Statistical Prediction Schemes for the Coiled-Coil Motif

by

Ethan Wolf

B.A., Wesleyan University (1991)

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 1996

© Massachusetts Institute of Technology 1996. All rights reserved.
Statistical Prediction Schemes for the Coiled-Coil Motif

by

Ethan Wolf

Submitted to the Department of Mathematics
on August 06, 1996, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis proposes a new multi-dimensional scoring approach for identifying and distinguishing trimeric and dimeric coiled coils. Practical issues in the implementation of the two-stranded coiled coil prediction algorithm PairCoil suggested by Berger are discussed. This algorithm is naturally extended to the domain of three-stranded coiled coils in the implementation of the MultiCoil program. The computations are probabilistically justified and based upon data gathered from a newly constructed three-stranded coiled-coil database comprising 6319 amino acid residues, as well as from the previously constructed two-stranded coiled-coil database. In addition to identifying coiled coils not predicted by previous two-stranded database programs, MultiCoil accurately classifies the oligomerization states of known dimeric and trimeric coiled coils. Analysis of the MultiCoil scores provides insight into structural features of coiled coils, including statistically justifiable estimates of the fraction of all protein residues that form three-stranded coiled coils and the fraction that form two-stranded coiled coils. Several methods for accounting for sampling errors in the databases are suggested and empirically analyzed with regard to the performance of the MultiCoil program. A second probabilistic algorithm for classifying a given coiled coil as dimeric or trimeric is also derived and implemented.

Thesis Supervisor: Bonnie Berger
Title: Assistant Professor of Applied Mathematics
Acknowledgments

Thanks to my parents who worried far too much that I’d be in the grave before my thesis was. I always appreciate having someone to talk to who at least pretends to understand me. I hope the future brings you all the good times you deserve. To my brother, who probably doesn’t understand me, I love you as is. It’s always great to hang out with my big bro. Any thesis would be incomplete without Ray and Satomi, two of the kindest people I have ever had the pleasure of knowing. And Andrew, an incredibly sweet guy despite the effort he goes to in order to hide it. Fruit boy, I’d love to insult you here, but its yust not the same without ju. And thanks to everyone else who thinks they should be thanked, I don’t like big lists (although Ravi gets a special mention for exemplifying wit, humor, intellectualism, art, honesty, corruption, hard work, laziness, friendship, disinterested neglect, cleanliness, pragmatism, extravagant travel at the expense of someone else’s grant money, technical wizardry, expertise in any subject he knows nothing about, self deprecating conversation, and studliness at all times). Mona, I greatly appreciate talking shop with you and your ability to always put everything in perspective. To the guts of MIT- Be, Bruce, and Phyllis- my deepest respect for the details, helping me procrastinate, and providing an island of humanity. Thanks to Dan Kleitman for a number of very helpful heated discussions. And, despite the fact that I never acted too happy to get advice when it meant I had to go implement yet another test, in hindsight, I appreciate the enthusiasm, suggestions, and insights of my advisers Bonnie Berger and Peter Kim. You have my immense gratitude for getting me involved in problems that I really want to work on. Thanks especially to Bonnie for all the confidence she has shown throughout. And of course, Stubs, thank you for being there to sleep through everything.

As with all good things in my life, I dedicate this thesis to the purest of them all. I miss you Monkey.
## Contents

1 Introduction
   1.1 Biological Background ........................................... 10
   1.2 Other Algorithms for Predicting Protein Folding ................ 15
   1.3 Predicting Coiled Coils using Databases of Known Coiled Coils ... 18

2 The PairCoil Program
   2.1 The Databases used in the PairCoil Implementation ............... 22
   2.2 Estimation of Parameters from the Databases .................... 24
   2.3 Performance of the PairCoil Implementation ..................... 25
   2.4 Converting Scores into Probabilities ............................ 27
   2.5 Results From the PairCoil Implementation ....................... 28

3 The MultiCoil Program
   3.1 Methods .......................................................... 32
      3.1.1 The Databases and Probability Estimates .................... 33
      3.1.2 The Algorithm ............................................... 34
      3.1.3 Converting Scores to Probabilities ......................... 35
      3.1.4 Estimating the Fraction of Residues that are Dimeric and Trimeric 36
      3.1.5 Coiled-Coil and Sequence Probabilities ..................... 38
      3.1.6 Using Fewer Dimensions for Improved Performance ............ 39
   3.2 Results ........................................................... 40
      3.2.1 Scoring Dimensions ......................................... 41
3.2.2 Score Distribution of Dimers, Trimers, and Non-coiled-coil Sequences .......... 40
3.2.3 Separation of Data Sets and Classification ............................................. 40
3.2.4 Classification of the Known Databases .................................................. 40
3.2.5 MultiCoil on "Unknown" Test Data ...................................................... 42
3.2.6 Predictions for Protein Sequences Derived from the GCN4 Leucine Zipper ..... 42
3.3 Discussion .................................................................................................... 43
   3.3.1 Performance Compared to PairCoil ....................................................... 43
   3.3.2 Differences Between the Datasets ......................................................... 43
   3.3.3 Improving Performance ........................................................................ 45
3.4 An Alternative Version of the MultiCoil program ....................................... 45

4 Estimating Probabilities from the Databases ............................................... 55
   4.1 Estimating Residue Correlation Probabilities ........................................... 55
   4.2 Formal Methods for Estimating Probabilities from a Sample ..................... 57
   4.3 Other Distributions for the a priori Probabilities ....................................... 58
   4.4 Accounting for Sample Errors in Other Fields of Research ....................... 60
   4.5 Methods of Data Smoothing .................................................................. 61
      4.5.1 Additive Smoothing ....................................................................... 61
      4.5.2 Good-Turing Estimates .................................................................. 61
      4.5.3 Absolute Discounting ..................................................................... 62
   4.6 Combining Smoothing Methods ................................................................ 62
      4.6.1 Multiple Probability Models ............................................................. 62
   4.7 Using only Structurally Important Heptad-Repeat Positions ....................... 63

5 The PairDiff Program: An Alternative Approach to Determining Oligomerization States 75

6 Conclusion ..................................................................................................... 79

A A Brief Guide for using the MultiCoil Program ............................................. 83
   A.1 Interpreting Scores .................................................................................. 84
      A.1.1 Interpreting Residue Scores .............................................................. 85
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>11</td>
</tr>
<tr>
<td>1-2</td>
<td>12</td>
</tr>
<tr>
<td>1-3</td>
<td>13</td>
</tr>
<tr>
<td>1-4</td>
<td>14</td>
</tr>
<tr>
<td>2-1</td>
<td>29</td>
</tr>
<tr>
<td>2-2</td>
<td>31</td>
</tr>
<tr>
<td>3-1</td>
<td>47</td>
</tr>
<tr>
<td>3-2</td>
<td>48</td>
</tr>
<tr>
<td>3-3</td>
<td>49</td>
</tr>
<tr>
<td>3-4</td>
<td>50</td>
</tr>
<tr>
<td>3-5</td>
<td>50</td>
</tr>
<tr>
<td>3-6</td>
<td>51</td>
</tr>
<tr>
<td>3-7</td>
<td>52</td>
</tr>
<tr>
<td>3-8</td>
<td>53</td>
</tr>
<tr>
<td>3-9</td>
<td>54</td>
</tr>
<tr>
<td>4-1</td>
<td>64</td>
</tr>
<tr>
<td>4-2</td>
<td>65</td>
</tr>
<tr>
<td>4-3</td>
<td>66</td>
</tr>
<tr>
<td>4-4</td>
<td>67</td>
</tr>
<tr>
<td>4-5</td>
<td>68</td>
</tr>
<tr>
<td>4-6</td>
<td>69</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The coiled-coil motif is composed of right-handed $\alpha$-helices wrapped around each other with a slight left-handed superhelical twist. Homo-oligomers form from monomers of the same $\alpha$-helical sequence, while hetero-oligomers form from distinct $\alpha$-helical monomers. Two-stranded (dimeric), three-stranded (trimeric), and four-stranded (tetrameric) versions of coiled coils are all possible. Two-stranded coiled coils have drawn particular interest because of the biologically important “leucine zipper” motif (55), found in several DNA-binding proteins. Three-stranded coiled coils have been located in influenza hemagglutinin and Moloney murine leukemia virus and are thought to play a role in membrane fusion (20, 36, 62).

Coiled-coil motifs are particularly well suited to computer-based prediction schemes due to the characteristic repeating pattern of hydrophobic residues spaced every four and then three residues apart. This pattern forms a heptad repeat $(abcdefg)_n$ of amino acids in which positions $a$ and $d$ tend to be hydrophobic and positions $e$ and $g$ are predominantly charged residues. The interactions between amino acids at these heptad positions in pairs of $\alpha$-helices are essential to the formation of the structure. The specificity of these interactions and the amount of packing space required for the residues at each position vary depending on the oligomerization state of the coiled coil (42).

Due to the regular structure of coiled coils, statistics-based prediction programs have met with recent successes in identifying coiled coils (64, 11). Statistics-based recognition methods built upon a database consisting of proteins known to form coiled coils take advantage of trends of amino acids that occur preferentially in particular heptad-repeat positions in the database. These prediction programs have typically been based on a database of two-stranded coiled coils, and have been shown to locate both homo- and hetero-oligomers.
This thesis extends the previous prediction programs to the problem of both identifying and distinguishing two-stranded and three-stranded coiled coils. New techniques were required, as the simplistic method of running one of the previous coiled-coil predictor programs using probabilities gathered from a database of known trimers was found to be insufficient for the task. The general structural similarities of dimeric and trimeric coiled coils result in the location of a number of trimeric coiled coils by programs when using only a two-stranded database, with certain trimers scoring higher than some of the dimers. Similarly, it was found that using a newly-constructed database of three-stranded coiled coils led to the identification of a number of dimeric coiled coils, with several scoring higher than a number of the trimers. The small size of the three-stranded database places additional constraints on the performance. The MultiCoil program developed in this thesis solves these problems by simultaneously computing scores from each database to obtain a multi-dimensional score vector. The strength of a residue's dimeric score relative to the strength of its trimeric score in this multidimensional scoring space determines its classification as a two-stranded coiled coil, a three-stranded coiled coil, or as a non-coiled coil. Another generalization of the previous statistical prediction schemes developed in this thesis yields a second algorithm for distinguishing dimeric coiled coils from trimeric coiled coils based upon conditional probability analysis, assuming that the region which forms the coiled coil is already known. In the remainder of this thesis this dimer/trimer differentiator will be referred to as the PairDiff program.

At a high level, the problem addressed by any statistical prediction program, can be structured into three stages. The first step involves choosing statistical parameters that represent the features essential to defining the structural motif. Once these "structurally important" parameters are chosen, they must be estimated by compiling a database of proteins known to form the structure. The frequencies of the events in the database are used to estimate the probability of those events in all occurrences of the motif. The second stage in prediction involves using these parameters to score an amino acid sequence with unknown structure. The statistical parameters are combined in such a way that the scores for sequences that form the motif can be separated from the scores of all other sequences. Converting these scores into a meaningful format for the program user to make a decision about the sequence structure is the third stage of statistical prediction. Scores are converted into probabilities that the protein forms the motif using methods from classification theory. Note that for the implementations of the algorithms presented in this thesis, the statistically important parameters were partially chosen based upon the performance of the program. The parameters tend to have biological significance, suggesting that the prediction programs pick out structurally important details.

This three-stage framework provides a structure in which to assess the contributions presented in this thesis. Two different methods are presented to attack the problem of locating and differentiating two- and three-stranded coiled coils. These scoring algorithms both build upon the PairCoil algorithm of Berger, et
Additionally, the MultiCoil prediction method presented here provides a general multi-dimensional framework which can be used with any statistical prediction scheme. As databases are created for other structures, this multi-dimensional classification scheme can easily be adapted to locate and distinguish these motifs. A probabilistic framework is maintained throughout the computations, giving mathematical justification to the algorithms. The remainder of the thesis addresses practical issues that affect the performance of these algorithms, and concentrates on the first and third stages of motif prediction discussed earlier. Theoretically, the scores computed by the PairCoil-style algorithms should represent probabilities. Practically, however, inaccuracies arise from the simplifying assumption that only certain statistical features of the amino-acid sequence are "structurally important". Drawing on ideas from the field of classification theory, this thesis presents several methods for converting scores into probabilities in chapters 2 and 3.

The second source of inaccuracies in the implementations arises from sampling errors in estimating the "structurally important" parameters from the database of known structures. Inaccuracies in these databases can adversely affect the performance of the program. Since there will always be inaccuracies in a sample, the goal is to make sure that these inaccuracies have the least detrimental affect on the particular program's performance. To this end, several methods for estimating parameters were implemented with the MultiCoil program and analyzed empirically in chapter 4. The performance of the program under the various schemes gives insight into both the scoring algorithm and the nature of the sample parameters.

1.1 Biological Background

The structure of a protein is typically broken into levels. The primary structure is given by the linear sequence of amino acids in the peptide chain. Each amino acid is composed of a central carbon atom (Cα) attached to an amino group (NH₂), a carboxy group (COOH), and a side chain (see Figure 1-1). The amino acid proline varies from the structure in Figure 1-1, as there is a second bond from the side chain to the backbone at the nitrogen. This eliminates one of the hydrogens in the amino group (NH₂), which significantly affects the role of proline in protein structure. The 20 possible side chains bestow each amino acid with properties that determine its interactions with other amino acids and with the environment. Protein sequences arise from amino acids via the formation of a peptide bond between the carboxy group of one amino acid and the amino group of another. The bond forms via the loss of a water molecule. This creates a protein backbone, with the side chains from each amino acid hanging off of the alpha carbons. There is also a "direction" of the backbone. One protein end will have a free amino group (the N-terminus) and the other ends with a carboxy group (the C-terminus).
Figure 1-1: The left panel contains a molecular diagram of an amino acid. $R$ represents the amino acid side chain, composed of a number of other molecules that define the specific amino acid. Proline has a link from $R$ to $N$, eliminating a hydrogen atom. The right panel shows the structure of the peptide backbone after the formation of a peptide bond via the loss of a water molecule.

When a peptide chain folds into a three dimensional structure, the folding is believed to be driven by attempting to minimize the energy of the system. The amount of space taken by a side chain, the distribution of charges on atoms, and the ability to form bonds with other spatially nearby atoms all contribute to the folding process. Since proteins typically occur in aqueous environments, cores of hydrophobic amino acids tend to form during protein folding. The first level of folding is believed to occur in the formation of regular local structures, called secondary structures. The two most common secondary structures are alpha helices and beta sheets. Both of these structures are locally stabilized by hydrogen bonds within the peptide backbone between the oxygen of one amino acid and the hydrogen from the $NH$ of another nearby amino acid. Since proline residues do not have this hydrogen attached to their backbone nitrogen, they often occur as helix breakers. One turn of an $\alpha$-helix contains 3.6 residues, with the side chains pointing outwards (see figure 1-2) (19).

The final level of folding is the tertiary structure, in which long range interactions between amino acids distant from each other in the primary structure occur via interactions between the secondary structures. The folded structure can either be described in the Cartesian model, by stating the coordinates of each atom, or in the torsional model, where the protein backbone is described by the angles between the central $C_\alpha$ carbon and the amino group (the $\psi$ angle) and between the $C_\alpha$ and the carboxy group (the $\phi$ angle).

The biological structure of the coiled-coil motif considered in this thesis arises as a super-secondary structure from the interaction between two or more $\alpha$-helices. The helices wrap around each other with a slight superhelical twist. Coiled coils can form from parallel $\alpha$-helices, which run the same direction from $N$ to $C$ terminus, and anti-parallel helices. The superhelical twist of the helices effectively shortens their repeating pattern from 3.6 residue per helical turn, so that the helices instead face each other every 3.5 residues. This
Figure 1-2: A molecular diagram of one turn of an α-helix. The hydrogen bonds across the peptide backbone stabilize the structure. For simplicity, hydrogens were only shown for the nitrogens, and only the two oxygens taking part in the hydrogen bond were shown. The side chains point outwards. A schematic view of the 3.6 residue helix turn also appears to the right.

creates a pattern of interactions between the two helices which repeats integrally every 7 residues. Thus, a heptad-repeat pattern is introduced in which the residues in each strand of the coiled coil can be labeled by one of seven position (a,b,c,d,e,f,g) (see figure 1-3). The residues in a given position have common properties determined by the environment faced by the residues in that position of the coiled coil. Positions a and d interact along the inner face of the two helices, and so they tend to be hydrophobic residues. Positions b, e and f tend to have fewer general constraints imposed upon them, since they are on the outer face of the coiled coil, and so their properties tend to depend on the environment in which the coiled coil occurs. Additional interactions between residues on the inside of a coiled coil, such as space constraints and charge interactions, are reflected by which pairs of residues occur in the two interacting positions. For example, positions g and e are commonly occupied by charged residues. Favored interactions involve oppositely charged pairs of residues.

