid of 20 different amino acids requires the 20-dimensional space. For example, 260 input units are required for a window size 13. The output has three units corresponding to H, E, and C. They are encoded in 100, 010 and 001, respectively. In our paper, the AA sequence based approach used the above encoding method. For the normalized score-based approach, each normalized score of a secondary structure symbol is encoded as following:

```
0000  for p<0.02
```
In this case, each amino acid is represented in 12 dimensional space. For example, we need 84 input units for a window size 7. Three output units (001, 010, 100 for H, E and C) are similarly needed.

Single-layer networks can learn only linear decision boundaries in the input space. Multilayer networks can represent some general nonlinear functions. In the paper, we also attempt to use 2-layer neural networks with 4, 10 and 40 hidden units.

**Support Vector Machine (SVM).** SVM is a type of learning machine developed by Vapnik and his co-workers (Cortes & Vapnik, 1995). Like other learning machines, such as multi-layer perceptrons and radial basis function networks, SVM is capable of classification and non-linear regression. Intuitively, SVM learns the boundary between samples belonging to classes by mapping the input samples into a high dimensional space, and seeking a separating hyperplane in this space (see Figure C-1). The hyperplane is called the optimal separating hyperplane (OSH). To train the SVM, we need to select a kernel function to implement the mapping from input data to the high dimensional space.

In our application, the radial basis function (RBF) is adopted:

\[ K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \| ^2) , \gamma > 0 \]

where the parameter \( \gamma \) is user specified. Another important regularization parameter \( C \) must be chosen, which controls the trade-off between model complexity and misclassified training examples. Given the above two parameters, by calculating the value of a decision function, the new vectors are classified. This decision function reflects the optimally separating hyperplane.
Figure C-1. A separating hyperplane in the feature space corresponding to a non-linear boundary in the input space. (Hua & Sun 2001)

Two classes denoted by circles and disks, respectively, are linear non-separable in the input space \( \mathbb{R}^d \). SVM constructs the optimal separating hyperplane (OSH) (continuous line) which maximizes the margin between two classes by mapping the input space \( \mathbb{R}^d \) into a high dimensional space, the feature space \( \mathcal{H} \). The mapping is determined by a kernel function \( K(x_i, x_j) \). Support vectors are identified with an extra circle.

Input into SVMs is a small window of amino acids. For the AA sequence based approach, the same encoding method as the neural networks is used for SVMs. For the normalized score based method approach, input into SVMs is the same as the input into the decision tree since SVMs can take linear data. For both approaches, the output is simply labeled to 1, 2, and 3 which are corresponding to H, E, and C.

**Running programs**

**Data Processing**

The programs are developed to randomly partition the entries in sequence and normalized score files into training and testing sets at ratio 4:1.

**Prediction and evaluation**

The Naive Bayes, decision tree, neural network programs are developed to learning the training sets, to predict the test sets, and to evaluate the prediction accuracies. The SVM program is downloaded at “A library for Support Vector Machines” supported by Chih-Chung Chang and Chih-Jen Lin (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) and is modified to accommodate the format of our input files.
Appendix D. Datasets CB513 and KS267

CB513

RS126

Rost and Sander selected 126 proteins with which to train and test secondary structure prediction algorithms. They defined the non-redundancy to mean that no two proteins in the set share more than 25% sequence identity over a length of more than 80 residues. (Cuff & Barton, 1999) It has long been known that the percentage identity is a poor measure of sequence similarity. Inevitably, the RS126 set contains similar sequences, if the sequences are compared using more complicated and advanced algorithms.

Cuff and Barton datasets and CB513

Comparison algorithm and SD or Z score. To measure the similarity between two protein sequences A and B, the standard dynamic programming algorithm (Needleman and Wunsch, 1970) is applied to align them. The alignment score is obtained, say \( V \). Then the order of amino acids of the two sequences is randomized. The alignment is done again. The alignment is repeated several times (e.g., 100 times). The alignment scores are recorded. The mean \( \bar{x} \) and standard deviation \( \sigma \) of the scores are calculated. The SD score or Z score is obtained by \( \frac{(V - \bar{x})}{\sigma} \).

CB554. The sequences come from 3Dee (http://www.compbio.dundee.ac.uk/3Dee/) database of structural domain definitions. In 3Dee, a non-redundant sequences set id created by the use of a sensitive sequence comparison algorithm and cluster analysis. This way a set of 1233 domains with no obvious pairwise sequence similarity are obtained. By removing multi-segment domains, the number turns to be 988. This set is further filtered only to permit X-ray crystal structures with resolution of 2.5Å or less. Finally, only 554 domains serve as CB554.

