COMPUTATIONAL GENOMICS:

GROUP MEMBERS:
Anshul Kundaje (EE) abk2001@columbia.edu
Daita Domnica Nicolae (Bio) ddc45@columbia.edu
Deniz Sarioz (CS) deniz@cs.columbia.edu
Joseph Gagliano (CS) jg327@columbia.edu

Predicting the Beta-helix fold from Protein sequence data

Abstract:

This project involves the study and implementation of Betawrap, an algorithm used to predict Beta-helix supersecondary structural motifs using protein sequence data. We begin with a biochemical view of proteins and their structural features. This is followed by a general discussion of the various methods of protein structural prediction. A structural viewpoint of the Beta-Helix is highlighted followed by a detailed discussion of Betawrap. We include the design and implementation details alongside. PERL was the programming language used for the implementation. The alpha helical filter was implemented with the help of pre-existing software known as Pred2ary. The implementation was run on a positive and negative test set, a total of 59 proteins. These tests sets were generated from the SCOP database by checking predictions made by the original Betawrap software and by directly looking at the structures in the PDB. Our program can be downloaded from betawrap.tar.gz. The archive contains a readme with necessary instructions. The program has been commented extensively. Some debugging statements have also be left as comments. The predictions made by our program are very accurate and we had a only 2 misclassifications in a set of 59 proteins.

Basic Biochemistry of Protein Structure:

Proteins are polymers of 20 different amino acids joined by peptide bonds. At physiological temperatures in aqueous solution, the polypeptide chains of proteins fold into a structure that in most cases is globular. The sequence of the different amino acids in a protein, which is directly determined by the sequence of nucleotides in the gene encoding it, is known as its primary structure (Figure 1a). This in turn determines how the protein folds into higher-level structures. The secondary structure of the polypeptide chain can take the form either of alpha helices or beta sheets, formed through regular hydrogen-bonding interactions between the N-H and C=O groups in the invariant parts of the amino acids in the polypeptide backbone (Figure 1b). Sometimes, the secondary structures can also link together to form motifs (also called supersecondary structures) by packing side-chains from adjacent alpha helices or beta sheets close to each other in space. Several motifs usually combine to form compact globular structures, which are called domains (usually encoded by an exon and constitute units of function), like the Beta-helix. One or more domains are folded into a tertiary structure (Figure 1c), which usually is a conformation that is favorable energetically and is stable. Many proteins are formed by association of the folded chains of more than one polypeptide. This constitutes the quaternary structure (Figure 1d).

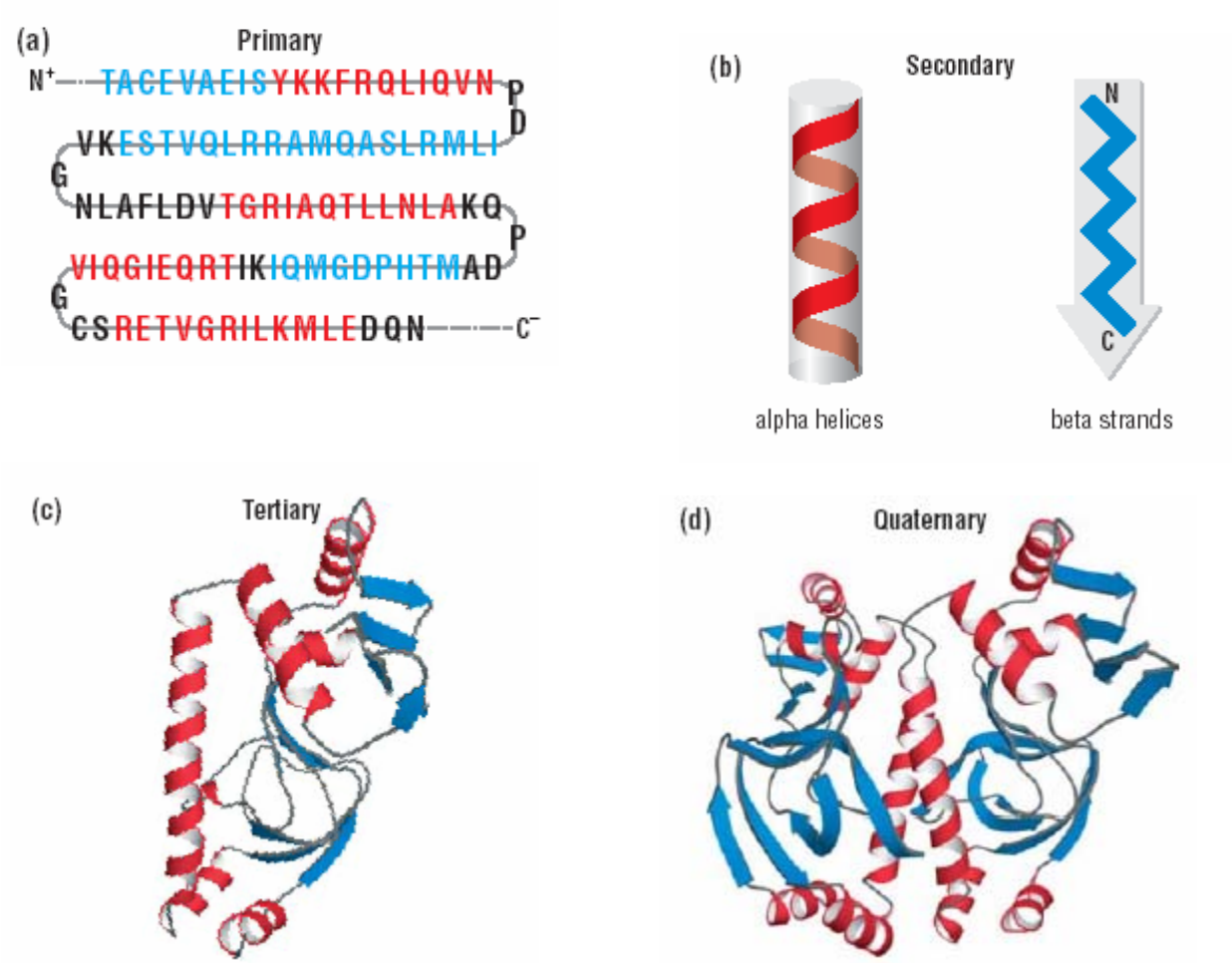


Figure 1

The amino acid side chains have tendencies to participate in interactions with each other and with water. These differences profoundly affect their contribution to stability and protein function.

Hydrophobic amino acids engage in van der Waals interactions only. Their tendency to avoid water and pack against each other is the basis for the hydrophobic effect. Alanine and Leucine are strong alpha helix-favoring residues, while Proline is rarely found in helices because its backbone N is not available for hydrogen-bonding required for helix formation. The aromatic side chain (containing the benzyl group) of Phenylalanine can sometimes participate in weakly polar interactions.

Hydrophilic amino acids are able to make hydrogen-bonds with one another, to the peptide backbone, to polar organic molecules and water. Some of them can change their charge state depending on the pH (acidity/basicity) or the immediate environment (microenvironment). Histidine is perhaps the most versatile of all residues with respect to its associations with other molecules in varying conditions, which explains why it is most often found in enzyme active sites. Cysteine is also very commonly found in enzyme active sites because of its thiolate anion which is one of the most powerful nucleophiles among the amino acids.

Amphipathic residues have both polar and non-polar character, making them ideal for forming interfaces. Lysine, Tyrosine, Tryptophan and Methionine are examples of amphipathic residues.