The physical interactions between the α-helices in a coiled coil are commonly represented by a "helical wheel" diagram (figure 1-4). This representation highlights the spatial locality of positions along the interior of the coiled coil. However, the diagram does oversimplify the picture. For example, while a residue in position g is affected by the residue in the following position e of the other helix, its interaction with the preceding residue in position e is physically blocked by other residue side chains. As the diagrams suggest, more of the coiled-coil surface is buried in the interior for higher oligomerization states. Dimeric coiled coils have
Figure 1-3: The super-helical twist wrapping two α-helices around each other to form a coiled coil. The right panel labels the heptad-positions in two turns of the helices by their environments.
strongly buried interactions between positions a and d. Positions e and g in trimeric and tetrameric coiled coils are also buried. And tetrameric coiled coils have positions b and c partially buried. A tetrameric version of the leucine zipper protein has 1640 Å² buried per helix, as compared to the 900 Å² buried for a dimeric leucine zipper (42, 40, 63).

Figure 1-4: Helical wheel diagrams for (a) a two-stranded, (b) a three-stranded, and (c) a four-stranded coiled coil. These diagrams indicate the spatial interactions between the different strands of the coiled coil. The helical wheels do not represent other factors, such as the presence of other residues blocking interactions between heptad-repeat position e and the following position g.

It has also been proposed that due to the more spatially extended conformation of β-branched amino acids (valine, isoleucine, and threonine), their occurrence at position d disfavors the formation of dimers, while their occurrence at position a disfavors the formation of trimers (leading to the favored formation of trimers when β-branched residues occur at both positions). Not only does the oligomerization state affect the spatial packing of residues, but it also affects the distinction between the “inside” of the protein where structural interactions occur, and the outside of the protein which is affected by the environment (42). The PairCoil algorithm
presented later was designed to take advantage of these interactions along the interior of the coiled coil.

1.2 Other Algorithms for Predicting Protein Folding

The field of protein structure prediction from amino acid sequences has recently been a diverse and rich field of research. The methods can be divided into algorithms which concentrate on predicting local secondary structures as building blocks for the tertiary structure (23), and those methods which attempt to deduce the entire tertiary structure, with secondary structures only appearing as a by-product. One approach to these problems involves determining similarities between the sequence of an unknown fold and sequences whose structures are already known. These approaches rely on gathering a diverse and representative data set of solved structures and measuring the fit of these structures to the unknown sequences. Energy functions have also been used to search a space of possible structures for the one that minimizes the energy of the folded sequence. Defining the search space and the energy function and efficiently searching the space dictate the performance of these algorithms.

The simplest model of folding using energy minimization is the lattice model, where each amino acid is reduced to a point on a lattice. The amino acids are classified as hydrophobic or hydrophilic. The protein sequence is then folded on the lattice based on maximizing the number of contacts between hydrophobic residues (or alternatively, minimizing the number of hydrophobic contacts with the environment and other hydrophilic residues). Hart and Istrail presented a 3/8 approximation algorithm for the problem on a cubic lattice (44). The cubic lattice is inherently limited. Parity problems make it impossible for residues at an even distance from each other in the sequence to contact each other on the lattice. Batzoglou and Decatur (8) gave a linear time 3/5 approximation algorithm for the problem on a triangular lattice, which does not suffer from the parity problem. They gave another constant factor approximation for a generalized model which allows for the energy function to be computed based upon assigning each amino acid a level of hydrophobicity. Such models are obviously limited since they ignore many of the properties which factor into folding, such as spatial constraints and charge interactions.

At the other extreme of energy functions are molecular dynamics approaches, which model the physical forces between the atoms in the protein (65, 58). The overwhelming computation requirements of force simulations currently make molecular dynamics approaches computing all interactions for a general protein impractical (41). Instead, simplifying assumptions must be worked into the energy function or the search space, limiting which atomic interactions contribute to the energy or which backbone confirmations are allowed. Though currently necessary, such simplifications are dangerous since the difference in the energy of
the native fold of a protein and other hypothetical folds is often extremely small. Some success has been achieved by molecular dynamics approaches limited to the coiled-coil and α-helical bundle motifs. Harbury et al. (41) used a parameterization of the coiled-coil backbone proposed by Crick (31) to constrain the energy search space. The molecular interactions of the protein residues were considered for the full side chains at coiled-coil positions a and d, and for truncated side chains at the e and g positions. Their approach correctly predicted the structures of a dimeric, a trimeric, and a tetrameric mutation of the GCN4 leucine zipper protein.

For predicting the structure of arbitrary proteins, Avbelj and Moult (4) based an energy function simplification on the assumption that early folding is dominated by local main chain electrostatic interactions, non-local main chain electrostatic interactions (through hydrogen bonding), and the burial of non-polar areas. In a similar method, Srinivasan and Rose (86) defined an energy function based on attractive and repulsive contact energies (scaled by distance), hydrogen bonding potentials, and a torsional potential energy to push torsional φ angles to negative values, except for glycine. Each of these contributions was computed in a simplified model based on biological knowledge. In both these algorithms, Monte Carlo methods were used to randomly simulate the folding of the protein. In the Avbelj and Moult algorithm, the torsional angles were randomly perturbed at each step, and then the energy of the new conformation was evaluated. If the perturbation caused atoms to overlap, the move was rejected. If the energy decreased, the move was accepted. An increase in the energy by ΔG was accepted in a randomized fashion based on the magnitude of the energy change with probability equal to the Boltzmann-weighted energy $e^{-ΔG/kT}$. The LINUS program of Srinivasan and Rose also performed Monte Carlo simulations with perturbations of the torsional angles. However, their perturbations were more rigidly defined to limit the search space. They defined four types of moves which were consistent with forming a helix, a sheet, a turn, or a random coil based on knowledge gathered from known structures. A new protein conformation was accepted with probability equal to the Boltzmann-weighted energy. Their computations were based on intervals of amino acids which grew in size with the progression of the algorithm. Thus, local interactions governed the early stage of their folding, while long-range interactions played a role in later stages. As in all Monte Carlo methods, numerous simulations of the protein folding process were used with both programs, and the structures which led to low energies were considered as the predicted fold.

To limit the search space, a number of “template” and “profile” methods have also been proposed. These methods use families of proteins with the same solved structure to define amino acid environments at each position in the structure. Unknown proteins are then aligned with these profiles or template environments, and predicted to fold into the same structure if the alignment is good. Sequence alignment is commonly done with a version of the Needleman-Wunsch algorithm (68). The success of the methods relies on the assumption that sequences with similar amino acid contents will form similar three-dimensional folded structures.
Bowie, Lüthy, and Eisenberg (18) applied profile methods with environments defined as buried, partially buried, or exposed according to the residue's solvent-accessible surface area. In addition, the environments of the solved structures were classified as α-helical, β-sheets, or other. In all, 18 environments were defined and used to determine alignments between the unknown sequence and the profiles. Bashford, Clothia, and Lesk (6) applied similar template alignment methods to the globin family of proteins. Their environments consisted of information about hydrophobicity, hydrophilicity, and size of the residues at the various positions gathered from several known globin structures. The globin structure was broken into six conserved structural regions. Insertions and deletions were allowed in the alignments between these regions, but were not allowed within the motifs.

"Threading" methods expand upon the ideas of aligning with solved structures. These methods replace the amino side chains in the solved protein structure with the side chains from an amino acid sequence being evaluated. These structures are then perturbed and the energy of the structure is evaluated. The success of threading in predicting a general fold requires a large database of solved backbone structures from which to generalize. Sippl applied threading methods with an energy function determined from the distribution on the physical distances between residues in the folded proteins from the solved database (84, 46). The energy was determined by interactions between residues at distance $k$ from each other in the primary sequence, where making $k$ small highlighted secondary structures, while long range interactions were captured by larger $k$.

Godzik, Kolinski, and Skolnick (39) implemented threading methods by embedding template structures based on a database of 125 solved structures into a lattice. For an unknown protein structure threaded on the template, the energy was based on contributions from: $<1>$ how well hydrophobic residues were buried in the interior and how well polar residues were exposed, $<2>$ pairwise interactions between contacting side chains (where side chains were considered to be in contact whenever the two side chains contained a pair of heavy (non-hydrogen) atoms closer than 5 Å), and $<3>$ interactions between triplets of residues. The energy function was broken into terms that depended only on the structure of that template fold, and terms which depended on the particular amino acids in each position. The terms depending only on the template structure defined a template energy function for scoring unknown proteins. The energy of the system of threaded amino acids was minimized to determine the best template fold for the unknown protein. The method then refined the structure by projecting the protein atoms onto a 3-dimensional cubic lattice. The threaded structure provided a base structure which was perturbed on the lattice using Monte Carlo dynamics to find the most stable conformation.

These threading approaches tend to be more successful when the protein under consideration is very similar to a protein in the database. In these cases, the threading and template methods often result in fairly accurate
tertiary structure predictions. However, the alignment of the threaded sequences with their actual secondary structures was found to be generally poor (57). In tests of other protein structure prediction schemes that required the use of alignments, it was found that optimizing the automatically predicted alignments by hand improved the ability to accurately predict secondary structure (32).

The previous energy functions were all explicitly defined. Methods from learning theory have also been proposed to recognize the best structure, essentially "learning" an energy function by training on the known data set. A neural network was constructed for classifying secondary structures as \(\alpha\)-helical, \(\beta\)-sheet, or random coil. It was found that a neural network without hidden units was able to obtain predictive accuracy competitive with hidden layer networks. The overall predictions were measured to be 63% accurate, comparable to the accuracy in the 60% range achieved by other secondary structure predictors based on local interactions (23, 59, 37). Hidden Markov models (HMM) from learning theory have also been used to predict protein structure from statistical data available from known structures (47, 5).

1.3 Predicting Coiled Coils using Databases of Known Coiled Coils

Due to the regular heptad-repeat pattern in coiled coils, statistical database methods have also proved to be the most successful in predicting this super-secondary structure. Lupas, Van Dyke, and Stock (64) proposed the COILS2 and NEWCOILS programs, based upon a scheme suggested by Parry (73). A database of known coiled coils was first compiled and the the frequency of each amino acid in each of the seven heptad-repeat positions was calculated. "Single" residue probabilities were derived from these frequencies by dividing the residue's frequency by the total count. The relative frequency of an amino acid was defined by dividing the observed residue probability in the coiled-coil database by the observed probability of the residue in all proteins (as computed from the GenBank database of proteins). The GenBank database represents a collection of protein sequences which have been sequenced by biologists. The score for a given heptad-repeat frame of a window was computed as the geometric average of the relative frequencies of the residues in that window. The score for the window was the maximum of the scores over all seven heptad-repeat frames. The score for a residue was the maximum score over all windows containing the residue. Assuming all residues within a sequence were independent, Berger (10) justified the NEWCOILS score in terms of an estimate for the probability that the window forms a coiled coil.

The simplifying assumption that the residues within a sequence are independent and inaccuracies in the probabilities estimated from the sample database decrease the accuracy of the score as a probability estimate. Therefore, in order to convert the scores into probabilities, the NEWCOILS method first computed a distri-
bution of scores on the set of coiled-coil residues, and on a set of known globular non-coiled-coil residues. Gaussians were fit to these distributions. Additionally, scores for the residues in GenBank were generated, approximating the distribution of scores in all proteins. Taking scaled combinations of the Gaussians fit to the coiled-coil and globular databases, the distribution on all GenBank proteins was approximated. The best fit was given by the scaled combination which assumed that there were 30 non-coiled-coil residues for each coiled-coil residue. For a NEWCOILS score of \( s \), conditional probability analysis was then used to estimate the probability that \( s \) represents a coiled-coil residue by:

\[
P(s) = \frac{1}{30} G_{cc}(s) / [G_{g}(s) + \frac{1}{30} G_{cc}(s)],
\]

where \( G_{cc} \) and \( G_{g} \) are the Gaussians for the coiled-coil and the globular score distributions. The NEWCOILS program has been quite successful, but has also been shown to be prone to "false positives", predicting non-coiled-coil regions that are \( \alpha \)-helical to be coiled coils (11).

The program PairCoil(11) sought to cut down on these false positive predictions by extending the statistical analysis to pairwise residue correlation probabilities. Pairwise probabilities pick up on interactions between pairs of residues within two strands of a homo-dimeric coiled coil, as well as interactions within each strand. By using this extra information in the computation of scores, PairCoil attempted to achieve increased accuracy in prediction. The PairCoil algorithm and its implementation provide the basis for the work of this thesis, and it is discussed in depth in the next chapter.

Hidden Markov models (HMM) (13) have also been proposed for locating coiled coils, based upon a similar probabilistic analysis used to derive the PairCoil algorithm (see the next chapter). The implementation of the methods in the program HMMCoil achieve similar performance to the PairCoil program (7). The HMM implementation differs from the PairCoil and NEWCOILS programs by considering transition probabilities of a residue in one heptad-repeat position following a residue in another heptad-repeat position. Transitions can cause register skips, or can cause a coiled-coil to begin or end at the residue. These transition probabilities were estimated from a database of known coiled coils, as well as from the best biological estimates available for the lengths of coiled coils and the fraction of residues in coiled coils.

All three of these coiled-coil prediction programs estimated the probabilities from a database of known dimeric coiled coils. This thesis extends the probabilistic framework of the PairCoil program to the problem of identifying and classifying coiled coils and their oligomerization state. The implementation is based upon pairwise residue correlations estimated from both two-stranded and three-stranded coiled-coil databases. The problem of distinguishing dimeric coiled coils from trimeric coiled coils was also concurrently considered.
by Woolfson and Alber (97) using statistical profile methods and single residue probabilities. Loosely speaking, their program makes a measure of the fit of the sequence being tested to a database of two-stranded coiled coils versus its fit to a database of three-stranded coiled coils. Their score is reminiscent of the NEWCOILS score, which used single probabilities to measure the fit between the sequence and the coiled-coil database versus its fit to a random sequence in GenBank. In fact, the PairDiff algorithm derived later in this thesis gives probabilistic justification to their score.

In the Woolfson approach, a coiled-coil predictor was first used to locate potential coiled-coil regions under loose criteria dominated by hydrophobic considerations. Their coiled-coil locator was designed to return proteins with uninterrupted heptad-repeat patterns and unambiguous registers on which to run the distinguisher. This COILER program selects sequences based on <1> the presence of heptad repeat patterns with hydrophobic residues in positions a and d containing at least 28 residues, <2> bounding the content of Proline and Glycine in the coiled-coil regions, and <3> limiting the hydrophobic content of the whole coiled-coil region to 65%.

A second round of processing was then applied to classify the predicted regions as either trimeric or dimeric. This second round distinguisher (called SCORER) made predictions based upon the single residue probabilities estimated from dimeric and trimeric coiled-coil databases. Ignoring an additive constant, the score is the sum over all residues and predicted heptad-repeat positions in the potential coiled-coil region of the logarithm of the ratio between the estimated probability for the residue in dimeric coiled coils and its estimated probability in trimeric coiled coils. Formally, let \( P_2 \) denote a residue probability observed in the dimeric database by dividing its frequency by the total number of samples, and let \( P_3 \) denote the residue probability observed in the trimeric database. Let \((r, p)\) denote a residue \( r \) in a heptad-repeat position \( p \). Then the SCORER score is:

\[
S(\text{coiled-coil region}) = \sum_{(r, p) \text{ in region}} \log \left( \frac{P_3(r, p)}{P_2(r, p)} \right) + c.
\]

This score attempts to capture whether the sequence has more residues in common with the dimeric database or the trimeric database. The derivation of the PairDiff algorithm later in this thesis gives probabilistic justification to this approach. Note that the number of terms summed depends on the length of the predicted coiled-coil region, so a region can obtain a score with higher magnitude based on its length or based on the strength of individual biases towards one oligomerization state. Woolfson and Alber showed that, for a specific choice of the additive constant and by limiting which (residue, heptad-repeat position) pairs contribute to the sum, the sequences in a selected dimeric data set scored above zero, while the sequences in a selected trimeric data set scored below zero. The sum was limited to register positions a, d, e, and g, which lie along
the interior face of the coiled coil. Residues which were “rare” in the assigned heptad-repeat position in either the trimeric or dimeric database were also excluded, and only values which differed significantly between the two databases were included. These exclusions were designed to make the score dominated by only the structurally important residues, while eliminating spurious effects from the sample database. The elimination of these terms cut the set of possible contributing residue, heptad-repeat position pairs from the \(140 = 20 \times 7\) originally possible to 17. The contributing residues were \(A, K, L, R,\) and \(S\) at position \(a; I, L,\) and \(V\) at position \(d; E, L, Q,\) and \(S\) at position \(e; E, L, N, Q,\) and \(R\) at position \(g.\)

While the SCORER program obtained separation on the data sets, there are several troubling issues. First, predictors based upon hydrophobic content that do not consider pairwise interactions between heptad-repeat positions have been shown to be prone to “false positives” (11, 97). Moreover, the amount of separation between the lowest scoring dimer and the highest scoring trimer does not appear to be significant. In addition, their score is computed over the entire length of the coiled coil, giving the same score to the entire region. This score could arise from small values over the length of a long coiled coil, or from a small number of residues contributing large values and dominating the score. The score does not give any indication of how biases for the oligomerization state vary across the length of the coiled coil. It is unclear from their paper how well the program locates coiled-coil regions relative to the number of “false positives”, and how well the program predicts the oligomerization state when run on larger samples of sequences.