CB396. The CB554 and RS126 are considered together. Using the comparison algorithm of AMPS (Barton, 1987), 11 and 21 sequence-similar pairs are found within RS126 and CB554, respectively, and 119 pairs are found to be similar between these two sets. Eliminating the 3 overlapped matching sequences of 119 and 21 pairs, there are 137 sequences in CB554 that either match a sequence in CB554 or in RS126, and thus are
removed from CB554. Further removing 21 sequences that do not have “full DSSP definitions” (i.e., those with more than 9 consecutive residues with incomplete backbones for which DSSP does not define a state), the CB396 is obtained.

CB513. As mentioned above, there are 11 sequence-similar pairs within RS126. One of each of the 11 pairs that have an SD score of ≥5 are removed from the RS126 set. Since two of them match more than one protein in this subset, this left 9 unique homologues that are actually removed from RS126. This set added to CB396 gives CB513.

KS267

This dataset is used by GOR IV (Garnier, et al. 1996) method. It was prepared by J. M. Levin and checked for homologous sequences with the help of V. Di Francesco. This database has been modified to restore the total length of the sequences as defined in the SEQRES field of the PDB file.

Appendix E. An example of .rsa file (1C75.rsa)

REM Relative accessibilites read from external file "standard.data"
REM File of summed (Sum) and % (per.) accessibilities for
REM RES_NUM All-atoms Total-Side Main-Chain Non-polar
REM ABS REL ABS REL ABS REL ABS REL
RES VAL A 22 109.91 72.6 39.77 34.8 70.14 188.8 42.95 37.2
RES ASP A 23 94.14 67.1 86.63 84.4 7.51 19.9 41.52 84.3
RES ALA A 24 12.77 11.8 8.50 12.2 4.27 11.1 8.50 11.9
RES GLU A 25 97.66 56.7 95.94 71.2 1.72 4.6 25.21 41.8
RES ALA A 26 40.23 37.3 39.06 56.3 1.17 3.0 39.14 54.8
RES VAL A 27 14.53 9.6 14.53 12.7 0.00 0.0 14.53 12.6
RES VAL A 28 0.59 0.4 0.59 0.5 0.00 0.0 0.59 0.5
RES GLN A 29 76.39 42.8 68.04 48.3 8.34 22.2 38.28 73.3
RES GLN A 30 149.61 83.8 118.23 83.9 31.38 83.7 39.08 74.8
RES LYS A 31 73.40 36.6 61.20 37.5 12.20 32.5 46.06 39.5
RES CYS A 32 25.43 18.9 25.03 25.9 0.40 1.1 25.31 25.8
RES ILE A 33 53.86 30.8 47.86 34.7 5.99 16.1 47.91 34.4
RES SER A 34 95.82 82.3 73.96 94.7 1.72 4.6 25.21 41.8
RES CYS A 35 61.30 45.6 61.30 63.4 0.00 0.0 61.30 62.6
RES HIS A 36 23.03 12.6 23.03 15.7 0.00 0.0 16.69 17.2
RES GLY A 37 8.56 10.7 8.56 26.5 0.00 0.0 8.56 22.8
RES GLY A 38 48.52 60.6 21.71 67.1 26.81 56.1 26.24 69.9
RES ASP A 39 98.46 70.1 86.62 84.4 11.83 31.4 51.82 105.2
RES LEU A 40 8.21 4.6 8.21 5.8 0.00 0.0 8.21 5.8
RES THR A 41 77.15 55.4 56.43 55.5 20.71 55.1 40.01 52.8
<table>
<thead>
<tr>
<th>Residue</th>
<th>Amino Acid</th>
<th>Residue</th>
<th>Amino Acid</th>
