Abstract

The Database of Macromolecular Motions: A standardized system for analyzing and visualizing macromolecular motions in a database framework

Werner G. Krebs

2001

A convergence of experimental and computer technologies made possible the study of macromolecular motions within a new conceptual, database-based framework. Macromolecular motion is typically the major link between biological structure and function. The number of solved structures of macromolecules that have the same fold and thus exhibit some degree of conformational variability is rapidly increasing. It is consequently advantageous to develop a standardized terminology for describing this variability. I have developed a database of macromolecular motions that classified protein motions into a limited number of categories, first on the basis of size (distinguishing between fragment, domain, and subunit motions) and then on the basis of packing. Furthermore, I have further developed a suite of automated tools, for use in conjunction with the database, for processing protein structures in different conformations. My system attempts to describe a protein motion as a rigid-body rotation of a small 'core' relative to a larger one, using a set of hinges. The motion is placed in a standardized coordinate system so that all statistics between any two motions are directly comparable. I found that while this model can accommodate most protein motions, it cannot accommodate all; the degree to which a motion can be accommodated provides an aid in classifying it. Furthermore, I perform an adiabatic mapping (a restrained interpolation) between every two conformations. This gives some indication of the extent of the energetic barriers that need to be surmounted in the motion, and as a by-product results in a 'morph movie.' I make these movies available over the web to aid in visualization. Users have already submitted hundreds of examples of protein motions to my server, producing a comprehensive set of statistics. I have also found automated means to cull thousands of putative protein motions from the PDB database and analyze them with my automated suite of tools, significantly augmenting my original database. I also describe GNU Queue, a popular, freely available distributed computing software tool that can be used to scale the computational demands of the database as its needs grow. The server is accessible at http://bioinfo.mbb.yale.edu/MolMovDB.

The Database of Macromolecular Motions: A standardized system for analyzing and visualizing macromolecular motions in a database framework

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
In Candidacy for the Degree of
Doctor of Philosophy

by
Werner G. Krebs

Dissertation Director: Mark Gerstein

December 2001

Table of Contents

Table of Illustrations, Figures, and Tables

Acknowledgements

The author gratefully acknowledges the financial support of the NIH Structural Genomics Program, the Keck Foundation and the National Science Foundation (Grant DBI-9723182). Numerous people have also either contributed entries or information to the database and morph server or have given us feedback on what the user community wants. I also wish to thank Informix Software, Inc. for providing a grant of its database software.

I would also like to thank Dr. Yuval Kluger, Dr. Qian Qian, Dr. Vadim Alexandrov, Nathaniel Echols, Cyrus Wilson, Ronald Jansen, Haiyuan Yu, Jochen Junker, and the entire Gerstein group for their assistance with this complex project. I would like to thank the members of my research committee (Prof. Dieter Soll and Prof. Jennifer Doudna), and the entire MB&B faculty for their invaluable advice and research assistance. I would also like to thank a number of outside faculty, including Prof. Cyrus Chothia (for agreeing to serve as outside reader), Prof. V. Ramakrishnan and Dr. James Ogle (30S Ribosome Data), Prof. Joel Sussman (Acetylcholinesterase Data), Prof. Eric Martz (Chime Interface to the Motions Database as well as comments). I would like to thank all of my teachers, past and present, for their support, guidance, and wisdom over the years that has made this present work possible.

I wish especially to thank Prof. Mark Gerstein, my research advisor, for his invaluable research guidance.

Chapter 1: Introduction

Lay of the Road

In my own everyday world, motion is everywhere around us. Distances separate us from each other and the physical objects we rely on for day-to-day existence. Motion is necessary to overcome these distances and to affect work on objects at these macroscopic scales. This introductory chapter will attempt to explain to the general reader how everyday concepts of motion and physical distance relate to my present work in the fields of biological databases and distributed computing, as well as explain how my present work fits into a larger picture of scientific advancements being made at the dawn of the 21st century. First, however, I will briefly describe the contents of the present volume.

Motion plays a key role in some of the most fundamental biological processes[1-3], everything from actin/CAMPK2/Calmodulin synapse tensioning (learning and memory)[4-6], regulation of intracellular metabolites and processes, cell transport, and cell division.[7-11] An understanding of macromolecular motions is therefore of general biophysical interest as well as of potential use in rational drug design. A convergence of innovations in experimental methodology and in the fields of databases, Internet, distributed computing, and artificial intelligence have enabled the study of macromolecular motions within a new conceptual framework. This new database framework, and, indirectly, some of the experimental biology and computer science innovations that made it possible, are the subject of this present work.

I have developed a comprehensive database of macromolecular motions and an as-

sociated suite of software tools that attempts to classify motions on the basis of size and packing. Chapter 2 describes principally the database, while Chapter 3 introduces its key software tools. Chapter 4 introduces additional software tools and implements artificial intelligence techniques for data-mining the database. Chapter 5 is a general Conclusion. Appendix A introduces Partslist, a companion database that includes data from the motions database; the motions database will eventually become integrated into Partslist. Appendix B is yet another published paper on the database, including a section on flexible linkers. Appendix C describes GNU Queue, an innovative, free software program developed by the author that is now in use by thousands of users across the world and is the subject of articles in the technical journals; GNU Queue is scientifically interesting from a computer science standpoint and is ideal for scaling the database's computations across a cluster of computers. Appendix D compares the output of some of the software tools (Chapter 3) with published results and provides advice to users on their advantages, disadvantages, and proper use. Finally, Appendix E is a technical conclusion that summarizes my work.

With the exception of Chapters 1,5 and Appendix D, all sections of the present work are based on materials that have undergone external review; Chapters 2, 3, and Appendix A were previously published in *Nucleic Acids Research*; Appendix B was published as a conference paper; Appendix C is based on materials available off the Internet from a prestigious Internet-standards organization, and Chapters 4 and Appendix E are presently in peer-review.

Motion and Civilization

Throughout much of history, science and the arts have focused principally on devising new means of motion to solve the macroscopic problems with which humanity was preoccupied. To defend land and resources, it was necessary to devise means to rapidly move armies across distances to concentrate force where it was most needed. Roads and bridges were built, leading in turn to an expansion of trade as regions began to specialize in the efficient production of specific goods. Equally important was the erection of physical barriers to prevent enemies from occupying land and resources: city walls, fortifications, and castles sprang up, thus concentrating markets into smaller, more defensible areas, and made cities economic as well as political and transportation centers. As the size of living cells grew, similar problems were overcome in similar ways; the metaphor of the cell as a miniature city is an apt one.

In human history, improvements in transportation were often seen as of vital economic importance: roads, horseshoes, steam and automobile locomotion. Equally important in everyday life were mechanical devices: pumps to irrigate fields; yokes to harness the mechanical energy of domesticated animals; and weapons to hunt prey and kill enemies. Although some chemical compounds were seen as important, discovery of new, important chemicals such as medicinal compounds was chiefly through accident or brute force search rather than rational investigation. Instead, inventors were chiefly concerned with motion, where their intellect could effect a rational improvement: new and improved mechanical devices harnessing new or cheaper materials, cleverer devices, and more sophisticated theories to achieve a motion invoking some desired physical result in a clev-

erer, faster, or more economical motion.

Beginning in the late nineteenth century and throughout the twentieth century, science became acutely aware of the problem of scale. Quantum mechanical and relativistic effects become important at smaller scales, and mechanical engineering, no longer the sole engine of scientific progress, became less important. The design of chemicals gradually became increasingly rational as the early chemical industry invented new products whose principal parts were small molecules. These acted through statistical effects (chance collisions with other molecules) rather than through the carefully designed mechanical effects typical of past inventions. Electronic and optical devices: vacuum tubes, transistors, semiconductor chips, and lasers—required the use of ever more sophisticated computation methods to understand the non-intuitive dynamics of quantum mechanics at these microscopic scales. The computation power of each generation of semiconductor devices designed its successors. In electronics and small molecule chemistry, mechanical motion, as such, became less important as other effects—statistical collisions, classical electromagnetic, and quantum mechanical influences—played a far role in the design of these products.

Eventually, however, improvements in all areas of chemistry, physics, and computer technology enabled an understanding of DNA and protein molecules, and in the last year of the 20th century, a determination of the complete DNA sequence of a single individual. These advances combined towards an understanding of the basic “parts” in living organisms: chiefly proteins read from the organism's master DNA blueprint.

In the microscope world of biological macromolecules, quantum mechanical and electromagnetic effects become subtler as physical scales become smaller. It becomes

increasing possible to affect change at a distance without actually “being there,” and motion's importance becomes far more subtle, much as in small molecule chemistry. Cells and living organisms are a key bridge between the simple physical phenomena of small molecules and the far more complex, ordered world macroscopic world we live in. Cells show complex organization on many different scales, and physical motion becomes increasingly important as we move from the scale of individual atoms (on the scale of angstroms) to the scale of humans and other large animals (meters), a range of ten orders of magnitude. The eucaryotic cell is already large and complex enough to be thought of as a miniature city, with power plants, factories, and waste disposal systems. Motion continues to play an important role in such principle cellular functions as intracellular transport, and, indeed, the macromolecules themselves. The latter concept is the topic of this thesis. Most proteins useful to living organisms are large enough that motion once again begins to become important, as evidenced by references to these biological building blocks as “parts,” and (in the case of huge proteins such as the DNA polymerase or the GroEL chaperone complexes) “huge machines” suggesting that these chemical complexes have become sufficiently large and complex that the physics of their operation is more analogous to macroscopic mechanical devices than the statistical mechanical mechanisms that we associate with smaller molecules. These mechanical functions of proteins play an essential role in almost every facet of life.

The Promise of Motion

To help introduce an undergraduate to protein motions, he was asked to write a brief essay on protein motions, explaining if, how, and why they were important and dis-

cussing some of the literature he had read as part of his assignment. “The importance of motions is profound,” he began. “Protein motion is one of the most researched topics in science today. The promise that it holds is immeasurable.”

He then went on to write, “Since any given organism may have millions of different proteins and those proteins may differ even within a species, the number of proteins that must be resolved are nearly infinite.” In fact, there are thought to be only roughly 100,000 proteins in the human organism[12], and, while there is some variation from individual to individual as well as different isoforms within an individual, it is generally so minor that these differences need not be resolved structurally. Individual variations at the DNA level more commonly lead to changes in expression and subtle changes in the protein's efficiency. Less common are total knock-outs of genes (which often lead to at least hereditary tendencies towards a disease state). Mutations or variations leading to noticeable changes in protein motions are probably least common of all[13]. Thus, while the number of proteins is sufficiently large to require database techniques, it is by no means infinite, and quite amendable to database approaches.

An Historic Opportunity

Yale Prof. Richard P. Lifton points out, “there's a bit of an Oklahoma land-rush feel to the examination of genomic sequence right now. Once all of the genes are identified, that's it for all of history. We're not ever going to have another period of discovery in human biology to match the one that we're in today”[14]. Similarly, there will be but one opportunity in history to decipher the important motions involved in the genome; once they have been resolved, this chapter in human history will be closed, and biology will

move on to conquer the fresh, new challenges of tomorrow. Thus, while protein motions may have promise, the promise of a database of macromolecular motions such as I have constructed here could scarcely be described as immeasurable. It is, however, a unique and timely opportunity that will soon pass away.

To paraphrase Shakespeare[15], "there is a tide in the affairs of databases,/ Which, taken at the flood, leads on to fortune; / Omitted, all the voyage of their life / Is bound in shallows and in miseries. / On such a full sea are we now afloat; / And we must take the current when it serves, / Or lose our ventures."

Developing the database

In the case of the database, Shakespeare's "tide in the affairs of men" was the technological situation in 1996. Rapidly advancing computer, database, and Internet technology and an exponentially growing number of structures in the Brookhaven Protein DataBank (PDB, http://www.pdb.bnl.gov, later to move to the RCSB, http://www.rcsb.org) finally made it possible to study protein motions in detail with an Internet-accessible framework. I found that it was possible to hierarchically classify proteins into a limited number of categories, and that the individual database entries would be of interest in structural biology and rational drug design. The database as a whole could be integrated with other databases (such as the Partslist Database, http://www.partslist.org) for use in gene annotation and drug target elucidation models. I also found that I could mine the resulting database by manual and machine learning techniques to create a comprehensive resource for biologists. The database would contain entries and analyses on nucleic acid motions as well as on protein motions. It would be constructed in such a way as to allow Internet coloration on database entries. Finally, and perhaps most important, it would provide a

suite of software tools (most notably the morph server) to help database users visualize and quantitatively analyze motion entries. I would eventually give the database it's own URL, http://www.molmovdb.org.

Chapter 2: A Database of Macromolecular Motions

Introduction

In this chapter, originally published in *Nucleic Acids Research*[16], I describe a database of macromolecular motions meant to be of general use to the structural community. The database, which is accessible on the World Wide Web with an entry point at http://bioinfo.mbb.yale.edu/MolMovDB, attempts to systematize all instances of protein and nucleic acid movement for which there is at least some structural information. It was developed in collaboration with Prof. Mark Gerstein. At present it contains ~120 motions, most of which are of proteins. Protein motions are further classified hierarchically into a limited number of categories, first on the basis of size (distinguishing between fragment, domain, and subunit motions) and then on the basis of packing. My packing classification divides motions into various categories (shear, hinge, other) depending on whether or not they involve sliding over a continuously maintained and tightly packed interface. In addition, the database provides some indication about the evidence behind each motion (i.e. the type of experimental information or whether the motion is inferred based on structural similarity) and attempts to describe many aspects of a motion in terms of a standardized nomenclature (e.g. the maximum rotation, the residue selection of a fixed core, etc). Currently, I use a standard relational design to implement the database. However, the complexity and heterogeneity of the information kept in the database makes it an ideal application for an object-relational approach, and I am moving it in this

direction. Specifically, in terms of storing complex information, the database contains plausible representations for motion pathways, derived from restrained 3D interpolation between known endpoint conformations. These pathways can be viewed in a variety of movie formats, and the database is associated with a server that can automatically generate these movies from submitted coordinates.

Background

Motions of macromolecules (proteins and nucleic acids) are often the essential link between structure and function; that is, motion is frequently the way a structure actually carries out a particular function. Protein motions[1-3], in particular, are involved in many basic functions such as catalysis, regulation of activity, transport of metabolites, formation of large assemblies and cellular locomotion. Highly mobile proteins have, in fact, been implicated in a number of diseases—e.g., the motion of gp41 in AIDS and that of the prion protein in scrapie[7-11].

Macromolecular motions are also of intrinsic interest because of their fundamental relationship to the principles of protein and nucleic acid structure and stability. They are, however, among the most complicated biological phenomena that can be studied in great quantitative detail, involving concerted changes in thousands of precisely specified atomic coordinates. Fortunately, it is now possible to study these motions in a database framework, by analyzing and systematizing many of the instances of protein structures solved in multiple conformations.

I present here a comprehensive database of macromolecular motions, intended to

be of use to those studying structure-function relationships (e.g. as in rational drug design[17]) and also to those involved in large-scale proteome or genome surveys. There are a number of reasons why it is favorable (and feasible) at present to construct such a database: (i) The amount of raw data (known protein and nucleic acid structures and sequences homologous to them) is rapidly increasing[18-20], and an increasing fraction of new structures have non-trivial motions (see below). (ii) The graphical and interactive nature of a database is particularly well-suited for presenting macromolecular motions, which are often difficult to represent on a static journal page.i (ii) A loose infrastructure of federated databases has emerged in the structural community, allowing the motions database to connect to a variety of information sources[21] (see list in caption to Figure 2.1).

Only one previous attempt has been made at the systematic classification of protein motions[22]. In indirectly related work, a dataset of protein interfaces has also been developed[23].

Overall Organization of the Database

The database exists as a set of coupled hypertext pages and graphic images available over the World Wide Web at http://bioinfo.mbb.yale.edu/MolMovDB .

As shown in Figure 2.1, using the database is simple and straightforward. One may browse either by typing various search keywords into the main page or by navigating through an outline. Either way brings one to the entries. Thus far, the database has ~120 entries, which refer to over 240 structures in the Protein Databank (PDB) (Table

[i] This is particularly true because many published papers about interesting motions do not precisely describe the relationship between the motion and specific publicly accessible coordinate files and viewing orientations. That is, many papers do not tell you that, say, the atomic coordinates for the open form have identifier 6LDH and those for the closed form, 1LDM, and that the motion is best viewed when looking down the crystallographic three-fold after fitting residues 5 to 90.

2.2).

Unique Motion Identifier

Each entry is indexed by a *unique motion identifier*, rather than around individual proteins and nucleic acids. This is because a single macromolecule can have a number of motions and the same essential motion can be shared amongst different macromolecules (see below). (The motion identifier is a short string like "igelbow," which attempts to evoke some characteristic of the motion or protein in the mnemonic style of the Swiss-sProt identifiers[24].)

Attributes of a Motion

In addition to the motion identifier, each entry has the following information:

(i) Classification. A classification number gives the place of a motion in the size and packing classification scheme for motions described below. In addition to its basic classification, a motion can also be annotated as being "similar-to" another motion, as is the case with motions in all the bacterial sugar binding proteins[25,26], or "part-of" or "containing" another motion in the same protein -- e.g. the domain closure in aspartate carbamoyltransferase is clearly part of and driven by a larger allosteric transition, involving the motion of subunits[27,28].

(ii) Structures. Databank identifiers are given for the various conformations of the macromolecule (e.g. open and closed). The identifiers have been made into hypertext links directly to the structure entries in the main protein and nucleic acid databases (PDB and NDB) and to sequence and journal cross-references via the Entrez and MMDB databases[29-33]. Links are also made to related structures via the Structural Classification of

Proteins (SCOP)[34,35]. In the more highly annotated entries, residue selections are given for the main rigid core, for other secondary cores moving rigidly relative to the main core, and for flexible hinge regions linking the cores.

(iii) Literature. Literature references are given. Where possible these are via Medline unique identifiers, allowing a link to be made into the PubMed database[31,32].

(iv) Blurb. Each entry has a paragraph or so of plain text documentation. While this is, in a sense, the least precisely defined field, it is the heart of each entry, describing the motion in intelligible prose and referring to figures, where appropriate.

(v) Standardized Nomenclature. For many entries I describe the overall motion using standardized numeric terminology, such as the maximum displacement (overall and of just backbone atoms) and the degree of rotation around the hinge. These statistics are summarized in Table 2.1. I also attempt to give the transformations (from ii) needed to optimally superimpose and orient each coordinate set to best see the motion (i.e. down screw-axis) and the selections of residues with large changes in torsion angles, packing efficiency, or neighbor contacts.

(vi) Graphics. Each entry has links to graphics and movies describing the motion, often depicting a plausible interpolated pathway (see below).

Hierarchical Classification Scheme based on Size then Packing

Size Classification: Fragment, Domain, Subunit

In the classification scheme currently in use, the most basic division is between proteins and nucleic acids. There are far fewer nucleic-acid motion entries than those of

proteins, reflecting the much larger number of known protein structures.[ii] Currently, the database includes the nucleic-acid motions evident from comparing various conformations of the known structures of catalytic RNAs and tRNAs (specifically, the Hammerhead ribozyme, the P4-P6 domain of the Group II intron, and Asp-tRNA[36,37,38 ,39,40]).

The classification scheme for proteins has a hierarchical layout shown in Figure 2.2. The basic division is based on the size of the motion. Ranked in order of their size, protein movements fall into three categories: the motions of subunits, domains, and fragments smaller than domains.[iii]

Nearly all large proteins are built from domains, and domain motions, such as those observed in hexokinase or citrate synthase[41,42], provide the most common examples of protein flexibility[1-3]. The motion of fragments smaller than domains usually refers to the motion of surface loops, such as the ones in triose phosphate isomerase or lactate dehydrogenase, but it can also refer to the motion of secondary structures, such as of the helices in insulin[43-45]. Often domain and fragment motions involve portions of the protein closing around a binding site, with a bound substrate stabilizing a closed conformation. They, consequently, provide a specific mechanism for induced-fit in protein recognition[46,47]. In enzymes this closure around a binding site has been analyzed in particular detail[13,48-51]. It serves to position important chemical groups around the substrate, shielding it from water and preventing the escape of reaction intermediates.

Subunit motion is distinctly different from fragment or domain motion. It affects

[ii] At the time of writing, the PDB contained in excess of 6600 protein structures, but less than 600 nucleic acids structures.

[iii] There is, of course, also the motion (i.e. rotation) of individual sidechains, often on the protein surface. However, this is on a much smaller scale than the motion of fragments or domains. It also occurs in all proteins. Consequently, sidechain motions are not considered to constitute individual motions in the database, being considered here a kind of background, intrinsic flexibility, common to all proteins.

two large sections of polypeptide that are *not* covalently connected. It is often part of an allosteric transition and tied to regulation[52,53]. For instance, the relative motions of the subunits in the transport protein hemoglobin and the enzyme glycogen phosphorylase change the affinity with which these proteins bind to their primary substrates[54,55].

Packing Classification: Hinge and Shear

For protein motions of domains and smaller units, I have systematized the motions on the basis of packing, using an expanded version of a scheme developed previously[1]. This is because the tight packing of atoms inside of proteins provides a most fundamental constraint on protein structure[56-61]. It is usually impossible for an atom inside a protein to move much without colliding with a neighboring atom, unless there is a cavity or packing defect[62,63].

Internal interfaces between different parts of a protein are packed very tightly[1,64,65]. Furthermore, they are not smooth, but are formed from interdigitating sidechains. Common sense consideration of these aspects of interfaces places strong constraints on how a protein can move and still maintain its close packing. Specifically, maintaining packing throughout a motion implies that the sidechains at the interface must maintain their same relative orientation and pattern of inter-sidechain contacts in both conformations (e.g. open and closed).

These straightforward constraints on the types of motions that are possible at interfaces allow an individual movement within a protein to be described in terms of two basic mechanisms, shear and hinge, depending on whether or not it involves sliding over a continuously maintained interface[1] (Figure 2.2). A complete protein motion (which can contain many of these smaller “movements”) can be built up from these basic mecha-

nisms. For the database, a motion is classified as *shear* if it predominately contains shear movements and as *hinge* if it is predominately composed of hinge movements. More detail on the characteristics of the two types of motion follow.

(i) Shear. The shear mechanism basically describes the special kind of sliding motion a protein must undergo if it wants to maintain a well-packed interface (Figure 2.3). Because of the constraints on interface structure described above, individual shear motions have to be very small. Sidechain torsion angles maintain the same rotamer configuration[66] (with <15° rotation of sidechain torsions); there is no appreciable mainchain deformation; and the whole motion is parallel to the plane of the interface, limited to total translations of ~2 Å and rotations of 15°. Since an individual shear motion is so small, a single one is not sufficient to produce a large overall motion, and a number of shear motions have to be concatenated to give a large effect — in a similar fashion to each plate in a stack of plates sliding slightly to make the whole stack lean considerably. Examples include the Trp repressor and aspartate amino transferase[67,68].

(ii) Hinge. Hinge motions occur when there is *no* continuously maintained interface constraining the motion (Figure 2.4). These motions usually occur in proteins that have two domains (or fragments) connected by linkers (i.e. hinges) that are relatively unconstrained by packing. A few large torsion angle changes in the hinges are sufficient to produce almost the whole motion. The rest of the protein rotates essentially as a rigid body, with the axis of the overall rotation passing through the hinges. The overall motion is always perpendicular to the plane of the interface (so the interface exists in one conformation but not in the other, as in the closing and opening of a book) and is identical to the local motion at the hinge. Examples include lactoferrin and tomato bushy stunt virus

(TBSV)[69,70].

Gerstein et al.[64,71] analyzed the hinged domain and loop motion in specific proteins (lactate dehydrogenase, adenylate kinase, lactoferrin). These studies emphasized how critical the packing at the base of a protein hinge is (in the same sense that the "packing" at the base of an everyday door hinge determines whether or not the door can close). Protein hinges are special regions of mainchain in the sense that they are exposed and have few packing constraints on them and are thus free to sharply kink (Figure 2.4). Most mainchain atoms, in contrast, are usually buried beneath layers of other atoms (usually sidechain atoms), precluding large torsion angle changes and hinge motions.

It is important to emphasize that most shear motions do, in fact, contain hinges (joining the various sliding parts) and that the existence of a hinge is not the salient difference between the two basic mechanisms -- rather it is the existence of a continuously maintained interface.

Other Classification

Most of the fragment and domain motions in the database fall within the hinge-shear classification. However, there are a number of exceptions, and I have created some special categories to deal with them.

(i) A special mechanism that is clearly neither hinge nor shear accounts for the motion. An example of this sort of motion is what occurs in the immunoglobulin ball-and-socket joint[72], where the motion involves sliding over a continuously maintained interface (like a shear motion) but because the interface is smooth and not interdigitating the motion can be large (like a hinge).

(ii) Motion involves a partial refolding of the protein. This usually results in dramatic

changes in the overall structure. Examples where both endpoints are known include the motion in the serpins and influenza virus haemagglutinin[73,74]. Also, included in this category are order-to-disorder transitions (as when a DNA recognition domain becomes ordered upon binding DNA), protein domains that only become structured upon oligomerization (e.g. leucine zipper dimerization domain), and pro-enzymes that dramatically change shape upon cleavage.

(iii) Motion cannot yet be classified. An example of this is the beta-sheet deformations in the TATA-box binding protein[75,76].

For the motions of subunits a different division is made (other than hinge or shear):

(i) Allosteric. Examples include hemoglobin and aspartate carbamoyltransferase[27,28,54].

(ii) Non-allosteric. Examples include the quaternary structure change in the BamHI endonuclease upon binding DNA[77].

(iii) Complex motions. Large protein motions which involve many subsidiary "sub-motions" (which in themselves can be classified as subunit or domain motions) are put into the category of complex motions. The lac repressor, which contains three distinct motions, provides a good example of this situation[78,79]. The first motion is an order-to-disorder transition that the headpiece domain undergoes when it binds DNA. A second motion involves a molecule binding between two other domains in the protein. This motion is essentially the same as the motion observed in another group of proteins, the bacterial periplasmic binding proteins[26]. However, it is coupled to a further subunit rearrangement that changes the overall DNA binding affinity of the protein and consequently is termed

an allosteric transition. Finally, a third motion involves another subunit motion (which is not linked to the allosteric transition) that allows the four reading head domains to bind sites on DNA with different spacing and curvature.

A breakdown of the categorization of entries in the current database is given in Table 2.2. At the time of this writing (version 1.7), the database describes 121 macromolecular motions which reference 241 PDB structures. The hinge mechanism is the most common classification in the database, accounting for 45% of the entries. Over 60% of the motions in the database are classified as domain motions. Interestingly, a greater percentage of fragment motions have structures for multiple conformations in the motion, probably reflecting the greater ease with which these smaller motions can be studied experimentally.

Annotation of Evidence related to the Motion

For each entry in the database, I have tried to indicate the evidence behind its description and classification: i.e. is it based on careful manual analysis of two conformations, automatic output of a conformation comparison program, inferred based on structure comparison, or inferred based on sequence comparison? Thus, a clear distinction is made in the database between the carefully documented, “gold-standard” motion in lactoferrin (i.e. as shown in Figure 2.4) and the much more tentatively understood motion in a protein that is a sequence homologue of another protein which is structurally similar to lactoferrin. I hope that this attention to the evidence behind the motion in the annotation will allow the database to grow rapidly and semi-automatically, without becoming cor-

rupted with false assertions.[iv]

Experimental information on macromolecular movements comes from a number of sources: X-ray structures of particular proteins and nucleic acids in different conformational states (typically "open" and "closed," but other configurations occur, e.g. in allostery and order-disorder transitions), NMR studies (e.g. Pf1 coat protein[80]), and time-resolved studies (e.g. ras, PYP, bacteriorhodopsin[81-83]). Some 95% of entries in the database have been studied by traditional x-ray crystallography, and 8% by NMR (Table 2.3). A smaller number have been investigated by other techniques, such as time-resolved crystallography.

Thus far, the discussion has focused only on "well-documented" motions, where high-resolution structures of at least two conformations (i.e. open and closed) are known. However, there is also the situation where one knows a single conformation of a given protein (A) is similar in structure to another protein (B) and that protein B has a well-documented motion. In this case, one can reasonably infer that protein A has a similar motion to that in protein B. Inferred motions are principally added to the database by finding sequence or structure homologues of a protein or nucleic acid already in the database. The inference is currently expressed as the top level in the preliminary classification scheme (Figure 2.2). For instance, heat-shock protein 70 is classified as having a "suspected shear motion" because of its structural similarity to hexokinase, which has a well-documented shear motion[84,85]. Furthermore, the motions initially suspected in actin and

[iv] It is worth noting that this approach to evidence is not always taken in the annotation of the sequence databanks and it now leading to problems with the advent of large-scale genome sequencing. For instance, the following often arises: A scientist biochemically and structurally characterizes a particular motif, say a zinc finger, in one protein (protein A). This is added to the database and annotated as a zinc finger. A second investigator sequences another protein (B), does a databank similarity search and finds this protein is similar to protein A. Based on this, protein B is annotated in the database as a zinc finger. Now a third investigator sequences protein C. This is found similar to B and is, consequently, thought to be a zinc finger. Clearly, the chain of evidence is getting much weaker.

phosphoglycerate kinase based on analogy to other proteins (i.e. hexokinase) have been subsequently verified by crystallography[1,86-88].

Motions can also be inferred based on a single known conformation and evidence based on requirements for the macromolecule's function, careful calculations, or small-angle scattering experiments. Examples include the motions in myosin[89], plasminogen[90], and acetylcholinesterase[91]. In total, about 78% of the motions have solved structures available for two or more conformations; for the remaining 22% the motions are inferred.

Computer Implementation as a Relational Database

Standard tools and approaches are currently used in the implementation of the database. A free relational database server engine, called mini-SQL[92], has been used with a schema that contains ~10 tables. Data entry has been done through a variety of methods: a web form, Microsoft Access and Excel (using ODBC connectivity or the dbf2msql program), or via the emacs text editor[93] (using a custom "mode" written in elisp). Initially, the web pages were generated "on the fly" in response to a query but then it was decided to pre-build most of them. This proved to be an unexpectedly good move as it allowed on-line search engines to automatically build indices up (e.g. AltaVista), enabling the database to be easily queried from outside. Because it is built using very standard tools, the database has been easily ported into a variety of programs (e.g. Oracle) and into a variety of PC mail-merge programs (for nicely formatted output). Although I plan to maintain pre-built pages in the future, I am investigating the use of high-speed web-database connectivity software (such as Informix's Web datablade) to allow instantaneous updates to the database's Web presence yet maintain a level of performance comparable to static pages.

In total, the database presently contains many disparate types of information: standardized annotation values, literature references, large blocks of free-text, three-dimensional structures, and motion pathways. This presents a particular challenge in terms of integrating the information in a comprehensible format. At present, many of the elements (e.g. movies) are stored outside of the central database (and accessed via stored pointers) or in the actual tables as large binary objects ("BLOBS"). I am presently migrating the database to an object-relational system made by Informix, a commercial product that traces its roots to the postgres database project at Berkeley[94-96]. The object-relational database model supports the referencing of complex datatypes in relational tables and sophisticated querying of these complex types through user-defined functions. There are also plans to develop a data-definition language for the database around mmCIF[97].

Representing Motion Pathways as "Morph Movies"

One of the most interesting of the complex data types kept in the database are "morph movies" giving a plausible representation for the pathway of the motion. These movies can immediately give the viewer an idea of whether the motion is a rigid-body displacement or involves significant internal deformations (as in tomato bushy stunt virus versus citrate synthase). Pathway movies were pioneered by Vorhein et al.[98], who used them to connect the many solved conformations of adenylate kinase.

Normal molecular-dynamics simulations (without special techniques, such as high temperature simulation or Brownian dynamics[99-101]) can not approach the timescales of the large-scale motions in the database. Consequently a pathway movie cannot be generated directly via molecular simulation. Rather, it is constructed as an interpolation be-

tween known endpoints (usually two crystal structures). The interpolation can be done in a number of ways.

(i) Straight Cartesian interpolation. The difference in each atomic coordinate (between the known endpoint structures) is simply divided into a number of evenly spaced steps, and intermediate structures are generated for each step. This was the method used by Vorhein et al. It is easy to do, only requiring that the beginning and ending structures be intelligently positioned by fitting on a motionless core. However, it produces intermediates with clearly distorted geometry.

(ii) Interpolation with restraints. This is the above method where each intermediate structure is restrained to have correct stereochemistry and/or valid packing. One simple approach is to energy minimize each intermediate (with only selected energy terms) using a molecular mechanics program, such X-PLOR[102].. The database, furthermore, is home to a server that applies this interpolation technique to two arbitrary structures, generating a movie. This server[103] is described more fully in Chapter 3.

Conclusion and Future Directions

I have constructed a database of macromolecular motions, which currently documents ~120 motions. To describe each motion I have developed a classification scheme based on size then packing (whether or not there is motion across a well-packed interface) and a way of annotating and classifying inferred motions. I also developed a standardized nomenclature, such as maximum atomic displacement or degrees of rotation. At present, I am only using standardized values culled from the literature. However, many of

these values can be computed automatically with software tools I am developing, allowing this process to be automated.

I anticipate that the database will constitute an important resource for the molecular biology community. In fact, I expect that the number of macromolecular motions will greatly increase in the future, making a database of motions somewhat increasingly valuable. My reasoning behind this conjecture is as follows: The number of new structures continues to go up at a rapid rate (nearly exponential). However, the increase in the number of folds is much slower and is expected to level off much more in the future as we find more and more of the limited number of folds in nature, estimated to be as low as 1000[18,104]. Each new structure solved that has the same fold as one in the database represents a potential new motion -- i.e. it is often a structure in different liganded state or a structurally perturbed homologue. Thus, as we find more and more of the finite number of folds, crystallography and NMR will increasingly provide information about the variability and mobility of a given fold, rather than identify new folding patterns.

Table 2.1: Standard Statistics for the Magnitude of the Motions

Value	Num. Entries	min	max	average
Maximum Cα displacement	11	1.5	60	12
Maximum Atomic Displacement	3	8.8	10	9.3
Maximum Rotation	12	5	148	24
Maximum Translation	2	0.7	2.7	1.7

The motions in the database range greatly in size, with maximum mainchain displacements between 1.5 and 60 Å. All the statistics are for version 1.7 of the database, based on the relatively small set of values culled from the literature. The averages are only approximate given the sparse nature of the data. I am developing software tools to extract these values automatically from structural data.

Table 2.2: Statistics for the Mechanism of the Motions

Size / Mechanism	Domain		Fragment		Subunit		Complex		Total	
Hinge	38	51%	16	59%					54	44%
Shear	14	19%	3	11%					17	14%
Partial Refolding	5	7%							5	4%
Allosteric					8	57%			8	7%
Other/Non-Allosteric	2	3%	1	4%	6	43%			9	7%
Unclassifiable	15	20%	7	26%			3	50%	25	20%
Notably Motionless									1	1%
Nucleic Acid							3	50%	3	2%
Known** / %category	53	72%	25	93%	11	79%	5	83%	94	77%
Suspected / %category	21	28%	2	7%	3	21%	1	17%	28	23%
Totals / %DB	74	61%	27	22%	14	11%	6	5%	122	100%

This table cross tabulates the two main classifying attributes of motions: their size (row heads) and their packing characteristics (column heads). I define a known motion (**) to be a motion with two or more solved conformations, and a suspected motion is defined to have only one or fewer solved conformations.

Table 2.3: Statistics for the Evidence about Motions

Experimental Technique	Entries studied by this technique	Fraction of database
All Techniques	122	100%
Traditional X-ray crystallography	116	95%
NMR	9	7%
Molecular Dynamics Simulations	4	3%
Time-resolved crystallography	3	2%
Circular Dichroism (CD)	2	2%
Fourier Transform Infrared Spectroscopy (FTIR)	1	<1%
Molecular Biology Studies of Motion	1	<1%

-

This table summarizes the number of motions studied by the various experimental techniques. I indicate the evidence behind a motion through listing information about the ex-

perimental techniques used, telling whether or not the motion is inferred, and giving a standardized "annotation level." I also timestamp all entries with creation and modification dates and associate the web presentation of the database with a clear version numbering scheme. Note percentages in this table do not add up to 100% as a motion can be studied by more than one technique.

Figure 2.1: The Motions Database on the Web

LEFT shows the World Wide Web "home page" of the database. One can type keywords into the small box at the top to retrieve entries. RIGHT shows an entry retrieved by such a keyword search (the entry for calmodulin). Graphics and movies are accessed by clicking on an entry page. (These have been deliberately segregated from the textual parts of the database since the interface was designed to make it easy to use on a low-bandwidth, text-only browser, e.g. lynx or the original www_3.0). An example of a segregated graphic for calmodulin is the movie shown in Figure 2.5. The main URL for the database is http://bioinfo.mbb.yale.edu/MolMovDB. Beneath this are pages listing all the current movies, graphics illustrating the use of VRML to represent endpoints, and an automated submission form to add entries to the database. The database has direct links to the PDB for current entries (http://www.pdb.bnl.gov); the obsolete database for out-of-date entries (http://pdbobs.sdsc.edu); scop for structure classification (http://scop.mrc-lmb.cam.ac.uk); Entrez/PubMed for literature citations (http://www.ncbi.nlm.nih.gov/PubMed); LPFC for core structures, (Library of Protein Family Core Structures, http://smi-web.stanford.edu/projects/helix/LPFC); and GeneCensus for information related to structural genomics (http://bioinfo.mbb.yale.edu/census)[30,105-107]. Through these links one can easily connect to other common protein databases such Swiss-Prot, Pro-Site, CATH, RiboWeb, and FSSP[24,108-112]. For all these links, PDB identifiers or PubMed unique IDs are used as for-

eign keys. External databases may also link to entries in the motions database by using PDB identifiers as foreign keys. In particular, the interface to the database is via the following URL convention: http://bioinfo.mbb.yale.edu/MolMovDB/search.cgi?pdb=*1abc*, where *1abc* is a PDB structure identifier referenced in the movements database. Further, information on the database's public interface and on linking external resources to it may be obtained by at http://bioinfo.mbb.yale.edu/MolMovDB/linkhelp.txt. I am developing transaction-processing features that allow authorized remote experts to serve as database editors and anticipate that these will become an important part of the interface in the future. (This figure as well as Figures 2.2, 2.3, 2.4, and 2.5 are adapted directly from the web presentation of the database, which is copyright, Gerstein & Krebs, 1998).

Fig. 2.1: The Motions Database on the Web

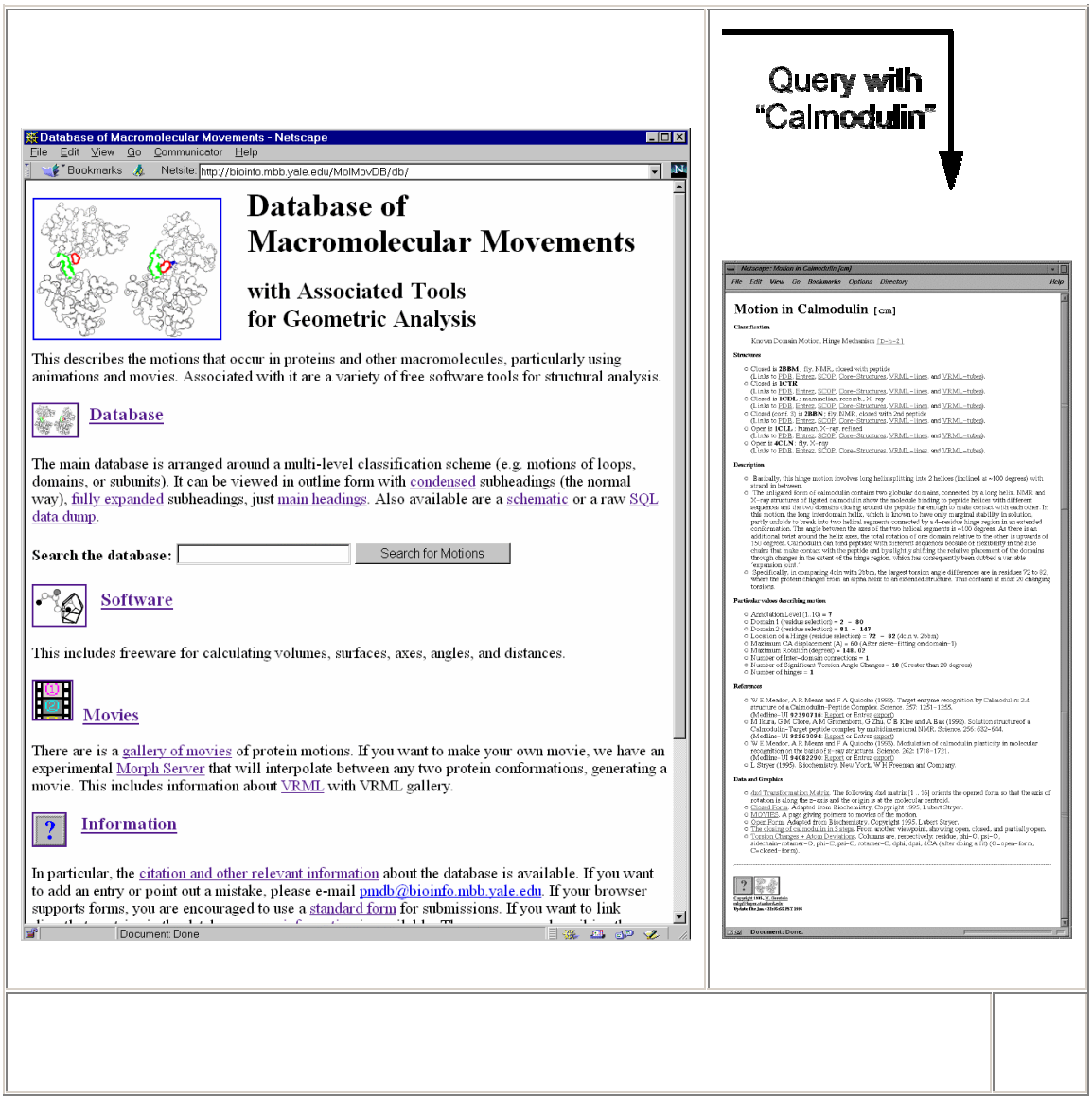


Figure 2.2: Schematic Showing the Overall Classification Scheme for Motions

LEFT, the database is organized around a hierarchical classification scheme, based on size (fragment, domain, subunit) and then packing (hinge or shear). Currently, the hierarchy also contains a third level for whether or not the motion is inferred. RIGHT is a schematic showing the difference between shear (sliding) and hinge motions. This figure adapted from the database and Gerstein et al.[1,64]. It is important to realize that the hinge-shear classification in the database is only "predominate" so that a motion classified as shear can contain a newly formed interface and one classified as hinge can have a preserved interface across which there is motion. The essential characteristics of the various motions are summarized below. To annotate a macromolecule's classification succinctly a three-letter short-hand code is used. It designates the major classification (Fragment, Domain, Subunit, Complex, or Nucleic acid), sub-classification (hinge, shear, allosteric, non-allosteric, RNA, or DNA), and whether or not the motion has been solved structurally in at least two conformations. For example, ‘D-h-2’ would indicate a domain hinge motion with at least two conformations solved.