Another approach which has been suggested for locating trimeric coiled coils involves “learning” the structure. Berger and Singh (12) implemented a learning procedure (LearnCoil) built upon the PairCoil program. The two-stranded coiled-coil database was used as the base data set from which to estimate probability parameters. The LearnCoil program was run iteratively on a large data set of unlabeled test sequences. Each sequence was randomly selected with the probability at which it was scored by the program. The selected sequences were then used to update the parameters estimated from the database using a Bayesian-like weighting scheme. Note that this procedure highlights the importance of accurately converting scores into probability estimates (the third stage of the statistical predictions programs). Inaccurate probabilities can lead to under-learning or over-learning. Using this procedure with the methods for converting scores into probabilities presented in the next chapter on a test data set that included a database of three-stranded coiled coils, LearnCoil was able to increase the recognition of the three-stranded database by PairCoil from 64% to 94%. Combining the LearnCoil methods with the multi-dimensional framework proposed later in this thesis is a promising topic for future research.
Chapter 2

The PairCoil Program

In order to take advantage of spatial interactions between pairs of residues in two strands of a coiled coil and within single strands, Berger applied the use of conditional probabilities. Assuming that all spatial dependencies could be captured by pairs of amino acids within the coiled coil, the probability of a region forming a coiled coil was approximated by a formula of single residue and pairwise probability terms (10). The interaction between residue $r_1$ in one strand of a homo-oligomeric coiled coil and residue $r_2$ of the other strand were captured by the pairwise dependency between the two residues $r_1$ and $r_2$ in the single sequence. These interactions were measured in terms of correlations and anti-correlations between pairs of residues in a database gathered from known coiled coils. In using conditional probabilities to derive a formula for the probability that a window of $w$ residues $z = z_1 z_2 \cdots z_w$ forms a coiled coil, it should be noted that this probability is either 1 or 0 for any given sequence. However, the problem can also be stated in terms of the probability that a sequence $X = X_1 X_2 \cdots X_w$ chosen randomly from some distribution forms a coiled coil and is identical with $z$. It is this probability that the PairCoil algorithm approximates. In the following discussion, whenever a coiled-coil probability is discussed, it is with respect to some fixed setting for the seven possible heptad-repeat positions for the residues in the window. Conditioning on the value of $X$ gives:

$$
Pr[z \text{ is coiled coil}] = \frac{Pr[X \text{ is coiled coil}, X_1 = z_1, X_2 = z_2, \ldots, X_w = z_w]}{Pr[X_1 = z_1, X_2 = z_2, \ldots, X_w = z_w].}
$$

(2.1)
The analysis for the denominator and the numerator proceeds similarly. In the following, let \( C_X \) denote the event "\( X \) is a coiled coil". Repeated application of the definition of conditional probability to the numerator gives:

\[
\Pr[C_X, X_1 = z_1, X_2 = z_2, \ldots, X_w = z_w] = \\
\Pr[X_1 = z_1|C_X, X_2 = z_2, \ldots, X_w = z_w]\Pr[C_X, X_2 = z_2, \ldots, X_w = z_w] = \\
\vdots \\
\Pr[X_1 = z_1|C_X, X_2 = z_2, \ldots, X_w = z_w]\Pr[X_2 = z_2|C_X, X_3 = z_3, \ldots, X_w = z_w] \\
\cdots \Pr[X_{w-1} = z_{w-1}|C_X, X_w = z_w]\Pr[X_w = z_w|C_X] \Pr[C_X].
\]

The computation of this formula still includes dependencies between all the residues in the window, which is too much to be estimated from a small database. If the assumption is made that pairwise dependencies between amino acids at distance \( d \) capture all dependencies between the residues in the sequence, then this formula reduces to:

\[
\Pr[C_X, X_1 = z_1, X_2 = z_2, \ldots, X_w = z_w] = \\
\Pr[X_1 = z_1|C_X, X_{1+d} = z_{1+d}] \cdots \Pr[X_{w-d} = z_{w-d}|C_X, X_w = z_w] \cdot \\
\Pr[X_{w-d+1} = z_{w-d+1}|C_X] \cdots \Pr[X_w = z_w|C_X] \cdot \Pr[C_X] \\
= \frac{\Pr[X_1 = z_1, X_{1+d} = z_{1+d}|C_X] \cdots \Pr[X_{w-d} = z_{w-d}, X_w = z_w|C_X]}{\Pr[X_{1+d} = z_{1+d}|C_X] \cdots \Pr[X_w = z_w|C_X] \cdot \Pr[C_X]} \\
\prod_{i=1}^{w-d} \frac{\Pr[X_i = z_i, X_{i+d} = z_{i+d}|C_X]}{\Pr[X_i = z_i|C_X]} \cdot \Pr[C_X].
\]

If the computation is allowed to go outside of the window in order to pick up dependencies at distance \( d \) for amino acids within \( d \) residues of the end of the window, then the formula becomes:

\[
\Pr[C_X, X_1 = z_1, X_2 = z_2, \ldots, X_w = z_w] = \\
\frac{\prod_{i=1}^{w} \Pr[X_i = z_i, X_{i+d} = z_{i+d}|C_X]}{\prod_{i=1}^{w} \Pr[X_i = z_i|C_X]} \cdot \Pr[C_X].
\]

These terms can be estimated from a coiled-coil database.
The formula for the denominator in equation 2.1 is computed analogously from the underlying distribution of all proteins, giving the probability estimate:

$$
Pr[z \text{ is coiled coil}] =
\frac{Pr[C_X] \cdot \prod_i Pr[X_i = z_i, X_{i+d} = z_{i+d}|C_X] \cdot \prod_i Pr[X_i = z_i]}{\prod_i Pr[X_i = z_i, X_{i+d} = z_{i+d}]}. \quad (2.2)
$$

Berger (10) showed that assuming that $X_i$ and $X_{i+d}$ are independent further reduces the formula to the quantity computed by the NEWCOILS algorithm raised to the 28th power and multiplied by the term $Pr[C_X]$, giving a probabilistic framework for their algorithm. The PairCoil algorithm for estimating the probability that the given sequence $z$ forms a coiled coil is based upon estimating the formula of pairwise dependencies.

### 2.1 The Databases used in the PairCoil Implementation

The implementation of the algorithm requires the estimation of each of the components in the formula of equation 2.2. For the probabilities conditioned on $X$ being a coiled coil, a sample database was constructed from sequences classified as two-stranded coiled coils in the literature, and extracted from two databases of proteins sequenced by biologists, GenBank (release 73, September 1992) and the Protein Identification Resource (PIR, release 34, September 1992). Before discussing the details of this coiled coil database, the choice of an underlying distribution for the random sequence $X$ must be discussed. Theoretically, the choice of this distribution should be unimportant, as long as it has positive probability for all possible input sequences $z$, since the derivation of the formula holds for all such distributions.

The simplest choice for the distribution for $X$ would be the uniform distribution, so each particular value for $X_i$ would have probability $1/20$ and each pair of residues would have probability $1/400$. However, in choosing the distribution on $X$, it is important to consider the estimation of the other parameters in the computation of the formula. The probabilities $Pr[X_i = z_i|C_X]$ and $Pr[X_i = z_i, X_{i+d} = z_{i+d}|C_X]$ which are used to estimate $Pr[X_i = z_i, \ldots, X_w = z_w|C_X]$ come from the sample database of coiled coils. In this database, any particular window of $w$ residues can occur multiple times as a subsequence of different coiled-coil regions in this database. Such repetitions conflict with the distribution induced by the uniform distribution, which induces a distribution in which each $w$ long window in the coiled coil dataset is equally probable. Instead, the distribution on the coiled-coil database is induced by the proteins which have been sequenced and shown to form coiled coils.
A better choice of the underlying distribution is the distribution determined by the GenBank database. The advantages of this distribution in the computation of the formula in equation 2.2 are highlighted by considering a rare event in the coiled-coil database. The event can be rare for two reasons: 1. the event may be unfavorable for forming coiled coils; 2. the event may be rare in all protein sequences, independent of whether the event is correlated with the sequence forming a coiled coil. In the second case, using GenBank as the underlying distribution in essence cancels out the affect of the small probability of the event in the coiled coil database, because the term also has a low probability in the denominator of the formula. An underlying uniform distribution would not make such a correction, since the denominator would be a constant \((1/20)^w\) independent of the protein sequence \(z\). It is important to keep in mind that while the PairCoil implementation using the GenBank distribution and the coiled coil database is the most natural, it is not the only option. The uniform distribution could be used, as long as the estimates for the probabilities conditioned on \(X\) being a coiled coil are estimated according to a coiled coil database constructed so that each window of length \(w\) occurs at most once. Since the formula from equation 2.2 was derived using the simplifying assumption that interactions between residues at distance \(d\) from each other capture all dependencies, the inaccuracies introduced into the computation by different underlying distributions could cause one implementation to perform better than another.

### 2.2 Estimation of Parameters from the Databases

The probability of residues and pairs of residues in GenBank are easy to compute. The frequency count of each residue (or each pair of residues separated by distance \(d\) in a protein sequence) is divided by the total number of residues (or pairs of residues at distance \(d\)) in the database. Since GenBank is an extremely large sample of protein sequences, it is reasonable to assume that these observed probabilities provide accurate estimates.

The estimation of the residue and pairwise probabilities for coiled coils is more difficult. The first concern is the sample database, which is limited by what is biologically known. The sequences included in the two-stranded database were four families of keratins; intermediate filaments consisting of vimentin, desmin, and glial fibrillary acidic protein; one family for each of neurofilament H, L, and M; lamins; myosins; tropomyosins; and paramyosins. The coiled-coil region for each family was determined by alignment with a proto-typical member of the family. Berger et al. (11) give the details of these coiled-coil regions.

Since the database consisted of a relatively small sample of coiled coils (226 sequences), the simple estimate of probabilities obtained by dividing the frequency of the event in the database by the total number of sample points is not the best estimate. Sample errors are especially damaging to the computation when they
result in "false zero frequency" events, even though the event might occur in a coiled coil outside of the sample database.

To cope with this problem, the implementation of the algorithm presented in Berger et al. (11) gave each zero frequency event (other than events involving the amino acid proline) a phantom frequency of 1/5, and then computed the probabilities based on these frequencies by dividing the event frequency by the total number of events. If the resultant probability for some pair of residues in a pair of heptad-repeat positions was greater than the product of the estimated single probabilities for each of those residues (which is probabilistically impossible), then the pairwise probability was set to this product instead. The pairwise probabilities were then renormalized. The value of 1/5 was empirically chosen based on the performance of the PairCoil program. The amino acid proline was automatically given a probability of 0 (with no phantom zero frequency), since prolines are known helix breakers due to their looped structure and inability to hydrogen bond along the protein backbone. Other methods for the estimation of single residue and pairwise probabilities from a sample database are discussed in greater depth later in this thesis.

The only term in equation 2.2 remaining to estimate is the probability that a random peptide sequence of length \( w \) (selected from GenBank) forms a coiled coil. The usefulness of the formula in equation 2.2 as a probability is however limited by two sources of error. Inaccuracies arise from the simplifying assumption that all dependencies between residues in the sequence are captured by residues at distance \( d \) from each other and from inaccuracies in the parameters estimated from the sample databases. These errors mean that the values computed by the formula do not correspond exactly to probabilities. Since the factor \( Pr[C_X] \) is present in the formula for all sequences scored, it occurs as a multiplicative constant in the formula. Let \( P_{\text{estimate}}(z) \) denote the probability estimate obtained from formula 2.2. Then, all information obtainable for classifying the sequence \( z \) as a coiled coil or a non-coiled coil is contained in the value: \( \log P_{\text{estimate}}(z) - \log Pr[C_X] \). This value is computed as the PairCoil score for the window \( W \) of \( w \) residues \( z_1 \cdots z_w \), assuming a particular heptad-repeat register \( r \) and using dependency distance \( d \). Denote this value by \( S_{w,r,d} \). Each window has seven such scores corresponding to the different possible values for the heptad-repeat register. The score for the window using dependency distance \( d \) is taken to be the maximum score over the heptad-repeat positions. Each residue \( j \) in the interior of a sequence is contained in \( w \) windows, so its PairCoil score \( S_d \) for interactions at distance \( d \) is defined to be the maximum of the scores for the windows containing it. Formally, let \( W \) range over all windows containing \( j \), and let \( r \) range over the seven possible values for the heptad-repeat position of the window:

\[
S_d(j) = \max_W \max_r S_{w,r,d}.
\]
Both $S_d(j)$ and the particular heptad-repeat position score $S_{W,r,d}$ are available as output from the PairCoil implementation.

The PairCoil implementation can compute scores based on multiple values for the interaction distance $d$. In particular, rather than assuming that all dependencies are captured by interactions at distance $d$ in deriving equation 2.2, suppose that pairwise interactions at distances $d_1, d_2, \ldots, d_k$ capture all dependencies. Then in the derivation of equation 2.2, the terms $Pr[X_i = z_i|C_X, X_{i+d} = z_{i+d}]$ would be replaced by $Pr[X_i = z_i|C_X, X_{i+d_1} = z_{i+d_1}, \ldots, X_{i+d_k} = z_{i+d_k}]$. Heuristic methods were used to approximate this value by conditioning on each distance separately. The PairCoil implementation presented in Berger et al. (11) approximated this value by the geometric mean of the probabilities for the individual distances:

$$(Pr[X_i = z_i|C_X, X_{i+d_1} = z_{i+d_1}] \cdots Pr[X_i = z_i|C_X, X_{i+d_k} = z_{i+d_k}])^{1/k}. $$

The computation of $P_{\text{estimate}}(x)$ in equation 2.2 proceeds the same as before, with each single residue probability for $X_{i+d}$ replaced by the geometric mean of the single residue probabilities for $X_{i+d_1}, \ldots, X_{i+d_k}$, and with each pair probability at distance $d$ replaced by the geometric mean of the pair probabilities at distances $d_1, \ldots, d_k$. The geometric mean tends to weight the value towards the minimum values, highlighting negative correlations between residues in the sequence being scored. This attribute is useful to the PairCoil implementation, which attempted to cut down on the number of “false positive” coiled-coil predictions arising from the NEWCOILS program.

### 2.3 Performance of the PairCoil Implementation

Using distances 1, 2, and 4 with windows of size $w = 30$, the PairCoil implementation achieved no overlap between the scores of the residues in the coiled-coil database and the scores of the residues in a database of known non-coiled-coil sequences (Figure 2-1(A)). The NEWCOILS program was unable to completely distinguish the non-coiled-coil data from the coiled-coil data, and the overlap is shown in pink in the figure. The non-coiled-coil database was constructed from all sequences with solved crystal structures that do not form coiled coils in the Protein Data Bank (PDB), and is represented by PDB-minus. In order to test the program in an unbiased manner on the known coiled coils, each coiled-coil sequence was removed from the database before testing. The probabilities estimated from the database were then recomputed from this modified coiled-coil database. The choice of distances 1, 2, and 4 was made empirically for the best performance, but they are
also structurally relevant, with distance 4 capturing the important d to a interaction, distance 2 capturing correlations between residues in positions e and g, and distance 1 capturing the spatial locality of neighboring residues. The window size of 30 was also chosen empirically (based on the proposal that most structurally stable coiled coils are at least four heptads long) (64).

2.4 Converting Scores into Probabilities

Figure 2-1, panel (B) also shows the distribution of scores for the two programs on the PIR-minus database, consisting of sequences from the PIR (release 38.09,1994) with all sequences in the coiled-coil database removed. The overlap between the scores in the PIR-minus database and the scores for the coiled-coil database represent sequences in the PIR-minus database which are likely to form coiled coils. The distribution of these scores was used to convert the PairCoil scores back into probabilities. Scores from the PIR-minus database are approximately Gaussian distributed, with extra mass on the right-hand tail of the distribution above the Gaussian fit to the data in the figure. When the scores from the coiled coil database are also included with the PIR-minus database, they contribute even more extra mass to the right-hand tail. A measure of the extra mass at each score value was used to convert the PairCoil scores into probabilities. An advantage of this conversion method is that it does not rely on the distribution of the PairCoil scores when run on the coiled-coil database. Since the scoring algorithm is based on statistics gathered from this database, this feature simplifies the conversion, since biased scores are not an issue.

