ANALYZING, EXPLORING, AND VISUALIZING NEAR-OPTIMAL PROTEIN SEQUENCE ALIGNMENTS

A Dissertation

presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

In partial fulfillment

of the requirements for the degree

Doctor of Philosophy

Systems Engineering

by

Michael Elliott Smoot

May 2005

Approval Sheet

The dissertation is submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Systems Engineering

Michael Elliott Smoot

This dissertation has been read and approved by the examining committee:

Stephanie A. Guerlain (Co-advisor)

Ellen J. Bass (Co-advisor)

Donald E. Brown (Chair)

William R. Pearson

Robert H. Kretsinger

Accepted for the School of Engineering and Applied Science:

James H. Aylor

(Dean, School of Engineering and Applied Science)

May 2005

Abstract

This dissertation describes my research into supporting the use of near-optimal protein sequence alignments by biologists. The research involves contributions to bioinformatics (investigating the relationship of near-optimal alignments to structural alignments) and cognitive systems engineering (developing a near-optimal sequence alignment solution space analysis system). The bioinformatics contributions show that the variation between structural alignments compares favorably with that of near-optimal alignments. The results indicate that analyzing near-optimal alignments can be used for developing higher quality homology models for sequences without known tertiary structure. This research further explores the relationship between structural and near-optimal alignments by developing a logistic regression model that predicts whether or not aligned pairs of amino acids in a set of near-optimal alignments are likely to be found in structural alignments. This work adds to cognitive systems engineering by demonstrating an effective system for supporting biologists in the exploration of large sets of near-optimal alignments. This support comes in the form of alignment visualization techniques and facilities for mixed-initiative interaction. Two visualizations were created, an animated pairwise alignment and a zoomable path graph, which provide alternative perspectives on sets of near-optimal alignments. A mixed-initiative interaction scenario is created by allowing users to dynamically edit and adjust alignments, which creates a feedback loop. This provides further insight into the alignment generation algorithms. The visualization techniques take advantage of the biological insights developed in the first section of this research to further increase the usefulness of the system. Two case studies demonstrate the utility of the near-optimal alignment solution space analysis system. One case study describes the use of our visualization and analysis system to confirm the homology of two distantly related proteins, Lacritin and Dermcidin. The sec-

ond case study describes how the visualization options, filtering, and mixed-initiative features of the system facilitated the development of an $O(n)$ space near-optimal alignment generation algorithm.

Acknowledgments

I would first like to thank my committee for the many opportunities to learn and grow that have been presented to me. I would like to thank Bill Pearson for welcoming a systems engineer into his lab. I greatly appreciate the insight and wisdom both Ellen Bass and Stephanie Guerlain have imparted over the years. I am also particularly grateful for their leadership and management abilities. I thank Don Brown and Bob Kretsinger for their useful guidance in my research. I am indebted to Mike Sierk who has been a skilled and thoughtful collaborator. I would like to express my gratitude to the entire Pearson Lab for their support over the years, particularly Dan Lavelle, Aaron Mackey, Justin Reese, Brandi Cantarel, Aaron Gussman, Fitz Elliot, Anne Westbrook, and Royden Clark.

The opportunity for me to pursue a Ph.D. studying bioinformatics would not have been possible without the generous support of the U.Va. Biotechnology Training Program. I would specifically like to thank Gordon Laurie for his support and encouragement and for taking a chance on a systems engineer. Navigating the academic bureaucracy has been made infinitely easier and less painful by the many Engineering School and Systems and Information Engineering department support staff. I would particularly like to thank Sharon Predmore, Margret Proffit, Jeanette Davis, Dee McCauley, Ann Wanner, Debra Hirst and Tammy Ramsey who run offices that should be the envy of all others.

The trials and tribulations of grad school have been shared by many good friends. I would like to thank Matthew Gray, Sue George, Matthew Wikswo, Dave Wotton, Lucy Pemberton, Marc Breton, Gary Holmes, Brian Lacey, Mark Mauss, Patricia Nordeen, Andrea Laue, Dave Forrest, Jim and Louise Gunderson, Jen Creasy, and everyone else with whom I've ever had the pleasure of exploring Virginia for their friendship and camaraderie.

I am also indebted to my immediate and extended family. I am grateful to my grandparents for their inspiration and guidance. Likewise, I am grateful to my sister Liane Cutforth and brother-in-law Craig Cutforth for their support and encouragement over the years. Despite no conscious contribution, young nephew Davis Cutforth also deserves thanks for the many pictures on my refrigerator that make me smile each day.

Finally, I would like to thank my parents, Art and Carol Smoot. Without their unwavering love and support, none of this would have been possible. This dissertation is for you.

Portions of this dissertation have been published in *Bioinformatics* 20(6) pp. 953-958, 2004 and appear here courtesy of Oxford University Press.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This research involves the study of near-optimal protein sequence alignments and aims to support biologists using the output of imperfect models. The motivation for this research stems from the reality that mathematical models fail to account for all of the variables, parameter values and constraints inherent in the phenomena they attempt to model. The simplifying assumptions inherent in the sequence alignment model can lead to results that are incorrect from a biological perspective. Therefore, the goal of this research is to better understand how the sequence alignment model, however flawed, can be used to better understand a protein alignments.

Proteins are responsible for most functions in living organisms. As a consequence, biologists are frequently interested in determining the function of unknown proteins isolated during experimentation. One way to do this is to compare unknown proteins with known ones. If two proteins are sufficiently similar, then it is likely that the two proteins are homologous (meaning they share a common ancestor) and therefore function in a similar way. The greater the similarity between the proteins is, the more recent their common ancestor and vice versa. By understanding this relationship, scientists can direct further research into the unknown proteins.

Proteins consist of long chains of amino acid residues that fold into 3-dimensional structures. The linear chain of amino acids is called the protein's sequence or primary structure and is the basis for most bioinformatics analysis. Sequence alignment has been used for more than 30 years as a fundamental tool in biology. A sequence alignment is simply an attempt to match the characters representing one sequence with the corresponding characters in a second sequence. The general

goal of alignment is to understand how similar or dissimilar one sequence is from the other. The most common usage is to detect homologous sequences in a database of DNA or protein sequences [1][2]. Once homology is established with a known sequence, inferences can be made about the structure, function, and significant residues of the unknown sequence. However, these inferences are often critically dependent upon the quality of the alignment between the two sequences. The usual gold standard by which sequence alignments are assessed is the structural alignment between the two proteins. This is a reasonable standard, since the three-dimensional structure contains more information than the one-dimensional sequence, and is ultimately required for a full description of a protein's function. Conversely, sequence based alignments are used to create homology models of protein families when only incomplete structural information is available.

Alignments in general, and protein sequence alignments in particular, are entirely abstract constructions. There is no natural process by which two different protein sequences align themselves. Alignment algorithms are mathematical models used to aid our understanding of the relationships between different sequences. This means there is no "right" or "correct" or "optimal" *alignment* as there is no absolute standard by which to evaluate an alignment. From a mathematical perspective, this makes the problem of sequence alignment ill-posed [3]. This does not, however, impede the biologist from extracting useful information from a reformulated model. By slightly changing the problem statement (such as treating amino acid position as if it were independent, allowing a constant rate of evolution), we can have a well posed problem yielding models that can be optimized [4][5]. To do so, a trade-off has been made with the implicit acknowledgment that an optimal solution to the well posed model is still only an approximation of what is necessary for the analysis. While this model can be optimized according to a particular metric, there is still no algorithmic way to determine whether an alignment is biologically sensible. Only a scientist with a deep understanding of the domain can accurately determine whether a particular alignment makes sense. We do not view this lack of standard as a problem, because the goal of sequence alignment is not to produce a solution, but rather to develop insight and understanding of the sequences involved.

Given this incomplete model of optimal sequence based algorithms, it seems reasonable to expect that the algorithms will create alignments that make no biological sense in certain circumstances.

Optimal sequence based algorithms are known to misalign key functional residues[1] [6]. In these situations, the alignment model has insufficient information to properly align the sequences. Missing is the knowledge (generally derived through physical experiments, possibly including the identification of the 3-dimensional structures of the proteins) that particular residues must align. Despite these failures, the many successful alignments that *are* produced lead us to believe that the incorrect alignments are close to being correct. That is, small changes in the alignment would improve the alignment from a biological perspective. Therefore, in cases where misalignments occur, we believe it is reasonable to search in a neighborhood surrounding the optimal solution (the near-optimal solution space) for a solution that does not share the same failures. Near-optimal alignments are sequence based alignments with scores that fall within a certain threshold of the algorithmically optimal score [7][8][9][10].

The goal of this work is to enhance our understanding of protein sequence alignments through the use of near-optimal alignments. The hypothesis explored in this dissertation is that a set of near-optimal solutions contains more information than a single algorithmically optimal alignment. Specifically, we hypothesize that near-optimal solutions can be used to further our understanding of 3-dimensional structural characteristics of proteins without having the actual 3-dimensional structure. In addition, we explore different visualization and control techniques that facilitate viewing and comprehension of sets of near-optimal alignments.

The dissertation contains four chapters that constitute the contribution to Bioinformatics and Systems Engineering followed by a concluding chapter. Chapter 2 describes the comparison of sets of near-optimal alignments with alternative structural alignments. This research demonstrates that sets of near-optimal alignments compare favorably to structural alignments. The results described in Chapter 2 motivate the research in Chapter 3. In Chapter 3 we develop a probabilistic model that predicts whether pairs of amino acids can be expected to align in structural alignments based on metrics derived from sets of near-optimal alignments. Chapter 4 describes the system developed for researchers to generate, visualize, and study near-optimal alignments. We describe the novel visualization techniques and algorithms developed to facilitate this study. Chapter 5 relates two case

[1] Residue is short for amino acid residue.

studies that demonstrate how the visualization software has been successfully used in real-world scientific discovery. Finally, Chapter 6 concludes the dissertation.

This research enhances our understanding of the relationship between sequence based and structural alignments and provides guidance that aids in the exploration and understanding of near-optimal alignments. The research manifests itself in a system that supports the display and exploration of near-optimal solution space.

A subset of the research presented in Chapter 4 and 5 has been published in [11] and another paper is in press [12]. Another manuscript describing the work in Chapters 2 and 3 is in the final stages of preparation.

1.1. Background and Significance

1.1.1. The Importance of Sequence Based Alignments in Modern Biology

Proteins are the building blocks of life. Ranging from enzymes that aid in digestion, to hemoglobin which transports oxygen in the blood, to the various structural proteins that comprise our muscles and bone, proteins are central to almost all aspects of biology. Protein molecules consist of specific sequences of amino acids which fold into 3-dimensional shapes that determine the function of the proteins. The twenty amino acids have a variety of chemical properties (e.g., acidic/basic, positive/negative charge, hydrophilic/hydrophobic) that, when combined, allow for the vast array of functions that proteins perform. The sequence of amino acids is referred to as the primary structure of a protein while the 3-dimensional shape the sequence folds into is referred to as the tertiary structure[2] [13]. The primary structure of a protein is relatively cheap and easy to find while finding the tertiary structure is substantially more expensive and time consuming. This is reflected in the number of sequences available compared to the number of protein structures available. There are far

[2] Secondary structure is the local packing of amino acids according to how hydrogen bonds form between the CO and NH of different amino acid residues. Secondary structures are either *alpha helices* where hydrogen bonds between every fourth residue form helical shapes or *beta sheets* where hydrogen bonds form between adjacent strands creating sheet-like shapes. Quaternary structure is when two or more separate strands of amino acids fold together into a single structure.

fewer known protein structures (28,648 structures as of December 7, 2004 in the Protein Data Bank [14]) than there are protein sequences (1,917,944 entries as of December 4, 2004 in the PIR-NREF [Protein Information Resource Non-redundant Protein Reference] database [15]). Contrast this with 20,533 structures and 976,519 sequences as of April 1, 2003 and it should be clear why we expect this trend to continue into the foreseeable future. Given this difference, the ability to predict the tertiary structure of a protein (and thus function) based only on a sequence of amino acids is very desirable. This, however, is an extremely difficult problem. While it is believed that sequence implies structure, we do not have a good understanding of the mechanism that proteins use to fold into their 3-dimensional structure [16]. This means we are not able to make accurate predictions of structure or function. Many efforts are underway to understand protein folding and structure prediction [17]. Many would argue that accurately predicting structure and function from a sequence of amino acids is the holy grail of modern biology.

While we may not yet be able to predict structure from sequence, there is much to be learned from studying protein sequences. The fundamental computational tool for analyzing sequences is the sequence alignment [18]. Alignments are a way of comparing one sequence to another and making inferences about unknown proteins. Among other activities, they are used for establishing relationships between sequences [19], establishing homology [20] and for sequence database searching [1][2]. Sequence alignment is therefore a fundamental activity in modern biological research.

While finding protein structures is a costly process, we do know the structures of several thousand important proteins. If we do happen to have the structures of two proteins being aligned, then it is possible to create a *structural alignment*. Structural alignment algorithms attempt to account for the 3-dimensional position of each amino acid when the alignment is being generated [21]. Because structural alignments account for more information about the proteins, they are thought to be superior to sequence based alignments. Aside from the relative unavailability of protein structures, the largest source of difficulty with structural alignment algorithms is that the additional information available for constructing the alignment increases the dimensionality of the problem such that heuristic algorithms are required to produce alignments. This means that there are several alternative structural alignment algorithms that each produce different alignments.

Given the expense of solving structures directly and our inability to predict structure, the current approach taken by the various high-throughput structural genomics projects is to solve as few structures as possible while ensuring that a homology model of sufficient quality can be constructed for every sequence. Current methods allow high-quality models to be produced when the sequence identity[3] is ~40% or higher, but statistically significant ($E() < 10^{-2}$) homology can be detected below 20% sequence identity, which leaves a large number of known homologs for which reliable models cannot currently be built. The biggest hurdle facing structure predictors in this range of sequence similarity is the accuracy of the alignment. After picking the proper template molecule, the next most important step in producing an accurate model is generating a biologically correct alignment between the template and sequence to be modeled. There is widespread agreement that most of the modeling efforts that fail in the 20-40% identity range fail due to poor alignment quality (alignments between sequences with >40% identity generally correspond closely with the structure-based alignment). Thus much effort has been spent attempting to improve alignment accuracy in this area of sequence similarity space.

This research uses sequence based alignments of proteins to help us further our understanding of the relationship between sequence based and structural alignments. This knowledge is a small step in the effort to predict or understand protein structure without knowledge of the structure.

1.1.2. Humans, Visualization, and Automation

The goal of this research, from a cognitive systems engineering perspective, is to study ways in which to make imperfect system models more useful. This research involves the integration of automation, information visualization, and human judgment.

We know that algorithmically optimal sequence alignments sometimes fail to align key functional residues [6]. Two complementary approaches address this problem. One involves improving the alignment algorithm and many such attempts have been made, with varying degrees of success [20][22]. The alternative explored here is the support of human expertise when generating and

[3] The percentage of amino acids that are aligned with identical amino acids.

interpreting results. Our approach involves presenting the researcher with a set of alternative alignments rather than a single "optimal" alignment. It is important to emphasize the point that we are not seeking one "correct" alignment, but rather using a solution space to enhance our understanding of the relationship between two sequences. The system we developed is therefore not precisely a decision support system, although it shares many characteristics that a decision support system might have.

Much of this research involves *information visualization*, which should be distinguished from *scientific visualization*. Card, et al. [23] defines scientific visualization as an attempt to represent a physical system on a display medium. In contrast, information visualization attempts to create a visual representation of *abstract* data with no natural or underlying physical form. The canonical example of information visualization is the simple XY scatter plot. While biological sequences have clear physical representations, alignments have no physical analogs and thus must be considered an exercise in information visualization.

As a way of presenting alternative alignments, this research explores the use of animation. The use of animation is common in scientific visualization as the need to express movement and change over time is necessary for accurate representations of real world systems. There seems to be relatively little use of animation in information visualization. As in the representation of real systems, animation is used to assist users in maintaining context awareness as the state of an abstract representation changes [24]. Perhaps the most common use of animation of abstract data is in the animation of computer science algorithms [25]. However, even in these situations the animation of the quick sort algorithm is not strictly information visualization as the task of ordering something by size has many clear physical analogs. Even the visualization of sequence alignments has some connection to the physical world, because all biological sequences have physical shapes.

While there is no literature directly related to the animation of near-optimal alignments, there is substantial support for the use of animation. Adding a time dimension to representations of physical systems can help comprehension and communication [26]. It is well understood that vision is a high bandwidth sensory organ [27]. However, the use of size, shape, and color limit the amount of information that can be displayed on one computer screen. The use of motion can increase

the bandwidth of one screen [28]. One example is the visualization of causal relationships [29]. Imagine one circle on a screen moving until it intersects with a second, stationary circle, the first circle stopping and second circle moving in the same direction. The inference to be made is that the first circle struck and therefore caused the second circle to move. Likewise, animation is used to signal transitions from one state or phase to another [26][30].

Of course, not everyone is convinced that animation is helpful. Specifically, Tversky et al. claim that animation does not facilitate learning [31]. The source of their complaint is that most studies that claim to compare the same data represented in static and animated graphics, in fact, show two different sets of data. That is, the animation shows more data than the static graphic does and thus there is no way of determining if the improvement in animated displays is the result of the animation or the extra data present. Regardless, there *is* an improvement in the amount of information communicated, which supports the notion of animation increasing bandwidth [28].

Closely related to information visualization is the art of visual data mining or visual data exploration. Visual data mining attempts to couple human perceptual capabilities with data and computational power to help induce useful models [32]. Beyond simply providing a visual representation of data, visual data mining attempts to uncover patterns and relationships in data that otherwise would have been obscured. Human perception is a powerful resource that is particularly amenable in situations where algorithms and computers have difficulty such as with noisy or non-homogeneous data or when the consumers of data are not trained in the mathematical and statistical methods necessary to interpret certain models. The process of visual data mining or exploration is usually construed to have a three step process: first overview, second zoom and filter, and finally detailed presentation as needed [33]. Part of this research focuses on the use of information visualization techniques that exploit human pattern recognition capabilities to help avoid problems in sequence alignment such as misaligned residues.

One of the key difficulties with near-optimal sequence alignments is the large number of alternative solutions produced. In contrast with visual data mining, the notion of exploring a solution space seems somewhat less developed. One example is for airplane route finding software [34]. Like sequence alignment, the routing system described uses a dynamic programming algorithm to generate

alternative solutions and, like our system, presents the alternatives to a user. Another example is the use of a browsing metaphor for comparing alternative CAD (computer aided design) generated estimations of hand drawn figures [35]. Other research looks at using visualizations and human knowledge to assist in computationally hard problems by suggesting regions for search and helping a computer out of local minima [36]. The common thread in these efforts and with this research is that in each case there is a large possible solution space with visual representations that are amenable to human interpretation.

The goal of all visualization efforts is to combine human perceptual capabilities with data and computational power to produce outcomes that are greater than the sum of their parts. Joint cognitive systems [34] reflect an approach to decision support systems that couple human decision making ability with the computational power of computers. However, it is well known that automation is a double edged sword. Computer automation can accomplish things that no human can hope to, yet automation can adversely affect our situational awareness[37], lull us into a sense of complacency [38] and frequently annoy [39]. Rather than just a human or just a computer making decisions, joint cognitive systems are an attempt to balance the strengths of humans and computers to create more effective problem solving and decision making systems. The misalignment of key functional residues in sequence alignment is an example where automatic techniques fail in the bioinformatics domain. We believe that with the introduction of human expertise by visualizing and generating alternative alignments, we can develop a system that will help create more biologically sensible alignments.

Levels of computer automation are generally classified on a scale ranging from maximum automation where a computer does everything without any input from or feedback to the human to no automation where a human must make all decisions and perform all actions [40]. Most decision support systems are built around a particular model of automation and tend to stay within a given level. Mixed-initiative systems are systems that allow for multiple levels of automation within a single system [41]. This means that users can, at one point, rely (or not) on a certain level of automation yet can change that level as circumstances demand. An idealized scenario allows a human to offload work to an automated system when the human is too busy or pressured to properly manage it. Other systems switch between user initiated actions and automatic action depending on

context and underlying plans available to the automation [42]. This system provides for various levels of automation and the ability to switch between levels. Specifically, users can edit their own alignments and have the alignments evaluated using the same criteria as those generated using automatic methods.

This work takes advantage of the coupling of overview and detail displays, a topic that has been studied in other domains as a way of increasing situational awareness and facilitating navigation [43][44]. Previous systems with such overview and detail displays are frequently discussed in the context of decision support roles [45][46] or in terms of browsing [47]. While our system is neither a decision support tool nor a browser, evidence suggests that the coupling of overview and detailed information, whether by separate panes [47] or zooming [48], show improved performance over single view, static displays.

The integration of other features into these different representations can improve the overall system. Alignment Viewer [49][50], a tool for viewing the results of DNA sequence database searches, uses animation, filtering, zooming, and icons to display large numbers of search results and to communicate different information about alignments. For example, their use of filtering can dynamically constrain the results presented to the user. Their comb metaphor accomplishes what our similarity/identity highlight does and it is easy to see how the comb metaphor could be used to communicate conserved regions or robustness. This provides evidence that our protein sequence alignment tool may also benefit from the integration of these features. While our system shares many of the same techniques used by the Alignment Viewer, the domains (DNA search results vs. protein sequence alignments) are different enough that a direct comparison is not possible.

1.2. Prior Results in Near-optimal Sequence Alignment

1.2.1. Near-optimal Alignment Generation

Sequence based alignment algorithms are variations on shortest path dynamic programming algorithms. The models introduced by Needleman and Wunsch [5] and refined by Smith and Waterman

[4] view the alignment of two sequences as a set of operations performed on one sequence that transform it into a second sequence. The operations are *match* (the amino acids are the same and thus make no change), *substitute* one amino acid for another, *insert* a new amino acid into the sequence, and *delete* an amino acid from the sequence. Each operation is given a score and then all operations are summed to produce an overall alignment score. Optimal solutions for this *model* can be found. In general, sequence alignment algorithms that produce one optimal solution are $O(n^2)$ in time and, as traditionally implemented, are $O(n^2)$ in space, although modern implementations [51] are $O(n)$ in space.

Depending on whether the alignment is local or global, the alignment score is referred to as the *Smith-Waterman score* or the *Needleman-Wunsch score*, respectively. A *global alignment* is an alignment that aligns the entirety of two sequences. A *local alignment* is the alignment of two high scoring subsequences of the original sequences. The score for matching and substituting amino acids are calculated according to predefined transition scoring matrices [52][53] which are empirically derived from manually created alignments of well studied proteins. The matrices contain the log-odds that one amino acid will evolve into another. Insertions and deletions are often referred to as *indels* because an insertion in one sequence is a deletion from the other and vice versa. Indels are represented in alignments as gaps (usually a '-' character) in one or the other sequence so we therefore talk about *gap penalties* when referring to how insertions and deletions are accounted for in an alignment score. Gap penalties are calculated according to an affine function where *(score) = (gap creation penalty) + (gap extension penalty) * (number of gaps).* More complicated functions have been proposed [22], but affine penalties are both easy to understand and computationally tractable. The combination of scoring matrix and gap penalties are referred to as the scoring parameters for an alignment.

The extent to which two sequences align is measured in two ways. The first measure is percent identity (also referred to as sequence identity), which is the number of amino acids in the alignment that align with an identical amino acid in the other sequence divided by the length of the alignment. Despite being intuitive, percent identity is not statistically rigorous. For this, we have the expectation value of the alignment. This is the expectation that an alignment of this quality can be expected to be found in a given database of protein sequences [54]. Expectation is calculated as part of a database

search [2][1] and expectation values are generally presented in terms of the database used to do the search.

Standard sequence alignment algorithms return only one algorithmically optimal solution. This is often a misleading result because in most alignments there are multiple alignments with the same optimal score. For this reason and because algorithmically optimal alignments are known to be incorrect in certain cases (i.e. functional residues not aligned), people have proposed exploring the near-optimal solution space [10]. Near-optimal alignment generation algorithms have been well understood for some time. Waterman and Byers developed an elegant algorithm for enumerating all alignments within a certain distance of optimal [7]. The difficulty with using all near-optimal alignments is that there are so many of them. Even sequences of modest length and similarity can produce many millions of alignments within a neighborhood close to optimal. To accommodate this, Zuker proposed an algorithm based on suboptimal points [10] for generating a diverse sample of near-optimal solutions [8]. This algorithm is based on his work in RNA folding [55]. Shortly thereafter, Saqi and Sternberg proposed another algorithm for generating a sample of alignments with the goal to produce alignments that are different from one another [56] using a method similar to that of Waterman and Eggert [57]. During the traceback phase of the algorithm, the Saqi-Sternberg method penalizes any edges[4] used so that on the next iteration, the alignment is less likely to reuse the edges already part of an alignment. These methods produce samples substantially smaller than Waterman-Byers would generate, usually anywhere from one or two dozen to several hundred.

1.2.2. Near-optimal Alignment Analysis

The use and analysis of near-optimal sequence alignments was detailed in a survey paper by Vingron in 1996. The primary use of near-optimal alignments has been to assess the reliability of different regions of an alignment [10]. Vingron and Argos introduced the notion of robustness [58], which is a measure for each edge in an alignment. The robustness of an edge is the difference between an optimal alignment that includes that edge and the best alignment that does not include that edge. The

[4] In this document, we refer to a pair of aligned amino acids as an edge. This terminology results from considering a set of alignments as a directed, acyclic path graph [9]. Each amino acid aligned with another amino acid or a gap represents one edge in the path graph. A single alignment is represented as one path through the path graph. The arrows in Figure 1.3 point to one edge that represents the alignment of the two amino acids at the opposite ends of the arrows.

greater the difference, the more robust a particular edge is because the more necessary that edge is to produce a high alignment score. Mevissen and Vingron introduced a relatively fast ($O(n^2)$) method for calculating robustness [59]. Their paper discusses the use of robustness as a reliability index. They show that alignment edges with high robustness have a higher probability of being correctly aligned according to a single structural alignment. While making use of near-optimal techniques, their methods are only ever applied to single algorithmically optimal alignments.

Marchler-Bauer et al. [60] performed an analysis that compares the alignment of protein domains in the SMART [61] and PFAM [62] databases[5] with VAST[63] alignments. Their conclusion is that sequence based and structural algorithms compare favorably with only a few common problems. Structural flexibility[6] was shown to cause structural alignments to fail for entire domains, but their research does not show evidence of small failures, that is functional sites misaligning. Their research provides further evidence that regions of low sequence identity are difficult for sequence based alignments to manage.

1.2.3. Near-optimal Alignment Visualization

Visualizations of alignments can aid biologists when investigating alignments and relationships between sequences. Prior to this work, there were two alternative ways of viewing alignments: the pairwise alignment and the path graph.

1.2.3.1. Pairwise Alignments

A pairwise alignment displays in detail one alignment of two sequences. It has been used by biologists for decades and is the de facto standard for displaying an alignment. This method displays one sequence in the row above the other such that the amino acid and gap characters of the two sequences align vertically. Depending on the software and the display capabilities, details such as how matching amino acids are highlighted vary between systems. Figure 1.1 is an example of

[5] SMART and PFAM are alignment databases generated from families of proteins using hidden Markov models.

[6] For several reasons, protein structures are not completely static. First, any proteins naturally change shape, meaning there is often not one fold shape. Second, the error and ambiguity inherent in structure determination [13] means structures can vary slightly.

Figure 1.1. Text-based Pairwise Alignment

An optimal ASCII text pairwise alignment of proteins 1AU8A (NCBI GI: 115725) and 1TGSZ (NCBI GI: 230350). Created using BLOSUM50 -12/-2.

```
                    10        20        30        40        50
1AU8A  ------IIGGRESRPHSRPYMAYLQIQSPAGQSRCGGFLVREDFVLTAAHCWGSNINVTL
             :.::     .. ::    :..  .:    ::: :.  ..:..:::::. :.:.: :
1TGSZ  VDDDDKIVGGYTCGANTVPY----QVSLNSGYHFCGGSLINSQWVVSAAHCYKSGIQVRL
               10        20            30        40        50

           60        70        80        90       100       110
1AU8A  GAHNIQRRENTQQHITARRAIRHPQYNQRTIQNDIMLLQLSRRVRRNRNVNPVALPRAQE
       :  ::.  :...: :.: ..: ::.::. :..:::::..:.  .  :  :  ..:: .
1TGSZ  GEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLNSRVASISLPTSCA
         60        70        80        90       100       110

          120          130       140       150       160       170
1AU8A  GLRPGTLCTVAGWG--RVSMRRGTDTLREVQLRVQRDRQCLRIFGSYDPRRQICVGDRRE
       .   :: : ..:::  . :     :.:. ..  .  : .:   . .     ..:.:  .
1TGSZ  S--AGTQCLISGWGNTKSSGTSYPDVLKCLKAPILSDSSCKSAYPGQITSNMFCAGYLEG
          120       130       140       150       160       170

            180       190       200          210       220
1AU8A  RKAAFKGDSGGPLLCNNVAHGIVSYGKSSGVP--PEVFTRVSSFLPWIRTTMRS-
        : . .::::::..:..  .::::.:.. .    : :.:.: ... ::. :. :
1TGSZ  GKDSCQGDSGGPVVCSGKLQGIVSWGSGCAQKNKPGVYTKVCNYVSWIKQTIASN
          180       190       200       210       220
```

a pairwise alignment generated by text-based software. In this particular representation, the rows representing the actual sequences begin with the sequence names; the numbers above and below the sequences indicate the positions of the amino acids within the sequences; a colon between the two rows of amino acids indicates an identical match at that position; and a single dot indicates a similar match. In the boxed region in Figure 1.1, the amino acids in positions 30-34 (CGGFL) in sequence 1AU8A (NCBI GI: 115725) align with the amino acids in positions 32-36 (CGGSL) in sequence 1TGSZ (NCBI GI: 230350). In this subsequence, the amino acids CGG and L align identically, while the F in 1AU8A is substituted for an S in 1TGSZ.

Pairwise alignments are very effective at communicating the low-level detail of how individual amino acids align. However, a pairwise alignment represents only one alignment. To display a set of alternative alignments, a new pairwise alignment would need to be created for each alternative. While it is trivial to generate a large number of pairwise alignments, it becomes very difficult to

compare differences between alternatives and understand the relationships captured by the set of alignments.

1.2.3.2. Path Graphs

Pairwise alignments excel at providing a detailed view of individual alignments, but they do not provide an overview of the entire set of alignments in one snapshot. An alternative visual representation to many text-based pairwise alignments is the dot plot described by Zuker [8]. See Figure 1.2 for an example. A dot plot places one sequence along the X axis and the second along the Y. If a pair of amino acids align, then a point is plotted at the indices of the amino acids. The advantage of a dot plot display, is that all near-optimal solution can be displayed in a single screen. One problem with dot plots is that there are frequently too many points to discern any alignment shape. A variety of refinements can fix this, for instance only plotting a point if three or more amino acids in a row align. This approach helps, but detailed analysis of how individual amino acids align is very difficult with this representation. Likewise it is impossible to discern how frequently certain sections of the alignments occur within a set.