<th>Residue</th>
<th>Amino Acid</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLY A 43</td>
<td>65.79</td>
<td>GLY A 43</td>
<td>65.79</td>
<td>GLY A 43</td>
<td>65.79</td>
</tr>
<tr>
<td>GLY A 44</td>
<td>102.69</td>
<td>GLY A 44</td>
<td>102.69</td>
<td>GLY A 44</td>
<td>102.69</td>
</tr>
<tr>
<td>GLY A 45</td>
<td>25.25</td>
<td>GLY A 45</td>
<td>25.25</td>
<td>GLY A 45</td>
<td>25.25</td>
</tr>
<tr>
<td>GLY A 46</td>
<td>104.64</td>
<td>GLY A 46</td>
<td>104.64</td>
<td>GLY A 46</td>
<td>104.64</td>
</tr>
<tr>
<td>LYS A 47</td>
<td>56.00</td>
<td>LYS A 47</td>
<td>56.00</td>
<td>LYS A 47</td>
<td>56.00</td>
</tr>
<tr>
<td>LYS A 48</td>
<td>29.37</td>
<td>LYS A 48</td>
<td>29.37</td>
<td>LYS A 48</td>
<td>29.37</td>
</tr>
<tr>
<td>LYS A 49</td>
<td>39.62</td>
<td>LYS A 49</td>
<td>39.62</td>
<td>LYS A 49</td>
<td>39.62</td>
</tr>
<tr>
<td>LYS A 50</td>
<td>105.90</td>
<td>LYS A 50</td>
<td>105.90</td>
<td>LYS A 50</td>
<td>105.90</td>
</tr>
<tr>
<td>LYS A 51</td>
<td>9.86</td>
<td>LYS A 51</td>
<td>9.86</td>
<td>LYS A 51</td>
<td>9.86</td>
</tr>
<tr>
<td>LYS A 52</td>
<td>10.81</td>
<td>LYS A 52</td>
<td>10.81</td>
<td>LYS A 52</td>
<td>10.81</td>
</tr>
<tr>
<td>LYS A 53</td>
<td>58.87</td>
<td>LYS A 53</td>
<td>58.87</td>
<td>LYS A 53</td>
<td>58.87</td>
</tr>
<tr>
<td>LYS A 54</td>
<td>108.86</td>
<td>LYS A 54</td>
<td>108.86</td>
<td>LYS A 54</td>
<td>108.86</td>
</tr>
<tr>
<td>LYS A 55</td>
<td>87.37</td>
<td>LYS A 55</td>
<td>87.37</td>
<td>LYS A 55</td>
<td>87.37</td>
</tr>
<tr>
<td>LYS A 56</td>
<td>57.75</td>
<td>LYS A 56</td>
<td>57.75</td>
<td>LYS A 56</td>
<td>57.75</td>
</tr>
<tr>
<td>LYS A 57</td>
<td>84.85</td>
<td>LYS A 57</td>
<td>84.85</td>
<td>LYS A 57</td>
<td>84.85</td>
</tr>
<tr>
<td>LYS A 58</td>
<td>114.20</td>
<td>LYS A 58</td>
<td>114.20</td>
<td>LYS A 58</td>
<td>114.20</td>
</tr>
<tr>
<td>LYS A 59</td>
<td>83.18</td>
<td>LYS A 59</td>
<td>83.18</td>
<td>LYS A 59</td>
<td>83.18</td>
</tr>
<tr>
<td>LYS A 60</td>
<td>7.73</td>
<td>LYS A 60</td>
<td>7.73</td>
<td>LYS A 60</td>
<td>7.73</td>
</tr>
<tr>
<td>LYS A 61</td>
<td>26.67</td>
<td>LYS A 61</td>
<td>26.67</td>
<td>LYS A 61</td>
<td>26.67</td>
</tr>
<tr>
<td>LYS A 62</td>
<td>64.81</td>
<td>LYS A 62</td>
<td>64.81</td>
<td>LYS A 62</td>
<td>64.81</td>
</tr>
<tr>
<td>LYS A 63</td>
<td>27.37</td>
<td>LYS A 63</td>
<td>27.37</td>
<td>LYS A 63</td>
<td>27.37</td>
</tr>
<tr>
<td>LYS A 64</td>
<td>15.49</td>
<td>LYS A 64</td>
<td>15.49</td>
<td>LYS A 64</td>
<td>15.49</td>
</tr>
<tr>
<td>LYS A 65</td>
<td>63.07</td>
<td>LYS A 65</td>
<td>63.07</td>
<td>LYS A 65</td>
<td>63.07</td>
</tr>
<tr>
<td>LYS A 66</td>
<td>99.16</td>
<td>LYS A 66</td>
<td>99.16</td>
<td>LYS A 66</td>
<td>99.16</td>
</tr>
<tr>
<td>LYS A 67</td>