The first problem for the conversion involved fitting an appropriate Gaussian to the data without the extra mass attributed to coiled coils. Given the biological data currently available, it was estimated that 1 out of every 50 residues in the PIR were in coiled coils. To fit a Gaussian to the histogram data, the mean was calculated so that the extra probability mass to the right of the mean would correspond to 1/50 of the total mass of the PIR. The standard deviation was then computed using only scores below that mean, where a Gaussian better fits the histogram data. The histograms for the PIR-minus and PIR corresponded well at values below this mean. The probability \( p(x) \) that a residue with a given PairCoil score \( x \) is a coiled coil was estimated as the ratio of the extra histogram mass above the Gaussian at that score to the total histogram mass at that score (Figure 2-1 (C)Right). A least square fit line was then used to approximate the probability data in the linear region from 10 to 90 percent. The prediction percentage for a given PairCoil score \( x \) was calculated using the equation \( p(x) = 7.45x + 25.84 \), where \( x \in [-2, 9] \). In this implementation, a PairCoil score of 3.24 corresponded to a 50% prediction. The NEWCOILS probabilities in figure 2-1 were computed as in Lupas et al. (64).
Figure 2.1: (A) A histogram representation of the distribution of the scores from NEWCOILS and PairCoil when run on the 58,217 residues in the two-stranded coiled-coil database (blue) versus the scores of the PDB-minus (63,116 residues) (green). The plotted NEWCOILS score is the logarithm of their window score, as that value is analogous to the PairCoil score. The overlap (pirk) in the NEWCOILS scores represents erroneous predictions. (B) Distribution of the scores when run on the PIR-minus database (7,322,501 residues) (green) and the coiled-coil database (blue). A Gaussian is fit to the PIR-minus scores. The area of overlap indicates residues in the PIR-minus database which are likely to be in a coiled coil. (C) Estimates of the probability that a given score represents a coiled-coil residue. The height of the histograms were normalized to give each data set the same area.
2.5 Results From the PairCoil Implementation

Using the output probability computations for PairCoil allows for the comparison of its performance with that of NEWCOILS. Using a 50% cutoff (the cutoff which Lupas et al. considered to be positive predictions), the two programs were run on the PDB database. The format of the listing is (PDB name, residue positions in the sequence, NEWCOILS prediction, PairCoil prediction). Both programs correctly classified the following six protein sequences as coiled coils: (1COS, 1-31, 100.0%, 100.0%); (3HMG, 51-91, 99.6%, 96.1%); (1ZTA, 1-35, 100.0%, 100.0%); (2ZTA, 1-34, 100.0%, 100.0%); (2TMA, 1-285, 100.0%, 100.0%); (1MLP, 1-58, 99.9%, 100.0%). The following 14 sequences out of the 286 non-coiled-coil sequences in the PDB-minus database scored as false positives with NEWCOILS, while not with PairCoil: (1ADA, 304-331, 49.5%, 0.0%); (2TSI, 290-317, 73.1%, 0.0%); (9LDB, 40-67, 86.6%, 2.2%); (1YPI, 137-164, 90.4%, 0.0%); (1CSG, 14-41, 51.2%, 0.0%); (1EMD, 191-218, 67.6%, 0.0%); (1FLX, 41-68, 58.7%, 0.0%); (1APK, 53-80, 91.8%, 0.0%); (1GPA, 102-129, 77.9%, 15.4%); (3BLM, 99-126, 56.7%, 31.3%); (1LE2, 33-60, 93.6%, 31.9%); (256B, 1-28, 73.7%, 12.4%); (2HPD, 197-224, 55.0%, 33.3%); (1LMB, 9-36, 91.5%, 35.2%). All these regions correspond to α-helical regions, but the X-ray crystal structures reveal that none of them correspond to coiled coils. These 14 false positive predictions out of the 20 sequences from the PDB predicted to form coiled coils by NEWCOILS corresponds to over 2/3 error rate on the positive predictions for the solved structures. The PairCoil program is therefore useful for eliminating false positives, while correctly identifying coiled-coil regions. It appears to be especially useful for distinguishing α-helical regions from coiled coils. Since PairCoil correctly classifies all coiled-coil sequences in the two-stranded coiled-coil database (scoring them above 50%), and since it eliminates the 14 false positives of NEWCOILS, while not introducing any new false positives, it represents a notable improvement. Figure 2-2 shows a scatter plot comparing the NEWCOILS and PairCoil sequence scores for all sequences in the PIR-minus database. The 50% cutoff for each program is marked. The plot indicates that the results of the two programs agree on the majority of residues. However, a large number of sequences that NEWCOILS scores as coiled-coil residues above 50% are not scored as coiled coils by PairCoil. It seems likely that a number of these scores correspond to α-helical regions which do not form coiled coils.
Figure 2-2: Scatter plot of the NEWCOILS score (x-axis) versus the PairCoil score (y-axis) for sequences in the PIR-minus database. The sequence score is defined to be the maximum score over all residue scores in the sequence. Corresponding probabilities are also labeled on the axes. The horizontal line marks the PairCoil 50% cutoff, and the vertical line marks the NEWCOILS 50% cutoff. The points in the lower-right quadrant correspond to sequences predicted as coiled coils by NEWCOILS, but not by PairCoil. There are many more of these sequences than sequences predicted as coiled coils only by PairCoil, corresponding to points in the upper-left quadrant.
Chapter 3

The MultiCoil Program

The MultiCoil program (96) extends the PairCoil program to the problem of identifying and distinguishing dimeric and trimeric coiled coils based upon data gathered from the previously constructed two-stranded coiled coil database and a newly-constructed database of three-stranded coiled coils. MultiCoil uses multi-dimensional score vectors to distinguish the structures, and uses techniques from classification theory in order to convert these scores into probability predictions. The multi-dimensional analysis allows for the simultaneous use of both the two- and the three-stranded databases to predict coiled coils, and automatically combine the various predictions. The scores from the different dimensions can reinforce each other so that residues which are weakly predicted in a number of the dimensions can be predicted much more strongly in the combined result. Note that a residue which has a stronger predicted PairCoil probability using one database over the other is not necessarily classified in that oligomerization state by MultiCoil, since the multi-dimensional distribution of the score is used for classification.

The performance of the MultiCoil program is measured both in terms of its ability to locate and to classify sequences of known two-stranded and three-stranded coiled coils, as well as its performance on databases of sequences with unknown structures. The program correctly classifies the oligomerization state of all sequences from the known databases, without returning any "false positives". As in PairCoil (11, 10), the use of pairwise residue correlations for the MultiCoil computations provides significantly better performance than can be achieved by using single frequency methods, based on trial implementations. In addition to classifying individual sequences, MultiCoil scores also give a measure of which pairwise residue interaction distances are most influential in differentiating dimers, trimers, and non-coiled coils. These results
support biological models proposing which residue interactions within coiled coils most affect their formation and oligomerization state (42, 63, 50). A maximum likelihood approach is also used to estimate the fraction of all sequences that contain trimeric coiled coils and the fraction that contain dimeric coiled coils. These parameters coincide remarkably well with the estimates made in Lupas et al. and Berger et al. (64, 11), and provide additional insight into the frequency with which trimeric and dimeric coiled coils occur. The distribution of the MultiCoil scores also support the hypothesis that trimeric coiled coils allow for more freedom in packing than dimeric coiled coils (97).

3.1 Methods

3.1.1 The Databases and Probability Estimates

The database of non-coiled-coil sequences was constructed from sequences in the Brookhaven Protein Data Bank (PDB) with solved crystal structures as for the PairCoil program, and is again represented by PDB-minus. Two databases of dimeric and trimeric coiled coils were also constructed. The construction of the two-stranded coiled-coil database was discussed earlier in this thesis and in Berger et al.(11). The three-stranded coiled-coil database was constructed from 42 sequences (6319 residues). The proteins in the three-stranded database have been characterized in the following references: laminins (families A, B1, and B2 chains) (81, 83, 82, 48, 78, 52, 91, 38, 94, 76, 77, 66, 67, 49, 69, 9, 35), fibrinogens (families α, β, and γ chains) (93, 74, 54, 25, 80, 24, 29, 30, 26, 92, 72, 17, 87), S.cerevisiae heat shock transcription factor (85), K. lactis heat shock factor (75), influenza virus hemagglutinin (20), Moloney murine leukemia virus (36), fibrinogen encoded by wac gene for bacteriophage T4 and K3 (34), gp17 leg protein in bacteriophages T7 and T3, as well as macrophage scavenger receptor protein (28).

Additional coiled-coil regions for fibrinogen and laminin sequences from species not discussed in the references were identified by using sequence alignments obtained from the program PILEUP (1). Fibrinogen strands were aligned with the corresponding human sequences characterized by Conway and Parry (28). The laminin A chains were aligned with Conway and Parry's trimeric coiled-coil predictions for mouse A chain. Strands from the B1 and B2 laminin families were aligned with the corresponding human sequences. Note that Drosophila A was not included due to its poor alignment with mouse A chain.

The guidelines for determining which regions of the sequences were to be included were as follows. The seven residues before and the seven residues after skips, deletions, and register shifts in alignments were cut because of uncertainties as to the exact location of such structural changes. The seven residues before and the
seven residues after each proline were cut from any coiled-coil regions, as prolines are helix breakers (27). Only coiled coils of at least 28 residues were included in the frequency counts, consistent with the finding that short, stable coiled coils are approximately four heptads long (71, 62, 56).

The counts of amino acids in heptad-repeat positions in the databases were used to estimate the corresponding single and pair probabilities for use in the MultiCoil algorithm, using phantom zero frequency 1/3 (see section 2.2). Note that in the implementation presented here, the single probabilities for residues B, Z, and X were calculated as follows. The probability of B was computed by averaging the probabilities of the Asparagine and Aspartic Acid residues. The probability of Z was obtained by averaging the probabilities of Glutamine and Glutamic Acid. An unknown residue, X, was given probability 1/20 in all databases, since there are 20 amino acids that could occur in the unknown position. Pairwise probabilities for these residues were computed analogously.

3.1.2 The Algorithm

The MultiCoil algorithm uses the PairCoil scorer implemented in Berger et al.(11) as a sub-process to compute scores for each residue in a sequence (note that these are the PairCoil scores before they are converted into probabilities). The results in this thesis are based on scoring windows of size 28 (four heptads). Scoring windows of size 21 and 14 are also available for predicting shorter coiled coils, but were found to predict less accurately. Rather than combining the pairwise probabilities to simulate interactions at multiple distances as in the PairCoil program, the multi-dimensional MultiCoil scorer computes a PairCoil score for each of the seven possible pairwise interaction distances d. The result is a vector of seven different predictive scores based on different models of interactions between the residues in the coiled coil. Two types of these PairCoil scores are computed for each of these interaction distances. Dimeric PairCoil scores are computed from the residue probabilities estimated from the dimeric database, and trimeric PairCoil scores are computed from the residue probabilities estimated from the trimeric database. Thus, MultiCoil computes a 14-dimensional score vector composed of seven dimeric scores and seven trimeric scores for each residue. These 14 scores are used to classify the residue as trimeric, dimeric, or as a non-coiled coil by assigning a probability to each.

3.1.3 Converting Scores to Probabilities

To convert a set of scores into probabilities, the scores for each class were assumed to be Gaussian distributed in each of the 14 dimensions. The expected range of scores for the data classes is therefore characterized by
14 means and by the 14x14 covariance matrix of a multivariate Gaussian distribution. Let \( \mu_{\text{class}} \) denote this vector of the 14 means for each class, and let \( \mu_{\text{class}}(k) \) denote the \( k \)th entry in this vector (the mean along the \( k \)th scoring dimension). Similarly, let \( \Sigma_{\text{class}} \) be the covariance matrix for the class, and let \( \Sigma_{\text{class}}(k_1, k_2) \) be the covariance between the scores from dimensions \( k_1 \) and \( k_2 \). These Gaussian parameters were experimentally determined by running the MultiCoil scoring program on the databases of known sequences for each of the two-stranded, three-stranded, and PDB-minus databases discussed in section 3.1.1. The means and the covariances of the MultiCoil scores on each database define the Gaussian for that data set.

The positive databases of dimers and trimers were handled specially when computing the Gaussian parameters in order to avoid biased scores, since the program uses probabilities estimated from these databases to score the sequences. Each residue in a sequence from a positive database was scored in two different ways, and these scores were averaged in order to obtain a final “unbiased” score. The sequence was first scored using a probability table created by removing that single sequence from the database. The sequence was then scored again using a significantly weakened probability table in which the entire family of sequences with which the test sequence was aligned in creating the database was also removed. This method was used in order to maintain the characteristics of the residue probabilities computed from the database, while making the test sequence score in a fashion similar to a sequence “unknown” to the database. For the dimers the families are described in section 2.2. For the trimers the families were: laminin A chains (including alpha chains and merosin); laminin B1 chains (including beta and S laminins); laminin B2 chains (including gamma); fibrinogen alpha chains; fibrinogen beta chains; fibrinogen gamma chains; and each of the other individual sequences described previously in the section “The Databases”.

Having fixed the Gaussians for each of the three data sets, an arbitrary score vector can be classified into one of these three classes of dimeric, trimeric, and non-coiled-coil structures. When run on an arbitrary sequence, the MultiCoil program computes a 14-tuple of score values \( \mathbf{x} \) for each position in the sequence. By standard statistical analysis (51), the probability that score \( \mathbf{x} \) belongs to each class is computed as follows. Let \( v_i(\mathbf{x}) \) denote the value at the score vector \( \mathbf{x} \) of the multivariate Gaussian determined for class \( i \) (where the classes range over \( i = \text{dim} \) for dimers, \( i = \text{trim} \) for trimers, and \( i = \text{non} \) for non-coiled coils). This value is given by the matrix computation:

\[
v_i(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i)},
\]

where \( n = 14 \) is the number of score dimensions, \( |\Sigma_i| \) is the determinant of \( \Sigma_i \), and \( y^T \) is the transpose of vector \( y \).
Conditional probability analysis gives that, for a score vector \( x \), the probability of being in particular class \( C \) is obtained by taking a \( \delta \) neighborhood \( x_{\delta} \) around the score:

\[
Pr(C|x) = \frac{Pr(C, \text{score } \in x_{\delta})}{Pr(\text{score } \in x_{\delta})} = \frac{Pr(C) \cdot Pr(\text{score } \in x_{\delta}|C)}{\sum_{i \in \text{dim,trim,non}} Pr(\text{score } \in x_{\delta}|i)}
\]

Taking the limit as \( \delta \) goes to zero gives that the probability of being in class \( i \) is the fraction of the total Gaussian value from all three classes that is contributed by that particular class, where the Gaussian for each class is weighted by an initial probability of being in that class. For these initial probabilities, let \( P_i \) denote the probability that a residue chosen at random from any protein sequence lies within class \( i \). Then the total Gaussian weight for a score \( x \) is:

\[
\text{total-gauss}(x) = P_{\text{dim}} \ast v_{\text{dim}}(x) + P_{\text{trim}} \ast v_{\text{trim}}(x) + P_{\text{non}} \ast v_{\text{non}}(x)
\]

This gives:

\[
Pr[\text{x belongs to class } i] = \frac{P_i \ast v_i(x)}{\text{total-gauss}(x)}
\]

Therefore, to convert scores to probabilities, estimates for the initial probabilities that a random protein residue is in a dimeric or trimeric coiled coil are needed (i.e. estimates for \( P_{\text{dim}} \) and \( P_{\text{trim}} \)). Note that for simplicity it is assumed that each residue can be classified as dimeric, trimeric, or non-coiled coil. This means that the fraction of non-coiled-coil residues is \( P_{\text{non}} = 1 - P_{\text{dim}} - P_{\text{trim}} \). Making this assumption simply means that other oligomerization states of coiled coils will be classified into one of these three classes.

### 3.1.4 Estimating the Fraction of Residues that are Dimeric and Trimeric

The values of \( P_{\text{dim}}, P_{\text{trim}}, \) and \( P_{\text{non}}, \) along with the three Gaussians, define a Gaussian mixture distribution on the space of score vectors. The Gaussians have already been fixed by fitting them to the MultiCoil scores on the known databases. Varying the initial probabilities \( P_{\text{dim}}, P_{\text{trim}}, \) and \( P_{\text{non}}, \) changes the distribution these Gaussians define on all residue scores. A second distribution on the score vectors is defined by running the MultiCoil scorer on all residues in all protein sequences. The two distributions should be fairly similar when \( P_{\text{dim}} \) and \( P_{\text{trim}} \) are fixed at their actual values. To this effect, the two unknown parame-
ters $P_{\text{dim}}$ and $P_{\text{trim}}$ are estimated in an unbiased manner using a maximum log likelihood analysis (45). The MultiCoil scoring program was run on the OWL database (release Dec 17, 1996) (16, 2) of proteins in order to approximate the distribution of score vectors on all proteins. The likelihood of a class $i$ given a score $x$ is the density of the score distribution for that class at $x$ (i.e. $v_i(x)$). Letting $P_{\text{dim}}$ and $P_{\text{trim}}$ vary, the two distributions agree best when the total log likelihood for the OWL scores is maximized, where the total log likelihood is given by:

$$\sum \log(P_{\text{dim}} \cdot v_{\text{dim}}(x) + P_{\text{trim}} \cdot v_{\text{trim}}(x) + P_{\text{non}} \cdot v_{\text{non}}(x))$$

and the sum is over all score vectors $x$ for residues in OWL. An approximate maximum was obtained by computing the log likelihood for all values of $P_{\text{dim}}$ ranging between 0 and 0.045, with steps of size 0.0015, and for all $P_{\text{trim}}$ values ranging between 0 and 0.03, with steps of size 0.001.