Naor and Brutlag [9] proposed an improvement to the dot plot that instead of drawing rough points, explicitly draws the alignment path graph. Figure 1.3 is an example of a static path graph generated using Naor and Brutlag's approach. Path graphs are a representation of an entire set of near-optimal alignments. Path graphs place one sequence along the horizontal axis and the other along the vertical axis. A diagonal line on the graph between the sequences, called an edge, indicates that the amino acids at the respective indices align with one another. A vertical edge indicates that a gap has been inserted into the horizontal sequence and a horizontal edge indicates that a gap has been inserted into the vertical sequence. The resulting set of edges form a directed, acyclic graph where one path through the graph (from top left to bottom right) represents one alignment. The benefit of the path graph is that all possible alignments in a set can be displayed at once providing an overview of a set of alignments. Reliably aligned sections are also easy to see because those sections have relatively few possible paths through those sections of the graph (e.g. the bottom right section of Figure 1.3), while variable sections have many possible paths (e.g. the top left section of Figure 1.3).

Figure 1.2. Dot Plot

Dot plot of proteins 1AU8A (NCBI GI: 115725) and 1UVTH (NCBI GI: 2781297). 1AU8A is along the top edge and 1UVTH is along the left edge. A point on the plot indicates that one or more amino acids align within a sliding window around that point. The darker the point is, the higher the score for that window is. It should be clear from this image, where the amino acids aren't even displayed, that anything other than general observations about the quality of the alignment is very difficult. Created using *dotter* with the BLOSUM62 scoring matrix, K=0.141, $\lambda = 0.319$, and a window size of 16 [64].

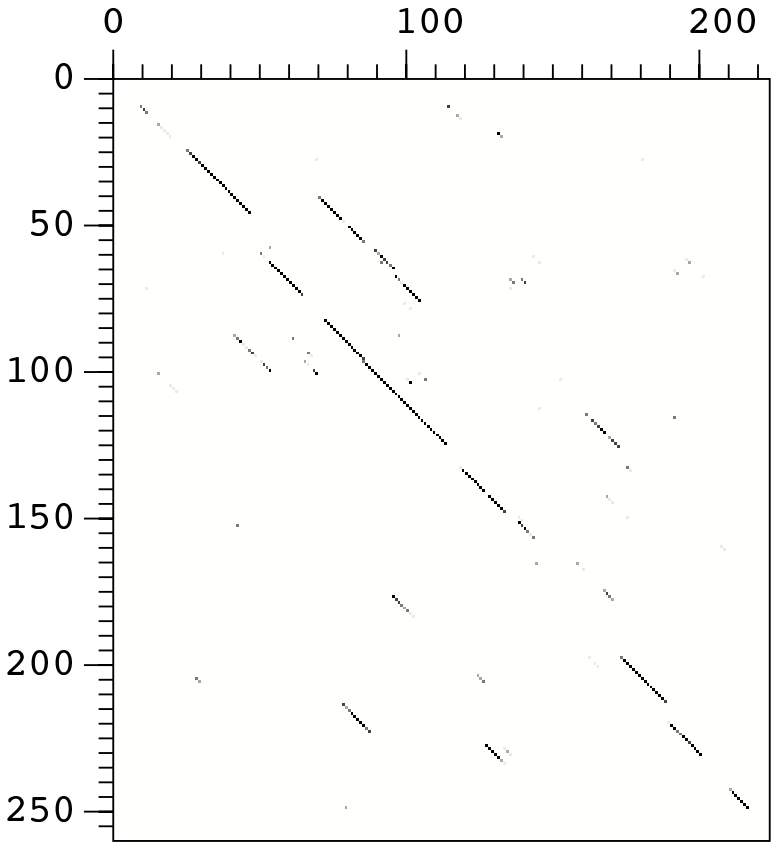
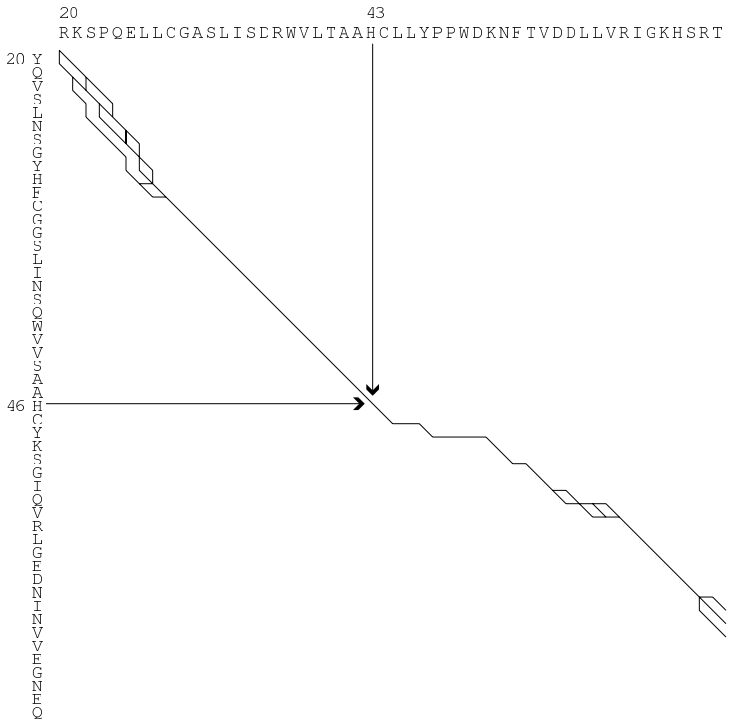


Figure 1.3. Path Graph

An example of a path graph representation of a set of alternative alignments. A vertical edge represents a gap in the top sequence, a horizontal edge represents a gap in the left sequence and a diagonal edge indicates a match. The section pointed to by the arrow indicates a section of the alignment thought to be reliable as there is only one path through the section. The upper left hand section of the alignment shows multiple paths indicating that there are several possible alignments of the sequences around position 25 in each sequence. While path graphs provide substantially more information than dot plots, it is still difficult to do amino acid level comparisons because the sequences are relatively far apart on the screen. The plot is a partial path graph of multiple near-optimal alignments of 1TGSZ (vertical, NCBI GI: 230350) and 1UVTH (horizontal, NCBI GI: 2781297) generated with the SUBOPT program [9] using the PAM250 matrix and a gap penalty of -3 per residue.



However, the path graph suffers from a number of drawbacks. First, all previous path graph software created a static display and static path graphs of large sequences are somewhat unwieldy. This is because path graphs for sequences longer than approximately 80 amino acids (effectively all proteins) are multi-scale (i.e. too large to entirely fit in one view) [65]. Users can choose only between a broad overview display where no detail is discernible and a detailed view where no sense of context is available. Even if the path graph being displayed is at a relatively detailed

resolution (like Figure 1.3), amino acid-by-amino acid comparison is difficult because the sequences are positioned perpendicular to one another.

A more scientifically problematic issue is that not all visible paths are actually solutions that fall within the near-optimal neighborhood. While each edge in the path graph is guaranteed to be part of at least one near-optimal alignment, an arbitrary path through the set of edges that comprise the graph is not guaranteed to be near-optimal. Thus, it is impossible to know which paths are actually included in the set of alignments. In addition, there is no way to discern which paths occur more frequently in variable sections of the alignments.

Figure 1.3 is a detailed view of a path graph and demonstrates many of the qualities of path graphs. Rather than the complete alignment, only positions 153-202 of an alignment of the proteins 1AU8A (vertical sequence on left) and 1TGSZ (horizontal sequence at top) are displayed. Even with a detailed view, it is very difficult to determine which amino acid in the left sequence aligns with which position in the top sequence. Again, note that this is not the complete set of alignments. If all 224 amino acids in 1AU8A had been displayed along with all 229 amino acids of 1TGSZ, it would be nearly impossible to see any detail whatsoever.

Thus, traditional pairwise alignments excel at allowing detailed analysis of individual alignments. However, they fail to provide any sort of overview capacity. Path graphs provide excellent overviews of sets of alignments, but fail to provide detailed perspective on the alignments, lose information by combining all alignments into one display, and do not distinguish the paths within the graph that are valid alignments in the near-optimal neighborhood.

Chapter 2

Comparison of Near-optimal Alignments with Structural Alignments

2.1. Introduction

Finding protein structures is much more expensive than finding protein sequences so there are fewer known protein structures than protein sequences. As a consequence, sequence based alignments remain an important tool for constructing homology models between sequences. However, sequence based alignment algorithms have difficulties constructing high quality alignments for sequences with less that ~40% sequence identity. To solve this problem, efforts have been made to assess whether sets of near-optimal alignments can improve the sequence alignment with respect to a structural alignment [58][10]. This can be done in two general ways: 1) by searching for an alignment that is closer to the structural alignment than the optimal one, and 2) by evaluating each pair of aligned residues, assigning them a reliability score, and using this reliability score to predict structural relevance.

The first method was used by Jaroszewski, et al. [66]. They examined alternative alignments generated in two ways: from a near-optimal alignment generation algorithm and by varying scoring parameters. They demonstrated that there is frequently an alignment in these sets that is closer to the structural alignment than alignments with the highest alignment score. They concluded that the two methods of generating alternative alignments have complementary (as opposed to redundant)

information, since the union of the two sets yielded many more alignments that matched a structural alignment than either of the single sets.

The implicit assumption of [66] and other work that compares sequence based alignments with structural alignments [6][67][58] is that the structural alignment is correct. In fact, there is no way to ensure an optimal structural alignment, and different algorithms sometimes produce quite different alignments. The hypothesis is that the differences between structural alignments are small with respect to the differences between sequence based and structural alignments, but this has not been examined quantitatively. This chapter describes research that tests this hypothesis. We compare four alternative structural alignments with sets of alternative sequence based alignments generated by varying scoring parameters and by sampling the near-optimal alignment space. The goal of this effort is to characterize how sets of near-optimal alignments compare to sets of structural alignments. This understanding provides the foundation for extracting information about structural alignments from sets of near-optimal alignments.

2.2. Methods

The pairs of proteins used to generate alignments in this research were domains from the same homologous superfamily in the CATH [68] structural domain database. The goal was to create a range of proteins from pairs with certain homology to pairs where homology determination with standard alignment based techniques was much more difficult. We selected pairs with a range of similarities: highly similar (SSEARCH [69] $0 < E() \leq 10^{-13}$), similar ($10^{-13} < E() \leq 10^{-5}$), barely statistically significant ($10^{-5} < E() \leq 10^{-2}$), and not statistically significant ($E() > 10^{-2}$). The average percent identities within each group, from lowest expectation to highest were 48.2%, 26.5%, 22.5%, and 20.1%, respectively. A structurally diverse set of protein pairs was selected from CATH, including members from the all α, all β, and mixed α/β structural classes. In all, there are 94 pairs from 39 superfamilies. The complete list of domains, expectation values, and percents identity can be seen in Table A.1 in Appendix A.

To compare the near-optimal sequence alignments to the structural alignments, we first had to generate the sets of near-optimal alignments. To limit the sizes of the sets of near-optimal alignments, we used the Zuker [8] algorithm because it ensures a diverse sample by forcing all near-optimal edges to be included in at least one alignment while at the same time preserving information about which edges within the set of all edges are used most frequently. We used the Waterman-Byers algorithm [7] in the particular case where all optimal alignments were desired.

To generate the structural alignments we used the Dali, LSQMAN, CE, and Matras algorithms. We chose these methods because they represent diverse techniques for building structural alignments and we had access to implementations of all four algorithms. We used the stand alone version of the Dali program [21], called DaliLite [70], obtained from the web site [71], with default parameters. We used the Linux version of the Combinatorial Extension (CE) program [72], obtainable at [73], also with default parameters. We used the Structal method as implemented in the LSQMAN program [74] from the Uppsala Software Factory: [75]. Specifically, we used the Fast Force and Improve commands to get an initial alignment, the DP command to implement the dynamic-programming method of Levitt and Gerstein [76], then the Global command to calculate the statistics based on the Gerstein and Levitt structural similarity score [76]. For Matras, we used the Linux version of the program provided by the authors [77] with default parameters.

We used multiple methods to compare different alignments with one another. There are two general ways in which this can be accomplished. The first is to calculate individual metrics for each alignment. Individual metrics can be calculated using only the alignment in question and algorithm parameters (e.g. the Needleman-Wunsch sequence alignment score). Pairwise metrics are dependent on a separate alignment, generally a "gold standard," to determine a comparison score. An example of a pairwise metric is the number of amino acids that are aligned identically by two alignments. Individual metrics will remain constant for a given alignment whereas a pairwise metric will change relative to the comparison alignment.

We used two individual scoring metrics in this analysis: a sequence-based score and a structure-based score. The sequence-based method is the Needleman-Wunsch global alignment score, using a selection of gap penalty and scoring matrix combinations. The structure-based method is the Structal

score introduced by Levitt and Gerstein [76]. LSQMAN can accept an input alignment and calculate the corresponding Structal score for the two structures (using the Xalign option), so we can calculate the Structal score for alignments produced by other sequence and structure alignment programs.

The pairwise metric used was the shift score described by Cline, et al. [78]. The shift score was chosen over statistics that count the number of shared edges between alignments. Comparing the percentage of residues identically aligned between two alignments is appealing in its simplicity. However, these values penalize an alignment the same whether a pair of residues is very close to being identically aligned, or if the pair is wildly divergent as long as the same number of amino acid pairs align. The shift score is a global measure of similarity between two alignments that quantifies this deviation and accounts for it in the final score. Because pairwise metrics require a benchmark alignment, we choose one alignment to act as the "gold standard" against which all other alignments are compared. We calculated shift scores using each of the four structural alignments and the near-optimal alignment with the highest structural score acting as the gold standard.

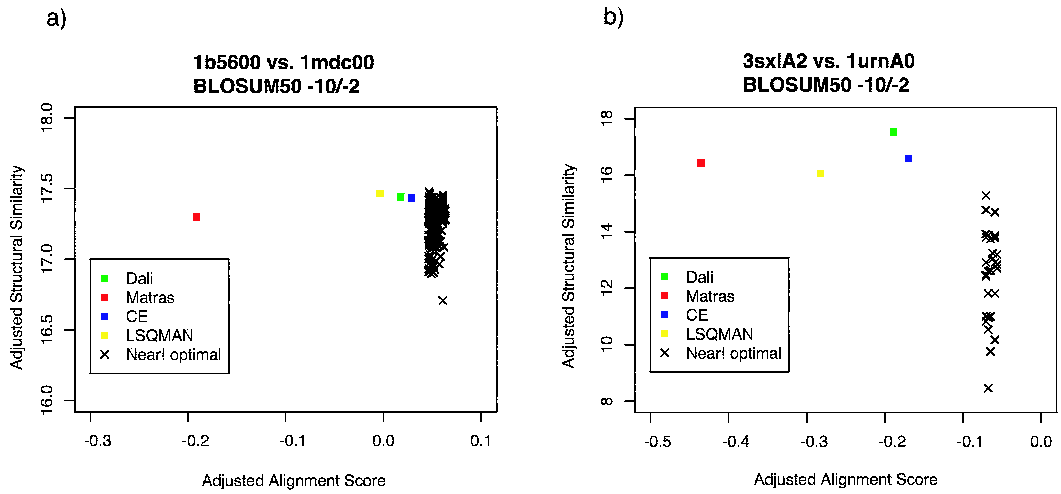
2.3. Results

Figures 2.1a and 2.1b contain the Needleman-Wunsch scores of different sets of near-optimal alignments along with the Needleman-Wunsch scores of the four structural alignments (X-axis) plotted against the structural similarity scores of those same alignments (Y-axis). Each point represents one alignment. The black X's represent the near-optimal alignments and colored shapes represent the various structural alignment algorithms.

Figure 2.1a is an example where the near-optimal alignment algorithm produces alignments that are as good as or better, from a structural perspective than the structural alignment algorithms according to the structal score [76] (see Appendix A for sequence details). This result motivates the remainder of this research. However, Figure 2.1a is for only one alignment and Figure 2.1b demonstrates, near-optimal alignments are not always better. As expected, each of these plots shows that the set of near-optimal alignments contains one or more alignments with the highest possible

Figure 2.1. Scatter Plots of Near-optimal Alignments with Structural Alignments

Scatter plots of sets of near-optimal alignments and structural alignments for one pair of sequences and one scoring parameter combination. The X-axis is the adjusted alignment score and the Y-axis is the adjusted structural similarity score. The colored square icons represent the structural alignments and the x icons represent the set of near-optimal alignments. Figure a shows a pair of proteins (1b5600 vs. 1mdc00 - see Appendix A for sequence details) where the set of near-optimal alignments contains alignments with structural similarity scores greater than any of the structural alignments (Y-axis), implying that those near-optimal alignments are better structural alignments. Figure b shows a different pair of proteins (3sxlA2 vs. 1urnA0 - see Appendix A for sequence details) where no near-optimal alignment improves upon the structural alignments.



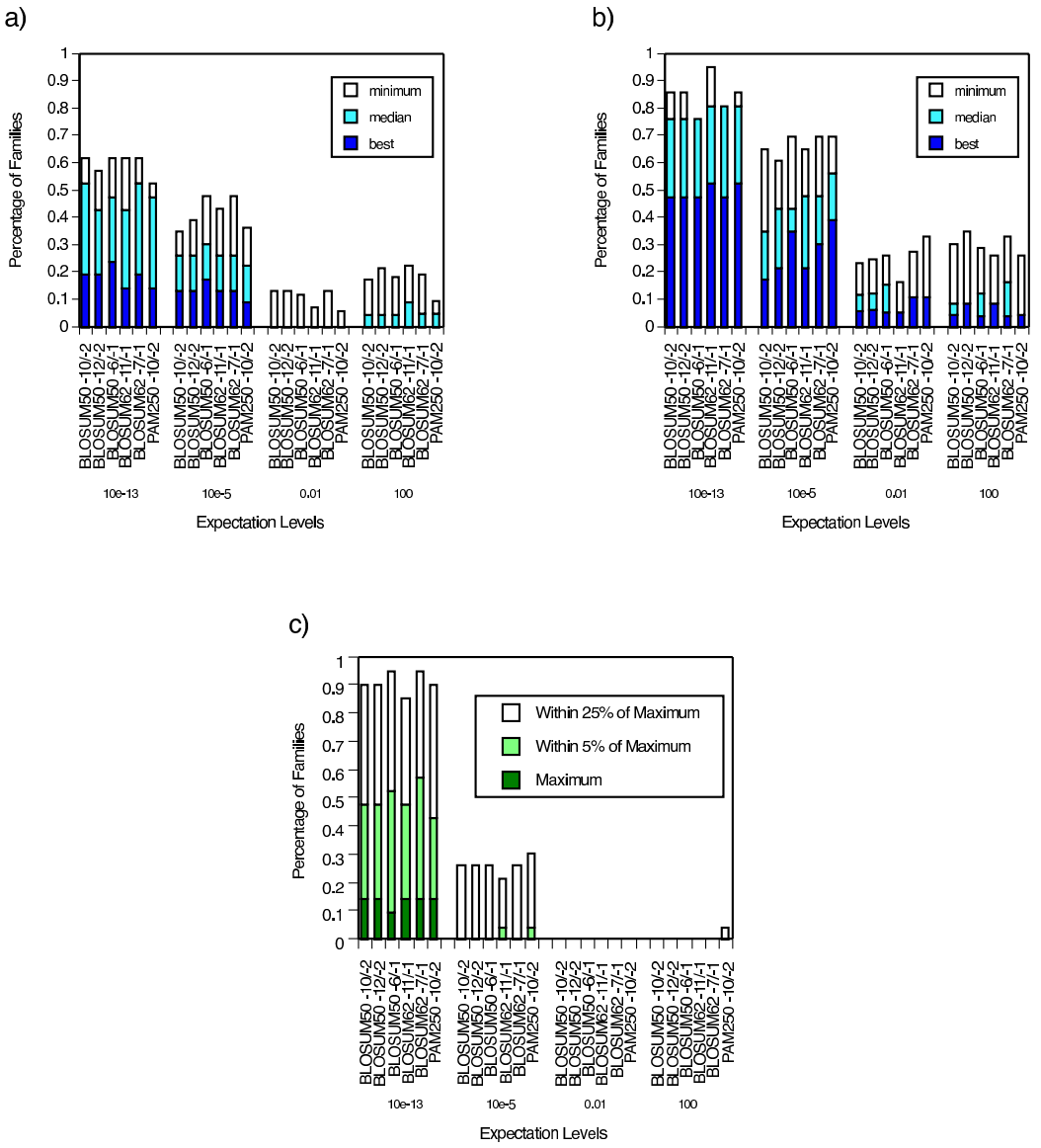
Needleman-Wunsch score. Figures 2.2a, b, and c illustrate how near-optimal alignment quality relates to structural alignment quality.

Figures 2.2a and 2.2b summarize the results found in Figures 2.1 for all 94 pairs of alignments and all scoring parameters. Figure 2.2a is for the results of a near-optimal neighborhood of 95% of optimal and Figure 2.2b is for the results of a near-optimal neighborhood of 75% of optimal. The X-axis represents expectation and has been grouped into the four levels of expectation used in this analysis. Each level cluster along the X-axis has six columns. Each column represents one combination of scoring matrix and gap penalties. The Y-axis of each plot represents the percentage of families (pairs of aligned proteins at that level of expectation) that meet the criteria specified by the column shading. The different shadings within the columns represent different thresholds at which one or more alignment within the set of near-optimal alignments meets a specified criterion. The lightest shading (minimum), meaning the highest point of each column, represents the percentage of families where at least one near-optimal alignment has a better structural similarity score than the structural alignment with the lowest structural similarity score. The next gradation (median) represents the percentage of families with at least one near-optimal alignment with a structural similarity score that is better than the average structural alignment score. The final gradation (best) represents the percentage of families with at least one near-optimal alignment better than the best structural alignment score.

The summaries in Figures 2.2a and 2.2b aggregate this information for our set of protein families. Figure 2.2b shows that for a broad range of proteins, near-optimal alignments have a better than 60% chance of producing alignments of comparable quality to structural alignments for alignments for a neighborhood of 75% of optimal and with expectation values less than 0.01. That chance is approximately 30% if the neighborhood is narrowed to 95% of optimal. The figures demonstrate that near-optimal solutions are often as good or better than structural alignments. This information provides evidence that near-optimal alignments can be used to improve alignments for sequences without solved structures. The differences between neighborhood size apparent in Figures 2.2a and 2.2b can also be used to help estimate the size of the neighborhood necessary for analysis. The larger the neighborhood is, the more likely it is to find a better near-optimal alignment.

Figure 2.2. Structal Score Summaries

Summary plots of the scatter plots seen in Figure 2.1, for all protein families studied. The X-axis represents the four levels of expectation and is clustered by scoring matrix combination with one column representing one expectation/scoring parameter combination. The Y-axis of each plot represents the percentage of families that meet the criteria specified by the column shading. The expectation ranges of the clusters are SSEARCH [69] ($0 < E() \leq 10^{-13}$), ($10^{-13} < E() \leq 10^{-5}$), ($10^{-5} < E() \leq 10^{-2}$), and ($E() > 10^{-2}$). The corresponding sequence identity thresholds are 48.2%, 26.5%, 22.5%, and 20.1%. In Figures a) and b) the lightest shading represents the percentage of families where at least one near-optimal alignment has a better structural similarity score than the worst structural alignment. Subsequent gradations represent the percentage of families with near-optimal alignments better than the median and best structural alignments. Figure a) plots this information for sets of near-optimal alignments built with a neighborhood of 95% of optimal. Figure b) plots this information for sets of near-optimal alignments built with a neighborhood of 75% of optimal. Figure c) represents different information than Figures a) and b). The Y-axis represents the percentage of families with at least one structural alignment with a sequence alignment score within the specified percentage of the optimal alignment score.



Figures 2.2a, 2.2b, and 2.2c also provide evidence that selection of scoring matrix plays a relatively small role in the quality of the sequence based alignments created. This is seen in the small variation between results for the same expectation levels in Figures 2.2a and 2.2b and is supported by χ^2tests of the different levels. For the 75% neighborhood and the lowest expectation level, the respective χ^2statistics for the minimum, median, and best percentages of families are 0.0236, 0.0043, and 0.0061, each with five degrees of freedom and each with p-values of 1. The results for other levels are similar. From this we can conclude that the selection of scoring matrix and scoring parameters play little or no role in the quality of sequence based alignments generated.

While Figures 2.2a and 2.2b provide a perspective on the sets of near-optimal alignments using a structural alignment score, Figure 2.2c provides a perspective of the structural alignments from the perspective of the Needleman-Wunsch score. The axes in Figure 2.2c are the same as for Figures 2.2a and 2.2b. The different shadings represent the percentage of families that have a structural alignment within a certain threshold of the optimal alignment score. The lightest shading and highest point of each column represents the percentage of families with at least one structural alignment within 75% of optimal. The next gradation represents the percentage of families with at least one structural alignment within 95% of optimal and the last gradation represents the 100%(optimal) threshold. Figure 2.2c provides reassurance that structural alignment algorithms produce results that are within a reasonable neighborhood of the optimal Needleman-Wunsch score.

Figure 2.3 summarizes the results of pairwise comparisons of sets of near-optimal alignments with structural alignments using the shift score [78]. Each column along the X-axis represents one pair of sequences that are aligned. The Y-axis represents a scaled shift score (exponentiated with base 100), which allows the display of shift scores between 0.8 and 1 to be less cluttered. The blue X icon represents the "gold standard" against which all other alignments for this pair of sequences was evaluated, in this case the Dali alignment. Dali was chosen arbitrarily (although Dali is perhaps the most widely used structural alignment algorithm) and the results with different gold standards are substantially similar. The red points represent the shift scores of the other structural alignments and the box plot represents the distribution of near-optimal alignments. In Figure 2.4, there is no blue icon (structural gold standard), rather the near-optimal alignment with the highest structal score is used as the gold standard. The neighborhood for the near-optimal alignments is 75% and the

Figure 2.3. Shift Score Summary Box-plots With Dali Gold Standard

Box-plots of the shift scores calculated for each near-optimal alignment and each structural alignment relative to the Dali alignment for each pair of proteins. The four different plots represent the four levels of expectation (the corresponding sequence identity thresholds are 48.2%, 26.5%, 22.5%, and 20.1%). The near-optimal neighborhood is 75% and the alignment scoring parameters are BLOSUM50 -10/-2. The families are ordered such that those families where the near-optimal box-plot intersects at least one structural alignment are to the left and highlighted with gray. Those that do not intersect are to the right. They are subsequently ordered by the minimum shift score, either near-optimal or structural.

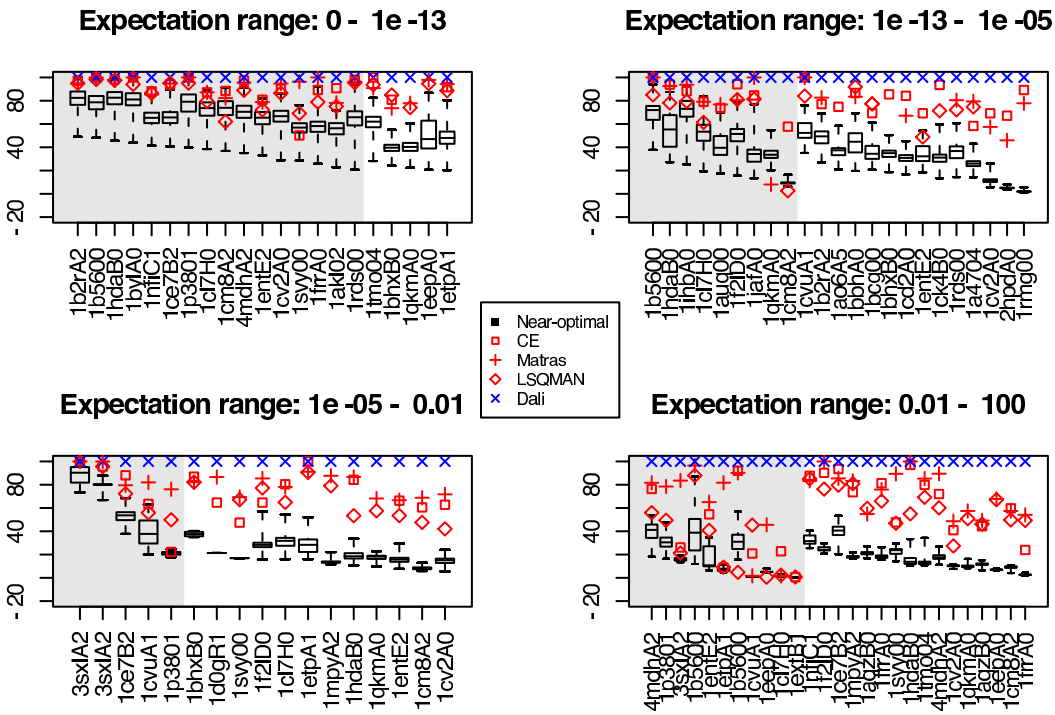
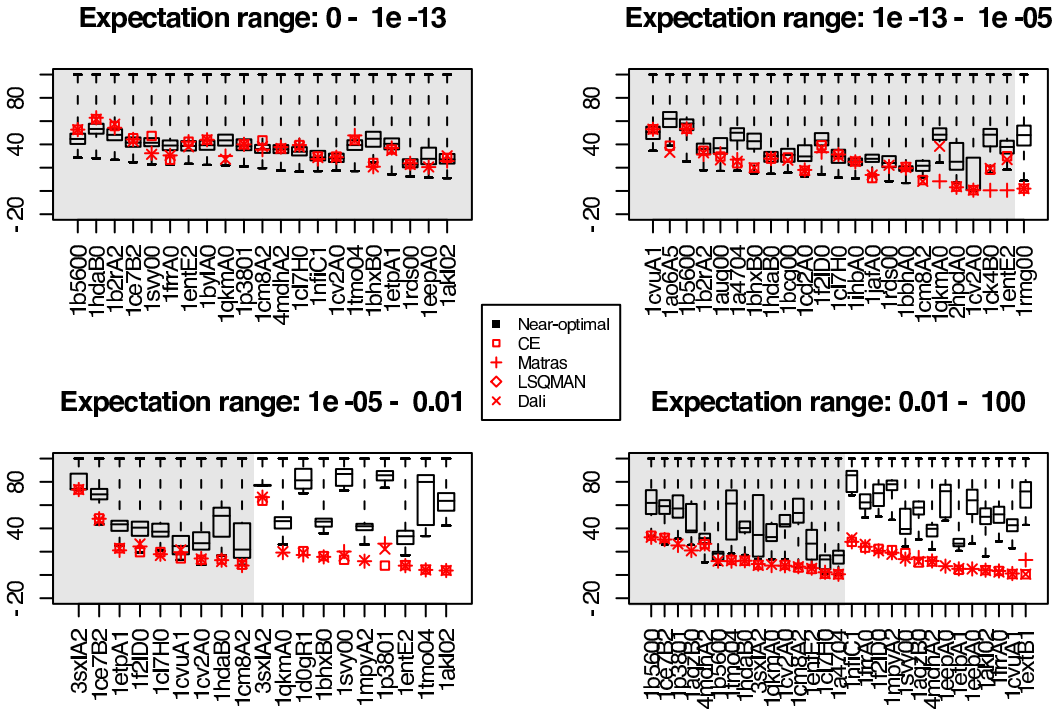


Figure 2.4. Shift Score Box-plots With Near-optimal Gold Standard

Box-plots of the shift scores calculated for each near-optimal alignment and each structural alignment relative to the near-optimal alignment with the highest structural similarity scoring for each pair of proteins. The families, near-optimal neighborhood, scoring parameters, expectation ranges, and ordering are the same as Figure 2.3.



alignment scoring parameters are BLOSUM50 -10/-2. The families are ordered such that those families where the near-optimal box-plot intersects at least one structural alignment are to the left and highlighted with gray. Those that do not intersect are to the right. They are subsequently ordered by the minimum shift score, either near-optimal or structural.