<td>13.28</td>
<td>LYS A 67</td>
<td>13.28</td>
<td>LYS A 67</td>
<td>13.28</td>
</tr>
<tr>
<td>LYS A 68</td>
<td>101.80</td>
<td>LYS A 68</td>
<td>101.80</td>
<td>LYS A 68</td>
<td>101.80</td>
</tr>
<tr>
<td>LYS A 69</td>
<td>82.99</td>
<td>LYS A 69</td>
<td>82.99</td>
<td>LYS A 69</td>
<td>82.99</td>
</tr>
<tr>
<td>LYS A 70</td>
<td>86.33</td>
<td>LYS A 70</td>
<td>86.33</td>
<td>LYS A 70</td>
<td>86.33</td>
</tr>
<tr>
<td>LYS A 71</td>
<td>86.33</td>
<td>LYS A 71</td>
<td>86.33</td>
<td>LYS A 71</td>
<td>86.33</td>
</tr>
<tr>
<td>LYS A 72</td>
<td>82.24</td>
<td>LYS A 72</td>
<td>82.24</td>
<td>LYS A 72</td>
<td>82.24</td>
</tr>
<tr>
<td>LYS A 73</td>
<td>37.67</td>
<td>LYS A 73</td>
<td>37.67</td>
<td>LYS A 73</td>
<td>37.67</td>
</tr>
<tr>
<td>LYS A 74</td>
<td>37.91</td>
<td>LYS A 74</td>
<td>37.91</td>
<td>LYS A 74</td>
<td>37.91</td>
</tr>
<tr>
<td>LYS A 75</td>
<td>64.92</td>
<td>LYS A 75</td>
<td>64.92</td>
<td>LYS A 75</td>
<td>64.92</td>
</tr>
<tr>
<td>LYS A 76</td>
<td>5.99</td>
<td>LYS A 76</td>
<td>5.99</td>
<td>LYS A 76</td>
<td>5.99</td>
</tr>
<tr>
<td>LYS A 77</td>
<td>142.98</td>
<td>LYS A 77</td>
<td>142.98</td>
<td>LYS A 77</td>
<td>142.98</td>
</tr>
<tr>
<td>LYS A 78</td>
<td>42.79</td>
<td>LYS A 78</td>
<td>42.79</td>
<td>LYS A 78</td>
<td>42.79</td>
</tr>
<tr>
<td>LYS A 79</td>
<td>71.95</td>
<td>LYS A 79</td>
<td>71.95</td>
<td>LYS A 79</td>
<td>71.95</td>
</tr>
<tr>
<td>LYS A 80</td>
<td>66.19</td>
<td>LYS A 80</td>
<td>66.19</td>
<td>LYS A 80</td>
<td>66.19</td>
</tr>
<tr>
<td>LYS A 81</td>
<td>0.00</td>
<td>LYS A 81</td>
<td>0.00</td>
<td>LYS A 81</td>
<td>0.00</td>
</tr>
<tr>
<td>LYS A 82</td>
<td>93.45</td>
<td>LYS A 82</td>
<td>93.45</td>
<td>LYS A 82</td>
<td>93.45</td>
</tr>
<tr>
<td>LYS A 83</td>
<td>27.24</td>
<td>LYS A 83</td>
<td>27.24</td>
<td>LYS A 83</td>
<td>27.24</td>
</tr>
<tr>
<td>LYS A 84</td>
<td>2.40</td>
<td>LYS A 84</td>
<td>2.40</td>
<td>LYS A 84</td>
<td>2.40</td>
</tr>
<tr>
<td>LYS A 85</td>
<td>0.62</td>
<td>LYS A 85</td>
<td>0.62</td>
<td>LYS A 85</td>
<td>0.62</td>
</tr>
<tr>
<td>LYS A 86</td>
<td>41.15</td>
<td>LYS A 86</td>
<td>41.15</td>
<td>LYS A 86</td>
<td>41.15</td>
</tr>
<tr>
<td>LYS A 87</td>
<td>76.68</td>
<td>LYS A 87</td>
<td>76.68</td>
<td>LYS A 87</td>
<td>76.68</td>
</tr>
<tr>
<td>LYS A 88</td>
<td>7.68</td>
<td>LYS A 88</td>
<td>7.68</td>
<td>LYS A 88</td>
<td>7.68</td>
</tr>
<tr>
<td>LYS A 89</td>
<td>10.50</td>
<td>LYS A 89</td>
<td>10.50</td>
<td>LYS A 89</td>
<td>10.50</td>
</tr>
<tr>
<td>LYS A 90</td>
<td>130.10</td>
<td>LYS A 90</td>
<td>130.10</td>
<td>LYS A 90</td>
<td>130.10</td>
</tr>
<tr>
<td>LYS A 91</td>
<td>64.90</td>
<td>LYS A 91</td>
<td>64.90</td>
<td>LYS A 91</td>
<td>64.90</td>
</tr>
</tbody>
</table>