The Protein Folding Problem:

Protein folding is a hierarchical event in which transiently formed local structural elements assemble to yield the native conformation. The way in which a polypeptide chain folds to assume its three-dimensional (3D) shape is a fascinating problem and also has extremely important practical consequences. Despite the fact that for nearly forty years it has occupied a central place in chemistry, biophysics, and branches of the experimental and computational sciences in general, it still presents a major obstacle to understanding.

Assigning three-dimensional protein folds to genome sequences is essential to understanding protein function. Although experimental three-dimensional structures are currently available for only a very small fraction of these sequences, computational fold assignment is able to assign folds to 20-30% of the sequences in various genomes. This percentage varies depending on the particular organism under analysis, on the sensitivities of the methods used and on the number of experimental structures available at the time the assignment is carried out. Thus, in the absence of experimental structures, computational methods aimed at assigning 3D models are likely to aid in the characterization of genome proteins.

It is generally accepted that the three-dimensional structure of a polypeptide chain is determined by its amino acid sequence. Nevertheless, similar folds can have very different sequences. One of the ultimate goals in sequence analysis is to predict the structure and function of a protein based solely on its sequence. In cases where the protein of interest shares at least 30% amino acid identity with another protein, the two proteins generally exhibit similar three-dimensional structure. Alternatively, when proteins are known to have similar structure but divergent sequences, consensus sequence motifs can be used to assess the function of unassigned sequences. These consensus motifs usually correspond to residues interacting with cofactors, substrate, or other proteins.

The increasing number of three-dimensional structures of proteins in the Protein Data Bank, complexed with appropriate ligands, provides an important tool for understanding the mechanisms of molecular recognition.

Three-dimensional fold assignments:

3D assignments to genome proteins can be divided into two classes. The first corresponds to clear assignments in which the sequence similarity between the genome protein and the assigned fold is above a predefined threshold. The second class corresponds to assignments in which the sequence similarity between the genome protein and the assigned fold is below the given threshold and, thus, sequence similarity alone cannot establish their validity.

Assignments by sequence similarity:

Standard sequence comparison techniques, such as BLAST or FASTA can be applied. These techniques are very efficient and reliable in detecting the vast majority of homologs above the, so-called, twilight zone of sequence similarity. The assignment is carried out as follows: given an ORF (open reading frame) from a newly sequenced organism, a database containing the sequences of known 3D structures is searched in order to detect sequence similarities above a predefined threshold. If a sufficiently high-scoring match is found, a 3D model for the new ORF can thus be assigned. The sequence-to-sequence alignment largely corresponds to the structural alignment that could be obtained had a structure existed for the genome sequence. Thus, from such assignments, relatively accurate 3D models can be built using homology modeling techniques. The fraction of the ORFs from complete genomes that can be assigned a structure depends on the following four factors: the particular organism under consideration; the date on which the study was carried out; the sensitivity of the method used; and the threshold used to consider an assignment valid, which, in turn, determines the rate of potential false positives.

Assignments with no sequence similarity:

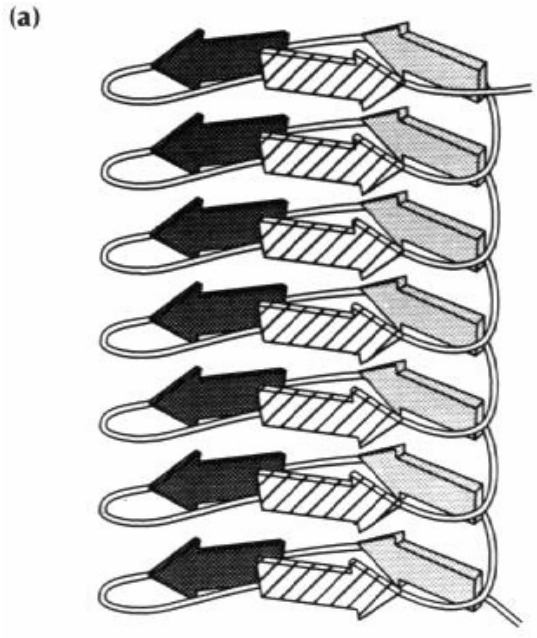
New proteins with no sequence similarity to proteins of known structure may represent a class of a new novel fold or may belong to a structural class of an observed fold. During evolution, structure is better conserved than sequence; consequently, although a new genome protein may show no sequence similarity to any protein of known structure, it may adopt a known fold. Two proteins with a similar 3D fold, but with no sequence similarity may be distant relatives belonging to the same superfamily (the sequences have diverged beyond the level of random similarity among unrelated proteins) or may be members of different superfamilies that have convergently adopted a similar 3D structure. In order to identify those genome proteins, with below threshold sequence similarity, that correspond to known folds, methods that are more sensitive than standard sequence-comparison procedures are needed. Two different types of methods have been used for this purpose: sequence-motif and sequence-profile methods and fold-assignment, fold-recognition or threading methods. Sequence-profile methods use evolutionary information from neighboring sequences in the sequence databases to build a profile. An iterative, sequence-profile method that is able to detect very distant relationships is the recently developed PSI-BLAST method. Fold-assignment or threading methods explicitly incorporate structural information from the available structures. In many cases, these methods are able to detect distantly divergent proteins, as well as unrelated proteins with a similar fold. Thus, fold-assignment methods appear to have a wider applicability than sequence-only methods. Assigning structures to genome sequences with little or no sequence similarity poses a greater challenge than pure sequence alignment. The claimed accuracy of the method applied is usually assessed using benchmarks of known structures. As predictions, some will be correct and some will not. Indeed, although several methods predict the same or a similar structure for a number of sequences, some of the predictions are mutually inconsistent; the number of assignable ORFs in thevarious genomes differs depending on the method used and some of the sequence-structure alignments may not be accurate enough to build useful homology models. Nevertheless, in the absence of experimental structures, many of the predictions can significantly aid further computational and experimental studies.

In this project we will focus on predicting the right-handed Beta-helix fold super-secondary structural motif in protein sequences. This method does not predict the entire

secondary structure of the protein. Rather based on a score it determines whether the protein contains a Beta-helix or not and specifies a P value for the prediction. It also lists tentative positions of the components of the Beta-helix. It lies half way between the two prediction methods described above as it does use information from previously identified Beta-helices to learn certain parameters but it also uses certain specific structural features (conservation of the length of the T2 turn and a highly conserved B2-T2-B3 template) to optimize the procedure.

The Beta-Helix Super-Secondary Structure:

The right-handed parallel Beta-helix motif is characterized by a series of progressive coils (called **rungs**) each of which contributes to the three long beta sheets that form a triangular prism shape to comprise the fold. The first type of parallel beta-domain was originally observed in Pectate Lyase C (Jurnak et al, 1993) (Figure 2). It is composed of a basic structural unit of **three parallel beta-strands** folded into a coil that has a rise of 4.86 Å. The structural unit is multiply repeated to form a large right-handed helix (Figure 2a). All of the parallel beta-strands are unusually short, ranging from three to five amino acids in length. Each of the parallel beta-strands in the basic structural unit forms a parallel beta sheet with parallel beta-strands in adjacent coils of the helix, giving rise to the overall appearance of a helix comprised of three parallel beta sheets. The cross-section of the helix is not circular, but rather L-shaped, because of the arrangement of the parallel beta sheets (Figure. 2b). One sheet lies perpendicular to the other two, which interact with one another in a type of parallel beta-sandwich formation. The parallel sheets are connected to each other by three regions of polypeptide strands with distinctive features. One region is composed entirely of a unique two-residue turn (the T2 turn) which forms an angle of approximately 120° between two parallel beta-strands. The average Φ and ψ torsion angles are 56.2° and 36.9° for the first residue in the turn and -102.9° and 156.9° for the second residue. The other two regions are composed of loops (T1 and T3) which vary in size and conformation. The polypeptide loops protrude from the parallel Beta-helix at distinct locations and probably form the active site regions on the external surface of the parallel Beta-helix. Because two of the loop regions connecting the parallel beta-sheets vary in size, the number and type of amino acids in each structural unit is variable. The minimum number of amino acids observed in a single coil is 22; no unique or repeated sequence pattern has been detected among the coils.