Figures 2.2 and 2.4 capture the variation within sets of near-optimal alignments in relation to the variation within sets of structural alignments. These figures show that while structural alignments are generally closer to another structural "gold standard" than near-optimal alignments, and near-optimal alignments are generally closer to one another, the solution spaces overlap in many cases. Even in the cases where there is no overlap, the figures show that in most cases the range of near-optimal alignment shift scores is close to the structural alignments. These plots provide clear evidence that near-optimal alignment space is not vastly different from structural alignment space. This result motivates our research into how to more effectively support the use of the information found in sets of near-optimal alignments.

2.4. Conclusions

This research demonstrates that near-optimal alignments compare favorably to structural alignments. We first note that sets of near-optimal alignments with reasonably sized neighborhoods often contain alignments with structural similarity scores that are better than structural alignments. As expected, this intersection increases as sequence identity increases. However, we show that it also occurs in pairs with lower percent identity. These results confirmed our expectations that near-optimal alignments could provide useful structural information.

This research also shows that the alignment solution space defined by sets of near-optimal alignments often intersect the solution spaced defined by structural alignments for the same proteins. This result contradicts the hypothesis that the variation between structural alignments is less than the variation between structural and sequence based alignments. It also demonstrates the relatively high variation between structural alignments and shows that the improvement of structural alignments

over sequence based alignments is generally small and does not always improve on a sequence based alignment.

Chapter 3

Predicting Structural Alignment Significance with Sets of Near-optimal Alignments

3.1. Introduction

This chapter involves the comparison of pairs of aligned amino acids in sets of alternative alignments with pairs of aligned amino acids in sets of structural alignments. The goal is to use the information found in sets of near-optimal alignments to assist in the construction of structural similarity models. Specifically, we use statistics about aligned pairs of amino acids from sets of near-optimal alignments to predict whether the same pair of amino acids is also aligned in one or more of the structural alignments.

Our work builds upon the work of Mevissen and Vingron [58] in which they introduce an edge reliability index called robustness. The robustness of an edge is a function of a set of near-optimal alignments and is a measure of the degree to which a particular edge contributes to the similarity score for a particular alignment. Specifically, robustness is the difference in alignment scores between the highest scoring alignment that includes the edge in question and the highest scoring alignment than excludes the edge. The greater the difference is, the more robust the edge is and the more important the edge is to the overall alignment score. Mevissen and Vingron demonstrated that the robustness of an edge accurately predicted whether the edge was also aligned in the structural alignment (see Figure 3.12).

An alternative measure of edge quality is the frequency that an edge occurs within a set of near-optimal alignments [11]. The expectation is that the more frequently an edge occurs within a set of alignments, then the more important that edge is to the alignment of the two proteins. The frequency of an edge is calculated as the number of near-optimal alignments that the edge occurred in divided by the total number of near-optimal alignments for that pair of proteins and scoring parameters.

A third technique for assessing the quality of an edge is to calculate the maximum bits-per-position score for the given edge. The bit score is a value derived from the alignment score that takes the statistical properties (common referred to as Altshul-Gish statistics) of the alignment scoring parameters into account [54]. The benefit of the bit-score is that it allows alignments created with different scoring parameters to be compared. Bits-per-position is simply the bit score divided by the length of the alignment. To calculate the maximum bits-per-position score we compare the bits-per-position score for each alignment that includes a particular edge and assign the maximum value to that edge. The benefit of maximum bits-per-position is that the bits-per-position score is a function of the entire alignment, whereas robustness and frequency are edge specific. This means that maximum bits-per-position captures the overall quality of the best individual alignment containing that edge while frequency and robustness reflect the set of alignments as a whole.

The goal of this chapter is to build a probabilistic model that considers whether robustness, frequency, and maximum bits-per-position can predict whether an edge is in a structural alignment and do so more effectively than robustness alone. Because the response variable in this case is dichotomous (whether or not an edge is aligned structurally), we use logistic regression [79] to construct a model. The logistic regression model is a generalized linear model in which a linear combination of predictor variables estimate the response variable through a logit link function:

$$\log(\frac{\pi(x)}{1-\pi(x)}) = B_0 + B_1x_1 + \ldots + B_nx_n \tag{3.1.1}$$

where

$$\pi(x) = \frac{e^{B_0+B_1x_1+\ldots+B_nx_n}}{1+e^{B_0+B_1x_1+\ldots+B_nx_n}} \tag{3.1.2}$$

is the logistic distribution probability density function with x_i representing the predictor variables and B_i representing the model parameter estimates. The result of a logistic regression model is a function that calculates the probability of a response given specific inputs. In this case, we develop a function that estimates the probability that a particular edge is part of a structural alignment.

This research explores the relationship between sets of near-optimal alignments and alternative structural alignments. We compare alternative structural alignments with each other and with sets of near-optimal alignments. We demonstrate that near-optimal alignments provide a foundation from which to explore structural alignments. Based on this information we construct a probabilistic model that uses the information contained in sets of near-optimal alignments to predict whether or not specific amino acid pairs (i.e. edges) are likely to be included in structural alignments.

3.2. Methods

The same protein families used for the structural comparison in Chapter 2 were used for constructing the logistic regression model. The data were partitioned into test and training data sets. The data were partitioned along family lines, meaning proteins from one family were used to either train the model or test the model, not both. This was done to most realistically portray real-world alignment situations where our model is unlikely to be used to assist is aligning proteins found in families used to construct the model. The families were distributed so that roughly the same levels of expectation were represented between models.

For the logistic regression we built models by varying three factors: alignment scoring parameter combinations, the near-optimal neighborhood size, and the sample size. The scoring parameter combination options were BLOSUM50 -10/-2, BLOSUM50 -12/-2, BLOSUM62 -11/-1, and the three combined (four levels). The possible neighborhood sizes were: all optimal alignments, those alignments with scores within 95% of optimal, and those alignments with scores within 75% of optimal (three levels). The sizes of the samples (i.e. number of edges) for each combination of neighborhood and scoring parameter combination can be seen in Appendix A in Tables B.1 and B.2. Given those values, we chose sample sizes of 500, 1000, 2000, 5000, and 10000 edges (five

levels). For the sets of optimal alignments, instead of 10000 edges, we used all available. Taken together, the levels created by these variables resulted in 60 possible models (four scoring parameter combinations, three neighborhood sizes, and five sample sizes).

We developed models for the four possible response variables. The response variables reflected the number of times a particular edge occurred within the set of structural alignments. The first set of models used a threshold of one out of four structural alignments meaning if an edge appeared in one or more structural alignments, the response variable was coded as 1 and 0 if the edge occurred in no alignments. The second set of models was developed using a response variable defined by a threshold of two out of four structural alignments. The third set of models used a response variable defined by a threshold of three out of four structural alignments and the fourth set of models only considered the edge structurally relevant if it was included in all four structural alignments. Combining the four different response thresholds with the 60 possible models resulting from the model factors, resulted in 240 models.

The predictors that were investigated for these models were the robustness of the edge, the frequency of the edge and the maximum bits-per-position of the edge. Each of the three predictor variables was normalized between 0 and 1.

To choose which of the three predictor variables were appropriate for consideration in our model, we undertook two analyses. The first was to build single parameter models using each of the possible predictors. Following the strategy described in [79], any variable with a p-value less than 0.25 should be considered for inclusion in the model. The 0.25 threshold is deliberately large to allow variables that individually may only be significant when interacting with other variables to be included. Single variable models were built using the scoring parameter/sample size combinations described above. We built models using a threshold of two (i.e. the edge was in at least two structural alignments). The second strategy was to use a stepwise model construction technique. Parsimony was measured with the Akaike Information Criterion[1] (AIC) [80]. The same sample sizes and scoring parameters were used for the stepwise analysis as for the single variable models.

[1] AIC is an information theory based, relative goodness-of-fit statistic used to compare alternative models based on a sample of data. The statistic produced attempts to balance the complexity of the model (number of parameters) with the fit to the data in an effort to prevent over-fitting. For our purposes, it is simply a statistic used to compare alternative models.

Once the predictor variables were selected, we performed a preliminary analysis to determine which of the model factors should be used for the final model construction. This was accomplished by building regression models for each model factor combination (60 models) and each response variable and analyzing the predictor coefficient estimates to determine if a relationship existed between the coefficients of models built using different factors.

Once the variable selection and model factor analysis were complete, we began the final model construction and analysis. The process for building our model includes the construction of four alternative models and then comparing the results to select the final model. The first model includes all three predictor variables, which we refer to as main effects. The next model includes interaction terms. The third model includes the main effects and second order polynomials of the main effects. The final model includes the main effects and second order polynomials of the main effects along with all of the interaction terms. The intercept is used in all models during final model construction.

The alternative models are compared in two different ways. First, they are compared by model deviance (the lower the deviance, the better the model). Model deviance is compared using the χ^2 test. The second test is to compare the performance of the models when classifying the test data set using the area under the ROC curves as the metric. The larger the area under the ROC curve is, then the better the classification performed by the model.

One underlying assumption of logistic regression is that the predictor variables are mutually independent (to avoid multicollinearity). Correlation analysis of the predictor variables was performed using a sample size of 5000 with edges drawn from the training set with a combination of the three scoring parameter possibilities (BLOSUM50 -10/-2, BLOSUM50 -12/-2, BLOSUM62 -11/-1). The estimated correlation coefficients and associated statistics can be seen in Table 3.1. These results show that there is some correlation between the predictor variables. However, the correlations are small in absolute value, so we do not have a strong reason to suspect that multicollinearity will be problem. A plot of the sample data can be seen in Figure 3.1.

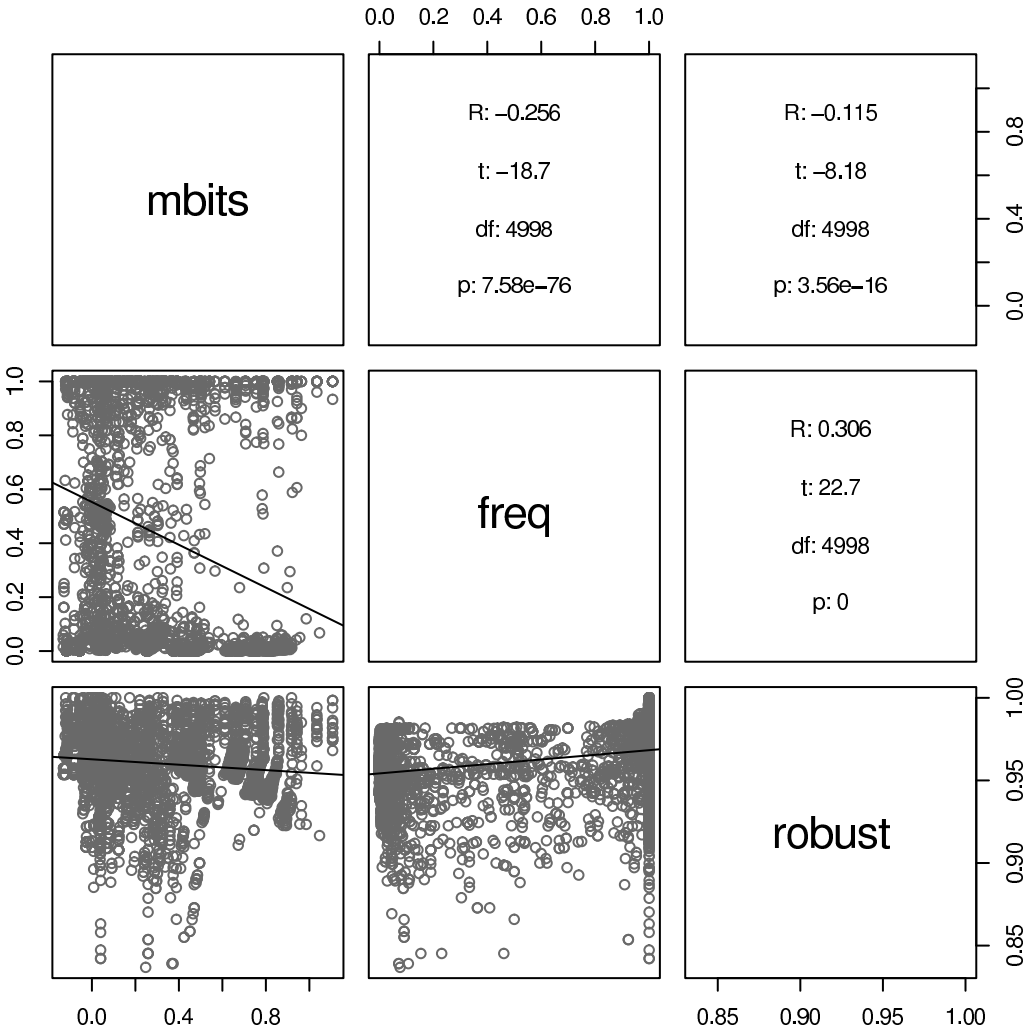
Table 3.1. Predictor Correlation Analysis Results

Predictor correlation analysis results. The results are from a sample containing 5000 edges drawn from the training set with a combination of the three scoring parameter possibilities (BLOSUM50 -10/-2, BLOSUM50 -12/-2, BLOSUM62 -11/-1).

	R	t	d.f.	p-value
frequency vs. max. bits-per-position	-0.256	-18.7	4998	$< 7.58 * 10^{-78}$
frequency vs. robustness	0.306	22.7	4998	0
robustness vs. max. bits-per-position	-0.115	-8.18	4998	$< 3.56 * 10^{-16}$

Figure 3.1. Predictor Correlation Plots

Correlation plots of possible logistic regression predictor variables. The plots in the lower left hand corners are scatter plots of the different predictor variables against one another. The top right hand corners are the correlation statistics for the respective predictor variables. The results are from a sample containing 5000 edges drawn from the training set with a combination of the three scoring parameter possibilities (BLOSUM50 -10/-2, BLOSUM50 -12/-2, BLOSUM62 -11/-1).



There is no universal goodness-of-fit measure for logistic regression.[2] The alternative recommended in the comprehensive comparison of goodness-of-fit tests for logistic regression by Hosmer, et al. is the use of a smoothed residual test statistic [82]. The p-value for this statistic is calculated in terms of the chi-squared distribution.

The logistic regression was performed using the R statistical computing system [83]. Models were built using default parameters for the *glm* (generalized linear model) function with the logit link function and the *lrm* (logistic regression model) function from the *Design* library [84] (see Appendices D and E for details).

3.3. Results

The comparison of near-optimal and structural alignments demonstrates that near-optimal alignments can provide information about structural alignments because there is overlap between their respective solution spaces. Not yet addressed, however, is how this information can be used in an effective manner. The logistic regression model provides one answer to this question.

3.3.1. Preliminary Variable Selection and Model Factor Analysis

The variable selection analysis showed that all possible predictor variables should be included in the final model. Single variable logistic models were built using each predictor variable, each scoring combination, and each sample size. The structural threshold was limited to two out of four structural alignments for an edge to qualify as structural, leaving us with 60 models for each predictor variable. Of the 60 models, 67% (40/60) of the frequency models, 93% (56/60) of robustness models, and 63% (38/60) of maximum bits-per-position models have p-values less than 0.25. These data provide no strong evidence that any one variable should be excluded from the analysis.

[2] The most common approach to assessing goodness-of-fit is by calculating the Pearson chi-squared statistic or the equivalent model deviance likelihood ratio [81]. However, these statistics are only valid when the number of covariate patterns (specific combinations of predictor variable values) in the sample data is substantially less than the number of samples. This is a common situation with categorical predictor variables, however it is much less common with continuous predictor variables. For models with continuous predictor variables the frequency of covariate patterns will generally be very low, which means the assumptions necessary behind the Pearson chi-squared statistic do not hold. As a consequence, a different test must be used.

An ancillary benefit of this analysis is the apparent trend showing that models built using only optimal alignments or with a neighborhood of 75% of optimal fare worse than models built with a neighborhood of 95%. This is reflected in the twofold and threefold increase in models with high p-values between 95% and optimal neighborhoods and 95% and 75% neighborhoods. Of the 180 single variable models constructed, a total of 46 had p-values greater than 0.25. Of these 46 models, 16 were from the optimal neighborhood, seven were from the 95% neighborhood, and 23 were from the 75% neighborhood. The likely reason for this is that if the neighborhood is too small, then not enough structural edges are included in the set to create an accurate model and if the neighborhood is too large, then the number of structural edges relative to the overall number of edges is too small, which again means there are not enough data with which to build an accurate model. This suggests that our model building efforts should be focused on the 95% of optimal neighborhood.

The alternative approach to variable selection was the stepwise construction of models where predictor variables were successively included based on minimizing the model AIC. Of the 60 models analyzed, each had a different combination of predictor variables selected for inclusion. Two models included only frequency, four models included frequency and maximum bits-per-position, four models included frequency and robustness, and 50 models included frequency, robustness, and maximum bits-per-position. This indicates that all predictor variables should be included in our analysis. This conclusion is further supported by close examination of the steps taken in the model building. Variable omissions occurred because of very small changes in AIC scores. The average difference between the entering AIC and the smallest AIC for particular models was 0.25% (with AIC scores ranging in size from 68 (small sample) to 7587 (large sample)). This tells us that the difference between possible variable selection strategies is minimal, which implies that all variables can be included in the model without significant detriment. The stepwise analysis confirms the result produced by single variable model analysis. We therefore include all possible predictor variables (maximum bits-per-position, robustness, and frequency) in our final model construction.

Given the result that our modeling efforts should be restricted to a neighborhood of 95% of optimal, we constructed 20 models (five sample sizes, four scoring parameter combinations) for each of the four response thresholds using the three predictor variables without interaction. Upon examining the estimated model parameters, two observations were made: 1) the frequency and maximum

Table 3.2. Preliminary Analysis Robustness/Intercept Estimate Correlation

Response Threshold	R	t	D.F.	p-value
1	-0.997	-56.8	18	9.4e-22
2	-0.995	-44	18	8.84e-20
3	-0.993	-36.6	18	2.38e-18
4	-0.996	-47	18	2.78e-20

bits-per-position coefficient estimates were nearly equivalent across models, and 2) for each model the estimated intercept was the negative of the robustness value. An analysis of the correlation between the intercept and robustness estimates for each response threshold showed extremely high correlation (3.2). This led us to conclude that either robustness or the intercept could be omitted from the *preliminary* analysis. While we have a concrete interpretation of robustness, we do not have an equivalent interpretation of the intercept term as it relates to the set of near-optimal alignments. For this reason, we chose to omit the intercept term. Once the 80 models were rebuilt using the new formula, we saw that for all models, the parameter estimate p-values were less than 2^{-16} and that now the estimated parameters for robustness, along with frequency and maximum bits-per-position were nearly equivalent across models. Using the Kruskal-Wallis test we tested whether the parameter estimates were independent of the two remaining model factors: sample size and scoring parameter combination. A 5x4 ANOVA test for each response threshold and each model parameter (Tables 3.3, 3.4, 3.5, and 3.6) shows that in each case the maximum bits-per-position coefficient was dependent on the scoring parameter combination while being independent of sample size. Robustness and frequency are independent of both factors. This result supports the results seen in Figure 2.2, which told us that scoring parameter selection did not substantially alter the quality of the near-optimal alignment generated.

3.3.2. Final Model Construction and Analysis

As a result of this preliminary analysis, we concluded that we could choose one sample size and be confident that the model would not change substantially. Because maximum bits-per-position was dependent on the scoring parameter combination, we selected the set with the combination of the 3 individual scoring parameter combinations. For brevity, we present only the model based on

Table 3.3. Response Threshold 1 Model Parameter ANOVA

	D.F.	S.S.E.	M.S.E.	F	p-value
Robustness Sa2.0mple Size	4	0.06091	0.01523	0.1231	0.9714
Robustness Scoring Parameters	3	0.33994	0.11331	0.9161	0.4623
Residuals	12	1.48424	0.12369		
Frequency Sample Size	4	0.14596	0.03649	0.4473	0.7725
Frequency Scoring Parameters	3	0.30795	0.10265	1.2583	0.3324
Residuals	12	0.97891	0.08158		
Maximum Bits Sample Size	4	0.1620	0.0405	0.1929	0.9374358
Maximum Bits Scoring Parameters	3	9.7937	3.2646	15.5503	0.0001953
Residuals	12	2.5192	0.2099		

Table 3.4. Response Threshold 2 Model Parameter ANOVA

	D.F.	S.S.E.	M.S.E.	F	p-value
Robustness Sample Size	4	0.07963	0.01991	0.1162	0.9742
Robustness Scoring Parameters	3	0.66266	0.22089	1.2896	0.3227
Residuals	12	2.05534	0.17128		
Frequency Sample Size	4	0.14190	0.03547	0.2787	0.8861
Frequency Scoring Parameters	3	0.59148	0.19716	1.5490	0.2528
Residuals	12	1.52743	0.12729		
Maximum Bits Sample Size	4	0.5443	0.1361	0.4915	0.7423000
Maximum Bits Scoring Parameters	3	10.1426	3.3809	12.2110	0.0005876
Residuals	12	3.3225	0.2769		

Table 3.5. Response Threshold 3 Model Parameter ANOVA

	D.F.	S.S.E.	M.S.E.	F	p-value
Robustness Sample Size	4	0.3194	0.0799	0.2907	0.8784
Robustness Scoring Parameters	3	0.9764	0.3255	1.1850	0.3566
Residuals	12	3.2959	0.2747		
Frequency Sample Size	4	0.28262	0.07065	0.3467	0.8413
Frequency Scoring Parameters	3	0.95350	0.31783	1.5597	0.2503
Residuals	12	2.44541	0.20378		
Maximum Bits Sample Size	4	0.3878	0.0969	0.2444	0.907573
Maximum Bits Scoring Parameters	3	12.5553	4.1851	10.5513	0.001106
Residuals	12	4.7597	0.3966		

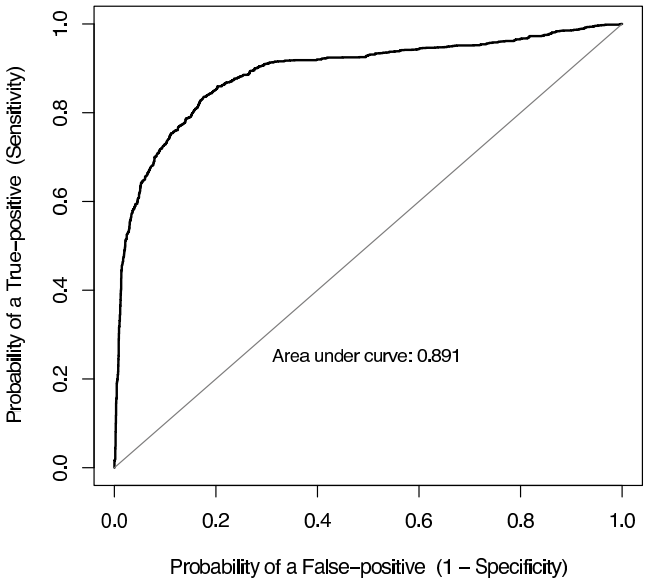
Table 3.6. Response Threshold 1 Model Parameter ANOVA

	D.F.	S.S.E.	M.S.E.	F	p-value
Robustness Sample Size	4	1.06422	0.26605	1.6891	0.2168
Robustness Scoring Parameters	3	0.61945	0.20648	1.3109	0.3162
Residuals	12	1.89015	0.15751		
Frequency Sample Size	4	0.66212	0.16553	0.9745	0.4570
Frequency Scoring Parameters	3	0.95178	0.31726	1.8677	0.1889
Residuals	12	2.03842	0.16987		
Maximum Bits Sample Size	4	1.2602	0.3150	1.7121	0.2118
Maximum Bits Scoring Parameters	3	19.7619	6.5873	35.7980	2.89e-06
Residuals	12	2.2082	0.1840		

Table 3.7. Logistic Regression Model Using Main Effects *Without* Interaction

	Coefficient	S.E.	Wald Z	P-value
Intercept	-9.806	1.9191	-5.11	0.0000
freq	4.720	0.1334	35.38	0.0000
robust	5.905	2.0162	2.93	0.0034
mbits	2.068	0.1849	11.19	0.0000

Figure 3.2. Model Main Effects *Without* Interaction ROC Curve



the 50% response threshold (two of four structural alignments) here. The final models for the other thresholds can be found in Appendix E. The final models all include the intercept term.

The first model we built used each of the three predictors (main effects) without interaction. Table 3.7 contains the result. The model goodness of fit test has a p-value of $3.39x10^{-10}$. The ROC curve (Figure 3.2) shows that model exhibits excellent classification according to [79]. Analysis of the residuals shows (Figure 3.3) a small amount of curvature in the Residuals vs Fitted and Scale-Location plots, which suggests that quadratic terms might be applicable. The Normal Q-Q plot in Figure 3.3 indicates that the residuals are not normally distributed, however this is not generally a large concern [79]. The Cook's Distance plot indicates 3 outliers, but from a sample of 5000, this is not a large concern.

The second model includes both main effects and all interaction terms. The model goodness of fit tests returns a p-value of 0.395, much higher than the first model. The results of the model can be seen in Table 3.8. ANOVA analysis of model deviance shows that the interaction terms do indeed

Figure 3.3. Model Main Effects *Without* Interaction Residual Analysis Plots

The Residuals vs Fitted and Scale Location plots are used to identify non-linearity in the residuals. In and ideal linear model, one would expect the residuals to form straight lines in both plots. The Normal Q-Q plot tests for normality of residuals and the Cook's Distance plot tests for outliers in the sample set.

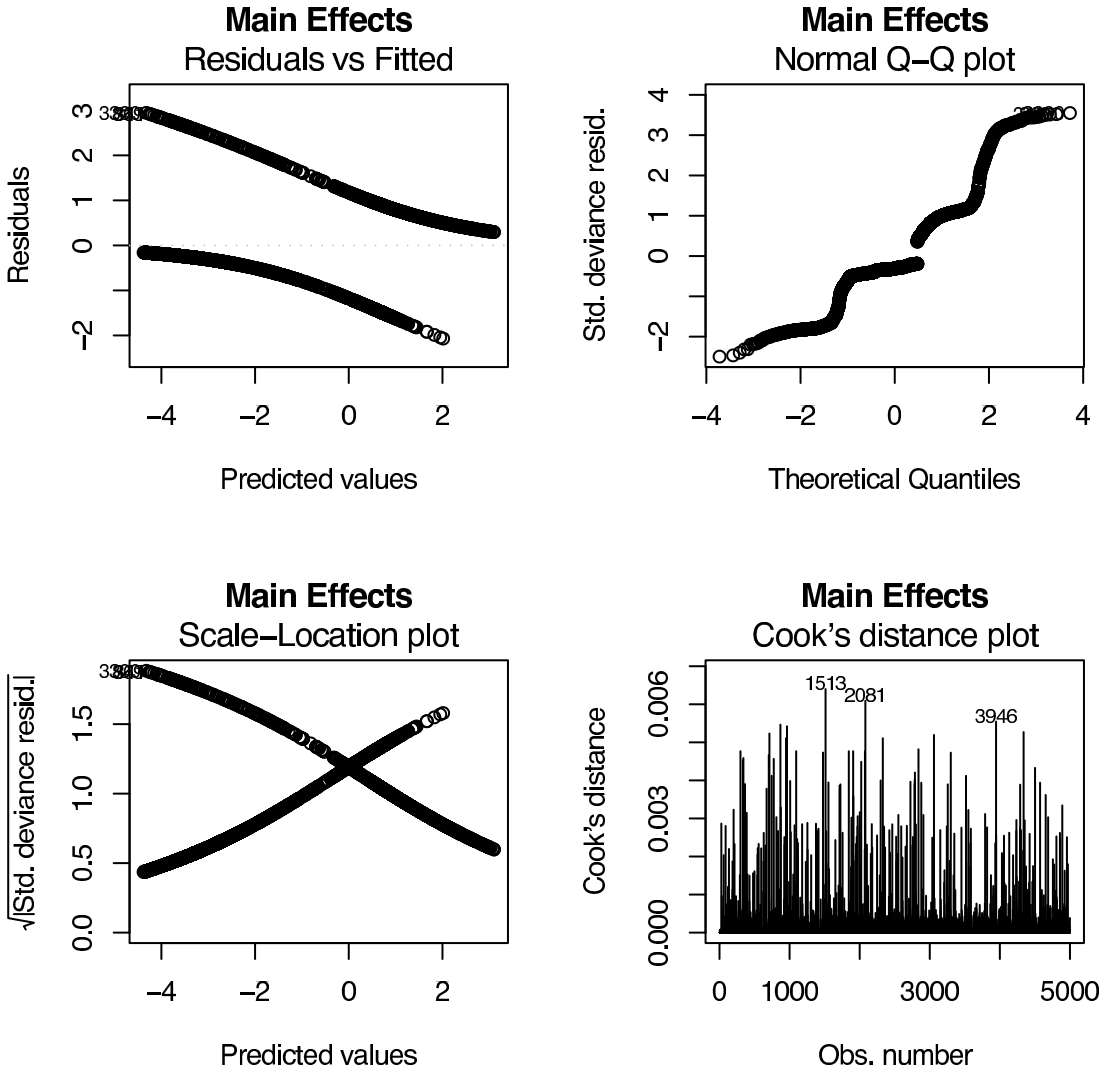


Table 3.8. Logistic Regression Model Using Main Effects *With* Interaction

	Coefficient	S.E.	Wald Z	P-value
Intercept	41.48	6.563	6.32	0.0000
freq	-49.35	7.788	-6.34	0.0000
robust	-46.07	6.857	-6.72	0.0000
mbits	-111.69	27.111	-4.12	0.0000
freq*robust	54.39	8.109	6.71	0.0000
freq*mbits	76.91	34.416	2.23	0.0254
robust*mbits	113.77	28.346	4.01	0.0001
freq*robust*mbits	-71.99	35.850	-2.01	0.0446

Table 3.9. Main Effects *Without* Interaction vs. Main Effects *With* Interaction ANOVA

The low p-value indicates that the models are significantly different with the Main Effect with Interaction model being superior because of the lower model deviance.