Fig. 2.2 Schematic Showing the Overall Classification Scheme for Motions

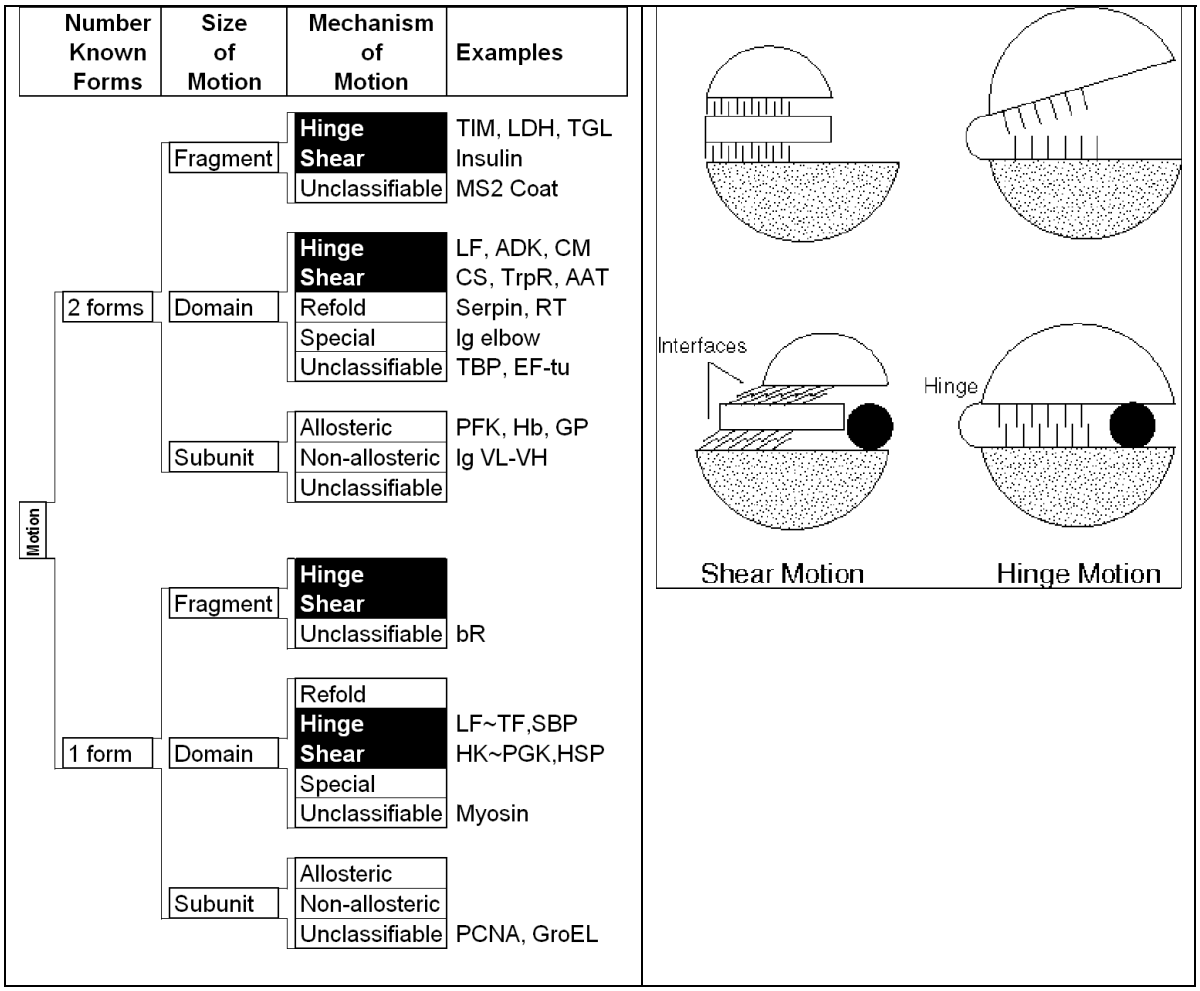
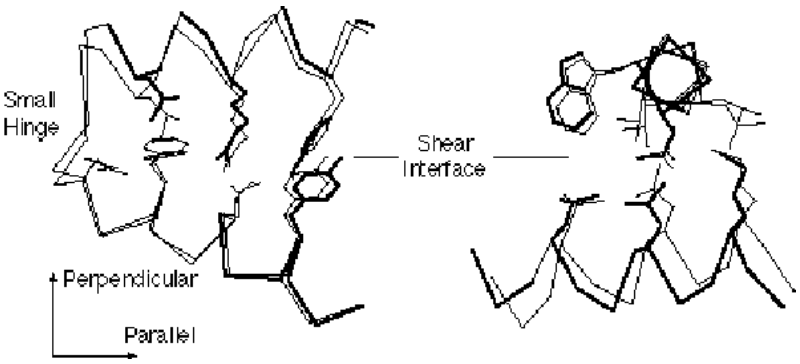


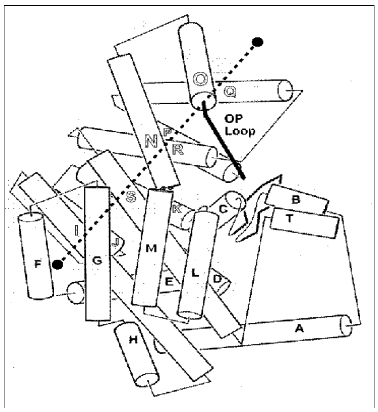
Figure 2.3: Close-up on the Shear Mechanism

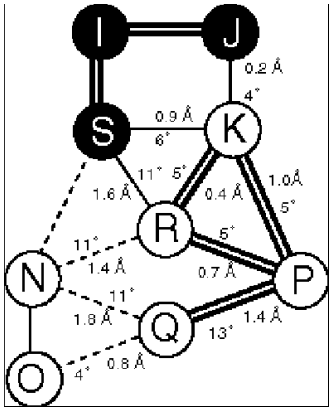
The figure gives a close up illustrating shear motion in one protein, citrate synthase[1,84]. TOP-LEFT and TOP-RIGHT show representative shear motions between close-packed helices. Note how the mainchain only shifts by a small amount and the sidechains stay in the same rotamer configuration. MIDDLE-LEFT, Cartoon of one subunit of citrate synthase (1CTS), gives an overall view of the protein showing that it is composed of many helices. The adjacent subunit is related by two-fold axis shown. (The small two-stranded sheet is omitted to improve clarity.) α-helices are represented by cylinders. The small domain contains helices N, O, P, Q, and R. The mobile OP helix is highlighted. MIDDLE-RIGHT gives details on the mobile interfaces. The orientation is perpendicular to the twofold axis. The particular section is indicated by the dotted line on the MIDDLE-LEFT subfigure. Selected helixes from both subunits are shown. (Upper-case letters are for one subunit and lower-case letters are for the other one.) The helices shown with white lettering on a black background are motionless, while those shown in black on white move appreciably. Edges indicate the existence of helix-helix packing in both the open and closed form. Double edges are nearly parallel packing (0-30°); single edges, intermediate packing (30°-60°); and dotted edges, crossed packing (60°-90° and on-end packing). There is no packing between helixes L and N because helixes L, M, G, and F are much higher (coming out of page) than O, N, Q, P, R, and K. S and I are long and make contacts with both sets. Note in the diagram how the dimer neatly divides into six

layers with the active site, indicated by a star, at the intersection between layers. This is representative of how proteins undergoing shear motions can be divided into layers. Part of one subunit is enlarged at the bottom of the diagram and shows the relative movements of the principal helices in citrate synthase. The shifts (in Angstroms) and rotations (in degrees) show local changes in the positions of pairs of packed helices (i.e. the movement in one helix in a pair relative to the other). Clearly, larger relative movements tend to be associated with more crossed helix-helix packing. BOTTOM shows how these small motions can be added together to produce a large overall motion. Specifically, many small motions add up to shift helix O by 10.1 Å and rotate it by 28°. The incremental motion in shear domain closure is shown by Cα traces of the whole protein and of a close-up of the OP loop. BLACK is the apo form; WHITE, holo form; GRAY, cumulative effect of motion over the K, P, and then Q helix-helix interfaces. (The apo form was fit to the holo form, first on the core, and then on the K, P, and Q helices.)

Fig. 2.3 Closeup on the Shear Mechanism







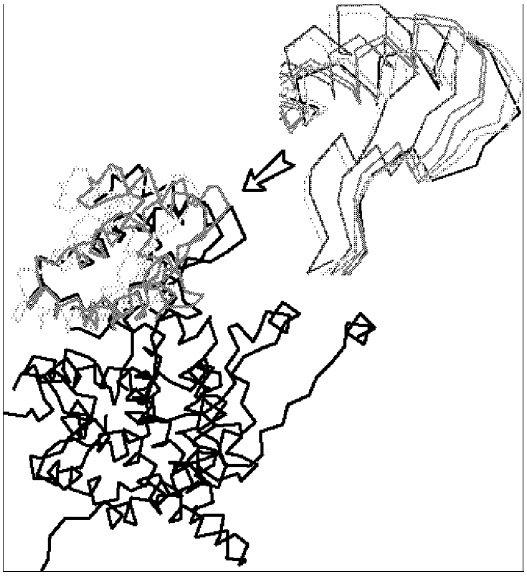


Figure 2.4: Close-up on the Hinge Mechanism

The figure shows the hinge motion in lactoferrin[1,64]. FAR-LEFT shows a ribbon drawing of the protein in the open conformation. The view is down the screw-axis, which is indicated in the figure by the circle with the dot in it. The screw-axis passes very close to the hinge region, which occurs in the middle of two beta strands (highlighted in bold). MIDDLE-LEFT and MIDDLE-RIGHT show the open and closed conformations in terms of space filling slices. A thick black line highlights the hinge region. Note how few packing constraints there are on the hinge in contrast to the other atoms in the protein. FAR-RIGHT shows a close-up of the hinge region. (The numbered residues correspond to the open circles in the ribbon drawing.) (Figure adapted from the database and Gerstein et al.[64]).

Fig. 2.4 Closeup on the Hinge Mechanism

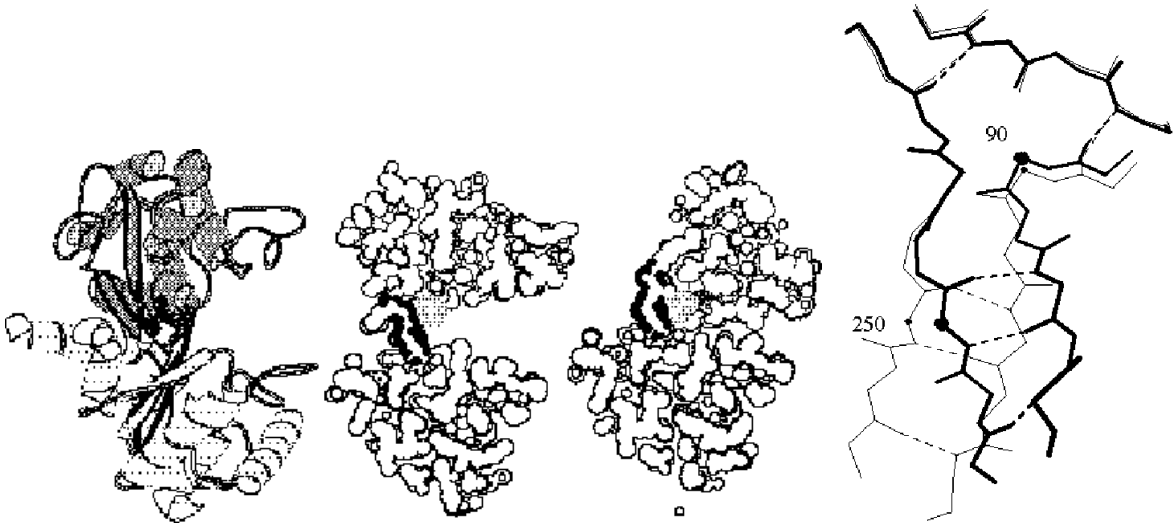


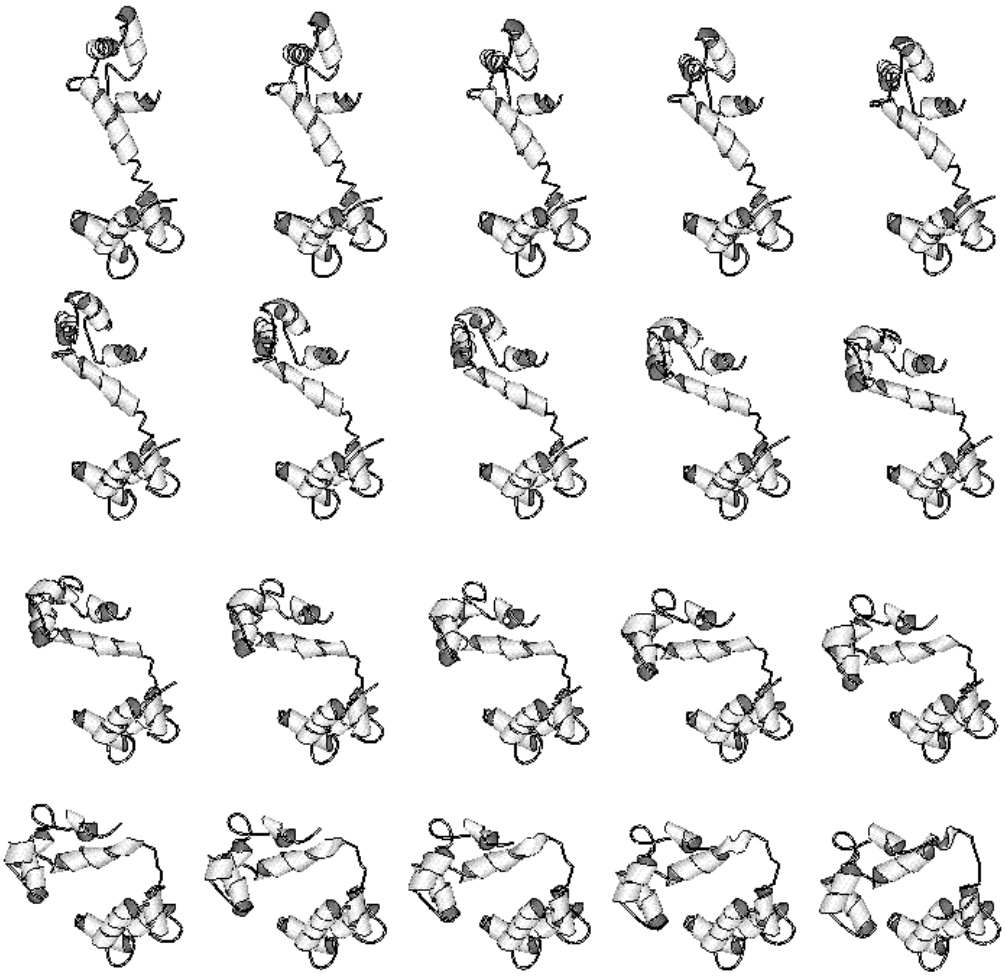
Figure 2.5: Interpolated Motion Pathways

A preliminary pathway of the hinge motion in the protein calmodulin is shown[113]. This was constructed by a variant of the second method, involving Cartesian interpolation with minimization of the intermediate structures using both stereochemical and packing terms. This and more than 30 other movies are available at http://bioinfo.mbb.yale.edu/MolMovDB/movie . For the actual generation of representations, currently one orientation is chosen (i.e. down the screw-axis) and then the animated intermediates are drawn in a variety of 2D-movie formats (MPEG, QuickTime, SGI movie format, MultiGIF, and so on). Preliminary 3D animation has been implemented using the new VRML-2 specification[114]; however, I have encountered some compatibility problems due to the great state of flux that VRML 2.0 browser software presently is in.

Calmodulin, which is shown in Figure 2.1 as well as in this figure, is one of the more highly annotated motions in the database. It provides a good example of how the overall annotation process works. A motion is initially brought to the database curator's attention either directly by researchers solving particular structures or indirectly by surveying the literature. Once the database curator (currently me) decides to add it to the database, he does a comprehensive literature search, usually via Medline, and retrieves from the original publications statistics associated with the motion. It is in itself quite a complex nomenclature problem to reconcile the many different terms used to describe motion and create truly standardized statistics (such as a well-defined maximum atomic displacement

or precise selections for hinge residues). This is one aspect of the larger problem of nomenclature that is becoming increasing important in bioinformatics[115]. Next, I fetch coordinate sets from the PDB and run various comparison programs on these structures (e.g. to calculate torsion angle differences, do least-squares fits, evaluate packing, etc.). Part of the process of conformation comparison is the generation of a "morph movie," such as the one shown in the figure. My server (Chapter 3) can produce a morph completely automatically. Typically, two structures are selected as being representative of the endpoints of the motion. Intermediate conformations are generated from these endpoints by linear interpolation with restraints applied at each interpolated time point to ensure realism. (For the case of calmodulin, bond length and angle restraints were applied.) The interpolated coordinates are joined into an animation through using any of a number of widespread molecular rendering software packages (e.g. Molscript or Rasmol[116,117]. Morphing and automatic conformation comparison generates a second, more standardized set of statistics, which can be compared against those culled from the literature. Finally, based on running programs and reading the literature, I decide on the motion classification and write the entry. Presently, much of this process is done manually but I hope to automate large amounts of it in the future. The automatic classification tool developed by Boutonnet et al.[22] may be useful in this regard. Because my database schema is flexible, it can readily accommodate different types of automatic and manual annotation.

Fig. 2.5 Interpolated Motion Pathways



Chapter 3: The Morph Server: A standardized system for analyzing and visualizing macromolecular motions in a database framework

Introduction

The number of solved structures of macromolecules that have the same fold and thus exhibit some degree of conformational variability is rapidly increasing. It is consequently advantageous to develop a standardized terminology for describing this variability and automated systems for processing protein structures in different conformations. In this chapter, originally published in *Nucleic Acids Research*[103], I describe how I have developed (in collaboration with Prof. Mark Gerstein) such a system as a "front-end" server to my database of macromolecular motions. My system attempts to describe a protein motion as a rigid-body rotation of a small "core" relative to a larger one, using a set of hinges. The motion is placed in a standardized coordinate system so that all statistics between any two motions are directly comparable. I find that while this model can accommodate most protein motions, it cannot accommodate all; the degree to which a motion can be accommodated provides an aid in classifying it. Furthermore, I perform an adiabatic mapping (a restrained interpolation) between every two conformations. This gives some indication of the extent of the energetic barriers that need to be surmounted in the motion, and as a by-product results in a "morph movie." I make these movies available over the web to aid in visualization. Many instances of conformational variability occur

between proteins with somewhat different sequences. I can accommodate these differences in a rough fashion, generating an "evolutionary morph." Users have already submitted hundreds of examples of protein motions to my server, producing a comprehensive set of statistics. So far the statistics show that the median submitted motion has a rotation of ~10º and a maximum Cα displacement of 17 Å. Almost all involve at least one large torsion angle change of >140º. The server is accessible at http://bioinfo.mbb.yale.edu/MolMovDB.

Background

Solved structures and related structural information on proteins is growing at an exponential rate. This is due chiefly to continuous technological progress in X-ray crystallography, NMR spectroscopy, and computer technology. As researchers solve structures at an ever-increasing rate, there occurs an obvious need for processing techniques to relate such structures to one another, beyond classification or structural alignment. Protein motions, as an essential link between structure and function, are an obvious area of relationships between protein structures in the databases. Motion is intimately related to the way a structure fulfills a particular function. Protein motions Protein motions[1-3] are involved in a wide variety of basic functions, including regulation, transport of metabolites, formation of large assemblies, and cellular locomotion. Examples can be found throughout nature, from local conformational changes involved in the binding of ligands[118] that occur in enzymatic reactions to the complex rearrangement of covalent bonds[119].

Obviously, one of the best ways to represent and communicate protein motions is through "movies," especially when they are made available over the web. There have been a number of previous efforts in this area. Vonrhein *et al*.[98,120] made a custom movie of calmodulin and placed it on the Web. Similar work has been done by Sawaya *et al.*, who created movies of crystal structures of polymerase beta[121]. Ray-traced 3-D molecular dynamics simulation of acetylcholinesterase from multagenesis data have also been made available[91,122,123]. More recently, movies from molecular dynamics simulations of protein folding (plp group)[124,125] have been available on the Internet. Xu *et al* used the techniques of normal mode analysis to produce a morph movie of GroEL from structural data.[126-129]

In this chapter I present a perspective on how protein motions can be put into standardized, consistent terms. I develop a simple model for protein motions involving rigid-body motion of parts, apply my model to actual cases, and measure how well it fits. My approach is embodied in an integrated Web server that provides tools to compare solved conformations of proteins involved in motion, generates statistics to characterize and classify them into a database, and automatically makes a morph movie to represent them. In addition, the server presents a database linking protein motions with custom movies of motions available at other sites, along with my own morphs generated automatically by the server upon request by members of the Internet community. My server and database have been used by Internet users to analyze a number of recent structures including human interleukin 5[130], bc1 complex[131,132], glycerol kinase[133,134], and lactoferrin[135,136]. It has also been used as a source of raw data in visualization tools[137] and in relation to other biological databases[138]. The Web server is accessible at: http://bioinfo.mbb.yale.edu/MolMovDB/morph. It is integrated with the Database of

Macromolecular Motions[139,140] and is also connected with a variety of tools for aligning protein folds and studying their occurrence in genomes[141-144]. Appendix D in the present volume compares the output of the standardized system with published results for several proteins and provides users with advice on its advantages, disadvantages, and proper use.

Information Flow

The best way to understand my approach is in terms of the "information flow" diagrammed in Figure 3.1. One starts by submitting two or more conformations[v] of a given protein to the server. Then, through a variety of transformations, the server classifies the motion in the database and produces an appealing movie.

Data sources

Solved conformations analysis as performed by the server's tools requires two kinds of information: (1) three-dimensional atomic coordinates of protein conformations as solved structure files (such as those at the PDB) and, more importantly, (2) information relating two or more of these solved structures, thus selecting them for analysis. (Such information, for instance, could come from the SCOP Database[34,35], from automated searching of databases for proteins related by structure or sequence, or from a simple user input form on the Web.) A selection scheme is important because the number of ordered pairs of PDB structures is rather large (more than 10000^2). Figure 3.2 diagrams the server in the larger context of data sources.

Alignment

Once a string of structures has been given to the server, the first step is to establish

[v] Given one conformation, a number of on-line tools and databases, such as the PDB, FSSP, SCOP, CATH, CE, and VAST can suggest a second conformation. I am currently investigating this.

equivalence (an alignment) between residues in the various proteins. This is necessary because the protein structures compared, while sharing some evolutionary or structural similarity, will, in general, not share the same amino acid sequence. Consequently, an alignment is necessary.

Because the server may be asked to simultaneously compare more than two sequences, an algorithm capable of simultaneously aligning multiple sequences (or structures) and potentially building an evolutionary tree must be used. For this purpose, I have chosen the AMPS algorithm[145-147]. In cases in which sequence alignment is inappropriate, such as for highly diverged homologs, I use the technique of structure alignment[143,144]. The latter method relies primarily on the use of 3D coordinates (i.e., solved PDB structures of proteins) to produce a sequence alignment otherwise analogous to an alignment produced purely from sequence information. As a result, the structural method is able to generate meaningful sequence alignments from both highly related proteins and completely unrelated proteins sharing similar structural features due to convergent evolution. Sequence alignment is used unless sequence similarity is below a user-defined cutoff, at which point structure alignment is used. The choice of approach (sequence or structural alignment) may also be forced by the user upon morph submission.

Superposition

One of the major aims of the server is to collect standardized statistics on the proteins involved in motions. Standardized statistics, such as maximum rotation or maximum Cα displacement, are computed with respect to a specific superposition and reference frame, and so the superposition algorithm is central to any conformational analysis tool.

The output of the alignment procedure establishes residue equivalencies that are used in an intelligent superposition of the structures onto one another. Traditional "all-atom" RMS superposition minimizes the RMS difference between Cα atoms in the open and closed conformations. In a simple hinge motion, e.g., Calmodulin, such an alignment fits the closed conformation symmetrically inside the open conformation (Figure 3.3). Amongst other things, the maximum Cα displacement computed from such a superposition is considerably underestimated from the common sense alignment, and the morph movie gives the impression of motion far more complicated than a simple opening of a hinge. Instead, I perform the superposition with a modified "sieve-fit" procedure[71,148]. The procedure is iterative. On each iteration the remaining Cα atoms are superimposed by a standard RMS fit, and then the pair of corresponding Cα atoms furthest apart are eliminated. This is repeated until approximately half of the atoms in the protein have been eliminated. Previously described uses of the "sieve-fit" procedure[71,149] used some sort of cut-off value to determine when to stop the procedure, typically RMS deviation. No single RMS deviation cut-off value has consistently worked well. However, I have found that by stopping the procedure after approximately half the atoms have been discarded, one of the "domains" thus selected generally corresponds approximately to a superset or a subset of a real domain in the structure, and is thus well suited for performing the subsequent axes transformations.

Orientation & Hinge Location

To locate the screw-axis, a "fit-refit" procedure, as described by Lesk & Chothia[84] is used. Following superposition of the starting and ending conformations, I only consider the set of eliminated atoms. I perform a RMS-fit of that set between the starting and end-

ing conformations; the server performs the new superposition (arbitrarily) on the ending conformation. A comparison of the new position of the ending conformation following this latest fit with its position following the "sieve-fit" procedure yields a geometric transformation whose screw axis is (approximately) the axis of the hinge motion, i.e., the location of the hinge, as has been published elsewhere[150]. Straightforward calculations allow characterization of the angle of rotation around the hinge axis.

If a significant hinge motion is present, the software uses these transformations to align the Z-axis of the coordinate frame parallel to the hinge axis so that, when the motion is rendered, viewers will look down the screw-axis of the hinge motion. The longest moment of the protein (long axis) is rotated (optionally) so that it is parallel to the Y-axis. Finally, the coordinate frame is translated so that the centroid of the initial conformation is in the center of the field of view.

The software also attempts to locate putative hinge regions using a simple and relatively fast algorithm. The algorithm looks for a persistent transition between the two domains identified by the program. The algorithm constructs a search window, initially with 24 residues. It examines each position along the peptide backbone in this window. If there is a persistent transitions (i.e., one-half of the algorithm's search window belonging to one "domain" and the other half to the other), a hinge is detected. If the program fails to find any hinges along the backbone chain, the window size is reduced by two, and the procedure is repeated until the window size has been shrunk down to twelve residues, at which point the program reports failure. Empirically, this crude but computationally inexpensive algorithm successfully finds many hinge regions, such as the hinge region for calmodulin, which agree well with published residue selections. In other cases, the algo-

rithm comes close, identifying a residue selection that borders on a hinge. Hinges may be displayed graphically via a "hinge movie" identifying the putative hinge region or regions in red.

In related work, Wriggers *et al* presented techniques to identify protein domains and common hinges using an adaptive least-squares fitting technique[151]; the user is presented with a number of options (spatial connectivity maintenance, significant structural difference filters) to ensure optimal hinge finding. For the remote user's convenience, my own hinge finder is at present fully automatic and presents no options to the user. It may be advantageous for us to provide such options in the future so that the user can override and improve on the putative hinge initially selected by my algorithm, although this would partially defeat my efforts at standardization. Maiorov *et al*[152] have developed a system which detects hinges by large-scale sampling of torsion angle space; this technique, while presumably more accurate, is also much more computationally expensive then my current technique. It may be useful for us to give the user the option of using alternate hinge finding engines in the future.

To illustrate the putative hinge finder, a frame from one such "hinge movie" is given in Figure 3.4, with the putative hinge identified in black. Superposition, orientation, and hinge-finding are relatively fast steps, requiring a fraction of a second of computer time on my server.

Homogenization

I have modified the X-PLOR package[102] to homogenize the stored coordinates. This problem is non-trivial[33,153]. The initial, solved intermediate, and final conformations

are parsed by X-PLOR and examined for missing non-hydrogen coordinates. These are filled in using energy minimization with the known coordinates of the molecule fixed at their solved positions. If these missing coordinates are available in another solved conformation, the coordinates from the superimposed and rotated conformation are used as an initial guess as to their likely positions. As written, filling-in of missing non-hydrogen coordinates is necessary for the energy minimization subsystems to work robustly with a large number of PDB files. It also ensures homogenized output of PDB files, which is required by the visual rendering subsystem.

Interpolation

The next step is in the dominion of what I refer to as the "interpolation engine." Once the structures have been homogenized in terms of solved atomic coordinates, interpolation may proceed. Under command of the script, the custom X-PLOR interpolation function is repeatedly called, each time evenly reducing the distance between the current structure and the final structure. When more than two solved conformations are present, the distance between the current structure and a solved intermediate conformation is evenly reduced instead. Each step is followed by a round of energy minimization to correct molecular stereochemistry and enforce rules of chemical reality on the structure. To ensure that the final frames are as accurate as possible, the solved endpoint structures are used for these. When solved intermediates are present, these are inserted as frames at regular intervals. The entire process takes only a few minutes to produce ten frames running on a 500 MHz Intel Pentium III workstation running Linux.

There are many possible interpolation strategies, and a number of tradeoffs between accuracy, various computational resources, time, and other are involved in the choice. For

this reason, in addition to my original adiabatic mapping engine, I offer the user two engines based on LSQMAN[154,155] (one Cartesian-based and another based on internal phi, psi coordinates), which are faster but appear to be less realistic. Users wishing to add their own, non-trivial interpolation engines may contact the authors to make arrangements to do so. For example, a user wishing to analyze a very large number of trajectories (10000 or more from, e.g., samplings from molecular dynamics simulations) or higher might wish to supply a simplified interpolation engine and make other arrangements to allow the computations to be completed in a reasonable amount of time.

I chose my original technique, known in the literature as adiabatic mapping[101] for reasons of computational efficiency. It is a technique that produces chemically reasonable morphs with a modest amount of computational power and thus is most suitable for a Web-based server. This remains the default interpolation engine for the server. Using this engine, the server can produce a realistic interpolation of a protein and have the results rendered and returned to the user in less than three minutes on a fast Pentium III machine. Using adiabatic mapping, I have also produced my own morph of the motion in GroEL which, although probably less accurate than the considerably more expensive technique of normal mode analysis[126-129], is probably good enough for most researchers seeking only a visual representation. How close my predicted pathways come to reality is perhaps best answered through the emerging technique of time-resolved x-ray crystallography[83,156]. Thus, an adiabatic mapping engine is much more suited to my goal of automatically interpolating a large percentage of the motions in my database.

Visual Rendering

With the intermediate conformations morphed, the molecule is now visually rendered. I have written a Perl script that produces VRML 2.0 (Moving 3-D Worlds) code[114,157] on-the-fly from the intermediate PDB files. The VRML 2.0 output is suitable for interactively viewing the moving 3D macromolecule in a VRML 2.0 Internet browser, such as SGI CosmoPlayer 2.0. The advantage of the 3D display format is that the remote Internet user may easily choose a preferred orientation and vantage point.

The molecule is also rendered as a 2-D movie in the MultiGif, Quicktime, and MPEG formats, as well as an Adobe Portable Document Format (PDF)[158] page showing the individual frames. Remote adjustment of vantage point and orientation is not possible in the simpler 2D video format, so the molecule is rendered with the screw axis perpendicular to the plane of the display device, as was computed during the orientation process. The molecule is rendered in three display types[116,117]: ribbons (with secondary structure indicated), lines (as a simple alpha chain), and ball and stick (showing all individual non-hydrogen atoms). The first two formats are also rendered into a small moving MultiGif icon to afford the database user with a quickly downloaded preview of the larger movies available.

Statistics

In the process, key standardized statistics are recorded. These include maximum Cα displacement, rotation angle in degrees around putative hinge regions, sequences of the putative hinge regions, average torsion angle change in the hinge region versus the overall average, distance of the putative hinge region from the screw axis, distance of the screw axis from the centroid, a structural comparison score between the two domains,

and a number of additional, useful statistics, such as the differences in torsion angles at every aligned position and the pseudo CHARMM/X-PLOR[102,159] energy at each point in the morph.

These statistics are detailed enough to perform an automatic preliminary classification of the motion and determine the location of the hinge relative to the transformed axes. (For example, a large rotation angle indicates a probable hinge motion.) A detailed description of my statistical results is given in Table 3.1 for five motions. Ranges and averages of some of these statistics after several hundred alignments are given in Table 3.2 along with similar but sparse statistics culled manually from the scientific literature for comparison.

For example, over approximately 175 motions submitted for analysis, the median motion has a maximum rotation of 9.5º over a range of 0 through 150º as computed by my algorithm, whereas the twelve motions culled from the scientific literature had an average rotation of 24º over a range of 5 through 148º. Similarly, my algorithms found a median maximum Cα displacement of 17 Å ranging from 0 to 81Å for the submitted motions, whereas eleven motions reported in the scientific literature average 12Å over a range of 1.5 through 60Å. Although most of the structures are very similar in sequence, the server has been able to accommodate sequence identity down to 8% for some motions (see Table 3.3). Most motions have at least one large torsion angle change (see Table 3.4).

The sparseness of manually culled data in Table 3.2 is due to the lack of a standardized nomenclature for these statistics in the scientific literature. It is worth noting that a different set of proteins had to be used for each of the manually culled tallies in Table

3.2. Because these statistics predate the server, they serve as a manual "gold standard" against which the results of the server may be compared. Table 3.1 presents a statistical description of motions in the database, a main scientific benefit of the server.

Integration with Database

Privacy is a concern with some submissions, so users are afforded the option to either keep their submissions secret until the results have been published or to cause the submission to appear immediately in an index. For each successfully completed morph, the server produces a Web page allowing easy download of the coordinates (as an archive of PDB files or in NMR format) or movies (in a number of video formats), in addition to displaying the molecule in the moving VRML format. The page includes the standardized statistics discussed above generated for the conformations used in the morph. This page may be accessed through a URL containing a special code that is emailed back to the submitting user when the morph is complete; for users seeking to keep their morphs private (for publication reasons), this URL serves as the user's password, allowing access to the morph page in the server. For public morphs, these pages are also accessible through an index, http://bioinfo.mbb.yale.edu/MolMovDB/movies.

The ultimate flow of information is circular. For each motion I either link it via a motion id to an existing entry in the Macromolecular Motions Database or I generate a new entry in the database. The results of analyzing particular ordered sets of structures ('strings' of structures) are entered under an appropriate identifier into the Database of Macromolecular Motions for further reference, and, in many cases, suggest further structures to study and analyze. Each comparison is assigned a unique ID entered into the "comparison table" in the database that references the IDs of the PDB structures in-

volved. These comparisons are, in turn, referenced by entries in the motions database (these references may be generated by comparing the IDs of the PDB structures referenced in each comparison table entry with the PDB structures referenced in each motion table entry.). Because many motions in the database are associated with more than two structures, more than one comparison is often possible and some database entries do reference multiple comparisons.

New movies, which lack a motion entry in the Database of Macromolecular Motions, have an entry automatically created with minimal or no annotation. This is indicated in the entry by setting the annotation level to zero. (Annotation levels range from 0 to 10. A level of "0" indicates the entry was automatically created with no human intervention. "10" indicates significant human intervention, typically in the form of a large amount of descriptive text present in the entry.) The user can annotate the new entry using an easy-to-use edit form displayed in his or her Web browser. Existing entries are also editable by the community through the same Web form with prior authorization from the database's maintainers. All changes are subsequently reviewed by the maintainers to assure quality control. In this way, the Database of Macromolecular Motions is used to classify and organize morphs submitted to the Morph Server.

Examples

To illustrate the technique of adiabatic mapping as implemented by the server, Figures 3.4 depicts the frames in five automatically generated morphs produced by the server's adiabatic mapping interpolation engine.

ADH. First is a "trivial" morph, Alcohol Dehydrogenase[160,161]. This morph is con-

sidered “trivial” because a true motion is involved, and the endpoint conformations are sufficiently close together that a pathway between the two—not involving chain breaks or clearly distorted geometry—is easy to construct in the mind’s eye. Therefore, one would intuitively expect software that claims to perform morphing to handle this case with similar ease. This is indeed the case, as can be seen in the figure, which depicts the frames generated by the morph server for the morph of Alcohol Dehydrogenase. The protein has very little movement; the figure shows the frames in the motion generated by my server with an arrow to indicate the region of movement. When the actual animation is played back, the arrow is not necessary, as the eye has evolved to be especially sensitive to motion and easily picks out the movement in the movie.

Recoverin. Recoverin[162-164] is an example of a “typical” morph. The morph is considered “typical” because a true motion is involved, as can be seen in the figure, the motion involves most of the molecule and is therefore qualitatively more extensive than that of a “trivial” motion such as Alcohol Dehydrogenase. The motion is sufficiently complicated that a simple linear interpolation would produce at least some obvious distortion and physical impossibilities. Nevertheless, the adiabatic mapping interpolation engine produces a realistic morph without chain breaks or clearly distorted geometry.

Pol-beta. A morph of DNA Polymerase Beta[121,165], considered “typical” for much the same reasons as recoverin.

GroEL. The user should be aware of a number of problems that can be encountered in the adiabatic mapping method. Problems arise for large deformations if the energy minimization methods cannot effectively remove the accumulated stresses[166]. These problems are endemic to all adiabatic mapping systems, including my Web server. This problem is illustrated in Figure 3.5d, which shows a morph of one subunit in GroEL[126-129], a "medium difficulty" morph because of the considerable atomic displacements between the starting and ending conformations. However, this GroEL motion still represents a true protein motion, and the server still produces a fairly realistic interpolation. One means of improving this morph would be to have the user select additional interpolation frames (and, hence, additional energy minimizations). (This is in a sense a "feature" that highlights which motions are sterically more difficult to achieve.)

DT. My model will, of course, break when fed an "impossible" morph, as shown in Figure 3.5e. The endpoint conformations are that of diphtheria toxin (DT), not a true motion but rather an example of domain swapping[167,168] in which the domains in DT have been solved while bound in two different configurations. For one conformation to "morph" into another, the easiest physically realistic route would be for one domain to unfold and refold. Indeed, the morph generated by the server does suggest a process of this sort.

While the morph server is unable to generate a physically realistic movie of this "motion," it does suggest that the morph server may be used as a quick visual tool in evaluating the validity of a proposed motion. Comprehensive statistics for all five morphs may

be found in Table 3.1.

Discussion

Statistics

In a majority of cases the structures of a given macromolecule involved in motions have been solved in two or more conformations, so that endpoints for the motion are available. This, in turn, means that automatic conformation comparison tools are possible, which, when applied *en masse* to the motions database, allow the generation of a consistent, standardized set of statistics characterizing the motions in the database. In the process of analyzing the structures, pathway interpolation is possible as well.

What constitutes an optimal morph?

Since the goal of the server was to output only a single interpolated pathway "morph", it is necessary to define more precisely what is desired. Define the "optimal morph" as the most likely (or most frequently taken) pathway between two conformations. In the large dimensional space of macromolecular atomic coordinate space, an infinite number of paths between conformations exist, so that establishing that a given local ensemble of pathways is the most statistically probable is, in general, computationally intractable. A more realistic approach would be simply to find a morph that is a reasonably good reaction coordinate that does not produce any large chemical distortions. This reduced the computational complexity of the problem, yet ensured that the resulting morph would be insightful, yet likely be very similar to the "optimal" morph.

Unlike the adiabatic interpolation engine used in the server, a number of interpolation engines on proteins have taken approaches that do not meet these criteria. With the exception of the simplest motions, simple linear interpolations of atomic coordinates without consideration of physical reality yields intermediates with clearly distorted geometry. In some cases, atoms may be significantly closer than their van der Waals radii would permit, or further apart than a chemical bond would reasonably be expected to allow. A more sophisticated approach to morph movies not currently taken by the server due to its stringent computational requirements, but one which might be added in the future, involves the use of normal mode analysis, such as was done on GroEL by Xu *et al*[126-129].

Conclusions

I have developed an integrated set of protein conformation comparison tools on the Web for use in conjunction with the Macromolecular Motions Database or as a standalone, publicly accessible server. When solved endpoint structures are available, the server can produce a useful comparison of the structures involved in protein motions. The server also implements a database of protein motions accessible on the Web or generated by Internet users through my server; this database is integrated into the Molecular Motions Database.

The server collects a number of statistics on the motion, including maximum Cα displacement and maximum rotation around the putative hinge, which are useful both in analyzing and classifying individual proteins and in generating a statistical picture of motions in the motions database as a whole. The software also homogenizes the incoming structures, attempting to solve for missing atoms using a molecular dynamics algorithm.

The server then uses an adiabatic mapping technique to generate a visually rendered interpolated pathway, or ‘morph’, of the motion or evolution of the protein. The homogenized endpoint coordinates and the generated intermediate coordinates are made available for download.

The software presents the visual representation, statistics, orientation, alignment, and interpolated coordinates to the user. At user option, these results may become public immediately or remain private until paper publication. Through an easy-to-use Web form, the user is afforded an opportunity to create a descriptive entry in the Database of Macromolecular Motions for the protein structures involved, referencing the morph results, as well. I have found the server useful in the analysis of protein motions and anticipate that use of the server will help standardize statistics and nomenclature for protein motions subsequently presented in the scientific literature.

Table 3.1: Comprehensive Statistics

	Statistic	*[Code]*	*Easy*	**Typical**		*Large*	*Impos-sible*
			ADH	**Reco-verin**	**DNA Pol-Beta**	*GroEL*	*Dipth-eria Toxin*
Input Structures	Motion ID	[ID]	adh	recvin	polbeta	groel	dt
	1st input frame	[inputframe0]	8ADH	1IKU	1BPD	1GRL	1DDT
	2nd input frame	[inputframe1]	6ADH	1JSA	2BPF	1AON	1MDT
	Size (Å) (in terms of window for rendering)	[max_x_or_y]	36	41	52	55	39
	Number of atoms	[natoms]	2887	1639	2697	3993	4110
	Number of residues	[nresidues]	374	201	335	548	535
Overall Motion	Overall RMS between first and last frames	[RMSoverall]	2.0	13	8.6	16	20
	Rotation (degrees)	[kappa]	4.9º	73º	9.9º	62º	62º
	Overall translation of centroid (Å)	[translation]	2.1	13	6.1	47	66
	X translation (Å)	[TransX]	1.1	-0.24	0.94	45	-45
	Y "" .	[TransY]	-0.95	-9.14	4.1	-2.1	-0.54
	Z "" .	[TransZ]	1.5	-9.78	-4.4	-10	48
1st Core	Number Cα's in 1st core	[AlignedCoreCAs]	187	95	160	259	262
	RMS of 1st core (Å)	[AlignedCoreRMS]	0.40	3.0	0.92	1.4	0.37
	Max Cα displacement in 1st Core (Å)	[MaxCore Deviation]	0.66	7.6	1.7	4.2	0.60
2nd Core	Num. Cα's in 2nd core	[2ndCoreCAs]	190	94	160	260	260
	RMS of 2nd core (Å)	[2ndCoreRMS]	2.9	18	12	23	29
	Max Cα displacement in 2nd core (Å)	[Max2ndCore Deviation]	7.1	38	28	49	60
	RMS of 2nd core (Å) after fitting on 1st core	[2ndCoreRMS postrefitting]	1.6	11	11	10	18
Hinge	Number of putative hinges detected	[NHinges]	0	0	0	1	1
	X position of 1st hinge (Å) rel. to centroid	[Hinge000X]	-	-	-	-4.7	-7.2
	Y position ""	[Hinge000Y]	-	-	-	11	-0.91
	Z position ""	[Hinge000Z]	-	-	-	3.3	-3.0
	1st Hinge Residue Selection	[Hinge000res]	-	-	-	380:403	352:375
	Sequence of 1st putative hinge	[Hinge000seq]	-	-	-	EVEM KEKK ARVE DALH ATRA AVEE	NLFQVVHNS YNRPAYSPG HKTQP
Screw Axis	Distance betw. screw-axis (x0) & centroid (Å)	[x0ToCentroid Distance]	21	8.4	23	30	39
	X displacement centroid from screw axis (Å)	[x0X]	-0.16	-0.5	-2.5	17	-20
	Y ""	[x0Y]	-5.0	-6.2	-5.2	-16	-24
	Z ""	[x0Z]	-20	5.7	-22	19	-24
	Distance between screw axis and 1st hinge (Å)	[Hinge000x0dist]	-	-	-	26	45
Torsion Angles	Max phi change (Max of Abs. degrees, 0º-180º)	[MaxPhi]	180º	180º	180º	180º	180º
	Max psi change	[MaxPsi]	180º	180º	180º	180º	170º
	Max alpha change	[MaxAlpha]	150º	180º	180º	180º	170º

Comprehensive Statistics for alcohol dehydrogenase, reoverin, DNA polymerase beta, GroEL and diphtheria toxin as Reported by the Server. These statistics were automatically generated by the server in the course of morphing alcohol dehydrogenase, recov-

erin, DNA polymerase beta, and the first chain of GroEL, and diphtheria toxin. They are reported here to two significant figures except where exact. A brief explanation for each statistic may be found above. More comprehensive explanations may be found on-line.

Table 3.2: Automatically gathered versus manually gathered statistics

		Hand-gathered statistics				Automatically collected motion statistics				
Value		Min	Max	Mean		Min	Max	Mean	Median	Stdev
Maximum Cα displacement (Å)		1.5	60	12		0.90	81	23	17	19
Maximum hinge rotation (º)		5	148	24		0.0	150	35	9.5	46

Comparison of statistics between automatically gathered (server gathered) and manually gathered statistics for maximum Cα displacement and maximum rotation. Despite the sparseness of the manually culled data, the statistics are roughly comparable. Maximum Cα displacement was calculated by first sieve-fitting the protein conformations. The 81Å motion in the database is due to Oxo-Acid-Lyase (5CTS to 1AJ8 in the PDB.) The 12 references reporting maximum rotation in the literature reported a mean maximum rotation of 24º, whereas the server found a mean maximum rotation of 35º over the 176 entries present at the time the table was generated. The mean is, however, skewed by some of the larger motions; the median displacement is much smaller. The maximum value of 150° is due to Oxidoreductase (1FMC -> 1HDC in the PDB. To collect the manual data, I found eleven entries in the Database of Macromolecular Motions citing manually gathered Cα displacement statistics from the literature, and twelve entries giving manually gathered maximum hinge rotations. (Some researchers reported only Cα displacement while others reported only maximum hinge rotation, so these correspond to different sets of proteins.) Automatic collection used a sample of 184 motions for Cα displacement and 176 motions for maximum hinge rotation.

Table 3.3: Structural Similarity Statistics

Statistic on 65 observations	Mean	Minimum	Maximum
Number of residues aligned	250	5	780
Trimmed RMS	2.4	0.24	16
Trimmed RMS p-value	0.041	0.0	0.96
Sequence percent identity	55	7.9	100
Sequence identity p-value	0.23	0.0	1.00
Sequence Smith-Waterman Score	1400	-7400	15000
Structural Similarity Score	4400	97	15000
Structural Similarity Score p-value	0.015	0.0	1.00

Morphs in the database were processed to eliminate redundancy (several PDB pairs have multiple morph movies of varying characteristics) and then fed into the Yale Structural Alignment Server (URL: http://bioinfo.mbb.yale.edu/align) based on structure alignment[145]. Structure alignment was able to structurally align 65 of the 78 non-redundant protein chain pairs. The results for 65 observations are shown in the table above to two significant figures.

On average, successful protein chain comparisons in the database have a sequence percent identity of 55%, although the server was able to successfully morph proteins with as little sequence identity as 7.9% identity and as high as 100% identity. Morphed proteins have a mean trimmed RMS (RMS after worst-fitting half of residues eliminated) of 2.4 Å, with a range between 0.245 Å to 16.46 Å. (Trimmed RMS differences at the high end of this range (16 Å) indicate (i) large changes in the relative positions of domains, either because of their reorganization or their being “swapped”; (ii) other experimental artifacts; or (iii) other errors in the input files or choice of input files.)

The server was able to successfully morph protein chains with p-values based on all three statistics (Trimmed RMS, Sequence percent identity, and Structural Similarity Score) near one, suggesting that some protein chain pairs in the database are unlikely to be related either evolutionarily or structurally.

Table 3.4: Torsion Angle Statistics

Name	Mean of max	Min of max	Max of max
Maximum Alpha Change	140º	16º	180º
Maximum Phi Change	180º	140º	180º
Maximum Psi Change	150º	23º	180º

Maximum Torsion Angle Changes is another example of the statistics collected by the server. For this table, maximum Alpha, Phi, and Psi Torsion Angle Changes were computed for 134 protein chain pairs in the database and reported here to two significant figures. The mean, minimum, and maximum of each statistic were computed for the table above. As expected, a motion can be found for each statistic with a torsion angle change of 180º, the maximum possible. Every motion involves at least one large phi angle change of at least 140º. However, a few morphs have only small psi and alpha torsion angle changes. Alpha is the dihedral angle relating virtual bonds connecting Cα atoms between residues along the peptide chain; it is computed by pretending each residue is an atom with center at its Cα atom.

Figures

Figure 3.1. Diagram of my approach.