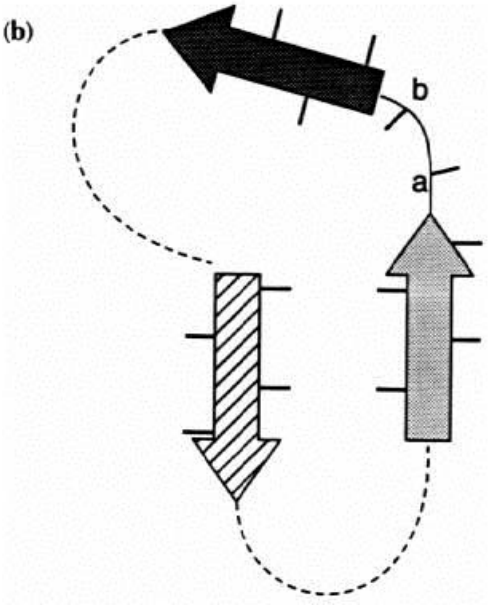


Figure 2a **Figure 2b**

Fig. 2: Schematic of the tri beta-strand class of parallel beta-domains.
The parallel beta-strands are represented by arrows that are distinctively marked according to their participation in one of three parallel-sheets. (a)View of a parallel Beta-helix perpendicular to the helical axis. The schematic illustrates seven complete coils of the basic structural unit; however, the minimum and maximum number of coils possible in a parallel Beta-helix are as yet undetermined. (b) Cross-sectional view of the tri beta-strand structural unit. Two of the parallel beta-strands interact with one another in a type of beta-sandwich interaction. The third parallel beta-strand lies approximately 120° relative to the first pair. The beta-strand markings correlate with the parallel beta sheets shown in (a). The straight lines protruding from the beta-strands represent the orientation of the side-chains. The position of the novel, highly repetitive two-residue turn is indicated by a and b.

In addition to their novel two-residue 120° turn, the first examples of the parallel beta-domains found in Pectate Lyases have some unique features with respect to the side chains. All side chains in the interior of the parallel Beta-helix are stacked with the side chains of adjacent coils, forming a linear arrangement parallel to the axis of the parallel Beta-helix. Some side chains that are oriented outward from the core are stacked as well. Among several types of interior stacks are polar stacks of Asparagine or Serine, aliphatic stacks of Alanine, Isoleucine, Leucine and Valine, and aromatic stacks of Phenylalanine or Tyrosine. The aromatic rings of the Phenylalanine or Tyrosine stacks more closely resemble base-pair stacking in double-stranded DNA than the perpendicular ring interactions commonly found in proteins. The side-chain stacking interactions are likely to contribute to the observed stability of the Pectate Lyases in solution. Another interesting feature of the Pectate Lyase parallel beta-domains is the finding that the amino acid side chains in the interior of the parallel Beta-helix are not limited to hydrophobic groups. There are polar as well as charged groups, all of which are neutralized either by maximal hydrogen-bonding or by electrostatic interactions. Moreover, the interior of the parallel Beta-helix is completely filled with side chains, leaving no room for a channel.

Other proteins reported to have parallel beta-domains include Pectate Lyases: PelC and PelE from *E. chrysanthemi* and Pel from *Bacillus subtilis*. All contain seven to eight coils of the tri beta-strand structural unit. A comparison of the parallel Beta-helices of the PelC and PelE structures reveals very little sequence similarity but a large amount of structural conservation within the core domain and in some external loops near the amino-terminal and carboxy-terminal branches. Most of the structural differences occur in loop regions near an external Ca^{2+} -binding site. These loops in B. subtilis Pel are so extended that the region is considered as a separate domain. All three proteins exhibit extensive side-chain stacks, most of which are found within the core of the parallel -helix. Another, unrelated protein, the *Salmonella typhimurium phage P22* tailspike protein (TSP) (Figure 3), has also been reported to have a tri beta-strand parallel beta-domain. TSP is a homotrimer, each subunit of which contains 13 coils of the tri beta-strand structural unit as well as multiple examples of the two-residue 120° turn. TSP, like the Pectate Lyases, is characterized by its thermostability. In contrast to the Pectate Lyases, however, TSP does not contain extended stacks of side chains in the interior of its parallel -domain. A most interesting structural similarity among all four proteins is the presence of an alpha-helix covering the amino-terminal end of the parallel Beta-helix domain. It is not known whether the alpha-helix serves only to prevent solvent from entering the semi-hydrophobic core or if it has another function. In contrast, the carboxy-terminal cap is very different in all four structures and may confer unique properties upon the proteins. The proteins all share a common enzymatic function, that of cleaving a polysaccharide. Pectate lyases cleave the alpha-(1,4) linkage of the Polygalacturonic acid component of plant cell walls and TSP cleaves the alpha-(1,3) linkage between Rhamnose and Galactose of the Salmonella O-antigen polysaccharide.