Model	Residual D.F.	Deviance D.F.	D.F.	Deviance	P-value (χ^2)
Main Effects *Without* Interaction	4996	3440.4			
Main Effects *With* Interaction	4992	3118.1	4	322.3	1.714e-68

improve the model (by decreasing deviance) by a significant amount (Table 3.9). The area under the ROC curve (Figure 3.4) also shows the improvement. However, inclusion of the interaction terms increases the apparent curvature in the residual plots (Figure 3.5). An additional worry is that the p-value of the model goodness of fit test is not significant.

The curvature apparent in the Residuals vs Fitted and Scale-Location plots (Figures 3.3 and 3.5) suggests possible polynomial behavior. Therefore, we first built a model that included second order polynomial terms, but excluded interaction. The complete model results can be seen in Table 3.10. The model goodness of fit test resulted in a p-value of 0. The ANOVA results comparing the Polynomial main effects with no interaction with the main effects with interaction model can be seen in Table 3.11. This analysis shows that polynomial main effects do not improve the model over the main effects with interaction. The polynomial predictor terms do not appear to improve and appear to worsen the apparent curvature in the Residuals vs Fitted or Scale-Location plots in Figure 3.6. The ROC curve in Figure 3.7 shows no improvement over the main effects with interaction model.

The final model we constructed included all polynomial main effects with interaction terms. The

Figure 3.4. Model Main Effects *With* Interaction ROC Curve

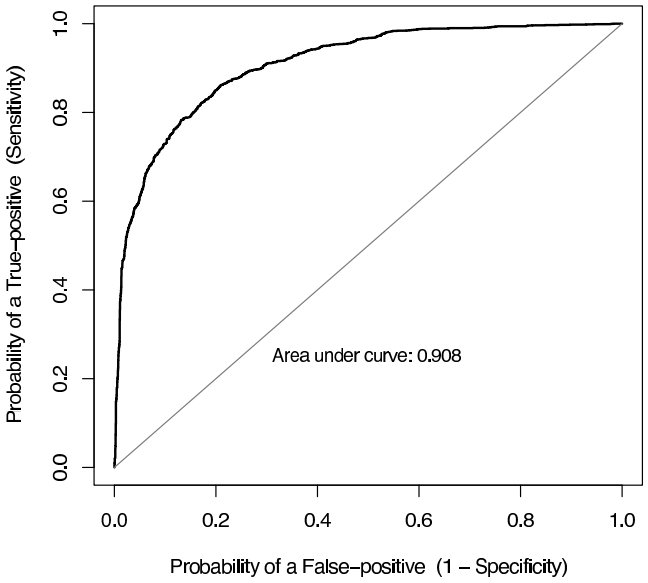


Table 3.10. Logistic Regression Model Using Polynomial Main Effects *Without* Interaction

F stands for frequency, R stands for robustness, and M stands for maximum bits-per-position. 1 stands for linear terms and 2 stands for quadratic terms.

	Coef	S.E.	Wald Z	P
Intercept	-1.503	0.05943	-25.30	0.0000
F1	153.176	4.59308	33.35	0.0000
F2	-9.012	2.89360	-3.11	0.0018
R1	14.224	3.07510	4.63	0.0000
R2	13.525	2.78759	4.85	0.0000
M1	45.554	3.92488	11.61	0.0000
M2	-22.094	3.48997	-6.33	0.0000

Table 3.11. Main Effects *With* Interaction vs. Polynomial Main effects *Without* Interaction ANOVA

Model	Residual D.F.	Deviance D.F.	D.F.	Deviance	P-value (χ^2)
Main Effects *With* Interaction	4992	3118.1			
Polynomial Main Effects *Without* Interaction	4993	3371.9	-1	-253.5	3.943e-57

Figure 3.5. Model Main Effects *With* Interaction Residual Analysis Plots

The Residuals vs Fitted and Scale Location plots are used to identify non-linearity in the residuals. In and ideal linear model, one would expect the residuals to form straight lines in both plots. The Normal Q-Q plot tests for normality of residuals and the Cook's Distance plot tests for outliers in the sample set.

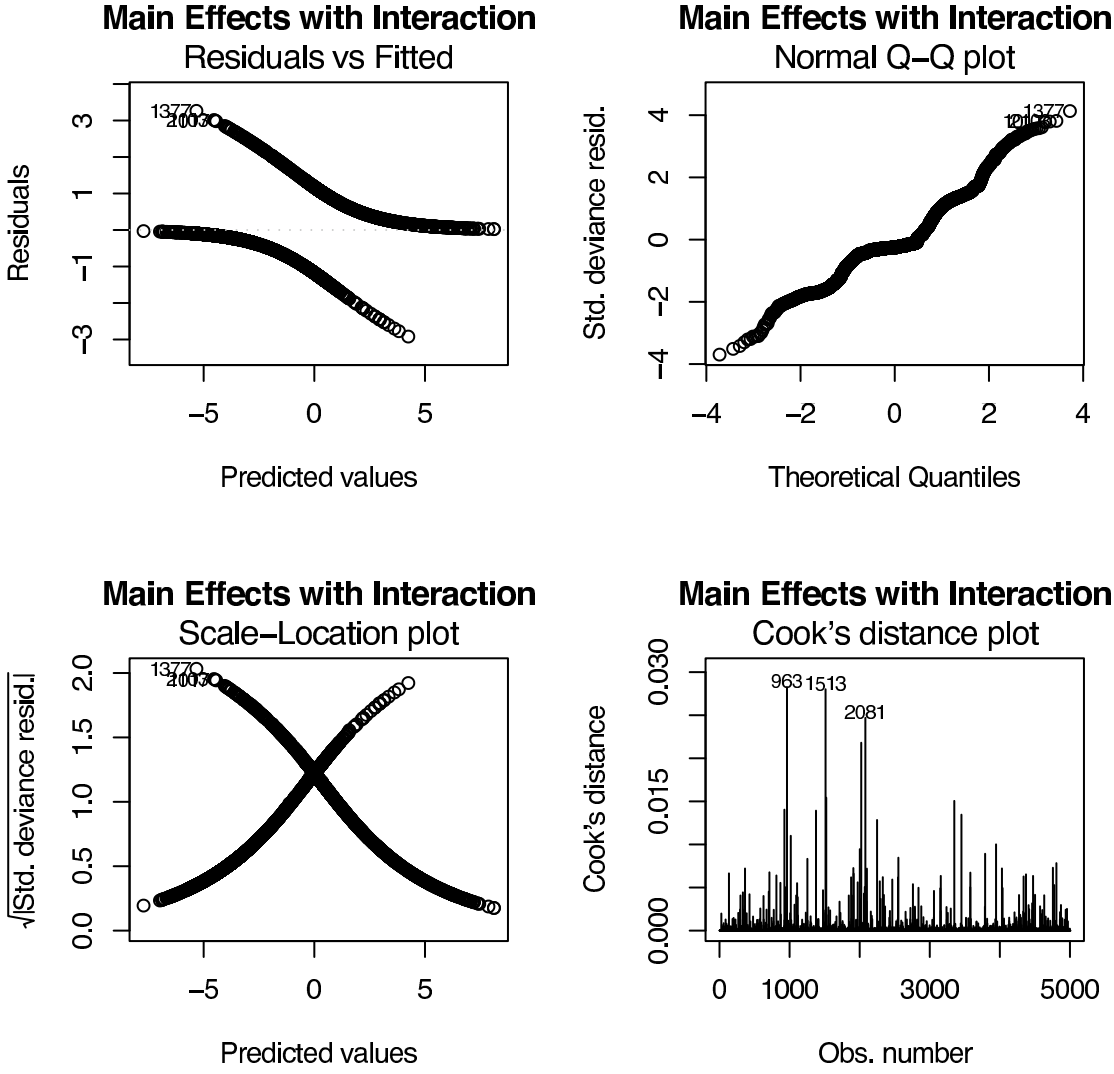


Figure 3.6. Polynomial Main Effects *Without* Interaction Residual Analysis Plots

The Residuals vs Fitted and Scale Location plots are used to identify non-linearity in the residuals. In and ideal linear model, one would expect the residuals to form straight lines in both plots. The Normal Q-Q plot tests for normality of residuals and the Cook's Distance plot tests for outliers in the sample set.

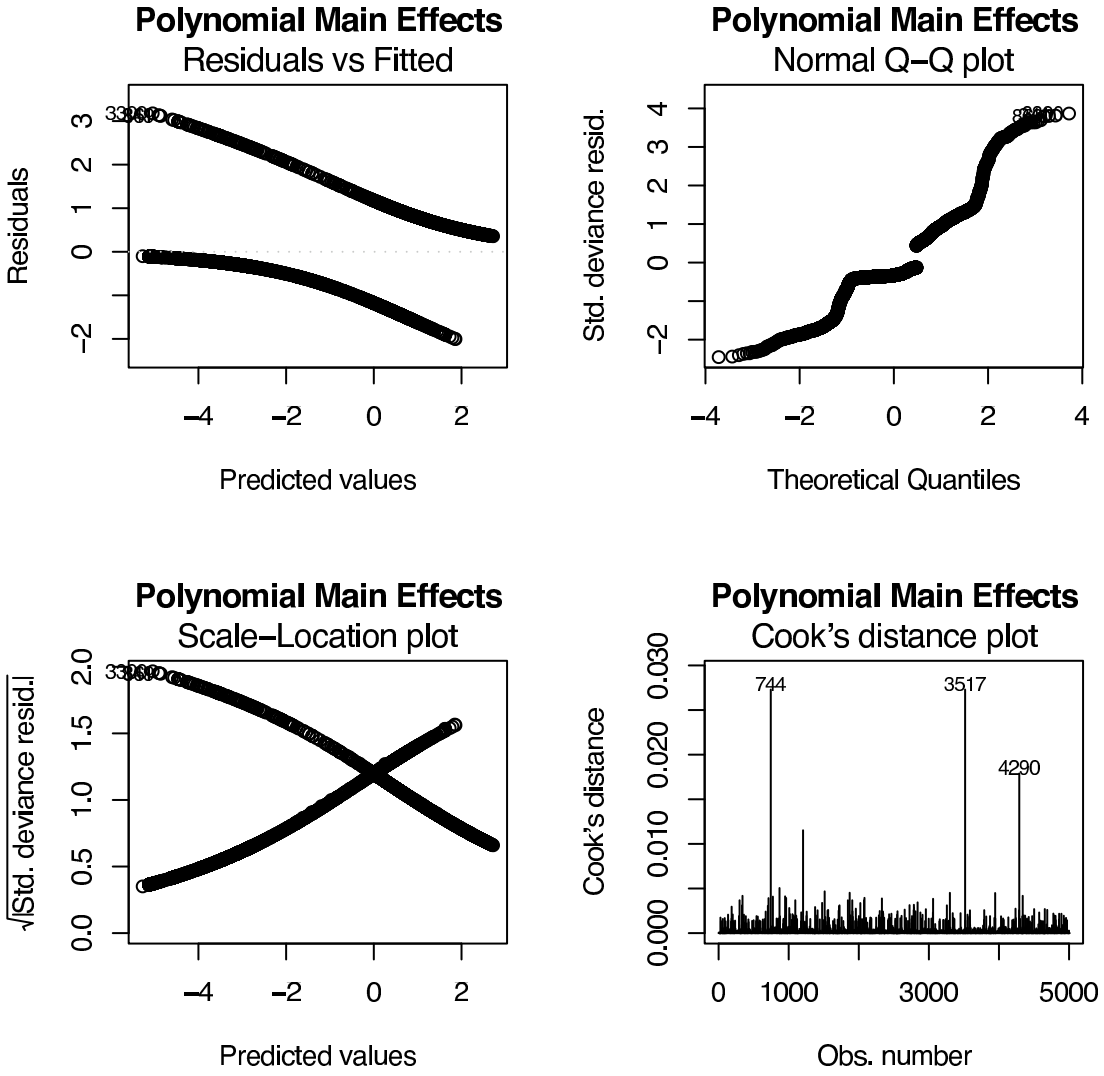
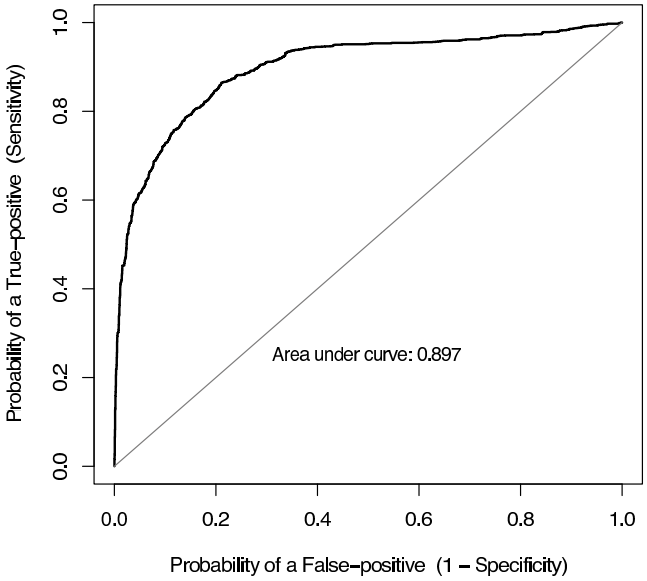


Figure 3.7. Polynomial Main Effects *Without* Interaction ROC Curve



results can be seen in Table 3.12. This model has model goodness of fit score of $9.88719 * 10^{-7}$. The ANOVA results comparing this model with the main effects with interaction model (our best model so far) are seen in Table 3.13. The analysis tells us that the polynomial main effects with interaction has produced the best model yet. However, the Residuals vs. Fitted and Scale-Location plots (Figure 3.8) show the worst curvature yet, while the Q-Q Normal plot indicates normally distributed residuals. The ROC curve (Figure 3.9) shows only a 1/1000th difference in the area under the curve between the main effects with interaction model (Figure 3.4).

The final model, which includes quadratic terms and all interactions and shows the best model deviance. By this measure, this is the best *model*. Additional evidence of the quality of the model is the increased apparent normality of the residuals. The difficulty is that the Residual vs. Fitted and Scale-Location plots show unusual curvature. Interestingly, the Residual vs. Fitted plot shows less actual curvature, than sharp bends. At the extreme ends of the X axis the lines appear straight, flat, and show that the model is essentially perfect (residual = 0, Figure 3.8). Towards the middle of the plot, the model shows residual error, but again the lines appear relatively straight. The implication is that the residuals are behaving according to two different linear functions. It is unclear why this occurs. Another worry is that the ROC curve shows no improvement over the model with interactions, but without quadratic terms. Figure 3.10 shows the ROC curves from each alternative model overlaid on one another. The final worry is that while the goodness of fit shows a significant

Table 3.12. Logistic Regression Model Using Polynomial Main Effects *With* Interaction

F stands for frequency, R stands for robustness, and M stands for maximum bits-per-position. 1 stands for linear terms and 2 stands for quadratic terms.

	Coef	S.E.	Wald Z	P
Intercept	-1.198e+00	1.214e-01	-9.86	0.0000
F1	1.468e+02	7.944e+00	18.48	0.0000
F2	1.382e+01	5.598e+00	2.47	0.0136
R1	4.622e+01	1.479e+01	3.12	0.0018
R2	3.117e+01	1.728e+01	1.80	0.0713
M1	2.937e+01	1.244e+01	2.36	0.0182
M2	-3.813e+00	1.046e+01	-0.36	0.7154
F1 * R1	5.705e+02	9.044e+02	0.63	0.5282
F2 * R1	1.951e+03	5.382e+02	3.62	0.0003
F1 * R2	-3.682e+02	1.108e+03	-0.33	0.7395
F2 * R2	1.417e+03	6.378e+02	2.22	0.0263
F1 * M1	2.908e+03	8.002e+02	3.63	0.0003
F2 * M1	7.887e+02	5.220e+02	1.51	0.1308
F1 * M2	-1.919e+03	6.566e+02	-2.92	0.0035
F2 * M2	4.938e+02	4.362e+02	1.13	0.2576
R1 * M1	7.819e+03	1.407e+03	5.56	0.0000
R2 * M1	4.536e+03	1.699e+03	2.67	0.0076
R1 * M2	3.675e+03	1.068e+03	3.44	0.0006
R2 * M2	2.267e+03	1.445e+03	1.57	0.1167
F1 * R1 * M1	-2.378e+05	8.572e+04	-2.77	0.0055
F2 * R1 * M1	2.085e+05	5.229e+04	3.99	0.0001
F1 * R2 * M1	-2.057e+05	1.082e+05	-1.90	0.0573
F2 * R2 * M1	1.260e+05	6.127e+04	2.06	0.0397
F1 * R1 * M2	-1.279e+05	6.493e+04	-1.97	0.0488
F2 * R1 * M2	1.356e+05	4.149e+04	3.27	0.0011
F1 * R2 * M2	-1.554e+05	9.197e+04	-1.69	0.0911
F2 * R2 * M2	9.515e+04	5.030e+04	1.89	0.0585

Table 3.13. Main Effects *With* Interaction vs. Polynomial Main Effects *With* Interaction ANOVA

Model	Residual D.F.	Deviance D.F.	D.F.	Deviance	P-value (χ^2)
Main Effects *With* Interaction	4992	3118.1			
Polynomial Main Effects *With* Interaction	4973	3024.84	20	93.29	8.635e-12

Figure 3.8. Polynomial Main Effects *With* Interaction Residual Analysis Plots

The Residuals vs Fitted and Scale Location plots are used to identify non-linearity in the residuals. In and ideal linear model, one would expect the residuals to form straight lines in both plots. The Normal Q-Q plot tests for normality of residuals and the Cook's Distance plot tests for outliers in the sample set.

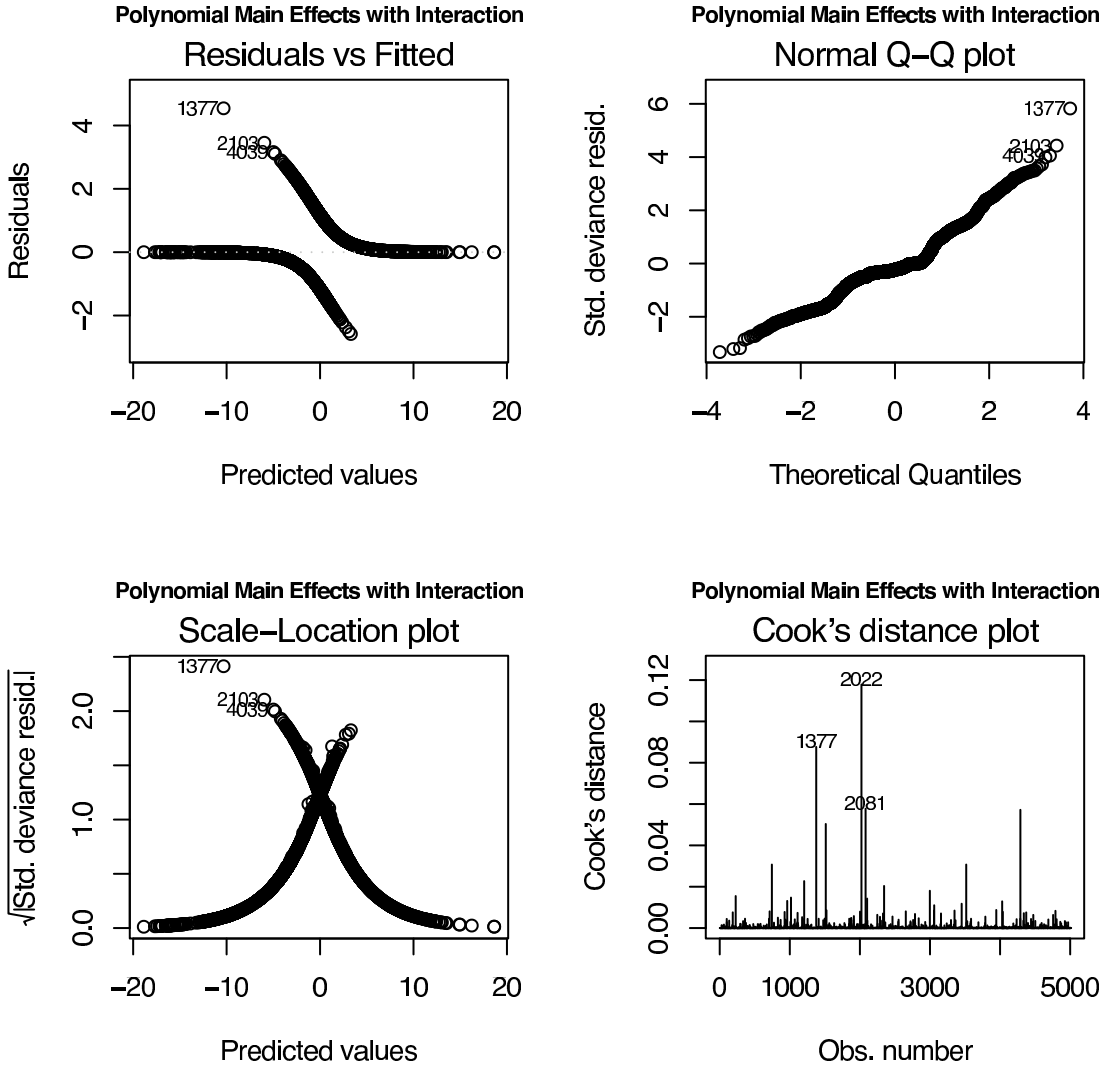
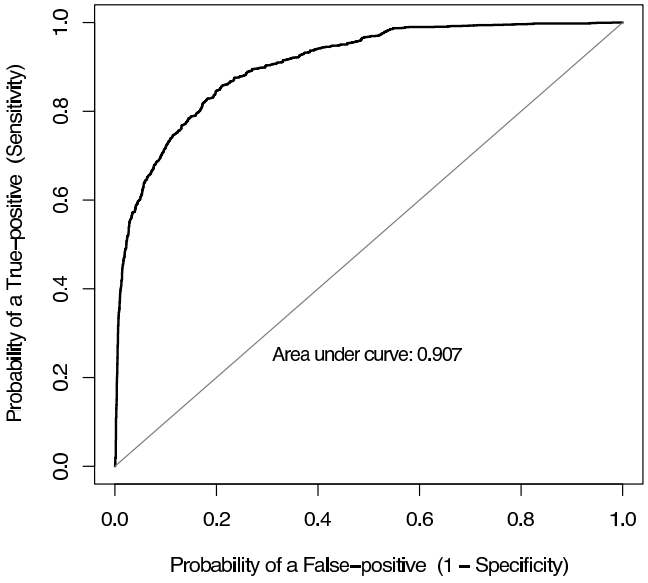


Figure 3.9. Polynomial Main Effects *With* Interaction ROC Curve



p-value, it is possible that the p-value is a result of the large sample rather than the quality of the model. Given the large model sample size, it is also reasonable to question the model deviance statistics.

The statistics tell us that the quadratic interaction model is the best, and perhaps as a *model*, it is. However, we are less interested in the model than in what the model accomplishes for us, meaning how well the model classifies edges. The ROC curves show that the ability of the models to classify data is essentially the same. Our sense of parsimony therefore dictates that we select the simplest model, the main effects without interaction. Using the standard described in [79], this model demonstrates excellent discrimination. The low p-value for the goodness of fit test, the significance of all variables in the model (Table 3.7), and the reasonable error bars apparent in the partial residual plots (Figure 3.11) provide further evidence that the selected model is appropriate and of high quality.

Finally, and most importantly, the model improves upon the use of robustness alone to predict structural significance as demonstrated in Figure 3.12. Figure 3.12 shows a Receiver Operating Characteristic (ROC) curve that summarizes the ability of robustness to directly estimate structural significance combined with the ROC curve for the main effects without interaction model. Prediction for robustness is accomplished by selecting a threshold value and assigning those edges with

Figure 3.10. ROC Curves of Each Alternative Model

Black represents the main effects model without interaction, blue represents the main effects model with interaction, red represents polynomial main effects without interaction, and green represents polynomial main effects without interaction.

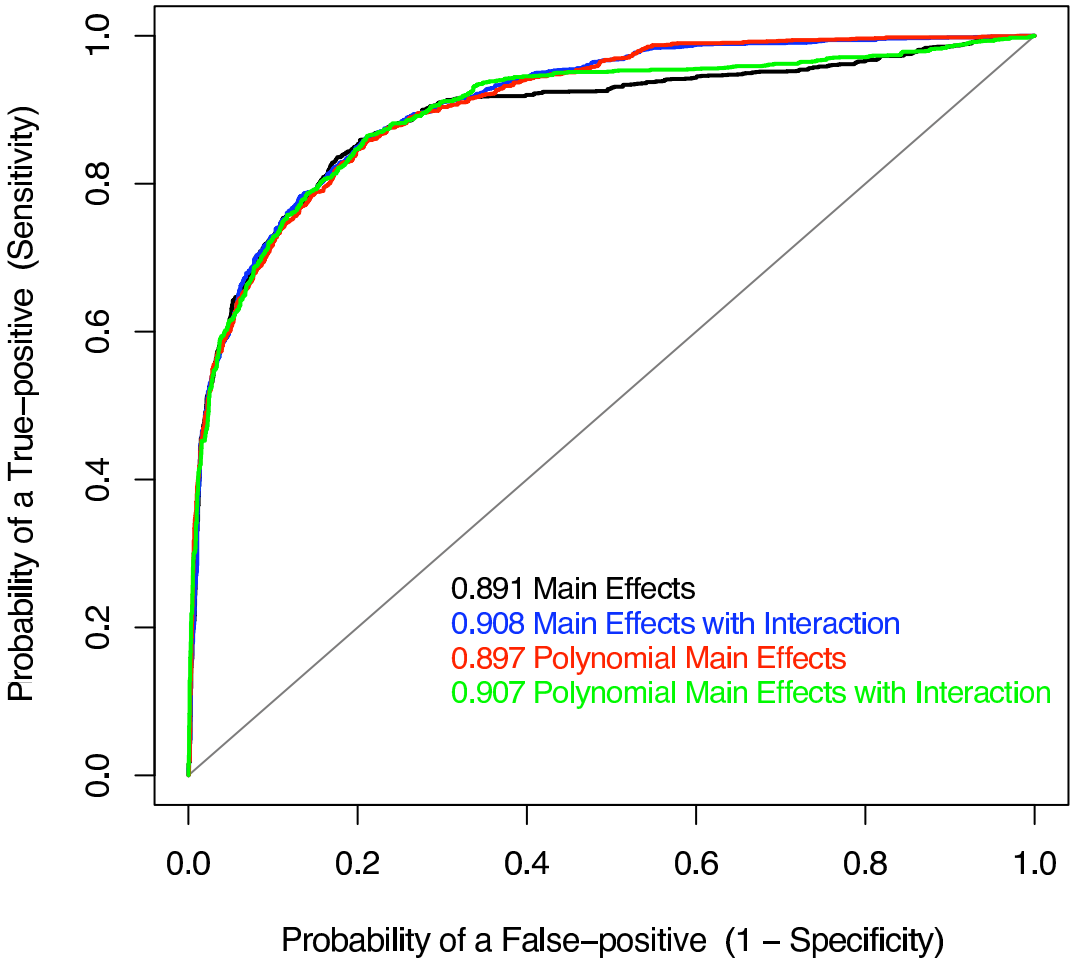


Figure 3.11. Partial Residual Plot

Partial residual plots for the three predictor variables in the main effects without interaction model (Table 3.7).