The information flow from databases, through the server, and then back again to databases is broken down into its component steps. Experimental data in the PDB and other databases is converted into a motion entry in the Database of Macromolecular Motions, from whence a morph movie is generated and statistics are collected. These results are subsequently stored in the Database of Macromolecular Motions. The interaction of the server with the peripheral parts in the figure ("Database Information", "Experimental Methods and Simulations", and "Users") is largely under users' control, although I am developing automated tools to generate comparisons automatically from databases such as SCOP. The results of a comparison are both returned to the user and referenced in the Database of Macromolecular Motions, hence the arrow back to "Database Information." The Web report extract information both from server results and from pre-existing information in the Database of Macromolecular Motions, if any, hence the arrow from "Database Information" to "Web Report."

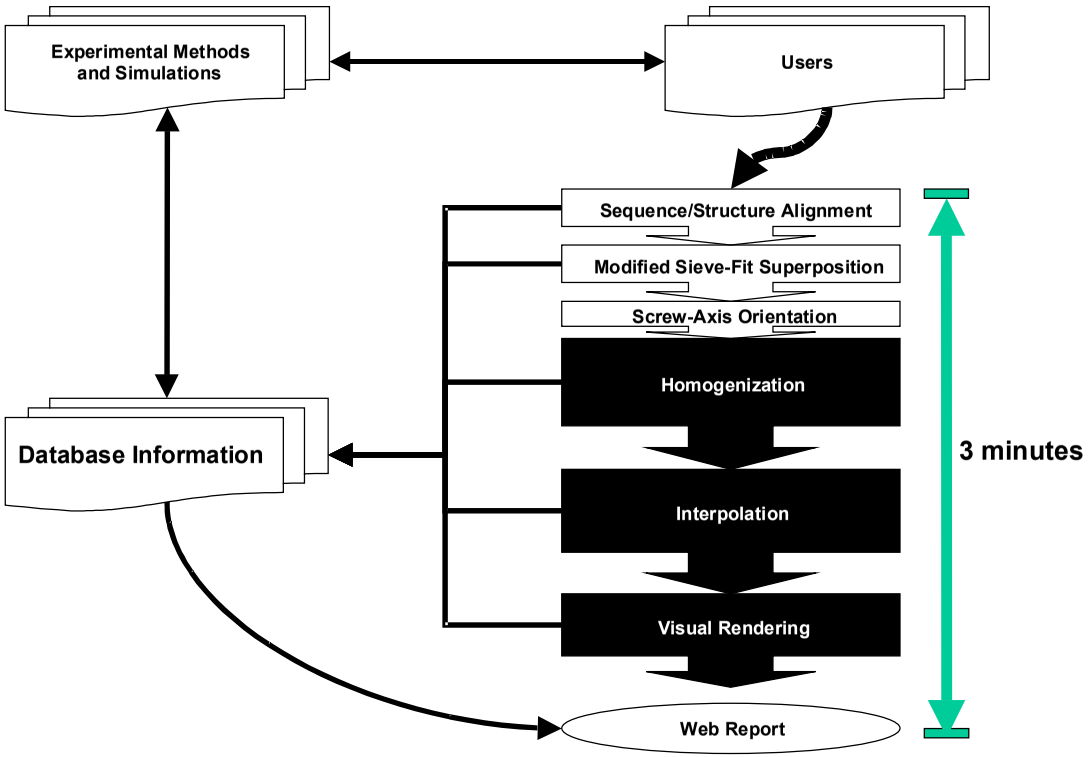

Experimental Methods and Simulations
Users
Sequence/Structure Alignment
Modified Sieve-Fit Superposition
Screw-Axis Orientation
Homogenization
Interpolation
Visual Rendering
Database Information
Web Report
3 minutes

Figure 3.2a (left):Linked Web Pages

Here, the information flow may be visualized as a series of linked Web pages. Users submit new motions to the server via either the Server Submission Form or via a simplified interface through the Structural Alignment Server's submission form. The query is processed by the server. If the morph operation is successful, the new morph is added to the Table of Morph Movies (which links off-site URLs as well). This table has links to both the morph's report form (from which the morph may be viewed) and also the associated motion entry in the Database of Macromolecular Motions is the motion has one. An entry is also created in the motion's entry in the Database of Macromolecular Motions, linking the motion's report to the report for the morph movie.

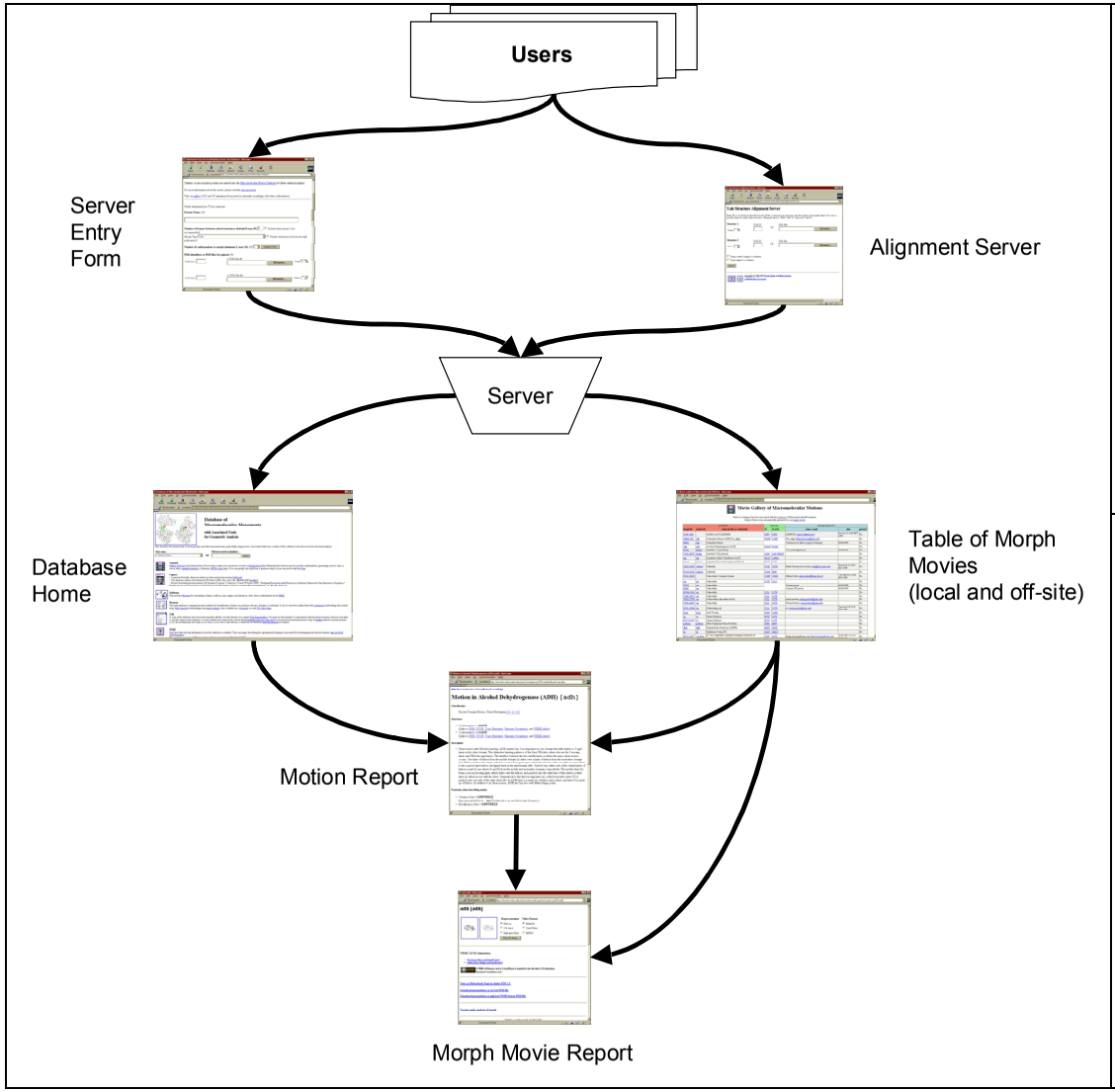
Figure 3.2b (top right): Database Main Page

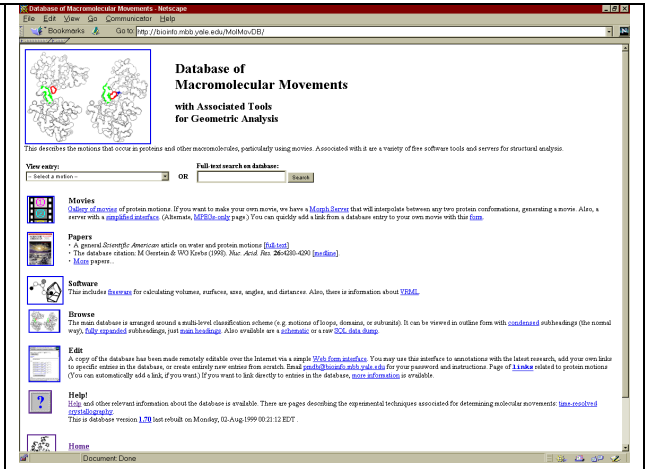
This is a blow up of main page of database from Figure 3.2a. The entry page of the Database of Macromolecular Motions, http://bioinfo.mbb.yale.edu/MolMovDB is shown above. Users may jump from this to entries on specific motions, many of which link morph movies, or to a table of morphs (Figure 2c).

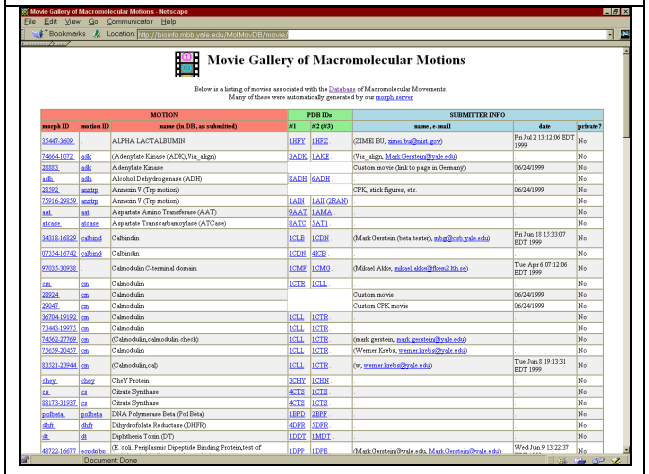
Figure 3.2c (bottom right): On-line Table of Morphs Page

This is a blow up of On-line Table of Morphs from Figure 3.2a. Screen shot of the on-line table of morphs web page at http://bioinfo.mbb.yale.edu/MolMovDB/morphs. In ad-

dition to linking to the Web report page for the morph, each entry links to the corresponding database motion entry (if applicable) and provides information on the PDB Ids used the generate the morph movie, along with the information on the submitting user. This table also references off-site morph URLs, and thus functions as a comprehensive database of protein morphs available on the Internet.



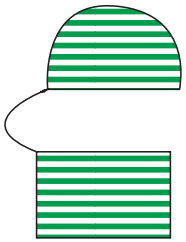




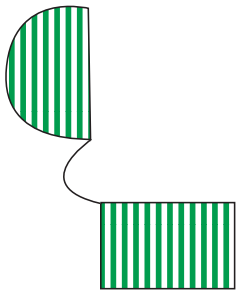
Figures 3.2a (left), 3.2b (top right), and 3.2c (bottom right)

Figure 3.3: Superposition of a Calmodulin-like protein undergoing a hinge motion.

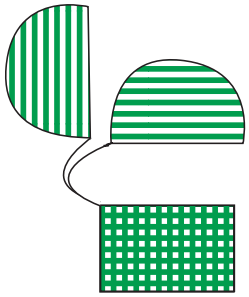
Structures 1 and 2 indicate the closed and open conformations, respectively. Compare "Global Fit", the superposition produced by a tradition least-squares fit of the structures, to "Core 1" and "Core 2", the two possible superpositions produced by sieve-fitting. The final panel depicts how a morph movie might appear using the "Core 2" superposition.



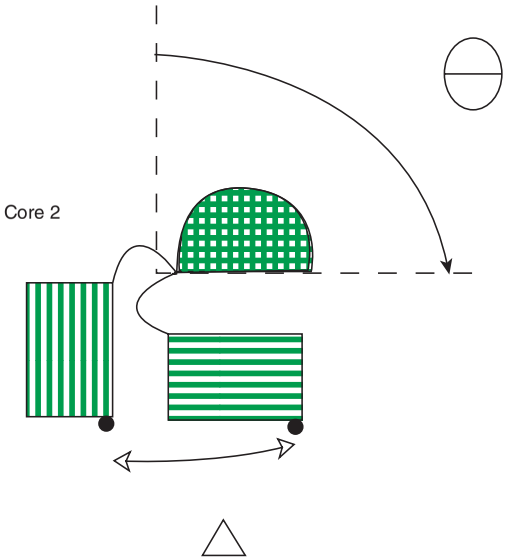
Structure 1

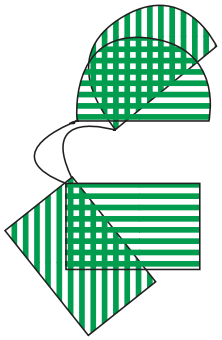


Structure 2



Core 1





Global Fit

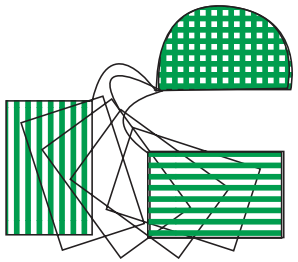


Figure 3.4: Putative Hinge Movie

A frame from a “hinge movie” of ras protein (PDB ID 4Q21 to 6Q21 morph intermediate frame) showing the putative hinge regions as identified by the server. The server identifies 71:82 and 118:129 as putative hinge regions in the motion, here shown in black.

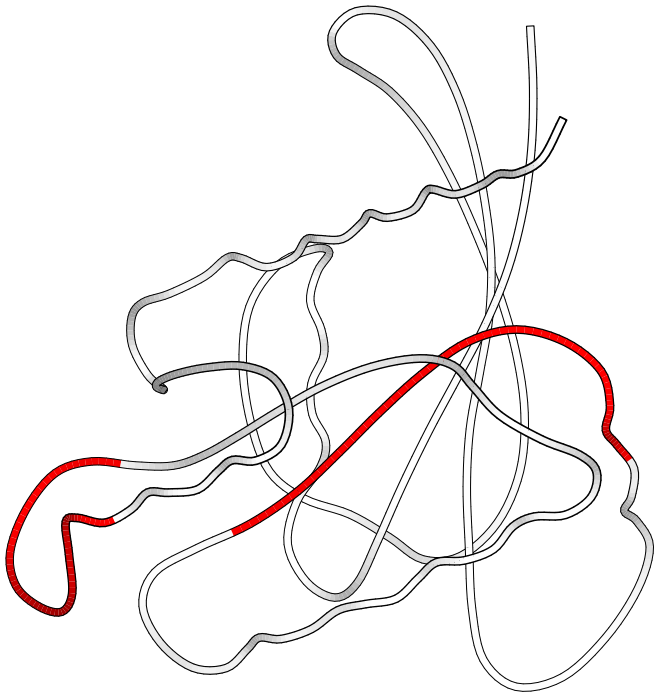
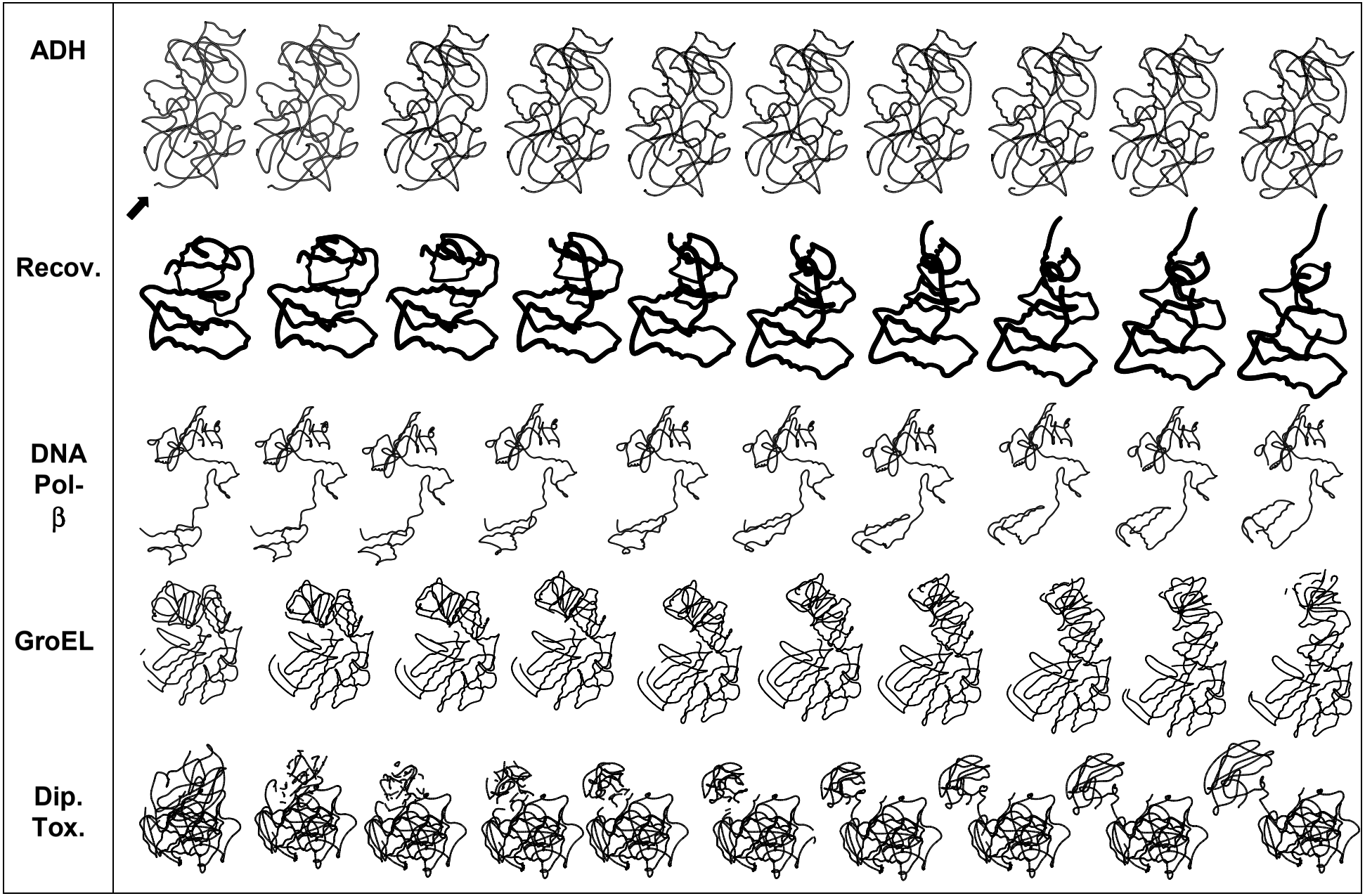


Figure 3.5: Sample morphs.

An automatic morph of alcohol dehydrogenase (key “adh”) as produced by my server. Alcohol dehydrogenase is a “trivial” case, as the motions involved are relatively small but nevertheless dramatic when viewed as a movie. It is shown in the top panel. The two panels below ADH show recoverin (1iku -> 1jsa) and DNA polymerase beta, respectively, which are “easy” cases. GroEL (key “groel”) is shown as an intermediate case, as the motions are much larger than in alcohol dehydrogenase. The morph can still be reasonably handled by server, as is especially dramatic on paper due to the large displacement of the motions involved. Ditheria Toxin (key “dt”) a hard or impossible case, because the rearrangement between the conformations does not involve a motion, but rather domain switching in the crystalline state. The poor quality of the morph provides the researcher with an immediate clue that the rearrangement pathway is unlikely to be a pure motion. The default MultiGif (or Moving Gif) using a combination of software, including Rasmol[117], Molscript[116], Ghostscript, and a gif to multigif utility, all driven through a Perl script. Additional software renders the molecule into Quicktime and MPEG formats to ensure display in a number of Internet browser environments. A simple HTML and Adobe PDF rendering of the sequence alignment of the residues between conformations is also available. In addition to visual output, the interpolated coordinates can also be downloaded as either an PDB NMR format archive or as an archive of PDB frames in the popular Unix Tape Archive (“.tar” file) format.


ADH
Recov.
DNA Pol-β
GroEL
Dip. Tox.

Chapter 4: Normal Mode Statistics-Based Automatic Classification of a Database of Macromolecular Motions

Overview

In this chapter, submitted to the journal *Proteins*, I describe how I have investigated protein motions using normal modes within a database framework. From a comprehensive set of structural alignments of the proteins in the PDB, I identified a large number of instances of protein flexibility, consisting of pairs of proteins that were considerably different in structure given their sequence similarity. On each pair in this dataset of "outliers," I performed geometric comparisons and adiabatic-mapping interpolations in a high-throughput pipeline, arriving at a list of 3814 motions and standardized statistics for each. I then did simplified normal-mode calculations on each protein in this “list”, determining the linear combination of modes that best approximated the observed motion. Based on this, I identified a statistic, mode concentration, related to the mathematical concept of information content, that describes the degree to which an observed motion can be summarized by a few modes. I investigated mode concentration in comparison to related statistics on mode combinations and correlated it with quantities characterizing protein flexibility (maximum backbone displacement or number of mobile atoms). To demonstrate the utility of mode concentration, I evaluated its ability to automatically classify the “list” motions into a variety of simple categories (e.g. whether or not they are “hinge-like”), in comparison with other quantities. This involved application of decision trees and feature selection to training and testing sets derived from merging the “list” of motions with manually classified ones. I integrated my normal mode calculations in the

Macromolecular Motions database, through a web interface at http://molmovdb.org/modes.

Introduction

Protein motions play a key role in a wide range of biological phenomena, including chemical concentration regulation, signal transduction, transport of metabolites, and cellular locomotion[118,119,139]. Motion is typically the way a structure actually carries out a specific function; for this reason, motions are an essential link between function and structure.

I previously developed a database of macromolecular motions[139,140,169], which consisted of crystallographically documented protein motions coupled to a collection of protein "morph" movies and related statistics[170]. Here, I cull ~4,000 putative motions from the PDB[171] using an automated technique and add these to statistics and movies in my existing database. I add new statistics calculated from an analysis of the normal mode vibrations of the protein pairs, and apply artificial intelligence feature analysis techniques to identify a useful statistic, mode concentration, that is computed from normal mode analysis. The aims of my present study are threefold:

(1) to build a pipelined biological database framework for the study of protein motions, consisting of (a) a raw experimental database (the PDB) (b) a condensed statistical representation (the macromolecular motions database and its associated analytical tools) (c) application of automated data mining techniques to the con-

densed representation to identify key statistics and move towards automatic classification (feature analysis on the macromolecular motions database);

(2) to make available a useful database of thousands of new, putative motions (full outlier set);

(3) to present the results of a normal mode analysis on the augmented database (working outlier set); and

(4) to present the results of automated data mining techniques (identification of key statistics).

My work builds upon a rich literature in macromolecular motions[172-175]. Motion related to proteins' mechanical function has mainly been studied experimentally by x-ray crystallography. Traditional x-ray crystallography has provided key insights into the relationships between conformational change and macromolecular function; GroEL[128] and beta-actin[86] are just two of many examples. Progress in the field of time-resolved x-ray crystallography[82,83,176] has also enhanced the study of biologically significant protein conformational change. Recently, it has become possible to study larger protein conformational changes via NMR[177]. Other approaches have focused on the use of computational methods[91,178-184].

Normal mode analysis is another computational approach that can be applied to protein conformational change. Widely used by spectroscopists for many years to associate IR and Raman experimental peaks with small molecule vibrational modes[185], advances in computer technology over the last few decades has made normal mode analysis of pro-

teins and other large molecules practical. This was first applied to proteins in the mid-80s and has subsequently been scaled up[186-192]. The concept of normal mode analysis is to find a set of basis vectors (normal modes) describing the molecule's concerted atomic motion and spanning the set of all $3N-6$ degrees of freedom. For very large molecules, it is often more of interest to try to find a small subset of these normal modes that seem in some way especially important. By modeling the interatomic bonds as springs and analyzing the protein as a large set of coupled harmonic oscillators, one can calculate a frequency of periodic motion associated with each normal mode, and then attempt to find normal modes with low frequencies. The low-frequency normal modes of proteins are thought to correspond to the large-scale real-world vibrations of the protein, and can be used to deduce significant biological properties. There is evidence to suggest[193-198] that proper, symmetric normal mode vibration of binding pockets is crucial to correct biological activity in some proteins.

The principal of normal mode analysis is to solve an eigenvalue equation of the form

$$\ddot{\mathbf{q}} + \mathbf{F}\mathbf{q} = \mathbf{0}$$

where the vector q is a vector representing the displacements in three dimensions of the various atoms of the molecule, and F is matrix that can be computed from the system's mass and potential energy functions. Solutions to the above system are vectors of periodic functions (the normal modes) vibrating in unison at the characteristic frequency of the mode.

In this chapter I apply normal mode analysis to the study of protein motions. Fundamentally, I use normal modes over MD and other related computational techniques for pragmatic reasons. It would not be possible to apply MD to ~4000 conformational changes. Furthermore, normal mode analysis gives a concise description of a motion (in terms of a small number of modes) that is ideal for subsequent statistical tabulation.

Normal mode analysis corresponds to an approximation of reality. It is not as accurate a model for conformational change as many other alternate techniques. However, solely from the standpoint of this study, it has several advantages over a technique like molecular dynamics. First, a typical normal mode analysis will consume orders of magnitude less computer time than a comparable molecular dynamics simulation, although normal mode analysis often requires far more memory. Second, normal mode analysis presents the proteins motions in terms of a simple, intuitive concept from classical physics: the vibrations of a coupled harmonic oscillator. These features make it attractive for use within my statistical database framework.

My normal mode analyses are related to the 'Essential Dynamics' (ED) methods of Berdensen[199], consisting of a principal components analysis of normal mode atomic displacements and how they relate to experimentally solved conformations. However, my analysis is formally different, and I take my analysis a step further by summarizing it statistically, which is appropriate given my database framework. Many of the problems customarily found in ED analyses also apply: e.g., the superfluous rotational and translational differences must be eliminated by superimposing the experimental structures to fix

at least one domain; in the process, the motion's screw-axis may be characterized[200]. Previously, I developed web software tools to solve these problems in a different way using purely experimental information[170]. I analyze a comprehensive database of thousands of putative protein motions, whereas existing publications limit their scope to single proteins or databases specific to certain types of proteins.

Materials and Methods

Data sources

Full Outlier Set

To identify a large dataset of proteins with conformational changes, Wilson et al.[201] performed automatic pairwise sequence, structure, and function comparisons on about 30,000 pairs of protein domains constructed according to fold classification (http://scop.mrc-lmb.cam.ac.uk/scop/)[34,35,143,202-204]. From these, they isolated the "full outlier set", which consists of about 4,400 pairs of likely protein motions.

Figure 4.1 shows how the full outlier set was created. Wilson et al.[201] plotted RMS structure alignment scores against sequence percent identity for the 30,000 SCOP domain pairs they identified from the PDB. They then binned the plot into one-percent wide bins. The mean RMS and standard deviation for the points in each one-percent bin were computed. Points lying more than two standard deviations above the mean were removed from the dataset and used to generate a new dataset, the outlier dataset, which ultimately consisted of 4,400 such pairs.

Workable Outlier Set

I ran the full outlier set through my protein morphing server[170]. I placed the resulting database of pre-processed PDB files, morph statistics, and movies, on the World Wide Web, organized by their SCOP fold classification. The new automated approach was able to process and generate several thousand new morph movies. As described below, the morph server acted as a filter, eliminating about 600 motion pairs. Next, I applied the normal mode analysis described below on the successfully morphed pairs, to produce a set of about 3,800 motion pairs, the "outlier set". In this chapter I concentrate exclusively on this new "workable outlier set" data. The dataset may be downloaded from http://bioinfo.mbb.yale.edu/molmovdb/datasets/workableoutliers.txt.

In order to perform feature analysis data mining on the outlier set, I classified two subsets of the workable outlier set (the "manual set" and the "extended set") into the classification schema of the Database of Macromolecular Motions[139] ("fragment", "domain", "subunit", "complex" on the basis of size and "hinge", "shear", "neither hinge nor shear" and "unclassifiable" on the basis of packing). Further details about this classification may be found in Gerstein et al.[139].

Manual Set

For the “manual set”, I performed a database merge of the “outlier set” against the previously published set of manually classified motions in the Database of Macromolecular Motions[139], the “1998 motions.” The PDB identifiers in each motion pair in the outlier set were checked for matches against the PDB identifiers associated with the 1998 motions. When a match was found (meaning the protein that had been manually classified), the motion pair was given the same classification as its constituent protein had been given in the database. 245 motion pairs met this criterion and were classified accordingly. Classifications in this manual training are expected to be accurate. (There was, however, one issue in applying this merge: GroEL is classified both as a subunit and a fragment motion. Because the Morph server analyzes single domains, not entire subunits, the fragment classification was used in this isolated case.)

Extended Set

To enlarge the training data for the machine learning analysis, I constructed a second, larger training set (the “extended set”). For a variety of physical reasons, proteins sharing the same fold family generally share a similar motion classification. Consequently, I constructed this set under the assumption that domains sharing a fold usually share a motion classification. The outlier set is constructed in such a way that both pairs always belong to the same fold family. It was therefore necessary only to determine the

SCOP fold classification[34,35] for each of the 245 motion pairs in manual training set and then assign the classification in the manual set to the entire SCOP fold family. Pairs in the outlier set belonging to this SCOP fold family then simply received the family's classification. In this way I identified a set of 1670 motions, which I call the “extended training set”. This set of classifications, although potentially less accurate than the manual training set, is still quite useful. Larger training sets can produce more accurate decision trees. For this reason it is possible that a decision tree produced from the larger extended training set may classify more accurately than one produced from the smaller, more accurate manual set, although this may seem counterintuitive. Comparing the decision trees produced by the manual and the extended training sets will serve as a useful check.

Preprocessing with Morph Server

I analyzed 3,814 proteins using this method from the full outlier set. Previously[170], I modified the X-PLOR package[102] to homogenize the stored coordinates, a non-trivial problem[33,153]. Filling-in of missing non-hydrogen coordinates was necessary for the energy minimization subsystems to work robustly with a large number of PDB files and ensured consistent numbering of atoms so the PDB files for the starting and ending conformations had to be pre-processed (“homogenized”) by the Morph Server[170]. Only pairs of protein conformations for which the morph server had successfully produced a movie were considered; this had the effect of filtering out pairs unlikely to involve a true motion, although no doubt some pairs which did not represent a true biological motion nevertheless did generate a movie. The Morph Server also removes overall rotation and translation motions from the input structure.

High-throughput Normal Mode Analysis of the Outlier Set

I used MMTK[205] to carry out normal mode analysis on the pre-processed PDB file pairs. The numerical Python module[206] made the linear algebra computations. A master Perl[207] script fed database information to the slave Python MMTK module. The results reported here were performed by computing the normal modes of the starting structures in each pair. Reversing the calculations by computing the normal modes of the ending structures did not appreciably alter the results.

Finding the normal modes themselves dominated the time and memory requirements of my analyses. In order to process the larger proteins in my database, I approximated each residue as a single, virtual atom centered at its C-α coordinate and selected the corresponding standard force field in MMTK[205]. This made the memory requirements of the normal mode analysis tractable on the lab's systems. To further accelerate the computations, I restricted MMTK to compute only the twenty lowest-frequency normal modes.

I used the MMTK deformation forcefield model. In this model, the energy is computed as the difference between some displaced model and the experimental structure using the formula:

$$E_i = \frac{1}{2}\sum_{j=1}^{N} k\left(\mathbf{R}_{ij}^{(0)}\right)\left[\left|\mathbf{R}_{ij}^{(0)} + \mathbf{d}_i - \mathbf{d}_j\right| - \left|\mathbf{R}_{ij}^{(0)}\right|\right]^2 \quad (0.1)$$

where k is a constant, $\mathbf{R}_{ij}^{(0)}$ is the distance from atom *i* to atom *j* in the experimental structure, $\mathbf{d}_i$ is the distance between the atom *i* in the displaced structure and the same atom in the ground-state experimental structure.

Each calculation averaged 20 seconds per protein pair on a 450-Mhz Pentium III processor with 0.7 Gigabytes of RAM running the Red Hat Linux operating system. An average analysis took about 100 Megabytes of memory to invert the matrix.

Theoretical Approach For Analysis of Normal Mode Statistics

I computed a number of key statistics on the normal modes, which I describe here.

Analysis of Observed Motion

The lowest frequency normal modes determined by Normal Mode Analysis may be represented as an $m \times n$ matrix $\mathbf{A}$, where *m* is three times the number of atoms in the system (one entry for each Cartesian axis), and *n* is the number of normal modes of interest. In this chapter, *n* is twenty.

Imagine a vector $\bar{\mathbf{v}}$ of length n, specifying some interesting linear combination of normal modes. Then $\mathbf{A}\bar{\mathbf{v}}$ is a vector of length m, representing a trajectory of atoms. If I let the vectors c_i and c_f be the vectors of length m giving the positions of the $m/3$ atoms in conformations C_i (starting) and C_f (ending), respectively. I determined these from my

database of motion, which has such data, chiefly derived from experimental sources such as x-ray crystallography.

If I now define a new vector $b = c_f - c_i$, or the differences between the ending and starting positions of each of the atoms of the structure along all three Cartesian axes, then I can find optimal v so that

$$\mathbf{A}\bar{\mathbf{v}} = \bar{\mathbf{b}} \tag{0.2}$$

In the normal case where $\dim \bar{\mathbf{v}} < 3N - 6$, this represents an over-determined system of linear equations, and may be solved by an appropriate numerical technique for solving linear least squares, such as Single Value Decomposition (SVD)[208] In practice, this is a very quick calculation, nearly instantaneous to the user.

Mode Concentration

Once $\bar{\mathbf{v}}$ has been computed, a statistic may be computed to summarize the information contained in the vector $\bar{\mathbf{v}}$:

$$\sum_{i=1}^{n} -|\mathbf{v}_i| \ln |\mathbf{v}_i| \tag{0.3}$$

This is the “mode concentration” of the vector.

In coding theory, information content is related to the negative entropy of a physical system. It specifies how much information is stored in a given set of numbers, and is typically used to compare the efficiencies of compression techniques. This statistic specifies how much movement is concentrated in any given mode, hence its name, “mode concentration.”

Overlap of Each Mode with Direction of Motion

For each motion pair, I computed the overlap (defined below) of each normal mode against the vectors giving the differences between the structures corresponding to the motions.

I defined the ‘overlap’ as the as the cosine of the angle between the mode and the direction of motion. ‘Average overlap’ is the ‘mean overlap’ over all atoms in the structure (i.e., $\frac{1}{n}\sum^{n}\frac{\bar{\mathbf{f}}\cdot\bar{\mathbf{b}}}{|\bar{\mathbf{f}}||\bar{\mathbf{b}}|}$, where $\bar{\mathbf{f}}$ and the individual atomic displacement vectors $\bar{\mathbf{b}}$ divided by the product of their lengths. The average absolute value of the cosine, $\frac{1}{n}\sum^{n}\frac{|\bar{\mathbf{f}}\cdot\bar{\mathbf{b}}|}{|\bar{\mathbf{f}}||\bar{\mathbf{b}}|}$ takes on larger values, but otherwise behaves similarly.) Larger average overlaps indicate that a given mode’s atomic displacement vectors are more similar in directionality to the vectors giving the differences between the PDB files. The mode of ‘maximum overlap’ is the

mode with the greatest 'average overlap' and most matches the protein motion's directionality.

S-correlation

A final means of quantifying the similarity between the displacement between the PDB structures and the normal modes is given by the formula

$$s = \sqrt{\sum_{j=1}^{n} j^2 O_j^2 - \left(\sum_{j=1}^{n} j O_j^2 \right)^2} \qquad (0.4)$$

where O_j is $\frac{\bar{\mathbf{b}} \bullet \bar{\mathbf{f}}_j}{|\bar{\mathbf{b}}||\bar{\mathbf{f}}_j|}$, the normalized dot product between some reference vector $\bar{\mathbf{b}}$ (in this case, the displacement between the PDB structures of the motion pair in question) and the $\bar{\mathbf{f}}_j$, in this case, the *j*th normal mode. This formula gives the s-correlation[197] between the reference vector and the set of normal mode displacement vectors, and may thus be used to provide a quantitative measure of the similarity in directionality between the displacements and the various normal modes. S-correlation was derived by analogy to the formula for the standard deviation of a probability distribution. Thus, the convention used to number the modes does not affect s-correlation in a meaningful way.

Other Analytic Measures

I calculated a number of other statistics (Tables 4.1 and 4.2A) similar to mode concentration and s-correlation. I defined the zeroth norm ("norm0") as simply the weight of the largest component (i.e., the largest value in the vector **v)**, the one norm as the average component, and the two norm as simply the Euclidean mean ("norm2") of the component's weight.

Results

Application of these Statistics to the Outlier Dataset

Figures 4.2 through 4.4 illustrate some properties of the above statistics on the outlier dataset. In particular, most often the low-frequency modes tend to be the ones with maximum overlap with the actual direction of motion (Figure 4.2). There is also a relationship between protein size (measured in number of residues), mode frequency, and maximum overlap (Figures 4.3 and 4.4).

Protein size (measured in number of residues) is negatively correlated to maximum overlap (Figure 4.3). Larger proteins have additional fragments that can be involved in a motion and, hence, additional degrees of freedom, decreasing the overlap between the tested normal modes and the observed motion. Maximum overlap decreases with protein size, but the effect is not dramatic, so it should be possible to design a standard analysis that works well on proteins comparable to those in my database. My results suggest (Figure 4.3) a statistical analysis standardized on the twenty lowest-frequency normal modes using the simplified Cα forcefield should be adequate even on the larger proteins in my database.

Increasing protein size (in residues) corresponds to modes of maximum overlap of decreasing frequency (Figure 4.4). A standard analysis concerned with larger proteins may need to consider more low-frequency normal modes than would suffice for smaller proteins. It would be desirable, given a protein of specific size, to deduce a frequency cut-off value, above which normal modes could be expected to be less useful in an analysis of motion. Analyses of individual proteins in the literature support the existence of such a cutoff[209,210] showing a slight dependency on the forcefield used. My results show that it is possible to determine such a cut-off frequency statistically from my database (Figure 4.4) and thereby empirically deduce a reasonable number of normal modes to use in a given type of analysis.

Comparison of mode concentration to other analytic measures

Results for the analytic statistics ("norm0", "norm1", and "norm2") were summarized. (Tables 4.1 and 4.2A) similar to mode concentration and s-correlation. These statistics, although superficially related to mode concentration, are not the same (Figure 4.5).

Validation of Mode Concentration with Feature Extraction Techniques

The physical and information theory basis of the mode concentration statistic suggested it might be useful in classification problems. Subsequent analysis via machine learning techniques (below) supports this.

Artificial intelligence feature analysis techniques provide one way of validating the usefulness of my mode concentration statistic. As described above, I created the manual and extended data sets as training sets to perform feature analysis. Using supervised machine learning techniques[211,212], I constructed two decision trees in S-Plus (MathSoft, Inc.) using the software's default parameters[211,213,214] (one for each of the two training sets) to classify the statistics in the morph server[170]. The use of S-Plus to construct decision trees from a specific training data set is a straightforward operation.

Decision trees attempt to partition the examples in the training set based on the values of individual statistics (Figure 4.6). In the actual decision tree, each statistic used in the classification decision appears in at least one branch junction. Features more relevant to the classification problem tend to appear earlier in the decision-making process, corresponding to a higher-level branch in the trees. By recording the depth any statistic first appears, decision trees may be used for feature analysis (Table 4.3). Mode concentration ranks prominently with a low depth, indicating that it appears high in the tree and is therefore useful for classifying motions.

Using appropriate, simple physical and mathematical concepts (normal mode analysis, singular value decomposition) I postulated several statistics (mode concentration and the various analytic norm measures) and confirmed my initial hypotheses using artificial intelligence techniques. These culled the morph server's[170] output of 36 physically-motivated statistics down to a set of nine "essential" statistics that proved most useful in this particular classification problem (Table 4.3), which agree roughly with my own sense

of the statistics most related to motion size. Similar databases of heterogeneous biological statistics may be "distilled" from a larger body of experimental data with these and similar techniques. In this case, the automatic classification features of the decision trees are only a side benefit. Feature analysis confirmed my earlier intuition that mode concentration can be useful for classifying motions.

Web and Database Integration

I used the results of my decision tree analysis (Table 4.3) to improve the ordering and presentation of statistics in Macromolecular Motions Database web reports (http://www.molmovdb.org). In addition, a new web tool (Figure 4.7) on this site graphically depicts output from the normal mode analysis as well as older experimental information. Users may perform analyses using the new tool by submitting a motion to the morph server[103]; the tools appear as options in the analysis menus below the results for the completed morph.

The new data from normal mode analysis has been integrated into both the Macromolecular Motions Database and the Partslist Database (http://www.partslist.org) as well[215]. (The Partslist Database is described in Appendix A; the Macromolecular Motions Database and the Partslist Database will eventually be merged.) This allows comparison by fold of motion and other data by a number of techniques, including regression analysis. Interactive users can test a number of statistics for correlation against the new data, as well as identify outlying folds that do not maintain the normal regression

well as identify outlying folds that do not maintain the normal regression pattern by mouse over.

Discussion

Applying Machine Learning Techniques to Heterogeneous Biological Database Problems

The Database of Macromolecular Motions is in some sense unique in that it provides a collection of heterogeneous statistics attempting to describe, in different ways, a single biological phenomena (a protein motion.) Heterogeneous databases of this sort tend to be rare in the sciences for a number of reasons, most notably: 1) easily conceptualized phenomena that are nevertheless complex enough that they can only be formally characterized through scores of statistics 2) When such datasets do occur the researchers tend to have a firmer grasp of the statistics (e.g., the statistics are gathered via surveys rather than gather these statistics by processing atomic coordinates from PDB files).

Artificial intelligence techniques may be applied to such databases to append additional, useful statistics to such heterogeneous databases, “distill” a database down to a set of “essential” statistics, as well as construct automatic classifiers. This has practical applications; pharmaceutical companies might mine existing biological databases to generate a refined, heterogeneous database describing potential drug targets within a statistical

framework. Artificial intelligence techniques can be used to extract key features and empirically assess the validity of new statistical models.

Conclusions

I have developed a framework that allows for a statistical study, in combination with my Database of Macromolecular Motions, of the importance of normal mode vibrations in biologically significant macromolecular motions. A statistic calculated from my analysis of normal mode displacements, mode concentration, is corroborated by feature selection corroborates as a useful statistic in classification. Feature selection techniques can be used to "summarize" databases of experimentally derived statistics into an especially salient set of "essential" statistics.

Examining the relationship between the aggregate directionality of the normal modes and structures' conformational change through a statistic such as mode concentration can be used to classify the motion ("fragment", "domain", or "subunit"). Normal modes have already been used[200] to identify dynamic protein domains. An analysis of the distribution of low-frequency normal mode trajectories should provide information about the type of protein motion and size of the domains involved in the motion. My data empirically supports earlier results[209] that analysis of only a small number of low-frequency modes should suffice for analysis of proteins comparable to those in my database. The database

can be used to determine statistically the cut-off for normal modes computed using different forcefields.

In addition to being made available through the Macromolecular Motions Database, my new data sets are integrated into the external Partslist database[215]. I have provided additional web tools associated with this chapter that allow molecular biologists to perform flexibility analysis on structures with putative motions, thereby identify key residues involved in the motion, and compare the results with similar analysis on the over 4,000 new motions now available in the database, as well as browse these motions by PDB ID and fold family.

Tables

Table 4.1: Definitions Table.

This table lists the various data sources used in this paper, giving the location of each, along with a brief explanation of its use or importance. It also defines key statistics and other terms used in subsequent tables as well as in the text of the paper.

TERM	Definition or URL Location
Macromolecular Motions Database	http://bioinfo.mbb.yale.edu/MolMovDB Used for classification and annotation of motions in outlier database
SCOP Database	http://scop.mrc-lmb.cam.ac.uk/scop/ Used for classification and annotation of motions via SCOP extension technique.
Wilson et al. set	As shown in Figure 1, a set of 30,000 of SCOP identifier pairs was constructed for Wilson CA, Kreychman J, and Gerstein M (2000), *J Mol Biol* 297: 233-49. This was then separated into two sets: the 30,000 pair "Wilson et al." set used in that paper, and the "Full Outlier Set" (described immediately below), which I use in this text. See the caption to Figure 1 for more information.
Full Outlier Set	Text file http://bioinfo.mbb.yale.edu/molmovdb/datasets/outliers.txt Pairs of proteins (SCOP domains) whose structural similarity score was more than two standard deviations above the mean structural similarity for their sequence similarity. See the caption to Figure 1 for more information on the construction of this set.
Workable Outlier Set	This is the subset of the full outlier set on which both morph server processing and normal mode analysis were successful. It consists of 3,814 motion pairs. http://bioinfo.mbb.yale.edu/molmovdb/datasets/workableoutliers.txt
Manual Training Set	This is the training set that was produced by examining the SCOP domains in the outlier set for matches against PDB IDs in the set of manually classified motions in the Database of Macromolecular (Gerstein and Krebs (1998) Nuc. Acid. Res., 26(18):4280). Matches received the same classification as in the database, which were determined by manual examination of the scientific literature. Thus, confidence in the accuracy of these classifications is high.
Extended Training Set	The outlier set was searched for pairs that shared the same SCOP fold family as pairs classified in the Manual Training Set; these then received an identical classification. I found empirically that, because proteins which share the same SCOP fold often share similar mechanisms, proteins with the same SCOP fold have a high probability of undergoing similar conformation change and, hence, sharing the same motion size classification. Consequently, these classifications should be accurate but are less reliable than the classifications in the Manual Training Set.
Classified Set	This is simply the entire workable outlier set (minus those already classified in the extended training set) run through the automatic classifier defined by the decision tree which I produced when I analyzed the extended training set.