### 3.1.5 Coiled-Coil and Sequence Probabilities

The methods presented in the previous sections compute probability predictions for each residue in the sequence. In general, residue probabilities can vary significantly over the length of a predicted coiled coil. It is desirable to combine the information into one score for the whole coiled coil or sequence. Coiled-coil probabilities and sequence probabilities are defined here in order to automate the decision process of classifying sequences as dimeric, trimeric, or non-coiled coil. The first step in this process involves defining when the residue probabilities indicate that the residue is likely to be in a coiled coil. To this effect, a bound between 0 and 1 is set so that residues with total coiled-coil probability above the bound are marked as coiled coil (where the residue's total coiled-coil probability is the sum of its dimeric and trimeric probabilities).

The coiled-coil scores are computed as follows. All residues with a predicted coiled-coil total probability above the bound are marked. Contiguous regions of marked coiled-coil residues are grouped as one coiled coil. Weighted averages for the dimeric and trimeric residue probabilities across the whole coiled coil are then computed to give the entire coiled coil a single dimeric and a single trimeric probability. The weight for a residue is taken to be the residue's total coiled-coil probability. The weighted average causes the strongest predicted coiled-coil residues to have the most influence on the predicted oligomerization state for the entire coiled coil. Formally, the dimeric probability for the entire coiled coil is computed from the MultiCoil probabilities as:

$$\frac{\sum_{\text{res in coil}} P_r[\text{res is coiled-coil}] \cdot P_r[\text{res is dimeric}]}{\sum_{\text{res in coil}} P_r[\text{res is coiled-coil}]}$$
and the trimeric probability is:

$$\frac{\sum_{\text{res in coil}} P_r[\text{res is coiled-coil}] \cdot P_r[\text{res is trimeric}]}{\sum_{\text{res in coil}} P_r[\text{res is coiled-coil}]}$$

Classification probabilities for the entire sequence are computed similarly. That is, the weighted average of all sequence residues above the bound is taken in order to compute a single dimeric probability and a single trimeric probability for the entire sequence. Taking the ratio of the trimeric (or dimeric) probability to the total coiled-coil probability gives the probability that the sequence is trimeric (or dimeric), assuming that the residues which score above the bound are, in fact, part of a coiled coil. This ratio represents a probability for differentiating trimeric coiled coils from dimeric coiled coils, and is called the trimeric (or dimeric) oligomerization ratio. In the following sections, care should be taken not to confuse the trimeric (or dimeric) oligomerization ratio with the trimeric (or dimeric) probability. The oligomerization ratio is a useful device for classifying coiled coils as dimeric or trimeric, while the probability gives a measure of the strength with which the coiled coil is located.

### 3.1.6 Using Fewer Dimensions for Improved Performance

By scoring along a subset of the 14 dimensions with Gaussian sub-matrices along those dimensions, the conversion of scores to probabilities by MultiCoil can be tailored more specifically to the biological problem being considered, namely separating dimeric coiled coils from trimeric coiled coils and separating both of these classes of sequences from the non-coiled-coil sequences. In addition, cutting the number of dimensions scored increases the speed of the program and decreases the effects of over-fitting to spurious statistical data. Scoring distances were chosen based on two criteria. Each scoring dimension defines a one-dimensional Gaussian distribution of scores on the three structural classes. In one objective sense, the dimensions that have the least area of overlap between their Gaussians are the best dimensions for correctly classifying sequences. The Gaussian overlaps were examined pairwise in order to determine which dimensions were best for distinguishing each pair of structural classes (Table 3.1). The overlap between two Gaussians was approximated by solving for the intersection between the two Gaussian equations. The area under the tail of one of the Gaussians to the left of this point, and under the tail of the other Gaussian to the right were used to approximate the overlap. The area under the tail of a Gaussian with parameters $\mu$ and $\sigma^2$ from $x$ to $\infty$ was solved as:

$$\frac{1}{\sqrt{2\pi}\sigma} \int_x^\infty e^{-\frac{(y-\mu)^2}{2\sigma^2}} \, dy.$$
Changing variables by \( \frac{z^2}{2} = \frac{(y-x)^2}{2\sigma^2} \) gives that the integral is equivalent to:

\[
\frac{1}{\sqrt{2\pi}} \int_{z=\pm\sigma}^{\infty} e^{-z^2/2} dz.
\]

This is the probability that a normal \( N(0, 1) \) variable is greater than \( \frac{x-\mu}{\sigma} \), which can be computed as \( \frac{1}{2} (1 - \text{erf}(\frac{x-\mu}{\sigma}/\sqrt{2})) \), where \( \text{erf} \) is the error function.

A second empirical measure of the best score dimensions was obtained by examining the quality of classification of the coiled-coil regions from the known data sets when running the \texttt{MultiCoil} algorithm using only that one distance as a scoring dimension. Both measures picked out distances 3, 4, and 5 as the “best” distances for the dimeric scoring dimensions, and distances 2, 3, and 4 as the “best” distances for the trimeric scoring dimensions. These dimensions were used to obtain the results reported here.

### 3.2 Results

#### 3.2.1 Scoring Dimensions

For each pair of Gaussians fit to the scores on the structural classes, Table 3.1 lists the three most relevant scoring distances that have the smallest (best) Gaussian overlap, which were used as the scoring distances for obtaining the results in this thesis. The magnitude of the overlap provides a measure of which scoring dimensions best capture the statistical patterns leading to the formation of coiled-coil structures. For distinguishing between the scores on the two-stranded database and the scores on the PDB-minus, dimeric scoring dimensions are most relevant. For distinguishing the three-stranded database from the PDB-minus, trimeric scoring dimensions are most relevant. For distinguishing two-stranded coiled coils from three-stranded coiled coils, both dimeric and trimeric scoring dimensions are relevant. Distances 3 and 4 had the smallest overlap between the two Gaussians for either coiled-coil database and the non-coiled-coil data. Additionally, for distinguishing dimers from trimers, distance 5 had a small overlap when using the dimeric table, and distances 2 was best for the trimeric table. These distances have biological relevance: distances 3 and 4 represent a to d and d to a interactions essential for coiled coil formation, while distances 5 and 2 may be useful for distinguishing trimers from dimers based upon g to e and e to g dependencies. These distances were chosen as the best distances for simultaneously locating coiled coils and distinguishing dimers from trimers.
3.2.2 Score Distribution of Dimers, Trimers, and Non-coiled-coil Sequences

The best fit to the multivariate Gaussian clusters using the maximum log likelihood analysis and the subset of scoring dimensions discussed earlier resulted when 1.5% of all residues were assumed to be dimeric and 0.9% were assumed to be trimeric. Running on the PIR (release 38.09, 1994) gave comparable estimates of 2.25% dimeric and 0.8% trimeric. A three-dimensional surface representing the magnitude of the log likelihood versus the values for $P_{dim}$ and $P_{trim}$ is shown from various perspectives in Figure 3-1. The estimates for $P_{dim}$ and $P_{trim}$ agree well with the estimate given in Berger et al. (11) that 1 in every 50 residues is in a coiled coil, and the estimate of Lupas et al. (64) that 1 in every 30 residues is in a coiled coil. These percentages are not exact; they are just the best estimates given the data and the scores from MultiCoil. The data is subject to forms of error. First, the data sets (OWL and PIR) which were scored as approximations of the set of all proteins are by no means complete and may be biased. Second, the databases used to represent dimers and trimers for the residue and pairwise probabilities are not at all complete. Additional data (especially in the case of trimers) could easily affect the results. Nevertheless, given the limited data currently available, these estimates are reasonable and informative.

3.2.3 Separation of Data Sets and Classification

The overlap in the residue scores for the known dimers, trimers, and non-coiled coils when plotted in a two-dimensional score space of dimeric PairCoil scores versus trimeric PairCoil scores is shown in Figure 3-2. This two-dimensional plot was used because of the difficulty of visualizing the actual 14-dimensional space. Each sequence from a coiled-coil data set was removed from the probability table for that data set before scoring in order to avoid bias. A cursory examination reveals that the scores for each data set fall into distinct clusters that have a Gaussian-like density.

3.2.4 Classification of the Known Databases

The MultiCoil oligomerization ratio score with a bound of 0 correctly classified all sequences in the coiled-coil databases as dimeric or trimeric. Figure 3-3 shows the distribution of the trimeric oligomerization ratios for the sequences in the two databases, as well as the total coiled-coil probability for each sequence. Each sequence was removed from its probability table before scoring. Classifying all oligomerization ratios under 50% as dimers, and all oligomerization ratios above 50% as trimers, the separation between the data sets is perfect. The worst (highest) trimeric oligomerization ratio by a dimer is F1;A27040: Chicken Neurofil-
ament triplet M protein, which scored 58% dimeric probability and 19% trimeric probability (25% trimeric oligomerization ratio). The worst (lowest) oligomerization ratio by a trimer was L25541: Human S laminin, which scored 30% dimeric probability and 38% trimeric probability (56% trimeric oligomerization ratio). In addition, the probabilities for the PDB-minus sequences are also plotted. The highest total coiled-coil probability from the PDB-minus was 33% (17% dimeric probability, 16% trimeric probability) by 1LE2: Lipoprotein. The other scores above 20% from the PDB-minus were: (1DPI: Nucleotidyl Transferase, 6% dimeric probability, 22% trimeric probability); (1HSC: Hydrolase, 4% dimeric probability, 21% trimeric probability); (1CSG: Cytokine, 0% dimeric probability, 22% trimeric probability). The structure of lipoprotein has been shown to include a four-α-helical bundle, which bears structural similarities to coiled-coils, but is not a coiled-coil (95). All dimers scored above the total probability predicted for lipoprotein. From the trimeric database, all but the lamprey fibrinogen gamma strand, the heat shock factor proteins, and chicken fibrinogen beta strand scored above lipoprotein. The lamprey fibrinogen strands align only moderately well with the other fibrinogens in the database, which may account for the moderate scores of lamprey fibrinogen. Heat shock factor proteins are also odd, and it has been hypothesized that a number of heat shock factor proteins can form both homo-trimeric coiled coils and hetero-dimeric coiled coils (79). The scores for the trimers scoring below 33% and their total coiled-coil probabilities were: K. lactis heat shock factor: 19% (1% dimeric probability, 18% trimeric probability); S. cerevisiae heat shock factor: 17% (3% dimeric probability, 14% trimeric probability); Lamprey fibrinogen gamma chain: 11% (1% dimeric probability, 10% trimeric probability); and Chicken fibrinogen beta chain: 6% (1% dimeric probability, 5% trimeric probability). All other sequences scored above 50% total coiled-coil probability, except for: X. laevis gamma fibrinogen: 48% (6% dimeric probability, 42% trimeric probability); Lamprey fibrinogen alpha2 chain: 35% (6% dimeric probability, 29% trimeric probability); Lamprey fibrinogen beta chain: 33% (12% dimeric probability, 21% trimeric probability).

Thus, using a cutoff of 50% on the total coiled-coil probability, there are no "false positive" sequences and no "false negatives" in the dimeric data set, and only four trimeric sequences which score below the highest scoring negative (33%). All of the coiled coils in the database have a non-zero predicted chance of forming a coiled coil (with the lowest probability at 6%). For the 253 sequences in the PDB-minus database, 201 scored 0% coiled-coil probability, with 236 scoring below 6%, and all but four scoring below 20%. It is expected that the performance will improve even more as other three-stranded coiled-coils are discovered and the three-stranded database is improved in quality and size.
3.2.5 MultiCoil on “Unknown” Test Data

The MultiCoil program was run on the envelope, spike, and glycoproteins obtained from the PIR and GenPept [a translated version of GenBank (release 73, September 1992)], as well as on a set of dimeric coiled coils with the leucine-zipper motif (50). A bound of 0.5 on the maximum residue coiled-coil probability over the sequence was used as a cutoff for finding coiled coils. Figure 3-4 shows the distribution of the sequence probabilities using the 0.5 cutoff (note that a sequence probability can be below 0.5 even though the maximum scoring residue is above 0.5). Using the 0.5 cutoff, 103 out of 1013 sequences in the envelope protein database were found as coiled coils (83 of which were classified as trimeric) and 41 out of 53 sequences were found from the dimeric database (39 of which were classified as dimeric). Using a cutoff of 0.1, 191 of the envelope proteins and 50 of the dimers were found.

It has been hypothesized that many of the envelope proteins are trimeric coiled coils. The distribution of MultiCoil scores supports this hypothesis. In general, the dimeric coiled coils clearly score as dimeric. It is of interest that as the total coiled-coil probability for the sequence increases, the clustering of the envelope proteins tends to be more in the trimeric region, while the two-stranded data set tends to cluster more tightly in the dimeric region of the space. It appears that misclassification occurs more often amongst coiled coils that are only weakly predicted. A few of the predicted coiled-coils in the envelope proteins along with predicted sequence probabilities and the region which scored above 50% include: (P1;VCLISA env polyprotein - simian AIDS retrovirus SRV-1, 11% dimeric, 67% trimeric, residues 427-471); (P1;VCMVCB env polyprotein - Cas-Br-E murine leukemia virus, 2% dimeric, 87% trimeric, residues 502-544); (P1;VGNZRL fusion glycoprotein - rinderpest virus (strain L), 8% dimeric, 59% trimeric, residues 135-167);

3.2.6 Predictions for Protein Sequences Derived from the GCN4 Leucine Zipper

Figure 3-5 gives an example of the residue probabilities across the yeast GCN4 sequence as displayed by the MultiCoil program. GCN4 forms a two-stranded coiled coil in its final 30 residues (70, 42), and is located as a two-stranded coiled coil by MultiCoil with total sequence probability of 80%. A number of mutations in the residues in the a and d positions of the GCN4 leucine zipper have been shown to change the preferred oligomerization of the coiled coil (42). Table 3.2 lists several of these mutations, the actual oligomerization state of the sequence, and the coiled-coil probability predicted by the MultiCoil program, along with the predicted trimeric oligomerization ratio. While none of the GCN4 mutations scores exceedingly strongly in any oligomerization state, using 50% as a cutoff, they are all classified correctly, excluding the tetrameric coiled coil. The a→V, d→L mutation, which forms both trimers and dimers, scores at similar strengths as
both a trimer and a dimer.

3.3 Discussion

3.3.1 Performance Compared to PairCoil

Since the MultiCoil scorer uses the PairCoil scorer as a sub-process, as expected, many of the regions found by PairCoil (with the dimeric database) are also found by MultiCoil. In fact, the probabilities for regions predicted by PairCoil showed striking agreement with the total residue probabilities predicted by MultiCoil, despite the fact that the two programs compute probabilities from scores in radically different ways. In general, the trimeric scorer from MultiCoil does not cause the program to miss coiled coils already found by PairCoil. Instead, it generalizes the program and allows even more coiled coils to be located and classified as dimeric or trimeric. Some examples of coiled coils which have similar MultiCoil and PairCoil predictions follow. The format of the listing is (name, MultiCoil total probability (trimeric oligomerization ratio), PairCoil probability, residue positions in the sequence). (P1;J04111 Human JUN, 93.1% (9%), 89.9%, 267-315); (P1;U12918 Human syntaxin, 74.4% (39%), 80.3%, 44-76). The scores for MultiCoil are significantly higher for some proteins: (P1;VGNZRL fusion glycoprotein - rinderpest virus (strain L), 67% (88%), 0%, 135-167); (P1;A38561 alpha-1,3-mannosyl-glycoprotein beta-1,2-N-acetylglucosaminyltransferase (EC 2.4.1.101) - rabbit, 60% (80%), 31.6%, residues 49-77). The NEWCOILS scores on these last two proteins are 71.8% and 24% respectively.

3.3.2 Differences Between the Datasets

The distribution of scores in figure 3-2 indicates that the separation between the PDB-minus and the dimers is better defined than the separation between the scores for the PDB-minus and the trimers. This observation supports the theory that trimers have fewer constraints enforced on them than dimers, because they have more packing freedom than dimers (97). An additional factor which contributes to this overlap is the small number of sequences which have been shown to contain three-stranded coiled coils, resulting in the small size of the three-stranded database. Hence, the trimers are more difficult to distinguish from non-coiled coils than dimers.

In addition, the scores for both coiled-coil data sets along the y-dimension of Figure 3-2 (representing PairCoil scores using the trimeric table) range over roughly the same values. The similar range of scores indicates that many of the properties needed to form dimers are captured by the statistical distribution of the
Table 3.1: The table shows the three most relevant scoring dimensions for distinguishing the pairs of data classes using the MultCoil program. A scoring dimension is determined by the table (dimeric or trimeric) and the scoring distance (1-7) used. This table was used to determine three distances for each table, which were used to compute the results of this chapter. Using only these distances reduces the problem of over-fitting to non-structural statistical features of the databases. The relevant distances had the smallest pairwise overlap of the Gaussians fit to the scores from that scoring dimension on the databases. Since it is to be expected that the dimeric scoring dimensions are most relevant to the two-stranded database and that the trimeric scoring dimensions are most relevant to the three-stranded database, the scoring dimensions were considered accordingly. The distances are listed in increasing order of overlap between the two Gaussians.