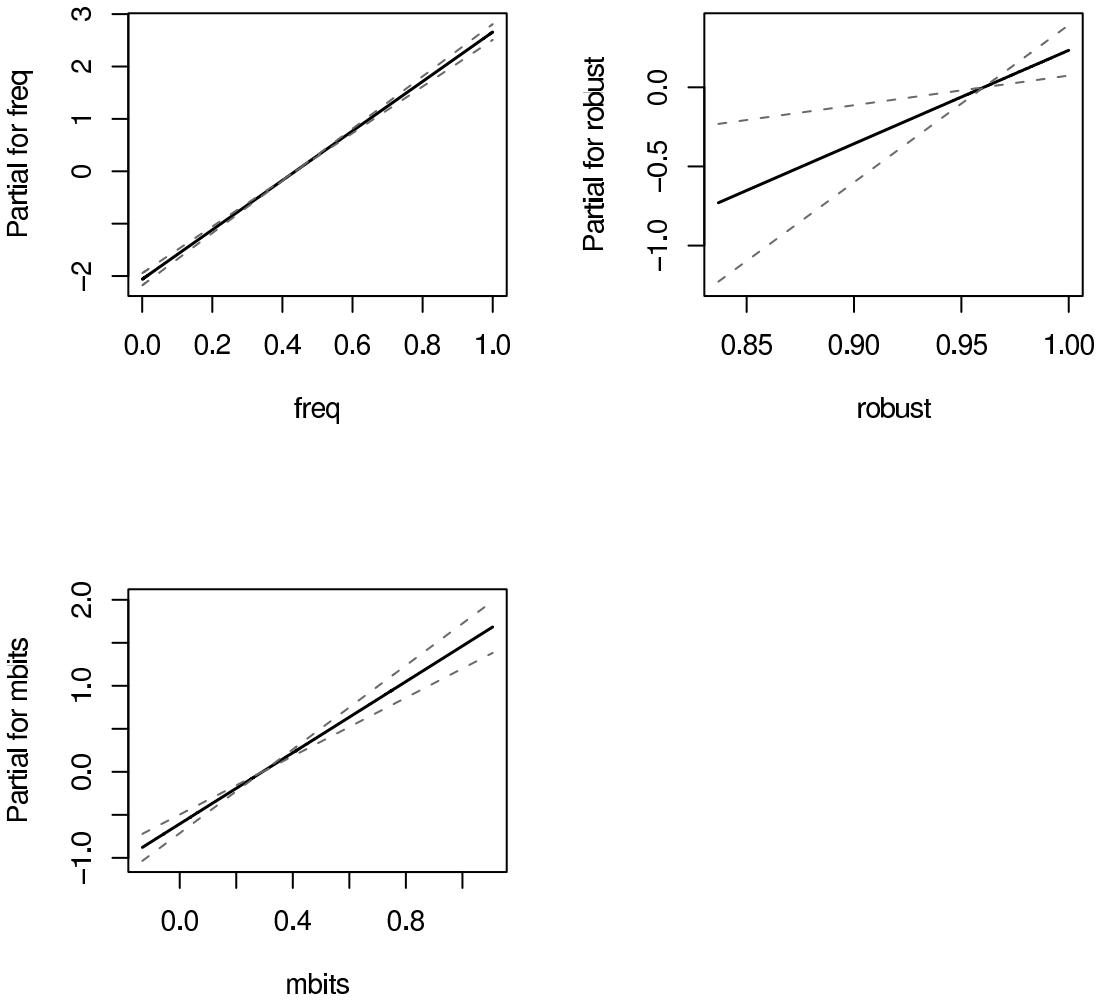
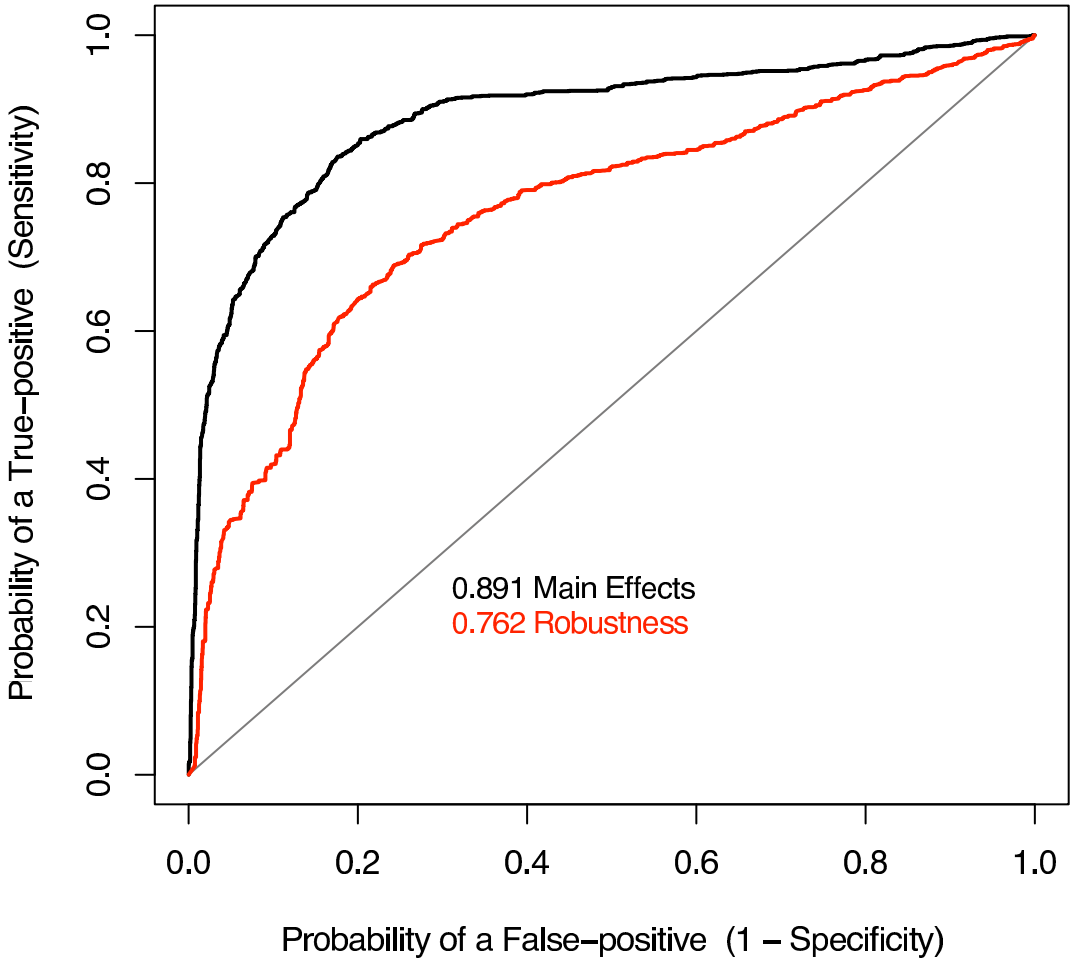


Figure 3.12. Main Effects *Without* Interaction vs. Robustness ROC Curves

Receiver Operating Characteristic (ROC) curve of robustness thresholds (red) and the logistic regression model using main effects without interaction (black). The larger the area is (closer to 1), the better the prediction is. The logistic regression model is described in Table 3.7. This curve represents an approximate 16% improvement over robustness alone. In both cases, the results are from a sample containing 5000 edges drawn from the training set with a combination of the three scoring parameter possibilities (BLOSUM50 -10/-2, BLOSUM50 -12/-2, BLOSUM62 -11/-1).



values above the threshold structural significance. The figure shows that robustness can accurately predict structural significance greater than 76% of the time while the main effects without interaction model predicts the structural significance of the edge greater than 89% of the time.

The information about structurally significant edges can be used to enhance construction of homology models for sequences with relatively low sequence identity, something that is currently very difficult.

3.4. Conclusions

This chapter details the construction of a probabilistic model that predicts which pairs of aligned amino acids in a set of near-optimal alignments are likely to be part of a structural alignment. Our model uses the frequency that edges occur in a set, the robustness of the edge, and the maximum bits-per-position for the edge to estimate this probability. The resulting model is very accurate and improves upon the use of robustness alone to directly estimate structural significance of edges by approximately 16% (89-76/76). The improved prediction will allow researchers to create better homology models by providing confidence that particular edges should or should not align.

The model has been shown to be robust to different variables affecting the model. We have shown that among the combinations examined, the selection of scoring matrix and alignment scoring parameters plays very little role in the quality of the information derived from the alignments. This is further confirmed by the results in Chapter 2 that show little variation in results between scoring parameter combination. The model has been shown to be independent of the sample size used to create the model and the threshold used to generate the response variable as well. Therefore, in addition to the demonstrated accuracy of the model, we can also have confidence that the model is robust to different modeling factors.

The results in Chapter 2 suggest that larger near-optimal neighborhoods are better because they are more likely to contain alignments that are as good or better than structural alignments. Clearly, larger solution spaces will have a better chance of finding a better alignment, but the logistic regression model building process demonstrates that this is not necessarily desirable. The modeling

results show that if the near-optimal neighborhood is too large, then the frequency of structural edges within the set of edges will be too low. Based on these results we conclude that a neighborhood of approximately 95% of optimal is sufficient for uncovering useful information about the proteins. From a human perspective, this is a useful result because smaller neighborhoods imply smaller numbers of alignments which are much more tractable from a cognitive perspective [34]. This is also useful from a computational perspective because large neighborhoods can result in hundreds of thousands of alignments (particularly for long sequences with low percent identity) whose generation can consume valuable time and computational resources.

The model developed has been integrated into our system (described in Chapter 4) that allows users to build and visualize sets of near-optimal alignments. The ability to explore alternative alignments afforded by the display combined with the predictive power of the logistic regression model provides a valuable tool for researchers to build high quality alignments of proteins with low percent identity.

Chapter 4

Generating and Visualizing Near-optimal Alignments

4.1. Introduction

Chapter 2 demonstrated that near-optimal alignments favorably compare to structural alignments. Chapter 3 showed how the information contained in sets of near-optimal alignments can be used to effectively predict structural alignment significance. This chapter integrates that research and newly developed techniques for visualizing and displaying near-optimal alignment information such that it can be used in biological applications.

4.1.1. System Goals

The primary goal of the system is to aid understanding of the relationship between two proteins by visualizing large sets of alternative alignments. Our system enhances the display paradigms of both the pairwise alignment and path graph in order to exploit the strengths of each and improve upon their weaknesses. To accomplish this goal, the system allows biologists to generate, display, and analyze large sets of alternative alignments. This strategy is predicated on the hypothesis that consistency and variation in sets of alignments can help predict reliably aligned sections of proteins. This hypothesis was confirmed in Chapter 3. To this end, the system communicates this information about a set of alignments while providing means for detailed, amino acid-by-amino acid comparison. Because of the importance of expert knowledge in the assessment of alignment quality, the system supports mechanisms that allow users to apply their expertise to the alignments

under consideration. These mechanisms include adding display features to identify any relevant annotations such as structural information, the ability to filter out alignments with known problems, and the ability to directly manipulate the alignments based on hypotheses under consideration.

A secondary goal for the system is to facilitate understanding of the alignment generation algorithms. To accomplish this, researchers have the ability to easily control and modify alignments and to understand how their alignments are scored by the alignment generation algorithms. In this way researchers can develop alignment of interest and see how their alignment corresponds with near-optimal neighborhood. In addition, researchers have the ability to easily substitute their own algorithms for those provided with the system to facilitate experimentation and algorithm development.

4.1.2. System Requirements

The system requirements are minimal. The only significant requirement is for the system to be web-based so that diverse users can use the system without the burden of installing specific software. An implication of the web-based system requirement is that the system runs on different operating systems to support users with different computer requirements.

4.1.3. System Design

The design of the system has three primary parts. First is algorithm input, responsible for collecting the necessary sequences, annotations, and algorithm parameters necessary to generate alignments. Second is the generation of the alignments themselves and third is the display and control of the alignments. The display and manipulation of sets of alignments is the primary contribution of this chapter. This part of the system consists of two alignment visualization techniques, an animated pairwise alignment and an enhanced path graph, along with features that allow users to exploit their expertise and customize the display to fit their specific needs. Support for expertise includes mechanisms to filter alignments, highlight annotations and different aspects of alignment sets, and the ability to edit alignments from within the software.

4.2. Algorithm Input

The sequences, alignment parameters, and annotations on the sequences are entered using a web form [85]. Sequences can be entered by cutting and pasting, or by specifying a NCBI[1] recognized accession or GI (GenInfo Identifier) number. The researcher then specifies the scoring matrix and gap penalty parameters. Multiple combinations of scoring matrix and gap penalties can be specified to generate a comprehensive set of alignments. The user also specifies the near-optimal neighborhood that the alignment generation algorithm should consider. The web form currently allows only global alignments to be created. Local alignments can be emulated by specifying subsequences to be aligned. The reason for this is that this research has only focused on global alignments, so we can not be sure that the conclusions made in Chapters 2 and 3 hold for local alignments.[2]

Sequence annotations are an important part of the display because they allow users to incorporate external knowledge into the display. Annotations are integrated into the display with highlighting mechanisms discussed in section 4.5.1 of this chapter. Annotations are imported into the system in a separate file which is encoded in General Feature Format (GFF2 [86]). Annotations may be imported directly from files already in GFF2 format from either the web form or when run as an application. When the display is run from the web, the system also fetches any available annotations from the NCBI databases.

4.3. Alignment Generation

Methods exist to generate all near-optimal alignments within a certain threshold of the optimal alignment [7]. However, the fundamental difficulty with generating all near-optimal alignments is the large number of alignments created. Even with relatively short sequences (~200 amino acids

[1] National Center for Biotechnology Information, part of the United States National Library of Medicine (http://www.ncbi.nlm.nih.gov). The various NCBI databases are primary repositories for much of the world's genomic and proteomic knowledge. The databases can all be queried via the internet providing an invaluable resource for researchers.

[2] Both the alignment generation and the display software actually *do* support local alignments, the features are merely hidden from the users. These features can be accessed by running the alignment generation and display applications directly rather than over the web.

in length) and within a modest neighborhood of the optimal score (1-2%), there can be millions of near-optimal alignments. Therefore, various algorithms have been developed to sample (i.e. reduce) the near-optimal alignment space [8][56]. The primary goal for these algorithms is to generate a diverse sample, meaning that the sample is drawn from all parts of the near-optimal solution space. In practice, this means that the alignments in the sample exhibit as much variation as the set of all near- optimal alignments. Even using sampling techniques, it is possible to end up with several thousand near-optimal alignments for a pair of sequences.

4.3.1. Zuker Algorithm

Alternative alignments for the display are generated using a C++ program that implements an algorithm described by Zuker [8]. The Zuker algorithm has three basic steps. First, a standard a dynamic programming technique aligns the two sequences, generating a forward alignment score matrix. We chose an implementation close to that described by Gotoh [22]. Unlike the most space efficient versions of this algorithm, the entire score matrix is maintained in memory. Once the forward score matrix is generated, the process is repeated on the reversed sequences, creating a reverse score matrix. Next, the forward and reverse score matrices are combined into what we call the Zuker matrix. The value for node i,j in the Zuker matrix is calculated by the score in the forward matrix at point i-1, j-1, plus the score in the reverse matrix at m-i+1, n-j+1 (where m and n are lengths of the respective sequences), plus the value of the scoring matrix for the residues at locations i and j in the respective sequences. The value at node i,j of the Zuker matrix is the optimal score of a global alignment that is constrained to align residue i in sequence one with residue j in sequence two. For a more detailed discussion of this matrix and the algorithm in general, see the original paper [8] and the RNA folding paper where the technique was originally developed [55].

During the creation of these two score matrices, a second pair of matrices is created that keeps track of the path used to generate the scores at each node in the respective score matrix. These path information matrices are used to reconstruct the actual alignments when generating a set of alignments. For a given point, i,j, the algorithm constructs the alignment by working forwards and backwards from that point through the information matrices. This process is repeated for every point

i,j where the value of that point in the Zuker matrix is greater than the near-optimal neighborhood threshold value specified in the input. The result is a sample of near-optimal alignments of the input sequences. As noted by Zuker, this algorithm only creates a sample of near-optimal alignments; at any given node there is often more than one direction that the alignment can follow that produces an alignment with the optimal or near-optimal score. A function randomly chooses one of the possible directions. The randomization ensures that more near-optimal gap edges will be included in the final set of alignments and reduces the occurrence of duplicate alignments. A property of our implementation of the Zuker algorithm is that the order that sequences are entered into the algorithm slightly alters the sample of alignments generated. This is a result of the algorithm to trace back through the score matrix to construct the alignment once the scores have been calculated.

The Zuker algorithm calculates an alignment for each match edge (where one residue aligns with another residue, as opposed to a residue aligning with a gap) that falls within the near-optimal neighborhood. For this reason, we can be certain that the sample created reflects the diversity in the entire set of near-optimal alignments for the specified neighborhood. This has been confirmed by overlaying path graphs generated with a Zuker sample and from all near-optimal alignments. The overlay showed that the only edges in the set of all near-optimal alignments that were not present in the Zuker sample were gap edges, something of relatively little biological interest.

4.3.2. Waterman-Byers Algorithm

While a sample of the near-optimal solution space is generally all that is necessary for most analyses, there are instances when it is desirable to have all near-optimal alignments. To this end, we have implemented the Waterman-Byers algorithm [7] for generating all near-optimal alignments. The algorithm uses the same forward dynamic programming technique as the Zuker algorithm to calculate a forward alignment score matrix. Unlike the Zuker algorithm, it is unnecessary to maintain a direction matrix. Once the forward alignment score matrix has been calculated and stored, the algorithm begins a simple stack based depth-first traversal of the path graph beginning at the m,n node of the alignment score matrix (m and n are the respective sequence lengths). As the tree is built, the reverse alignment score is calculated up to the given node. Next, the algorithm determines which of

the possible branches from that node will maintain a score within the near-optimal threshold. This is calculated by summing the score of the branch edge, the score of the tree to that point, and the value of the alignment score matrix at the end of the branch. As the tree is built, an alignment is also created, which explains why the direction matrix is unnecessary. Once the traversal reaches the 0,0 node, the alignment is output and the algorithm retreats to the last branch and repeats the process.

Even with today's fast computers and large amounts of disk space, it is relatively easy to overburden a computer when using the Waterman-Byers algorithm. The longer the sequences and the lower the percent identity they share, the larger the set will be. For example, using scoring parameters BLOSUM62, gap open of -6, gap extend of -1, and a neighborhood of 0% (only optimal alignments) the sequences 1cv2A0 and 1cqzB0 yield 402,978 alignments.

4.4. Display Sub-system

Our display consists of two primary modes for viewing alignments: 1) an animated pairwise alignment that displays a large number of alignments in sequence and 2) a path graph display built using zooming display techniques. The two display modes are supplemented by mechanisms for selectively highlighting different aspects of the displays, the ability to filter alignments from the defined set, and the ability to manually edit and create alignments to be considered along with the generated set. These features facilitate the application of domain expertise to the display.

4.4.1. Animated Pairwise Alignments

To help in understanding the relationships and how alignments vary within a set, we describe a method for visualizing large sets of pairwise alignments using animation, specifically the rapid serial visualization technique [87]. We also provide context regarding how one alignment compares to all others in the set and provide some overview information concerning relationships in the alignments. By animating the alignments, many alignments are presented while maintaining the pairwise

alignment paradigm that is so effective for detailed comparisons. The animation is generated by displaying each alignment for a short period of time, much like the frames in a movie. The resulting effect of the animation is analogous to astronomical blink comparators (blink microscopes) where alternating images of the same view are displayed and any objects in one image but not the other appear to flicker and become salient.

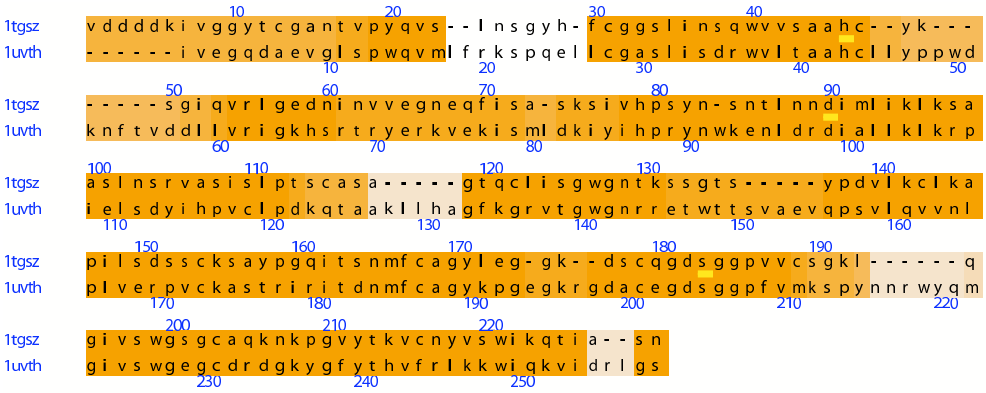
By itself however, the visual effect of displaying one alignment after the other does not help illuminate relationships because different numbers of gaps before reliably aligned sections cause these sections to shift in the display window. We therefore developed the steady display algorithm, which steadies the text on the screen when sections of several alignments are invariant. The visual effect can be described as "islands of stability" (Figure 4.1). Reliably aligned sections of alignments become salient because they appear steady on the screen. Conversely, variable sections become apparent because they appear to move on the screen. We further emphasize the steady display by highlighting the background of the amino acids where a darker background color indicates a more frequent (and thus reliable) alignment. The combination of the animation and highlighting provides overview information regarding reliably aligned sections within a set of alignments.

Coupled with the pairwise alignment display is a screen containing alignment information including the alignment score and the parameters used to create and score the alignment. This can be valuable for distinguishing alignments in the same set that are created using different parameters.

Traditionally, near-optimal alignments are presented together in a single display with a path graph or dot plot [8][9]. This representation effectively highlights sections of high variability between alignments, but these displays lose information by combining all alignments into one display. Figure 1.3 provides an example of a partial path graph of a subset of near-optimal alignments of two serine proteases 1TGSZ (NCBI GI: 230350) and 1UVTH (NCBI GI: 2781297). The alignment shows variability at the beginning and end of the alignment by drawing the multiple paths that the alignment could follow. In contrast, in the middle of the alignment there is only one path and hence only one way for the amino acids to align (this is expected as the ’H’ in the active site is aligned). The primary difficulty with a path graph display is determining which of the alignment paths are more likely. Additionally, path graphs are difficult to use for residue level analysis, because the

Figure 4.1. Conserved Highlight

A screen shot of a near-optimal alignment of 1TGSZ (NCBI GI: 230350) and 1UVTH (NCBI GI: 2781297). The alignment was created using BLOSUM50 -10/-2 and a near-optimal neighborhood of 95%. The steady display, conserved highlight, numbering, names, and customized active site highlights are selected. The three active site residues of the serine protease catalytic triad (46H, 90D, 183S in 1TGSZ; 43H, 99D, 205S in 1UVTH) are highlighted with yellow rectangles between the residues. The orange highlighting indicates the most conserved regions of the alignment. The darker the orange, the more conserved the region. The alignment is substantially conserved around all three active sites.



actual sequences are placed perpendicular and hence distant from one another. Readers of path graphs may have difficulty mentally mapping horizontal, vertical and diagonal edges into insertions and deletions (e.g., with which amino acids in 1UVTH does the 25th amino acid in 1TGSZ align?). Options for annotating a path graph with additional information are also limited.

Traditional text based alignments, with one sequence placed above the other (as seen in BLAST and FASTA output), are ideal for displaying the precise residue to residue mapping between the two sequences. But it is difficult to show alternative text based alignments and highlight the differences between the alignments. One strategy puts alternate alignments above and below the optimal alignment in some regions. This becomes more difficult when the number of gaps differs among the alignments, as this changes the overall alignment length. As the number of alignments increases, the difficulty increases.

Although it is relatively straight-forward to display a large set of alignments as successive frames in a movie, the naive approach does not make it easy to distinguish constant from variable alignment regions. We seek to highlight the residues that consistently align with one another and distinguish them from those positions that are more variable. To do this, the sections of an alignment that

are most consistent should remain steady on the screen while more variable regions should move around. This is not possible with the conventional constant spaced character placement used by BLAST and FASTA.

To address this difficulty, we developed an algorithm for placing pairs of residues (the two residues that align an edge in a path graph) according to the frequency with which the pairs occur in the set of all alignments and the relative position of that pair within an alignment. The result is a display where residues that consistently align with one another remain stationary in the display, while those that align with many different residues appear to move about.

The first step is to determine the relative position of each aligned pair of residues in relation to the overall length of the alignment. To calculate this we divide the index (position in the alignment) of the pair by the length of the alignment. Next this relative position is averaged with all of the other relative positions for the given pair, to produce an average relative position for the pair. Pairs can occur in many different alignments within a set of solutions. The frequency that each pair occurs with respect to the number of alternative alignments is also calculated.

When we are rendering the text on the screen, we use the average relative position of given pairs to determine the placement of the particular pair. The average relative position is multiplied by the width of the display to get the exact position on the screen where the pair will be rendered. The width of the display is determined by the longest alignment.

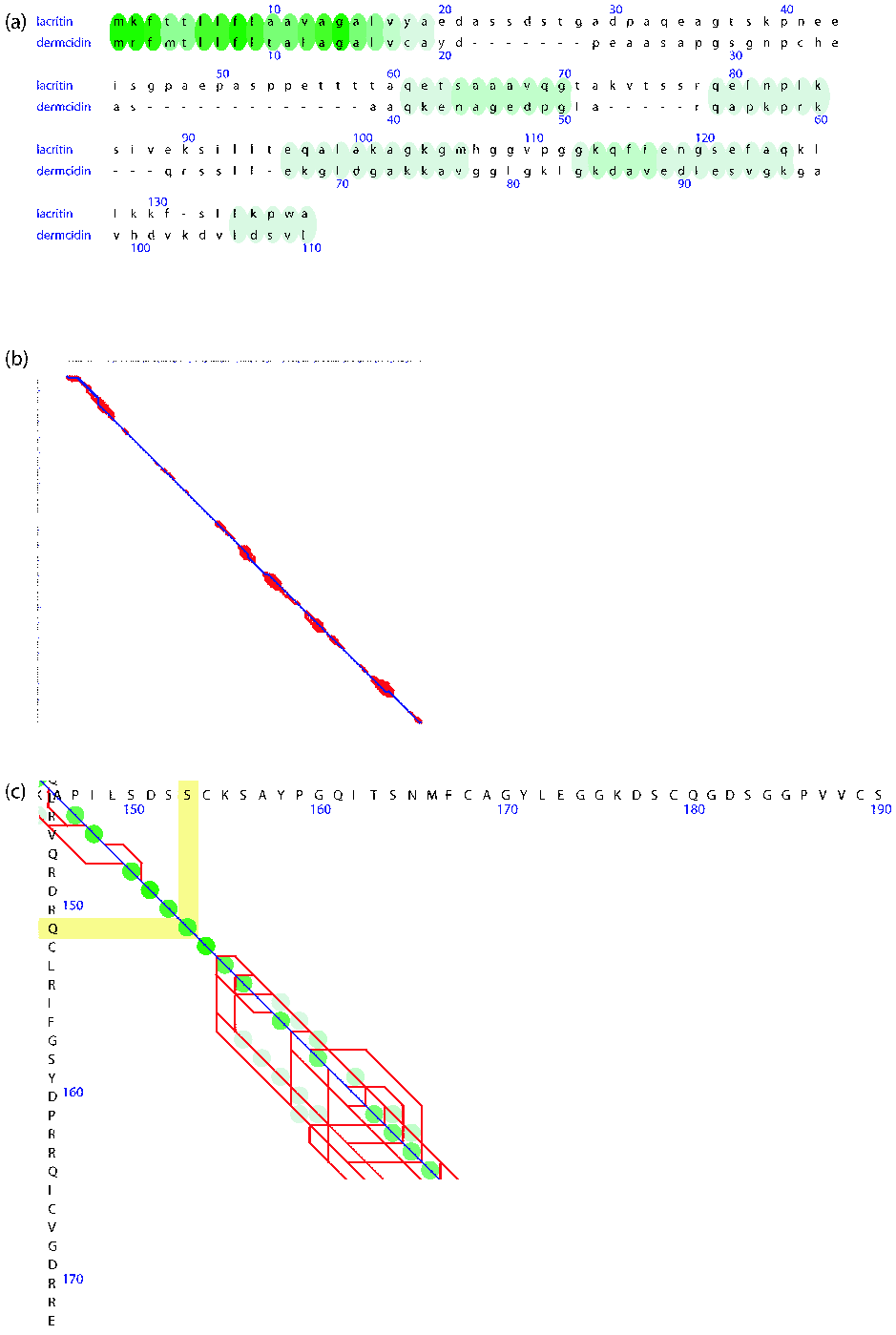
Each pair has only one average relative position and is therefore rendered in the same location on the screen every time. The visual effect of stability is an emergent property of the data. Pairs that appear in a large number of alignments appear stationary on the screen, since they are always rendered in the same location. Residues that are part of pairs that appear infrequently move around, since the different pairs have different average relative positions.

4.4.2. Zoomable Path Graph

Path graph representations of sets of alignments are generally lacking in the detail necessary for amino acid-by-amino acid comparisons of sequences. Because the path graph provides an excellent

Figure 4.2. Pairwise Alignment and Path Graphs

(a) An image of the pairwise alignment of Dermcidin (NCBI GI:16751921) and Lacritin (NCBI GI:15187164) in steady display mode and with robustness highlighting. (b) Zoomed-out view of an entire enhanced path graph of a set of near-optimal alignments of proteins 1AU8A (NCBI GI: 115725) and 1TGSZ (NCBI GI: 230350) . (c) Zoomed-in detailed view of an enhanced path graph of the same set of alignments of 1AU8A and 1TGSZ. Both sets of alignments created using BLOSUM50 -10/-2 with a neighborhood of 95% of optimal.



overview of the set of alignments, we developed an enhanced version of the path graph that supports detailed analysis. Our path graph was built using panning and zooming technology [88] that obviates many of the problems with static displays.

As mentioned previously, a significant problem with path graphs is one of scale. If the path graph is large enough that amino acid level comparisons are possible, then it is generally impossible to see the entire path graph in one window. Likewise, for any but the shortest sequences, if the entire length of the sequences is visible, then it is generally impossible to discern any detail. The implication is that users need a way to change the scale of the graph to see both detailed views as well as broad overviews. Zooming elegantly solves this problem by allowing users to seamlessly transition from broad overviews to more detailed perspectives on the path graphs.

In a zoomed-in view of a path graph, users must be able to move the section of the graph in view. The usual solution of scroll bars does not work in this case because the path graphs are not necessarily symmetrically diagonal in shape. Thus, any amount of scrolling across would be accompanied by some variable amount of scrolling down and vice versa. The need for complicated scroll bar interaction is eliminated by the zooming and panning navigation paradigm. Users can quickly navigate to desired locations by either panning the display to the region or zooming into the region.

A problem with the zooming and panning paradigm is the placement of the textual amino acid labels. In a static plot, the characters are generally placed along the top and left axes. When zoomed-in, however, the characters fall outside of the range of view. To accommodate this problem, our system uses floating sequence labels that stay centered over their respective edges and maintain the same scale as the path graph as users navigate around the path graph.

One difficulty in interpreting path graphs is due to the distance between the amino acid labels of the two sequences. This perpendicular layout makes it difficult to see which amino acids are aligned with which. The drag and pan paradigm solves the problem of physical distance between the labels. To see how the amino acids of a particular section align, it is simply a matter of dragging that section of the alignment to the top left corner of the display where the sequences intersect and are consequently relatively close to one another. An additional feature that further facilitates amino acid

level comparison is context sensitive mouse-over highlighting. The amino acids being aligned are highlighted when the mouse hovers over a particular edge in the graph. Gaps are represented using this highlighting scheme by shrinking the width of the highlight box such that it falls between the two amino acids where the gap is logically inserted.

Figures 4.2b and 4.2c are examples of a zoomable path graph generated by our system. The images are of the same path graph, but at different resolutions. Figure 4.2b shows a zoomed-out, overview of the enhanced path graph. No detail is discernible, however the red sections indicate unreliably aligned sections. Figure 4.2c shows a zoomed-in, detailed view of the enhanced path graph. The yellow highlight boxes show the amino acids that align for the edge at the intersection of the yellow boxes. The blue path indicates the alignment currently under consideration. The red paths indicate alternative alignments and the saturation of the green circles indicates robustness. The hue and saturation of the green circles are identical to those used in the animated pairwise alignment, which facilitates comparison between displays.

Another issue is that not all visible paths are near-optimal paths. Our system uses animation to show the valid paths. Just as the pairwise alignment cycles through the set of alignments displaying them in sequence, the path graph cycles through the alignment set. As the animation cycles, the current alignment is displayed in blue while the rest of the paths are displayed in red.

4.5. Support for Expertise

Human expertise is the only way to evaluate the quality of an alignment. To support the application of this expertise, our system includes filtering and highlighting mechanisms. Additionally, users can directly edit alignments to create their own. Coupled with the alignment generation algorithms, this creates a mixed-initiative interaction scenario.

4.5.1. Highlights

Highlights provide a mechanism for enhancing the display and presentation of alignments. By supplementing the information in the sequences alone we allow users to apply their expertise to achieve novel and unexpected results. All display annotations may be turned on and off by the user to suit individual preferences. There are two types of highlights: edge quality highlights and external highlights. Edge quality highlights are computed directly from the set of near-optimal alignments, are provided in all displays, and require no external data. External highlights depend on information not derived from or inherent to the set of near-optimal alignments. These highlights include sequence annotations automatically downloaded from external databases and highlights defined by the researcher.