TERM	Definition or URL Location (con't)
Mode Concentration	This is discussed extensively in the text. It is a simple measure of how much the protein's motion is concentrated into any single low-frequency normal mode.
#CAatoms	Number of C-alpha atoms in the protein
Residuals	This is the Euclidean length of the residual difference between the atomic displacements between protein pairs and the SVD fit of the normal modes to the atomic displacements (in Angstroms)
Norm0	Maximum Value of the SVD displacement vector (unitless)
Norm1	Mean of the SVD displacement vector (unitless)
Norm2	Root-mean-square of the SVD displacement vector (unitless)
Frequency	The frequency in relative units of the normal mode with the highest SVD coefficient.
Ranking Overlap	Rank of the normal mode with the largest overlap (unitless). Overlap is defined in the caption to Figure 2.
Maximum Overlap	Value of the largest overlap (unitless quantity). Overlap is defined in the caption to Figure 2.
Size of 2^{nd} Core	This is the number of residues in the 2^{nd} core (the 2ndCoreCAs key in the database). This is typically related to the size of the protein, although in poorly matches protein pairs the number can be less.
Trimmed RMS	This is the trimmed RMS score, as defined in Wilson CA, Kreychman J, and Gerstein M (2000), *J Mol Biol* 297: 233-49 and Gerstein and Krebs (1998) Nuc. Acid. Res., 26(18):4280.
Maximum CA Movement	This is the largest movement (in Angstroms) of any residue during the course of the motion, as computed by the Morph Server.
Number of Atoms	This is the number of atoms in the protein as computed by the Morph Server. (Atoms in non-standard amino acids are excluded.) This is a measure of the size of the protein.
Energy of Frames	The Morph Server computes energies for the various intermediate structures. These show a strong relationship to the sequence similarity between the two structures, and are indicator of how "good" a given morph is. The relationship of intermediate energies (energy of 4^{th} frame, for example) with endpoint frames (energy of 8^{th} frame, for example) can sometimes provide a rough sense of activation energies.
Translation	In hinge motions, the approximate translation (in Angstroms) the moving domains undergoes in the course of the motion, as automatically computed by the morph server. (This number is also computed for non-hinge motions, where it is less meaningful.)
Hinge Rotation	In hinge motions, the rotation (in degrees) of the moving domain around the screw axis in the course of the motion, as automatically computed by the morph server. (This number tends to be small in non-hinge motions.)
Number of Hinges	The number of putative hinges, or flexible linkages involved in the motion, as determined by the Morph Server
Traditional RMS	This is simply the traditional RMS score between the domains.
Rank of Norm0 Mode	This is a software index that identifies the normal mode contributing the most to the motion as computed within my SVD framework. (The same normal mode that sets norm 0.)

Table 4.2A: New Statistics Added to Morph Server

This gives a summary of new statistics added to morph server. This table presents mean, standard deviation, minimum, maximum, and median values for the new statistics that were added to the database following normal mode analysis of approximately 3,800 motion pairs in the database. The statistics are defined in Table 4.1.

key	#CAatoms	Residuals	Norm1	Norm2	Frequency	Ranking Overlap	Maximum Overlap
mean	220	480	-0.001	540	3.1	2.7	0.0031
std. dev.	110	660	0.051	360	0.89	3.6	0.005
minimum	39	0.23	-0.14	15	4.2E-08	0	4.7E-5
maximum	1000	8800	0.15	2700	8.6	19	0.11
median	210	330	0.00093	520	3.1	1	0.0017

Table 4.2B: Training Set Statistics

This table compares the percentages and absolute counts of domain, fragment, and sub-unit motions in each of the classified, extended, and manual training sets. Definitions of the different sets in the header are given in the text as well as Table 4.1. "Count" gives the number of times the particular motion size classification (Domain, Fragment, and Subunit) occurs in that dataset. "Percent" is the percentage out of the total number ("Total") of domain, fragment, and subunit motions in the dataset. The two columns on the left for the auto-classified set ("count" and "percent") represent a prediction made by an auto-classifier; the remaining columns represent observations.

	Predicted		**Observed**			
Motion Size	**Classified Set**		**Extended set**		**Manual Set**	
	Count	***Percent***	Count	***Percent***	Count	***Percent***
Domain	2165	*95%*	1549	*93%*	180	*73%*
Fragment	94	*4%*	107	*6%*	50	*20%*
Subunit	14	*1%*	14	*1%*	15	*6%*
Totals	2273	*100%*	1670	*100%*	245	*100%*

Table 4.3: Automatic Ranking of Statistics

This table indicates the earliest depth of the supervised machine learning decision tree each statistic first occurs, thus quantifying the relevance of each statistic to the particular motion property at hand ("fragment", "domain", or "subunit" motion, in this case).

Database Statistic	Depth in Tree Built upon Extended Set	Depth in Tree Built upon Manual Set
Size of 2nd Core	1	1
Trimmed RMS	3	2
Maximum CA Movement	5	2
Number of Atoms	4	3
Mode Concentration	6	4
Energy of 2nd frame	6	4
Translation	4	5
Hinge Rotation (Degrees)	4	6
Number of Hinges		6
Energy of 3rd frame		6
Norm0 (maximum value)	5	9
Energy of 9th frame	3	
Number of Residues	5	
Frequency	5	
Residuals	6	
Norm1 (average norm)	6	
Rank of Norm0 Mode	7	
Traditional RMS	8	
Norm2 (Euclidean norm)	8	
Energy of 4th frame	9	
Energy of 9th frame	9	
Energy of 8th frame	13	

Figures

Figure 4.1: Construction of Full Outlier Set

The crosses on this page illustrate motion pairs plotted in terms of RMS structure alignment scores against sequence percent identity for the 30,000 SCOP domain pairs Wilson et al.[201] identified from the PDB. Data points were binned into one-percent wide bins, and the mean RMS and standard deviation in each one-percent bin was computed. As described in Wilson et al.,[201] points more than two standard deviations above the mean were removed from the original 30,000 pair dataset (red crosses) and used to compose the full outlier set (green crosses), which ultimately consisted of 4,400 such pairs.

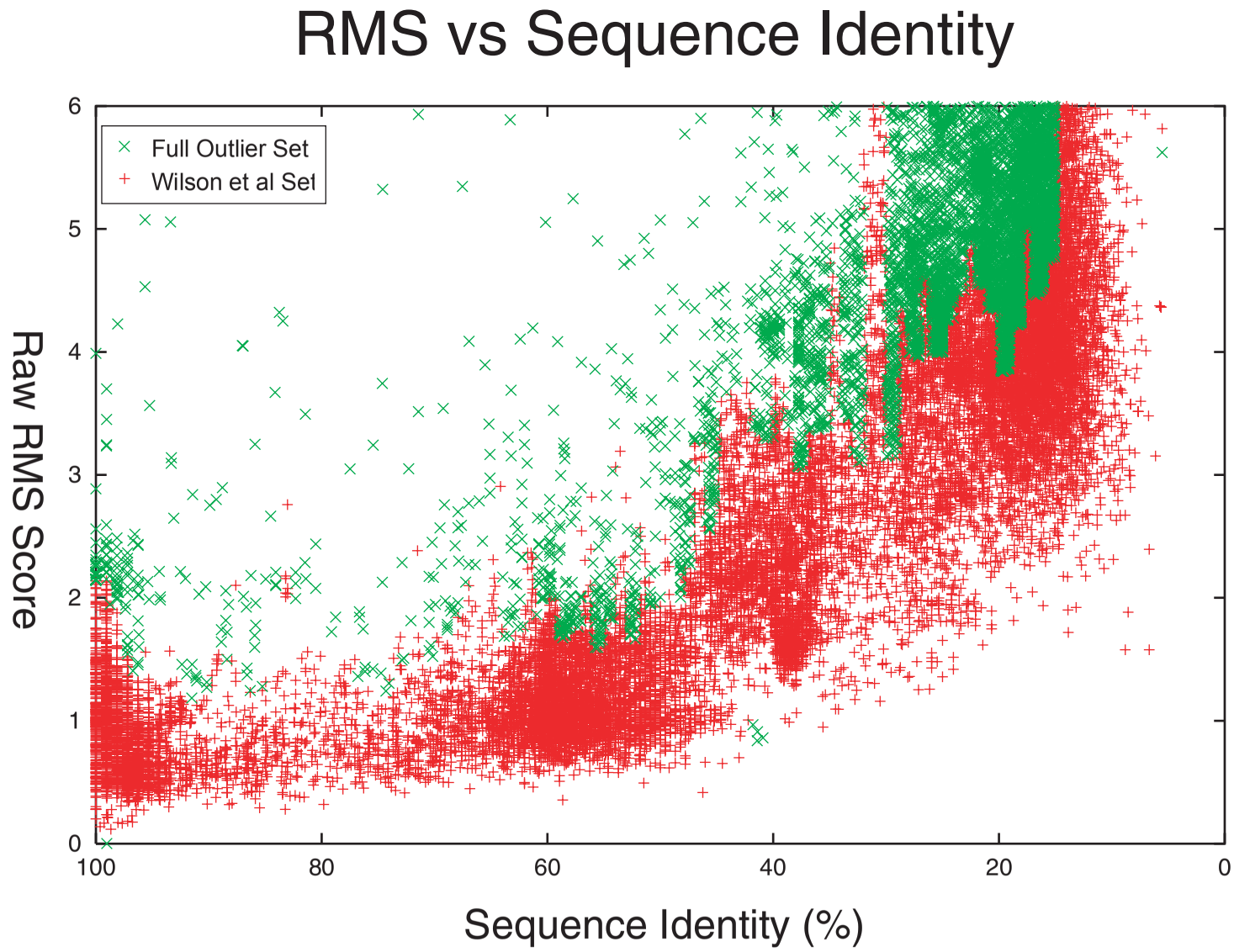

RMS vs Sequence Identity
Full Outlier Set
Wilson et al Set
Raw RMS Score
Sequence Identity (%)
0
1
2
3
4
5
6
100
80
60
40
20
0

Figure 4.2: Histogram of Greatest Overlap

My software places the twenty lowest-frequency normal modes in an array, thereby assigning each normal mode an index, from zero to nineteen. Increasing index numbers identify higher-frequency normal modes. I computed the overlap of each normal mode and recorded the index of the normal mode of greatest overlap. I plotted the number of times each index had greatest overlap in this histogram.

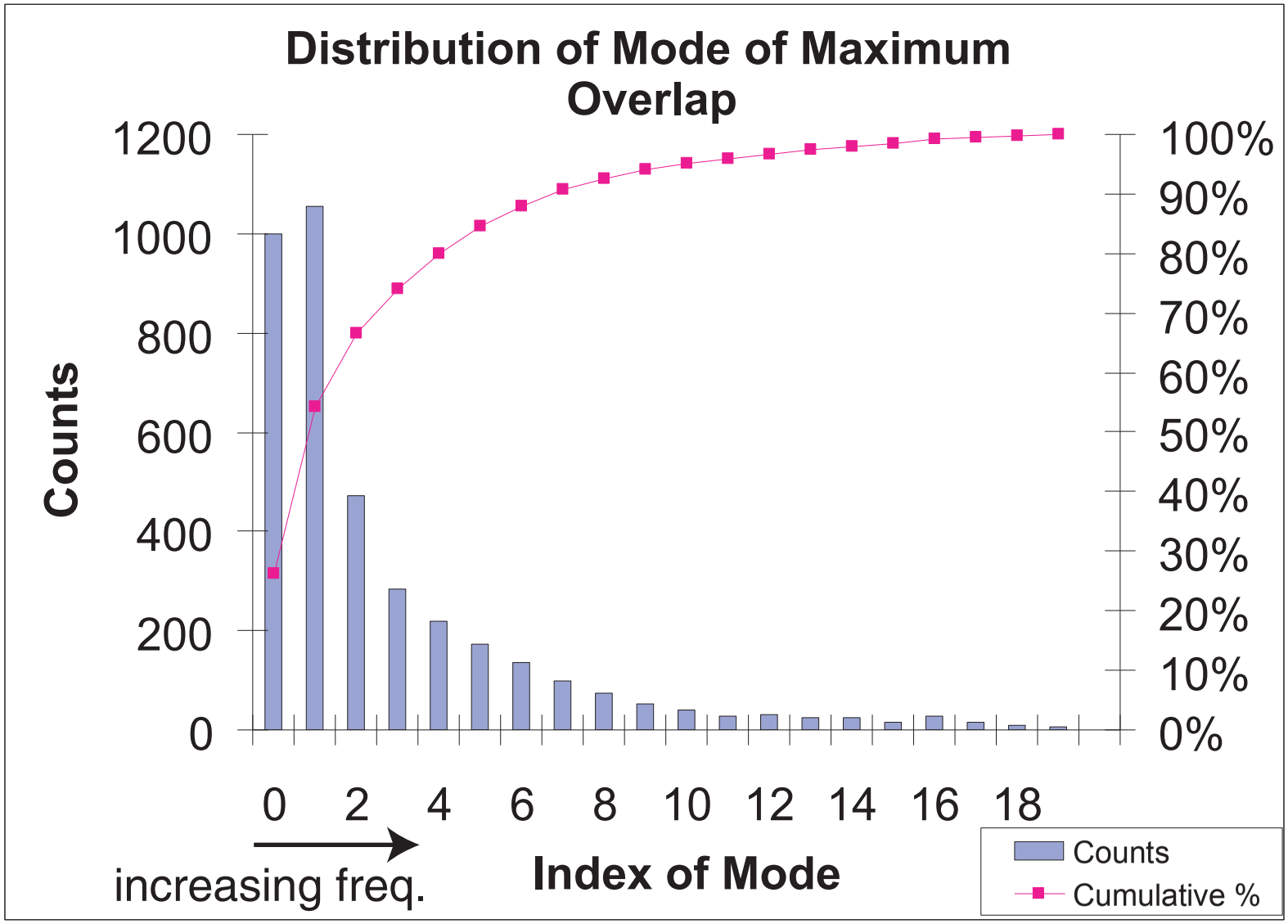

Distribution of Mode of Maximum Overlap
Counts
1200
1000
800
600
400
200
0
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0
2
4
6
8
10
12
14
16
18
increasing freq.
Index of Mode
Counts
Cumulative %

Figure 4.3: Relationship between protein size and maximum overlap.

To make the effect clearer, the y-values were binned into groups of 15 residues. The mean and standard deviation were computed for the values in each bin, with the results plotted. Each heavy horizontal bar indicates the mean in each bin, while the vertical bars indicate two standard deviations above and below the mean.

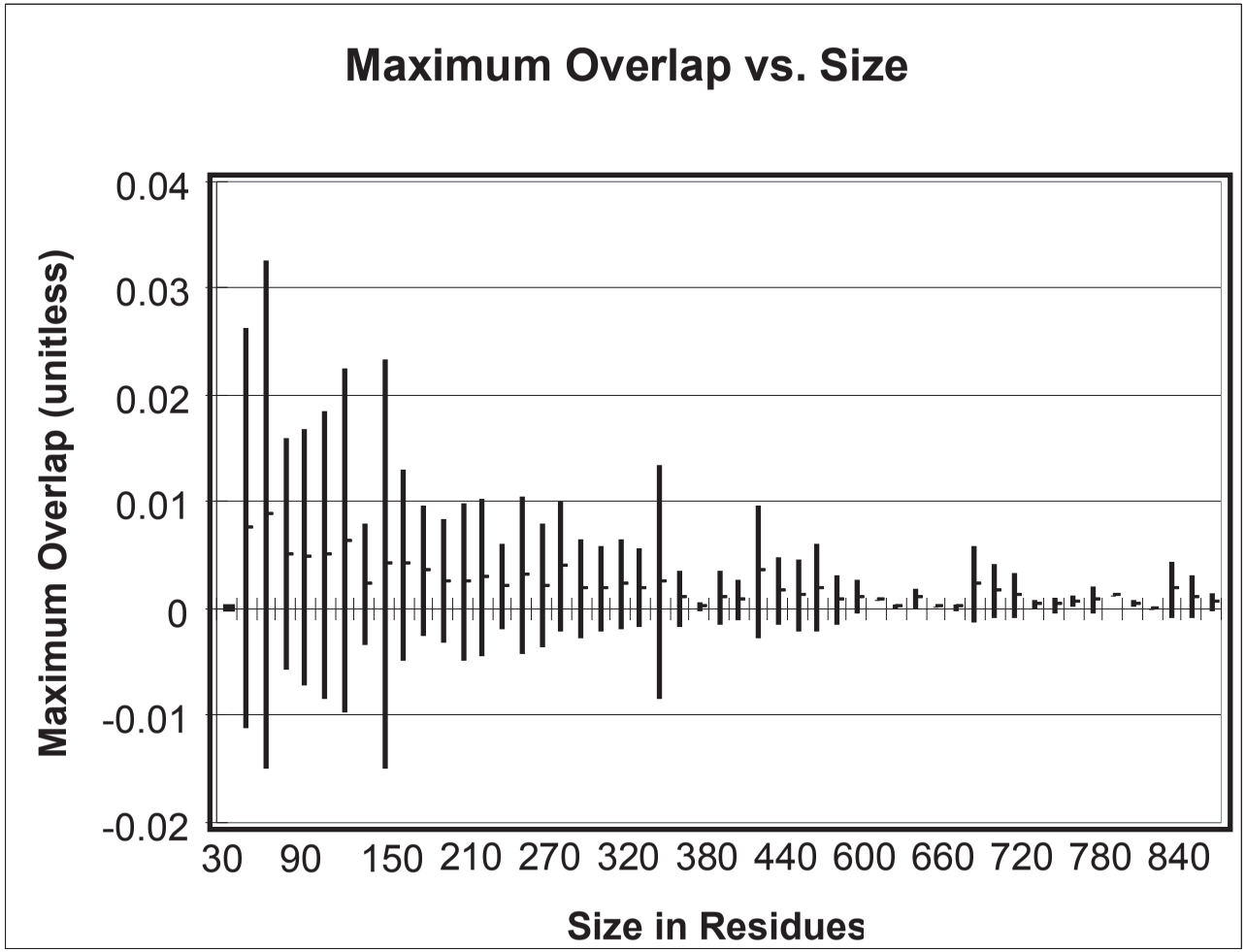

Maximum Overlap vs. Size
Maximum Overlap (unitless)
0.04
0.03
0.02
0.01
0
-0.01
-0.02
30
90
150
210
270
320
380
440
600
660
720
780
840
Size in Residues

Figure 4.4: Negative correlation between the frequency of the mode of maximum overlap and protein size.

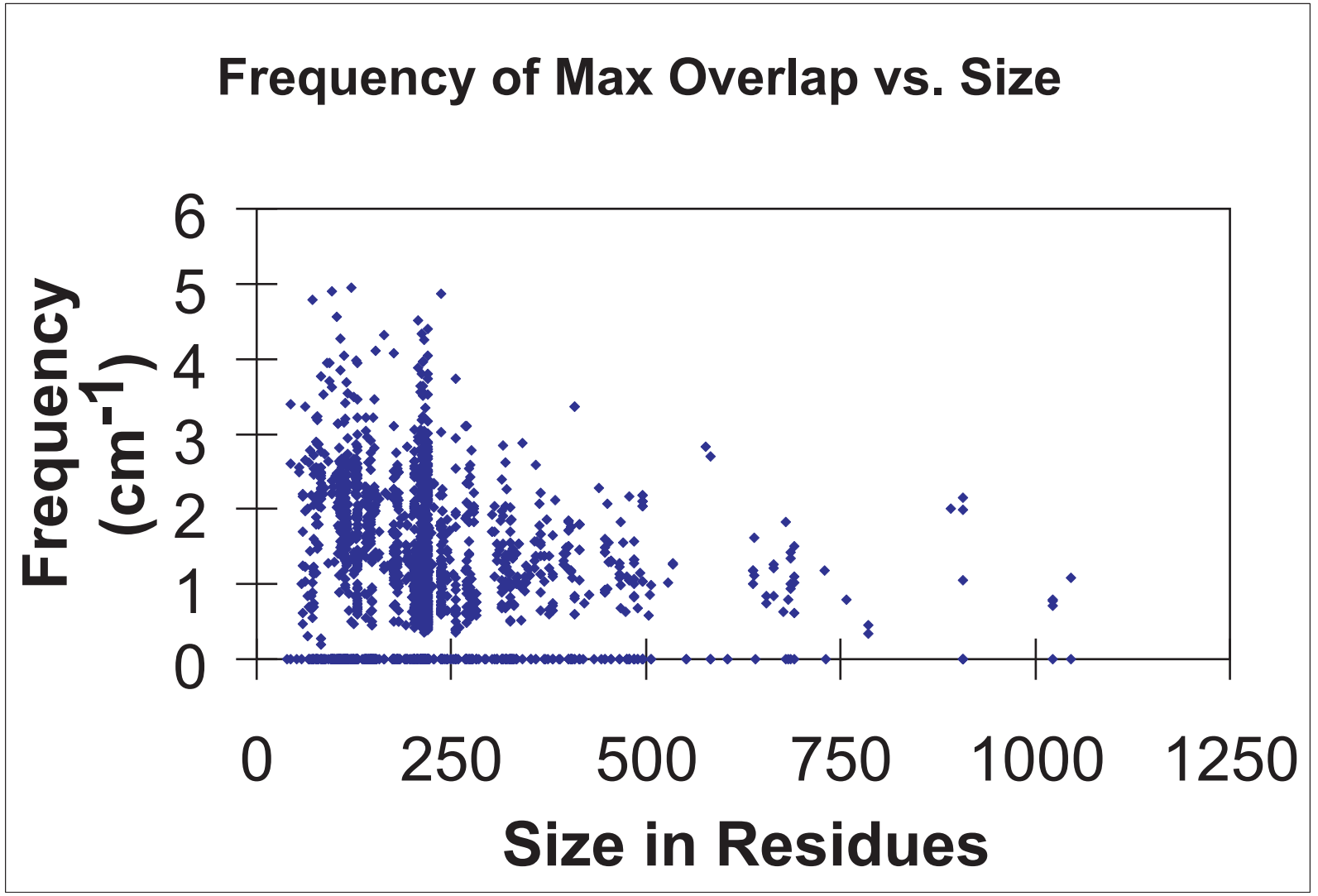

Frequency of Max Overlap vs. Size
Frequency (cm-1)
6
5
4
3
2
1
0
0
250
500
750
1000
1250
Size in Residues

Figure 4.5: Relationship between mode concentration and norm0 (concentration of motion in the mode with greatest concentration).

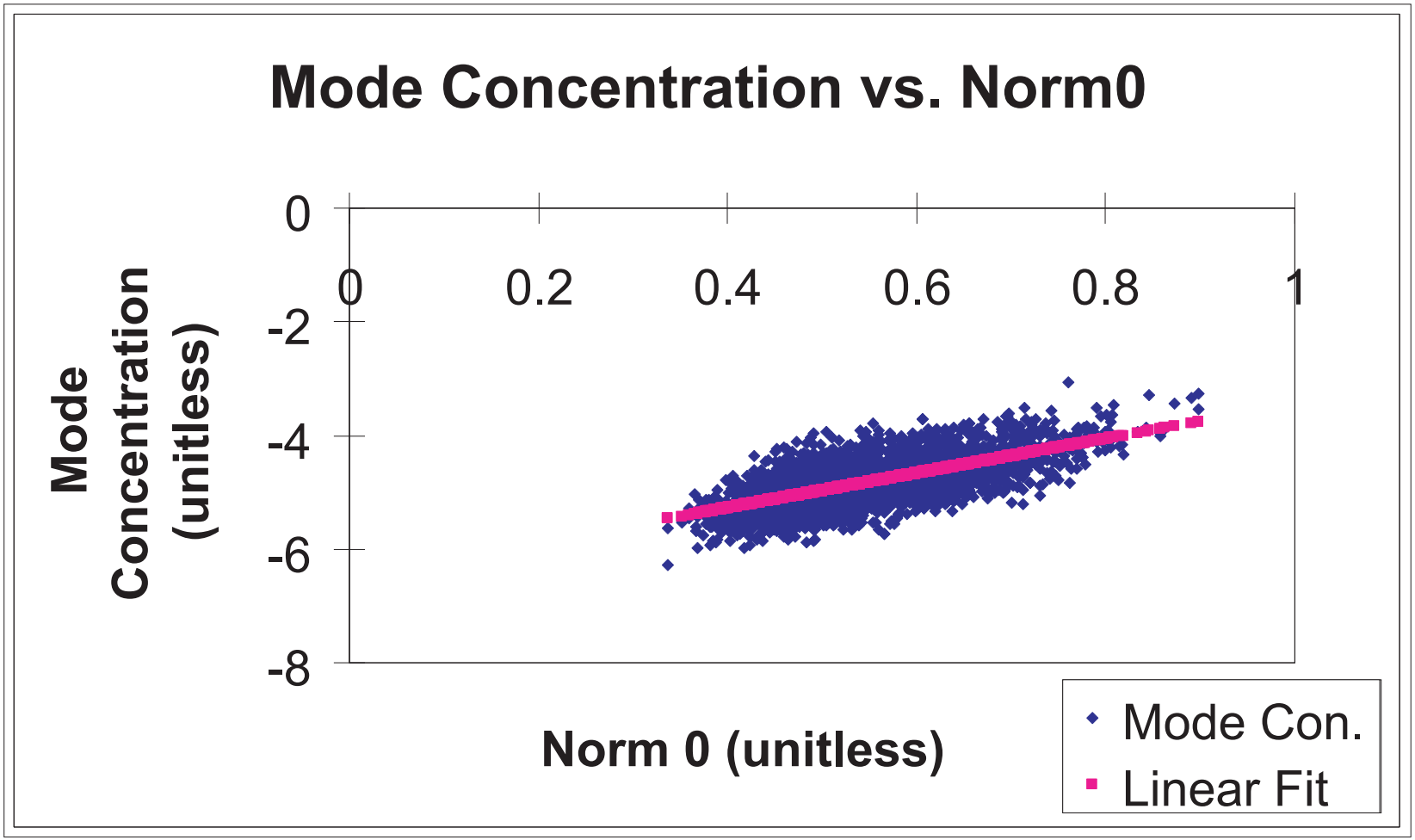

Mode Concentration vs. Norm0
Mode Concentration (unitless)
0
-2
-4
-6
-8
0
0.2
0.4
0.6
0.8
1
Norm 0 (unitless)
Mode Con.
Linear Fit

Figure 4.6: Decision Tree Concepts.

Two decision trees (not shown here) were generated by S-Plus (MathSoft, Inc.) using default parameters from the 245-element manual training set and the 1,670-element extend training set (defined in Table 4.1). These trees classify motions as “fragment”, “domain”, or “subunit”. The decision tree associated with the extended training set defined an automatic classifier (implemented in Perl by examination of the tree) that produced the “classified set.”

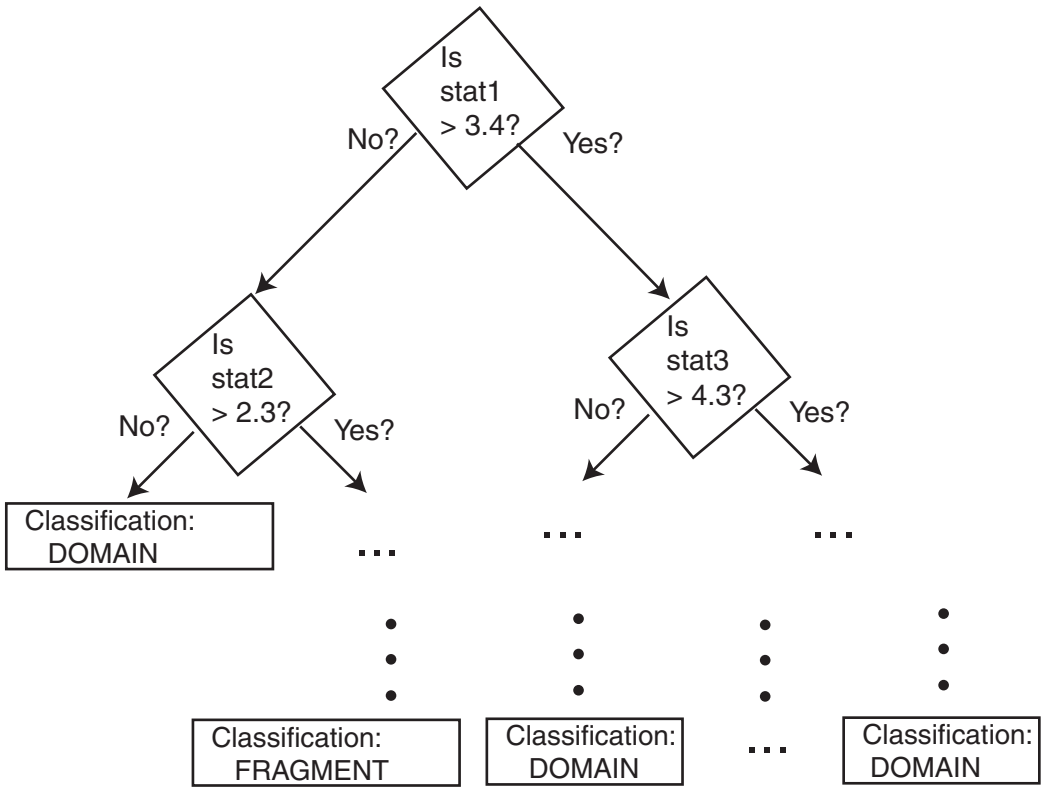
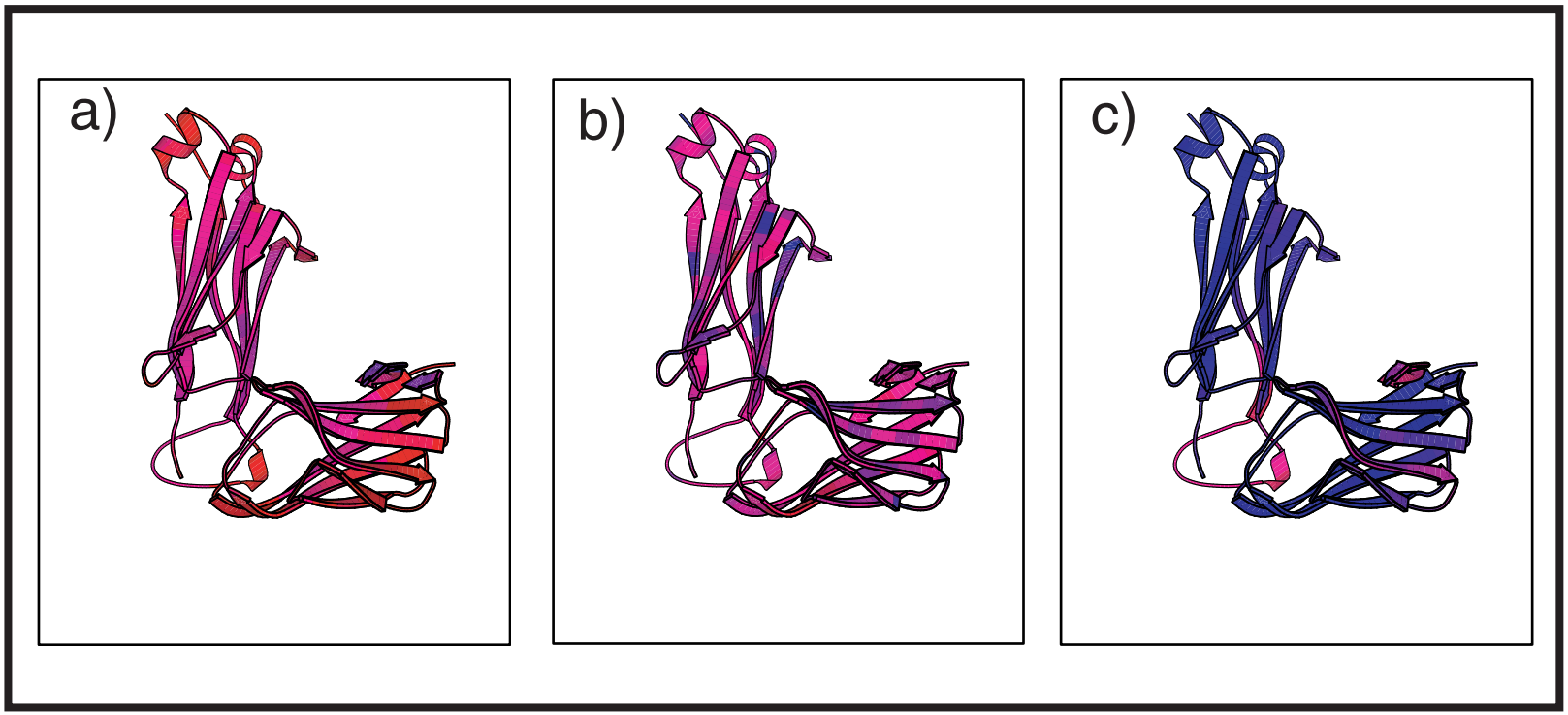

Is
stat1
> 3.4?
No?
Yes?
Is
stat2
> 2.3?
No?
Yes?
Is
stat3
> 4.3?
No?
Yes?
Classification:
DOMAIN
Classification:
FRAGMENT
Classification:
DOMAIN
Classification:
DOMAIN

Figure 4.7: New Web Tools

Output of new set of Web tools associated with normal mode analysis that the user may request on any protein for which a PDB structure file is available. The URL for this server is http://www.molmovdb.org; these features may accessed by browsing to a specific movie and selecting one of these analyses from the menu. Panel B performs a normal mode flexibility analysis on the structure. Regions that are more flexible are colored in red, while less flexible regions are colored in blue. Panel A gives similar information, using experimental temperature factors supplied in the PDB file, if available. Panel C, shows the parts of the protein that actually move, as calculated from comparison of the starting and ending PDB structures for the motion. Areas that move are colored in red, while areas that remain stationary are colored in blue. The user may compare these three panels to deduce structural information. Hinge locations involved in the motion may be deduced, as these are highly flexible regions (as identified by panels A and B) located near the moving domains (show in red in panel C).


a)
b)
c)

Chapter 5: Conclusion

I have developed a database of Macromolecular Motions and associated suite of software tools that attempts to characterize protein and nucleic acid motions within a database and statistical framework. My work is summarized in Appendix E.

Sustaining the Database

Now that the effort has been made to establish the Database of Macromolecular Motions, it is considerably less labor-intensive to maintain the database, keeping it up-to-date and useful to the scientific community that has grown accustom to its presence. In this way, the database differs from other scientific works, in that it is never really completely finished, but always growing a little to keep up the latest advances and the latest new thinking in the various disciplines it touches. The database is more like a library, largely static, but always changing a little, constantly requiring some maintenance to keep the library in good working order, although never as much as that which was required to construct the collections in the first place. For these reason, although databases, like libraries, can be built by individuals, they are always maintained (in at least some respect) by organizations, so that responsibility for their continued existence and accessibility does not rest on a single, mortal individual but rather on some sort of more permanent, institutional structure. It would therefore be irresponsible not to conclude by mentioning that continued care of the database rests with the members Gerstein laboratory. The current plan is to merge the Macromolecular Motions Database with the Partslist Database. A group of half-a-dozen or more individuals will now be responsible for the day-to-day operations of

the increasingly complex database, with Dr. Vadim Alexandrov slated to become the custodian of the portions of Parslist derived from the motions database once its current and sole custodian, the author, leaves the group. In this way, the database will continue to remain a valuable scientific resource, at least for the foreseeable future.

Only the Beginning

The Database of Macromolecular Motions represents a first attempt to systematically understand biologically macromolecules as moving parts compromising some important mechanical function that may be compared and studied (through phylogeny trees); copied, modified, and redesigned (through the nascent field of protein engineering); or jammed or interfered with (through rational drug design). Because we have only begun to conceptualize proteins as "parts" with an important motion, our efforts are necessarily crude, like the devices of the early Greeks. Scientists have recently conceptualized GroEL as "a two-stroke engine," in which the two halves of GroEL allosterically bind ATP at opposite points during their cycle. They used language directly borrowed from the early days of engineering. With time, our understanding of the important biological parts will grow, and we will develop greater finesse as we conceptualize their mechanical motions, moving away from crude descriptions such "two-stroke engine" and developing a more sophistical mathematical understanding of how amino acid sequence and the thermodynamical properties of the GroEL engine, so that we understand how to change the GroEL engine's stroke and timing the same we understand how to change these properties of a real, macroscopic engine. We are somewhere between Hero, the ancient Greek

who built a primitive steam engine from a tea kettle for amusement, and James Watt, whose improvements made steam power practical. The next steps are, surely, the most interesting.

Appendix A: PartsList: a web-based system for dynamically ranking protein folds based on disparate attributes, including whole-genome expression and interaction information

Introduction

As the number of protein folds is quite limited, a mode of analysis that will be increasingly common in the future, especially with the advent of structural genomics, is to survey and re-survey the finite parts list of folds from an expanding number of perspectives. This chapter, originally published in *Nucleic Acid Research*[215], describes how Dr. Qian Qian has developed a new resource (in collaboration with a number co-authors including Prof. Mark Gerstein, Brad Stenger, Cyrus A. Wilson, J. Lin, Ronald Jansen, Sarah A. Teichmann, and myself) called PartsList, that lets one dynamically perform these comparative fold surveys. In addition to currently containing data derived from the Macromolecular Motions Database, the PartsList and Macromolecular Motions Databases will eventually become merged into one complex database, hence its inclusion as an appendix in this work. PartsList is available on the web at http://bioinfo.mbb.yale.edu/partslist and http://www.partslist.org. The system is based on the existing fold classifications and functions as a form of companion annotation for them, providing "global views" of many already completed fold surveys. The central idea in the system is that of comparison

through ranking; PartsList will rank the ~420 folds based on more than 180 attributes. These include: (i) occurrence in a number of completely sequenced genomes (e.g. it will show the most common folds in the worm vs. yeast); (ii) occurrence in the structure databank (e.g. most common folds in the PDB); (iii) both absolute and relative gene expression information (e.g. most changing folds in expression over the cell cycle); (iv) protein-protein interactions, based on experimental data in yeast and comprehensive PDB surveys (e.g. most interacting fold); (v) sensitivity to inserted transposons; (vi) the number of functions associated with the fold (e.g. most multi-functional folds); (vii) amino acid composition (e.g. most Cys-rich folds); (viii) protein motions (e.g. most mobile folds); and (ix) the level of similarity based on a comprehensive set of structural alignments (e.g. most structurally variable folds). The integration of whole-genome expression and protein-protein interaction data with structural information is a particularly novel feature of his system. He provides three ways of visualizing the rankings: a profiler emphasizing the progression of high and low ranks across many pre-selected attributes, a dynamic comparer for custom comparisons, and a numerical rankings correlator. These allow one to directly compare very different attributes of a fold (e.g. expression level, genome occurrence, and maximum motion) in the uniform numerical format of ranks. This uniform framework, in turn, highlights the way that the frequency of many of the attributes falls off with approximate power-law behavior (i.e. according to V^{-b}, for attribute value V and constant exponent b), with a few folds having large values and most having small values.

Background

Protein folds can be considered the most basic molecular parts. There are a very limited number of them in biology. Currently, about 500 are known, and it is believed that there may be no more than a few thousand in total[216-218]. This number is considerably less than the number of genes in complex, multicellular organisms (>10,000 for multicellular organisms[219]). Consequently, folds provide a valuable way of simplifying and making manageable complex genomic information. In addition, folds are useful for studying the relationships between evolutionarily distant organisms since, in making comparisons, structure is more conserved than sequence or function.

In a general sense, how should one approach the analysis of molecular parts? A simple analogy to mechanical parts may be useful in this regard. Given the "parts" from a number of devices (e.g. a car, a bicycle, and a plane) one might like to know which ones are shared by all and which are unique (say, wings for a plane). Furthermore, one might want to know which are common, generic parts and which are more specialized. Finally, one might like to organize the parts by a number of standardized attributes (e.g. the most flexible parts, the parts with the most functions, and the biggest parts). PartsList aims to provide answers to simple questions such as these for the domain of protein folds.

Properties related to protein folds can be divided into those that are "intrinsic" versus "extrinsic". Intrinsic information concerns an individual fold itself—e.g. its sequence, 3D structure, and function—while "extrinsic" information relates to a fold in the context of

all other folds—e.g. its occurrence in many genomes and expression level *in relation* to that for other folds. Web-based search tools already provide intrinsic information about protein structures in the form of reports about individual structures. Valuable examples include the PDB Structure Explorer[171], PDBsum[220], and my MMDB[221]. However, current resources lack the ability to fully present extrinsic information.

Likewise, while there are many databases storing information related to individual organisms (e.g. SGD, MIPS and FlyBase[222-224]), comparative genomics (PEDANT and COGs[223,225]), gene expression (GEO, the Gene Expression Omnibus at the NCBI, and ExpressDB[226]), and protein-protein interactions (DIP and BIND[227,228]), none of these integrates gene sequences, protein interactions, expression levels and other attributes with structure. (However, it should be mentioned that the Sacc3D module of SGD and PEDANT do tabulate the occurrence of folds in genomes.)

PartsList is arranged somewhat differently from most other biological resources. In a usual database (e.g. GenBank[229]) the number of entries increases as the database develops, while each entry has a fairly fixed number of attributes to describe it. In contrast, PartsList is envisioned to have a relatively stable number of entries, i.e. the finite list of protein folds, while the attributes that describe each entry are expected to increase considerably. In the current version of PartsList the properties for a protein fold include: amino acid composition, alignment information, fold occurrences in various genomes, statistics related to motions, absolute expression levels of yeast in different experiments, relative expression ratios for yeast, worm, and *E. coli* in various conditions, information

on protein-protein interactions (based on whole genome yeast interaction data and databank surveys), and sensitivity of the genes associated with the fold to inserted transposons.

One reason to build the database is to compare protein folds in a rich context and in a unified way. This was achieved through ranking. This allows users to directly compare very different attributes of a fold in a uniform numerical format. The rankings can be visualized in three ways: a profiler emphasizing the progression of high and low ranks across many pre-selected attributes, a rankings comparer for custom comparisons, and a numerical rankings correlator. This can help users gain insight into the functions of protein folds in the context of the whole genome. His system makes it very easy to answer questions like: “What is the most common fold in the worm as compared to *E. coli*?” “What is the most highly expressed fold in yeast and how does this compare to the fold that changes most in expression level during the cell-cycle?” And "which fold has the most protein-protein interactions in the PDB and is it highly ranked in terms of protein motions?"

One of the strengths of the uniform numerical system of ranks in PartsList is that it puts everything into a common framework so that one can see hidden similarities in the occurrence of parts ordered according to many different attributes. In particular, as is described below, he found that the frequency of many of the attributes falls off according to a power-law distribution (i.e. according to V^{-b}, for attribute value V and a constant b), with a few folds having large attribute values and most having small values. For instance, there are only a few folds that occur many times in the yeast genome, and most only oc-

cur once or twice. Likewise, most folds only have a few functions associated with them, but there are a few "Swiss-army-knife" folds that are associated with many distinct functions. Similar power-law-like expressions have been found to apply in a variety of other situations relating to proteins—for instance, in the occurrence of oligo-peptide words[230-232], in the frequency of transmembrane helices[233] and sequence families with given size[234], and in the structure of biological networks, with a few nodes having many connections and most have only a few[235,236].

PartsList is built on top of the Structural Classification of Proteins (SCOP)[105] fold classification and acts as an accompanying annotation to this system. SCOP is divided into a hierarchy of five levels: class, fold, superfamily, family and protein. The "parts" in his system can be either SCOP folds or superfamilies. However, sometimes for ease of expression "folds" in this chapter often refers to both "folds and/or superfamilies." We currently use 420 folds and 610 superfamilies in PartsList. Each is represented by a representative domain, which is also the key for each entry of protein fold.

While Dr. Qian and collaborators chose to use the SCOP classification, he could equally well have based the system on the other existing fold classifications, e.g. CATH[237], FSSP[238], or VAST[239,240]. Moreover, for most attributes, Dr. Qian could also have developed his system around non-structural classifications of protein parts—e.g. Pfam[241], Blocks[242], or SMART[243]. However, basing it around actual structural folds has the advantage that each part is more precisely and physically defined.

Attributes that can be ranked: Information in the system

Currently the attributes for each entry (i.e. protein fold) can be separated into several main categories: statistical information from a comprehensive set of structural alignments, amino-acid composition information, fold occurrences in various genomes, expression levels in different experiments, protein interactions, macromolecular motion, transposon sensitivity and miscellaneous.

Dr. Qian and collaborators have developed a formalism for expressing each of the attributes, which is described in Table A.1. In the table the term *PART* refers to either fold or superfamily, depending on which of these is being ranked. Essentially, Dr. Qian has a database of attributes where each attribute is given a standardized description and associated with a precise reference. In the following, some main categories of attributes are described.

Genome Occurrence

The data in this category reveal fold occurrences in 20 different genomes, including 4 archaea, 2 eukaryotes, and 16 bacteria; (additional details online).

The data were obtained in the following fashion: Once a library of folds has been constructed, representative sequences can be extracted[244]. Then one can use these to search genomes by comparing each representative sequence against the genomes using the standard pairwise comparison programs, FASTA[245] and BLAST[246] and well-established thresholds[247].

Alternatively, one can build up profiles by running each representative sequence against PDB with PSI-Blast and then comparing these profiles against each of the genomes. This later procedure is more sensitive than pairwise comparison and relatively efficient once the profiles are made up. However, in doing large-scale surveys one has to be conscious of the potential biases introduced due to the profiles being more sensitive for larger families, which often results in the big families getting even bigger.

After the structure assignment, it becomes easy to enumerate how often a fold or structure feature occurs in a given genome or organism. Detailed information can be found in[233,248-250]. This pools assignments from previous work[251,252].

Alignment

Number of Structures. Dr. Qian and collaborators (including myself) did a comprehensive set of structural alignments of structures in the PDB structure databank[143,144,201]. The number of structures and aligned pairs used in these comparisons, which are based around Astral[244], give approximate measures of the occurrence of folds in the PDB. Comparison of these values to those for genome occurrence provides a measure of how biased the composition of the PDB is[253].

Sequence Diversity. The scores from the alignments indicate the sequence diversity between the related structures within folds or superfamilies, in terms of percent sequence identity and a sequence-based P-value. P-values are useful measures of statistical signifi-

cance of the similarity calculation. A P-value is the probability that one can obtain the same or better alignment score from a randomly composed alignment. A smaller P-value is less likely to have been obtained by chance than a larger P-value. Large P-values close to 1.0 indicate that the similarity is characteristically random and thus insignificant.