| 74 |
Note that the blue column is the percentage values we used to determine the residue status (whether buried, intermediate or exposed).

Appendix F. An example of prediction (1C75_A)

Do the local alignment against PDB

The matching file of BLAST against sequence 1DLW_A is shown as following with many matches skipped:

BLASTP 2.2.8 [Jan-05-2004]


Query= mailnly-Alpha_length_71_1C75:A CYTOCHROME C-553
(71 letters)

<table>
<thead>
<tr>
<th>Score</th>
<th>E Value</th>
<th>Cache Code</th>
<th>Chain</th>
<th>Description</th>
<th>Length</th>
<th>E-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>162</td>
<td>8e-041</td>
<td>pdb</td>
<td>1B7V</td>
<td>A Chain A, Structure Of The C-553 Cytochrome From Bacil...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>159</td>
<td>6e-040</td>
<td>pdb</td>
<td>1K3G</td>
<td>A Chain A, Nmr Solution Structure Of Oxidized Cytochrom...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>487D</td>
<td>H Chain H, Seven Ribosomal Proteins Fitted To A Cryo-EI...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1AD2</td>
<td>Ribosomal Protein L1 Mutant With Serine 179 Replaced ...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1EG0</td>
<td>N Chain N, Fitting Of Components With Known Structure I...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1GIY</td>
<td>C Chain C, Crystal Structure Of The Ribosome At 5.5 A Re...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1DM4</td>
<td>A Chain A, Crystal Structure Of The Succinate Complexed...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1DC4</td>
<td>A Chain A, Crystal Structure Of The Uncomplexed Form Of...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1BDF</td>
<td>A Chain A, Structure Of Escherichia Coli Rna Polymerase...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>pdb</td>
<td>1E58</td>
<td>A Chain A, E.Coli Cofactor-Dependent Phosphoglycerate M...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>24</td>
<td>43</td>
<td>pdb</td>
<td>1JDL</td>
<td>A Chain A, Structure Of Cytochrome C2 From Rhodospiril...</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>23</td>
<td>58</td>
<td>pdb</td>
<td>1G4A</td>
<td>E Chain E, Crystal Structures Of The Hslvu Peptidase-A...</td>
<td>25</td>
<td>13</td>
</tr>
</tbody>
</table>
Cytochrome C553 (Reduced) (Nmr, 36 Structures) 23 58
Chain A, Crystal Structure Of Mj0109 Gene Product Ino... 23 58
Chain A, Trigonal Crystal Form Of Heat Shock Locus U ... 23 58
Chain E, Structural Model Of The [fe]-HydrogenaseCYTO... 23 58
Chain A, 2.0 A Structure Of Hypothetical Protein From... 23 58
Chain B, The Ferredoxin-Cytochrome Complex Using Hete... 23 58
The Y64a Mutant Of Cytochrome C553 From Desulfovibrio... 23 58
Chain E, Hslv-Hslu From E.Coli >gi|11513636|pdb|1E94|... 23 58
Chain A, Acetyl-Coa Carboxylase Carboxyltransferase D... 23 77
Chain A, Reca Hexamer Model, Electron Microscopy >gi|... 22 104
Structure Of The Reca Protein-Adp Complex >gi|443432|... 22 104
Chain A, Crystal Structure Of The Methanosarcina Bark... 22 104
Chain A, Mechanism Of Processivity Clamp Opening By T... 22 139
Chain O, Glycolysis, Oxidoreductase, Nad Mol_id: 1; M... 22 139
Chain A, Oxydized Nitrite Reductase From Pseudomonas ... 18 1464
Diphtheria Toxin (Dimeric) >gi|576189|pdb|1MDT|A Chai... 18 1464
Elm, A104w Mutant Of Rh. Blastaica Porin 18 1464
Elm, K50a, R52a, D97a, E99a Mutant Of Rh. Blastaica Porin 18 1464
Chain A, Hexadecaheame High Molecular Weight Cytochrom... 18 1964
Chain A, Crystal Structure Of The R158q Mutant Of Mal... 18 1964
Chain A, D34 Region Of Escherichia Human Ankyrin-R And Linker 18 1964
Chain A, Crystal Structure Of The S155c Mutant Of Mal... 18 1964
Chain A, Sulfate Respiration In Desulfovibrio Vulgari... 18 1964
Chain A, The Crystal Analysis Of Beta-Keroacyl-[acyl ... 18 1964
Chain A, Structure Of The Hypothetical Protein Tm0936... 18 1964
Chain A, Nmr Solution Structure Of Reduced E. Coli Gl... 18 2635
Chain A, Roles Of His291-Alph And His146-Beta' In Th... 18 2635
Chain A, Crystal Structure Of Flt3 18 2635
Chain A, Human Branched-Chain Alpha-Keto Acid Dehydro... 18 2635
Chain A, Rb12p, Beta-Tubulin Binding Post-Chaperonin ... 18 2635
Chain A, Solution Structure Of The Cysteine-Rich Doma... 17 3536
Chain A, Betaine Aldehyde Dehydrogenase From Cod Live... 17 3536
Chain A, N-Terminal Fragment Of Sigr From Streptomyce... 17 3536
Chain A, Asymmetric Complex Between Hslv And I-Domain... 17 3536
Chain A, The Structure Of Bacillus Subtilis Rbsd Comp... 17 3536
Chain A, Structure Of E. Coli Threonyl-Trna Synthetase... 17 4745
Chain A, L-Phenylalanine Dehydrogenase Structure In T... 16 6366
Chain A, Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain A, Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain B, L-Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain B, L-Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain B, L-Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain A, L-Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain B, Phenylalanine Dehydrogenase Structure In Terr... 16 6366
Chain A, Crystal Structure Of The Hsluv Protease-Chap... 16 6366
Chain A, Crystal Structure Of Escherichia Coli Mobb. ... 16 6366
Chain A, The Structure Of The Human Retinoid-X-Recept... 16 6366
pdb|1IM2| A Chain A, Hslu, Haemophilus Influenzae, Selenomethionine ...
pdb|1L5J| A Chain A, Crystal Structure Of E. Coli Aconitase B. gi|
pdb|1NP6| A Chain A, Crystal Structure Of Escherichia Coli Mobb >
pdb|1JQ0| A Chain A, Crystal Structure Of C4-Form Phosphoenolpyruvate...
pdb|1KOE| Endostatin gi|
pdb|1DY0| A Chain A, Murine Endostatin, Crystal Form II gi|
pdb|1C25| Human Cdc25a Catalytic Domain gi|
pdb|1NW1| A Chain A, Crystal Structure Of Choline Kinase gi|
pdb|1PS9| A Chain A, The Crystal Structure And Reaction Mechanism ...