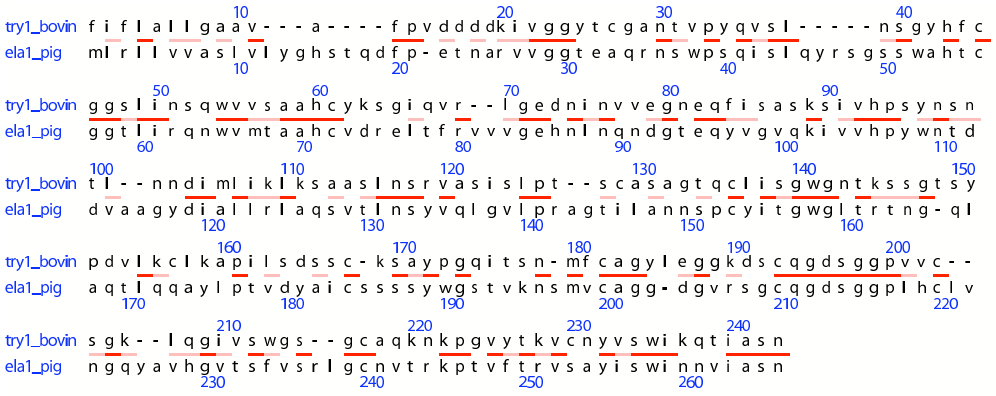
4.5.1.1. Edge Quality Highlights

Alone, the Steady Display algorithm provides powerful visual evidence of reliably aligned regions of a set of alignments. However, we can also use the recorded pair-frequency information to color the background of each pair according to the frequency of the pair (Figure 4.1). The most frequent pairs are colored a saturated orange, with the color gradually decreasing in saturation in proportion to the frequency of the pairings. Thus, in Figure 4.1, the regions around each of the three residues in the serine protease catalytic triad of 1TGSZ and 1UVTH are the most saturated. The least frequent pairings have a white background. In the specific alignment shown, the lightest colors correspond to regions that are not present in 1TGSZ. This coloring provides further visual indication of the consistency of certain sections of alignments.

Another highlight provides an alternative estimate of reliably aligned sequences. Robustness, discussed earlier, is a measure of how important a given pair of amino acids is to the overall score of that alignment [58] (Figure 4.2a). Pairs of amino acids with high robustness values have been shown to correlate with pairs of amino acids known to align using 3-dimensional structural alignments [58]. Similar to how we displayed the frequency highlight in [11], the more robust a pair of amino acids, the darker is the background shading. Together with the steady display effect, the background

Figure 4.3. Similarity Highlight

A TRY1_BOVIN (NCBI GI:2507249) and ELA1_PIG (NCBI GI:119253) alignment with the steady display, names, numbers, identity and similarity options selected. Constructed using BLOSUM50 -10/-2 with near-optimal neighborhood of 95%.



coloring for frequency and robustness provide powerful visual evidence of an invariant property of the set of alignments (i.e. those sections of the alignments that are consistently aligned). See Figure 4.2a for an example of the pairwise alignment display with robustness highlighting. Robustness is indicated by the green ovals where higher saturation indicates higher robustness. This figure also shows the variable spacing between amino acid characters that results from the steady display algorithm. Evidence of the unequal spacing of the characters is seen in the uneven right hand sides of the rows.

Beyond the background highlights that provide assessments of alignment quality, the display provides sequence identity information and orientation highlights. The display can highlight matching (red) and similar (pink) residues (Figure 4.3), which provides visual cues of sequence identity. For ease of orientation and navigation the display allows the sequence names to be displayed along with numbers that help identify locations within the sequences.

4.5.1.2. External Highlights

If the researcher specifies a protein by the GID/accession number then we also have access to the annotation information available in the NCBI databases. If any alpha helices or beta strands are found in this information, we provide an option for those annotations to be displayed. The icons for both the alpha helices and beta strands are designed to combine into a more meaningful icon

Figure 4.4. Secondary Structure Highlight

A TRY1_BOVIN (NCBI GI:2507249) and ELA1_PIG (NCBI GI:119253) alignment with the steady display, names, numbers, and secondary structure information highlighted. Constructed using BLOSUM50 -10/-2 with near-optimal neighborhood of 95%. Note how when only one sequence has a secondary structure annotation at a given location the icon appears gray and incomplete, but when both sequences have an annotation at the same position, the icon changes color and appears complete. The alpha helix icons that appear like arcs or upside down U's; when these two regions align, the helix symbols combine to form a loop symbolizing a helix. Likewise, the icons for beta strands come together to form an arrow.

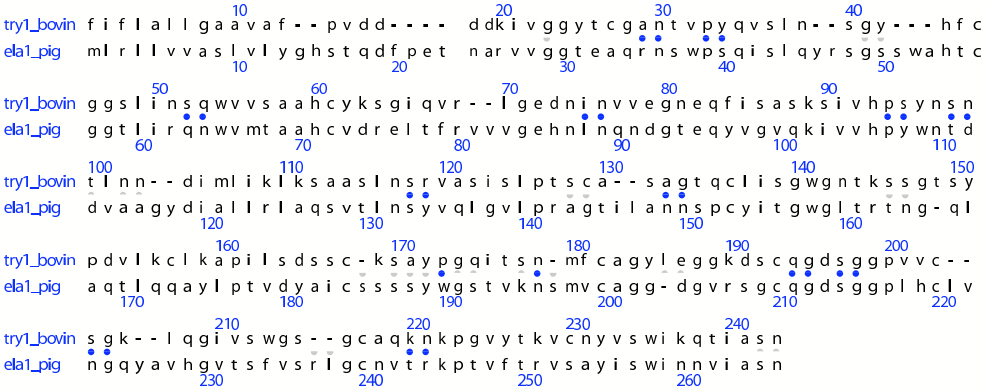
when both residues in a given pair share the same secondary structure. For alpha helices, an arch beneath the upper sequence combines with an X above the lower sequence to create a small loop that symbolizes a helix (Figure 4.4). Similarly, the icons for beta strands combine to produce an arrow. In addition to these automated annotations, users can specify their own annotations to be highlighted on the alignments. These highlights allow emergent visual annotations like small loops to appear when alpha-helix regions of proteins align.

While we believe the specified alpha-helix and beta strand icons make the most sense for those particular annotations, users are not constrained to those icons. We have developed a system that allows fine-grained control of display icons. Part of the management is the ability to map different display icons to specific annotations which allows users to create highly customized displays. Figure 4.5 is an example where hydrogen bonded turn annotations are highlighted using circle icons. The choice of circle icons was arbitrary and could have been any one of the other available icons (e.g. helix, strand, circle, triangle, or rectangle).

Apart from automatically downloaded annotations, users are also able to define and create their own annotations using two other mechanisms. First, users are able to directly edit and create annotations

Figure 4.5. Custom Highlight

A TRY1_BOVIN (NCBI GI:2507249) and ELA1_PIG (NCBI GI:119253) alignment with the steady display, names, numbers, and hydrogen bonded turns highlighted with circles. Constructed using BLOSUM50 -10/-2 with near-optimal neighborhood of 95%. The choice of circles was arbitrary and could have been a different icon, such as the rectangles used in Figure 4.1. The icon choice is made by the user.



within the software. This includes specifying specific locations to highlight and mapping icons to the newly defined annotations. The yellow rectangles highlighting the functional residues in Figure 4.1 were manually entered into the system. Annotations can also be imported in GFF format by the user from within the application. This is of use when users have pre-existing annotations that are not available in public databases.

4.5.2. Filtering

Sets of near-optimal alignments can be very large making them intellectually unwieldy. To solve this problem and to allow biologists to focus on specific features of alignments, we have provided means for filtering alignments. Filters are a mechanism for winnowing large numbers of alternative alignments. When a filter is applied, only alignments that pass the filter are displayed. Two filters have been implemented and a clearly defined software interface allows for the easy creation of new filters. The edge filter allows users to specify ranges of amino acids in each sequence that must align with one another. For instance, if particular functional amino acids are known to align, then a filter can be created that omits all alignments that do not align the specified regions. The second filter allows users to omit or include alignments within a particular range of scores. This filter allows

biologists to control their exploration of particularly large solution spaces and to explore hypotheses about differences between solutions at different thresholds of the optimal score.

4.5.3. Mixed-initiative Interaction

Mixed-initiative interaction involves dynamically substituting human judgment for different levels of computer automation [41][89]. Alignments are simply mathematical models optimized according to particular scoring assumptions and as a consequence do not always produce alignments that are biologically correct. As the near-optimal solution space is very large and is only sampled by most algorithms, there might be no alignment that improves upon the optimal solution. However, a near-optimal alignment usually exists that is close to improving upon the optimal. In these situations users can substitute their judgment for that of the computer, effectively acting as the alignment generation algorithm. Users are able to select any alignment and edit it by adding and removing gaps. The resulting alignment is then added to the set of alignments under consideration. The new alignment score is calculated according the same scoring parameters as the other alignments so that users can evaluate the quality of their reasoning by comparison with the mathematical model. By creating this feedback loop, users will be able to develop a better understanding of the limitations of the algorithms or flaws in their own reasoning by comparing their output with that of the alignment algorithms.

4.6. System Implementation

The system consists of three separate programs: 1) a Perl CGI script that collects the sequences, annotations and alignment parameters using the BioPerl [90] libraries and then manages the creation of alignments, 2) C++ code that generates the set of near-optimal alignments defined by 1), and 3) the Java code that displays the alignments. The display code is written so that it may be run either as an applet within a web browser, as a Webstart application, or as a stand alone application. This modular design facilitates different modes of interaction with the system and provides the freedom

necessary to use the display in novel ways. The software described has been available on the internet at [85]. Screenshots of the display can be seen in Figures 4.6a and b.

4.6.1. Alignment Transmission

Once the set of optimal and near-optimal alignments is generated, the alignments are formatted for transmission to the display software. The actual alignments are encoded using the FASTA -m9c encoding. For protein and DNA alignments, matches, insertions, and deletions are encoded by '=', '+', and '-' followed by the length of the match, insertion or deletion. Thus, the alignment:

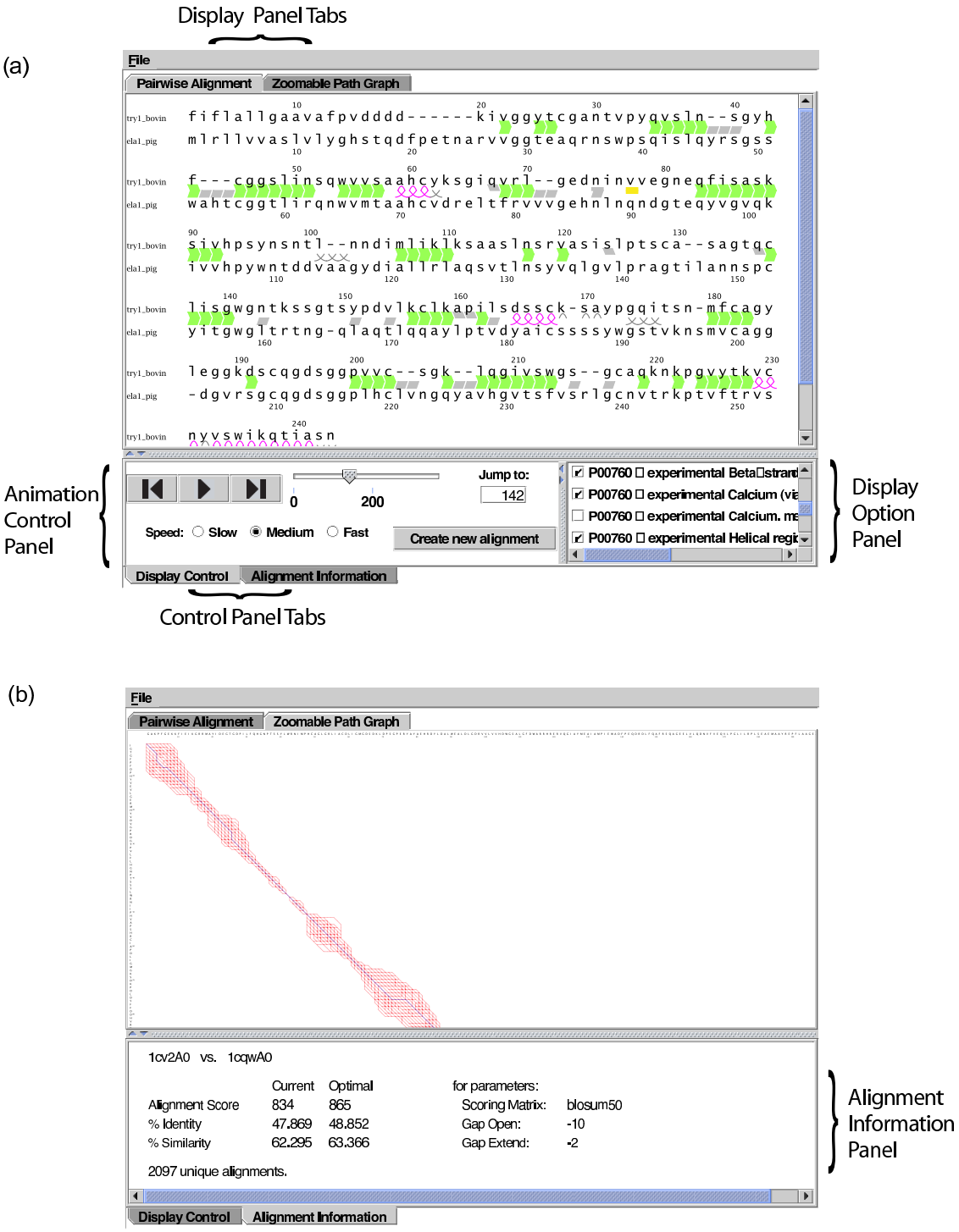
```
PYL-IDGSSHITQS
:..:::__..::.
PLVEIDG--MLTQT
```

would be encoded as: "=3-1=3+2=5". The parameter information, sequence data, and alignment information file is human readable and editable text.

4.6.2. Export

When run as an application (rather than an applet), users have the necessary permissions to write files to local hard drives. We therefore allow users to save various data in the system such as sets of alignments, single alignments in text format, and sets of annotations. We have also implemented a mechanism that allows researchers to export images generated by the system in the Scalable Vector Graphics (SVG) [91] format. This feature allows researchers to produce publication quality images of alignments and path graphs that include all customized highlights and annotations.

Figure 4.6. Alignment Display Screenshots



4.7. Conclusion

The animated pairwise display is a novel visualization technique that effectively highlights invariant regions of sets of near-optimal alignments. As demonstrated in Chapter 3, this information provides insight into possible structural significance. Combined with the different highlighting schemes, the animated pairwise alignment display provides a flexible interface for visualizing and exploring large sets of near-optimal alignments. The zoomable path graph provides a mechanism for visualizing entire sets of alignments in one screen. It solves the problems of scale inherent in static path graphs by using zooming technology. By providing mouse-over highlights we improve the usability of the path graph for detailed analysis. By providing filters we allow users to adjust and constrain the set of alignments being displayed without the computationally intensive task of recreating the set with new parameters. The ability to manually edit and create alignments combines human expertise with algorithmic efficiency, thereby creating a mixed-initiative interaction environment. This feature can help users develop a deeper understanding of the algorithms behind sequence alignment. By combining the detail-oriented pairwise alignment with the overview-oriented path graph we have developed a powerful system for exploring protein alignments in particular and sequence alignments in general. The effectiveness of the software is evaluated in terms of two case studies related in Chapter 5.

Chapter 5

Case Studies

To evaluate our system, we discuss two case studies where the system described in Chapter 4 was successfully used in scientific endeavors.

5.1. Dermcidin vs. Lacritin Homology Confirmation

This first case involves the alignment of the proteins Lacritin (NCBI GI:15187164) with Dermcidin (NCBI GI:16751921). Lacritin is a secretion enhancing factor that increases exocrine secretion in the lacrimal gland (i.e. tear ducts) [92]. Dermcidin is a protein that is hypothesized to play a role in breast cancer tumorigenesis [93]. Both proteins have been well studied in the laboratory, and as a consequence, information about locations of various functional regions in the proteins was available prior to this analysis. What was not known, however, was whether the two proteins are homologous (i.e. share a common ancestor).

The homology relationship between the two proteins was first hypothesized by a group at the Dana Farber Cancer Institute based on functional characteristics of the proteins [G. Laurie, personal communication, Nov. 2004]. This hypothesis could only be partially validated using other techniques. This is not an unreasonable result because the sequences are less than 30% identical (meaning less than 30% of the amino acids in the sequences match exactly). This range of sequence identity is the so-called "twilight zone" [94] of homology detection because sequences often do not have enough amino acids in common to develop the statistics used to establish homology. In this case, using

the BLASTP [2] and PRSS [1] programs, the researchers were able to establish weak statistical significance supporting the hypothesis that the proteins are homologous. Statistical significance is expressed in terms of the expected number of times an alignment score as good or better than that of Lacritin aligned with Dermcidin would be found in a given database of sequences. In this case, using the default BLASTP parameters (BLOSUM62 scoring matrix, gap open of -11, and gap extend of -1), the expected value for a database of 10,000 sequences was 0.01375, which is just less than the significance threshold of 0.02. While this result is statistically significant, it is a borderline case. This result could not be further supported using a structural alignment because neither protein structure had been solved. Therefore, the researchers used our near-optimal alignment system to analyze the proteins and increase their confidence in the homology of the proteins.

The researchers began their analysis by entering the sequence information about both sequences into the system website and specifying the alignment parameters. Alignment parameters were chosen based on empirical results [95] and personal experience (the parameters used were the BLOSUM50 scoring matrix, gap create = -10, gap extend = -2, near-optimal neighborhood of 95% of optimal). The generation algorithm yielded a set of 55 near-optimal alignments. The researchers then viewed the set of alignments and performed their analysis using the animated pairwise alignment display.

When viewed in the animated pairwise alignment display with either the robustness or frequency highlight selected, the most salient section of the alignment is the first 20 amino acids of each protein. This subsequence appears bright green in the display (Figure 4.2a). In addition to being brightly colored, this section of the alignment also appears most steady in the animation (i.e. that as alternative alignments are displayed, this section does not appear to change). Prior knowledge of both proteins told the researchers that this region represents the signal peptide. It was expected that the signal peptide regions would align because they are common to many different secretory proteins and are expected to be very similar because they serve the same function. The salience of this region demonstrates that our system helps identify highly conserved regions. However, because the signal peptide is used only for transport during protein synthesis, and otherwise plays no function in the behavior of the protein, it was of relatively little biological interest to these researchers. This is an example of how expert knowledge of this particular protein helped direct the researcher's attention to regions of more immediate interest.

Beyond the signal peptide are several other regions of interest. These regions were identified a priori in laboratory experiments and our system was used to confirm hypotheses about them [92]. With the robustness highlight selected, the second most salient region of interest is amino acids 60-70 in Lacritin and 40-50 in Dermcidin (Figure 4.2a). This section includes two hypothesized O-glycosylation sites in Lacritin. Similarly, a hypothesized N-glycosylation site surrounding amino acid 120 of Lacritin is also highly salient among the conserved regions of the alignment. These regions are of interest from a functional perspective and alignments of functional regions are evidence of homology. Our system allowed researchers to develop confidence in the alignments of these regions because of the characteristics indicated by the salience, namely consistent alignment of those regions across the set of alignments and the robustness highlighting. This was new evidence of homology for the researchers. The display was also useful to the researchers because the identification of interesting regions confirmed previous results through different means. The display also indicated three other salient regions. Based on current knowledge, it is unclear whether the additional regions are of functional interest. However, the highlights suggest areas to consider in future research.

One section of Dermcidin that clearly does not align with Lacritin is positions 20-40 of Dermcidin. This is indicated in two primary ways by our system. First, the amino acids appear to move on the screen as alternative alignments are presented. Second, the variability is made evident by the absence of any coloring from either the frequency or robustness highlights. These visual signals indicate that there is no way to consistently align those subsections of the two sequences. Additionally, the presence of a large number of gaps in the alignments of this section suggests that these regions do not match. This subsection of Dermcidin was shown by Porter et al. [93] to nearly match a mouse cachectic factor. A cachectic factor is an agent that causes a general loss of health, in this case related to breast cancer. The fact that this section does not appear to align with Lacritin tells us that the cachectic factor is most likely not present in Lacritin, which is good news for our tear ducts.

By correctly identifying different regions of the proteins that were expected to align, identifying regions not expected to align, and providing levels of confidence in those assessments, the system allowed the researchers to conclude that the proteins are homologous. It is important to note that

this conclusion is based on the judgment of the researchers. The role our display system played in this process was to provide the additional information needed to reach that conclusion.

The researchers did not use the zoomable path graph for their analysis. We speculate that this is because they were unfamiliar with the path graph paradigm, which is likely because heretofore path graph software has not been widely available. Some of the information used by the researchers in this case can also be seen in an un-enhanced path graph, but not all. Path frequency can be partially inferred by presence of large or small numbers of edges, but the un-enhanced path graph lacks the robustness highlight that was an important tool for the researchers.

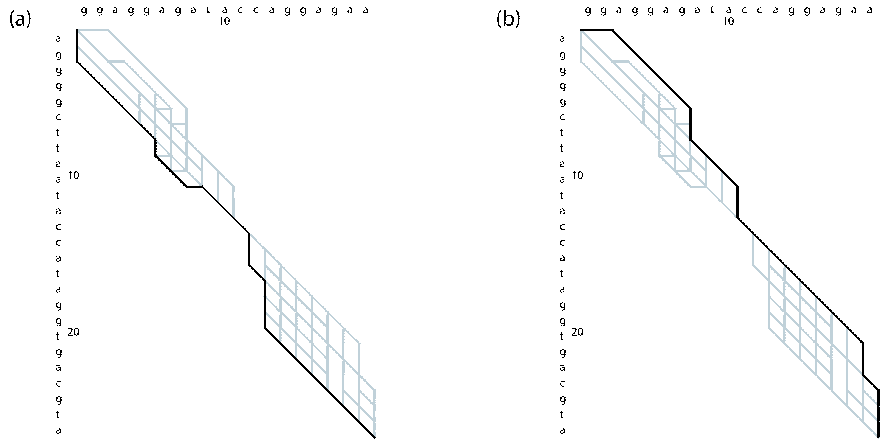
5.2. Near-optimal Alignment in Linear Space

This second case involves the use of our system to assist in the development of space efficient near-optimal alignment generation algorithms. Current generation algorithms require $O(mn)$ space (where m and n are the lengths of the sequences being aligned) [7][8]. Huang et al. [96] and Myers and Miller [51] demonstrate techniques for generating single alignments in $O(n)$ space. Our efforts involved adapting these techniques to generate sets of near-optimal alignments. Central to this effort were algorithms described by Huang et al. for finding the set of edges that comprise the lower left and upper rightmost boundary paths of a path graph, henceforth called Left and Right. Figure 5.1 shows images of small path graphs with the left and right boundary paths highlighted. Our strategy started with creating implementations of the Left and Right algorithms.

This case describes a part of the process in developing the space efficient alignment generation algorithm: using the path graph display to verify the correctness of the Left and Right algorithm implementations. Table 5.1 summarizes our final strategy and the system features used to accomplish this goal. Starting with candidate implementations of the Left and Right algorithms, we needed to validate that the algorithms correctly returned the left and rightmost paths. To do this, we used the Waterman-Byers algorithm [7] to calculate all near-optimal alignments within the specified threshold. We used this set to verify that the solutions generated by the Left and Right algorithms actually fell on the left and rightmost boundaries of the path graph. The Waterman-Byers algorithm was used

Figure 5.1. Leftmost and Rightmost Path Graphs

Views of the same path graph with the (a) bottom leftmost alignment highlighted and (b) top rightmost alignment highlighted. The sequences are random. They were selected to be short enough to display clearly in one panel (no zooming necessary) and different enough so that distinct leftmost and rightmost boundaries would be visible. The sequences and used solely for illustrative purposes.



to ensure that every possible edge was included in the set of alignments being displayed. It was then a matter of looking at the path graph and verifying that the Left algorithm returned the leftmost boundary of the Waterman-Byers set and the Right algorithm returned the rightmost boundary.

We chose relatively short, artificially created sequences for this task so that the Waterman-Byers set would not be too large for the large neighborhood of optimal chosen. The reason for the large neighborhood was so that there would be visually distinct left and rightmost boundaries. The alignment parameters used were the BLOSUM50 scoring matrix, gap open of -10, gap extend of -2, and a near-optimal neighborhood of 75%. The result was an optimal score of 60 and a near-optimal threshold of 45.

We then created two candidate alignments using our Left and Right algorithms. Once created, we combined them with the alignments generated by the Waterman-Byers algorithm so that they could be displayed together in our system. The end result was a set of 179 alignments (177 generated alignments and two candidate alignments). The path graph generated from this set of alignments can be seen in Figure 5.1.

Table 5.1. Case Two Overview

Overview of the steps performed and system features used in Case two.

Step	Task	System Feature Used
1	Generate candidate alignments and the Waterman-Byers set; combine the two.	The algorithm implementation we are testing and the alignment generation software.
2	View all alignments.	Path graph display.
3	Attempt to find candidate alignments.	Animation of path graph and filtering.
4	Manually create left and right alignments.	Alignment editor, path graph display, and pairwise display.
5	Find manually created alignments.	Animation of path graph and filtering.
6	Verify manually created alignments.	Path graph display.
7	Determine scores of manually created alignments.	Alignment information screen.

Our system is not capable of highlighting more than one alignment at a time in the path graph display. It is not possible to specify an alignment a priori for special treatment or guarantee the order in which the alignments are highlighted. These deliberate design decisions ensure that the user does not fixate on one "optimal" alignment and ignore the rest of the set. However, this meant that we first had to find the boundary path alignments within the set. The first step was to watch the path graph animation to find our candidate alignments. Upon first view, we did not notice the alignments in the animation. Instead of stepping through the alignments one-by-one, we decided to limit our search space by filtering out those alignments that did not match some criteria found in the left and rightmost edges. This meant making an edge filter that included only alignments that contained a few edges along either of the boundaries. Stated in terms of the pairwise alignment, this meant specifying pairs of amino acids that must align. With increasingly tight filters (including more and more edges along the boundaries), it became apparent that the left and rightmost paths were not in our set of alignments. The Waterman-Byers algorithm should have returned all alignments within the specified threshold and we were confident that the output from the Left and Right algorithms was included in the set. The two possible explanations for this inconsistency were that our implementation of Waterman-Byers was flawed and was not generating all, or that the Left and Right implementations were incorrect and returning erroneous alignments.

Determining which explanation was correct was relatively straightforward. Instead of relying on the

Left and Right algorithms to create the boundary alignments, we created them manually. We used the alignment-editing feature in the system to create the two alignments. We selected an alignment to edit that was close to what we wanted, meaning it included several of the boundary edges we were interested in, and then edited the alignment so that it fell completely on the boundary. This process was facilitated by both the pairwise display and the path graph display. The path graph display made the path perceptible while the pairwise display helped us see how the amino acids of the alignment actually aligned. The need for the pairwise alignment displays was further emphasized because the alignment-editing screen presents the alignment in a pairwise fashion. Once finished, we included each of the manually created alignments with the original set.

To verify that the new alignments were correct, we viewed them in the path graph with the rest of the set. To see the exact two alignments, we applied the filters previously created and quickly found the new alignments. It was clear that we had created the boundary alignments correctly, because when they were highlighted, the proper boundary edges of the path graph were highlighted.

Now that we had our target alignments, we had to verify that either the new alignments were improperly excluded from the set by the Waterman-Byers algorithm or that our Left and Right implementations were incorrect. This was accomplished by viewing the alignment information screen for each of the manually created alignments. This screen told us that the leftmost alignment had a score of -10 and the rightmost had a score of -2, both well below the near-optimal threshold of 45. This meant that the alignments were properly excluded from the set of all alignments by the Waterman-Byers algorithm. It also meant that our Left and Right implementations were incorrect.

This information was useful beyond evaluating the candidate Left and Right implementations. It also led to the conclusion that arbitrary paths in the path graph created by a set of near-optimal alignments were not necessarily near-optimal themselves. This effort also provided more evidence that our implementation of Waterman-Byers was done correctly.

The display system provided a number of tools that facilitated this analysis. First, it was easy to visualize the set of all near-optimal alignments in the path graph. We could also clearly see what the target alignments for the Left and Right algorithms should be. The notion of left and right is entirely

absent from the pairwise display paradigm. Had the pairwise display been the only mechanism for visualizing alignments, we would have had to compute the left and rightmost paths. Computing these paths is not necessarily difficult, but it would be more error prone and far more work that simply seeing the edge on the path graph.

Given the relatively large number of alignments to manage, the filters provided a simple mechanism for quickly narrowing our search set. The successive application of tighter and tighter filters provided the first evidence that the Left and Right output might not have been correct.

Once we were aware that a problem existed with our candidate alignments, the ability to edit and create new alignments from within the application and to subsequently add the alignments to the set of alignments for display made the validation process much quicker. Central to our ability to determine whether the manually created alignments were in the Waterman-Byers set was the system's ability to calculate the alignment scores of the manually created alignments. This demonstrates the value of a mixed-initiative paradigm where both human generated and computer generated alignments can be created and compared directly.

After iterating through this process several times we were able to generate correct implementations of the Left and Right algorithms and continue in our efforts to create a $O(n)$ space near-optimal alignment generation algorithm.

Chapter 6

Conclusion

The research described in this dissertation contributes to the body of knowledge in both the Systems Engineering and Bioinformatics disciplines as they relate to information visualization and sequence alignment. Sequence based alignment remains an important tool for modern biologists because of the inexpensive and readily available protein sequence information and the relative lack of structural information. Near-optimal alignments provide an opportunity to exploit this sequence information in novel ways. Visualization techniques are an important tool for bioinformatics researchers for managing the ever increasing amount of available information. This research enhances our understanding of the relationship between sequence based near-optimal alignments and structural alignments and provides guidance that aids in the exploration and understanding of near-optimal alignments. The research manifests itself in a software system that uses novel visualization techniques to support the generation, display, and exploration of near-optimal solution space. Chapters 3 through 5 describe the analysis, model building, and software that constitute this research.

Chapter 2 describes the comparison of sets of near-optimal alignments with alternative structural alignments. This research demonstrates that sets of near-optimal alignments compare favorably to structural alignments. Prior to this research we did not understand how closely near-optimal alignments could approximate structural alignments. Nor did we understand how well structural alignments generated using different algorithms compared to one another. We now understand that near-optimal alignments can meet and exceed the quality of structural alignments. While this occurs more frequently as percent identity increases, we also show that it occurs with low percent identity alignments. We also understand the extent to which the near-optimal alignment space intersects

with the structural alignment space. We have disproved the hypothesis that the variation among structural alignments is less than the variation between structural and near-optimal alignments. Together, these results demonstrate that the near-optimal alignment solution space often intersects with the structural solution space and that structural solutions cannot be guaranteed to be better than near-optimal alignments. The implication of this work is that the information contained in near-optimal alignments should be useful in understanding structural alignments.

The results of Chapter 3 demonstrate that information derived from near-optimal alignments can be used to better understand structural alignments. This is accomplished by construction of a probabilistic model that accurately predicts whether or not particular pairs of amino acids can be expected to align in structural alignments. We built a logistic regression model that incorporates three metrics derived from sets of near-optimal alignments: the frequency that an edge occurs within a set of near-optimal alignments, the robustness of an edge, and the maximum bits-per-position score for an edge. These predictor variables are shown to predict with 89% accuracy whether or not a given edge is part of a structural alignment. This is a greater than 16% improvement over prior results that use robustness alone to predict structural significance. The modeling results also provide insight into the size of the near-optimal neighborhood that should be constructed. We have found that a neighborhood of 95% of optimal provides a reasonable compromise between enough variation within the set of alignments to uncover interesting edges, yet not so large as to become unmanageable or that the predictive power of edges becomes obscured by large numbers. These results provide a concrete mechanism for researchers to identify interesting regions of alignments, predict which parts will likely be of significance, and potentially improve homology models for protein sequences without structural information.