<table>
<thead>
<tr>
<th>Data Sets to Distinguish</th>
<th>Two-stranded</th>
<th>Three-stranded</th>
<th>Three-stranded</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Database</td>
<td>Database and</td>
<td>Database and</td>
</tr>
<tr>
<td></td>
<td>PDB-minus</td>
<td>PDB-minus</td>
<td>Two-stranded</td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td>Database</td>
<td>Database</td>
</tr>
<tr>
<td>Scoring Dimension</td>
<td>Dimeric</td>
<td>Trimeric</td>
<td>Dimeric Distances</td>
</tr>
<tr>
<td>with the Smallest</td>
<td>Distances</td>
<td>Distances</td>
<td>3, 5, and 4 and 6</td>
</tr>
<tr>
<td>Overlap Between Gaussians Fit to Data Set Scores</td>
<td>4,3, and 1</td>
<td>4,3, and 2</td>
<td>Trimeric Distances</td>
</tr>
</tbody>
</table>

Table 3.2: The MultCoil predictions for various mutations of GCN4:Leucine Zipper versus the actual oligomerization states. The first column gives the residues that were placed in registers a and d in the leucine-zipper region of the mutated proteins. The second column gives the actual oligomerization state of the protein. The third column gives the sum of the MultCoil dimeric and trimeric predicted probabilities, representing the confidence with which the sequence is predicted to form a coiled coil. The final column of trimeric oligomerization ratios represents how confidently the program predicts the sequence to form a trimer, given that it forms a coiled coil. Values near 0% are dimeric predictions, and values near 100% are trimeric predictions.

<table>
<thead>
<tr>
<th>Substitution in last 30 residues:</th>
<th>Oligomerization</th>
<th>Total MultCoil Probability</th>
<th>Trimeric Oligomerization Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unmodified</td>
<td>dimeric</td>
<td>.80</td>
<td>18%</td>
</tr>
<tr>
<td>a→ I, d→ L</td>
<td>dimeric</td>
<td>.88</td>
<td>23%</td>
</tr>
<tr>
<td>a→ I, d→ L</td>
<td>trimeric</td>
<td>.27</td>
<td>81%</td>
</tr>
<tr>
<td>a→ L, d→ I</td>
<td>tetrameric</td>
<td>.54</td>
<td>76%</td>
</tr>
<tr>
<td>a→ V, d→ L</td>
<td>forms trimers and dimers</td>
<td>.81</td>
<td>31%</td>
</tr>
<tr>
<td>a→ L, d→ V</td>
<td>trimeric</td>
<td>.76</td>
<td>70%</td>
</tr>
</tbody>
</table>
three-stranded coiled-coil data set. However, the reverse is not true. The scores for the trimeric data set along the z-dimension (representing PairCoil scores using the dimeric table) overlap significantly with the non-coiled-coil scores, and are noticeably lower than the scores for the two-stranded coiled coils. Thus, the dimeric table fails to pick out some of the statistical features which are important for distinguishing three-stranded coiled coils from non-coiled-coil sequences. This is consistent with the fact that a number of coiled coils in the three-stranded database are not located or are not located in the correct register by either the PairCoil or the NEWCOILS programs.

3.3.3 Improving Performance

The best way to improve the performance of the MultiCoil program does not involve changing the program, but instead, expanding the trimeric data set. The three-stranded coiled coil database consists of only 6319 residues, a small fraction of the size of the 58191 residue two-stranded database. The current lack of a large database of trimeric coiled coils limits the accuracy of the statistical information that can be extracted from the database. 6319 is a small sample of residues for estimating the pairwise residue probabilities with high accuracy, since there are only approximately $\frac{6319}{7} = 903$ samples at each heptad-repeat position. These sample are spread across the 400 possible sample events (i.e., the occurrence of a particular pair of amino acids in the database). This under sampling tends to lower the scores of sequences containing under sampled amino-acids. These low scoring sequences also adversely affect the probability computations, as they create overlap between the Gaussians fit to the PDB-minus and the trimeric database scores. As more trimeric coiled-coil structures are discovered, their addition to the database should improve the separation between the PDB-minus and the trimeric data set.

3.4 An Alternative Version of the MultiCoil program

Recall that the MultiCoil predictions are based upon the multi-dimensional score vector composed of the maximum scoring heptad-repeat positions from each of the scoring dimensions. These maximum scores are not constrained to be from the same predicted heptad-repeat positions. This allows for multiple predicted registers to combine and strengthen the overall prediction. This method is referred to as the "Combined Register Method". Another approach which was implemented with the MultiCoil program involves first converting the MultiCoil scores into trimeric and dimeric probabilities for each of the possible heptad-repeat positions. The register with the maximum trimeric or dimeric probability is then taken to be the predicted
probability and register for the residue. Call this method the "Maximum Register Method".

The computations used by the program proceed without any changes from the Combined Register Method. However, slight modifications are needed in order to compute the Gaussian parameters for the conversion of scores into probabilities. The Maximum Register Method is based upon choosing the register with the maximum predicted probability, but this cannot be done until after the Gaussian parameters have been fixed. The choice of which register scores to fit the Gaussian distribution to must therefore be reconsidered. For the positive data sets of two-stranded and three-stranded coiled coils the choice is straightforward. The Gaussians are fit to the scores computed for the actual heptad-repeat positions available from the coiled-coil databases.

For the PDB-minus database the choice is trickier. Two options were implemented. In the first, the Gaussian was fit to the score vectors composed of the maximum heptad-repeat position score from each dimension. Thus, the Gaussian was identical to the Gaussian from the Combined Register Method. The justification for this method is that the maximum score vector represents the worst possible outcome for classifying the residue as a non-coiled coil. The disadvantage of this method is that using the worst possible score vector tends to raise the scores in the distribution from the scores that would actually occur for the PDB-minus in the maximum probability register. As a result, this method could result in false negative predictions on coiled-coil test data. In fact, using this method led to the location of only 79 sequences from the envelope proteins using a 50% bound, as opposed to the 103 sequences located using the Combination Register Method.

A second approach involves fitting a Gaussian to the distribution of scores determined by the score vectors for all heptad-repeat positions of each residue in the PDB-minus. One disadvantage of this method is that the inclusion of all heptad-repeat position scores tends to lower the overall scores of the distribution from the scores that would arise from the maximum probability register. This could possibly have the adverse affect of increasing the chances of false positives. These two methods are referred to as Maximum Register Method 1 and 2 respectively. The separation between the sequence scores of the data sets for the Maximum Register Methods 1 and 2 using phantom zero frequency 1/3 are shown in figures 3-6 and 3-7 respectively. Note that because the predicted trimeric register can differ from the predicted dimeric register, the sum of the two predictions can be greater than 100%. For such sequences there are multiple registers which have different predicted oligomerization states. As expected, the Maximum Register Method 2 led to more false positive predictions. The scores of the Maximum Register Method 1 on the envelope proteins and Leucine Zipper proteins is also shown in figure 3-8. 40 of the leucine zipper motif sequences and 79 of the envelope proteins were located when using a bound of .5. A two dimensional plot of the residue scores for the Maximum Register Method 2 is shown in figure 3-9. The lower scores for the PDB-minus can clearly be seen by comparison with figure 3-2. The performance of the methods suggest that they may be worthy of further testing and optimizing.
Figure 3-1: Plots of the log likelihood surface from various perspectives. The color is proportional to the surface height (red is the highest point), and represents the total log likelihood (minus the maximum value) for values of $P_{\text{dim}}$ and $P_{\text{trim}}$. The upper left panel shows the entire surface, while the other three panels project the surface onto each of the three coordinate planes. The upper right panel indicates that the actual value for $P_{\text{dim}}$ probably lies somewhere between 0.01 and 0.025. The lower left panel suggests that the actual value for $P_{\text{trim}}$ lies between 0.005 and 0.015. The total log likelihood is maximized at $P_{\text{dim}} = 0.015$ and $P_{\text{trim}} = 0.009$. 
Figure 3-2: Distribution of residue scores for the data sets of two-stranded coiled coils (58191 residues), three-stranded coiled coils (6319 residues), and PDB-minus (34940 residues). Scores from the two-stranded database are in blue, scores from the three-stranded database are in red, and scores from the PDB-minus are in green. The x-dimension scores were computed by PairCoil using distances 3, 4, and 5, with probabilities from the two-stranded database, and the score on the y-dimension was computed using distances 2, 3, and 4, with the three-stranded probability table (where the distances were combined using the method of Berger et al. (11)).
Figure 3-3: The separation between sequences in the two-stranded database (X) and sequences in the three-stranded database (O) given by the MultiCoil trimeric oligomerization ratio. The y-axis represents the predicted coiled-coil probability for each sequence. Larger y values indicate greater confidence that the sequence contains a coiled coil. The z-axis represents the trimeric oligomerization ratio for each sequence. Values greater than 0.5 represent trimeric predictions by the program. The further the value from 0.5, the greater the confidence in the prediction. Sequences from the non-coiled-coil dataset PDB-minus(*) are also plotted.
Figure 3-4: Distribution of the predicted sequence probabilities and oligomerization ratios for MultiCoil run on a database of sequences containing the leucine-zipper motif (X) and for the envelope spike proteins and glycoproteins (O), which are hypothesized to include a number of trimeric coiled coils. The y-axis represents the predicted coiled-coil probability for each sequence. The x-axis represents the trimeric oligomerization ratio for each sequence. A bound of 0.5 was used in the computation of these sequence probabilities.

<table>
<thead>
<tr>
<th>GCN4 protein</th>
<th>Dimer Probabilities in Black</th>
<th>Trimer Probabilities in Gray</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq Prob for bound 0.05</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>Dimer 0.66</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>Trimer 0.14</td>
<td>0%</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-5: An example of the residue probabilities as plotted by MultiCoil for the 281 residue yeast GCN4 sequence. The final 30 residues form a two-stranded “leucine zipper” coiled coil. The predicted dimeric probability is plotted in black and the trimeric probability is plotted in gray. The MultiCoil probability predictions for classifying the entire sequence are also shown at the left of the figure, using a bound of 0 in the computations.
Figure 3-6: The separation between sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) given by the MultiCoil program using the Maximum Register Method 1. The "phantom zero frequency" was set to 1/3 for estimating the residue probabilities.
Figure 3-7: The separation between sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) given by the MultiCoil program using the Maximum Register Method 2. The “phantom zero frequency” was set to 1/3 for estimating the residue probabilities.
Figure 3-8: Distribution of the predicted sequence probabilities using the Maximum Register Method 1 for MultiCoil run on a database of the leucine zipper motif (X) and the envelope spike proteins and glycoproteins (O). A bound of 0.5 was used in the computation of these sequence probabilities.
Figure 3-9: Distribution of residue scores using the Maximum Register Method 2 for the two-stranded database (blue), the three-stranded coiled coils (red), and PDB-minus (green). The x-dimension scores were computed by Paircoil using distances 3, 4, and 5 with probabilities from the two-stranded database, and the score on the y-dimension was computed with distances 2, 3, and 4 with the three-stranded probability table (where the distances were combined using the method of Berger et al.(11)).
Chapter 4

Estimating Probabilities from the Databases

4.1 Estimating Residue Correlation Probabilities

Estimating the residue probabilities for the data sets is perhaps the most important step for optimizing the performance of any statistically based prediction scheme. Three factors contribute to these estimates: the quality of the databases used as approximations of the actual data sets, the method used to convert the residue frequencies from these databases into probability estimates, and the effects that the inaccuracies in these estimates have on the program's final predictions. For the dimeric database the quality issue was dealt with by the sheer size of the database. The dimeric database was constructed from 58191 coiled-coil residues drawn from 226 sequences, resulting in on the order of $\frac{58191}{2} = 8313$ sample points for each pair of heptad-repeat positions. The sample size of the trimeric database is unfortunately much smaller. The three-stranded coiled-coil database was constructed from 6319 coiled-coil residues drawn from 42 sequences. The database is nearly a factor of ten smaller than the dimeric database.

In optimizing the parameters approximated from the databases, it is essential to consider that the databases are not "random samples" of coiled-coil proteins. Instead, they are biased by the current biological knowledge of coiled-coil structures. This knowledge is intrinsically linked with protein function and previous discoveries.
Entire families of coiled-coil proteins may be missing from the database, simply because they have not yet been discovered.

Many of the coiled-coils in the two-stranded database were located by alignment with other prototypical coiled coils. For the two-stranded database, 12 prototypical proteins divide the database into 12 families. The repetition within a family leads to reenforcement of the factors which are maintained throughout the family, and are most important to the formation of the coiled coil, while diluting some spurious data. The inclusion of aligned sequences also cuts down on the number of “false zero frequency” events. The greater number of samples result in better estimates of which events are structurally significant (either by their absence, or their high frequency counts). However, the inclusion of multiple versions of the same protein within a family can lead to over sampling of structurally unimportant residues common to that family, leading to higher probability estimates for those residues, and lower estimates for other residues from other under sampled families.

The current biological knowledge of three-stranded coiled-coils is sparser than that of two-stranded coiled-coils. However, just as many “families” of proteins were gathered for both databases (namely 12). Since only 42 sequences compose these families in the trimeric database, it is clear that the problems of oversampling the “common” non-structural events is not nearly as significant. Instead, it is the small frequency events which tend to be most problematic. Each occurrence of an event carries more weight, and so, non-structural events gain importance with just one appearance in the database. “False zero frequency” samples are also more common with the small database and can have devastating effects on the scoring algorithms. The estimation of the residue probabilities must be tailored to the database and tailored to the program performance. The goal faced by the prediction program is not to obtain the “best” probability estimates for the events in the data sets, but to obtain estimates which have the least detrimental effects on the performance of the program.

Woolfson and Alber (97) experienced similar problems with their dimeric and trimeric databases, with low frequency events initially dominating their scoring method. To compensate, they limited the terms which could contribute to their score to be residues in the a, d, e, and g heptad-repeat positions. Amino acids which were “rare” in either of the databases or which had similar observed probabilities in the dimeric and trimeric databases were also excluded. Excluding these terms from their score can alternatively be implemented by adjusting the estimates of the probabilities so that the excluded events have the same estimated probability for each database, and therefore contribute nothing to the score. In effect, the sample database is only used to estimate the probabilities of events which are deemed to be accurately sampled and structurally important.

For PairCoil, “false zero frequency” events are the most problematic, as setting the probability of an event to zero completely zeros out the probability predicted for any subsequence containing that event to form a coiled coil. In the PairCoil implementation, each zero frequency event was given a phantom frequency
of 1/5. The probabilities were then computed by dividing by the total adjusted frequency count over all events.

4.2 Formal Methods for Estimating Probabilities from a Sample

The problem of estimating parameters can be stated probabilistically. Given a prior distribution on the value of the parameter before making any samples, and given the outcomes of the sample, an a posteriori distribution on the parameter given the samples is desired. This framework lends itself to using Bayesian conditional probability analysis to compute the a posteriori distribution, using the mean of the distribution as the estimate for the parameter. The parameter being estimated is the probability \( p \) of a particular pair of residues occurring in a pair of heptad-repeat positions, given that there were \( x \) occurrences of the pair in the database out of \( n \) samples.

Sampling from the space of all coiled coils is just a binomial process, where the event occurs with probability \( p \). For large \( n \), the distribution on the observed probability of a hit is approximated by a normal distribution around mean \( p \) with variance \( \sigma^2 = p(1-p)/n \). To simplify the analysis, it is now assumed that the prior distribution on \( p \) is also normally distributed around an initial guess \( p_0 \), with variance \( \sigma_0^2 \). The final distribution (33, 14) on \( p \) obtained using conditional densities to calculate the distribution on \( (p|\{ x \text{ hits in } n \text{ samples} \}) \) is again a normal distribution with mean:

\[
p_n = \frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2} \frac{x}{n} + \frac{\sigma^2}{n\sigma_0^2 + \sigma^2} p_0.
\]

This a posteriori estimate for \( p \) is a weighted average of the observed probability \( x/n \) and the a priori probability \( p_0 \). Letting \( \sigma_0^2 = k\sigma^2 \), the weight on the observed probability is \( nk/(nk + 1) \).

The factor \( k \) represents a measure of the relative confidence between the probability obtained from the sample database and the a priori initial probability estimate. If \( k \) is small, then the a priori probability distribution has relatively small variance, giving it more weight in determining the final a posteriori probability estimates. In a sense, \( k \) can be used to factor into the computation a measure of how "good" the database is which is not captured merely in the size of the database. Making \( k \) small can somewhat account for over sampling of high frequency events and under sampling of low frequency events due to the limitations on constructing a biological database.