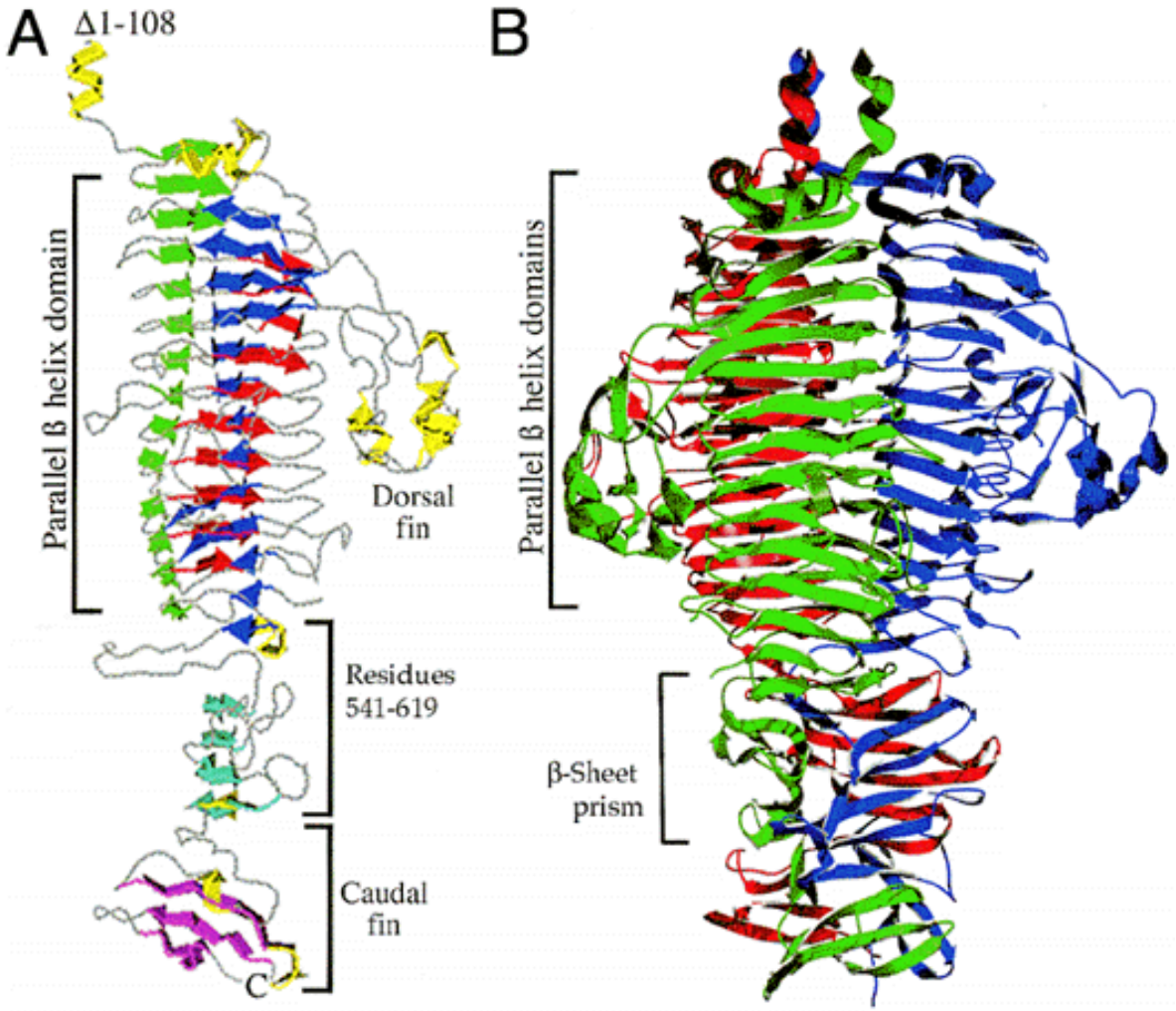


Figure 3

Fig. 3 Ribbon diagram of the P22 tailspike protein at 2 Å resolution (residues 109-666) (Steinbacher et al., 1994).
A:Tailspike subunit with the N-terminus at the top and the C-terminus at the bottom. Coiled yellow arrows indicate alpha-helices, and the other colored arrows show the locations of five beta sheets.
B: Native tailspike trimer in same orientation as subunit in A. The subunits are colored separately.

Introduction to Betawrap:

The Betawrap program use Beta-strand interactions to detect members of the parallel right-handed Beta-helix superfamily, a group of proteins characterized by widely divergent sequences but strong core structural similarities. It has been known that in beta-structural motifs amino acid residues can exhibit marked statistical preferences. However, because residues in the stacking beta-strands that are close in 3D and instrumental in the fold are far apart in the linear sequence, these preferences are hard to exploit. The Beta-helix fold has a topology which makes the prediction of interacting

residues in the stacked beta-strands more tractable. While the known Beta-helices vary in the number of rungs and the length of the turn regions, the beta-strand portions of the rungs (each layer of the stack) have patterns of pleating and hydrogen-bonding which are well conserved across the superfamily.

The main component of the Betawrap program is a novel wrapping algorithm that searches for the aligning parallel beta strands in successive rungs of the stack. With reference to Figure 4, we have 3 beta strands B1, B2 and B3 separated by 3 turns T1, T2 and T3.

The T2 turn is more conserved with a majority of rungs having a 2 residue turn at this location.

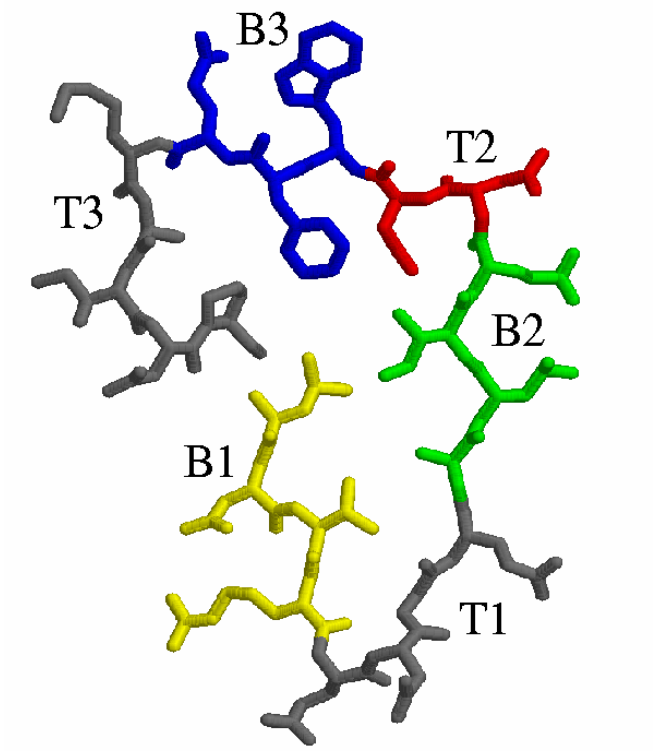


Figure 4a.

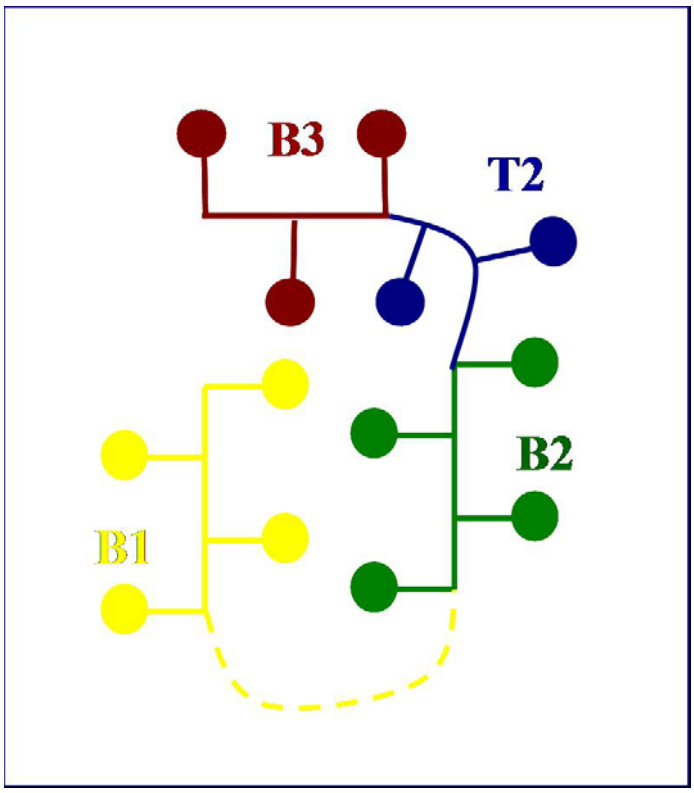


Figure 4b.

Figure 4: The rung structure

a) Top View of a single rung of a Beta helix parsed by the algorithm into beta strands B1, B2 and B3 and the intervening turns T1,T2 and T3. The alternating pattern of the strands before and after T2 is conserved across the superfamily.

b) Symbolic view of the rung. The conserved region is taken to be 4 residues of B2 (Green) in the inward-outward alternating positions, the T2 turn (2 residues in Blue) and 3 residues of B3 (brown)

Also there are no known Beta-helices having fewer than 6 consecutive rungs. Such a stack of rungs is known as a **Wrap**. Based on this fact, the algorithm looks for wraps made up of five consecutive rungs.