The results to this point have shown that near-optimal alignments contain useful structural information and one technique for extracting this information. Chapter 4 describes our research into visualizing near-optimal alignments that attempts to facilitate their use by researchers without demanding programming expertise. We have developed a system that improves upon the traditional paradigm of studying single, algorithmically optimal alignments as a means for understanding the relationship between two proteins. The two parts of our strategy involve visualizing large sets of alternative, near-optimal alignments and supporting the introduction of expert knowledge. The

visualizations of alternative alignments consist of an overview path graph that provides perspective on the entire set of alignments and a detailed pairwise animation that allows for close examination of alternative alignments. The ability for users to exploit expert knowledge is facilitated by the application of highlights and filters and the ability to directly create and edit alignments.

The system is not a replacement, but rather a supplement to existing sequence analysis techniques like single, optimal alignments, database searching methods, and others. The utility of our system is that it allows detailed analysis of sets of protein alignments that was heretofore difficult to accomplish. Of particular interest is the ability to provide insight into alignments with low percent identity where other tools lose effectiveness, such as those between Lacritin and Dermcidin. The process of analyzing large sets of alignments rather than single alignments provides more information about how the two sequences align. This extra information helps develop confidence that certain sections of proteins are reliably aligned and is valuable to all alignments, not just those with low percent identity. Information about reliably aligned regions can be used to predict interesting regions of alignments.

We have demonstrate that the software does actually enhance performance by presenting two case studies in Chapter 5. Together, the two case studies demonstrate how the different features of the system can be used to effectively explore sequence alignments and their associated algorithms. The first case study shows how researchers used the animated pairwise display to confirm the homology of two proteins with weakly statistically significant expectation values. In addition to confirming past research, this increases our confidence in the ability of the software to predict regions of interest in future cases. The second case study demonstrates the flexibility of the system and its ability to be used in novel ways to support exploration and scientific discovery. This case describes how the software was used to facilitate the implementation of an $O(n)$ space near-optimal alignment generation algorithm.

In conclusion, we believe that the research presented here and the system developed represent significant improvement in our understanding of near-optimal alignments and the ability of researchers to closely study protein sequence alignments.

6.1. Future Work

This work could be extended in many ways. Some possible future projects include further analysis of the relationship between structural and near-optimal alignments, further refinement of the logistic regression model, and improvements in the software system.

The results in Chapter 2 (Figure 2.1) suggest that the using near-optimal alignments in conjunction with structural alignment algorithms could improve structural alignment algorithms. It would be interesting to study whether it would advantageous to use the near-optimal alignment with the highest structal score to either seed a structural alignment algorithm (i.e. use the alignment as an initialization point for structural alignment heuristics) or as a structural alignment itself.

While we believe that the logistic regression model is robust and stable across different inputs, it might be useful for study this further. In particular, it would be useful to characterize the performance of the logistic regression model between pairs of only two proteins rather than a sample consisting of edges from many different pairs. It would also be interesting to combine the four response variables described in Chapter 3 into a single categorical variable and develop a model based on that.

The enhanced path graph solves many of the problems inherent with static path graph presentations However one problem remains: because not all paths through the path graph are valid near-optimal alignments, the path graph can be misleading. Our solution of highlighting individual paths with animation is somewhat dissatisfying. Because this technique requires animation to see all of the valid alignments, we lose some of the benefit of a single overview display. One potential solution would be to expand the path graph into three dimensions with depth providing perspectives on single alignments.

There are several aspects of the display system that could provide beneficial results. From a usability perspective, future work on the display system should involve evaluating the effectiveness of the display software in usability studies to further refine the user interface. In terms of display features, the ability to highlight the path graph in a manner analogous to the pairwise display could

be substantially enhanced. It would also be interesting to explore the ability to dynamically filter alignments by dragging the mouse to select regions that should or should not align. This technique could be used in either the pairwise or path graph display.

A feature that has been repeatedly requested by users and would extend the functionality of the system is the ability to display alternative multiple alignments[1]. An initial problem would be the generation of alternative multiple alignments because, although possible [97], it is not clear that "efficient" translates to interactive speed. The techniques used for the animated pairwise alignment would work for multiple alignments, however it is unclear whether the path graph techniques could be applied.

[1] Alignments of three or more sequences.

Bibliography

[1] W. R. Pearson and D. J. Lipman, "Improved tools for biological sequence analysis," *Proceedings of the National Academy of Sciences*, vol. 85, pp. 2444–2448, 1988.

[2] S. F. Altshul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, pp. 403–410, 1990.

[3] A. M. Denisov, *Elements of the Theory of Inverse Problems*. Utrecht, Netherlands: VSP, 1999.

[4] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, pp. 195–197, 1981.

[5] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequences of two proteins," *Journal of Molecular Biology*, vol. 48, pp. 443–453, 1970.

[6] M. A. S. Saqi, R. B. Russell, and M. J. E. Sternberg, "Misleading local sequence alignments: implications for comparative protein modeling," *Protein Engineering*, vol. 11, pp. 627–630, 1998.

[7] M. S. Waterman and T. H. Byers, "A dynamic programming algorithm to find all solutions in a neighborhood of the optimum," *Mathematical Biosciences*, vol. 77, pp. 179–188, 1985.

[8] M. Zuker, "Suboptimal sequnce alignment in molecular biology alignment with error analysis," *Journal of Molecular Biology*, vol. 221, pp. 403–420, 1991.

[9] D. Naor and D. L. Brutlag, "On near-optimal alignments of biological sequences," *Journal of Computational Biology*, vol. 1, no. 4, pp. 349–366, 1994.

[10] M. Vingron, "Near-optimal sequence alignment," *Current Opinion in Structural Biology*, vol. 6, pp. 346–352, 1996.

[11] M. E. Smoot, S. A. Guerlain, and W. R. Pearson, "Visualization of near optimal sequence alignments," *Bioinformatics*, vol. 20, no. 6, pp. 953–958, 2004.

[12] M. E. Smoot, E. J. Bass, S. A. Guerlain, and W. R. Pearson, "A system for visualizing and analyzing near-optimal protein sequence alignments," *Information Visualization*, vol. Forthcoming, p. Accpeted for publication, 2005.

[13] C. Branden and J. Tooze, *Introduction to Protein Structure*. New York: Garland Publishing, 1999.

[14] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliand, T. N. Bhat, H. Weissig, I. N. Shindyalv, and P. E. Bourne, "The protein data bank," *Nucleic Acids Research*, vol. 28, pp. 235–242, 2000.

[15] C. H. Wu, H. Huang, L. Arminski, J. Castro-Alvear, Y. Chen, Z.-Z. Hu, R. S. Ledley, K. C. Lewis, H.-W. Mewes, B. C. Orcutt, B. E. Suzek, A. Tsugita, C. R. Vinayaka, L.-S. L. Yeh, J. Zhang, and W. C. Barker, "The protein information resource: an integrated public resource of functional annotation of proteins," *Nucleic Acids Research*, vol. 30, pp. 35–37, 2002.

[16] K. A. Dill, "Dominant forces in protein folding.," *Biochemistry*, vol. 29, pp. 7133–55, Aug 1990.

[17] E. E. Lattman, "Protein structure prediction: A special issue," *Proteins: Structure, Function, and Genetics*, vol. 23, no. 3, 1995.

[18] D. W. Mount, *Bioinformatics: Sequence and Genome Analysis*. Cold Spring Harbor Laboratory Press, 2001.

[19] C. Notredame, D. G. Higgins, and J. Heringa, "T-coffee: A novel method for fast and accurate multiple sequence alignment," *Journal of Molecular Biology*, vol. 302, pp. 205–17, Sep 2000.

[20] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis*. Cambridge University Press, 1998.

[21] L. Holm and C. Sander, "Mapping the protein universe," *Science*, vol. 273, pp. 595–602, 2 Aug. 1996.

[22] O. Gotoh, "An improved algorithm for matching biological sequences," *Journal of Molecular Biology*, vol. 162, pp. 705–708, 1982.

[23] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in Information Visualization Using Vision to Think*. San Francisco, CA: Morgan Kaufmann Publishers, 1999.

[24] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst, "Animated exploration of dynamic graphs with radial layout," in *Proceedings of the IEEE Symposium on Information Visualization 2001*, IEEE Computer Society, 2001.

[25] M. E. Tudoreanu, R. Wu, A. Hamilton-Taylor, and E. Kraemer, "Empirical evidence that algorithm animation promotes understanding of distributed algorithms," in *Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments*, IEEE Computer Society, 2002.

[26] B. B. Bederson and A. Boltman, "Does animation help users build mental maps of spatial information," Tech. Rep. 98-11, University of Maryland HCI Lab, 1998.

[27] C. Ware, *Information Visualization: Perception for design*. Addison-Wesley, 2004.

[28] L. Bartram, "Can motion increase user interface bandwitch in complex systems," in *Proceedings of IEEE SMC 1997 Intelligent Systems for the 21st Century*, vol. 2, 1997.

[29] C. Ware, E. Neufeld, and L. Bartram, "Visualizing causal relations," in *Proceedings of IEEE Information Visualization 99, October 1999*, 1999.

[30] G. Robertson, K. Cameron, M. Czerwinski, and D. Robbins, "Animated visualization of multiple intersecting hierarchies," *Information Visualization*, vol. 1, pp. 50–65, 2002.

[31] B. Tversky, J. B. Morrison, and M. Betrancourt, "Animation: can it facilitate?," *International Journal of Human-Computer Studies*, vol. 57, pp. 247–262, 2002.

[32] M. C. F. De Oliveira and H. Levkowitz, "From visual data exploration to visual data mining: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 378–394, 2003.

[33] D. A. Keim, "Information visualization and visual data mining," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1–8, 2002.

[34] E. D. Brill, J. M. Flach, L. D. Hopkins, and S. Ranjithan, "Mga: A decision support system for complex, incompletely defined problems.," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, no. 4, pp. 745–757, 1990.

[35] C. Essert-Villard, P. Shreck, P. Mathis, and J.-F. Dufourd, "Combination of automatic and interactive tools for solution space browsing," in *Proceedings of the 2003 International Conference on Geometric Modeling and Graphics*, IEEE Computer Society, 2003.

[36] D. Anderson, E. Anderson, N. Lesh, J. Marks, K. Perlin, D. Ratajczak, and K. Ryall, "Human-guided simple search: Combining information visualization and heuristic search," in *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM International conference on information and knowledge management.*, pp. 21–25, 1999.

[37] M. R. Endsley and E. O. Kiris, "The out-of-the-loop performance problem and level of control in automation," *Human Factors*, vol. 37, no. 2, pp. 381–394, 1995.

[38] R. Parasuraman, R. Molloy, and I. L. Singh, "Performance consequences of automation-induced complacency," *The International Journal of Aviation Psychology*, vol. 3, no. 1, pp. 1–23, 1993.

[39] R. Parasuraman and V. Riley, "Humans and automation: Use, misuse, disuse, abuse," *Human Factors*, vol. 39, no. 2, pp. 230–253, 1997.

[40] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, vol. 30, pp. 286–297, May 2000.

[41] J. F. Allen, E. Horvitz, and C. I. Guinn, "Mixed-initiative interaction," *IEEE Intelligent Systems*, vol. 14, no. 5, pp. 14–23, 1999.

[42] R. St.Amant and P. R. Cohen, "Interaction with a mixed-initiative system for exploratory data analysis," in *Proceedings of the 2nd Internation Conference on Intelligent User Interfaces*, pp. 15–22, ACM, 1997.

[43] C. Plaisant, D. Carr, and B. Shneiderman, "Image browsers: Taxonomy, guidelines, and informal specifications," *IEEE Software*, vol. 12, pp. 21–32, 3 1995.

[44] T. A. Keahey, "The generalized detail-in-context problem," in *Proceedings of the 1998 IEEE Symposium on Information Visualization*, pp. 44–51, IEEE Computer Society, 1998.

[45] C. Plaisant and B. Shneiderman, "Organization overviews and role management: Inspiration for future desktop environments," in *Proceedings of the Fourth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1995*, pp. 14–22, IEEE, 1995.

[46] P. Baudisch, N. Good, V. Bellotti, and P. Schraedley, "Keeping things in context: A comparative evaluation of focus plus context screens, overviews, and zooming," in *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, vol. 4, pp. 259–266, 2002.

[47] C. North, B. Shneiderman, and C. Plaisant, "User controlled overviews of an image library: A case study of the visible human," in *Proceedings of the first ACM international conference on Digital libraries*, pp. 74–82, ACM, April 1996.

[48] B. B. Bederson and J. D. Hollan, "Pad++: A zooming graphical interface for exploring alternate interface physics," in *Proceedings of the 7th annual ACM symposium on User interface software and technology*, pp. 17–26, ACM, November 1994.

[49] E. Chi, P. Barry, E. Shoop, J. Carlis, E. Retzel, and J. Riedl, "Visualization of biological sequence similarity search results," in *IEEE Visualization '95 1995*, pp. 44–51, IEEE, 1995.

[50] E. Chi, J. Riedl, E. Shoop, and P. Barry, "A novel visualization method for biological sequence similarity reports," *Electronic Imaging*, vol. 9, no. 4, pp. 394–403, 2000.

[51] E. W. Myers and W. Miller, "Optimal alignments in linear space," *Computer Applications in Biosciencs*, vol. 4, no. 1, pp. 11–17, 1988.

[52] M. O. Dayhoff, "Survey of new data and computer methods of analysis.," *Atlas of protein sequence an structure*, vol. 5, no. 3, 1978.

[53] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences*, vol. 16, pp. 10915–10919, 1992.

[54] S. F. Altshul and W. Gish, "Local alignment statistics," *Methods in Enzymology*, vol. 266, no. 27, pp. 460–480, 1996.

[55] M. Zuker, "On finding all suboptimal foldings of an rna molecule," *Science*, vol. 244, pp. 48–52, Apr. 1989.

[56] M. A. S. Saqi and M. J. E. Sternberg, "A simple method to generate non-trivial alternate alignments of protein sequences.," *Journal of Molecular Biology*, vol. 219, pp. 727–732, 1991.

[57] M. S. Waterman and M. Eggert, "A new algorithms for best subsequence alignments with application to tRNA-rRNA comparisons," *Journal of Molecular Biology*, vol. 197, pp. 723–728, 1987.

[58] M. Vingron and P. Argos, "Determination of reliable regions in protein sequence alignment," *Protein Engineering*, vol. 3, pp. 565–569, 1990.

[59] H. T. Mevissen and M. Vingron, "Quantifying the local reliability of a sequence alignment," *Protein Engineering*, vol. 9, no. 2, pp. 127–132, 1996.

[60] A. Marchler-Bauer, A. R. Panchenko, N. Ariel, and S. H. Bryant, "Comparison of sequence and structure alignments for protein domains," *Proteins: Structure, Function and Genetics*, vol. 48, pp. 439–446, 2002.

[61] J. Schultz, F. Milpetz, P. Bork, and C. P. Ponting, "Smart, a simple modular architecture research tool: identification of signaling domains," *Proceedings of the National Academy of Sciences*, vol. 95, pp. 5857–5864, 1998.

[62] E. L. Sonnhammer, S. R. Eddy, and R. Durbin, "Pfam: a comprehensive database of protein domain families based on seed alignments," *Proteins*, vol. 28, pp. 405–420, 1997.

[63] T. Madej, J.-F. Gibrat, and S. H. Bryant, "Threading a database of protein cores," *Proteins*, vol. 23, pp. 356–369, 1995.

[64] E. L. Sonnhammer and R. Durbin, "A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis.," *Gene*, vol. 167, pp. GC1–10, Dec 1995.

[65] G. W. Furnas and B. B. Bederson, "Space-scale diagrams: Understanding multiscale interfaces," in *Proceedings of CHI'95*, 1995.

[66] L. Jaroszewski, W. Li, and A. Godzik, "In search for more accurate alignments in the twilight zone," *Protein Science*, vol. 11, pp. 1702–1713, 2002.

[67] G. J. Barton and M. J. E. Sternberg, "A strategy for the rapid multiple alignment of protein sequences," *Journal of Molecular Biology*, vol. 198, pp. 327–337, 1987.

[68] C. Orengo, A. Michie, S. Jones, D. Jones, M. Swindells, and J. Thornton, "CATH–a hierarchic classification of protein domain structures.," *Structure*, vol. 5, pp. 1093–108, Aug 1997.

[69] W. Pearson, "Searching protein sequence libraries: comparison of the sensitivity and selectivity of the smith-waterman and fasta algorithms.," *Genomics*, vol. 11, pp. 635–50, Nov 1991.

[70] L. Holm and J. Park, "Dalilite workbench for protein structure comparison.," *Bioinformatics*, vol. 16, pp. 566–7, Jun 2000.

[71] C. Sander and L. Holm, "Dalilite download site: ftp://ftp.ebi.ac.uk/pub/contrib/holm/dl/," April 2005.

[72] I. N. Shindyalov and P. E. Bourne, "Protein structure alignment by incremental combinatorial extension (ce) of the optimal path," *Protein Engineering*, vol. 11, no. 9, pp. 739–747, 1998.

[73] I. N. Shindyalov and P. E. Bourne, "Ce download site: http://cl.sdsc.edu/ce.html," April 2005.

[74] G. Kleywegt, "Use of non-crystallographic symmetry in protein structure refinement.," *Acta Crystallogr D Biol Crystallogr*, vol. 52, pp. 842–57, Jul 1996.

[75] G. Kleywegt, "Lsqman download site: http://xray.bmc.uu.se/usf/," April 2005. Uppsala Software Factory.

[76] M. Levitt and M. Gerstein, "A unified statistical framework for sequence comparison and structure comparison," *Proceedings of the National Academy of Sciences*, vol. 95, pp. 5913–5920, May 1998.

[77] T. Kawabata and K. Nishikawa, "Protein structure comparison using the markov transition model of evolution.," *Proteins*, vol. 41, pp. 108–22, Oct 2000.

[78] M. Cline, R. Hughey, and K. Karplus, "Predicting reliable regions in protein sequence alignments," *Bioinformatics*, vol. 18, no. 2, pp. 306–314, 2002.

[79] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*. John Wiley & Sons, Inc., second edition ed., 2000.

[80] H. Akaike, "An information criterion," *Mathematical Sciences*, vol. 14, pp. 5–9, 1976.

[81] A. J. Dobson, *An Introduction To Generalized Linear Models Second Edition*. Chapman & Hall/CRC, 2002.

[82] D. Hosmer, T. Hosmer, S. L. Cessie, and S. Lemeshow, "A comparison of goodness-of-fit tests for the logistic regression model," *Stat Med*, vol. 16, pp. 965–80, May 1997.

[83] R Development Core Team, *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2004. 3-900051-07-0.

[84] F. E. Harrell, *Design: Design Package*, 2004. R package version 2.0-9.

[85] M. E. Smoot, "Noptalign website: http://fasta.bioch.virginia.edu/noptalign," April 2005.

[86] R. Durbin and D. Haussler, "Gff website http://www.sanger.ac.uk/software/formats/gff," April 2005.

[87] R. Spence, "Rapid, serial and visual: a presentation technique with potential," *Information Visualization*, vol. 1, pp. 13–19, 2002.

[88] B. B. Bederson, J. Meyer, and L. Good, "Jazz: an extensible zoomable user interface graphics toolkit in java," in *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pp. 171–180, 2000.

[89] E. Horvitz, "Principles of mixed-initiative user interfaces," in *Proceedings of CHI '99*, (Pittsburgh, PA, USA), pp. 159–166, ACM Press, 1999.

[90] J. E. Stajich, D. Block, K. Boulez, S. E. Brenner, S. A. Chervitz, C. Dagdigian, G. Fuellen, J. G. R. Gilber, I. Korf, H. Lapp, H. Lehvaslaiho, C. Matsalla, C. J. Mungall, B. I. Osbourne, M. R. Pocock,

P. Shattner, M. Senger, L. D. Stein, E. Stupka, M. D. Wilkinson, and E. Birney, "The bioperl toolkit: Perl modules for the life sciences," *Genome Research*, vol. 12, pp. 1611–1618, 2002.

[91] W3C SVG Working Group, "Svg website http://w3c.org/tr/svg," April 2005.

[92] S. Sanghi, R. Kumar, A. Lumsden, D. Dickinson, V. Klepeis, V. Trinkaus-Randall, H. Frierson, and G. Laurie, "cDNA and genomic cloning of lacritin, a novel secretion enhancing factor from the human lacrimal gland," *Journal of Molecular Biology*, vol. 310, pp. 127–39, Jun 2001.

[93] D. Porter, S. Weremowicz, K. Chin, P. Seth, A. Keshaviah, J. Lahti-Domenici, Y. K. Bae, C. L. Monitto, A. Merlos-Suarez, J. Chan, C. M. Hulette, A. Richardson, C. C. Morton, J. Marks, M. Duyao, R. Hruban, E. Gabrielson, R. Gelman, and K. Polyak, "A neural survival factor is a candidate oncogene in breast cancer," *Proceedings of the National Academy of Sciences*, vol. 100, pp. 10931–6, Sep 2003.

[94] B. Rost, "Twilight zone of protein sequence alignments," *Protein Engineering*, vol. 12, no. 2, pp. 85–94, 1999.

[95] J. T. Reese and W. R. Pearson, "Empirical determination of effective gap penalties for sequence comparison," *Bioinformatics*, vol. 18, no. 11, pp. 1500–1507, 2002.

[96] X. Huang and W. Miller, "A time-efficient, linear-space local similarity algorithm," *Advances in Applied Mathematics*, vol. 12, pp. 337–357, 1991.

[97] T. Shibuya and H. Imai, "Enumerating suboptimal alignments of multiple biological sequences efficiently," in *Proc. 2nd Pacific Symp. Biocomputing*, pp. 409–420, Jan 1997.

Appendix A

Protein Data

Table A.1. Protein List

A listing of the CATH IDs, expectation, percent identity, the number of alignments, and the sample set for each pair of protein domains used in Chapters 2 and 3. The number of alignments is the number of near-optimal alignments generated with a neighborhood of 95%, BLOSUM50 -10/-2, BLOSUM50 -12/-2, and BLOSUM62 -11/-1.

Sequence 1 CATH ID	Sequence 2 CATH ID	Expectation	Percent Identity	Number of Alignments	Sample
1a4704	1b90A3	8.2e-09	34.6	5552	test
1a4704	1cyg03	1.3	18.6	991	test
1akl02	1srp02	0	51.8	80203	train
1akl02	1cglA0	0.00096	22.4	16573	train
1akl02	1bqqM0	0.14	20.2	15053	train
1ao6A5	1uor03	2.2e-07	19.5	3110	train
1aqzB0	1rtu00	0.013	21.9	4587	train
1aqzB0	1rds00	1.7	23.3	5099	train
1auq00	1ao3A0	1.6e-08	20.8	15586	train
1b2rA2	1bx0A2	1.4013e-45	52.1	37782	test
1b2rA2	1amoA4	8e-09	29.5	16106	test
1b2rA2	1ndh02	0.00068	21.3	7389	test
1b5600	1pmpA0	5.9e-37	55.6	20160	train
1b5600	1dc9A0	1.1e-09	24.4	7769	train

1b5600	1lfo00	0.044	19.4	5126	train
1b5600	1mdc00	0.22	21.2	3826	train
1bbhA0	1cpq00	9.3e-08	26.3	8859	test
1bcg00	1b7dA0	9.6e-07	33.3	2888	train
1bhxB0	1autC2	4.4e-14	29.9	8933	train
1bhxB0	1ddjA2	1.8e-10	26.2	8726	train
1bhxB0	1b0fA2	0.0057	19.0	3165	train
1bhxB0	2kaiB0	0.22	15.0	6991	train
1bylA0	1qtoA0	3.7e-35	58.5	17467	train
1cd2A0	1vdrA0	2.7e-08	24.2	16259	test
1ce7B2	2aaiB2	1.1e-24	51.6	17944	test
1ce7B2	1abrB1	0.0041	21.8	4050	test
1ce7B2	1ce7B1	0.12	22.7	3840	test
1ck4B0	1ao3A0	1.6e-09	24.0	16822	train
1cl7H0	1ae6H1	1.9e-28	48.9	19096	test
1cl7H0	2hmiD1	1e-10	27.3	12500	test
1cl7H0	2rhe00	0.00051	26.7	5799	test
1cl7H0	1cf8H2	0.1	13.6	5918	test
1cm8A2	1erk02	0	42.8	56043	test
1cm8A2	1agwA2	5.8e-10	24.6	22469	test
1cm8A2	1ckjA2	0.00056	19.6	13187	test
1cm8A2	1csn02	0.29	17.2	11238	test
1cv2A0	1cqwA0	0	48.5	118203	train
1cv2A0	1cqzB0	6.1e-09	13.7	586749	train
1cv2A0	1a7uA0	0.004	21.8	35270	train
1cv2A0	1qj4A0	1.2	18.0	17847	train
1cvuA1	1xkbA1	6.6e-06	38.1	617	test
1cvuA1	1autL1	0.0046	29.2	701	test

1cvuA1	1klo02	0.65	28.8	710	test
1d0gR1	1extB1	0.0028	17.2	316	test
1eepA0	1b3oA0	6.7e-40	37.5	100615	train
1eepA0	1rpxA0	0.025	18.8	17775	train
1eepA0	1gylA0	1.1	19.6	36481	train
1entE2	1aptE2	3.7e-33	49.4	29628	train
1entE2	1psn02	3.7e-09	28.2	16910	train
1entE2	1mpp02	0.0039	22.9	12557	train
1entE2	1pfzA2	0.11	15.5	7959	train
1etpA1	1cnoG0	1.4e-14	44.1	7212	test
1etpA1	1fcdC1	0.0012	26.4	2659	test
1etpA1	1b7vA0	0.2	25.0	1737	test
1extB1	1tnrR2	0.37	22.0	1065	train
1f2lD0	1dokA0	7.4e-10	35.1	2593	train
1f2lD0	1qe6D0	0.00092	25.7	1570	train
1f2lD0	1tvxB0	1.6	20.5	543	train
1frrA0	1fxiA0	5.8e-28	58.3	10316	test
1frrA0	1qlaB1	0.023	24.3	2276	test
1frrA0	1c4aA1	0.24	25.3	1225	test
1hdaB0	1outB0	2.2e-35	47.9	23460	test
1hdaB0	1myt00	2.1e-10	26.8	9599	test
1hdaB0	1hbiA0	0.0026	22.3	4883	test
1hdaB0	1ash00	0.12	21.9	3958	test
1ihbA0	1awcB0	1.6e-09	25.9	9350	test
1jafA0	2ccyA0	1.8e-10	36.4	11138	test
1mpyA2	1dhy02	0.0061	23.7	5245	test
1mpyA2	1mpyA1	0.37	19.3	3639	test
1nfiC1	1iknC0	4.1e-26	42.6	13248	test

1nfiC1	1a02N2	0.013	18.7	2338	test
1p3801	1jnk01	4.6e-27	43.3	12894	train
1p3801	1vr2A1	0.00081	23.0	2917	train
1p3801	1bygA1	0.35	17.7	1290	train
1qkmA0	1qktA0	0	57.0	82962	test
1qkmA0	1dkfB0	1.3e-10	19.9	19257	test
1qkmA0	2prgB0	0.0021	20.2	17130	test
1qkmA0	1gwxA0	0.96	16.4	12728	test
1rds00	1fus00	1e-24	54.6	11891	train
1rds00	1rtu00	7.6e-10	31.4	8707	train
1rmg00	1czfA0	3.7e-08	19.7	54615	train
1svy00	1d0nA2	1.3e-15	36.0	6915	train
1svy00	1d0nA4	0.0082	21.3	1842	train
1svy00	1d0nA6	0.18	23.4	1860	train
1tmo04	1eu1A4	3.9e-37	52.0	36455	train
1tmo04	1fdi04	0.0095	22.3	4410	train
1tmo04	2napA4	1.3	19.1	4341	train
2hpdA0	1egyA0	2.3e-07	19.5	81064	test
3sxlA2	1b7fA1	0.00017	25.6	280	train
3sxlA2	1ha102	0.0022	17.1	323	train
3sxlA2	1urnA0	0.17	15.6	799	train
4mdhA2	1bdmB2	2e-38	49.4	37450	test
4mdhA2	1d3aA1	0.022	20.9	7201	test
4mdhA2	1ldm02	0.29	20.5	6898	test

Appendix B

Sample Statistics

Table B.1. Training Sample Sizes

The number of edges for each scoring parameter combination and near-optimal neighborhood in the training sample.

Neighborhood	BLOSUM50 -10/-2	BLOSUM50 -12-2	BLOSUM62 -11/-1	Combined
Optimal	7271	7158	7223	21652
95%	17482	15249	14042	46773
75%	127634	114691	106611	348936

Table B.2. Testing Sample Sizes

The number of edges for each scoring parameter combination and near-optimal neighborhood in the test sample.

Neighborhood	BLOSUM50 -10/-2	BLOSUM50 -12-2	BLOSUM62 -11/-1	Combined
Optimal	6576	6433	6525	19534
95%	12880	11220	10735	34835
75%	83798	70518	75636	229952

Appendix C

Model Factor ANOVA Results

ANOVA results determining whether model parameter estimates were independent of sample size and scoring parameter combination for each of the 4 possible response thresholds.

```
[1] "Threshold:  1"
[1] "Robust"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.06091 0.01523  0.1231 0.9714
factor(alg)     3 0.33994 0.11331  0.9161 0.4623
Residuals      12 1.48424 0.12369
[1] "Frequency"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.14596 0.03649  0.4473 0.7725
factor(alg)     3 0.30795 0.10265  1.2583 0.3324
Residuals      12 0.97891 0.08158
[1] "Maximum bits-per-position"
               Df Sum Sq Mean Sq F value    Pr(>F)
factor(sample)  4 0.1620  0.0405  0.1929 0.9374358
factor(alg)     3 9.7937  3.2646 15.5503 0.0001953 ***
Residuals      12 2.5192  0.2099
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


[1] "Threshold:  2"
[1] "Robust"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.07963 0.01991  0.1162 0.9742
factor(alg)     3 0.66266 0.22089  1.2896 0.3227
Residuals      12 2.05534 0.17128
[1] "Frequency"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.14190 0.03547  0.2787 0.8861
factor(alg)     3 0.59148 0.19716  1.5490 0.2528
Residuals      12 1.52743 0.12729
[1] "Maximum bits-per-position"
               Df  Sum Sq Mean Sq F value    Pr(>F)
factor(sample)  4  0.5443  0.1361  0.4915 0.7423000
factor(alg)     3 10.1426  3.3809 12.2110 0.0005876 ***
Residuals      12  3.3225  0.2769
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


[1] "Threshold:  3"
[1] "Robust"
               Df Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.3194  0.0799  0.2907 0.8784
factor(alg)     3 0.9764  0.3255  1.1850 0.3566
Residuals      12 3.2959  0.2747
[1] "Frequency"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.28262 0.07065  0.3467 0.8413
factor(alg)     3 0.95350 0.31783  1.5597 0.2503
Residuals      12 2.44541 0.20378
[1] "Maximum bits-per-position"
```

```
               Df  Sum Sq Mean Sq F value   Pr(>F)
factor(sample)  4  0.3878  0.0969  0.2444 0.907573
factor(alg)     3 12.5553  4.1851 10.5513 0.001106 **
Residuals      12  4.7597  0.3966
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


[1] "Threshold:  4"
[1] "Robust"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 1.06422 0.26605  1.6891 0.2168
factor(alg)     3 0.61945 0.20648  1.3109 0.3162
Residuals      12 1.89015 0.15751
[1] "Frequency"
               Df  Sum Sq Mean Sq F value Pr(>F)
factor(sample)  4 0.66212 0.16553  0.9745 0.4570
factor(alg)     3 0.95178 0.31726  1.8677 0.1889
Residuals      12 2.03842 0.16987
[1] "Maximum bits-per-position"
               Df  Sum Sq Mean Sq F value   Pr(>F)
factor(sample)  4  1.2602  0.3150  1.7121   0.2118
factor(alg)     3 19.7619  6.5873 35.7980 2.89e-06 ***
Residuals      12  2.2082  0.1840
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Appendix D

Model Analysis Output

The logistic regession results as returned by R.