This framework gives another justification for the PairCoil method of adding a phantom 1/5 frequency to the zero frequency events. More generally, consider the case where each zero frequency event is given fre-
quency \( f \). Let \( z \) denote the number of zero frequency events in the sample. For the residue pairwise correlations on the two-stranded coiled-coil database \( z \) tends to be around 200, while on the smaller three-stranded database \( z \) tends to be around 250. The initial estimates for the residue probabilities are set to \( p_0 = 1/z \) if the event is a zero frequency sample, and \( p_0 = 0 \) otherwise. Intuitively, this prior distribution formalizes the fact that it is expected that a number of the zero frequency samples will be "false zeroes". These "false zeroes" can only be under samples, since negative frequencies are impossible. The occurrence of false zero frequency samples also means that other events must be oversampled. When the factor \( k \) is set to \( 1/(fz) \), the a posteriori probabilities are identical to the probability estimates computed by PairCoil. As the number of zero frequency events \( z \) and \( f \) get smaller, and as \( n \) gets larger, more confidence is placed in the observed sample data than the prior distribution. In particular, the smaller \( z \) is, the closer it is to the actual number of zero frequency events in all coiled coils. The size of \( z \) therefore provides a measure of the quality of the database. Note also that as \( n \) gets larger \( z \) is expected to get smaller since a better sample is obtained.

The factor \( 1/f \) can be thought of as another adjustment for optimizing the relationship between \( \sigma^2 \) and \( \sigma_0^2 \). Making \( f \) larger accounts for the biases inherent in compiling a database of coiled coils from solved structures. Grouping the factor \( 1/(fz) \) with the factor \( n \) in the a posteriori probability estimate effectively gives a sample of size \( n/(fz) \). As \( f \) is made larger, the confidence in the observed sample is made smaller as more probability is given to the zero frequency events. For the best performance of PairCoil with the two-stranded coiled-coil database the "phantom zero frequency" \( f \) was empirically chosen to be \( 1/5 \).

### 4.3 Other Distributions for the a priori Probabilities

The method for estimating probabilities from the database implemented in PairCoil can also be applied to the databases used by the MultiCoil program. However, the fact that MultiCoil uses PairCoil as a subprocessor does not mean that the probability estimates used for PairCoil are necessarily the best estimates for use with MultiCoil. Inaccuracies in estimates from one database may be "cancelled out" by the other database when the MultiCoil scores are converted into probabilities. For example, when running PairCoil on the two-stranded database, setting the "phantom zero frequency" to \( 1/2 \) rather than \( 1/5 \) resulted in overlap between the scores of the known two-stranded coiled coils and the solved non-coiled-coil structures. In trial runs of MultiCoil, it was found that phantom zero frequencies of \( 1/2 \) could be used to estimate the probabilities from the two-stranded database, without resulting in any non-coiled-coil sequences being falsely predicted as coiled-coils, or any two-stranded coiled-coils being falsely predicted as not forming the coiled-coil structure. The additional dimensions of scores used by MultiCoil for different databases al-
low for different tolerances of inaccuracies in the probability estimates. Figure 3-3 displays the performance of the program's sequence scores when using these probability estimates derived with "phantom zero frequency" 1/3. Figures 4-1 and 4-2 give examples of the effects of using different "phantom zero frequencies" to estimate the probabilities from the databases. The phantom zero frequencies of 1/3 (instead of 1/5) were judged to yield better separation of the databases for the Combination Register Method for MultiCoil.

The "phantom zero frequency" approach is not the only distribution to consider on the prior probabilities. However, the determination of the weighting factor k to compute the a posteriori estimates for other distributions must be judged empirically for each individual distribution to obtain optimal performance. Using the results from the "phantom zero frequency" method as a model of the approximate magnitude of k, k was set to 1/60 in trial implementations with other prior distributions. This value approximates the value of 1/(fz) used in the "phantom zero frequency" computations when f is between 1/4 and 1/3. The first of the natural distributions takes the a priori probabilities p0 to be the GenBank probabilities. Having absolutely no knowledge about coiled coils, this would be the correct distribution. A second natural distribution for estimating the pairwise residue correlations is the product of the two single residue probabilities estimated from the database for the individual residues in the pair. This distribution corresponds to assuming at the outset that each pair of residues is independent. A third natural distribution arises when trying to improve the probability estimates for only the trimeric database used in MultiCoil, since it is much less reliable than the dimeric database. Without any additional knowledge about the structure of trimeric coiled coils, the properties of trimeric coiled coils are expected to be similar to dimeric coiled coils. The plots of the performance of the MultiCoil sequence scores using these various methods are plotted in figures 4-3, 4-4, and 4-5. For these plots, only the trimer pair probabilities were estimated using the new a priori probabilities. All other database probabilities were estimated using phantom zero frequencies of 1/5.

While each of these probability distributions has a natural justification for its choice as the prior probabilities, they each have an underlying disadvantage when used as a priori probabilities with a program such as MultiCoil for locating and distinguishing two- and three-stranded coiled coils. The GenBank distribution is fairly similar to the residue probabilities for all non-coiled-coil sequences. Thus, the non-coiled-coil residue probabilities are used to estimate the coiled-coil residue probabilities, despite the fact that these estimated probabilities are later used by the program to differentiate coiled-coil sequences from non-coiled-coil sequences. A similar problem arises with using the dimeric probability estimates for the prior distribution on the trimeric probabilities, since the program uses these estimated probabilities to differentiate dimeric coiled coils from trimeric coiled coils. The use of the product of the single residue probabilities to estimate the pairwise probabilities also clashes with the theory behind the PairCoil style programs. PairCoil and
MultiCoil attempt to use residue correlations and anti-correlations to differentiate the data sets. Taking the product of the single probabilities corresponds to initially assuming that all residue pairs are independent. These conflicts do not mean that the methods are wrong. They just mean that the weighting factor used to combine the prior probabilities and the observed probabilities must be chosen carefully. Putting too much weight on the prior probabilities can adversely affect performance.

For comparison with figures 4-3, 4-4, and 4-5, a plot of the separation obtained using the observed probabilities without any modifications for the three-stranded pair probabilities is shown in figure 4-6. It is unclear whether any of these methods can be fine tuned to out perform the phantom zero frequency method. More experimentation with the weighting parameter and perhaps combining various priors (for example, combining one of these natural distributions with the “phantom zero frequency” approach) may lead to improved performance, much as the phantom zero frequency improves the plot in figure 4-6 to yield the separation in figure 4-1.

4.4 Accounting for Sample Errors in Other Fields of Research

A number of methods for estimating parameters have been proposed outside of the formal framework provided by a priori and a posteriori distributions. The methods of data smoothing have been well studied in the field of natural language recognition. On the surface, the estimation of parameters in language recognition is extremely similar to the estimation of residue probabilities. In natural language recognition, an extremely large sample database of text (for example, from the Wall Street Journal) is used to estimate “unigram” probabilities (the probability of each word occurring) and “bigram” probabilities (the probability of two words occurring one after the other in a sentence). Much as PairCoil uses conditional probabilities to break the probability of a given sequence forming a coiled coil into a formula of pairwise and single residue probabilities, the probability of an English sentence can be broken into probabilities of bigrams and unigrams. Chen(22) gives an excellent overview of many of the techniques of data smoothing used in language recognition. Unfortunately, the best methods for language recognition are not necessarily the best techniques for estimating the probabilities for coiled-coil datasets due to essential differences in the nature of the databases. The Wall Street Journal database represents a large and fairly random sample of sentences. The coiled-coil databases are inherently biased by the sequences which have drawn enough biological interest to be studied.
4.5 Methods of Data Smoothing

4.5.1 Additive Smoothing

A similar approach to the phantom zero frequency method is additive smoothing. "Plus-delta" smoothing involves adding \( \delta \) to each sample (22). While the performance obtained using plus-delta smoothing to estimate the trimer pair probabilities improved upon the performance using the observed frequencies, it was found to be inferior to the phantom zero frequency method. A plot of the MultiCoil sequence scores using \( \delta = .4 \) is shown in figure 4-7.

4.5.2 Good-Turing Estimates

In the Good-Turing method, the adjusted frequency for any event which has a sample frequency of \( k \) is:

\[
f_{a,i,k} = (k + 1) \frac{n_{k+1}}{n_k},
\]

where \( n_k \) is the number of events with a sample frequency of \( k \). The Good-Turing method redistributes all of the frequency from the \( k + 1 \) frequency events to the \( k \) frequency event. Letting \( p_i \) denote the actual probability of the \( i \)th event, the Good-Turing probability estimates are approximations to \( E(p_i|k \) sample observations) (22). Practical implementations of the Good-Turing method must smooth the \( n_k \) values so that they are all non-zero.

The Good-Turing method can also be applied with "backoff probabilities" in order to adjust the way the probability is redistributed to each of the frequency \( k \) events (21). Letting \( T_k \) denote the total frequency being distributed to events with observed frequency \( k \). Then each \( k \)-event \( e \) gets frequency proportional to its backoff probability \( b_p_e \). Letting \( B P_k = \sum_{\text{events } e \text{ with } b_p_e} \) denote the total backoff probability for \( k \)-events, event \( e \) is given frequency \( T_k \frac{b_p_e}{B P_k} \). In the case of the Good-Turing method, \( T_k = (k + 1)n_{k+1} \). For estimating pairwise residue probabilities, any of the "natural" probability distributions discussed as a priori probability distributions make natural choices for the backoff probabilities.

A modified version of the Good-Turing method was implemented with MultiCoil in which only the observed frequencies less than \( k_{\text{max}} \) were modified, where \( k_{\text{max}} \) is the smallest \( k \) such that \( k \leq \frac{n_{k+1}}{n_k} (k + 1) \leq k + 1 \) does not hold. \( k_{\text{max}} \) is the first \( k \) such that the redistributed frequency would be more than 1 unit different from the sampled frequency. This prevents a sample frequency of \( k \) from obtaining more redistributed frequency than an event with sampled frequency \( k + 1 \). It is of interest that the amount of probability redis-
tributed to the zero frequency elements tended to be comparable to the values from the phantom zero frequency methods. The results of these probability estimates on the MultiCoil program are shown in figure 4-8.

4.5.3 Absolute Discounting

A last method of data smoothing tested was the Ney-Kneser absolute discounting method (53). In this method, a fixed frequency $f$ between 0 and 1 is subtracted from each non-zero frequency event. The total subtracted amount is then added back to every event, weighted by the backoff probability distribution. For the implementation with MultiCoil the backoff probability was chosen to be the uniform distribution. This causes the method to redistribute probability from the non-zero frequency events to the zero frequency events, and is therefore similar to the phantom zero frequency method of PairCoil. However, because a fixed amount is subtracted from each non-zero frequency term, the small frequency samples are affected more notably than in the phantom zero frequency method. Figure 4-9 plots the results of the fixed discount method, where the discount frequency was set to $f = .5$.

4.6 Combining Smoothing Methods

Using a combination of the smoothing methods can improve the program performance. The best performance obtained as measured by sequence scores for MultiCoil was obtained by combining the phantom zero frequency method with additive smoothing. Setting the phantom zero frequency to $1/3$ and using a $\delta = .2$ term for additive smoothing of the trimeric probability estimates resulted in the separation plotted in figure 4-10. Only four coiled-coil sequences in this plot score lower than the highest scoring negative. Combining the phantom zero frequency method with other methods which were presented in this chapter could possibly lead to even better performance.

4.6.1 Multiple Probability Models

Another approach to combining different smoothing methods involves viewing each of the smoothed probability distributions as a possible model of the distribution. Rather than using the algorithm to compute a single prediction score based upon one of these models, instead, a score is computed for each model. These scores are combined in a weighted sum based on the probability that the model is the best one (90). Letting $S(M)$
denote the prediction score from the probability model $M$, the score $S$ would be computed as:

\[ S = \sum_{\text{models } M} S(M)Pr[M]. \]

The disadvantage of the method is that it requires the computation of a score for each model, which can be time consuming. Additionally, the accuracy of the different models must be valued by estimating $Pr(M)$. However, it could be an effective approach for weighting between a small number of models of the probability parameters that have different performance characteristics with the program.

### 4.7 Using only Structurally Important Heptad-Repeat Positions

An attempt at scoring based on only “structurally important” heptad-repeat positions, as used by Woolfson and Alber (97), was also implemented. Probabilities for events involving positions b, c, and f were all set to the corresponding GenBank probabilities for the coiled-coil databases. The phantom zero frequency method was then used to estimate the probabilities for the remaining positions. Positions a, d, e, and g were chosen as structurally relevant since they lie along the inner face of the coiled coil. The performance of the MultiCoil program when using all distances was moderate (figure 4-11). However, the performance was poor when only using distance 3, 4, and 5 for the dimeric scores and distances 2, 3 and 4 for the trimeric scores. Further research is necessary to accurately assess the method. A better measure of the method would be obtained by testing on a data set of true unknowns, as using structurally important positions is a method which is designed to avoid over fitting to the training sets.
Figure 4-1: Separation between the two-stranded database (X), the three-stranded database (O), and the PDB-minus(*) when using "phantom zero frequency" 1/5 in both databases.
Figure 4-2: Separation between the two-stranded database (X), the three-stranded database (O), and the PDB-minus(*) when using "phantom zero frequency" 1/5 in the dimeric databases and 1/3 in the trimeric database.
Figure 4-3: The separation using the GenBank distribution as prior probabilities for estimating the trimer residue correlation probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. The weighting factor $k$ was set to $1/60$. The "phantom zero frequency" was set to $1/5$ for estimating the other database probabilities.
Figure 4-4: The separation using the distribution determined for the two-stranded pairwise probabilities as prior probabilities for estimating the trimer residue correlation probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(∗) are plotted. The weighting factor k was set to 1/60. The “phantom zero frequency” was set to 1/5 for estimating the other database probabilities.
Figure 4-5: The separation using the distribution determined by the product of the single residue probabilities as prior probabilities for estimating the trimer residue correlation probabilities. The sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(∗) are plotted. The weighting factor $k$ was set to 1/60. The "phantom zero frequency" was set to 1/5 for estimating the other database probabilities.
Figure 4-6: The separation using the observed probabilities without any modifications to estimate the trimer residue correlation probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. The "phantom zero frequency" for other probability estimates was set to 1/5 for estimating the other database probabilities.
Figure 4-7: The separation using delta-smoothing with $\delta = .4$ to estimate the trimer residue and residue pair probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. The “phantom zero frequency” for the dimer database was set to 1/3.
Figure 4-8: The separation using the modified Good-Turing method to estimate the trimer residue and residue pair probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. The "phantom zero frequency" for the dimer database was set to 1/3.
Figure 4-9: The separation using the fixed discount method with discount $f = .5$ to estimate the trimer residue and residue pair probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. The "phantom zero frequency" for the dimer database was set to 1/3.
Figure 4-10: The separation using phantom zero frequency 1/3 and additive smoothing with $\delta = .2$ to estimate the trimer residue and residue pair probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted.
Figure 4-11: The separation using phantom zero frequency 1/3 to estimate probabilities at positions a, d, e, and g. All other positions were assigned the corresponding GenBank probabilities. Sequences in the two-stranded database (X), sequences in the three-stranded database (O), and sequences in the PDB-minus(*) are plotted. All scoring distances were used.
Chapter 5

The PairDiff Program: An Alternative Approach to Determining Oligomerization States

In this section the problem of distinguishing two-stranded coiled coils from three-stranded coiled coils is again considered. However, here the coiled coil is assumed to have been located already. Instead of identifying and distinguishing coiled coils, as in the MultiCoil program, the program is given the location and the heptad-repeat positions of a known coiled coil and must classify it as either trimeric or dimeric. This method implemented in the PairDiff algorithm relates to the SCORER program of Woolfson and Alber (97) in the same way that the PairCoil program generalized the NEWCOILS program. Much as the analysis of Berger (10) provided a probabilistic framework for the NEWCOILS program, the analysis presented here provides formal probabilistic justification for the SCORER program, assuming that all residues within a coiled coil are independent.

Mathematically speaking, distinguishing dimeric coiled coils and trimeric coiled coils amounts to computing an estimate of:

$$Pr[\text{region is trimeric}|\text{region is a coiled coil}]$$
Define \( z \) to be the coiled-coil region of the sequence, let \( C_2 \) denote the class of two-stranded coiled coils, let \( C_3 \) denote the class of three-stranded coiled coils, and let \( C \) denote all dimeric and trimeric coiled coils. For simplicity it is assumed that the classes \( C_2 \) and \( C_3 \) are disjoint. Then the probability is computed as:

\[
Pr[z \in C_3 | z \in C] = \frac{Pr[z \in C_3 \text{ AND } z \in C]}{Pr[z \in C]} = \frac{Pr[z \in C_3]}{Pr[z \in C_2] + Pr[z \in C_3]}
\]

(5.1)

Note that this is exactly the same formula as used by the MultiCoil program in the computations of the trimeric oligomerization ratio. For MultiCoil, each term on the right hand side of the equation was replaced by the probabilities estimated by the MultiCoil program in locating the coiled coils.