The assumption is that four B2 residues preceding T2 and three B3 residues following T2 participate in the beta-sheet interactions. Thus, per rung, the T2 turn is identified based on stacking preferences both in the turn and in the B2-B3 sections mentioned. Potential wraps are then created based on the alignment method described below. The location of strand B1 is then filled in for each rung to complete the parse of a generated wrap. The potential wraps are then run through 2 filters and those that survive contribute to the probability of the protein sequence belonging to the Beta-helix superfamily.

The Algorithm:

Step1: We generate possible candidates for the B2-T2-B3 segments by simply looking for the following conserved **template** **[ΦXΦX][Xψ][XΦX]**.

[ΦXΦX]	4 B2 residues	Inward-outward-inward-outward
[Xψ]	2 T2 residues	Outward-inward
[XΦX].	3 B3 residues	Outward-inward-outward

Φ – matches any one of [A,F,I,L,M,V,W or Y] all of which are hydrophobes and hence must point inward
Ψ – matched any amino acid except [D,E,R or K] all of which are ionisable amino acids
X – matches any amino acid

The segment sequence and the position of the first residue for each segment are stored. Also, the segments are stored in a sequential manner indicating their relative positions with respect to each other. Thus, these segments can also be addressed by their positional index (e.g. rung1, rung2 etc.)

Step2: Using each of these 9 amino acid rung-segments as seeds, find its **alignment score** with every other rung-segment except itself. We fix all self-alignment scores as a very large negative number (-1000000). This is to partially prevent the problem explained in Caveat1. The scoring methodology is explained below:

For a pair of aligned segments the beta sheet alignment score is the **weighted sum of seven alignment scores** for each of the pairs in the beta sheets B2 (4 residues) and B3 (3 residues). A weight of 1 is given to the scores for inward pairs and ½ for the scores of the outward pairs, to reflect the fact that environment of the inward residues is better conserved than that of the outward residues.

$$S' = 1*[R(1,1') + R(3,3') + R(8,8')] + \tfrac{1}{2}*[R(2,2') + R(4,4') + R(7,7') + R(9,9')] \quad \text{----------(1)}$$

R(n,n') represents the alignment score of the residues in the n^{th} position in the 9 residue segments. It should be read as n' aligned to n. The alignment score is natural logarithm of the conditional probability that a residue of type n' will align with residue n (thus n is the given residue and n' is scored conditionally with respect to n), given their relative orientation to the core (inward/outward).

Basically, we have 2 weight matrices of the forms shown below one for the outward case and the other for the inward case. The matrices are supposed to be learnt from a

database of aligned amphipathic beta sheets. This database excludes the Beta-helices themselves to prevent overfitting. The database is explained in more detail later. However, we used values already provided by the original Betawrap authors.

Also to improve the performance several bonuses and penalties are added:

- Based on a few known Beta-helix structures, a mean native gap between two adjacent rungs was calculated. The meaning of a native gap was assumed to be the distance (in terms of number of unmatched residues) between the B2-T2-B3 segment of one rung and the B2-T2-B3 segment of the next rung. We exclude the 9 matched residues of the first B2-T2-B3 segment that fall within this gap.
  E.g. If the first residue of B2-T2-B3 segment is at position 20 and that of the next rung is at position 40.
  The gap (consisting of unmatched residues) = | (40 – 20) | – 9 = 11

$$Gap = abs(B2 - B2') - 9 \qquad \text{----------(2)}$$

B2 => position of first residue in B2-T2-B3 segment of a rung
B2'=> position of first residue in B2-T2-B3 segment of next rung

Per protein (of the set of known Beta-Helix structures) we average the gaps between all the adjacent rungs.

$$\text{The mean native gap} = \sum(\text{average gap per protein}) / (\text{number of proteins}) \qquad \text{----------(3)}$$

We obtained a mean native gap of 12.

The score S' obtained for rung-rung segment alignment by using equation1. is adjusted using a gap penalty that penalizes alignments that leave too many or too few residues unmatched.

$$\text{The gap penalty} = P^{*}\ \text{floor}(\ \text{abs}(Gap - \text{mean native gap}) / Q) \qquad \text{----------(4)}$$

P and Q are parameters that help the gap penalty scale appropriately with the score S'. They are to be learnt from the training data. We used values of 1 and 10 respectively based on suggestions from the original authors of Betawrap.

Hence, the final score per rung-rung segment alignment is given by

$$S = S' - \text{Gap penalty} \qquad \text{----------(5)}$$

- As can be seen from table 1 for inward residues, highly charged residues (D, E, R and K) are penalized thus effectively disallowing them from appearing in the inward positions of a beta strand.
- The score matrices also take care of a bonus added when two aromatic residues (F, Y or W) appear stacked on top of each other.
- Another bonus is added a stack of beta-branched aliphatic residues (V or I) are seen in an alignment.
- The inward residues (C, S, T and N) are found to stabilize the structure and thus are given high alignment scores in general.

Step3: We then generate two matrices that hold the scores and indices of the top 5 scoring potential next rungs for each of the rungs generated from step1. To elaborate, we take each rung-segment (say rung1) from the set of rungs-segments generated by step1 and look in the matrix of rung-rung segment alignment scores to find the top 5 scoring alignments (with respect to rung1). We store the indices and the alignment score in two matrices.

Step4: Now we use the information gathered by the above 3 steps to generate potential wraps. The principle behind the algorithm is explained below after which the algorithm implementation itself is described in detail.

The algorithm is based on the fact that if we have the correct location of a single T2 turn (with B2 and B3 fixed around it) we can use the matrices generated in step3 to identify the 5 best scoring candidates for the next rung (Figure 5). Now we use each of these rungs as seeds and once again use the matrices of step3 to look up the 5 next best scoring candidates. Thus, we develop a tree structure with the root nodes as all the rung segments generated in step1. There are 5 outward branches per node (rung) representing the 5 best candidates for the next rung. Each of these has 5 outward branches and so on until we get a tree of depth 4 hops i.e. 5 nodes. Now all paths starting from the root nodes consisting of 5 nodes (4 hops) represent potential wraps. Thus, **a wrap is a stack of 5 rungs** (nodes).

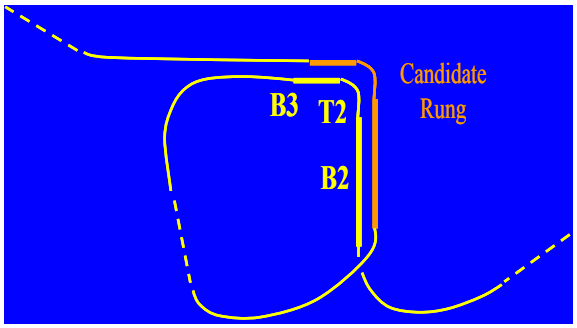


Figure 5

Caveat 1: However, we must note that not all such paths represent potential rungs. We must avoid loops i.e. repeated visits to a node (rung) in a particular wrap. We cannot have a rung of the form **1**-2-**1**-3-4 (rung indices) since each of the rungs in a wrap has to be unique. This problem is partly taken care of by disallowing a rung to be present in its own top 5 next candidate list by giving a rung self-alignment score of -100000 as mentioned in step2. This prevents the particular case of consecutive rung repeats such as 1-1-…. However, this doesn't take care of cases such as 1-2-1…. The solution to caveat2 solves this problem.