Structural Diversity. Dr. Qian and collaborators (including myself) also gave analogous measures of the diversity of the structures with a given fold, allowing one to rank folds by their degree of variability. Dr. Qian tabulates untrimmed and trimmed RMS, along with the structural P-value. RMS, root-mean-squared deviation in alpha carbon positions, has been the traditional statistic that gauges the divergence between two related structures. Smaller RMS scores indicate more closely related structures. However, sometimes a few ill-fitting atoms may significantly increase the RMS of structures known to be similar. To compensate for this Dr. Qian also reports a "trimmed" RMS for a conserved core structure (computed using the *wgkalign* algorithm, developed by myself), which is based on the better fitting half of the aligned alpha-carbons, and structural P-value, which compensates for other effects such as structure size. For details, see Wilson et al.[201].

Composition

This allows us to see which folds are most biased in composition of particular amino acids. Dr. Qian and collaborators use various levels of the Astral clustering of the SCOP sequences to arrive at the composition[244].

Expression

Three techniques are frequently used to obtain genome-wide gene expression data. They are Affymetrix oligonucleotide gene chips, SAGE (Serial Analysis of Gene Expression), and cDNA microarrays[254-256]. SAGE and, to some degree, gene chips measure the absolute expression levels (in units of mRNA transcripts per cell), while microarrays are used to obtain the expression level changes of a given ORF as the ratio to a reference state.

A main motivation for expression experiments is often to study protein function and to characterize the functions of unannotated genes. However, this does not preclude relating other attributes of proteins, such as their structure, to expression data. For instance, it may be that highly expressed protein folds share a number of characteristics, such as a particularly stable architecture or a composition biased in a certain way. Relating expression and structure involved matching the PDB structure database against the genome and then summing the expression levels of all ORFs containing the same fold. However, if one is trying to find genes expressed in a particular metabolic state, PartsList is not the right place to look.

Absolute. The absolute expression level data gives a good representation of highly expressed genes. All the experiments currently indexed by PartsList are for yeast. For each experiment, in addition to ranking based on the average expression level for a fold, Dr. Qian and collaborators also consider the composition in the transcriptome and the enrichment of this value relative to its composition in the genome. Transcriptome composition is the fractional composition of a fold (relative to that for other folds) in the mRNA

population. In other words, it is the composition of a fold in the genome weighted by the expression levels of each of the genes. The enrichment is the relative change between the composition of a fold in the genome and the transcriptome. Further details are provided in previous reports[257,258]. Dr. Qian reports values for experiments from a number of different labs[256,259-261] and a single reference set that merges and scales all the expression sets together.

Ratio. The expression ratio data shows the most actively changing genes over a period of time (e.g. cell cycle) or based on a change in states (e.g. healthy vs. diseased). Source data for expression ratios are the fluctuations in expression of a certain fold over a period of time (e.g. the cell cycle). These are measured in terms of standard deviations for a particular fold, which is calculated from the average of the expression ratio standard deviations for each gene that matches the fold structure.

Interactions

Information on protein-protein interactions is derived from surveys of the contacts in the PDB and the experiments in yeast.

PDB. To determine which domains interact with one another in the PDB entries indexed by SCOP (9,580 at the time of the analysis), the coordinates of each domain were parsed to check whether there are five or more contacts within 5 Å to another domain, as described in[262]. The distance of 5 Å was chosen, as this is a conservative threshold for interaction between two atoms, where the atoms are either Cα's or atoms in side-chains. The

5-contact threshold was chosen to make sure the contact between the domains was reasonably extensive. (In fact, the number of domains identified as contacting each other hardly changed for thresholds between 1 and 10 contacts and 3 to 6 Å distances).

Yeast. The interactions between structural domains in the yeast genome were obtained by assigning protein structures to the yeast proteins using PSI-BLAST and PDB-ISL as described in Teichmann et al[263,264]. Assigned structural domains contained within the same ORF that were adjacent within 30 amino acids were assumed to interact. (This is generally true of the domains in the PDB, with a few exceptions, such as domains in transcription factors like adjacent zinc fingers, or variable and constant immunoglobulin domains.) To derive intermolecular interactions in the yeast genome Dr. Qian and his collaborators combined three sets of protein-protein interactions: (i) the MIPS web pages on complexes and pairwise interactions (February 2000)[223], (ii) the global yeast-two-hybrid experiments by Uetz et al.[265] and (iii) large-scale yeast two-hybrid experiments by Ito et al.[266]. Out of all these pairwise interactions known for yeast ORFs, there is a limited set in which both partners are completely covered by one structural domain (to within 100 residues). This set of protein pairs was used to derive a further set of domain contacts in the yeast genome as described in[262].

Motions

Information on motions is from the Macromolecular Motions Database[16,170]. I considered a set of approximately 4400 motions automatically identified by examining the PDB and

a smaller, manually curated set of motions. For each fold I determined the number of entries in the motions database that are associated with it. Then over this set of motions I either averaged or took the maximum value of a number of relevant statistics describing the motion, i.e. the maximum Cα displacement in the motion, the overall rotation of the motion, and the energy difference between the start and endpoints of structures involved in the motion.

Transposon Sensitivity

Ross-MacDonald et al.[267] developed a procedure for randomly inserting transposons throughout the yeast genome. They investigated the phenotypes resulting from each insertion in 20 different growth conditions in comparison to wild-type growth. The experiment for each insertion in each condition was repeated several times. If the observed phenotype of the mutant deviates from the average wild-type phenotype, this could be either because of a real effect of the mutation on the cell or it could just a be typical variation of the phenotype of wild-type cells. Dr. Qian and collaborators developed a P-value score that measures the degree of confidence that the observed phenotype results from randomly changing wild-type cells. The negative logarithm of this P-value rises with the significance of the phenotype measurements and can be understood as the sensitivity of the cell to mutations in a particular gene. Dr. Qian and collaborators calculated a value for the transposon sensitivity for protein folds by geometrically averaging the P-values of the associated genes.

Miscellaneous

The miscellaneous section includes any information that does not fit into a major category. It includes: number of pseudogenes in worm associated with a fold[268], total number of functions and number of enzymatic functions associated with a fold[269], the average length of the sequence, and the year the domain structure was originally determined.

Errors

The above data, of course, have systematic and statistical errors. For some attributes Dr. Qian, Dr. Teichmann, Prof. Gerstein, Mr. Jansen, and I expect considerably smaller errors than others. For instance, Prof. Gerstein and I expect the numbers related to the sequence composition of different folds (e.g. the Ala composition) to be particularly accurate, since the only factors affecting these are errors in the underlying sequence of the protein and in the scop fold classification itself. In contrast, there is a considerable known rate of false positives associated with the global protein interaction experiments using the two-hybrid method[265,270], and this suggests statistics based on yeast interactions may be somewhat less accurate. Furthermore, the precise values for the rankings in PartsList are also contingent on the evolving contents of various databanks. Thus, over time as more structures are determined, one should expect statistics such as the most common folds in a particular genome to change somewhat. I first authored a very detailed discussion of the expected errors in the various quantities in PartsList; it is available on the web from the help section.

Ranking all the folds based on extrinsic information

The PartsList resource facilitates exploring extrinsic information by dynamically ranking protein folds in different contexts, such as genome and expression levels. Dr. Qian and collaborators provide three tools for visualizing the rankings: Comparer, Correlator, and Profiler. The overall structure of PartsList is schematically shown in Fig. A.1.

Comparer

The motivation behind Comparer is to allow one to rank folds according to a given attribute and then see the ranks associated with other attributes. The ranking attribute and the additional attributes are selected by the user. Figure A.2(a) shows an example. The most common folds in *E. coli* are shown alongside three other attributes: fold occurrence in yeast, fluctuation in expression level during the yeast cell cycle, and fluctuation in expression level in *E. coli* during heat shock. Which displayed attribute is used to rank the folds can be easily changed; in the example in Figure A.2(a) the report can be re-sorted based on the other three attributes by clicking on arrows.

Profiler

In principle, Profiler presents the same information as Comparer. However, it shows the progressing pattern for several pre-selected categories and is intended to give people an easy-to use interface that gives some simple views of the data. Figure A.2(b) shows an example that highlights the phylogenetic pattern of fold occurrence in 20 genomes.

Correlator

Correlator uses linear and rank correlation coefficients to measure the association between two selected attributes. The difference between these two types of correlation coefficients is that the former relates to the actual values while the latter relates to the ranks among the samples. The interpretation of the linear correlation coefficient can be completely meaningless if the joint probability distribution of the variables is too different from a binormal distribution. This is the reason for introducing the rank correlation coefficient. Correlator provides both coefficients for the selected quantities. In most cases, they are close. For example, the linear correlation coefficient and rank correlation coefficient for fold occurrence in genomes *A. fulgidus* and *M. jannaschii* (Aful and Mjan) are 0.88 and 0.77, respectively, while the corresponding coefficients for fold occurrence in *A. fulgidus* and *S. cerevisiae* (Scer) are 0.52 and 0.48, respectively. This is not surprising, as the first two genomes are both Archaeal, while in the second comparison one genome belongs to Archaea (Aful) and another to Eucarya (Scer). As one would expect, the fold occurrences for the more closely related genomes have a higher correlation.

In addition to the coefficients, Correlator displays a scatter plot to aid in visualizing the correlation between the selected fold attributes. Figure A.2(c) shows the scatter plot for the second example above: the correlation between occurrences in the *A. fulgidus* and *S. cerevisiae* genomes. One can easily observe that some folds appear frequently in Scer but seldom or never in *A. fulgidus*. By clicking on a point on the plot, one obtains detailed information about the corresponding fold. This kind of plot can reveal interesting folds

with certain relationships between attributes even though in some cases the overall correlation coefficients between the two attributes are almost zero (i.e. no correlation).

Power-Law Behavior of Many Disparate Attributes

Going back and forth between Correlator and Comparer allows one to see interesting relationships between disparate attributes of proteins. Figure A.3 illustrates a comparison of two attributes, functions and interactions. It shows a ranking of the folds that have the most interactions in the PDB in comparison to those that have the most functions. It is immediately apparent that there are only a few folds with large values of either attribute, i.e. many functions or interactions. Moreover, the most multi-functional folds also have the most distinct interactions with other folds, suggesting that a few a folds may function as general-purpose parts.

In fact, the uniform system of ranks in PartsList shows that "only a few folds having large values for an attribute" is a generally true statement for many of the disparate attributes catalogued by the system. Moreover, the falloff from high to low values for a given attribute often follows a power-law distribution. That is, the normalized frequency F that a number of distinct folds have a particular attribute value V follows a functional form like:

$$F(V) = a\, V^{b}$$

where a and b are constants. Note that *F(V)* is just the number of folds with an attribute value *V* divided by the total number of folds and that on a log-log plot this function becomes a straight line with slope -*b*. Often the attribute value *V* itself reflects the *occurrence* of a fold in a particular context—e.g. V could be the number of times a given fold occurs in a particular genome. Quantities that follow a power-law-like behavior are often said to have a form like that of Zipf's law, which often occurs in the analysis of word frequency in documents[271].

Thus far, this general conclusion is described in language sufficiently abstract to accommodate the many different types of attributes in PartsList. A few concrete examples will make the conclusion clearer. For instance, Dr. Qian and collaborators found that in genomes most folds occur only once while there are only a very few folds that occur many times. An illustration is shown in the upper panel of Fig. A.5 for *E. coli*. The x-axis is the number of times a particular fold occurs in the *E. coli* genome and the y-axis shows the number of distinct folds that have same occurrence. (This is normalized by dividing by the total number of folds so that the maximum value on y-axis is 100%.) From the log-log format of the plot, one can immediately see that the falloff obeys a power-law, with a few folds occurring many times and most only once or twice. The middle panel shows other attributes that display similar power-law-like behavior, including expression level in yeast, number of functions associated with a fold, and number of protein-protein interactions found in the PDB. Of course, not all attributes follow a power-law. The lower panel shows two of these less typical attributes: Asp composition in a fold and average number of residues involved in a motion.

One of the strengths of the uniform numerical system of ranks in PartsList is that it puts everything into a common framework so that one can see similarities across disparate attributes. Dr. Qian believes it would be difficult to see a common power-law behaviour for many aspects of protein structure without PartsList.

Traditional Single-Structure reports

In addition to the tools that compare and relate the extrinsic properties of protein folds, Dr. Qian provides traditional reports that are more focused on an individual structure.

Occurrence report. This allows users to see the number of times that a fold corresponding to the queried protein structure occurs in various genomes. This gives a phylogenetic profile of the occurrence of a particular fold in 20 genomes, similar in spirit to the fold patterns discussed earlier[233].

Function report. This summarizes the functional classification of the queried PDB structure. It merges a number of functional classifications, including FlyBase[224], ENZYME[272], GenProtEC[273] and MIPS[223]. His approach to functional classification is described in a number of previous publications[201,269]. In short, Dr. Qian used pairwise comparison to cross-reference the PDB domains against Swissprot. Depending on whether they had an Enzyme Commission number, Dr. Qian and collaborators were able to divide all entries into enzymes and nonenzymes, a division that represents the highest level in his classifi-

cation. (For the enzyme category, Dr. Qian only transferred Enzyme Commission numbers to those SCOP domains with a one-to-one match to a Swissprot enzyme.) In the absence of an EC-type classification for nonenzymes, Dr. Qian assigned functions to nonenzymatic SCOP domains according to Ashburner's original classification of Drosophila protein functions. This classification is derived from a controlled vocabulary of fly terms, is available on the web, and is loosely connected with the FLYBASE database[224]. It has recently been superceded by the GO functional classification[274]. MIPS and GenProtEC classifications to SCOP domains were assigned based on sequence comparisons to classified yeast and *E. coli* ORFs, respectively. The SCOP domain most closely matching each ORF classified in MIPS or GenProtEC was assigned the corresponding MIPS or GenProtEC function number. Only matches of 80% sequence identity or greater were considered.

Alignment report. This gives detailed information on structural alignments available between pairs of protein domains associated with a fold. A pair viewer is provided, which gives many key statistics about the alignment (e.g. RMS, sequence identity, number of fit atoms, etc.), in addition to a listing of the actual aligned residues. Both HTML and parseable text views are available.

Interaction report. This shows all the pairs of protein-protein interactions associated with a fold based on either the PDB survey or yeast genome data.

Rank report. This highlights the top-five and bottom-five ranked attributes associated with a fold. It also shows all attributes ordered by the rank they are given in that fold. It, thus, highlights for a particular fold the attributes with respect to which it most stands out. That is, it highlights the "outlier attributes" of each fold, the way each fold is most unique. The rank report could be used, for example, by a protein engineer interested in determining the unique properties of a structure he is working on.

PDB report. This summarizes all the information concerning a domain or a representative PDB structure. It includes: (i) a summary of the occurrence report; (ii) a summary of the alignments available for structures in the same superfamily and fold; (iii) a description of motions and motion-movies associated with the structure in the Macromolecular Motions database[16,170]; (iv) a summary of the merged functional classification; (v) a core structure, if available[275]; (vi) ranking tables of the queried structure in various datasets; and (vii) a summary of the interactions report. Figure A.4 shows a sample PDB report for structure 1AMA.

Fold report. This lists all the SCOP domains associated with the queried fold and provides information (similar to that in the PDB report) that is common to all -- i.e. genome occurrence, alignment report, and rankings.

Summary and Discussion

Dr. Qian and collaborators have developed a web-based system for dynamically ranking protein folds based on disparate attributes, including fold occurrence in various genomes, expression level, alignment statistics, protein-protein interactions, motion statistics, and transposon sensitivity. Three ranking tools are provided—Comparer, Profiler, and Correlator—which can help users to place one fold in context of all other ones. The uniform system of ranks employed by PartsList provides a good framework for comparing different experiments and gaining a broad perspective on the complexity of genomes.

Dr. Qian anticipates that PartsList will have a relatively stable number of entries (i.e. folds), while for each entry the attributes that describe it will increase over time. (In fact, when the Partslist interface was extended to the Macromolecular Motions Database, this was no longer strictly true—the Macromolecular Motions Database contains a large number of entries, and the interface had to be modified to handle a larger number of entries.) In the future as experiments yield new information, PartsList will include more and more attributes. In particular, Dr. Qian anticipates that much new expression information will be incorporated. Dr. Qian, his collaborators and I also plan to develop a form to allow automatic submission of new ranking attributes and to encourage people to submit any ranking information.

Figures and Tables

Table A.1: Attributes Ranked by Partslist

This table shows all the attributes ranked by PartsList. The formalism for specifying an attribute has two parts: an overall category, denoted by a single uppercase symbol, and some parameter choices, which are denoted by lower-case arguments to the first symbol. Some examples for folds will suffice to make this clear: G(aful) is genome occurrence of a particular fold in *A. fulgidus*; M(nhinges,goldstd) is the maximum value of the number of hinges statistic from surveying a set of motions in the gold-standard subset of the Macromolecular Motions Database, where this statistic is only calculated for the entries in the motions database that are associated with a particular fold; And I(pdball,inter) is the number of distinct types of protein-protein interactions found in a survey of the PDB, subject to the restriction that the interactions must be between folds on different chains.

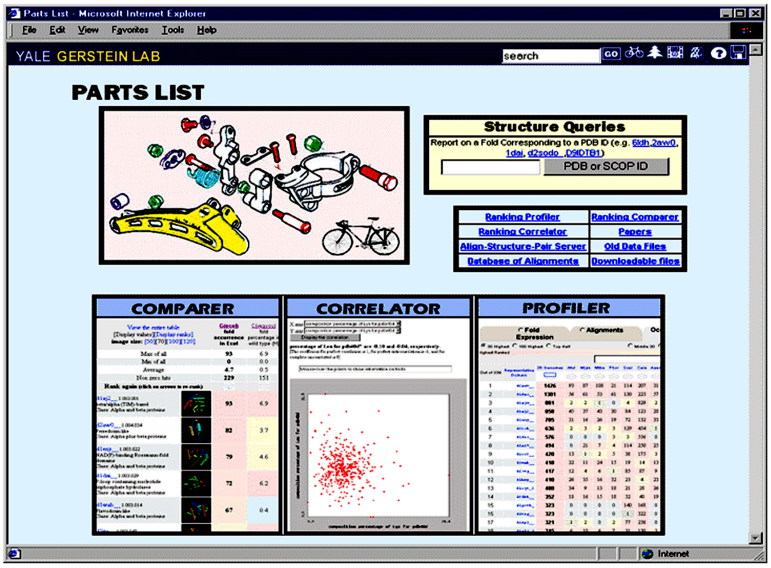
Category	Symbol	Definition of Symbol	Attributes in Category
Genome Occurrence	G(x)	Number of times a particular PART occurs in genome x. (These are based on PSI-blast comparisons between PDB and the genomes with an e-value cutoff in these comparisons of .0001.)	20
Expression	L(e)	average expression level over all genes that contain a this PART.	8
	C(e)	PART composition of the yeast transcriptome in expression level experiment e. This refers to the fraction of the mRNA population with this PART as opposed to all other parts. (This is only applicable to expression experiments, such as SAGE and GeneChips, that measure absolute mRNA levels in copies per cell.)	8
	E(e)	Transcriptome enrichment compared to genome in experiment e. (Transcriptome enrichment is defined as percentage difference of PART composition in the transcriptome and the genome. In symbols: E(e) = [C(e)-G(Scer)] / G(Scer) .)	8
	F(r)	Expression level fluctuation in experiment r. (This is the standard deviation in the expression ratio measurement R(i,t) over a timecourse, viz: $<(R(i,t)-<R(i,t)>)^2>$ where one averages over all times t and genes i that have a particular PART.	7
Alignments	V(f)	The number of aligned pairs in pair-set f.	2
	U(f)	RMS deviation in C atoms averaged over all alignments in pair-set f	2
	R(f)	Similar to U(f) for pair-set f but only the best fitting half of the atoms are included in the calculation	2
	S(f)	Average percentage identity between pairs of aligned proteins in pair-set f	2
	P(f)	Average sequence P-value for pair-set f	2
	Q(f)	Average structural P-value for pair-set f	2
Compositions	N(p)	The number of structures associated with a particular PART in dataset p.	2
	B(a,p)	Composition of amino acid a in a particular PART where one averages over all structures in dataset p associated with the PART	40
Motion	M(s,d)	The maximum value of statistic s derived from surveying set of motions d in the Macromolecular Motions Database for a particular PART, where s is only calculated from the entries in the database that are associated with the PART.	7
	A(s,d)	Similar to M(s,d) but now we take the average instead of the maximum.	7
Interaction	I(y,c)	For a given PART, the number of types of protein-protein interactions in interaction dataset y subject to the restriction c regarding whether or not the proteins are on the same chain. The number of interaction types is the number of distinctly different PARTs that interacts with a given PART.	24
	J(y,c)	For a given PART, the total number of types of interactions in interaction dataset y subject to the restriction c regarding whether or not the proteins are on the same chain. Here we show all interactions observed not just the number of distinct PART-PART interactions tabulated in I(y,c).	24
Transposon	T(b)	The sensitivity of the cell to a transposon inserted into genes containing a particular PART under different growth condition b. The sensitivity was indicated by negative logarithm of a P-value, which measures the degree to which the observations for one particular gene could have resulted from wild-type cells that randomly change their phenotype.	20
Miscelleneous	X(q)	Various miscellaneous ranks	5
Total			182

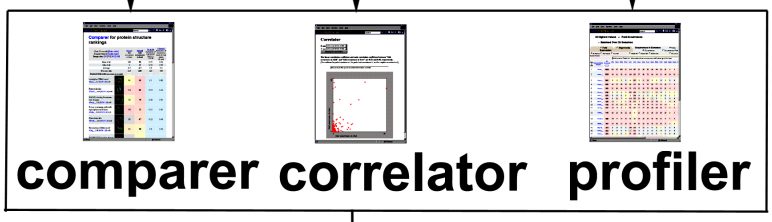
Attributes	Value	Description
Genome x =	aful	Archaeoglobus fulgidus
	mjan	Methanococcus jannaschii
	mthe	Methanobacterium thermoautotrophicum
	phor	Pyrococcus horikoshii
	scer	Saccharomyces cerevisiae
	cele	Caenorhabditis elegans
	aaeo	Aquifex aeolicus
	syne	Synechocystis sp.
	ecol	Escherichia coli
	bsub	Bacillus subtilis
	mtub	Mycobacterium tuberculosis
	hinf	Haemophilus influenzae Rd
	hpyl	Helicobacter pylor
	mgen	Mycoplasma genitalium
	mpne	Mycoplasma pneumoniae
	bbur	Borrelia burgdorferi
	tpal	Treponema pallidum
	ctra	Chlamydia trachomatis
	cpne	Chlamydia pneumoniae
	rpro	Rickettsia prowazekii
Absolute Expression Expt. e =	vegsam	GeneChip mRNA expression analysis of 6200 yeast ORFs under vegetative growth conditions.
	vegyou	GeneChip mRNA expression analysis of 5455 yeast ORFs under vegetative growth conditions.
	sage	mRNA expression analysis of 3788 yeast ORFs determined by Serial Analysis of Gene Expression.
	matea	GeneChip mRNA expression analysis of yeast mating type a strain grown on glucose.
	mateal	GeneChip mRNA expression analysis of yeast mating type alpha strain grown on glucose
	gal	GeneChip mRNA expression analysis of yeast mating type a strain grown on galactose
	heat	GeneChip mRNA analysis of yeast mating type a strain grown on glucose at 30 degree before a 39 degree heat shock.
	ref	Reference transcriptome. This is a scaling and merging of the above experiments.
Microarray Expt. r=	cdc28	cDNA microarray genome-wide characterization of mRNA transcript levels for CDC28 synchronized yeast cells during the cell cycle.
	cdc15	cDNA microarray genome-wide characterization of mRNA transcript levels for CDC15 synchronized yeast cells during the cell cycle.
	alpha	Analysis using cDNA microarrays of yeast mRNA levels after synchronization of cell cycle via alpha arrest factor
	diaux	Genome-wide cDNA microarray analysis of the temporal program of yeast mRNA expression accompanying the metabolic shift from fermentation to respiration
	spor	cDNA microarray genome-wide analysis to assay changes in gene expression during sporulation.
	heatec	cDNA microarray experiment and analysis on 4290 E.coli ORFs after exposure of the bacteria to heat shock.
	deve	Analysis of genome wide changes during successive larval stages using cDNA microarrays of ~12000 C. elegan ORFs.
Pair set f=	all	All pairs within a PART included in the calculations in Wilson et al. (For example, for fold rankings this would be the total number of pairs within a fold.)
	foldonly	A subset of the pair-set "all" that only includes pairs between structures that are in the same PART but different sub-PART. (If PART is fold, then sub-PART is superfamily; If PART is superfamily, then sub-PART is family.)
Amino Acid a=		Ala, Cys, Asp, Glu, Phe, Gly, His, Ile, Lys, Leu, Met, Asn, Pro, Gln, Arg, Ser, Thr, Val, Trp, Tyr.
Data set p=	pdb100	All structures within the fold (as defined by SCOP pdb100d)
	pdb40	Similar to pdb100 but now using a version of the PDB clustered at 40% similarity (as defined by SCOP pdb40d)

Attributes	Value	Description
Interaction type y=	pdball	Interactions for a PART are computed with all other PARTS in the PDB databank based on the distances between atoms in the coordinate files. Five or more contacts between atoms separated by less than 5 A was considered a valid PART-PART contact.
	pdba	A subset of "pdball". Interactions for a PART are computed just with all-alpha proteins (SCOP class 1) in the PDB.
	pdbb	Similar to "pdba" but now just with all-beta proteins (SCOP class 2).
	pdbab	Similar to "pdba" but now just with mixed helix-sheet proteins (SCOP class 3 and 4)
	scerall	Interactions for a PART are computed with all other PARTS based on the yeast two-hybrid experimental data. In particular, interactions between structural domains in the yeast genome were obtained by assigning protein structures to the yeast proteins. Structural domains contained within the same ORF that were within 30 amino acids were assumed to interact in an intramolecular fashion. To derive intermolecular interactions, we combined three sets of protein-protein interactions: (i) the MIPS web pages on complexes and pairwise interactions (February 2000)(9), (ii) the global yeast-two-hybrid experiments by Uetz et al. (45) and (iii) large-scale yeast two-hybrid experiments by Ito et al. (46). Out of all these pairwise interactions known for yeast ORFs, there is a limited set in which both partners are completely covered by one structural domain (to within 100 residues).
	scera	A subset of "scerall". Interactions for a PART are computed just with all-alpha proteins (SCOP class 1) in the yeast experiment.
	scerb	Similar to "scera" but now just with all-beta proteins (SCOP class 2).
	scerab	Similar to "scera" but now just with mixed helix-sheet proteins (SCOP class 3 and 4)
Interaction restriction c=	inter	The interaction must occur between PARTS in different chains
	intra	The interaction must occur between PARTS in the same chain.
	none	The union of "inter" and "intra". Interactions can occur in PARTS on the same or different chains.
Motion statistic s=	nresidue	Number of residues
	maxcadev	Maximal displacement of an C atom, in angstroms, of any residue during the motion (after fitting on the first core).
	rmsoverall	Overall RMS of two structures after they are superimposed by a sieve-fit technique. Note that they are larger than traditionally used RMS (details see ref.).
	nhinges	Number of hinges involved in the motion.
	kappa	The rotation (in degrees) around the screw axis necessary to superimpose two domains of motion.
	transe	Transition energy of the motion (maximum energy less minimum energy over the motion) (in kcal/mole).
	deltae	Absolute value of energy difference between the "starting" and "ending" conformations of a motion (in kcal/mole).
Motion dataset d=	goldstd	list of ~220 "gold-standard" manually curated motions
	auto	list of ~4000 conformational different proteins based on analyzing the SCOP database for similar proteins with large conformational differences (as measured by RMS) but close sequence similarity
Transposon conditions b=	caff	YPD + 8mM caffeine
	cyss	Cyclohexmide hypersensitivity: YPD + 0.08 gml^{-1} cycloheximide at 30^0C
	wr	White/red colour on YPD
	ypg	YPGlycerol
	calcs	Calcofluor hypersensitivity: YPD+12 gml^{-1} calcoluor at 30^0C
	hyg	YPD + 46 gml^{-1} hygromycin at 30^0C
	sds	YPD + 0.003%SDS
	bens	Benomyl hypersensitivity: YPD + 10 gml^{-1} benomyl
	bcip	YPD + 5-bromo-4-chloro-3-indolyl phosphate at 37^0C
	mb	YPD + 0.001% methylene blue at 30^0C
	benr	Benomyl resistance: YPD + 20 gml^{-1} benomyl
	ypd37	YPD at 37^0C
	egta	YPD + 2mM EGTA
	mms	YPD + 0.008% MMS
	hu	YPD + 75mM hydroxyurea
	ypd11	YPD at 11^0C
	calcr	Calcofluor resistance: YPD + 0.3 gml^{-1} calcofluor at 30^0C
	cycr	Cyclohexmide resistance: YPD + 0.3 gml^{-1} cycloheximide
	hhig	Hyperhaploid invasive growth mutants
	nacl	YPD + 0.9M NaCl
Misc. quantities q=	pseu	Number of pseudogenes in worm genome matching a particular PART
	func	Total number of functions associated with this PART. (In this survey all non-enzyme functions were lumped into a single category.)
	enz	Total number of enzymatic functions associated with this PART.
	size	Average length of a PART in the pdb40d clustering of the PDB.
	age	The year of the first structure that is part of the PART was determined.

Figure A.1: Overall Structure of Partslist

Three tools (Profiler, Comparer, and Correlator) provide an easy way to access and manipulate the display of the dataset. With these tools, users can isolate interesting folds and obtain fold reports about them. Further clicks take one to PDB report, which gives detailed information about an individual structural domain, including its genome occurrence, alignment information, molecular motions, functional annotation, interactions, and core structure.





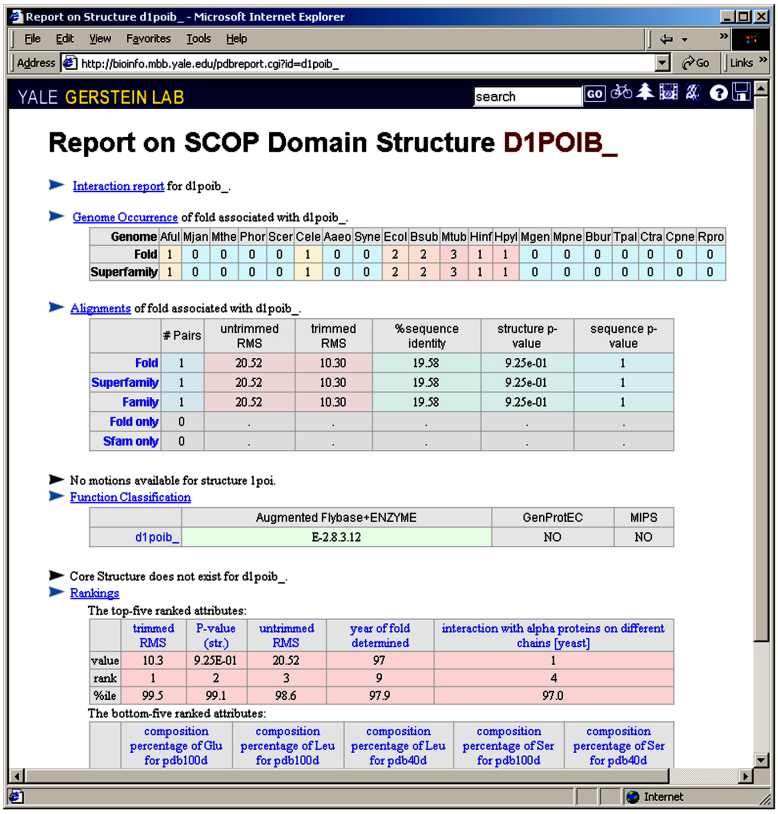
fold report

interaction

occurrence

motion

alignment



PDB report

core structure

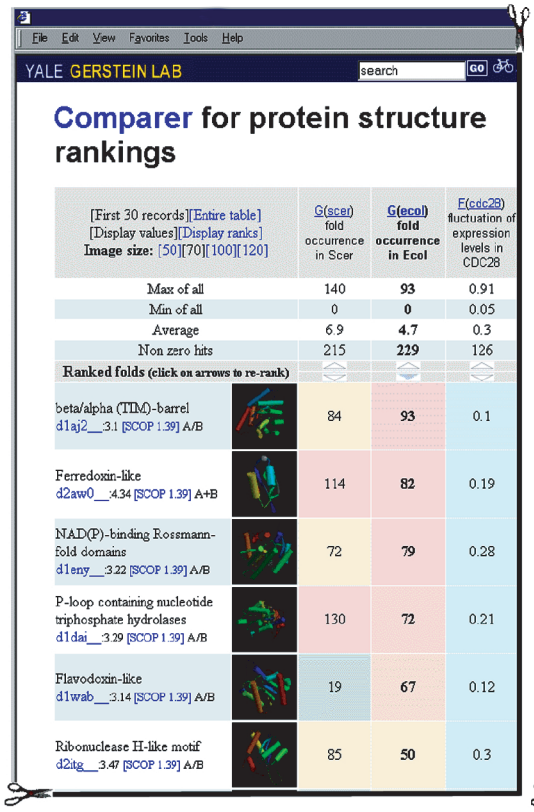
function

rank report

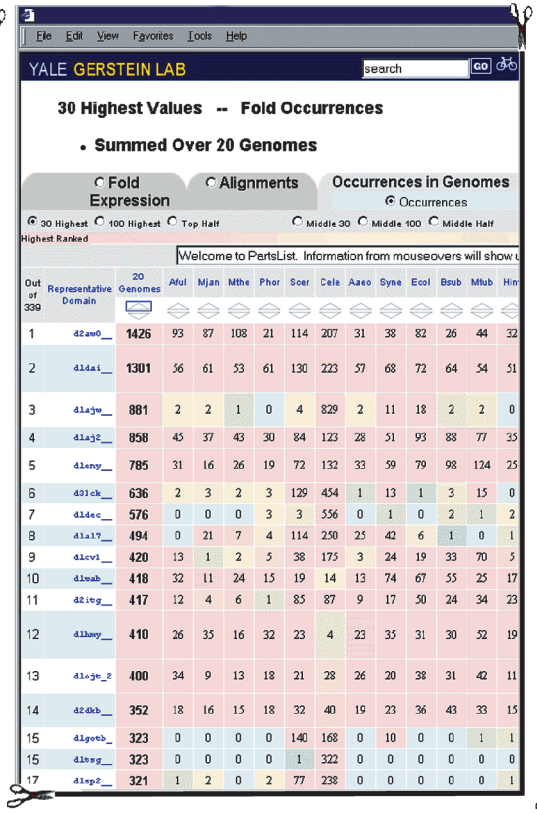
Figure A.2: Sample Displays

Sample displays. **(A)** a sample Comparer display: the four selected attributes are the fold genome occurrence in yeast, the analogous quantity for *E. coli*, fluctuation of expression level for CDC28 synchronized yeast cell during the cell cycle , and the corresponding values for *E. coli* to heat shock. (Using the nomenclature in Table A.1 these quantities are G(scer), G(ecol), F(cdc28), and F(heatec).) The folds are ranked in terms of fold occurrence in *E. coli* and the most common fold here is the TIM-barrel (represented by the SCOP domain d1aj2__). If one clicks the "Display ranks" button, the values in the cells will be replaced by the ranks in their respective columns. By clicking the "re-rank" arrows, one can also obtain other views by sorting on other attributes. **(B)** Shows the occurrences of folds in 20 genomes in Profiler. **(C)** Shows the correlation between the fold occurrences in the *A. fulgidus* and *S. cerevisiae* genomes (G(aful) and G(scer)) . Both linear and rank correlation coefficients are calculated. The linear correlation coefficient is defined as: $R = \frac{1}{N-1}\mathbf{X} \bullet \mathbf{Y}$, where $\mathbf{X}$ and $\mathbf{Y}$ are two vectors with N elements. Each element of the **X** vector is normalized thus: $X_i = \frac{X_i' - \overline{X}}{\sigma_x}$, where $\overline{X}$ and σ_x are the average and standard deviation of the values of the original data vector **X**', respectively. **Y** is normalized in a similar fashion. For two perfectly correlated datasets, $R = 1$, while for two completely uncorrelated datasets, $R = 0$. If X_i is replaced by its rank among all the other X_i in the sample (i.e., 1,2,3…,N), then one gets the rank correlation coefficient. A scatter plot is also shown to help in visualizing this correlation.

A. Comparer



B. Profiler



C. Correlator

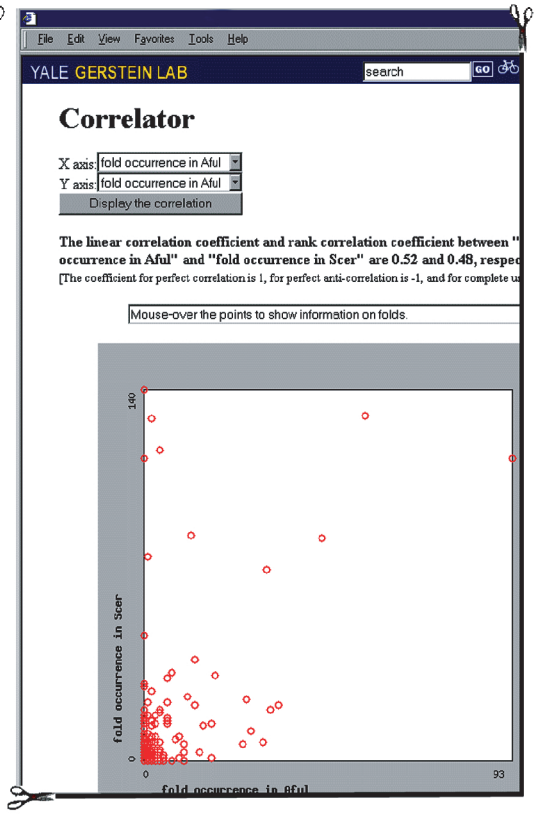
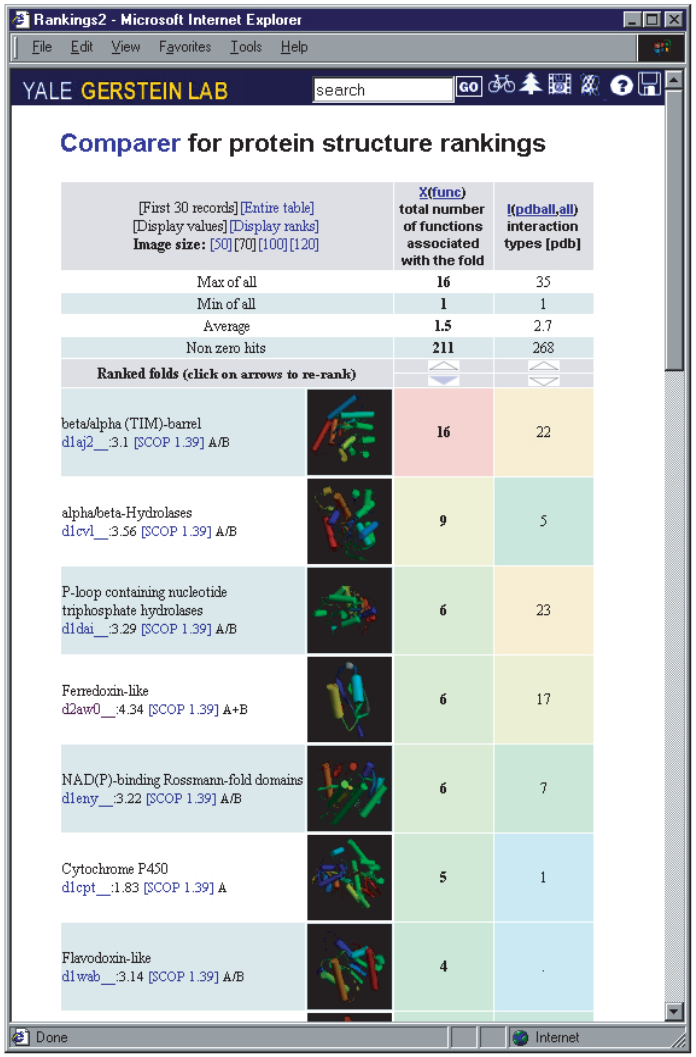


Figure A.3: Relations between functions and protein-protein interactions

The relation between the number of functions associated with a protein fold and the number of distinct protein-protein interactions it has (based on a survey of the PDB databank). These are X(func) and I(pdball,none) using the nomenclature in Table A.1. This relationship can be displayed both in Comparer (left) and Correlator (right).


Rankings2 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
YALE GERSTEIN LAB
search
GO
Comparer for protein structure rankings
[First 30 records] [Entire table]
[Display values] [Display ranks]
Image size: [50] [70] [100] [120]
X(func) total number of functions associated with the fold
I(pdball,all) interaction types [pdb]
Max of all 16 35
Min of all 1 1
Average 1.5 2.7
Non zero hits 211 268
Ranked folds (click on arrows to re-rank)
beta/alpha (TIM)-barrel d1aj2__:3.1 [SCOP 1.39] A/B 16 22
alpha/beta-Hydrolases d1cvl__:3.56 [SCOP 1.39] A/B 9 5
P-loop containing nucleotide triphosphate hydrolases d1dai__:3.29 [SCOP 1.39] A/B 6 23
Ferredoxin-like d2aw0__:4.34 [SCOP 1.39] A+B 6 17
NAD(P)-binding Rossmann-fold domains d1eny__:3.22 [SCOP 1.39] A/B 6 7
Cytochrome P450 d1cpt__:1.83 [SCOP 1.39] A 5 1
Flavodoxin-like d1wab__:3.14 [SCOP 1.39] A/B 4 .
Done
Internet

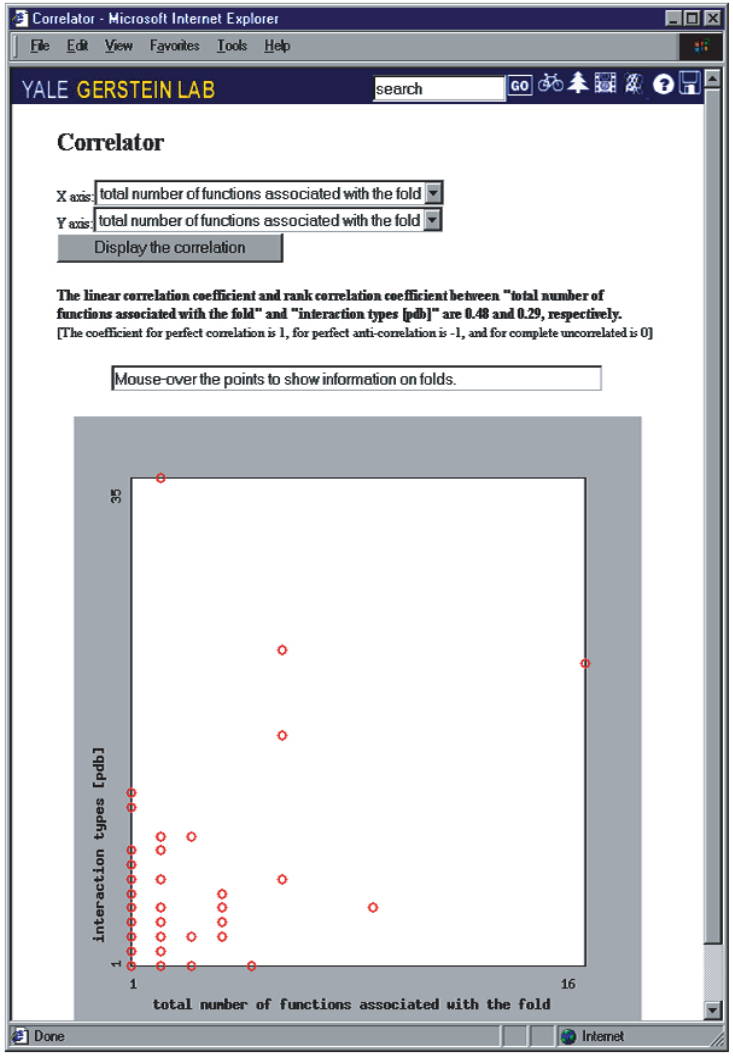
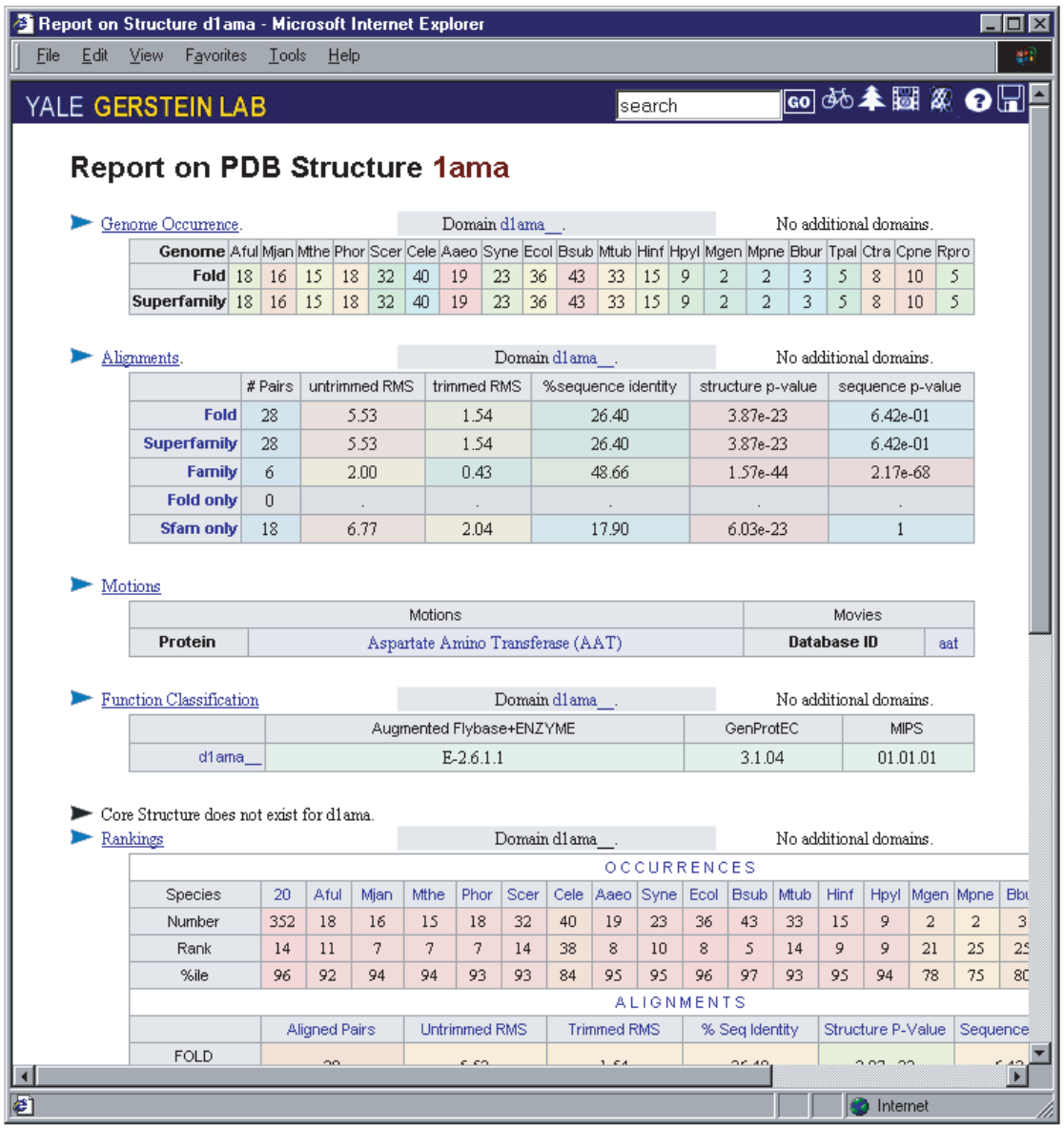

Correlator - Microsoft Internet Explorer
File Edit View Favorites Tools Help
YALE GERSTEIN LAB
search
GO
Correlator
X axis: total number of functions associated with the fold
Y axis: total number of functions associated with the fold
Display the correlation
The linear correlation coefficient and rank correlation coefficient between "total number of functions associated with the fold" and "interaction types [pdb]" are 0.48 and 0.29, respectively.
[The coefficient for perfect correlation is 1, for perfect anti-correlation is -1, and for complete uncorrelated is 0]
Mouse-over the points to show information on folds.
35
1
1
16
interaction types [pdb]
total number of functions associated with the fold
Done
Internet

Figure A.4: A sample PDB report for structure 1AMA.