An alternative approach is presented here. Instead of estimating each of the two probability terms on the right hand side of the equation, a single score vector derived from the probability formula on the right hand side is computed. This score vector is then converted into a probability using the same techniques of classification theory as used by the MultiCoil program. The exact same methods of conditional probability analysis used earlier to derive the PairCoil scoring method in equation 2.2 can be applied to the oligomerization probability formula. Let \( X = X_1 \cdots X_w \) denote a random sequence chosen from the underlying distribution. The quantity for the oligomerization probability in equation 5.1 is inverted for the purpose of computing it in a framework which is exactly analogous to the PairCoil score. Cancelling the common factor of \( Pr[X_1 = z_1, \ldots, X_w = z_w] \) in each of the terms and assuming that all dependencies are captured by pairwise interactions at distance \( d \), the equation becomes:

\[
\frac{1}{Pr[z \in C_3 | z \in C]} = \frac{Pr[z \in C_2] + Pr[z \in C_3]}{Pr[z \in C_3]} = 1 + \frac{Pr[X \in C_2 | X_1 = z_1, \ldots, X_w = z_w]}{Pr[X \in C_3 | X_1 = z_1, \ldots, X_w = z_w]}
\]

\[
= 1 + \frac{\prod_{i} Pr[X_i = z_i | X \in C_2]}{\prod_{i} Pr[X_i = z_i | X \in C_3] ; \prod_{i} Pr[X_i = z_i | X \in C_3]}
\]

See the derivation of the PairCoil algorithm for the details on the derivation of this formula using conditional probability analysis. This quantity minus one is exactly the same quantity as obtained in equation 2.2 for the PairCoil score, where now the probabilities from the two-stranded database are normalized
by the probabilities from the three-stranded database instead of GenBank. While the PairCoil probability computed a measure of whether the sequence was more similar to the coiled-coil database than GenBank, the basic intuition behind this score is that it measures whether the sequence is more similar to the two-stranded database or the three-stranded database. Taking the logarithm of the quantity minus 1, and subtracting off the constant term \( \log \frac{Pr[X \in C_2]}{Pr[X \in C_3]} \) results in a score which captures all of the variable information of the probability formula, and can therefore be used to classify sequences as dimeric or trimeric. The PairDiff program was written to implement this algorithm by minor modifications to the PairCoil code. For the window \( W \) with a given heptad-repeat positions for the sequence \( z \), let \( D_W(z) \) denote the PairDiff differentiator score:

\[
D_W(z) = \log \left( \frac{\prod_{i} Pr[X_i = z_i, X_{i+d} = z_{i+d} | X \in C_2]}{\prod_{i} Pr[X_i = z_i | X \in C_2]} \cdot \frac{\prod_{i} Pr[X_i = z_i | X \in C_3]}{\prod_{i} Pr[X_i = z_i, X_{i+d} = z_{i+d} | X \in C_3]} \right) = \sum_{i \in W} \left( \log \frac{Pr[X_i = z_i, X_{i+d} = z_{i+d} | X \in C_2]}{Pr[X_i = z_i, X_{i+d} = z_{i+d} | X \in C_3]} - \log \frac{Pr[X_i = z_i | X \in C_2]}{Pr[X_i = z_i | X \in C_3]} \right).
\]

If all residues in the sequence are assumed to be independent, then the formula for \( D_W(z) \) reduces to the score computed by the SCORER program of Woolfson and Alber (97). This derivation gives probabilistic justification to their score as the variable portion of a formula for the probability of being dimeric or trimeric based upon simplifying assumptions (independent residues for the SCORER program and pairwise dependencies for the PairDiff program). The SCORER program extends the methods of the NEWCOILS program, and the PairDiff program naturally extends PairCoil.

There are two major distinctions between the PairDiff program and the PairCoil program. The PairDiff program takes a coiled-coil region as input, and so does not need to take a maximum of the scores over the possible heptad-repeat positions as done in PairCoil. Additionally, taking the maximum of the scores over all windows containing a residue does not make sense. Since \( Pr[z \in C_3] = 1 - Pr[z \in C_2] \), taking the maximum over window scores for the probability estimate of one oligomerization state corresponds to taking the minimum over the windows for the other oligomerization state. There are instead two options: 1. compute scores only for windows, not for individual residues; 2. give each residue a score which is a function of the window scores for all the windows containing it. The SCORER program used the first of these options, where their windows corresponded to the entire coiled-coil region. However, it is desirable to convert the PairDiff scores into probabilities of the region being dimeric or trimeric using the methods from classification theory implemented for the MultiCoil program. Since the length of the window can
affect the magnitude of the score, it is necessary to either use fixed window lengths or to normalize the score by dividing by the length of the coiled coil. Using fixed window lengths also eliminates problems that can arise from biases in the coiled-coil locator. The preprocessor could be biased towards finding longer coiled coils in one particular oligomerization state.

In addition to computing window scores, the PairDiff program is capable of computing residue scores based on the scores of all windows containing the residue. Offhand, one might consider giving each residue the average score of all the windows containing it. Averaging the scores does not correspond to averaging the probabilities however, as the scores are derived from the inverse of $Pr[z \in C_3|z \in C]$. Averaging the scores would destroy the probabilistic framework used to justify the scores. In fact, it is difficult to probabilistically justify any simple combination of the scores which does not first convert them into the probability being estimated and then average those terms. Drawing on the multi-dimensional scoring methods of the MultiCoil algorithm, a different tact was taken. Instead of combining the different window scores containing a residue, each residue was given two scores derived from the window scores. The two scores were taken to be the maximum score over all windows containing the residue and the minimum score over all windows containing the residue. These two scores correspond respectively to the “most dimeric” window and the “most trimeric” windows for the residue. The PairDiff algorithm can therefore compute residues scores which are two-dimensional vectors. If more than one distance is used as a scoring dimension as in the MultiCoil program, the score vector is $2d$-dimensional, where $d$ is the number of distances used. The techniques presented in chapter 3 are used on these score vectors to convert the PairDiff residue and window scores into probability estimates for the oligomerization state of the coiled coil.
Chapter 6

Conclusion

The multi-dimensional scoring system introduced in this thesis provides a simple framework to approach statistical classification problems. At this point, the most major limitation appears to be the quality of the databases which are used to estimate the statistical features of coiled coils. The greatest immediate performance gains will result from addressing the problems which arise from the small trimeric database. Merging the MultiCoil program with the LearnCoil program (12) would allow new sequences to be added to either the dimeric or the trimeric database as they are located and classified by the MultiCoil program, bootstrapping the databases to create larger samples. The merger may dramatically improve the confidence with which trimeric coiled coils are located by the program. The resulting more diverse database could improve the ability to separate weakly scoring coiled-coil regions from high scoring negatives. Another approach to the small database problem involves improving the estimation of residue probabilities from the database. Despite the limited success currently realized in figure 4-11, using only structurally relevant heptad-repeat positions in the computations as done by Woolfson and Alber (97) is a promising approach to significantly improving the performance of the program. Avoiding over fitting to the databases in this way could especially help in locating new coiled coils.

Another methodology issue to consider for the MultiCoil program involves a thorough comparison of the Combination Register Method with the Maximum Register Methods. In initial trial runs of the Maximum Register Method 1, the performance was found to be comparable with the Combination Register Method, with slightly lower probabilities assigned to some coiled coils not in the database, but also with slightly lower probabilities assigned to sequences in the PDB-minus. The Maximum Register Method 2 is predicted to raise
the probability assigned to new coiled coils, due to the overall lowering of the Gaussian distribution fit to the scores of the PDB-minus. It is however unclear whether the Maximum Register Method 2 can be optimized to avoid false positive predictions.

A slightly different approach to improving the recognition of new coiled-coil regions involves using other coiled-coil prediction schemes as sub-processors, in addition to the PairCoil scoring dimensions currently used. Since the NEWCOILS program over predicts, resulting in false positive predictions on non-coiled-coil α-helical regions, and since PairCoil tends not to predict these regions, using each of these programs as a scoring dimension may be advantageous for locating more potential coiled-coil regions without false positives.

The multi-dimensional approach could also be used to create a general protein fold prediction algorithm from numerous algorithms already in use. For example, currently there are a great variety of secondary structure prediction algorithms which obtain moderate predictive success (in the 60% range). By using each predictive scheme as the dimensions of a new secondary structure prediction algorithm, the accuracy could be increased (as long as the incorrect predictions are not the same for all the schemes). The MultiCoil approach could also possibly be used to differentiate a number of different types of α-helical structures (for example, to differentiate α-helical bundles from coiled-coils).

As better biological databases are compiled, the multi-dimensional methods presented here could be extended to locating and distinguishing parallel and anti-parallel coiled coils. Similar work could possibly be useful with homo-oligomers and hetero-oligomers. However, to build a database of hetero-oligomeric coiled coils at this point would require significant advances in biological knowledge. The pairwise interactions between strands would have to be gathered in a database using alignments between the two strands. While the probabilistic framework of the PairCoil program would extend to the problem of hetero-oligomers, such data is generally not available. In addition, to truly correctly predict hetero-oligomers, a program would have to align potential strands and compute a score based on the pairwise interactions between the alignments. The PairCoil and MultiCoil programs deal with these problems by assuming that the pairwise interactions between strands are roughly estimated by the pairwise correlations within each strand, since the structurally relevant heptad-repeat positions tend to encounter similar environments. The major disadvantage with this simplification is situations such as the following: while pairs of like charged elements are disfavored at interacting positions within a homo-oligomer, they are not disfavored in a single strand of the hetero-oligomer, since both charges can interact with an opposite charge in the other strand. The MultiCoil and PairCoil programs have nonetheless proved useful for locating hetero-oligomers.

Future improvements to the methods will undoubtedly involve incorporating additional biological information into the statistical schemes. Considering the dynamics of packing at a and d positions has already been
shown to effectively predict the oligomerization states of the GCN4 leucine zipper (41). Other non-statistical features which may be relevant included capping sequences, disulfide bonds, and initiation sites of folding. For instance, certain charged amino acids and several short regions at the C-terminal end of laminin have been shown to be prerequisite for the formation of the coiled-coil structure (88, 89). In fact, these results showed that different regions in the sequence are critical for the formation of dimers and the formation of trimers, and that trimers are able to form even when the dimers are unable to form. The intermediate formation and stability of dimeric coiled coils during the folding of trimeric coiled coils brings up another interesting issue that makes the goal of "perfect" separation of dimers and trimers by a program unrealistic. One can only hope to quantify the tendencies of the oligomerization state using the prediction programs.

Disulfide bonds between cysteine residues in two strands of the coiled coils in laminin and fibrinogen are another relevant biological features that might be used to improve performance. While these bonds do not bring about the formation of a coiled-coil structure, once the structure has formed they stabilize it. The cysteine residues can occur outside of the coiled-coil region, and hence, their occurrence is not necessarily captured by the statistical analysis of current programs. Considering the stabilization of structures could potentially be beneficial in predicting whether a trimeric structure is favored over an intermediate dimeric structure. Applying threading methods to analyze the energy of the three-dimensional predicted coiled-coil structures could additionally improve the accuracy of the predictions.

The current programs often also have troubles correctly locating the starts and ends of coiled-coil regions and the occurrence of register skips. Perhaps considering "helix capping boxes" and stop signals (43, 3) could aid in localizing these features. Correctly locating register shifts could also help in distinguishing dimers from trimers, since it has been suggested that register shifts are more common in three-stranded coiled coils. Part of the difficulty that the current programs face with localizing these structural changes arises from the window based approach of the programs, as windows "de-localize" the scores. A program such as HMMCoil\textsuperscript{1} (13, 7) which uses hidden Markov chains for coiled coil prediction could improve the recognition of structural changes. HMMCoil could then be used a subprocessor with the MultiCoil program. Problems in recognizing the starts and ends of coiled coils may also have to do with tertiary structure. For instance the 14-3-3 zeta-subtype protein has been shown to fold back on itself to form an anti-parallel two-stranded coiled coil (60). The statistical predictor programs locate a single coiled-coil region overlapping the end of the first strand and the start of the second strand. It is possible that tertiary structure influences the spatial positioning within the strand to favor the formation of certain coiled coils over others. At this stage, such predictions are beyond the scope of secondary structure prediction schemes.

\textsuperscript{1} The program HMMCoil is currently under development in the research group of Bonnie Berger.
It is apparent that there are very many issues left to explore in the specialized field of statistically predicting coiled coils. There is no reason to avoid incorporating additional biological knowledge to improve the predictions and automate the process of weeding out bad predictions. The grand problem of definitively predicting a protein structure can best be solved by expanding the methods to incorporate statistical database methods, energy minimization approaches, and simplifications based on the biological knowledge. The multi-dimensional scoring methods provide one framework for merging different approaches. The algorithms from this thesis represent a small first step.
Appendix A

A Brief Guide for using the

MultiCoil Program

For the results reported in this thesis the scoring dimensions used with MultiCoil were distances 3, 4, and 5 with the dimeric database and distances 2, 3, and 4 with the trimeric database. These distances were chosen to simultaneously locate and distinguish two and three-stranded coiled coils. Other sets of distances may be more suited to other applications. If a region is known to be in a coiled coil and MultiCoil is used only as a dimeric/trimeric distinguisher, distance 4 may not be the most relevant trimeric distance (see table 3.1). The MultiCoil and PairDiff programs allows for the scoring dimensions to be modified at the user’s discretion. Changing the scoring dimensions changes which characteristics of the structures in the known databases are emphasized for classification.

Additional improvement in performance can be obtained by tailoring the parameters $P_{dim}$ and $P_{trim}$ to the sequences being tested. These parameters represent the a priori probabilities that a random residue in a sequence is in a dimeric or trimeric coiled coil. They were optimized for locating coiled-coil sequences from a database of random sequences, without returning non-coiled-coil sequences. However, if the program is being run on a sequence which is known to contain a coiled coil, it may be beneficial to raise these parameters given that initial knowledge. The initial knowledge changes the initial probabilities. For a sequence known to contain a coiled coil, eliminating “false positives” is less important than classifying the type of coiled coil, and the parameters may be adjusted accordingly.
A last parameter which can be adjusted is the bound used to determine which residues are classified as coiled-coils when computing the sequence scores and the coiled-coil scores. For scanning through a large database, the bound can be set relatively high in order to return only the most probable coiled coils. When running MultiCoil on a set of sequences thought to contain coiled coils, the bound can be set lower in order to classify even sequences with a weak propensity for coiled coil formation.

A.1 Interpreting Scores

The sequence and coiled-coil scores are the most straightforward scores to interpret. If the total coiled-coil probability is high (e.g., above 75%) then there is high confidence that the region forms some sort of a coiled coil. If most of that probability is contributed by either the predicted dimeric or the predicted trimeric probability, then the region tends towards that oligomerization state. In sequence regions where there is a significant amount of probability contributed by both oligomerization states, there are several interpretations. The region could very well be dimeric or trimeric (with scores falling in the area of overlap between the dimeric and trimeric data sets in Figure 3-2). Scoring in this overlap area could also indicate that the bias towards one particular oligomerization state is not overwhelming, as with the GCN4 mutation which forms both dimers and trimers (table 3.2). Another possible interpretation of probabilities and oligomerization ratios that are not very strong could be that the region is a coiled coil, but is neither dimeric or trimeric.

Interpreting very weakly predicted coiled-coil regions is another difficult problem. As can be seen from Figures 3-2 and 3-3, the scores for known trimers and known non-coiled coils overlap. Hence, a coiled-coil probability under 50% does not mean that the region is definitely not in a coiled coil. It merely means that there is a greater chance for that region to be in a non-coiled-coil structure than in a coiled-coil structure. Weakly predicted regions may also possess some properties of coiled coils, but not fit the exact structure of a coiled coil. For example, M29975: Simian immunodeficiency virus (SIV), env polyprotein scores 15% total coiled-coil probability (with 13% of that trimeric). It has been suggested that SIV and HIV form a bundle of 6 alpha-helices which could possibly be grouped as a trimeric coiled-coil core surrounded by the three other helices, or as a trimer of dimeric complexes (15, 61). Lipoprotein scores 44% total coiled-coil probability and forms a four α-helical bundle.

It is especially important to weight the confidence placed in the accuracy of the oligomerization ratio by the strength of the coiled-coil region. It was found in trial runs that the accuracy of the oligomerization ratio predictions decreased as the total predicted coiled-coil probability decreased. Since the low scores for the trimeric database tend to overlap with the high scores for the PDB-minus, many of the weakly predicted coiled coils
tended to score as trimeric. This effect could also be a function of the fact that a larger number of aligned sequences were used to construct the dimeric database. The greater repetition causes known coiled coils to often be located with higher scores than "unknown" sequences. The scores based on the smaller trimeric database are less fit to the database.

A.1.1 Interpreting Residue Scores

The MultiCoil residue scores (Figure 3-5) indicate the regions of a coiled-coil which contribute most to the predicted oligomerization state. The distribution of these scores along the coiled coil can be used to shed light on whether certain regions of the coiled coil are more influential in determining the oligomerization state than other parts. In the case of sequences which are not strongly classified in one state by the coiled-coil scorer, the distribution of residue scores might provide additional information to help hypothesize the structure of the region.

An additional point of interest occurs when examining the register predicted by MultiCoil for a coiled coil. More than one heptad-repeat register can have a high score. The scores output by MultiCoil come from the maximum scoring register along each dimension, but the probabilities predicted for the other registers also provide information. While one predicted register may give the coiled coil an apparent dimeric structure, another register may predict a trimeric coiled coil. Knowledge of these predictions can be used to better understand the predicted structure.
Bibliography