Caveat 2: Also, a wrap cannot have nodes with positions such as: 3-**70**-**45**-120-180 or 180-**3**-**76**-42-2 (hypothetical rung indices). This is because the rung indices represent their relative positions in the sequence and a wrap is formed by rungs in either ascending or descending order. In the first case, we have rung3 followed by rung70. All rungs following rung70 in a potential must have indices > 70. Similarly for the second case, we have rung180 followed by rung 3. All rungs following this one must have indices < 3. We have implemented this in our program. This also takes care of the problem mentioned in caveat1. Implementing the above mentioned points significantly

reduces the computational complexity of the wrap generating algorithm by preempting path searches that violate the conditions.

In this way, all potential wraps of 5 rungs are generated. We know only the B2-T2-B3 portions of the constituent rungs. We define the score of the whole wrap based on this score.

The score of a wrap is defined as the average of its constituent rung-segment alignment scores

The collection of wraps is now subject to 3 stages of filtering as explained later. We finally score the protein based on the top 12 scores of the wraps that survive the filtering. Averaging the top 12 wrap scores rules out spurious hits to sequences in which a single high-scoring wrap is found by chance.

The score of the protein as a candidate Beta-Helix is the average of the scores of the top 12 highest scoring wraps that survive the filtering stage

If less than 12 wraps remain after the filtering stage the protein is rejected.

The Algorithm Implementation in detail: We first look for all wraps in the forward direction (ascending order of indices), and then in the backward direction (descending order of indices). To be more specific, we do a depth-first search in each direction, starting at every different node. For the forward case, we can exclude the last 4 rungs and for the backward case, we can exclude the first 4 rungs because there is no chance of getting a wrap of length 5 in the respective directions starting from those.

We use a separate array of 5 to hold the constituent rung indices of each wrap to keep track of the nodes in a particular path of our depth-first search. For each rung, we basically iterate over all the five top scoring rungs in the appropriate direction. In each iteration, we add the appropriate alignment scores of parent to child (score) to a common score variable ($score). Once we reach the required depth (5 nodes), we push the current set of nodes in the path into an array, along with the score, and push that array into another array (so that we can sort by score later). Before we exit each of the nested loops, we simply subtract the last alignment score from common score variable ($score) to go back to the score at the previous level. We could have more efficiency if we used an array just like the one for node values to hold the scores but our implementation saves memory.

We use a **threshold** to filter out wraps in the process of the search so that we avoid very poorly scoring wraps. We chose the threshold as -20 for the whole wrap. All wraps scoring below this value were rejected before the filtering stage.

Design Considerations: This algorithm could have been made much more elegant if we used a recursive subroutine called with a "depth" variable (as well as the starting node position and the direction), but this iterative version simulates it just as well. If we did need to experiment with different "depth" values (which means number of rung-rung alignments = number_of_rungs_in_wrap - 1), we could have written a recursive subroutine for this. In the original paper, the depth is fixed to 4. (Then again, it would have been tricky to print the names of the ancestors or put them into a file at each node,

which is really the purpose of the algorithm rather than just finding the one best score. The recursive algorithm would probably have used a global array or two just like the ones we are using, and write to it using depth.)

We also realize that it would have been more efficient if we had used some amount of dynamic programming, but with each node (rung) possibly lying at any position in the wrap, preceded and succeeded by any other node (rung) (for which the score in either direction is dependent) we could not think of a meaningful dynamic programming approach. According to the original authors, the Betawrap program uses dynamic programming for this step of wrap generation. After all this, we just sort by scores and move to the filtering stage. Forward and backward wraps are grouped together.

Step 5 The B1-T1 Filter: Although the relative positioning of the rungs in a wrap is fixed by the above procedure, the positions of the B1 strands are not determined. The algorithm is supposed to score potential placement of the B1 strands into the parse using the same strand-strand alignment scores described in Step 2. The process is guided by a second gap score learned from the distributions of the T1 turn lengths in known Beta-Helix structures. There is a marked preference for T1 turns of length three, four and five. The highest scoring B1 parse is chosen for the wrap. The score for this B1 parse does not change the score of the wrap; however, a wrap is rejected if a B1 parse scoring above a predetermined threshold cannot be found.

We were unable to understand exactly how this was to be done. Hence, we developed two other filters and the classification of the test proteins was very accurate when we ran our program using them. They are explained below:
Filter1: The length of the T3-B1-T1 segments cannot be less than 8 residues in most cases assuming the **minimum lengths** of T1, B1 and T3 as 3, 3 and 2 respectively. We used this to filter out sequences whose T3-B1-T1 lengths were less than **8** (Hence, the minimum distance between the first B2 residue of one rung segment and that of the next rung segment is 9 + 8 = 17… This is the value we used in our program). Also, we limited the **maximum** T3-B1-T1 segment **lengths to 46** because beyond this length the wrap would be quite unstable. Probably, a better method might be to penalize a wrap for large T3-B1-T1 segments with lengths above this threshold, but we used the simpler method as we unable to learn the exact formula to be used for the penalty.
Filter2: Logically, if a B1 is to exist between two consecutive B2-T2-B3 segments we should be able to find a sequence of the form $[\Phi X \Phi X](\Phi$ and X taking the same values as in Step1). Here we assumed a minimum length of 3 for B1 which is reasonable as there are no known beta sheets with smaller number of residues. We search all T3-B1-T1 segments in all potential wraps for this template. If no match is found in even one of the segments in a wrap, the wrap is rejected. We constrained the search for a B1 template match by offsetting the search region by 4 residues from the end of a B2-T2-B3 segment (to accommodate a T3) and 3 residues from the beginning of the next B2-T2-B3 segment (to accommodate a T1).

Step 6 The Hydrophobic filter: Once the complete wraps are generated, they are filtered based on residues found at 2 positions in the turns. The first positions of the T1 and T2 turns show distinctive residue preferences and the large hydrophobics (V, I, L, F, M and W) are strongly disfavored. Hence, a wrap is rejected if it has more than a single hydrophobic at the T2 positions in its constituent rungs. This part of the filter has been implemented. We use a flag matrix to indicate whether a wrap is valid or not. A flag

value of 1 for a wrap indicates that it is a valid wrap and a value of 0 indicates that it is invalid. Also, a penalty is supposed to be assessed if the total number of hydrophobics at the first positions of the T1 and T2 turns exceeds a threshold to be learned from the training data. Since we were unable to locate the T1 turns we were not able to implement this section of the filter.

Step 7 The Alpha-Helix Filter: We used an alpha helix predictor called Pred2ary to predict residues that are part of alpha helices in the protein sequence. The original Betawrap uses a GOR IV implementation but since no such software was publicly available and due to lack of time to implement it ourselves we decided to use Pred2ary. Wraps are filtered on the basis of their predicted alpha content, with the aim of removing all Beta-Helix parses which overlap with alpha regions. If a 9 residue rung-segment has more than 5 residues overlapping with the predicted alpha helix regions then the wrap containing that rung is rejected. This is not exactly the filtering procedure proposed in the original paper but we found this technique simple enough for our implementation.

As mentioned before, if after filtering we have less than 12 valid wraps we reject the protein. In our program we print out the top 25 wraps if they exist.

Also, in the original Betawrap the protein sequence is prefiltered for trans-membrane alpha helices using a GES hydrophobicity scale, a window size of 21 and threshold of ~2 kcal/mol. These predicted helices are removed and the query sequence in broken down into subsequences which are scored individually. We haven't implemented this part.

The Training Database:

To generate the residue alignment scores a variant of the so called PDB-minus database was used.
The PDB-minus database was constructed from the PDB_select 25% database with the Beta-helices removed.
PDB_select is a subset of PDB in which no 2 proteins have sequence similarity greater than a cut-off which in this case is 25%. The database contained 1346 proteins.
The actual database used was constructed from PDB-minus by looking for alternating patterns of residue accessibility (inward/outward alternating). 650 protein chains were filtered out and these were used to obtain the residue alignment scores in Table 1.