> pdb|1B7V| A Chain A, Structure Of The C-553 Cytochrome From Bacillus Pasteurii To 1.7 A Resolution
pdb|1C75| A Chain A, 0.97 A "ab Initio" Crystal Structure Of Cytochrome C-553 From Bacillus Pasteurii
pdb|1K3H| A Chain A, Nmr Solution Structure Of Oxidized Cytochrome C-553 From Bacillus Pasteurii
pdb|1N9C| A Chain A, Structure And Dynamics Of Reduced Bacillus Pasteurii Cytochrome C: Oxidation State Dependent Properties And Implications For Electron Transfer Processes
Length = 71

Score = 162 bits (375), Expect = 8e-041
Identities = 51/51 (100%), Positives = 51/51 (100%)

Query: 1 VDAEAVVQQHCISCHGGDLTGASAPAKAGANYSEEILDIIILNGQGGMP 51
Sbjct: 1 VDAEAVVQQHCISCHGGDLTGASAPAKAGANYSEEILDIIILNGQGGMP

> pdb|1K3G| A Chain A, Nmr Solution Structure Of Oxidized Cytochrome C-553 From Bacillus Pasteurii
Length = 71

Score = 159 bits (368), Expect = 6e-040
Identities = 50/50 (100%), Positives = 50/50 (100%)

Query: 2 DAEAVVQKCISCHGGDLTGASAPAKAGANYSEEILDIIILNGQGGMP 51
Sbjct: 2 DAEAVVQKCISCHGGDLTGASAPAKAGANYSEEILDIIILNGQGGMP

> pdb|487D| H Chain H, Seven Ribosomal Proteins Fitted To A Cryo-Electron Microscopic Map Of The Large 50s Subunit At 7.5 Angstroms Resolution
Length = 224

Score = 25.2 bits (52), Expect = 13
Identities = 10/18 (55%), Positives = 13/18 (72%), Gaps = 1/18 (5%)
Query: 30 AGANY-SEEILDIIILNG 46
AGA+Y EEI+ IL+G
Sbjct: 85 AGADYVGEIIQKILDG 102

>pdb|10BI|A Chain A, Crystal Structure Of The G130a Mutant Of Malonamidase E2 From Bradyrhizobium japonicum
pdb|10BI|B Chain B, Crystal Structure Of The G130a Mutant Of Malonamidase E2 From Bradyrhizobium japonicum
Length = 414
Score = 18.0 bits (35), Expect = 1964
Identities = 6/8 (75%), Positives = 6/8 (75%)