The report summarizes the relevant information for this domain, including genome occurrences, alignment, motions, function classification, core structure and rankings. By clicking on the headers, one can get the detailed reports for these quantities.



Report on PDB Structure 1ama

Genome Occurrence. Domain d1ama__. No additional domains.

Genome	Aful	Mjan	Mthe	Phor	Scer	Cele	Aaeo	Syne	Ecol	Bsub	Mtub	Hinf	Hpyl	Mgen	Mpne	Bbur	Tpal	Ctra	Cpne	Rpro
Fold	18	16	15	18	32	40	19	23	36	43	33	15	9	2	2	3	5	8	10	5
Superfamily	18	16	15	18	32	40	19	23	36	43	33	15	9	2	2	3	5	8	10	5

Alignments. Domain d1ama__. No additional domains.

	# Pairs	untrimmed RMS	trimmed RMS	%sequence identity	structure p-value	sequence p-value
Fold	28	5.53	1.54	26.40	3.87e-23	6.42e-01
Superfamily	28	5.53	1.54	26.40	3.87e-23	6.42e-01
Family	6	2.00	0.43	48.66	1.57e-44	2.17e-68
Fold only	0	.	.	.	.	.
Sfam only	18	6.77	2.04	17.90	6.03e-23	1

Motions

Motions		Movies	
Protein	Aspartate Amino Transferase (AAT)	**Database ID**	aat

Function Classification. Domain d1ama__. No additional domains.

	Augmented Flybase+ENZYME	GenProtEC	MIPS
d1ama__	E-2.6.1.1	3.1.04	01.01.01

Core Structure does not exist for d1ama.

Rankings. Domain d1ama__. No additional domains.

OCCURRENCES																	
Species	20	Aful	Mjan	Mthe	Phor	Scer	Cele	Aaeo	Syne	Ecol	Bsub	Mtub	Hinf	Hpyl	Mgen	Mpne	Bbu
Number	352	18	16	15	18	32	40	19	23	36	43	33	15	9	2	2	3
Rank	14	11	7	7	7	14	38	8	10	8	5	14	9	9	21	25	25
%ile	96	92	94	94	93	93	84	95	95	96	97	93	95	94	78	75	80

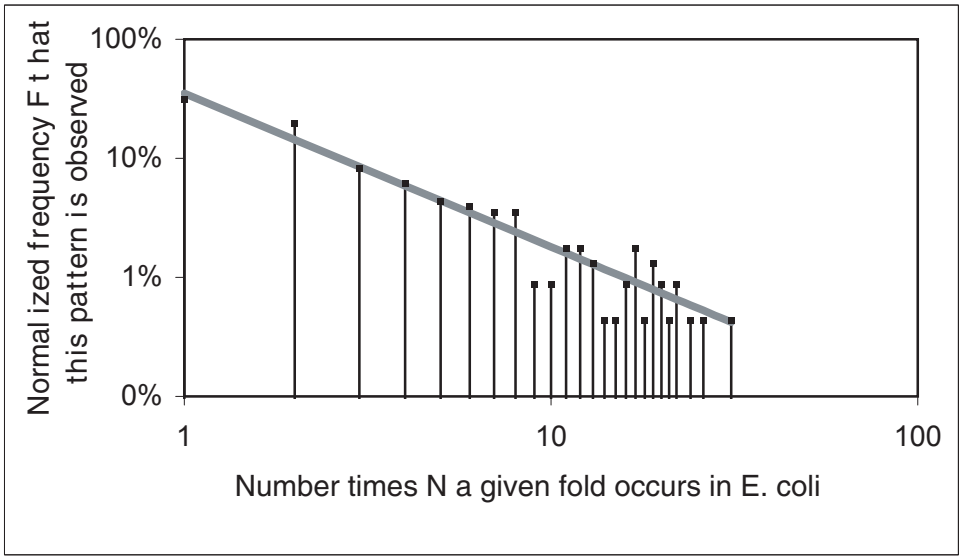
ALIGNMENTS						
	Aligned Pairs	Untrimmed RMS	Trimmed RMS	% Seq Identity	Structure P-Value	Sequence
FOLD						

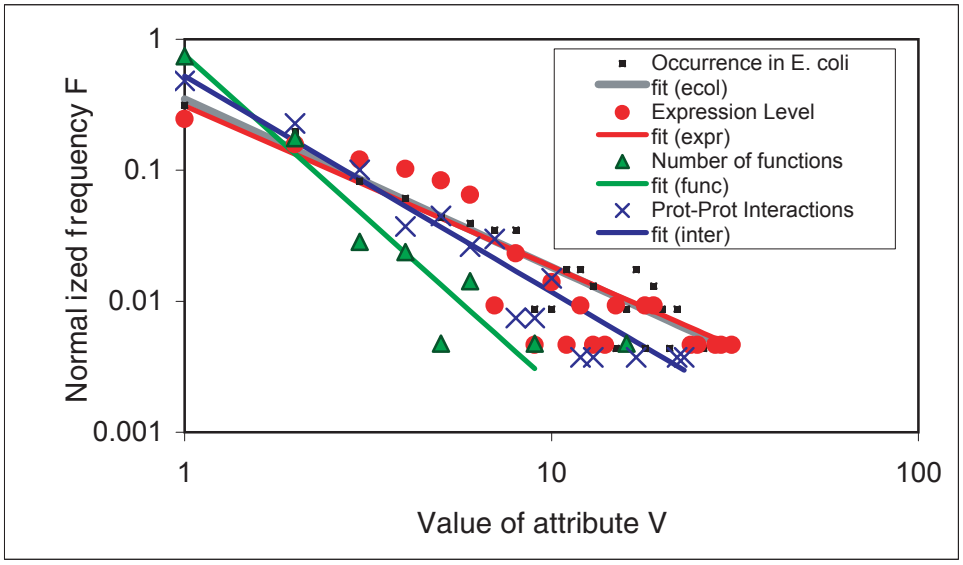
Figure A.5: Some novel relationships that are highlighted by the PartsList system.

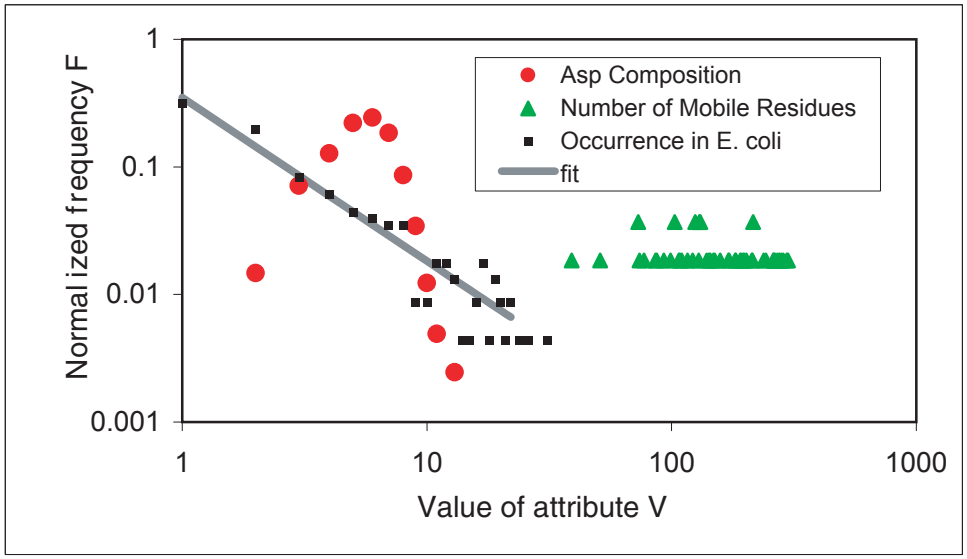
Upper panel shows the occurrence of folds in the *E. coli* genome plotted on a log-log scale -- i.e. G(ecol) using the nomenclature in Table A.1. The x-axis is the fold occurrence in the genome, while the y-axis is the number of folds with a particular occurrence. The fit of the points to a straight line shows that the falloff obeys a power law with constants a=0.35 and b=1.3 (see text).

Middle panel shows other attributes that also follow power-law behavior: the average expression level according to his merged and scaled set (L(ref) with a=.3 and b=1.2), the number of protein-protein interactions (I(pdball,none) with a=.52 and b=1.6), and the number of functions (X(func) with a=.76 and b=2.5).

Lower panel shows some attributes that do not follow power-law behavior: the Asp composition of the fold (B(Ala,pdb100)) and the number of mobile residues during a motion (M(nresidue,auto)). The fold occurrence in *E. coli* is plotted as a reference.


Normalized frequency F that this pattern is observed
100%
10%
1%
0%
1
10
100
Number times N a given fold occurs in E. coli


Normalized frequency F
1
0.1
0.01
0.001
1
10
100
Value of attribute V
Occurrence in E. coli
fit (ecol)
Expression Level
fit (expr)
Number of functions
fit (func)
Prot-Prot Interactions
fit (inter)


Normalized frequency F
1
0.1
0.01
0.001
1
10
100
1000
Value of attribute V
Asp Composition
Number of Mobile Residues
Occurrence in E. coli
fit

Appendix B: Studying Macromolecular Motions in a Database Framework: From Structure to Sequence

Overview

In this chapter, originally published elsewhere[140], I describe database approaches taken in our lab in the study of protein and nucleic acid motions. In collaboration with Prof. Mark Gerstein I have developed a database of macromolecular motions, which is accessible on the World Wide Web with an entry point at http://bioinfo.mbb.yale.edu/MolMovDB. This attempts to systematize all instances of macromolecular movement for which there is at least some structural information. At present it contains detailed descriptions of more than 100 motions, most of which are of proteins. Protein motions are further classified hierarchically into a limited number of categories, first on the basis of size (distinguishing between fragment, domain, and subunit motions) and then on the basis of packing. My packing classification divides motions into various categories (shear, hinge, other) depending on whether or not they involve sliding over a continuously maintained and tightly packed interface. I quantitatively systematize the description of packing through the use of Voronoi polyhedra and Delaunay triangulation. In addition to the packing classification, the database provides some indication about the evidence behind each motion (i.e. the type of experimental information or whether the motion is inferred based on structural similarity) and attempts to describe many aspects of a motion in terms of a standardized nomenclature (e.g. the maximum rotation, the residue selection of a fixed core, etc). Currently, I use a standard relational design to implement the database. How-

ever, the complexity and heterogeneity of the information kept in the database makes it an ideal application for an object-relational approach, and I am moving it in this direction. The database, moreover, incorporates innovative Internet cooperatively features that allow authorized remote experts to serve as database editors. The database also contains plausible representations for motion pathways, derived from restrained 3D interpolation between known endpoint conformations. These pathways can be viewed in a variety of movie formats, and the database is associated with a server that can automatically generate these movies from submitted coordinates. Based on the structures in the database I have developed sequence patterns for linkers and flexible hinges and are currently using these for the annotation of genome sequence data.

Introduction

Motion is frequently the way macromolecules (proteins and nucleic acid) carry out particular functions; thus motion often serves as an essential link between structure and function. In particular, protein motions are involved in numerous basic functions such as catalysis, regulation of activity, transport of metabolites, formation of large assemblies and cellular locomotion. In fact, highly mobile proteins have been implicated in a number of diseases—e.g., the motion of gp41 in AIDS and that of the prion protein in scrapie[7-11]. Another reason for the study of macromolecular motions results from their fundamental relationship to the principles of protein and nucleic acid structure and stability.

Macromolecular motions are amongst the most complicated biological phenom-

ena that can be studied in great quantitative detail, involving concerted changes in thousands of precisely specified atomic coordinates. Fortunately, it is now possible to study these motions in a database framework, by analyzing and systematizing many of the instances of protein structures solved in multiple conformations. I summarize here some recent work done in collaboration with Prof. Gerstein relating to the construction of a database of protein motions[139] and the use of Voronoi polyhedra to study packing[65]. I also present some preliminary results relating to creating sequence patterns for hinges and flexible linkers that I obtained in collaboration with Prof. Gerstein, Ronald Jansen, and Ted Johnson.

Table B.1. Statistics for the Mechanism of the Motions. This table cross-tabulates the two main classifying attributes of motions: their size (row heads) and their packing characteristics (column heads). I define a known motion to be a motion with two or more solved conformations, and a suspected motion is defined to have only one or fewer solved conformations. (Adapted from Gerstein and Krebs (1998).[139])

Size / Mechanism	Domain		Fragment		Subunit		Total	
Hinge	38	51%	16	59%			54	45%
Shear	14	19%	3	11%			17	14%
Partial Refolding	5	7%					5	4%
Allosteric					8	57%	8	7%
Other/Non-Allosteric	2	3%	1	4%	6	43%	9	7%
Unclassifiable	15	20%	7	26%			22	18%
Notably Motionless							1	1%
Complex							2	2%
Nucleic Acid							3	2%
Known / % category	53	72%	25	93%	11	79%	94	78%
Suspected / % category	21	28%	2	7%	3	21%	27	22%
Totals / % DB	74	62%	27	23%	14	12%	121	100%

Morph of "Calmodulin"

Figure B.1 (preceding page). The Motions Database on the Web. LEFT shows the World Wide Web "home page" of the database. One can type keywords in the small box at the top to retrieve entries. RIGHT shows a protein 'morph' (animated representation) for calmodulin referenced by the database, along with the start of the database entry. Graphics and movies are accessed by clicking on an entry page. (These have been deliberately segregated from the textual parts of the database since the interface was designed to make it easy to use on a low-bandwidth, text-only browser, e.g. lynx or the original www_3.0.) The main URL for the database is http://bioinfo.mbb.yale.edu/MolMovDB**.** Beneath this are pages listing all the current movies, graphics illustrating the use of VRML to represent endpoints, and an automated submission form to add entries to the database. The database has direct links to the PDB for current entries (http://www.pdb.bnl.gov); the obsolete database (http://pdbobs.sdsc.gov) for obsolete entries; scop (http://scop.mrc-lmb.cam.ac.uk); Entrez/PubMed (http://www.ncbi.nlm.nih.gov/PubMed/medline.html); and LPFC (http://smi-web.stanford.edu/projects/helix/LPFC). Through these links one can easily connect to other common protein databases such Swiss-Prot, Pro-Site, CATH, RiboWeb, and FSSP[19,24,30,31,108-110,287].

Table B.2 Standard Statistics for the Magnitude of the Motions. The motions in the database range greatly in size, with maximum mainchain displacements between 1.5 and 60 Å. All the statistics are for version 1.7 of the database, based on the relatively small set of values culled from the literature. The averages are only approximate given the sparse nature of the data. I am developing software tools to extract these values automatically from structural data. (Adapted from Gerstein and Krebs (1998).[86])

Value	Num. Entries	min	max	average
Maximum Cα displacement	11	1.5	60	12
Maximum Atomic Displacement	3	8.8	10	9.3
Maximum Rotation	12	5	148	24
Maximum Translation	2	0.7	2.7	1.7

The Database

The primary public interface to the database consists of coupled hypertext documents available over the World Wide Web at http://bioinfo.mbb.yale.edu/MolMovDB. As shown in Figure B.1, use of the web interface is straightforward and simple. The database may be browsed either by typing various search keywords into the main page or by navigating through an outline. Either way brings one to the entries. Thus far, the database has ~120 entries, which reference over 240 structures in the Protein Databank (PDB) (Table B.1).

Unique Motion Identifier

Each entry is indexed by a *unique motion identifier*, rather than around individual proteins and nucleic acids. This is necessary because a single macromolecule can not only have a number of motions, but the essential motion can be shared amongst a number of different macromolecules.

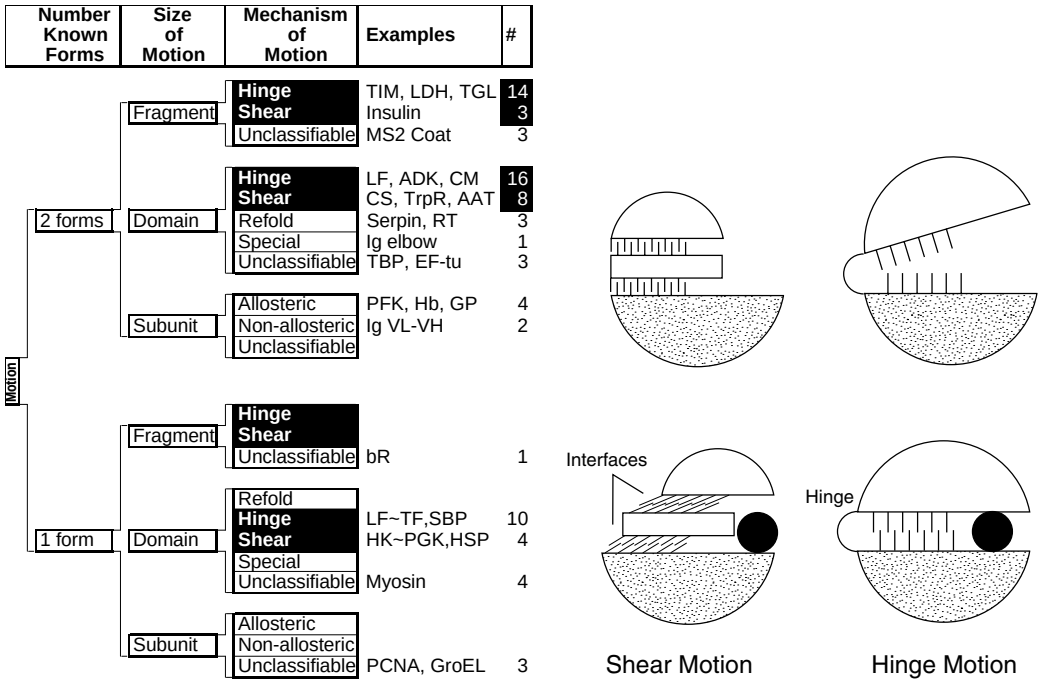


Figure B.2. Schematic Showing the Overall Classification Scheme for Motions. TOP-LEFT, the database is organized around a hierarchical classification scheme, based on size (fragment, domain, subunit) and then packing (hinge or shear). Currently, the hierarchy also contains a third level for whether or not the motion is inferred. TOP-RIGHT is a schematic showing the difference between shear (sliding) and hinge motions. Figure adapted from Gerstein et al.[Gerstein, 1993 #517; Gerstein, 1994 #769]. It is important to realize that the hinge-shear classification in the database is only "predominate" so that a motion classified as shear can contain a newly formed interface and one classified as hinge can have a preserved interface across which there is motion. The essential characteristics of the various motions are summarized below. (Adapted from Gerstein and Krebs (1998).)

Attributes of a Motion

In addition to the motion identifier, each entry has the following information:

Structures.

Brookhaven Protein DataBank (PDB) identifiers are given for the various conformations of the macromolecule (e.g. open and closed). The identifiers have been made into hypertext links directly to the structure entries in the main protein and nucleic acid databases (PDB and NDB) and to sequence and journal cross-references via the Entrez and MMDB databases[29-33]. Links are also made to related structures via the Structural Classification of Proteins (SCOP)[34,35].

Literature.

Literature references are given. Where possible these are via Medline unique identifiers, allowing a link to be made into the PubMed database[31,32].

Documentation.

Each entry has a paragraph or so of plain text documentation. While this is, in a sense, the least precisely defined field, it is the heart of each entry, describing the motion in intelligible prose and referring to figures, where appropriate.

Standardized Nomenclature.

For many entries I describe the overall motion using standardized numeric terminology, such as the maximum displacement (overall and of just backbone atoms) and the degree of rotation around the hinge. These statistics are summarized in Table B.2. I also attempt to give the transformations (from ii) needed to optimally superimpose and orient each coordinate set to best see the motion (i.e. down screw-axis) and the selections of residues with large changes in torsion angles, packing efficiency, or neighbor contacts.

Graphics.

Many entries have links to graphics and movies describing the motion, often depicting a plausible interpolated pathway (see below).

Hierarchical Classification Scheme Based on Size Then Packing

Size Classification: Fragment, Domain, Subunit

The most basic division in the current classification scheme is between proteins and nucleic acids. There are currently far fewer nucleic-acid motion entries than those of proteins, reflecting the much larger number of known protein structures.[vi] At present, the database includes the nucleic-acid motions evident from comparing various conformations of the known structures of catalytic RNAs and tRNAs (specifically, the Hammerhead ribozyme, the P4-P6 domain of the Group II intron, and Asp-tRNA[36-40]).

The classification scheme for proteins has the hierarchical layout shown in Figure B.2. The basic division is based on the size of the motion. Ranked in order of their size, protein movements fall into three categories: the motions of fragments smaller than domains, domains, and subunits.[vii] Nearly all large proteins are built from domains, and domain motions, such as those observed in hexokinase or citrate synthase,[41,42] provide the most common examples of protein flexibility[1-3].

The motion of fragments smaller than domains usually refers to the motion of surface loops, such as the ones in triose phosphate isomerase or lactate dehydrogenase, but it can

[vi] At the time of writing, the PDB contained in excess of 6600 protein structures, but less than 600 nucleic acids structures.

[vii] There is, of course, also the motion (i.e. rotation) of individual sidechains, often on the protein surface. However, this is on a much smaller scale than the motion of fragments or domains. It also occurs in all proteins. Consequently, sidechain motions are not considered to constitute individual motions in the database, being considered here a kind of background, intrinsic flexibility, common to all proteins.

also refer to the motion of secondary structures, such as of the helices in insulin[43-45].

Often domain and fragment motions involve portions of the protein closing around a binding site, with a bound substrate stabilizing a closed conformation. They, consequently, provide a specific mechanism for induced-fit in protein recognition[46,47]. In enzymes this closure around a binding site has been analyzed in particular detail[13,48-51]. It serves to position important chemical groups around the substrate, shielding it from water and preventing the escape of reaction intermediates.

Subunit motion is distinctly different from fragment or domain motion. It affects two large sections of polypeptide that are *not* covalently connected. It is frequently part of an allosteric transition and tied to regulation[52,53]. The relative motions of the subunits in the transport protein hemoglobin and the enzyme glycogen phosphorylase change the affinity with which these proteins bind to their primary substrates[54,55] and are good examples.

Packing Classification: Hinge and Shear

For protein motions of domains and smaller units, I have systematized the motions on the basis of packing, using a scheme developed previously[1,139]. This is because the tight packing of atoms inside of proteins provides a most fundamental constraint on protein structure[56-61]. Unless there is a cavity or packing defect, it is usually impossible for an atom inside a protein to move much without colliding with a neighboring atom[62,63].

Internal interfaces between different parts of a protein are packed very tightly[1,64,65]. Furthermore, they are not smooth, but are formed from interdigitating sidechains. Common sense consideration of these aspects of interfaces places strong con-

straints on how a protein can move and still maintain its close packing. Specifically, maintaining packing throughout a motion implies that the sidechains at the interface must maintain their same relative orientation and pattern of inter-sidechain contacts in both conformations (e.g. open and closed).

These straightforward constraints on the types of motions that are possible at interfaces allow an individual movement within a protein to be described in terms of two basic mechanisms, shear and hinge, depending on whether or not it involves sliding over a continuously maintained interface[1] (Figure B.2). A complete protein motion (which can contain many of these smaller "movements") can be built up from these basic mechanisms. For the database, a motion is classified as *shear* if it predominately contains shear movements and as *hinge* if it is predominately composed of hinge movements. More detail on the characteristics of the two types of motion follows.

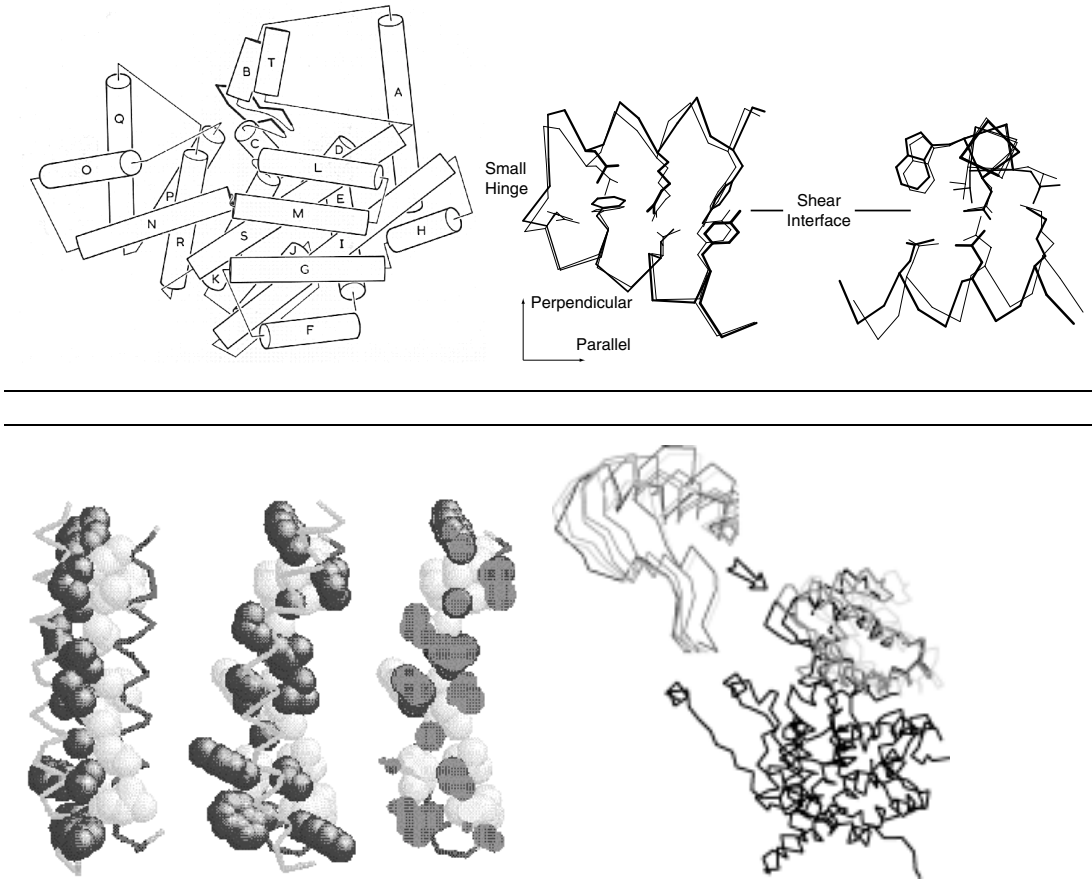


Figure B.3. Closeup on the Shear Mechanism. The figure gives a close up illustrating shear motion in one protein, citrate synthase[41,42]. TOP-LEFT, Cartoon of one subunit of citrate synthase (1CTS) gives an overall view of the protein showing that it is composed of many helices. The adjacent one is related by two-fold axis shown. The small two-stranded sheet is omitted to improve clarity. a-helices are represented by cylinders. The small domain contains helices N, O, P, Q, and R. TOP-MIDDLE and TOP-RIGHT show representative shear motions between close-packed helices. Note how the mainchain only shifts by a small amount and the sidechains stay in the same rotamer configuration. BOTTOM-LEFT highlights the "knobs into holes" interdigitation of two close-packed helices. BOTTOM-RIGHT shows how these small motions can be added together to produce a large overall motion. Specifically, many small motions add up to shift helix O by 10.1 Å and rotate it by 28°. The incremental motion in shear domain closure is shown by Ca traces of the whole protein and of a closeup of the OP loop. BLACK is the apo form; WHITE, holo form; GRAY, cumulative effect of motion over the K, P, and then Q helix-helix interfaces. (The apo form was fit to the holo form, first on the core, and then on the K, P, and Q helices.) (Parts adapted from Gerstein and Krebs (1998).)

Shear. As shown in Figure B.3, the shear mechanism basically describes the special kind of sliding motion a protein must undergo if it wants to maintain a well-packed interface. Because of the constraints on interface structure described above, individual shear motions have to be very small. Sidechain torsion angles maintain the same rotamer configuration[66] (with <15° rotation of sidechain torsions); there is no appreciable main-chain deformation; and the whole motion is parallel to the plane of the interface, limited to total translations of ~2 Å and rotations of 15°. Since an individual shear motion is so small, a single one is not sufficient to produce a large overall motion, and a number of shear motions have to be concatenated to give a large effect — in a similar fashion to each plate in a stack of plates sliding slightly to make the whole stack lean considerably. Examples include the Trp repressor and aspartate amino transferase[67,68].

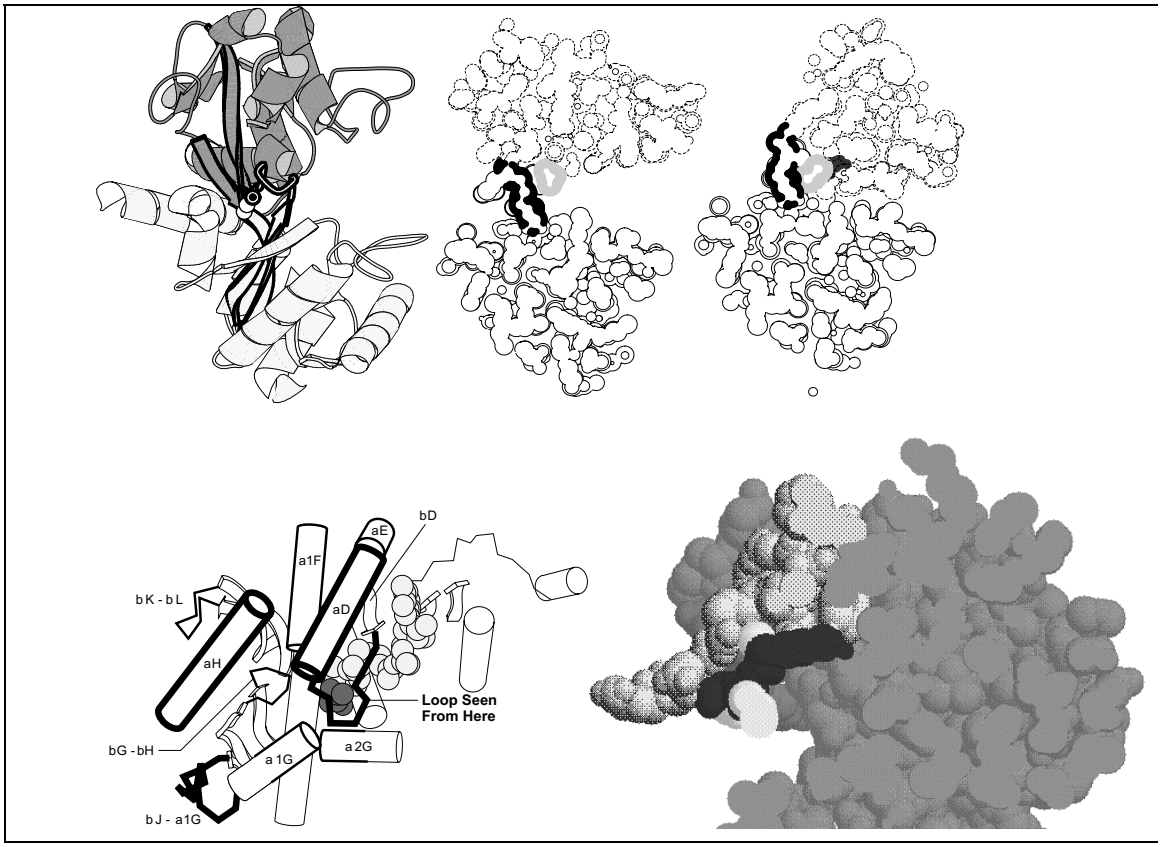


Figure B.4. Close-up on the Hinge Mechanism. The figure shows the hinge motion in lactoferrin[20,45]. FAR-LEFT shows a ribbon drawing of the protein in the open conformation. The view is down the screw-axis, which is indicated in the figure by the circle with the dot in it. The screw-axis passes very close to the hinge region, which occurs in the middle of two beta strands (highlighted in bold). MIDDLE-LEFT and MIDDLE-RIGHT show the open and closed conformations in terms of space filling slices. The hinge region is highlighted by a thick black line. Note how few packing constraints there are on the hinge in contrast to the other atoms in the protein. (Figure adapted from Gerstein (1993).[45]) BOTTOM-LEFT shows the placement of a mobile loop in another protein, lactate dehydrogenase. BOTTOM-RIGHT shows a close-up of this loop that highlights the absence of close-packing at the base of the hinge. Hinge mainchain is shown in black (first hinge) and almost white (second hinge). Rest of protein is shown in shades of gray.

As shown in Figure B.4, hinge motions occur when there is *no* continuously maintained interface constraining the motion. These motions usually occur in proteins that have two domains (or fragments) connected by linkers (i.e. hinges) that are relatively unconstrained by packing. A few large torsion angle changes in the hinges are sufficient to produce almost the whole motion. The rest of the protein rotates essentially as a rigid body, with the axis of the overall rotation passing through the hinges. The overall motion is always perpendicular to the plane of the interface (so the interface exists in one conformation but not in the other, as in the closing and opening of a book) and is identical to the local motion at the hinge. Examples include lactoferrin and tomato bushy stunt virus (TBSV)[69,70].

Gerstein et al.[64,71] analyzed the hinged domain and loop motion in specific proteins (lactate dehydrogenase, adenylate kinase, lactoferrin). These studies emphasized how critical the packing at the base of a protein hinge is (in the same sense that the "packing" at the base of an everyday door hinge determines whether or not the door can close). Protein hinges are special regions of the mainchain in the sense that they are exposed and have few packing constraints on them and are thus free to sharply kink (Figure B.4). Most mainchain atoms, in contrast, are usually buried beneath layers of other atoms (usually sidechain atoms), precluding large torsion angle changes and hinge motions.

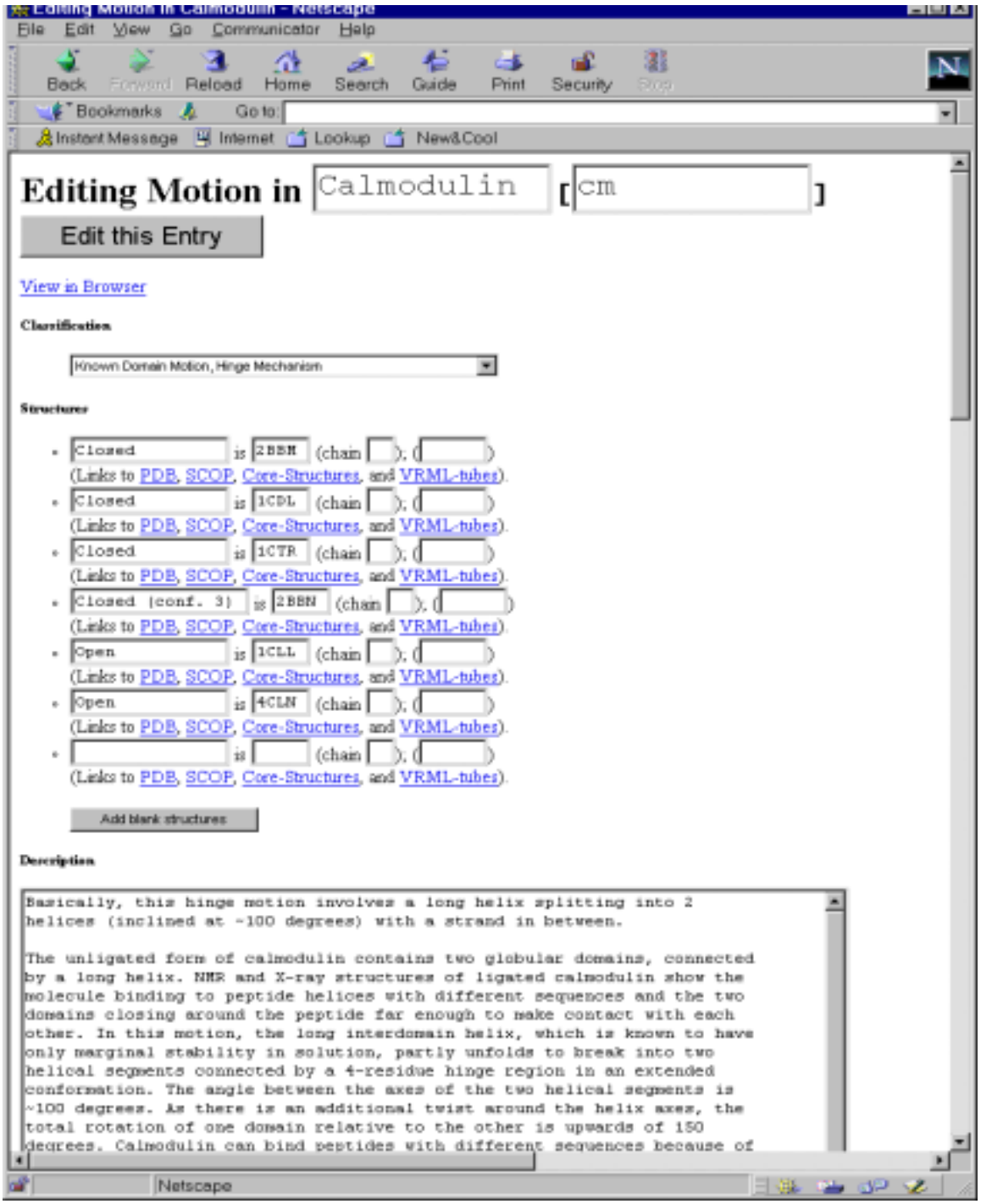


Figure B.5. Editing a motion remotely over the Internet. The Database of Macromolecular Movements features an innovative Web form (shown here) that allows authorized remote users to collaborate and edit motions from remote sites around the world. Saved changes to motions may be previewed to see how they would appear to an end user and then applied to the database. If desired, saved changes can be made to appear immediately in the public Web interface to the database.

It is important to note that because most shear motions do, in fact, contain hinges, (joining the various sliding parts) the existence of a hinge is not the salient difference between the two basic mechanisms. Instead, it is the existence of a continuously maintained interface.

Other Classification

Most of the fragment and domain motions in the database fall within the hinge-shear classification. However, I have created additional categories to deal with the small number of exceptions.

Data Entry

One innovative feature of the database is that it allows authorized remote researchers to enter motions in their area of expertise directly into the database via a Web form. Authorization to edit a given motion entry, if necessary, works in conjunction with the standard password feature built into modern Web browser systems. The layout of the Web form is analogous to that of a normal HTML page describing a motion in the database, except that the various fields have been replaced by textboxes and pull-down selectors to make the Web page editable. The user retrieves either a blank form or a form corresponding to a pre-existing motion entry, makes appropriate changes remotely over the Internet via his or her Web browser, and then simply clicks the ‘Submit’ button to save changes into the database. Depending on whether or not the user has editing privileges over a particular motion entry, the changes may be published immediately or upon further approval by the database maintainers. The remote user may immediately preview the edited motion entry to see what it will look like once it becomes public.

The Web form system (Figure B.5) takes advantage of advanced features of the Informix Dynamic Server with Universal Option to enable user previews. The Web Datablade module allows database content to be dynamically and rapidly translated into Web content with little additional overhead compared to static pages. Because updates to the database can be translated instantaneously into updated Web content, remote editors are able to preview their changes as it will appear to the end database user instantaneously before submitting or publishing them. Previously, I stored the database using the MSQL database software package, which is freely available to academic users. Unlike the commercial Informix system, the MSQL package does not support Application Program Interfaces (APIs) that allow for an efficient, rapid translation of database content into Web content. Consequently, it was necessary to store the Web interfaces as static HTML files on the server. For Web content to remain current, these pages would need to be rebuilt each time the database changed, a time-consuming process that would have prevented accurate previews. In addition, the Informix database system also features state-of-the-art transaction concurrency and logging, important features when multiple users are simultaneously updating the database.

In this way, the database takes full advantage of the cooperatively features of the Internet and modern database software, allowing experts in distant parts of the world to collaborate simultaneously on macromolecular motions. In addition to accelerating the rate at which the database may be populated, this feature improves the accuracy and timeliness of existing database entries by allowing them to be edited, revised, and updated, if necessary, by experts in the field.

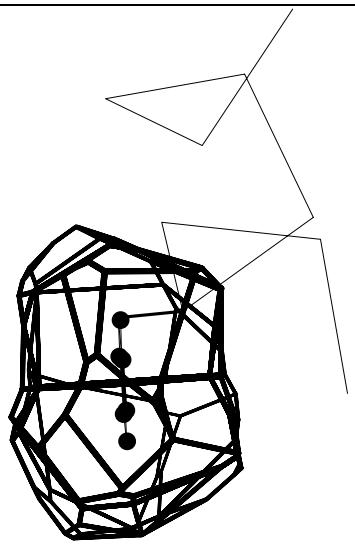


Figure B.6. Voronoi Polyhedra. Two representative Voronoi polyhedra from 1CSE (subtilisin). On the left is shown the polyhedron around the sidechain hydroxyl oxygen (OG) of a serine. On right is shown the six polyhedra around the atoms in a Phe ring.

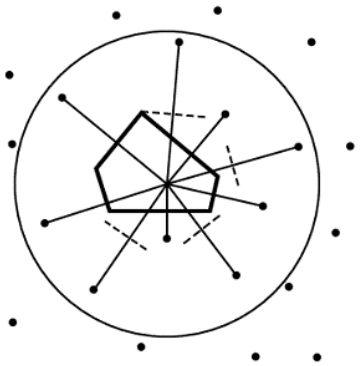


Figure B.7. The Voronoi Polyhedra Construction. A schematic showing the construction of a Voronoi polyhedron in 2-dimensions. The asymmetry parameter is defined as the ratio of the distances between the central atom and the farthest and nearest vertex.

Internet Hits

The database is currently receiving over 65,000 hits from over 45,000 sites each month. Internet traffic on the database's main web server grew approximately exponentially between November, 1997, and February 1998, with database usage doubling approximately every other month during this period. In recent months, database usage has continued to grow, albeit at a somewhat reduced rate. I expect this trend to continue as the database becomes established in the structural biology community.

Standardized Tools For Protein Motions

Quantification of packing using Voronoi polyhedra

Packing clearly is an essential component of the motions classification. Often this concept is discussed loosely and vaguely by crystallographers analyzing a particular protein structure—for instance, "Asp23 is packed against Gly38" or "the interface between domains appears to be tightly packed." I have attempted to systematize and quantify the discussion of packing in the context of the motions database through the use of particular geometric constructions called Voronoi polyhedra and Delaunay triangulation.[64]

Voronoi polyhedra are a useful way of partitioning space amongst a collection of atoms. Each atom is surrounded by a single convex polyhedron and allocated the space within it (Figure B.6). The faces of Voronoi polyhedra are formed by constructing dividing planes perpendicular to vectors connecting atoms, and the edges of the polyhedra result from the intersection of these planes.

Voronoi polyhedra were originally developed (obviously enough) by Voronoi[288] nearly a century ago. Bernal and Finney[289] used them to study the structure of liquids in the 1960s. However, despite the general utility of these polyhedra, their application to proteins was limited by a serious methodological difficulty: while the Voronoi construction is based around partitioning space amongst a collection of "equal" points, all protein atoms are not equal: some are clearly larger than others (e.g. sulfur versus oxygen). Richards[290] found a solution to this problem and first applied Voronoi polyhedra to proteins in 1974. He has, subsequently, reviewed their use in this application[59,60].

Voronoi polyhedra are particularly useful in studying the packing of the protein interior. This is because the construction of Voronoi polyhedra allocates all space

amongst a collection of atoms; there are no gaps as there would be if one, say, simply drew spheres around the atoms. Thus, the volume of cavities or defects between atoms are included in their Voronoi volume, and one finds that the packing efficiency is inversely proportional to the size of the polyhedra. This indirect measurement of cavities contrasts with other types of calculations that measure the volume of cavities explicitly[291]. Moreover, since protein interiors are tightly packed, fitting together like a jig-saw puzzle, the various types of protein atoms occupy well-defined amounts of space. This fact has made the calculation of standard volumes for residues in proteins[57,292] a worthwhile proposition.

Voronoi polyhedra calculations have been applied to other aspects of packing in protein structure. In particular, they have been used to study protein-protein recognition[293], protein motions[64], and the protein surface[65,294-296]. As the Voronoi volume of an atom is a weighted average of the distances to all its neighbors (where the contact area with a neighbor is the weight), Voronoi polyhedra are very useful in assessing interatomic contacts[296-298]. Furthermore, the faces of Voronoi polyhedra have been used to characterize protein accessibility and to assess the fit of docked substrates in enzymes[299,300].