The remaining parameters used by Betawrap are to be learned as follows. There are seven Beta-Helix families of closely related proteins in the SCOP database. Seven cross validations are to be performed as mentioned in the original Betawrap paper such that each cross validation run uses members of one family as training sequences and the members of other families as test sequences. The parameters that are learned include the various thresholds such as the alpha helix overlap and the B1 score threshold, the mean turn lengths, mean native gaps and hydrophobic count threshold. We learnt the mean turn lengths by running our program on 6 proteins (These 6 were not included in the positive test set). The other parameters we learnt were the value of threshold, and the minimum and maximum lengths of the T3-B1-T1 sequences. Other parameters were directly taken from the original paper.

Test Sets & Results:

The original Betawrap program is supposed to distinguish between proteins that have Beta-helices and those that don't. It also assigns a score and a P value to the protein sequence indicating how reliable the prediction is. The P value is a rough estimate of the likelihood that a randomly chosen non-beta-helix sequence from the PDB would attain a similar score. This P-value depends only on the raw score -- it doesn't take into account either the length of the query sequence or the total number of query sequences. The P-value is estimated by fitting a normal distribution to the scores of the non-beta-helix sequences in a non-redundant version of the PDB. We did not implement this P value criterion for rejection. Instead we simply reject the protein if less than 12 wraps survive the filtering stages.

The original Betawrap also tries to predict the positions of the B1 and B2 rungs in the wraps but is not totally accurate in doing so. It classifies proteins very well and a histogram plot (Figure 6) of protein scores for the SCOP Beta-helices and the PDB-minus database is shown below. 1091 proteins were rejected and these have been arbitrarily been given a score of -30. The solid line histograms represent the Beta-helices predicted from the PDB-minus database and the dotted histograms represent 12 known beta-helices. Our program was able to detect the B2 positions accurately but we did not implement a search for the B1 positions.

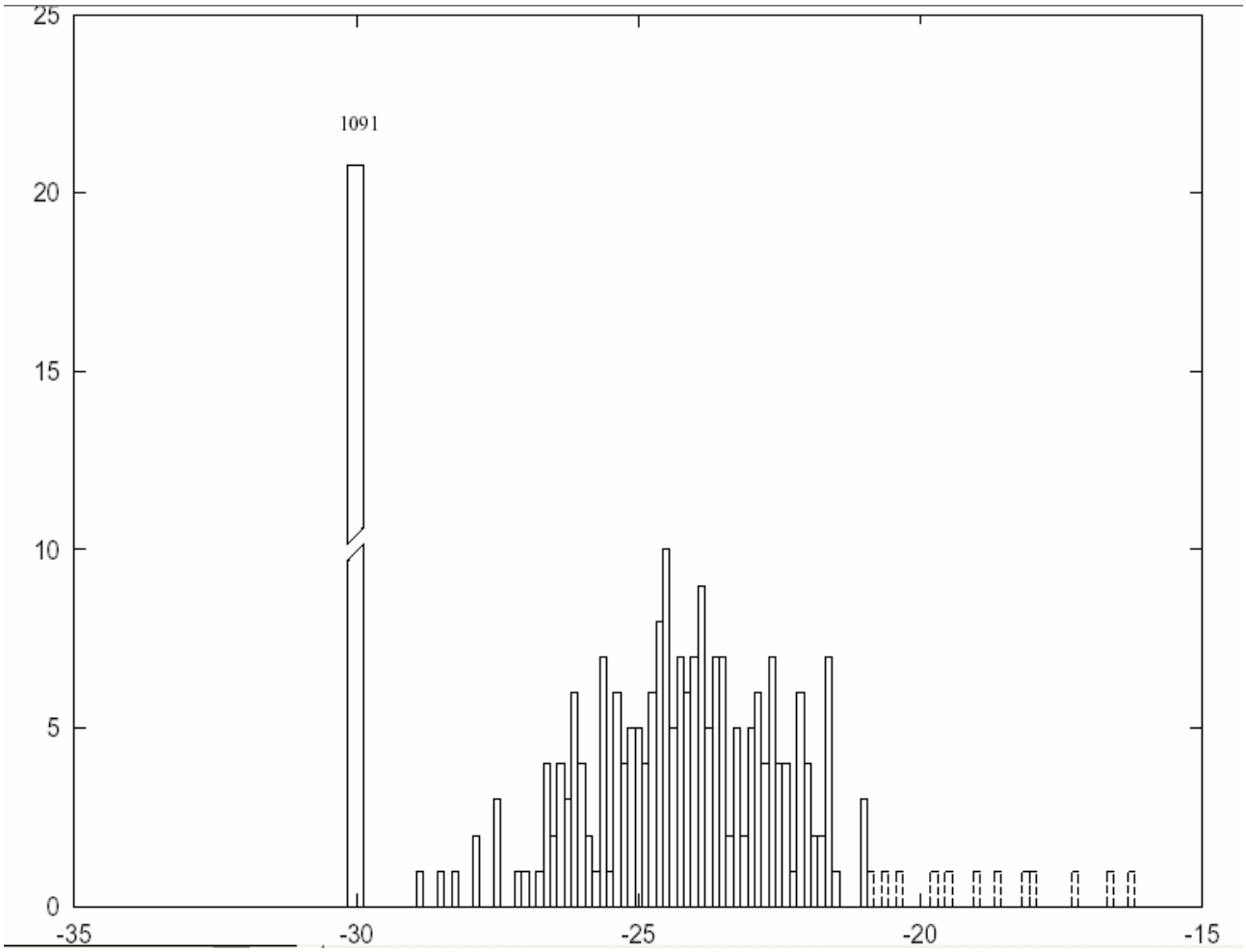


Figure 6

Figure 6: Histogram of protein scores as computed by the original Betawrap program. The dotted histograms represent 12 known Beta-helices; the solid histograms represent represent 255 proteins from the PDB-minus database predicted as being Beta-helices and the remaining 1091 proteins from the PDB-minus database that were rejected as being Beta-helices are shown to have an arbitrary score of -30.

SCOP Family	Name	Source	PDB	Rank	Score
Pectate Lyase	Pectate Lyase E	*Erwinia chrysanthemi*	1PCL	1	-16.02
Pectate Lyase	Pectate Lyase C	*Erwinia chrysanthemi*	1PLU	2	-16.44
Pectate Lyase	Pectate Lyase	*Bacillus subtilis*	1BN8	3	-18.42
Pectin Lyase	Pectin Lyase B	*Aspergillus niger*	1QCX	1	-17.09
Pectin Lyase	Pectin Lyase A	*Aspergillus niger*	1IDK	2	-17.99
Galacturonase	Polygalacturonase	*Erwinia carotovora*	1BHE	1	-18.80
Galacturonase	Polygalacturonase II	*Aspergillus niger*	1CZF	2	-19.32
Galacturonase	Rhamnogalacturonase A	*Aspergillus aculeatus*	1RMG	3	-20.12
P22 Tailspike	P22 Tailspike	*S. typhimurium* Phage P22	1TSP	1	-20.46
P.69 Pertactin	P.69 Pertactin	*Bordetella pertussis*	1DAB	1	-17.84
Chondroitinase	Chondroitinase B	*Flavobacterium heparinium*	1DBO	1	-19.55
Unclassified	Pectin Methylesterase	*Erwinia chrysanthemi*	1QJV	1	-20.74

Figure 7: The 12 known Beta-helices and their scores as predicted by the original Betawrap

We generated a positive test set and a negative test set by searching the SCOP database for Beta-helices and non-Beta-helices (based on predictions by the original Betawrap and by structural views in PDB). The negative test set was divided into two parts. The negative alpha test set is a set of 12 proteins with predominantly alpha content. Our program ran perfectly on these proteins and we obtained no false positives. The negative beta test set consists of proteins with predominantly beta content and this negative test set was a real challenge. Our program ran very on this too and we obtained only 1 false positive from a set of 14. The positive test set consists of all known beta-helices (33) from the SCOP database. We once again obtained 100% accuracy in detecting them as beta-helices. When detecting the top 12 wraps and the B2 positions in these wraps we were able to get results close to those generated by the original Betawrap but with a few wraps out of order. The example below clarifies this statement.