Query: 21 GASAPAID 28
GAS AID
Sbjct: 254 GASVQAID 261

>pdb|1090|A Chain A, Crystal Structure Of The S131a Mutant Of Malonamidase E2 Complexed With Malonamate From Bradyrhizobium japonicum
pdb|1090|B Chain B, Crystal Structure Of The S131a Mutant Of Malonamidase E2 Complexed With Malonamate From Bradyrhizobium japonicum
pdb|109P|A Chain A, Crystal Structure Of The S131a Mutant Of Malonamidase E2 Complexed With Malonate From Bradyrhizobium japonicum
pdb|109P|B Chain B, Crystal Structure Of The S131a Mutant Of Malonamidase E2 Complexed With Malonate From Bradyrhizobium japonicum
Length = 414
Score = 18.0 bits (35), Expect = 1964
Identities = 6/8 (75%), Positives = 6/8 (75%)

Query: 21 GASAPAID 28
GAS AID
Sbjct: 254 GASVQAID 261

>pdb|1N11|A Chain A, D34 Region Of Human Ankyrin-R And Linker
Length = 437
Score = 18.0 bits (35), Expect = 1964
Identities = 4/6 (66%), Positives = 6/6 (100%)

Query: 37 EEILDI 42
+EILD+
Sbjct: 421 DEILDV 426
>pdb|109Q|A Chain A, Crystal Structure Of The S155c Mutant Of Malonamidase E2 From Bradyrhizobium Japonicum
pdb|109Q|B Chain B, Crystal Structure Of The S155c Mutant Of Malonamidase E2 From Bradyrhizobium Japonicum
pdb|10CH|A Chain A, Crystal Structure Of The S155c Mutant Of Malonamidase E2 From Bradyrhizobium Japonicum
pdb|10CH|B Chain B, Crystal Structure Of The S155c Mutant Of Malonamidase E2 From Bradyrhizobium Japonicum
Length = 414
Score = 18.0 bits (35), Expect = 1964
Identities = 6/8 (75%), Positives = 6/8 (75%)
Query: 21 GASAPAID 28
       GAS AID
Sbjct: 254 GASVQAID 261

>pdb|1H29|A Chain A, Sulfate Respiration In Desulfovibrio Vulgaris Hildenborough: Structure Of The 16-Heme Cytochrome C Hmca At 2.5 A Resolution And A View Of Its Role In Transmembrane Electron Transfer
pdb|1H29|B Chain B, Sulfate Respiration In Desulfovibrio Vulgaris Hildenborough: Structure Of The 16-Heme Cytochrome C Hmca At 2.5 A Resolution And A View Of Its Role In Transmembrane Electron Transfer
pdb|1H29|C Chain C, Sulfate Respiration In Desulfovibrio Vulgaris Hildenborough: Structure Of The 16-Heme Cytochrome C Hmca At 2.5 A Resolution And A View Of Its Role In Transmembrane Electron Transfer
pdb|1H29|D Chain D, Sulfate Respiration In Desulfovibrio Vulgaris Hildenborough: Structure Of The 16-Heme Cytochrome C Hmca At 2.5 A Resolution And A View Of Its Role In Transmembrane Electron Transfer
Length = 514
Score = 18.0 bits (35), Expect = 1964
Identities = 4/5 (80%), Positives = 4/5 (80%)
Query: 11 CISCH 15
       CI CH
Sbjct: 83 CIGCH 87

>pdb|1DY0|A Chain A, Murine Endostatin, Crystal Form Ii
pdb|1DY1|A Chain A, Murine Endostatin, Crystal Form Iii
Length = 188

Score = 15.9 bits (30), Expect = 8542
Identities = 5/8 (62%), Positives = 6/8 (75%)

Query: 8 QQKCISCH 15
+QK SCH
Sbjct: 163 EQKAASCH 170

pdb|1C25| Human Cdc25a Catalytic Domain
Length = 161

Score = 15.9 bits (30), Expect = 8542
Identities = 5/9 (55%), Positives = 7/9 (77%)

Query: 38 EILDILNG 46
E I+ L NG
Sbjct: 28 EIMASVLNG 36

pdb|1NW1|A Chain A, Crystal Structure Of Choline Kinase
pdb|1NW1|B Chain B, Crystal Structure Of Choline Kinase
Length = 429

Score = 15.9 bits (30), Expect = 8542
Identities = 4/7 (57%), Positives = 6/7 (85%)