Voronoi polyhedra have many uses beyond the analysis of protein structures. For instance, they have also been used in the analysis of liquid simulations[301] and in weighting sequences to correct for over- or under-representation in an alignment[302]. In non-biological applications, they are used in "nearest-neighbor" problems (trying to find the neighbor of a query point) and in finding the largest empty circle in a collection of points[303]. The dual of a Voronoi diagram is a Delaunay triangulation. Since this triangula-

tion has the "fattest" possible triangles, it is convenient for such procedures as finite element analysis. Furthermore, the border of Delaunay triangulation is the convex hull of an object, which is useful in graphics[303].

The simplest method for calculating volumes with Voronoi polyhedra is to put all atoms in the system on a grid. Then go to each grid-point (i.e. voxel) and add its volume to the atom center closest to it. This is prohibitively slow for a real protein structure, but it can be made somewhat faster by randomly sampling grid-points. It is, furthermore, a useful approach for high-dimensional integration[302] and for the curved dividing surface approach discussed later.

More realistic approaches to calculating Voronoi volumes have two parts: (1) for each atom find the vertices of the polyhedron around it and (2) systematically collect these vertices to draw the polyhedron and calculate its volume.

In the basic Voronoi construction (Figure B.7), each atom is surrounded by a unique limiting polyhedron such that all points within an atom's polyhedron are closer to this atom than all other atoms. Points equidistant from two atoms are on a plane; those equidistant from three atoms are on a line, and those equidistant from four centers form a vertex. One can use this last fact to easily find all the vertices associated with an atom. With the coordinates of four atoms, it is straightforward to solve for possible vertex coordinates using the equation of a sphere.[*] One then checks whether this putative vertex is closer to these four atoms than any other atom; if so, it is a vertex.

In the procedure outlined above, all the atoms are considered equal, and the divid-

[*] That is, one uses four sets of coordinates (x,y,z) to solve for the center (a,b,c) of the sphere: $(x-a)^2+(y-b)^2+(z-c)^2=r^2$. (This method can fail for certain pathological arrangements of atoms that would not normally be encountered in a real protein structure; see Proacci and Scateni304. Procacci, P. & Scateni, R. A General Algorithm for Computing Voronoi Volumes: Application to the Hydrated Crystal of Myoglobin. *Int. J. Quant. Chem.* **42**, 151-1528 (1992).).

ing planes are positioned midway between atoms (Figure B.6). This method of partition, called bisection, is not physically reasonable for proteins, which have atoms of obviously different size (such as oxygen and sulfur). It chemically misallocates volume, giving an excess to the smaller atom.

Two principal methods of re-positioning the dividing plane have been proposed to make the partition more physically reasonable: method B[290] and the radical-plane method[305]. Both methods depend on the radii of the atoms in contact (R_1 and R_2) and the distance between the atoms (D).

Representing Motion Pathways as "Morph Movies"

One of the most interesting of the complex data types kept in the database are "morph movies" giving a plausible representation for the pathway of the motion. These movies can immediately give the viewer an idea of whether the motion is a rigid-body displacement or involves significant internal deformations (as in tomato bushy stunt virus versus citrate synthase). Pathway movies were pioneered by Vorhein et al.[98], who used them to connect the many solved conformations of adenylate kinase.

Normal molecular-dynamics simulations (without special techniques, such as high temperature simulation or Brownian dynamics[99-101]) cannot approach the timescales of the large-scale motions in the database. Consequently a pathway movie cannot be generated directly via molecular simulation. Rather, it is constructed as an interpolation between known endpoints (usually two crystal structures). The interpolation can be done in a number of ways.

Straight Cartesian interpolation. The difference in each atomic coordinate (between the known endpoint structures) is simply divided into a number of evenly spaced steps,

and intermediate structures are generated for each step. This was the method used by Vorhein et al. It is easy to do, only requiring that the beginning and ending structures be intelligently positioned by fitting on a motionless core. However, it produces intermediates with clearly distorted geometry.

Interpolation with restraints. This is the above method where each intermediate structure is restrained to have correct stereochemistry and/or valid packing. One simple approach is to minimize the energy of each intermediate (with only selected energy terms) using a molecular mechanics program, such as X-PLOR[102]. As described in Chapter 3, the database provides a server that applies this interpolation technique to two arbitrary structures, generating a movie.

Analysis of Amino Acid Composition of Linker Sequences

Now that I have developed a database of protein motions, an essentially structure-orientated database, I want to use this to help interpret the mass of sequence data coming out of genome sequencing projects. In this way I am extrapolating ideas developed on the (relatively) smaller structure database to the much larger sequence database. I propose to do this through the calculation of two propensity scales for amino acids to be in linkers or flexible hinges.

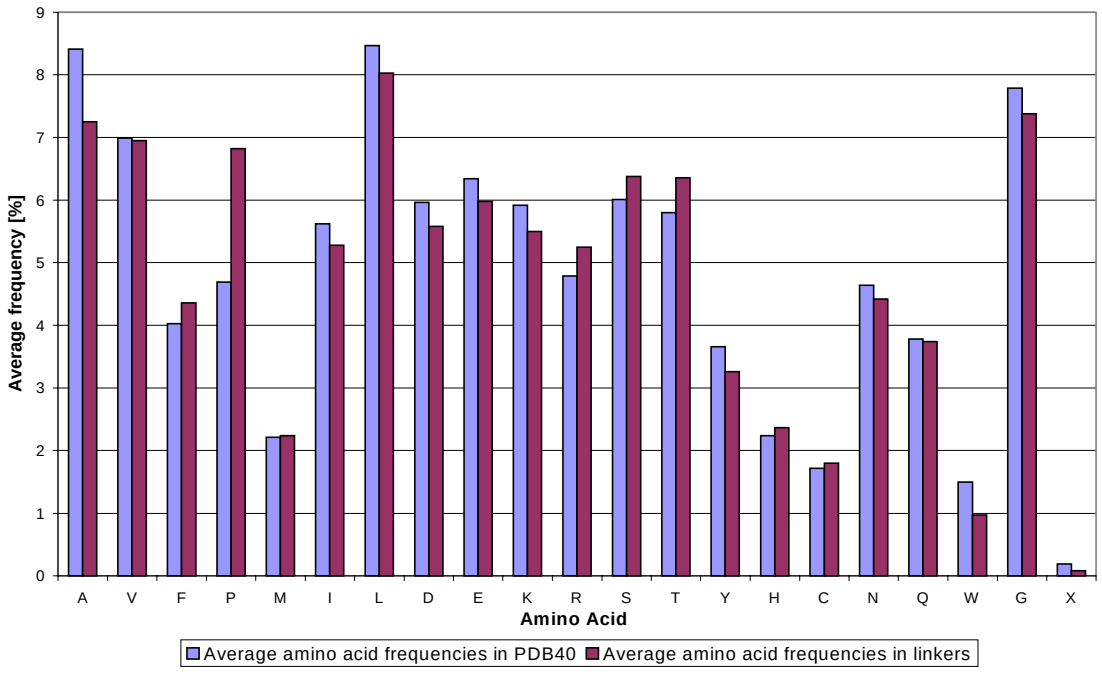


Figure B.8. Comparison of the average amino acid composition in linker sequences and proteins in general (as represented by the PDB40 database).

Solved protein structures typically reveal different domains of proteins and linker regions between these domains. Linker regions are typically flexible, and, as such, form the basis for the hinge regions that allow two protein domains or fragments to move relative to each other as a part of a hinge mechanism.

Information about the amino acid composition of linker sequences can potentially be used to predict protein domains in protein sequences of unknown structure. In particular, a profile of flexible linker regions might be used to predict the location of domain hinges, for structural annotation of genome sequences.[93] Here I present some preliminary results involving two methods for statistical analysis of linker sequences.

Propensities for Linkers in General

My first method of analysis of linker sequences includes both flexible as well as inflexible linkers. In this method I have arbitrarily defined a linker sequence as the 16

residue region centered around the peptide bond linking two domains.

The analysis of the amino acid composition of linker sequences is an example of deriving sequence information from structural information. The structural information (i.e., the location of protein domains) can be found in the Structural Classification of Proteins (SCOP)[34,35]. SCOP contains several databases of amino acid sequences of protein domains. In my study, the PDB40 database provided by SCOP has been used to create a database of linker sequences. The PDB40 database comprises a subset of proteins in the Protein Data Bank (PDB) with known structure selected so that, when aligned, no two proteins in the subset show a sequence identity of 40% or greater. Thus, the data set is not biased towards protein structures listed multiple times in the PDB. I was able to extract 234 linker sequences from the PDB40 database, although the PDB40 database itself contains about 1,500 protein sequences. This mainly reflects the fact that many proteins consist of only a single domain and therefore contain no linker region.

Figure B.8 compares the average amino acid composition of the linker sequences with the average amino acid composition of the PDB40 database, while Table B.3 shows in more detail the profile of the amino acid composition at each of the sixteen positions in the linker sequence. For an interpretation of these results it is important to compute two-sided P-values to determine which amino acids show statistically different frequencies in linkers than in the database as a whole. (A two-sided P-value represents the probability that, in a data set of equal size drawn at random from the PDB40 database, a given amino acid would have a frequency of occurrence as different as or more different from its occurrence in the entire PDB40 database than what was actually observed in the linker subset.) Figure B.9 shows the P-values for the average amino acid composition in the linkers.

Table B.3. Profile of the amino acid composition in linker sequences for every single linker position in detail compared with the PDB40 averages. A linker has been arbitrarily defined as the 16 residue region centered around the peptide bond (between positions 8 and 9) linking two domains. Positions where the amino acid frequency is less than the PDB40 average have a gray background.

																	PDB40 average
A	8.6	7.8	4.7	5.6	6.0	8.6	9.5	5.6	4.7	6.5	5.6	7.3	6.9	9.1	9.5	9.9	8.4
V	6.0	8.2	8.2	6.0	8.2	5.6	9.1	6.0	8.2	4.7	6.0	4.7	7.3	9.1	5.2	8.6	7.0
F	4.7	3.9	6.5	3.5	2.6	2.6	6.0	2.6	4.7	3.0	4.3	6.0	5.2	4.3	4.3	5.6	4.0
P	3.9	6.5	6.0	6.0	5.2	9.1	6.9	10.8	9.1	10.3	9.9	6.0	8.6	2.6	4.7	3.5	4.7
M	4.7	1.3	1.3	2.6	2.6	0.0	1.7	1.7	4.3	3.0	1.3	1.3	2.2	1.7	3.0	3.0	2.2
I	5.6	3.5	7.3	6.5	3.9	6.0	3.9	3.5	5.2	6.9	4.7	2.6	4.7	8.6	5.6	6.0	5.6
L	11.6	9.1	11.2	6.0	16.4	7.3	4.3	6.5	8.2	3.5	7.3	5.2	7.3	6.5	10.3	7.8	8.5
D	4.7	6.5	6.0	3.9	6.0	4.7	5.6	8.6	4.3	3.9	3.5	7.3	6.9	7.3	4.3	5.6	6.0
E	5.2	5.2	3.9	6.5	4.7	4.7	7.8	4.7	6.5	4.3	6.5	9.1	7.3	5.2	8.6	5.6	6.3
K	5.2	6.5	3.9	5.6	5.2	6.9	4.7	4.7	6.0	7.8	3.9	6.5	5.2	5.2	3.0	7.8	5.9
R	5.2	3.9	4.7	9.1	6.5	5.2	5.2	5.6	5.6	4.7	6.0	5.2	5.2	4.7	3.0	4.3	4.8
S	7.8	6.0	5.2	6.9	6.5	8.2	6.9	6.5	3.5	6.0	9.5	7.8	4.3	3.9	8.6	4.7	6.0
T	4.7	5.6	3.0	5.6	6.5	9.5	6.9	6.0	6.5	11.2	7.3	6.5	6.0	4.7	8.2	3.5	5.8
Y	2.2	3.9	6.5	3.0	3.5	2.2	2.6	3.5	2.2	3.9	2.6	2.2	3.0	3.5	3.5	4.3	3.7
H	1.7	3.5	3.0	3.5	3.5	2.6	3.5	2.2	2.2	0.9	1.7	2.2	1.7	2.6	1.3	2.2	2.2
C	1.7	2.6	0.9	1.3	1.7	2.6	0.4	2.2	0.9	1.3	4.7	1.7	1.7	3.9	0.4	0.9	1.7
N	4.7	3.9	3.5	6.5	3.0	4.3	2.6	3.0	5.6	5.2	3.5	6.5	3.9	6.0	3.0	5.6	4.6
Q	3.9	5.2	3.5	5.2	2.6	0.9	3.0	2.2	3.5	4.7	3.5	2.2	6.5	4.3	4.3	4.7	3.8
W	1.3	0.9	0.9	2.6	0.4	0.9	0.4	0.9	0.4	1.3	0.0	1.3	0.4	0.9	2.2	0.9	1.5
G	6.0	6.0	9.9	4.3	5.2	8.2	9.1	13.4	8.2	6.9	8.2	8.6	5.6	6.0	6.9	5.6	7.8
X	0.4	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	

I was able to conclude, with better than 98% confidence, that linker regions are proline-rich and alanine- and trypthophan-poor. In particular, the statistical evidence that linkers are proline-rich is unusually strong and is significant at better than the hundredth-of-a-percent level. Table B.4 shows the P-values of the amino acids at each of the sixteen linker positions.

In Table B.4 and Figure B.9 the amino acids have been roughly grouped according to the attributes hydrophobic, charged, and polar (following the classification of Branden and Tooze[306]). As shown in Table B.4 and Figure B.9, the frequencies of the remaining amino acids in linkers are not statistically different from the database as a whole at the 5% significance level.

The statistical significance of the results of the computed amino acid averages can be assessed by comparing the composition of the linker sequences with random data sets of sequences of the same length and the same amount taken from the PDB40 database. The number of times a single amino acid occurs in multiple random data sets follows the binomial distribution according to the familiar equation:

$$P^N(k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Table B.4. P-values for the profile of the amino acid composition of linker sequences for every single position in the linkers. P-values less than 0.05 are represented by a gray background. The low P-values for proline in positions 6 to 11 are most conspicuous. The classification according to the attributes hydrophobic, charged, and polar (Branden and Tooze[76]) does not provide a satisfactory explanation for the observed levels of amino acids (see also Figure B.9).

A	.908	.728	4e-2	.125	.196	.908	.562	.125	4e-2	.293	.125	.561	.415	.729	.562	.416	hydrophobic
V	.577	.481	.481	.577	.481	.417	.224	.577	.481	.184	.577	.184	.841	.224	.285	.338	
F	.598	.911	.059	.666	.276	.276	.126	.276	.598	.449	.836	.126	.393	.836	.836	.235	
P	.573	.207	.346	.346	.737	2e-3	.114	5e-5	2e-3	1e-4	3e-4	.346	4e-3	.134	.971	.385	
M	1e-2	.366	.366	.717	.717	2e-2	.637	.637	3e-2	.433	.366	.366	.961	.637	.433	.433	
I	.990	.155	.267	.585	.257	.793	.257	.155	.772	.408	.571	4e-2	.571	5e-2	.990	.793	
L	.084	.754	.136	.186	3e-5	.541	2e-2	.280	.882	6e-3	.541	.071	.541	.280	.312	.705	
D	.442	.750	.966	.185	.966	.442	.821	.089	.296	.185	.108	.389	.556	.389	.296	.821	charged
E	.476	.476	.127	.936	.327	.327	.384	.327	.936	.211	.936	.092	.545	.476	.158	.653	
K	.638	.730	.194	.842	.638	.538	.457	.457	.945	.243	.194	.730	.638	.638	.061	.243	
R	.793	.530	.974	2e-3	.240	.793	.793	.575	.575	.974	.389	.793	.793	.974	.215	.742	
S	.269	.990	.599	.578	.774	.166	.578	.774	.101	.990	2e-2	.269	.283	.176	.095	.425	polar
T	.498	.897	.069	.897	.673	2e-2	.485	.886	.673	5e-4	.328	.673	.886	.498	.121	.127	
Y	.234	.864	2e-2	.619	.872	.234	.402	.872	.234	.864	.402	.234	.619	.872	.872	.612	
H	.619	.237	.455	.237	.237	.740	.237	.939	.939	.166	.619	.939	.619	.740	.354	.939	
C	.997	.336	.345	.647	.997	.336	.139	.634	.345	.647	2e-2	.997	.997	2e-2	.139	.345	
N	.942	.597	.404	.193	.251	.820	.143	.251	.500	.710	.404	.193	.597	.326	.251	.500	
Q	.937	.281	.804	.281	.359	2e-2	.562	.206	.804	.460	.804	.206	3e-2	.684	.684	.460	
W	.810	.459	.459	.193	.197	.459	.197	.459	.197	.810	.055	.810	.197	.459	.452	.459	
G	.324	.324	.233	5e-2	.139	.823	.482	1e-3	.823	.621	.823	.643	.218	.324	.621	.218	
X	.717	.717	.752	.752	.752	.752	.752	.752	.717	.752	.752	.752	.752	.752	.752	.752	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	

Here, p is the probability that the amino acid occurs in the PDB40 database, and $P^n(k)$ is the probability that the amino acid occurs k times in a data set of n samples (n = 234 for the distribution of every *single* of the sixteen linker positions and n = 234 x 16 for the distribution of the linker *average*). The ratio k/n represents the fraction of the amino acid in the data set. Knowledge of the distribution functions of the amino acids then allows the calculation of P-values from the cumulative distribution function:

$$CDF^n(k) = \sum_{i=0}^{k} P^n(i)$$

The value of $CDF^n(k)$ is the probability that the number of counts of an amino acid in a random data set would be less than k. Consequently, if o and e represent the observed and expected counts, then the two-sided P-value is given by $1\text{-}CDF^n(e+|o\text{-}e|) +$

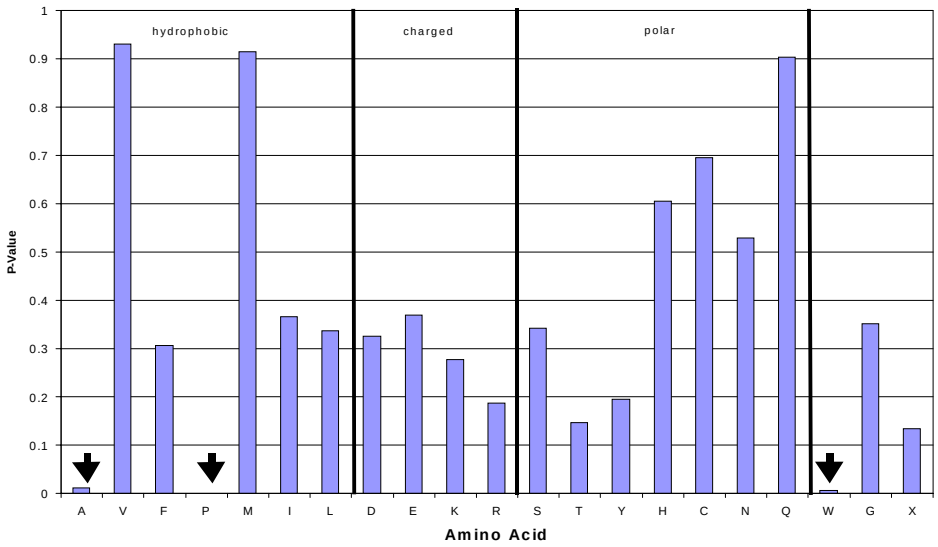


Figure B.9. P-values for the average amino acid compositons in linker sequences. The P-values of alanine, proline, and tryptophan are close to zero. The difference between the content of these amino acids in linkers and protein sequences in general (as represented by the PDB40 database) is statistically significant at better than 98% confidence.

$CDF^n(e-|o-e|)$. This is simply the probability that the number of counts observed in a random subset of PDB40 would take on a value more different from what was expected than what was observed. In order to assign a P-value to an amino acid frequency in the linkers data set, the discrete values of the cumulative distribution function have been linearly interpolated. In most cases, it is also possible to obtain a satisfactory approximation to the P-values by applying the two-sided significance test to the Normal approximation of the Binomial distribution.

Towards Propensities for Flexible Linkers

A variant on this procedure involves focusing just on linkers that are known to be flexible. My Database of Macromolecular Motions contains residue selections for known protein hinge regions (i.e., flexible linkers) that have been culled from the scientific literature. These sequences have been verified manually to be true flexible linker regions,

and thus this database constitutes a potential “gold standard” free from algorithmic biases that can be used as a starting point in the development of propensity scales and other research leading towards algorithmic techniques. By expanding these residue selections slightly with a predetermined protocol and extracting the corresponding sequences from the PDB, a series of sequences of known flexible linkers may be obtained. A FASTA search with a suitable cutoff (e.g., e-value 0.001) may then be performed on known linker sequence to obtain a series of near homologues (Table B.6).These homologues can then be arranged into a multiple alignment (via the CLUSTALW) program[307,308] and the multiple alignment can be fused into a variety of consensus pattern representations, such as Hidden Markov Models or simply consensus sequences[309-313]. A sample multiple alignment for the hinge in calmodulin is shown in Table B.6 and a number of consensus sequences are shown in Table B.5. The amino acid composition may be averaged over all the different hinges and different positions within a hinge to give a single composition vector for flexible hinges. Finally, this can be compared to the overall amino acid composition or that of linkers to obtain a preliminary scale of amino acid propensity in mobile linkers, as shown in Table B.7. This can be compared with the scale of amino acid propensities in linkers as obtained by the procedure previously described and shown in Table B.3.

Table B.5. Example of protein flexible linker consensus sequences extracted from the Macromolecular Movements Database. The database contains residue selections for known hinge regions (flexible linkers) culled from the scientific literature. Sixteen of these residue selections were then "grown" slightly in both directions according to a fixed protocol. Each selection was assigned a linker ID, which is based either on a PDB ID or on the macromolecular movements database motion ID plus possible an optional additional numeric suffix to identify the specific residue selection used. A FASTA search with a cutoff of 0.01 was then performed on each sequence to obtain near homologues. The consensus sequence corresponding to each linker ID is given here.

Linker ID	Linker Consensus Sequence
4cln	MARKMKDTDSE
6ldh	AGARQQEGESRLNLVQRNVNIFKF
adenkin1	VPFEVI
adenkin2	LRLTA
adenkin3	GEPLIQRDDDKE
adenkin4	AYHAQTE
anxbreat	MKGAGT
anxtrp1	YEAGELKWG
anxtrp2	EETIDRET
dt	LFQVVHNS
enolase	GASTGIY
enolase2	SDKS
lfh_hinge1	QTHY
lfh_hinge2	RVPS
ras	AGQEEYSAMRDQYMR
tbsv	PQPTNTL

Conclusion and Future Directions

I have developed a number of database-based techniques for the study of macromolecular motions. I have constructed a database of macromolecular motions, which currently documents ~120 motions, and have developed a classification scheme for the database based on size then packing (whether or not there is motion across a well-packed interface). The database incorporates innovative cooperatively features, allowing authorized remote experts to act as database editors via the Internet. I also developed a standardized nomenclature, such as maximum atomic displacement or degrees of rotation. I am developing automated tools to analyze protein and nucleic acid structures and sequences with possible motions, to extract standardized statistics on

quences with possible motions, to extract standardized statistics on macromolecular motions from structural data, and allow the database to be more readily populated.

I expect that the number of macromolecular motions will greatly increase in the future, making a database of motions somewhat increasingly valuable. My reasoning behind this conjecture is as follows: The number of new structures continues to go up at a rapid rate (nearly exponential). However, the increase in the number of folds is much slower and is expected to level off much more in the future as we find more and more of the limited number of folds in nature, estimated to be as low as 1000[18,104]. Each new structure solved that has the same fold as one in the database represents a potential new motion -- i.e. it is often a structure in a different liganded state or a structurally perturbed homologue. Thus, as we find more and more of the finite number of folds, crystallography and NMR will increasingly provide information about the variability and mobility of a given fold, rather than identifying new folding patterns.

Databases potentially represent a new paradigm for scientific computing. In an (oversimplified!) cartoon view, scientific computing traditionally involved big calculations on fast computers. The aim in these often was prediction based on first principlese.g. prediction of protein folding based on molecular dynamics. These calculations naturally emphasized the processor speed of the computer. In contrast, the new “database paradigm” focuses on small, inter-connected information sources on many different computers. The aim is communication of scientific information and the discovering of unexpected relationships in the data – e.g. the finding that heat shock protein looks like hexokinase. In

contrast to their more traditional counterparts, these calculations are more dependent on disk-storage and networking rather than raw CPU power.

Table B.6.: Example of FASTA results.

This table gives an example of sequences that might be obtained from a FASTA run on a known flexible linker sequence. In this case, the output of one FASTA run on the OWL database using the flexible linker region from Calmodulin (4cln) with a cutoff (e-value) of 0.001

OWL	ID
CALN_CHICK	MARKMKDTDSE
MUSCAMC	MARKMKDTDSE
CALM_PATSP	MARKMKDTDSE
CALM_PYUSP	MARKMKDTDSE
CALM_METSE	MARKMKDTDSE
CALM_STIJA	MARKMKDTDSE
CALM_HUMAN	MARKMKDTDSE
CALM_DROME	MARKMKDTDSE
HSCAM3X1	MARKMKDTDSE
CALM_EMENI	MARKMKDTDSE
CALM_NEUCR	MARKMKDTDSE
CALM_ELEEL	MAKKMKDTDSE
NEUCLMDLN	MARKMKDTDSE
SSO4B01	MARKMKDTDSE
CALL_ARBPU	MARKMKETDSE
CALM_PLECO	MARKMRDTDSE
CALL_HUMAN	MARKMKDTDNE
CALS_CHICK	MARKMRDSDSE
CALM_PHYIN	MARKMKDTDSE
CALM_PNECA	MARKMKDVDSE
CALM_TRYBB	MARKMQDSDSE
CALM_TRYCR	MARKMQDSDSE
S53019	MARKMKDTDSE
TRBCMRSG	MARKMQDSDSE
CALM_HORVU	MARKMKDTDSE
JC1033	MARKMKDTDSE
CAL1_PETHY	MARKMKDTDSE
CAL6_ARATH	MARKMKDTDSE

Table B.7: Flexible Linker Propensity Scale.

A FASTA search with a cutoff of 0.01 was performed on sixteen flexible linker sequences, as described in the text. Amino acid frequency in the flexible linker sequences and their near homologues obtained in the FASTA search were tabulated and divided by the amino acid sequence frequency in the PDB to obtain the preliminary propensities given in this table. (The high propensity shown for methionine may be an artifact arising from methionine's presence as the first residue in many proteins.)

Residue	Propensity
A	1.3268
C	0.1097
D	1.1684
E	1.4702
F	0.5624
G	1.2972
H	0.4806
I	0.4462
K	1.0519
L	0.5303
M	2.6603
N	0.7729
P	0.4051
Q	1.8076
R	1.8013
S	0.8269
T	0.9002
V	0.6865
W	0.308
Y	1.3375

Appendix C: Load-Balancing Bioinformatics Computations using GNU Queue

Introduction

Many bioinformatics computations are too complex to be run on a single CPU; instead, they require the computational resources of a Beowulf-style cluster of loosely coupled workstations. This chapter, based on materials reviewed by and available from the prestigious Internet Engineering Task Force (the *de-facto* standards organization for the Internet) describes GNU Queue, a freely available utility for load-balancing interactive software on Unix clusters that was originally developed by the author while in college (although not written up until now). This software is now the subject of an Internet software development collaboration project involving multiple developers. GNU Queue is scientifically interesting from a Computer Science perspective; it is related to the thesis project in that it is ideal for distributing the sort of easily-parallelizable typically run by bioinformaticists, such as FASTA runs as well as of obvious use in accelerating database calculations as the user base grows. It has been the subject of articles in the technical trade press[314] and has thousands of users around the world. This chapter describes the protocols used by the basic GNU Queue system to communicate between GNU Queue processes. Extensions to GNU Queue, such as the extensive queue_manager extension developed by Texas Instruments, Inc., are described elsewhere; the homepage for GNU Queue, http://www.gnuqueue.org, is a suggested starting point for finding additional published materials on GNU Queue.

In recent years, workstation clusters have become popular. This change is being driven by advancements in computer hardware and their associated economies of scale. The rise of cluster computing means a change in the way UNIX and GNU/LINUX users access their systems. As clusters become larger and more complex, new and existing user abstractions will need to be developed and improved to ensure that users can continue to exploit cluster resources efficiently with minimal re-training. GNU Queue[314-316] expands familiar UNIX user abstractions to implement a batch-queuing system for interactive jobs. I believe GNU Queue will become a popular tool among system administrators, both as a batch queuing system and as a tool for cluster administration.

Popular batch queuing systems in common use today often requiring the training of ordinary users in complicated batch scripting languages. Instead, GNU Queue uses a streamlined, one-line command syntax to submit jobs. It then effectively uses the standard UNIX and GNU/LINUX shell commands to manage remotely executing jobs. Job status can be checked with 'jobs', jobs can be backgrounded and foregrounded with 'bg' and 'fg', the job can be killed with 'kill', and the shell notifies the user when the job has terminated. This is done with local shell job control and signaling through Queue's proxy daemon mechanism. Queue can be used as a local replacement for rsh and ssh to hosts within a cluster under single administrative control. Queue also supports the more traditional email-based load-balancing and distributed batch-processing facilities using a number of criteria to decide where to send jobs. However, by default, a GNU Queue user interacts with remote jobs in the same familiar manner that he or she is used to handling

locally executing jobs. This can significantly reduce user demands upon system administrators for training and batch queuing support.

GNU Queue has a combination of features not found in any of the batch queuing systems[317] in common use today. Like Platform Computing's LSF commercial product[318] is able to load-balance and distributed interactive jobs across a network. Unlike Platform Computing, GNU Queue is non-commercial; its source code may be easily obtained by anyone. Unlike Generic NQS[319], and the Portable Batch System[320], free systems popular because of the sophisticated scripting languages they support, GNU Queue offers a deliberately streamlined usage, in which jobs are submitted in a single command that could easily be implemented as a shell built-in. This latter feature has caused GNU Queue to see some usage in Beowulf[321] clusters.

In a typical installation, GNU Queue is a true cluster management system, whose only form of scheduling is to find a sufficiently unloaded server to run a job. I have added true distributed cluster computing features. Like Condor[322], GNU Queue can perform process migration on a GNU/Linux system with an appropriately patched Linux kernel. This allows GNU Queue to move running processes from one server to another as changing conditions and server loads dictate, allowing for much more flexible and efficient scheduling. Unlike Condor, GNU Queue can also migrate interactive jobs, and its source code is GPL'd making it available to system administrators at non-academic institutions. Because utilization of these features requiring patching the kernel, the cluster management features of GNU Queue remain the most widely used features of the application.

At present, GNU Queue effectively implements two scheduling algorithms. The standard scheduling algorithm is essentially a wide-area algorithm[323]. Individual servers can accept jobs from either clients or other servers. Each server continuously monitors its status as well as the status of several peer machines. Based on this, it decides whether to run the job, hold the job in storage for later action, or transfer the job to one of its peers. Unlike typical batch queuing systems such as LSF, GNQS, PBS[323], there is no "master" scheduler[324]. This feature has an impact on both the relative fault-tolerance and scalability of GNU Queue as compared with these more traditional systems. I am working on an intelligent peer-selection protocol for each server in the hopes of developing a system that can be scaled Internet-wide.

GNU Queue offers a second, more traditional job scheduling algorithm. This comes in the form of the extensive "queue_manager" package developed by Monica Lau of Texas Instruments, which implements a more traditional central manager process. This has the advantage over the standard protocol that the central manager process is able to enforce cluster-wide usage limitations as well as maintain a centralized record of usage statistics. The queue_manager package, supplied with the distribution, may be selected at compile time to make cluster administration easier than with the default scheduler, but potentially at the cost of fault-tolerance and scalability. I hope eventually to merge features from the queue_manager scheduler with the standard scheduler to allow a "wide-area" scheduler that nevertheless maintains a distributed database of cluster statistics and resource usage.

GNU Queue is currently the only non-commercial batch queuing system that supports load balancing of interactive jobs. Unlike many batch queuing systems popular today, Queue's syntax has been deliberately streamlined to make its use much like that of a shell built-in command. I believe that GNU Queue will make a useful tool for both system administration and load balancing while reducing the required user re-training and support.

The protocols described in this chapter include those used to facilitate network load-balancing (including reporting to GNU Queue processes of host load averages, 'virtual load averages', and/or other information used to determine job routing), process input-output, and process control information (two-way signal and termination code reporting).

The homepage for GNU Queue is http://www.gnuqueue.org .

Transport Layer Protocols

By popular demand, implementations of the GNU Queue protocols complaint with this memo wrap their socket communications as application data under RFC-2246 Transport Layer Security (TLS) Protocol[325] socket connections; these, in turn, are based on TCP sockets[326]. This document makes no specification as to the TLS ciphers that should be used, although a combination of 3DES[327], MD5[328], and no compression is sug-

gested when permitted by law. Compliant implementations may elect to use the less secure RC4 cipher protocol[329] or simple plaintext in order to make the code more exportable and less encumbered by legal restrictions, where necessary. The insecure TCP/IP protocol may be substituted for TLS/TCP/IP in experimental implementations of the protocol for testing purposes.

Mutual Authentication Protocol

This memo defines two protocols that require GNU Queue clients and servers to mutually authenticate themselves to one another. This authenticate scheme is used in both the job control file transfer protocol and the rlogin-like protocol.

Upon connecting establishment, both protocols require the process accepting the inbound socket connection normally to check the IP address of the server host to see if it is in an ACL (Access Control List) of hosts allowed to connect to the client. The ACL typically also specifies a unique user ID common to all the hosts in the cluster under which the server process is expected to run. Consequently, the client may optionally attempt to use the identd service[330] to determine the user id of the connecting GNU Queue server process, or, if the server process normally runs as root, check that the originating TLS socket has bound a reserved port. For this reason, if GNU Queue processes are running with full operating system privileges, they should bind a reserved port before connecting to one another.

Following connection establishment in the job control file transfer and rlogin-like protocols, GNU Queue processes authenticate themselves to each using a scheme similar to the HTTP Digest Authentication[331] scheme. The job control file transfer protocol uses a cluster-wide master password as the shared secret; digest authentication is used, because it ensures that the master password is never transmitted in the clear, even when an insecure transport layer protocol implementation is used. The rlogin-like protocol uses a job-specific, one-time cookie as the shared secret. Following Franks[331], the client (server in the digest authentication sense) sends a challenge ("nounce") which is combined with the cookie (shared secret) to generate a hash using a secure hash function. GNU Queue uses SHA1[332], the modified Secure Hash Algorithm, as the hash function, because at the time of writing it was perceived to be more secure than MD5, which was used as the hash function in Franks[331]. The implementation of SHA1 used by GNU Queue (see source code, http://www.gnuqueue.org) converts the 160-bit binary hash into its 40 character ASCII hexadecimal expansion.

A known potential weakness in this approach is that a malicious (or false) client (server in the HTTP Digest sense) could choose which challenge ("nounce") to send to the server and observe the replies. This ability to choose the plaintexts encrypted with the SHA1 algorithm is a form of known plaintext attack, known to make cryptanalysis much easier[333]. The solution is to have the server (Digest authentication client) send a challenge of its own ("cnounce") which is incorporated into the final hash. In GNU Queue, the nounce and cnounce are ASCII and may not be more than 20 characters. This gives the client (server in the Digest authentication sense) much less control over the hash to be

encrypted.

Thus the exchange is:

GNU Queue socket originator (digest authentication client):

ASCII cnounce string<linefeed>

GNU Queue socket acceptor (digest authentication server):

ASCII nounce string<linefeed>

GNU Queue socket originator:

ASCII-SHA1(concat(cookie,nounce,cnounce))<linefeed>

This proves that the two GNU Queue processes both know the shared secret without revealing it to either party or sending it over the network. In the rlogin-like protocol, if the either side fails to send the correct response within a reasonable period of time (10 seconds in a LAN environment is suggested), the client drops the connection and returns to the top of the waiting loop under the assumption that the server is trying to run a left-over job for an old GNU Queue client that has since died. A GNU Queue server, sensing

a lost connection in the rlogin-like protocol, deletes the job control file under this same assumption.

If the connection still exists at this point, the roles are reversed, so that the GNU Queue socket acceptor can authenticate itself to the GNU Queue socket originator.

As above, if the rlogin-like protocol client does not produce a valid response to the server's challenge within some reasonable period of time (10 seconds in a LAN environment is suggested), the server assumes that this is a different GNU Queue client. It drops the connection and deletes the job control file associated with the cookie. The client, sensing the dropped connection, returns to the top of the waiting loop in the rlogin-like protocol. Failed authentication in the job control file transfer protocol is more ominous: the connection is dropped and the error is logged. Like HTTP Digest Authentication the scheme is still vulnerable to man-in- the-middle attacks.

The exchange is:

GNU Queue socket acceptor (now digest authentication client):

ASCII cnounce string<linefeed>

GNU Queue socket originator (digest authentication server):

ASCII nounce string<linefeed>

GNU Queue socket acceptor:

ASCII-SHA1(concat(nounce,cookie,cnounce))<linefeed>

originator:

<null> { or <octal 001> if there is an error }

At this point, both GNU Queue processes have successfully proven that know the shared secret.

The digest authentication scheme used by GNU Queue is essentially a symmetric cryptographic system. TLS supports asymmetric cryptographic key certificates as a means of authentication using a scheme such as RSA. This scheme has some advantages over the shared secret scheme used by GNU Queue. Rather than having both processes know a single shared secret, each process has a private and a public key. The process signs a "certificate" with its cryptographically strong public key. The nature of the asymmetric cryptographic process is such that the other party can verify that the certificate has been signed by the private key using the published public key, but it is extremely hard to generate the private key corresponding to the published public key. Multiple keys could be used. There could be a published "master key" allowing execution of jobs

through an organization, and varying types of "sub-master keys" allowing execution of jobs only within specific departments. Keys could have expiration dates as well, making it possible to gradually phase out older keys. The shared secret scheme used by GNU Queue is much simpler. An identical shared secret master node password must be used on every node in the cluster. However, this scheme is completely adequate for the systems GNU Queue is currently used on. As it relies only a one-way hash, it is unencumbered by legal restrictions attached to public key and even some symmetric key cryptographic algorithms. The protocol has the additional advantage that it still retains some security when used with the insecure but widely available TCP/IP protocol.

Job Control File

In the initial negotiation, the submitting host writes a "job control file." This contains a superset of UNIX environment information about the remote command to be executed, current directory, remote user id, remote group id, additional remote group ids, UNIX environment variables, UNIX nice value, and UNIX resource limits (if supported). (An NT implementation of a GNU Queue client would simply supply reasonable values for the UNIX variables. A user's typical UNIX environment variables might be supplied to the NT client initially by a UNIX program, or might simply be generated synthetically in the manner of the UNIX login program.) It also specifies GNU Queue options, such as whether or not to establish an rlogin-like[334] connection between the running job and the user (see below), whether to allocate a virtual tty for the job, or whether to run the job in batch mode and, if so, whether to mail the output of the batch process to the user.

The exact details of the job control file are important, except that the first null-terminated string in the job control file contains the file's version string, which the job receiving process must support. Also, the file must somewhere contain an ASCII one-time magic cookie, which will subsequently be used for authentication and as a job control file ID. (Care should be taken to ensure that this cookie is as random as reasonably possible, although the means to do this are not discussed in this protocol.) The file assumes an eight- bit format.

The version strings "VERSION0" and "VERSION1" modes of the job control file must be at least partially supported by all versions of GNU Queue compliant with this memo, and is suitable for creating interoperable versions of GNU Queue clients.

These have the following format:

<start of file> "VERSION0" file version specifier (no quotes) <null> ASCII UNIX Username<null> ASCII optional UNIX password<null> or simply <null>

An ASCII job-specific magic cookie, used for authentication<null>

UNIX current directory<null>

Null-terminated ASCII UNIX environmental variable key=value pairs, terminated

by two consecutive <nulls>s

null-terminated strings giving the UNIX arguments of the command to be run <end of file>

<start of file>

"VERSION1"

file version specifier (no quotes)<null>

4-byte UNIX User ID (UID) integer in network ordering ASCII Username<null>

ASCII email address<null>

ASCII job name<null>

ASCII space<null>

or ASCII destination hostname<null>

Binary variable, 0=rlogin-like mode, 1=batch mode An ASCII job-specific magic cookie, used for authentication<null>

4-byte IP address of host with queue process in network ordering 2-byte IP address of port on which queue process is listening (in the rlogin like mode; otherwise this is undefined.)

List of optional null-terminated UNIX environmental variables, terminated by two nulls.

4-byte UNIX audit ID (same as UID except on HP) in network ordering 4-byte

UNIX effective user ID (normally same as UID) in network ordering

4-byte UNIX effective group ID (normally same as group ID)

4-byte UNIX group ID (integer in network byte ordering)

4-byte number of group ids integer in network byte ordering, followed by this number of group id integers in network byte ordering.

4-byte integer in network ordering which is 1 if standard input is a tty, 0 otherwise (This is also set to zero if the user has forcibly disabled tty options, or if the client is running on an OS which does not support ttys.)

4-byte integer in network ordering which is 1 if standard output is a tty, 0 otherwise. 4-byte integer in network ordering which is 1 if standard error is a tty, 0 otherwise.

4-byte integer in networking byte order giving size of terminal data structure, if any, followed by a terminal data structure of this size {0 on non-UNIX flavor systems}.

A list of null-terminated strings giving the arguments of the command to be run remotely, terminated by two consecutive nulls.

A 4-byte integer in network byte ordering giving the umask value of the client's environment (clients on non-UNIX systems should set this to hexadecimal 022).

A 4-byte integer in network byte order setting the nice value of the process (non-UNIX clients set this to decimal 20).

A 4-byte integer in network byte order giving the size of an optional UNIX resource limit data structure. If non-zero, eight resource limit data structures of this size

<end of file>

Node Selection Protocol

Once a valid job control file exists, a GNU Queue server (typically, "queued") or GNU Queue client (typically, "queue") must determine whether to run the job locally or export the job control file to another machine. (The client, "queue" always decides to export the job, if only to the local "queued" server process.)

The node to which the job control file is to be sent is determined by the node selection protocol. This memo describes node selection protocol VERSION0. Future revisions of this document may include more sophisticated node selection protocols, including highly scalable hierarchical querying schemes designed for large networks, or protocols that perform capability queries of potential target nodes.

In VERSION0 of this protocol, however, the cluster is surveyed by simply querying every node in the cluster once per job submitted. This is done by establishing a TLS[325] socket connection to the server on a predetermined port reserved for this purpose.

The format is as follows:

<start of socket information> "QUERY"<linefeed> "VERSION0"<linefeed> (this is the version specifier string, without quotes.) job control file version string<linefeed> from job control file ASCII queue-name<linefeed> <end of socket information>

If the client's IP address is in the server's ACL (Access Control List), the server responds to this stream with a network-ordered (big endian) 4-byte float value in the

standard IEEE 754 single precision floating-point format, usually equivalent to the local C "float" type in either forward or reverse byte ordering. If the job is rejected (e.g., the socket stream does not begin with "VERSION0"<null>, the job control file version is not understood, or the job queue has too many jobs) it may be rejected by returning the magic value 1e08 as the load average. If the client's IP address is not in the server's ACL, it may immediately close the connection.

Note that the load average returned is considered a "virtual load average" calculated specifically for the particular job queue and the protocol version string. It may take any of a number of factors into account, including the traditional operating system load average. Typically, 1e08 is the magic value returned if the batch queue couldn’t start new jobs; e.g., it is already running the maximum number of jobs in that batch queue on that node.

A GNU Queue client typically proceeds by surveying each node in the cluster with this protocol. Usually, it will elect the node that returns the lowest "load average" in the selected batch queue. At this point, it will send the file to the elected node.

This is again done by establishing a TLS/TCP/IP socket connection to the serving process (typically "queued") on the elected host.

The following information is sent:

<start of socket transmission>

"JOBCONTROLFILE"<linefeed>

"VERSION0"<linefeed> protocol version specifier string, without quotes.

"VERSION1"<linefeed>specifying the first string in the job control file (must be VERSION1 in this memo.)

The previously described digest authentication protocol exchange now follows with cluster master password used as cookie.

If authentication is successful, the protocol continues: Name of batch queue run is be run in <linefeed> The integer "-1" sent as signed four byte integer, network order Job control file as binary data <end of socket transmission>

The server responds with a null if all is well; otherwise it responds with a non-null byte. The behavior of the client upon receiving a non-null byte is unspecified. (Typically, it may try another host before giving up with an error to the user.) Also, the behavior of the server in receiving a file from a machine whose IP address is not in the server's access control list is also undefined; normally the connection will simply be terminated.

Note that the first string in the job control file establishes the version number of the file.

The host that receives the job control file may choose to run the job if conditions are favorable. In this case, it becomes the job receiving process, or server, described in my next section, and, depending on the contents of the job control file, may attempt to connect with the original job submitting process listed in the job control file (referred to somewhat confusingly as the "client" process in the next section) via the rlogin-like protocol described immediately below. If this connection is attempted, and the connection fails, the job and job control file may be discarded. Likewise, if the job is run, the job control file is discarded.

Alternatively, after a suitable delay, the node may decide that conditions for running the job are unfavorable. Typically, the node cannot run the job in the specified batch queue because it is already running the maximum number of jobs in that queue, or because the load average has exceeded the maximum load average allowed for starting jobs in this queue. In this case, it may turn itself into a client and follow the above querying protocol to locate a more suitable host. If it finds a more suitable host (one in which a query does not respond return a load of "1e08" for this batch queue), it may act as a client and retransmit the job control file to this new server using the previously described protocol. It is important that the query protocol return "1e08" when it cannot start new jobs in a given batch queue to prevent needless shuttling of jobs between cluster nodes.

Secure Rlogin-like Protocol Description

The GNU Queue protocol can optionally provide a remote-echoed, locally flow-controlled virtual terminal based on TLS/TCP/IP between job submitting process (GNU Queue client and TLS socket server) and job receiving process (GNU Queue server and TLS socket client). This option is controlled by the job control file, which might defeat this feature and instead require that process output be sent back to the user as email, for example. The contact port is configured at compile time, but may be assigned in a future draft of this document. An eight-bit transparent stream is assumed.