```
[1] "=================================================================="
[1] "Formula used: "
struct ~ freq + robust + mbits
[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
(Intercept)         freq       robust        mbits
     -9.806        4.720        5.905        2.068
Degrees of Freedom: 4999 Total (i.e. Null);  4996 Residual
Null Deviance:     6237
Residual Deviance: 3440  AIC: 3448
[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0689  -0.3647  -0.2583   0.5933   2.9445
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -9.8063     1.9190  -5.110 3.22e-07 ***
freq          4.7204     0.1334  35.382  < 2e-16 ***
robust        5.9045     2.0161   2.929  0.00340 **
mbits         2.0680     0.1848  11.188  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6236.6  on 4999  degrees of freedom
Residual deviance: 3440.4  on 4996  degrees of freedom
AIC: 3448.4
Number of Fisher Scoring iterations: 5
[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                      4999     6236.6
freq      1   2634.8      4998     3601.9       0.0
robust    1     15.7      4997     3586.1 7.345e-05
mbits     1    145.8      4996     3440.4 1.460e-33
[1] "plot glm"
[1] "predict"
[1] "ROC"
[1] "Area under ROC curve:"
     Model Area.adj p.adj     Area p-value binorm.area
1 Model  1 0.890626     0 0.890626       0          NA
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3421 1579
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      5e-11    2796.25          3          0      0.908      0.817
     Gamma      Tau-a         R2      Brier
     0.819      0.353      0.601      0.104
```

```
          Coef   S.E.   Wald Z P
Intercept -9.806 1.9191 -5.11  0.0000
freq       4.720 0.1334 35.38  0.0000
robust     5.905 2.0162  2.93  0.0034
mbits      2.068 0.1849 11.19  0.0000
[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
         5.199493e+02          5.325435e+02          2.005560e+00
                    Z                      P
        -6.279619e+00          3.394047e-10
[1] "================================================================="
[1] "Formula used: "
struct ~ freq * robust * mbits
[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
      (Intercept)              freq            robust             mbits
            41.48            -49.35            -46.07           -111.69
      freq:robust         freq:mbits       robust:mbits  freq:robust:mbits
            54.39             76.91            113.77            -71.99
Degrees of Freedom: 4999 Total (i.e. Null);  4992 Residual
Null Deviance:     6237
Residual Deviance: 3118  AIC: 3134
[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9195  -0.3509  -0.1939   0.2794   3.2653
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)         41.484      6.563   6.321 2.60e-10 ***
freq               -49.354      7.788  -6.337 2.34e-10 ***
robust             -46.071      6.857  -6.719 1.83e-11 ***
mbits             -111.693     27.111  -4.120 3.79e-05 ***
freq:robust         54.390      8.109   6.707 1.99e-11 ***
freq:mbits          76.912     34.416   2.235   0.0254 *
robust:mbits       113.770     28.346   4.014 5.98e-05 ***
freq:robust:mbits  -71.989     35.850  -2.008   0.0446 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6236.6  on 4999  degrees of freedom
Residual deviance: 3118.1  on 4992  degrees of freedom
AIC: 3134.1
Number of Fisher Scoring iterations: 7
[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
                   Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                                  4999     6236.6
freq                1   2634.8      4998     3601.9       0.0
robust              1     15.7      4997     3586.1 7.345e-05
mbits               1    145.8      4996     3440.4 1.460e-33
freq:robust         1     11.7      4995     3428.7 6.237e-04
freq:mbits          1    279.3      4994     3149.4 1.092e-62
robust:mbits        1     27.1      4993     3122.4 1.976e-07
freq:robust:mbits   1      4.2      4992     3118.1 3.957e-02
[1] "plot glm"
[1] "predict"
[1] "ROC"
[1] "Area under ROC curve:"
     Model Area.adj p.adj     Area p-value binorm.area
1 Model  1 0.908444     0 0.908444       0          NA
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3421 1579
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      2e-10     3118.5          7          0      0.928      0.855
     Gamma      Tau-a         R2      Brier
     0.856       0.37      0.651      0.099
                    Coef    S.E.   Wald Z P
Intercept             41.48  6.563  6.32  0.0000
freq                 -49.35  7.788 -6.34  0.0000
robust               -46.07  6.857 -6.72  0.0000
mbits               -111.69 27.111 -4.12  0.0000
freq * robust         54.39  8.109  6.71  0.0000
freq * mbits          76.91 34.416  2.23  0.0254
robust * mbits       113.77 28.346  4.01  0.0001
freq * robust * mbits -71.99 35.850 -2.01  0.0446
[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
          494.9982380           493.2419211          2.0653061
                    Z                      P
            0.8503906              0.3951080
[1] "================================================================="
[1] "Formula used: "
struct ~ poly(freq, 2) + poly(robust, 2) + poly(mbits, 2)
```

```
[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
     (Intercept)    poly(freq, 2)1    poly(freq, 2)2  poly(robust, 2)1
          -1.503           153.176            -9.012            14.224
poly(robust, 2)2   poly(mbits, 2)1   poly(mbits, 2)2
          13.525            45.554           -22.094
Degrees of Freedom: 4999 Total (i.e. Null);  4993 Residual
Null Deviance:     6237
Residual Deviance: 3372  AIC: 3386
[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0035  -0.3220  -0.2718   0.5070   3.1777
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)       -1.50335    0.05943 -25.297  < 2e-16 ***
poly(freq, 2)1   153.17566    4.59308  33.349  < 2e-16 ***
poly(freq, 2)2    -9.01214    2.89360  -3.115  0.00184 **
poly(robust, 2)1  14.22428    3.07510   4.626 3.73e-06 ***
poly(robust, 2)2  13.52522    2.78759   4.852 1.22e-06 ***
poly(mbits, 2)1   45.55368    3.92487  11.606  < 2e-16 ***
poly(mbits, 2)2  -22.09358    3.48996  -6.331 2.44e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6236.6  on 4999  degrees of freedom
Residual deviance: 3371.9  on 4993  degrees of freedom
AIC: 3385.9
Number of Fisher Scoring iterations: 6
[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
                 Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                              4999     6236.6
poly(freq, 2)     2   2635.5      4997     3601.1       0.0
poly(robust, 2)   2     46.0      4995     3555.1 1.006e-10
poly(mbits, 2)    2    183.2      4993     3371.9 1.639e-40
[1] "plot glm"
[1] "predict"
[1] "ROC"
[1] "Area under ROC curve:"
     Model  Area.adj p.adj      Area p-value binorm.area
1 Model  1 0.8973603     0 0.8973603       0          NA
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3421 1579
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      1e-09    2864.75          6          0      0.909      0.818
     Gamma      Tau-a         R2      Brier
      0.82      0.354      0.612      0.102
          Coef    S.E.    Wald Z P
Intercept  -1.503 0.05943 -25.30 0.0000
1         153.176 4.59308  33.35 0.0000
2          -9.012 2.89360  -3.11 0.0018
1          14.224 3.07510   4.63 0.0000
2          13.525 2.78759   4.85 0.0000
1          45.554 3.92488  11.61 0.0000
2         -22.094 3.48997  -6.33 0.0000
[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
           509.008457            522.269358              1.538323
                    Z                       P
            -8.620361              0.000000
[1] "================================================================="
[1] "Formula used: "
struct ~ poly(freq, 2) * poly(robust, 2) * poly(mbits, 2)
[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
                                  (Intercept)
                                    -1.198e+00
                                 poly(freq, 2)1
                                     1.468e+02
                                 poly(freq, 2)2
                                     1.382e+01
                                poly(robust, 2)1
                                     4.622e+01
                                poly(robust, 2)2
                                     3.117e+01
                                 poly(mbits, 2)1
                                     2.937e+01
                                 poly(mbits, 2)2
                                    -3.813e+00
                poly(freq, 2)1:poly(robust, 2)1
                                     5.705e+02
```

```
               poly(freq, 2)2:poly(robust, 2)1
                                     1.951e+03
               poly(freq, 2)1:poly(robust, 2)2
                                    -3.682e+02
               poly(freq, 2)2:poly(robust, 2)2
                                     1.417e+03
                poly(freq, 2)1:poly(mbits, 2)1
                                     2.908e+03
                poly(freq, 2)2:poly(mbits, 2)1
                                     7.887e+02
                poly(freq, 2)1:poly(mbits, 2)2
                                    -1.919e+03
                poly(freq, 2)2:poly(mbits, 2)2
                                     4.938e+02
              poly(robust, 2)1:poly(mbits, 2)1
                                     7.819e+03
              poly(robust, 2)2:poly(mbits, 2)1
                                     4.536e+03
              poly(robust, 2)1:poly(mbits, 2)2
                                     3.675e+03
              poly(robust, 2)2:poly(mbits, 2)2
                                     2.267e+03
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)1
                                    -2.378e+05
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)1
                                     2.085e+05
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)1
                                    -2.057e+05
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)1
                                     1.260e+05
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)2
                                    -1.279e+05
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)2
                                     1.356e+05
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)2
                                    -1.554e+05
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)2
                                     9.515e+04
Degrees of Freedom: 4999 Total (i.e. Null);  4973 Residual
Null Deviance:     6237
Residual Deviance: 3025  AIC: 3079
[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5832  -0.3931  -0.1777   0.1821   4.5436
Coefficients:
                                                Estimate Std. Error z value
(Intercept)                                    -1.198e+00  1.214e-01  -9.862
poly(freq, 2)1                                  1.468e+02  7.944e+00  18.476
poly(freq, 2)2                                  1.382e+01  5.598e+00   2.469
poly(robust, 2)1                                4.622e+01  1.479e+01   3.125
poly(robust, 2)2                                3.117e+01  1.728e+01   1.804
poly(mbits, 2)1                                 2.937e+01  1.244e+01   2.361
poly(mbits, 2)2                                -3.813e+00  1.046e+01  -0.365
poly(freq, 2)1:poly(robust, 2)1                 5.705e+02  9.044e+02   0.631
poly(freq, 2)2:poly(robust, 2)1                 1.951e+03  5.382e+02   3.625
poly(freq, 2)1:poly(robust, 2)2                -3.682e+02  1.108e+03  -0.332
poly(freq, 2)2:poly(robust, 2)2                 1.417e+03  6.378e+02   2.222
poly(freq, 2)1:poly(mbits, 2)1                  2.908e+03  8.002e+02   3.634
poly(freq, 2)2:poly(mbits, 2)1                  7.887e+02  5.220e+02   1.511
poly(freq, 2)1:poly(mbits, 2)2                 -1.919e+03  6.566e+02  -2.923
poly(freq, 2)2:poly(mbits, 2)2                  4.938e+02  4.362e+02   1.132
poly(robust, 2)1:poly(mbits, 2)1                7.819e+03  1.407e+03   5.557
poly(robust, 2)2:poly(mbits, 2)1                4.536e+03  1.699e+03   2.670
poly(robust, 2)1:poly(mbits, 2)2                3.675e+03  1.068e+03   3.442
poly(robust, 2)2:poly(mbits, 2)2                2.267e+03  1.445e+03   1.569
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)1 -2.378e+05  8.572e+04  -2.774
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)1  2.085e+05  5.229e+04   3.987
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)1 -2.057e+05  1.082e+05  -1.901
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)1  1.260e+05  6.127e+04   2.057
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)2 -1.279e+05  6.493e+04  -1.970
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)2  1.356e+05  4.149e+04   3.268
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)2 -1.554e+05  9.197e+04  -1.689
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)2  9.515e+04  5.030e+04   1.892
                                               Pr(>|z|)
(Intercept)                                     < 2e-16 ***
poly(freq, 2)1                                  < 2e-16 ***
poly(freq, 2)2                                 0.013552 *
poly(robust, 2)1                               0.001781 **
poly(robust, 2)2                               0.071264 .
poly(mbits, 2)1                                0.018246 *
poly(mbits, 2)2                                0.715352
poly(freq, 2)1:poly(robust, 2)1                0.528176
poly(freq, 2)2:poly(robust, 2)1                0.000289 ***
poly(freq, 2)1:poly(robust, 2)2                0.739535
poly(freq, 2)2:poly(robust, 2)2                0.026314 *
poly(freq, 2)1:poly(mbits, 2)1                 0.000279 ***
poly(freq, 2)2:poly(mbits, 2)1                 0.130775
poly(freq, 2)1:poly(mbits, 2)2                 0.003467 **
poly(freq, 2)2:poly(mbits, 2)2                 0.257598
```

```
poly(robust, 2)1:poly(mbits, 2)1               2.74e-08 ***
poly(robust, 2)2:poly(mbits, 2)1               0.007592 **
poly(robust, 2)1:poly(mbits, 2)2               0.000578 ***
poly(robust, 2)2:poly(mbits, 2)2               0.116748
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)1 0.005533 **
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)1 6.70e-05 ***
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)1 0.057317 .
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)1 0.039712 *
poly(freq, 2)1:poly(robust, 2)1:poly(mbits, 2)2 0.048795 *
poly(freq, 2)2:poly(robust, 2)1:poly(mbits, 2)2 0.001083 **
poly(freq, 2)1:poly(robust, 2)2:poly(mbits, 2)2 0.091129 .
poly(freq, 2)2:poly(robust, 2)2:poly(mbits, 2)2 0.058549 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6236.6  on 4999  degrees of freedom
Residual deviance: 3024.8  on 4973  degrees of freedom
AIC: 3078.8
Number of Fisher Scoring iterations: 9
[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
                                             Df Deviance Resid. Df Resid. Dev
NULL                                                          4999     6236.6
poly(freq, 2)                                 2   2635.5      4997     3601.1
poly(robust, 2)                               2     46.0      4995     3555.1
poly(mbits, 2)                                2    183.2      4993     3371.9
poly(freq, 2):poly(robust, 2)                 4     35.0      4989     3336.8
poly(freq, 2):poly(mbits, 2)                  4    246.6      4985     3090.2
poly(robust, 2):poly(mbits, 2)                4     30.8      4981     3059.4
poly(freq, 2):poly(robust, 2):poly(mbits, 2)  8     34.5      4973     3024.8
                                            P(>|Chi|)
NULL
poly(freq, 2)                                     0.0
poly(robust, 2)                                1.006e-10
poly(mbits, 2)                                 1.639e-40
poly(freq, 2):poly(robust, 2)                  4.550e-07
poly(freq, 2):poly(mbits, 2)                   3.444e-52
poly(robust, 2):poly(mbits, 2)                 3.322e-06
poly(freq, 2):poly(robust, 2):poly(mbits, 2) 3.255e-05
[1] "plot glm"
[1] "predict"
[1] "ROC"
[1] "Area under ROC curve:"
     Model  Area.adj p.adj      Area p-value binorm.area
1 Model  1 0.9070894     0 0.9070894       0          NA
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3421 1579
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      3e-09    3211.79         26          0      0.931      0.862
     Gamma      Tau-a         R2      Brier
     0.863      0.373      0.665      0.097
          Coef       S.E.      Wald Z P
Intercept -1.198e+00 1.214e-01 -9.86  0.0000
1          1.468e+02 7.944e+00 18.48  0.0000
2          1.382e+01 5.598e+00  2.47  0.0136
1          4.622e+01 1.479e+01  3.12  0.0018
2          3.117e+01 1.728e+01  1.80  0.0713
1          2.937e+01 1.244e+01  2.36  0.0182
2         -3.813e+00 1.046e+01 -0.36  0.7154
1 * 1      5.705e+02 9.044e+02  0.63  0.5282
2 * 1      1.951e+03 5.382e+02  3.62  0.0003
1 * 2     -3.682e+02 1.108e+03 -0.33  0.7395
2 * 2      1.417e+03 6.378e+02  2.22  0.0263
1 * 1      2.908e+03 8.002e+02  3.63  0.0003
2 * 1      7.887e+02 5.220e+02  1.51  0.1308
1 * 2     -1.919e+03 6.566e+02 -2.92  0.0035
2 * 2      4.938e+02 4.362e+02  1.13  0.2576
1 * 1      7.819e+03 1.407e+03  5.56  0.0000
2 * 1      4.536e+03 1.699e+03  2.67  0.0076
1 * 2      3.675e+03 1.068e+03  3.44  0.0006
2 * 2      2.267e+03 1.445e+03  1.57  0.1167
1 * 1 * 1 -2.378e+05 8.572e+04 -2.77  0.0055
2 * 1 * 1  2.085e+05 5.229e+04  3.99  0.0001
1 * 2 * 1 -2.057e+05 1.082e+05 -1.90  0.0573
2 * 2 * 1  1.260e+05 6.127e+04  2.06  0.0397
1 * 1 * 2 -1.279e+05 6.493e+04 -1.97  0.0488
2 * 1 * 2  1.356e+05 4.149e+04  3.27  0.0011
1 * 2 * 2 -1.554e+05 9.197e+04 -1.69  0.0911
2 * 2 * 2  9.515e+04 5.030e+04  1.89  0.0585
[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
         4.839148e+02          4.791093e+02         9.819274e-01
                    Z                      P
         4.893871e+00          9.887179e-07
Analysis of Deviance Table
Model 1: struct ~ freq + robust + mbits
```

```
Model 2: struct ~ freq * robust * mbits
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4996     3440.4
2      4992     3118.1    4    322.3 1.714e-68
Analysis of Deviance Table
Model 1: struct ~ freq + robust + mbits
Model 2: struct ~ poly(freq, 2) + poly(robust, 2) + poly(mbits, 2)
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4996     3440.4
2      4993     3371.9    3     68.5 8.962e-15
Analysis of Deviance Table
Model 1: struct ~ freq + robust + mbits
Model 2: struct ~ poly(freq, 2) * poly(robust, 2) * poly(mbits, 2)
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4996     3440.4
2      4973     3024.8   23    415.5 1.119e-73
Analysis of Deviance Table
Model 1: struct ~ freq * robust * mbits
Model 2: struct ~ poly(freq, 2) + poly(robust, 2) + poly(mbits, 2)
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4992     3118.1
2      4993     3371.9   -1   -253.8 3.943e-57
Analysis of Deviance Table
Model 1: struct ~ freq * robust * mbits
Model 2: struct ~ poly(freq, 2) * poly(robust, 2) * poly(mbits, 2)
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4992    3118.13
2      4973    3024.84   19    93.29 8.635e-12
Analysis of Deviance Table
Model 1: struct ~ poly(freq, 2) + poly(robust, 2) + poly(mbits, 2)
Model 2: struct ~ poly(freq, 2) * poly(robust, 2) * poly(mbits, 2)
  Resid. Df Resid. Dev   Df Deviance P(>|Chi|)
1      4993     3371.9
2      4973     3024.8   20    347.0 1.813e-61
```

Appendix E

Final Logistic Regression Models

The final model for a sample size of 5000, combined scoring parameters, and the response threshold of 25% (1 of 4 structural alignments) is as follows, where l_{SI1} is the log odds of structural inclusion:

$$l_{SI1} = -7.418 + (4.374 * frequncy) + (3.919 * robustness) + (1.816 * maxbits) \quad \text{(E.0.1)}$$

The final model for a sample size of 5000, combined scoring parameters, and the response threshold of 50% (two of four structural alignments) is as follows, where l_{SI2} is the log odds of structural inclusion:

$$l_{SI2} = -9.806 + (4.720 * frequency) + (5.905 * robustness) + (2.068 * maxbits) \quad \text{(E.0.2)}$$

The final model for a sample size of 5000, combined scoring parameters, and the response threshold of 75% (three of four structural alignments) is as follows, where l_{SI3} is the log odds of structural inclusion:

$$l_{SI3} = -9.857 + (4.692 * frequency) + (5.594 * robustness) + (2.337 * maxbits) \quad \text{(E.0.3)}$$

The final model for a sample size of 5000, combined scoring parameters, and the response threshold of 100% (four of four structural alignments) is as follows, where l_{SI4} is the log odds of structural inclusion:

$$l_{SI4} = -11.337 + (4.829 * frequency) + (6.152 * robustness) + (3.210 * maxbits) \quad \text{(E.0.4)}$$

The complete R output describing the models and their statistics follows.

```
TRAINING DIR:  train.family.logistic
TEST DIR:      test.family.logistic

=========================================================

SAMPLE size: 5000  alg id: 95  scr id: 95

[1] "Formula used: "
struct ~ freq + robust + mbits

[1] "threshold: 0.25"

[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
(Intercept)         freq       robust        mbits
     -7.418        4.374        3.919        1.816
Degrees of Freedom: 4999 Total (i.e. Null);  4996 Residual
Null Deviance:     6375
Residual Deviance: 3715  AIC: 3723

[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9683  -0.4220  -0.3045   0.6491   2.7682
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -7.4178     1.8625  -3.983 6.81e-05 ***
freq          4.3744     0.1199  36.482  < 2e-16 ***
robust        3.9190     1.9566   2.003   0.0452 *
mbits         1.8158     0.1721  10.551  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6375.3  on 4999  degrees of freedom
Residual deviance: 3714.7  on 4996  degrees of freedom
AIC: 3722.7
Number of Fisher Scoring iterations: 5

[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                      4999     6375.3
freq      1   2526.9      4998     3848.3       0.0
robust    1      7.5      4997     3840.8 6.002e-03
mbits     1    126.1      4996     3714.7 2.991e-29

[1] "plot glm"

[1] "termplot"
                   2.5 %    97.5 %
(Intercept) -11.06909553 -3.766462
freq          4.13931518  4.609448
robust        0.08313872  7.754844
mbits         1.47839592  2.153122
```

```
[1] "mikes gof"
Sum of squared errors     Expected value|H0                    SD
         5.657670e+02          5.731394e+02          1.847317e+00
                    Z                      P
        -3.990839e+00          6.583988e-05

[1] "lrm print"
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3326 1674
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      8e-14    2660.52          3          0      0.898      0.795
     Gamma      Tau-a         R2      Brier
     0.798      0.354      0.573      0.113
          Coef   S.E.   Wald Z P
Intercept -7.418 1.8625 -3.98  0.0001
freq       4.374 0.1199 36.48  0.0000
robust     3.919 1.9566  2.00  0.0452
mbits      1.816 0.1721 10.55  0.0000

[1] "lrm anova"
                Wald Statistics          Response: struct
 Factor     Chi-Square d.f. P
 freq       1330.90    1    <.0001
 robust        4.01    1    0.0452
 mbits       111.33    1    <.0001
 TOTAL      1554.34    3    <.0001

[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
         5.657670e+02          5.731394e+02          1.847317e+00
                    Z                      P
        -3.990839e+00          6.583990e-05

[1] "threshold: 0.5"

[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
(Intercept)         freq       robust        mbits
     -9.806        4.720        5.905        2.068
Degrees of Freedom: 4999 Total (i.e. Null);  4996 Residual
Null Deviance:     6237
Residual Deviance: 3440  AIC: 3448

[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0689  -0.3647  -0.2583   0.5933   2.9445
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -9.8063     1.9190  -5.110 3.22e-07 ***
freq          4.7204     0.1334  35.382  < 2e-16 ***
robust        5.9045     2.0161   2.929  0.00340 **
mbits         2.0680     0.1848  11.188  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6236.6  on 4999  degrees of freedom
Residual deviance: 3440.4  on 4996  degrees of freedom
AIC: 3448.4
Number of Fisher Scoring iterations: 5

[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                      4999     6236.6
freq      1   2634.8      4998     3601.9       0.0
robust    1     15.7      4997     3586.1 7.345e-05
mbits     1    145.8      4996     3440.4 1.460e-33

[1] "plot glm"

[1] "termplot"
                2.5 %    97.5 %
(Intercept) -13.568352 -6.044214
freq          4.458888  4.981989
robust        1.952034  9.856979
mbits         1.705647  2.430403

[1] "mikes gof"
Sum of squared errors     Expected value|H0                    SD
         5.199493e+02          5.325435e+02          2.005560e+00
                    Z                      P
        -6.279625e+00          3.393903e-10
```

```
[1] "lrm print"
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3421 1579
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      5e-11    2796.25          3          0      0.908      0.817
     Gamma      Tau-a         R2      Brier
     0.819      0.353      0.601      0.104
          Coef   S.E.   Wald Z P
Intercept -9.806 1.9191 -5.11  0.0000
freq       4.720 0.1334 35.38  0.0000
robust     5.905 2.0162  2.93  0.0034
mbits      2.068 0.1849 11.19  0.0000

[1] "lrm anova"
                Wald Statistics          Response: struct
 Factor     Chi-Square d.f. P
 freq       1251.44    1    <.0001
 robust        8.58    1    0.0034
 mbits       125.14    1    <.0001
 TOTAL      1456.58    3    <.0001

[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
         5.199493e+02          5.325435e+02          2.005560e+00
                    Z                      P
        -6.279619e+00          3.394047e-10

[1] "threshold: 0.75"

[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
(Intercept)         freq       robust        mbits
     -9.857        4.692        5.594        2.337
Degrees of Freedom: 4999 Total (i.e. Null);  4996 Residual
Null Deviance:     5948
Residual Deviance: 3416  AIC: 3424

[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9967  -0.3533  -0.2378   0.5003   3.0664
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -9.8572     1.9130  -5.153 2.57e-07 ***
freq          4.6921     0.1416  33.128  < 2e-16 ***
robust        5.5935     2.0126   2.779  0.00545 **
mbits         2.3367     0.1851  12.624  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 5948.3  on 4999  degrees of freedom
Residual deviance: 3416.3  on 4996  degrees of freedom
AIC: 3424.3
Number of Fisher Scoring iterations: 6

[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                      4999     5948.3
freq      1   2323.6      4998     3624.7       0.0
robust    1     18.4      4997     3606.3 1.763e-05
mbits     1    190.0      4996     3416.3 3.203e-43

[1] "plot glm"

[1] "termplot"
                 2.5 %    97.5 %
(Intercept) -13.607566 -6.106786
freq          4.414464  4.969798
robust        1.648002  9.539068
mbits         1.973856  2.699632

[1] "mikes gof"
Sum of squared errors     Expected value|H0                    SD
           522.084794            540.451531          2.048829
                    Z                      P
            -8.964505              0.000000

[1] "lrm print"
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
```

```
   0    1
3590 1410
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      6e-10       2532          3          0      0.901      0.801
     Gamma      Tau-a         R2      Brier
     0.803      0.325      0.571      0.104
          Coef   S.E.   Wald Z P
Intercept -9.857 1.9130 -5.15  0.0000
freq       4.692 0.1416 33.13  0.0000
robust     5.594 2.0126  2.78  0.0054
mbits      2.337 0.1851 12.62  0.0000

[1] "lrm anova"
               Wald Statistics          Response: struct
 Factor     Chi-Square d.f. P
 freq       1097.48    1    <.0001
 robust        7.72    1    0.0054
 mbits       159.36    1    <.0001
 TOTAL      1258.67    3    <.0001

[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
           522.084794            540.451531              2.048829
                    Z                       P
            -8.964505              0.000000

[1] "threshold: 1"

[1] "glm print:"
Call:  glm(formula = formula, family = binomial, data = trainSet)
Coefficients:
(Intercept)         freq       robust        mbits
    -11.337        4.829        6.152        3.210
Degrees of Freedom: 4999 Total (i.e. Null);  4996 Residual
Null Deviance:     5205
Residual Deviance: 3093  AIC: 3101

[1] "glm summary:"
Call:
glm(formula = formula, family = binomial, data = trainSet)
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.51254  -0.34266  -0.18743  -0.08881   3.26720
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -11.3374     2.0334  -5.576 2.47e-08 ***
freq          4.8294     0.1700  28.415  < 2e-16 ***
robust        6.1516     2.1443   2.869  0.00412 **
mbits         3.2095     0.1965  16.332  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 5205.1  on 4999  degrees of freedom
Residual deviance: 3092.7  on 4996  degrees of freedom
AIC: 3100.7
Number of Fisher Scoring iterations: 6

[1] "glm anova:"
Analysis of Deviance Table
Model: binomial, link: logit
Response: struct
Terms added sequentially (first to last)
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                     4999     5205.1
freq      1   1737.6      4998     3467.4       0.0
robust    1     32.2      4997     3435.2 1.364e-08
mbits     1    342.5      4996     3092.7 1.858e-76

[1] "plot glm"

[1] "termplot"
                2.5 %    97.5 %
(Intercept) -15.323762 -7.351127
freq          4.496200  5.162586
robust        1.947861 10.355392
mbits         2.824279  3.594784

[1] "mikes gof"
Sum of squared errors     Expected value|H0                    SD
           470.28502             494.90148              2.19845
                    Z                       P
           -11.19719              0.00000

[1] "lrm print"
Logistic Regression Model
lrm(formula = formula, data = trainSet, x = TRUE, y = TRUE, maxit = 20)
Frequencies of Responses
   0    1
3925 1075
       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
      5000      1e-07    2112.34          3          0      0.895       0.79
     Gamma      Tau-a         R2      Brier
```

```
     0.793      0.267      0.533      0.094
          Coef    S.E.   Wald Z P
Intercept -11.337 2.0334 -5.58  0.0000
freq        4.829 0.1700 28.41  0.0000
robust      6.152 2.1443  2.87  0.0041
mbits       3.210 0.1965 16.33  0.0000

[1] "lrm anova"
               Wald Statistics          Response: struct
 Factor     Chi-Square d.f. P
 freq       807.36     1    <.0001
 robust       8.23     1    0.0041
 mbits      266.74     1    <.0001
 TOTAL      941.56     3    <.0001

[1] "lrm gof"
Sum of squared errors     Expected value|H0                    SD
            470.28502              494.90148               2.19845
                    Z                      P
            -11.19719               0.00000
```