Query: 32 ANYSEEE 38
A+Y E+E
Sbjct: 9 AHYDEDE 15

pdb|1PS9|A Chain A, The Crystal Structure And Reaction Mechanism Of E. Coli 2,4-Dienoyl Coa Reductase
Length = 671

Score = 15.9 bits (30), Expect = 8542
Identities = 6/9 (66%), Positives = 7/9 (77%)

Query: 24 APAIDKAGA 32
A AI+ AGA
Sbjct: 234 AQAIEAAAG 242

Database: PDB protein database
Posted date: Apr 11, 2004 2:20 AM
Number of letters in database: 776,069,666
Number of sequences in database: 2,761,148

<table>
<thead>
<tr>
<th></th>
<th>K</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda</td>
<td>0.343</td>
<td>0.285</td>
</tr>
</tbody>
</table>

Gapped

<table>
<thead>
<tr>
<th></th>
<th>K</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda</td>
<td>0.294</td>
<td>0.110</td>
</tr>
</tbody>
</table>

Matrix: PAM30

Gap Penalties: Existence: 9, Extension: 1
Number of Hits to DB: 30,228
Number of Sequences: 50930
Number of extensions: 145
Number of successful extensions: 41
Number of sequences better than 10000.0: 41
Number of HSP’s better than 10000.0 without gapping: 41
Number of HSP’s successfully gapped in prelim test: 0
Number of HSP’s that attempted gapping in prelim test: 0
Number of HSP’s gapped (non-prelim): 41
length of query: 71
length of database: 12,121,459
effective HSP length: 23
effective length of query: 48
effective length of database: 10,950,069
effective search space: 525603312
effective search space used: 525603312
T: 16
A: 15
X1: 15 (7.4 bits)
X2: 35 (14.8 bits)
X3: 58 (24.6 bits)
S1: 30 (16.7 bits)
S2: 30 (15.9 bits)

Calculation of normalized scores of the residues being some elements with intermediate results

The matching matrix

The length of the query sequence is 71. After eliminating duplicate matches, we get 75 matches. The following is the matrix of secondary structure elements of the matches,
arranged from position 1 to 71 corresponding to the query sequence. The '-' indicates a position without structure information.
The normalized score matrix

This matrix gives the normalized scores of each residue being some specific secondary structure elements. So each residue has three entries, namely \( s(H) \), \( s(E) \) and \( s(C) \). These calculations are based on the weights of the matches. \( s(H) \) of position \( i \) is defined as the sum of the weights of the matches whose corresponding secondary structure element is H, normalized by the weight sum of all the matches who have a secondary structure element at position \( i \) of query sequence. \( s(E) \) and \( s(C) \) are similarly defined. Following is part of the matrices.

\[
\begin{array}{cccccccccc}
\text{\( s(H) \)} & 0.33 & 0.43 & 0.76 & 0.88 & 0.88 & 0.72 & 0.63 & 0.55 & 0.57 & 0.54 & 0.51 \\
\text{\( s(E) \)} & 0.36 & 0 & 0 & 0 & 0 & 0.19 & 0.29 & 0.19 & 0.18 & 0.16 & 0.067 \\
\text{\( s(C) \)} & 0.3 & 0.57 & 0.24 & 0.12 & 0.12 & 0.089 & 0.088 & 0.26 & 0.25 & 0.3 & 0.42 \\
\end{array}
\]

\[
\begin{array}{ccccccccccc}
\text{\( s(H) \)} & \cdots & & & & & & & & & \\
\text{\( s(E) \)} & \cdots & & & & & & & & & \\
\text{\( s(C) \)} & \cdots & & & & & & & & & \\
\end{array}
\]

\[
\begin{array}{cccccccc}
\text{\( s(H) \)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{\( s(E) \)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{\( s(C) \)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

Prediction of secondary structures

The prediction is based on the normalized score matrix. If \( s(H) \) is the highest among \( s(H) \), \( s(E) \) and \( s(C) \), then H will be the final prediction. When ties appear, choose the prediction in following order, C, H and E. The following is the final prediction for 1C75_A:

ECHHHHHHHHHHHHHHHHHHHHCCCCCCCCC
REFERENCES


Acknowledgements

Special thanks are extended to Dr. Jernigan for main ideas, detailed implementations, and analysis; to Dr. Margaritis for result analysis, implementation improvement, and methodology guide. We thank Dr. Honavar for valuable suggestions. We thank Dr. Kloczkowski for providing the test sets, for helps in result evaluation and background knowledge introduction. We thank Dr. Sen and Dr. Kundu for useful discussions. We also thank Shihe Ma for computer assistance, as well as the help in implementing and running some AI programs and some valuable suggestions.