We used the following PDB sequence id: 2pec representing Pectate Lyase C

```
>c2pec
ATDTGGYAATAGGNVTGAVSKTATSMQDIVNIIDAARLDANGKKVKGGAYPLVITYTGNEDSLINAAAANI
CGQWSKDPRGVEIKEFTKGITIIGANGSSANFGIWIKKSSDVVVQNMRIGYLPGGAKDGDMIRVDDSPNVW
VDHNELFAANHECDGTPDNDTTFESAVDIKGASNTVTVSYNYIHGVKKVGLDGSSSSDTGRNITYHHNYYN
DVNARLPLQRGGLVHAYNNLYTNITGSGLNVRQNGQALIENNWFEKAINPVTSRYDGKNFGTWVLKGNNIT
KPADFSTYSITWTADTKPYVNADSWTSTGTFPTVAYNYSPVSAQCVKDKLPGYAGVGKNLATLTSTACK
```

The output obtained from the original Betawrap is shown in Figure8

Name	Best Wrap	P-value	Raw Score
c2pec	105 - 227	4.1e-07	-16.55

Sequence details

c2pec

BetaWrap P-value: 4.1e-07
Betawraps:

Wrap #	Rung 1		Rung 2		Rung 3		Rung 4		Rung 5	
	B1	B2	B1	B2	B1	B2	B1	B2	B1	B2
1	105	113	133	141	169	178	191	205	219	227
2	133	141	169	178	191	205	219	227	242	250
3	82	91	105	113	133	141	169	178	191	205
4	169	178	191	205	219	227	242	250	268	276
5	105	113	133	141	169	178	191	205	227	237
6	133	141	163	171	191	205	219	227	242	250
7	105	113	133	141	163	171	191	205	219	227
8	105	113	133	141	169	178	191	227	242	250
9	56	64	82	91	105	113	133	141	169	178
10	105	113	133	141	169	178	191	205	242	250

Figure 8: Ouput of the Original Betawrap for Pectate Lyase C

The output obtained from our program is shown in Figure 9: The first 5 numbers in each row indicate the B2 positions in a potential wrap and the 6th number refers to the score of that wrap.

```
Valid Wraps
The B2 positions
205 178 141 113 91  -11.2190879434208
 205 178 141 113 94  -11.5034358290067
 94 113 141 178 205  -11.5565705673774
 227 205 178 141 113  -11.5903960487345
 91 113 141 178 205  -11.6180589806121
 227 178 141 113 91  -11.7216692415541
 219 178 141 113 91  -11.7512876322015
 91 113 141 178 219  -11.7915244484754
 217 178 141 113 94  -11.8503335583775
 250 227 178 141 113  -11.8903325987681
 250 219 178 141 113  -11.9594152902084
 227 178 141 113 94  -12.00601712714
 113 141 178 205 227  -12.0819470794477
 205 180 141 113 91  -12.0896513711885
 276 250 227 205 178  -12.2082845605233
 118 141 178 205 227  -12.2752586007465
 94 113 141 180 219  -12.3534066363319
 205 180 141 113 94  -12.3739992567744
 250 227 205 178 141  -12.3976144116188
 250 217 178 141 113  -12.4124192620181
 91 113 141 180 219  -12.4148950495665
 94 113 141 180 205  -12.4306128346318
 113 141 178 219 250  -12.446852140815
 227 205 180 141 113  -12.4609594765022
 120 141 178 205 227  -12.4771733264143
```

Figure 9: Ouput of the Original Betawrap for Pectate Lyase C

As can be seen there is a lot of overlap between the two outputs. Wrap 1 in Figure8 has B2 positions as

113	141	178	205	227

This is exactly the same as row 4 in our output read in the reverse direction (since it is generated by the backward parse we output the rungs in descending order).
227 205 178 141 113 -11.5903960487344

Also all of the other wraps in the original Betawrap output have been detected by our implementation. Similar results are obtained for other sequences as well. We wrote 2 scripts to run the program on all positive and negative test sequences. The following files also present in the gzipped file show the outputs for all test sequences.
Positive test run: file1
Negative test run: file2

Comments:

We believe our implementation is rather accurate as can be seen from our results in terms of the number of false positives (1) and negatives (1) in a large test set of 59 proteins. Most of our time was spent deciphering the original paper and another major stumbling block was the implementation of the Wrap Generator. We developed a computationally efficient technique for the wrap generator that does not involve dynamic programming. Two innovative filters were also developed which work very well. All members of the group contributed to the project. We would also like to thank Prof. Christina Leslie for her guidance and Phil Bradley (co-author of the original paper) for his valuable comments.

Downloadable files:

[1] The complete set of files: betawrap.tar.gz
[2] The positive test set: pos.tar
[3] The negative test set: neg.tar
[4] Positive test set results: positiveTestResults.txt
[5] Negative test set results: negativeTestResults.txt
[6] The score matrix for exposed residues: exp.txt
[7] The score matrix for buried residues: bur.txt
[8] Readme for the program: readme.txt
[9] The original Betawrap paper: betawrap.pdf

References:

[1] Phil Bradley, Lenore Cowen, Bonnie Berger et al. ***Predicting the Beta-Helix fold from Protein Sequence Data RECOMB 2001***
[2] Dym O, Eisenberg D (2001) ***Sequence-structure analysis of FAD-containing proteins Protein Science 10: 1712-1728***
[3] Tsai CJ, Ma B, Sham YY, Kumar S, Wolfson HJ, Nussinov R. 2001 ***A hierarchical, building-block-based computational scheme for protein structure prediction IBM J. Res. & Dev. 45: No. 3/4***

[4] Fischer D, Eisenberg D. 1999 ***Predicting structures for genome proteins Current Opinion in Structural Biology 1999, 9:208-211.***
[5] Kreisberg JF, Betts SD, King J. 2000 ***Beta-Helix core packing within the triple-stranded oligomerization domain of the P22 tailspike Protein Science 9: 2338-2343.***
[6] Jurnak F, Yoder MD, Pickersgill R, Jenkins J. 1994 ***Parallel beta-domains: a new fold in protein structures Current Opinion in Structural Biology 4, 4:802-806.***
[7] Steinbacher S, Seckler R, Miller S, Steipe B, Huber R, Reinemer P. 1994. ***Crystal structure of P22 tailspike protein: Interdigitated subunits in a thermostable trimer. Science 265:383-386.***
[8] Branden C, Tooze J. 1991, 1999 (2nd Ed.) ***Introduction to Protein Structure, Garland Publishing, Inc. New York & London.***
[9] Larry Wall: ***Programming PERL (O'Reilly Publications)***
[10] James Tisdall: ***Beginning PERL for Bioinformatics (O'Reilly Publications)***