The initial exchange involves the mutual digest authentication scheme described in the section "Mutual Authentication Protocol." The job receiving process (typically, the "queued" GNU Queue server process") is the socket originator. The shared secret is the job specific one-time cookie from the job control file. (For this reason, the job control file transmission protocol TLS stream should normally be encrypted.)

Following successful mutual authentication (as indicated by the final null byte from "queued" to the "queue" client), the job receiving process ("queued" server) opens a second outgoing socket and determines the local port number of this socket. This is sent to the client on the original socket as a null terminated ASCII string.

At this point, a connection consisting of two two-way TLS/TCP/IP sockets is established between server and client.

GNU Queue main loop

For the duration of the connection, the client sends its standard input stream to the server by transparently copying it to the first TLS/TCP/IP socket. The client also copies incoming data on this socket transparently to its standard output.

Similarly, the server redirects incoming data on this first socket to the standard input of the running process being remotely controlled by transparently copying it to either the processes' controlling virtual tty or a UNIX pipe to the processes' standard in. Standard output from the process (either directly from the processes' standard output into a UNIX pipe, or via the master end of a controlling virtual tty) is similarly redirected into this socket by verbatim copying.

If a virtual tty is not being used to control the running process, the server is sometimes able to distinguish the running processes' standard output from its standard error. In this case, standard error output is read from a UNIX pipe connected to the running processes' standard error output. This is then send to the client by copying this data verbatim into the second socket.

Signal Information from Client to Server

Client-side implementation

If the client receives a signal from the operating system that supports signals (e.g., the client process receives a UNIX SIGSTOP signal to suspend) it sends this to the server as a simple byte containing the number of the signal sent. (Future implementations may negotiate a signal-translation map between client and server, whereby the client may learn how to translate a signal number into that used by the server. Current implementation used the system numbering found in the RedHat Linux 6.2 operating system, which shares the important signal numbers -- SIGHUP, SIGTERM, SIGKILL, SIGSTOP, SIGCONT, SIGTSTP, SIGPIPE, etc. -- with other flavors of Unix. If the client is running a different operating system and wishes to send a signal which has a different number than its equivalent under Linux, the signal number is translated to the Linux numbering scheme before being sent to the server. Similarly, if the server is not running Linux, it should either translate non-standard signal numbers from Linux to the equivalent under its operating system, or simply ignore numbers for non-standard signals.)

If the client is running on an operating system that does not support signals (e.g., Windows NT), some other means of allowing the user to send signals to the running process is normally provided, such as a graphical user interface listing sensible signals to send to the process being controlled by the server.

If the client receives SIGWINCH (the UNIX terminal window size change signal) and client and server have previously negotiated a window size structure as well as use of

a virtual tty, the SIGWINCH number is followed by the client's new window size structure as a "struct winsize" data structure as implemented under RedHat Linux 6.2; the format of this data structure is not likely to change in future implementations of the Linux operating system. Clients running on non-UNIX operating systems, such as Windows NT, are unlikely to have a useful equivalent of a terminal window size change, and therefore should not send SIGWINCH to the server.

Server-side implementation

If the server receives a byte on the second socket, it should send this signal to the process (typically via a signal() system call under UNIX.) If the signal number matches SIGWINCH and it has negotiated a window size structure (by, e.g., negotiating homogeneous cluster mode) as well as use of a virtual tty, it should first read the Linux "struct winsize" structure from the socket and adjust the virtual tty for the process it is controlling to match the information in the "struct winsize" structure.

Signal Information Flow from Server to Client

The server monitors the process it is controlling (e.g., using the wait() system call) for normal termination and/or termination or suspension by a signal. The process allocates a signed char. If the process terminated or was suspended by a signal, the signal number is recorded as a negative value (under most flavors of UNIX, there are no more than 64 signals). Otherwise, the exit value (under UNIX) is noted; if the exit is less than

zero or greater than 127, the signed char is set to 127. Otherwise, the signed char is set to the exit value code. This signed char (positive or zero for normal exit, negative for termination or suspension by signal) is sent to the client by setting the OOB (Out of Bounds) data marker in the application data stream on the second TLS socket (standard error socket) to the current position and transmitting the byte.

The client is able to distinguish this termination/suspension byte from normal standard error information by monitoring the OOB marker. When it points to a byte in the stream, the sign of the byte is tested. If it is negative, the process controlled by server has received a signal, and the client takes appropriate action (Under GNU Queue for UNIX, the client sends itself the signal, first performing any appropriate signal number mapping in a non-homogeneous environment). If the number is positive, the process controlled by the server has terminated normally, and the client takes appropriate action (GNU Queue clients for UNIX terminate with this value.)

Connection Closure

Normally, the death of the process running under the server's control will trigger the client to terminate via the OOB mechanism just described. The client should ensure all pending socket input and output has been processed before terminating or sending itself a potentially terminal signal. Similarly, the server should ensure there is no more incoming data from the client before sending signals to the process under control.

If the TLS/TCP/IP connection closes abnormally in either direction, the client or server process that notices the close should perform an orderly shutdown, restoring terminal modes (on the client side) and/or killing the running process in an orderly fashion (on the server side by, e.g., a SIGTERM followed a few seconds later by a SIGKILL).

Security Considerations

The GNU Queue protocol (as implemented), like rlogin[334] and rsh, allows a user to set up a class of trusted users and/or hosts which will be allowed to execute jobs as him- or herself without the entry of a password. Also like rlogin and rsh, compromise of one of the trusted hosts opens ALL the systems so configured[335].

Unlike rlogin and rsh however, the GNU Queue protocol (as commonly implemented) requires each the IP address(es) of each trusted host to be explicitly listed in the global Access Control List (wildcards are not supported), which is only supposed to list hosts in the user's immediate cluster. Hosts in a GNU Queue cluster already share certain security-related attributes (such as mounting a common networked filesystem or use shared passwords for ease of use) so this security caveat is less likely to be a major issue for GNU Queue than it is for other protocols, such as rlogin and rsh. While GNU Queue may allow compromise of the entire GNU Queue cluster from a single cluster node, unlike rlogin and rsh this will not, in general, allow compromise of other GNU Queue clusters under separate administrative control.

GNU Queue was originally written with small, local clusters in mind, which can be assumed to have relatively secure networks. In the past, widespread use of plaintext passwords mean that compromise of these networks resulted in compromise of the entire cluster through no fault of GNU Queue. Today, however, widespread use of network switches and secure authentication and communication protocols such as ssh and kerberos means that the GNU Queue protocol could be the weak link in the chain were it not to rely on a secure protocol such as TLS for reasonably secure communications.

Other potential areas of concern include denial-of-service attacks. While already somewhat reduced by the use of IP Access Control Lists in the standard implementation, situating the GNU Queue cluster behind a firewall can further mitigate these risks.

A final concern might include attempts at client or server process spoofing. These spoofing attacks in general require that the malicious party already has shell access to one or both machines -- the malicious party is merely attempting to gain additional privileges. Properly configured, these risks have been addressed by the standard implementation of the protocol. When processes have root privileges available (installed by an administrator) secure ports are required. Otherwise, a facility for identd[330] checking is available, but an identd server must be properly installed in the cluster. A further security precaution involves the use of a job-specific one-time ASCII pad, shared between client and server by means of the secure TLS protocol, to mutually authenticate client and server via a cryptographic digest algorithm. A poor, non-random generation of the one-time pad could compromise this approach, as could insecure communications if the job control file

is transmitted in the clear.

Appendix D: Comparison of Morph Server Analysis with Published Results

Introduction

The problems of protein chemistry and protein motions are sufficiently complicated to require real human intelligence to understand adequately, at least for the foreseeable future. Neither the morph server (Chapter 3)—nor any computer software program, for that matter—could ever hope to replace a human expert, nor was it ever intended to. The question of how close morph server output comes to accepted values is therefore of interest. In this appendix I compare morph server output with previously published results.

The morph server arose out of necessity. The Database of Macromolecular Motions required the development of custom software tools to automate the complex task of finding, analyzing, visualizing, and organizing the many thousands of protein motions in the databases. The morph server was intended to automate some, but not all, of the tasks normally performed by a human expert so as to make the database project tractable.

As with any software program, care must be taken by experimentalists in interpreting morph server output. Whenever possible, server outputs should be manually re-evaluated and validated by a human expert using traditional techniques before using them in publication.

Intended Users

The morph server is intended to do three things:

(1) It is intended to allow users to conduct a systematic analysis of a database of protein motions (potentially, thousands of protein motions extracted from the entire PDB) to determine statistical trends across different categories of protein motions.

(2) It is intended to allow crystallographers to perform a quick, `first-efforts' analysis of new experimental data. One frequently finds in the literature that the solution of multiple conformations of proteins is often sufficient grounds in and of itself for publication. Consequently, these papers often lack important statistics describing the motion. The morphing server provides crystallographers with an easy-to-use, fast, standardized tool for analysis of protein motions when there is insufficient time to have a human perform an expert analysis. It should help standardize and encourage the reporting of key protein motion statistics in the literature.

(3) It is intended to illustrate protein motions as `morph movies' to make protein motions more intuitive. Morph movies can and have provided scientists with new insights into protein motions.[133,177,336,337] Scientists such as Prof. Eric Martz at the University of Massachusetts have also found the morph server to be a useful educational tool and have developed their own specialized interfaces to make the morph server and its graphical output more accessible to the general public.

Input File Cautions

Input files should be carefully selected and checked prior to submission.

(1) Ideally, input files will consist of two pairs of PDB files providing experimental data on two different conformations of the same protein from the same species.

(2) Resolutions of input files should be roughly comparable and of good quality.

(3) Sequence information in the input files should be clear and consistent throughout the PDB files. (The morph server uses PDB files internally because the mmCIF[97] format did not exist at the time the morph server was created. The RCSB PDB recommends obtaining PDB files by downloading them in mmCIF format and converting these to PDB format using free software that they provide. This procedure will result in 'cleaner' and more consistent PDB files than obtaining the older, non-sanitized PDB-format files archived and distributed by the RCSB.)

(4) Currently, the server only morphs individual chains. The chain letter for each conformation should be specified to the server.

Input files containing lots of missing atoms or other imperfections may produce less than desirable results. Similarly, the input files should actually describe a motion (users have been known to submit pairs of PDB files that, while describing different conformations of the same or similar proteins, do not actually involve a real motion.)

As with any scientific computer application, “garbage in, garbage out.” Better quality input data will often result in more accurate morph server output.

Statistical Cautions

Upon successful morphing, the user should check the morph movie to make sure it appears reasonable. Excessive chain-breaks or other unrealistic geometry can indicate a problem with the morph (or the input files), especially if the preceding section on “input file cautions” was not followed.

If the morph looks reasonable, one can then proceed to examine the statistics reported. In addition to the morph movie and the color plots showing the areas involved in the motion, the server generates a plethora of statistics, including torsion angle statistics, energies involved in the transitions, and normal mode statistics.

I discuss three such statistics here. These particular three types of statistics stand out because they have often been calculated manually by experts in the past (by manually superimposing protein structures on a computer workstation, for example) and are frequently published in the scientific literature.

(1) Torsion Angle Changes. I have done a detailed comparison of the server's output and previously published torsion angle change data for adenylate kinase.[338] There is a high degree of agreement (Table D.1).

(2) C-alpha displacement. This measures the largest movement of a C-alpha atom over the course of the motion. It is highly dependent on the superposition algorithm. I use a version of the `sieve-fit' superposition algorithm originally developed by Lesk et al.[71,148] modified to work in a more automatic manner. Because of the quality nature of the superposition algorithm used, I believe the C-alpha numbers the server produces are normally accurate within expected error. Manual measurements have in the past been done by a variety of means (including the technique of using manual manipulation of protein structures on a graphical interface). Also, actual numbers are dependent on the exact structures used in the calculations, and the use of newer, higher-resolution structures can change the numbers somewhat. Therefore, some differences between the server's output and previously published results can be expected (Table D.2).

(3) Rotation around the hinge. Of the motion statistics normally manually determined and reported by experimentalists, this is the hardest to determine using a completely automatic procedure. Obtaining an accurate measurement for rotation around a hinge is difficult because protein motions are never completely rigid-body motions around which one can define and accurately measure a precise geometric concept such as 'rotation.' Moreover, protein backbones are not

straight lines (and sometimes change over the course of the motion), making completely accurate measurement of hinge rotation difficult by any method.

The server, however, has implemented an algorithm which is designed to give a first-approximation to the rotation around a hinge provided the motion involved is a true hinge-motion with reasonably rigid domains. The estimate of rotation returned by the server is likely to be accurate only when the motion involves a hinge motion. Rotation estimates for shear motions are likely to be off. Currently no good algorithms exist to automatically distinguish hinge and shear motions, although attempts have described in the literature[151].

The server's rotation estimate is not intended to replace a scientific expert manually measuring rotation around the hinge, as the scientific expert can compensate for deviations from a true rigid-body protein, as well as compensate for shear effects and other factors which may throw off the algorithm. However, different experts can use slightly different methods in measuring rotation. Whatever its current limitations, the algorithm does have the benefit of offering a standardized number.

Individual examples

LDH

Automatic analysis (our morph ID d1m1da2-d11dm_2) suggest Cα displacement of ~20Å, somewhat larger than the published value of ~11Å Angstroms.[1,71]

The discrepancy for LDH is larger than for the other motions, but still within the expected error. It may be explained by a difference of the structures used or by a difference in procedure. (For example, the structures used in the published calculation may have been superimposed using traditional RMS superposition as opposed to the better "trimmed" RMS superposition used by the server.) In view of the agreement between the server's output and accepted Cα displacement for many other proteins, the discrepancy for LDH, still with expected error limits, should not unduly concern users.

TIM

TIM was determined to have a Cα displacement of approximately 7Å angstroms by expert analysis[1,339], ~5Å angstroms by my software (Table D.2). This difference is within the expected error.

Insulin

The published Cα displacement for insulin is 1.5Å[1,45], identical to within expected error to the morph server's 2Å estimate (Table D.2).

Citrate Synthase

Citrate synthase is a shear mechanism motion (with some hinge character), and consequently the maximum rotation measurement computed by the server is expected to be inaccurate. (If one examines the morph movie, one sees that there is some hinge character to the shear motion, and this may explain the value of six degrees generated.)

The literature reports that citrate synthase undergoes an approximately 10Å shift and a rotation of 28 degrees [84]. The software finds a movement of approximately 12Å, but predicts a rotation of only 6.5 degrees. The error in the rotation calculation should not be surprising as the algorithm is only accurate for hinge motions.

Calmodulin

The morph server predicts a rotation of 140 degrees (as opposed to 154 by expert analysis[1]) and Cα dipslacement of 58 Å (60 Å is the published number[1]) The algorithm also correctly locates one of the hinges in Calmodulin 59-82 (the published hinge is 72-82[1]. Hinge location can be highly subjective, and the algorithm is designed to err on the side of caution by returning large hinge residue selections.).

Conclusions

Automatic analysis of protein motions with the morph server is more or less instant once experimental data is available. Comparison of the numbers computed by the morph server for Calmodulin and other protein motions with accepted results demonstrate that

the morph server's output is more than adequate for use as a preliminary step in the analysis of many protein motions.

Prior to the development of the morph server, analysis of protein motions traditionally required manual examination of protein conformations on a graphics workstation by a human expert. One disadvantage with manually obtained data is that experts may sometimes use different methods and therefore disagree on final numbers. Because of the cost and time required to obtain expert opinions, individual experts will only be able to look at a small fraction of the protein motions in literature. Consequently, published results may not be completely comparable because different experts may have used slightly different methods of analysis.

A principal advantage of the morph server is that it codifies rules and introduces consistency into the analysis of protein motions. Working computer algorithms will always produce the same numbers given the same experimental input data. The morph server algorithm can be applied to the entire PDB, producing results that are at least consistent given the same input data. The algorithm itself may, of course, be called into question, in which case it is hoped that its existence encourages a better algorithm to be developed to replace it.

The morph server produces good quality numbers for maximum Cα displacement, maximum rotation, and torsion angle statistics for most but obviously not all protein motions (Tables D.1 and D.2). It does not replace a human expert.

Of the three frequently published statistics discussed in this text, the morph server has greatest difficultly in determining maximum rotation (Tables D.2 and D.3). Partially, this is because the algorithm is designed to work only for hinge motions. Currently, no good automatic means of distinguishing hinge and shear motions exist, although some attempts have been described in the literature[151]. Automatic determination of maximum rotation angles in protein motions is, in general, going to be difficult due to the algorithmic problems that must be overcome. This is most clearly illustrated in the case of adenylate kinase, where the morph server correctly determines the rotation for one of the two pairs of joints (30°) but fails to realize that a second pair of joints is also involved and that the total rotation is actual 90°—human intervention is required to realize that a two-stage rotation is involved. Still, the morph server produces reasonable estimates of rotations for many hinge motions (Table D.2). One problem with hinge rotation data is that it is not consistently reported in the literature; by providing a fast, easy means of obtaining a preliminary estimate of hinge rotation I hope the existence of the morph server will encourage more scientists to publish this useful statistic.

Although the server's rotation and hinge-finding computations do not replace expert analysis, they can nevertheless provide preliminary information to assist experts in focusing their efforts. An experimentalist with high-quality input coordinates can run his or her data through the morph server, and, if the output produces a sensible morph movie of the motion, can then examine the rotation estimate and hinge information output by the server. If the server predicts a sizeable rotation on a quality morph involving a hinge mo-

tion, the experimentalist can then call in a human expert to begin examining the actual protein motion for a rotation. In this case, the human should examine the protein motion to ensure that shear effects or a multiple stage rotation have not distorted or qualified the morph server's estimate of the rotation involved. Conversely, while a low rotation value for a quality morph of a hinge motion is evidence against a significant rotation, it is not evidence if the motion mechanism involves a significant shear component as the algorithm was designed for hinge motion mechanisms. Therefore, a human expert should always be consulted if information on the rotation is desired for shear motions. Similarly, the morph server provides a good first estimate for the location of hinges in domain motions, but an expert should be consulted prior to publication to verify and, if necessary, refine the morph server's hinge locations.

The results (Table D.2) support my intuitive assessment that the morph server is quite good in determining maximum Cα displacement owing, in part, to the quality superposition algorithm it uses. In the case of LDH, one might ask whether or not the manual determination was in error or used a different set of structures than the automatic determination. The server can also be reasonably relied upon to produce quality torsion angle change statistics (Table D.1).

My recommendations as to data quality of the server statistics discussed in this chapter are summarized in Table D.3. The server produces reasonable estimates of maximum Cα displacements, maximum rotation, and torsion angle changes for most but obviously not all protein motions. It does not replace a human expert. Rather, the server is useful as a

fast preliminary step in the scientific analysis of a protein motion and as a means to make tractable a database-wide analysis of protein motions.

Tables

Table D.1: Comparison of torsion angle analysis

This table gives a detailed comparison of automatic morph server torsion angle analysis for ADK (1ak3 vs. 1ake) against published, manually-determined data (Table 3 in Gerstein et al.[338]). The columns on the left give the results automatically computed by the morph server (morph ID 811597-5540) in the process of generating a morph. For pragmatic reasons, the morph server uses slightly different residue numbering than what was used in the literature.[338] (The morph server will apply artificial intelligence rules to renumber structures intelligently when it encounters inconsistencies in numbering within or between PDB structures, which is what has happened here.) The columns on the right reproduce the published data for the same protein.[338] Torsion angle changes in parentheses indicate that these torsion angle changes cancel $\Delta\psi_i$ is small in magnitude approximately equal to $\Delta\phi_{i+1}$; the server's algorithm mimics the published result.

As one can see, the server's output is highly consistent with the published result. The server's torsion angle tools makes it possible to perform—quickly and on thousands of protein motions—sophisticated analyses similar to those published in the literature on single protein motions.[338]

Residue Number (server)	Residue Name	Δϕ (server)	Δψ (server)		Joint	Residue Name (published)	Δϕ (published)	Δψ (published)
112	PRO	-3.8	1.3		I	115	-4	1
113	ASP	-2.7	(18.7)			116	-3	(19)
114	GLU	(-14.5)	(17.4)			117	(-15)	(17)
115	LEU	(-15.9)	38.4			118	(-16)	38
116	ILE	-12.7	-6.7			119	-13	7
117	VAL	-1.3	1.6			120	-1	2
118	ASP	-6.0	9.0					
119	ARG	3.8	(-5.4)		II	122	4	-5
120	ILE	(7.5)	-6.7			123	7	-7
121	VAL	30.5	-41.5			124	30	-41
122	GLY	-4.5	18.1			125	-5	18
123	ARG	15.1	(9.4)			126	15	9
124	ARG	(-9.8)	-15.9					
125	VAL	3.4	14.6					
126	HIS	-6.3	-2.0					
127	ALA	5.3	-163.6					
128	PRO	162.3	44.0					
129	SER	-26.2	8.8					
130	GLY	-6.1	0.1					
131	ARG	-4.5	7.7					
132	VAL	-11.9	-7.5					
133	TYR	-0.9	24.6					
134	HIS	-5.0	2.5					
135	VAL	3.6	-155.8					
136	LYS	-147.4	-89.2					
137	PHE	39.6	28.8					
138	ASN	-10.9	-19.4					
139	PRO	-1.7	5.6					
140	PRO	-3.3	(2.3)					
141	LYS	(-2.7)	23.3					
142	VAL	-15.7	-0.7					
143	GLU	-27.4	178.7					
144	GLY	-158.8	-32.2					
145	LYS	-2.0	10.3					
146	ASP	-0.6	4.8					
147	ASP	3.2	-1.2					
148	VAL	10.5	-20.1					
149	THR	0.1	12.5					
150	GLY	-8.9	-0.1					
151	GLU	4.2	(-6.3)		III	154	4	-6
152	GLU	(7.4)	(-14.6)			155	7	(-14)
153	LEU	(10.2)	-1.8			156	(10)	-2
154	THR	-5.0	25.1			157	-5	25
155	THR	12.8	16.1			158	13	16
156	ARG	-0.9	-9.9			159	-1	-10
157	LYS	-3.8	174.4			160	-4	-185
158	ASP	170.8	48.0			161	171	48
159	ASP	-7.6	57.8			162	-8	58

160	GLN	-33.3	11.4			163	-33	11
161	GLU	4.1	(9.9)			164	4	(10)
162	GLU	(-8.3)	-6.2			165	(-8)	-6
163	THR	2.1	0.2					
164	VAL	11.1	-8.0					
165	ARG	11.9	-15.3					
166	LYS	2.0	3.0					
167	ARG	-1.1	5.3					
168	LEU	4.7	4.8					
169	VAL	-2.7	-5.6					
170	GLU	-1.4	6.9					
171	TYR	-0.1	-1.9		IV	174	0	-2
172	HIS	-13.9	(11.7)			175	-14	(12)
173	GLN	(-16.0)	7.1			176	(-15)	7
174	MET	-53.2	(25.4)			177	53	(25)
175	THR	(-16.9)	(-28.0)			178	(-17)	(-27)
176	ALA	(17.2)	6.8			179	(17)	7

Table D.2: Comparison of C-alpha displacement and rotation measurements

This table compares morph server output for Cα displacement and rotation measurements for specific protein motions against accepted values from the literature. It also provides additional morph server torsion angle output. I have also included additional automatic data that is sometimes manually determined and included in the scientific literature, such as the maximum $\Delta\phi$, $\Delta\psi$, $\Delta\alpha$ torsion angle changes that take place over the course of the motion. (A detailed comparison of torsion angle change data to previously published results may be found elsewhere (Table 1).) I also include the amino acid residues responsible for the maximum Cα displacement and the maximum torsion angle changes. As the table shows, there is generally very good agreement between the server's output and the accepted values for maximum Cα displacement. Maximum rotation is more difficult to compute in a fully automated fashion. Nevertheless, for hinge motions involving a single rotation, agreement between the server's output and accepted values is generally quite good. The morph server should be more than adequate as a tool for experimentalists making preliminary efforts to obtain these values.

Protein	Maximum Cα Displacement (Å) (Published)[1]	Maximum Cα Displacement (Å) (Morph Server)	Hinge Rotation (degrees) (Published)[1]	Hinge Rotation (degrees) (Morph Server)		Max Δϕ	Max Δψ	Max Δα	Server ID
LDH	~11	20 (281 LYS)	No data	~20		177 (233 ASP)	178 (264 LEU)	180 (266 ARG)	d1mlda1-d1ldm_1
Insulin	~1.5	1.8 (4 GLU)	No data	~6*		80 (9 SER)	62 (1 GLY)	27 (7 CYS)	805030-2760
TIM	~7	5 (253 LYS)	No data	~3		175 (155 GLY)	177 (1 MET)	172 (59 ILE)	tim
Citrate Synthease	~10	12 (311 SER)	28	~7*		167 (52 VAL)	132 (51 LEU)	167 (369 ASN)	cs
Calmodulin	60	58 (117 THR)	148.02	140		142 (75 LYS)	180 (78 ASP)	166 (5 THR)	cm
Glutamate Dehydrogenase		21 (383 GLU)	~13	10		179 (106 SER)	179 (274 ASP)	178 (13 LEU)	d1gdha1-d1psda1
TBSV	~14	13 (67 ILE)	~22	18		177 (21 LEU)	180 (44 SER)	180 (22 ALA)	33905-15471
T4 Lysosyme mutant		13 (53 ASN)	~32	23		94 (136 SER)	96 (135 LYS)	38 (54 CYS)	lzm
Adenylate Kinase		33 (149 THR)	~29 (1st pair of joints) and ~60 (2nd pair of joints)	28		177 (100 GLY)	179 (200 LYS)	178 (187 GLU)	d2ak3a_-d1akea_

*shear motion; algorithmic agreement with published result not expected.

Table D.3: Current data quality guidelines for individual statistics.

Torsion angle analysis and C-alpha displacement are dependent mainly upon the quality of the input data files and possibly upon the quality of the superposition algorithm; quality results can therefore be expected on quality input data. Rotational and hinge finding sometimes requires additional, algorithmically complex judgments to be made, and this reduces the quality of the automatic output. Although not discussed in this appendix, “energy” values for the interpolated intermediate “structures” obtained by the morph server in producing a morph movie are qualitative; while they can sometimes provide useful scientific insights these numbers are at best a rough guide. The server also provides a wealth of non-quantitative information in the form of a morph movie, as well as additional quantitative data not historically found in the literature and therefore not discussed in this appendix.

Analysis	Torsion Angle Analysis	C-alpha Displacement Measurements	Hinge Finding	Rotational Measurements	Intermediate Energies
Data Quality	Excellent	Very good	Good	Good	Fair

Appendix E: Condensed Description of Database and Morph Server

Introduction

Function can be thought of as being linked to structure by means of macromolecular motions (i.e. those of proteins and nucleic acids) are often the essential link between structure and function. Because of their relationship to the principles of protein structure and stability, macromolecular motions, moreover, are of great intrinsic interest. By systematizing and analyzing many of the instances of protein structures solved in multiple conformations, it is now possible to study these motions within a database framework.

This chapter, currently in peer-review elsewhere as a separate paper[340], may be thought of as a technical conclusion or summary of the present work on my comprehensive database of macromolecular motions and its associated suite of software tools. The database is intended to be useful to those studying structure function relationships (in particular, rational drug design[17]) and also those involved in large-scale protein or genome surveys. Shakespeare's "tide in the affairs of men" began to come in around the mid-1990s for a number of reasons: (i) The amount of raw data (known protein structures and sequences homologous to them) was exponentially increasing[19,20], and an appreciable fraction of new structures had non-trivial motions. (ii) The graphical and interactive nature of a database was particularly well suited for presenting macromolecular motions, which are often difficult to represent on a static printed page. (ii) A loose federation of databases had emerged in the structural community, allowing the motions database to connect to variety

of information sources. There had been only one previous attempt made at the systematic classification of protein motions[22].

One of the best and most obvious ways to communicate protein motions is through "movies," especially when they are made available over the web. Vonrhein *et al.*[98,120], Sawaya *et al*, and other groups have made custom movies of protein motions available over the web[121,124-129].

I presented a perspective on how protein motions can be put into standardized, consistent terms. I developed a simple model for protein motions involving rigid-body motion of parts, apply my model to actual cases, and measured how well it fits. An integrated Web server provides tools to compare solved conformations of proteins involved in motion, generates statistics to characterize and classify them into a database, and automatically makes a morph movie to represent them. In addition, the server database links protein motions with custom movies of motions available at other sites, along with my own morphs generated automatically upon request by members of the Internet community by the server. Internet users have used my server and database to analyze a number of structures including human interleukin 5[130], bc1 complex[131,132], glycerol kinase[133,134], and lactoferrin[135,136].

The Web morph server is accessible at: http://bioinfo.mbb.yale.edu/MolMovDB/morph. It is integrated with the Database of Macromolecular Motions[139,140] (http://bioinfo.mbb.yale.edu/MolMovDB later http://www.molmovdb.org) and is also

connected with a variety of tools for aligning protein folds and studying their occurrence in genomes[141-144] as well as being integrated into the Partslist Database (http://www.partslist.org)[215].

The database and its associated suite of software tools have been found useful in a number of contexts[177].

Classifying Protein Motions Hierarchically: The Database of Macromolecular Motions

Unique Motion Identifier

A single protein or nucleic acid can have a number of motions and the same essential motion can be shared amongst different macromolecules. For this reason, each entry is indexed by a *unique motion identifier*, rather than around individual macromolecules.

Attributes of a Motion

Each entry has the following information in addition to the motion identifier:

(i) Classification. A classification number gives the place of a motion in the size and packing classification scheme for motions described below. In addition to its basic classification, a motion can also be annotated as being particularly "similar-to" one in another, or "part-of" or "containing" another motion in the same protein.

(ii) Structures. The identifiers have been made into hypertext link that link indirectly to the structure entries at the RCSB and to sequence and journal cross-references

via the Entrez database[31,32]. Links are also made to related structures via the Structural Classification of Proteins (SCOP)[34].

For most entries I describe the overall motion using standardized numeric terminology, such as the maximum displacement (overall and of just backbone atoms), the degree of rotation around the hinge, and residues with large torsion angle changes when these numbers are available from the scientific literature. (The morph server attempts to automatically compute these values from the structures.). Each entry has links to graphics and movies describing the motion, often depicting a plausible interpolated pathway.

Size Classification

Proteins motions were first ranked in order of their size (subunit, domain, and fragments). Domain motions, such as those in hexokinase or citrate synthase[41,42], provide the most common examples of protein flexibility[1-3]. Usually, the motion of fragments smaller than domains refers to the motion of surface loops, such as the ones in triose phosphate isomerase or lactate dehydrogenase. It can also refer to the motion of secondary structures, such as of the helices in insulin[43-45]. Domain and fragment motions are important for a variety of protein functions, and usually involve portions of the protein closing around a binding site, with a bound substrate stabilizing a closed conformation. Subunit motions are distinctly different, and often involve allosteric effects.

Packing Classification

For fragment and domain protein motions I have systematized the motions on the basis of the packing of atoms inside of proteins, which is a fundamental constraint on protein

structure[56-60]. Interfaces between different parts of a protein are usually packed very tightly. Consequently, two basic mechanisms for protein motions, hinge and shear, are proposed depending on whether or not there is a continuously maintained interface preserved through the motion. A complete protein motion can be built up from a number of these basic motions. For the database, a motion is classified as "Shear" if it is predominately a shear motion and "Hinge" if it is predominately composed of hinge motions.

The shear mechanism basically describes the special kind of sliding motion a protein must undergo if it wants to maintain a well-packed interface; these constraints mean that individual shear motions are constrained to be very small.

(ii) Hinge. When no continuously maintained interface constrains the motion, a hinge motion occurs. Typically, these motions usually occur in proteins with two domains (or fragments) connected by linkers (i.e. hinges) that are relatively unconstrained by packing. The whole motion may be produced by a few large torsion angle changes.

Over 60% of the motions in the database are classified as domain motions, while the hinge mechanism is the most common mechanistic classification in the database, accounting for 45% of the entries. Reflecting the greater ease with which smaller motions can be studied experimentally, a greater percentage of fragment motions have structures for multiple conformations in the motion. Most of the fragment and domain motions in the database fall into the hinge or shear classification.

(i) A special mechanism that is clearly neither hinge nor shear accounts for the motion. An example of this sort of motion is what occurs in the immunoglobulin ball-and-socket joint[72], where the motion involves sliding over a continuously maintained interface (like a shear motion) but because the interface is smooth and not interdigitating the motion can

be large (like a hinge).

(ii) Motion involves a partial refolding of the protein. This usually results in dramatic changes in the overall structure.

(iii) Motion can not yet be classified is a catch-all category.

Subunit motions are classified differently as either allosteric, non-allosteric, or unclassifiable.

(iv) Complex motions. Finally, large protein motions which cannot easily be classified as subunit motions are classified as complex movements. For example, the order-to-disorder transition that the headpiece domain undergoes when it binds DNA. Another example involves a molecule binding between two other domains in the protein, such as observed in the bacterial periplasmic binding proteins[26].

Annotation of Evidence related to the Motion

For every entry in the database, I indicated the evidence behind its description and made a clear distinction between the carefully analyzed, “gold-standard” motions and the much more tentatively understood motions, such those only understood as sequence homologues. The database currently describes approximately 120 "gold standard" motions, as well as larger set of some 6,000 motions automatically culled from the RCSB PDB.

Analyzing and Representing Protein Motions: The Morph Server

Protein motions can be put into standardized, consistent terms. I developed a statistical characterization of macromolecular motions using the significant 'standardized values' that describe each motion, such as maximum atomic displacement or degrees of rotation. My system attempts to describe protein motions as a rigid-body rotation of a small "core" relative to a larger one, using a set of hinges. To ensure all statistics between any two motions are directly comparable, the motion is placed in a standardized coordinate. Although my model can accommodate most protein motions, it cannot accommodate all, and the degree to which a motion can be accommodated provides an aid in classifying it. I perform an adiabatic mapping (a restrained interpolation) between every two conformations. Thousands of examples of protein motions have already been submitted to my server, producing a comprehensive set of statistics.

The morph server is integrated into the main Database of Macromolecular Motions and provides tools to compare solved conformations of proteins involved in motion, generates statistics to characterize and classify them into a database, and automatically makes a morph movie to represent them. In addition, the server presents a database linking protein motions with custom movies of motions available at other sites, along with my own morphs generated automatically by the server upon request by members of the Internet community. My server and database have been used by Internet users to analyze a number of recent structures including human interleukin 5[130], bc1 complex[131,132], glycerol kinase[133,134], and lactoferrin[135,136].

The database contains graphics showing the structures and some representation of the pathway for the motion, in addition to its textual elements. Without special techniques, such as high temperature simulation or Brownian dynamics[99,100], normal dynamics simulations cannot approach the timescales of the large-scale motions in the database. Rather, using the technique of adiabatic mapping, a pathway movie is produced as an interpolation between known endpoints (usually two crystal structures). This is a modification of straight Cartesian interpolation, adding the addition of energy minimization after each Cartesian interpolation step. This procedure produces interpolated frames with much more realistic geometry.

I have developed a Database of Macromolecular motions along with an integrated set of protein conformation comparison tools on the Web for use in conjunction with the database or as a stand-alone, publicly accessible server. The server can produce a useful comparison of the structures involved in protein motions when solved endpoint structures are available. The server then uses an adiabatic mapping technique to generate a visually rendered interpolated pathway, or 'morph', of the motion or evolution of the protein.

The server also collects a number of statistics on the motion, including maximum Cα displacement and maximum rotation around the putative hinge. These are useful both in analyzing and classifying individual proteins and in generating a statistical picture of motions in the motions database as a whole. The software then presents the visual representation, statistics, orientation, alignment, and interpolated coordinates to the user. I

have found the server useful in the analysis of protein motions and anticipate that use of the server will help standardize statistics and nomenclature for protein motions subsequently presented in the scientific literature.

References

1. Gerstein, M., Lesk, A. M. & Chothia, C. Structural Mechanisms for Domain Movements. *Biochemistry* **33**, 6739-6749 (1994).
2. Bennett, W. S. & Huber, R. Structural and Functional Aspects of Domain Motion in Proteins. *Crit. Rev. Biochem* **15**, 291-384 (1984).
3. Janin, J. & Wodak, S. Structural domains in proteins and their role in the dynamics of protein function. *Prog. Biophys. Mol. Biol.* **42**, 21-78 (1983).
4. Schutt, C. E., Kreatsoulas, C., Page, R. & Lindberg, U. Plugging into actin's architectonic socket. *Nat Struct Biol* **4**, 169-72. (1997).
5. Page, R., Lindberg, U. & Schutt, C. E. Domain motions in actin. *J Mol Biol* **280**, 463-74. (1998).
6. Schutt, C. (2001), personal communication.
7. Wade, N. in *New York Times* A1 and 9 (New York, 1997).
8. Donne, D. G. et al. Structure of the recombinant full-length hamster prion protein PrP(29–231): The N terminus is highly flexible. *Proc. Natl. Acad. Sci. USA* **94**, 13452–13457 (1997).
9. Chan, D. C., Fass, D., Berger, J. M. & Kim, P. S. Core structure of gp41 from the HIV envelope glycoprotein. *Cell* **89**, 263-73 (1997).
10. Peretz, D. et al. A conformational transition at the N terminus of the prion protein features in formation of the scrapie isoform. *J Mol Biol* **273**, 614-22 (1997).
11. Harrison, P. M., Bamborough, P., Daggett, V., Prusiner, S. B. & Cohen, F. E. The prion folding problem. *Curr Opin Struct Biol* **7**, 53-9 (1997).

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

13. Stryer, L. *Biochemistry* (W H Freeman and Company, New York, 1995).

14. Lifton, R. in *Chronicle of Higher Education* (2000).

15. Shakespeare, W., Act 4, Scene 3.

16. Gerstein, M. & Krebs, W. A database of macromolecular motions. *Nucleic Acids Res* **26**, 4280-90 (1998).

17. Kuntz, I. D. Structure-Based Strategies for Drug Design and Discovery. *Science* **257**, 1078-1082 (1992).

18. Brenner, S. E., Chothia, C. & Hubbard, T. J. Population statistics of protein structures: lessons from structural classifications. *Curr Opin Struct Biol* **7**, 369-76 (1997).

19. Orengo, C. A., Jones, D. T. & Thornton, J. M. Protein superfamilies and domain superfolds. *Nature* **372**, 631-634 (1994).

20. Holm, L. & Sander, C. Mapping the Protein Universe. *Science* **273**, 595-602 (1996).

21. Williams, N. How to get databases talking the same language [news]. *Science* **275**, 301-2 (1997).

22. Boutonnet, N. S., Rooman, M. J. & Wodak, S. J. Automatic analysis of protein conformational changes by multiple linkage clustering. *J. Mol. Biol.* **253** (1995).

23. Tsai, C. J., Lin, S. L., Wolfson, H. J. & Nussinov, R. A dataset of protein-protein interfaces generated with a sequence-order-independent comparison technique. *Journal Of Molecular Biology* **260**, 604-620 (1996).

24. Bairoch, A. & Boeckmann, B. The Swiss-Prot Protein-Sequence Data-Bank. *Nucl. Acids Res.* **20**, 2019-2022 (1992).

25. Shilton, B., Flocco, M., Nilsson, M. & Mowbray, S. Conformational changes of three periplasmic receptors for bacterial chemotaxis and transport: the maltose-,glucose/galatactose- and ribose-binding proteins. *J. Mol. Biol.* **264**, 350-363 (1996).

26. Vyas, N. K., Vyas, M. N. & Quiocho, F. A. Comparison of the periplasmic receptors for L-arabinose, D-glucose, and D-ribose — structural and functional similarity. *J. Biol. Chem.* **266**, 5226-5237 (1991).

27. Stevens, R. C., Gouaux, J. E. & Lipscomb, W. N. Structural consequences of effector binding to the T state of aspartate carbamoyltransferase: crystal structures of the unligated and ATP- and CTP- complexed enzymes at 2.6 A resolution. *Biochemistry* **29**, 7691-7701 (1990).

28. Stevens, R. C. & Lipscomb, W. N. A molecular mechanism for pyrimidine and purine nucleotide control of aspartate transcarbamoylase. *Proc. Natl. Acad. Sci.* **89**, 5281-5285 (1992).

29. Berman, H. M. et al. The nucleic acid database a comprehensive relational database of three-dimensional structures of nucleic acids. *Biophys J* **63**, 751-759 (1992).

30. Abola, E., Sussman, J., Prilusky, J. & Manning, N. Protein Data Bank archives of three-dimensional macromolecular structures. *Meth. Enz.* **277**, 556-571 (1997).

31. Schuler, G. D., Epstein, J. A., Ohkawa, H. & Kans, J. A. Entrez: Molecular Biology Database and Retrievel System. *Meth. Enz.* **266**, 141-162 (1996).

32. Epstein, J. A., Kans, J. A. & Schuler, G. D. WWW Entrez: A Hypertext Retrieval Tool for Molecular Biology. *2nd Ann. Int. WWW Conf.*, (in press) (1994).

33. Hogue, C. W., Ohkawa, H. & Bryant, S. H. A dynamic look at structures: WWW-Entrez and the Molecular Modeling Database. *Trends Biochem Sci* **21**, 226-9 (1996).

34. Murzin, A., Brenner, S. E., Hubbard, T. & Chothia, C. SCOP: A Structural Classification of Proteins for the Investigation of Sequences and Structures. *J. Mol. Biol.* **247**, 536-540 (1995).

35. Hubbard, T. J. P., Murzin, A. G., Brenner, S. E. & Chothia, C. SCOP: a structural classification of proteins database. *Nucleic Acids Res* **25**, 236-9 (1997).

36. Scott, W. G., Finch, J. T. & Klug, A. The crystal structure of an all-RNA hammerhead ribozyme: a proposed mechanism for RNA catalytic cleavage. *Cell* **81**, 991-1002 (1995).

37. Pley, H. W., Flaherty, K. M. & McKay, D. B. Three-dimensional structure of a hammerhead ribozyme [see comments]. *Nature* **372**, 68-74 (1994).

38. Cate, J. H. et al. Crystal structure of a group I ribozyme domain: principles of RNA packing [see comments]. *Science* **273**, 1678-85 (1996).

39. Rees, B., Cavarelli, J. & Moras, D. Conformational flexibility of tRNA: structural changes in yeast tRNA(Asp) upon binding to aspartyl-tRNA synthetase. *Biochimie* **78**, 624-31 (1996).

40. Ruff, M. et al. Class II aminoacyl transfer RNA synthetases: crystal structure of yeast aspartyl-tRNA synthetase complexed with tRNA(Asp). *Science* **252**, 1682-9 (1991).

41. Remington, S., Wiegand, G. & Huber, R. Crystallographic refinement and atomic models of two different forms of citrate synthase at 2.7 and 1.7 Å resolution. *J. Mol. Biol.* **158**, 111-152 (1982).

42. Bennett, W. S., Jr & Steitz, T. A. Glucose induced conformational change in yeast hexokinase. *Proc. Natl. Acad. Sci. U.S.A.* **75**, 4848-4852 (1978).

43. Abad-Zapatero, C., Griffith, J. P., Sussman, J. L. & Rossman, M. G. Refined Crystal Structure of Dogfish M_4 Apo-lactate Dehydrogenase. *J. Mol. Biol.* **198**, 445-67 (1987).

44. Wierenga, R. K. et al. The crystal structure of the "open" and the "closed" conformation of the flexible loop of trypanosomal triosephosphate isomerase. *Proteins* **10**, 93 (1991).

45. Chothia, C., Lesk, A. M., Dodson, G. G. & Hodgkin, D. C. Transmission of conformational change in insulin. *Nature* **302**, 500-505 (1983).

46. Koshland, D. E. Protein Shape and Biological Control. *Sci. Am.* **229**, 52-64 (1973).

47. Koshland, D. E., Jr. *Proc. Natl. Acad. Sci. USA* **44**, 98-104 (1958).

48. Anderson, C. M., Zucker, F. H. & Steitz, T. Space-filling models of kinase clefts and conformation changes. *Science* **204**, 375-380 (1979).

49. Knowles, J. R. Enzyme catalysis: not different, just better. *Nature* **350**, 121-4 (1991).

50. Sampson, N. S. & Knowles, J. R. Segmental Movement: Definition of the Structural Requirements for Loop Closure in Catalysis by Triosphosphate Isomerase. *Biochemistry* **31**, 8482-8487 (1992a).

51. Knowles, J. R. To build an enzyme... *Phil. Trans. R. Soc. Lond. B* **332**, 115-121 (1991).

52. Perutz, M. Mechanisms of cooperativity and allosteric regulation in proteins. *Quart. Rev. Biophys.* **22**, 139-236 (1989).

53. Evans, P. R. Structural aspects of allostery. *Curr. Opin. Struc. Biol.* **1**, 773-779 (1991).

54. Fermi, G. & Perutz, M. F. *Haemoglobin and Myoglobin* (Claredon Press, Oxford, 1981).

55. Johnson, L. N. & Barford, D. Glycogen Phosphorylase. *J. Biol. Chem.* **265**, 2409-2412 (1990).

56. Richards, F. M. & Lim, W. A. An analysis of packing in the protein folding problem. *Quart. Rev. Biophys.* **26**, 423-498 (1994).

57. Harpaz, Y., Gerstein, M. & Chothia, C. Volume Changes on Protein Folding. *Structure* **2**, 641-649 (1994).

58. Levitt, M., Gerstein, M., Huang, E., Subbiah, S. & Tsai, J. Protein Folding: the Endgame. *Ann. Rev. Biochem.* **66**, 549-579 (1997).

59. Richards, F. M. Calculation of Molecular Volumes and Areas for Structures of Known Geometry. *Methods in Enzymology* **115**, 440-464 (1985).

60. Richards, F. M. Areas, Volumes, Packing, and Protein Structure. *Ann. Rev. Biophys. Bioeng.* **6**, 151-76 (1977).

61. Gregoret, L. M. & Cohen, F. E. Novel method for the rapid evaluation of packing in protein structures. *J Mol Biol* **211**, 959-974 (1990).

62. Hubbard, S. J. & Argos, P. Cavities and packing at protein interfaces. *Protein Science* **3**, 2194-2206 (1994).

63. Hubbard, S. J. & Argos, P. A functional role for protein cavities in domain-domain motions. *J. Mol. Biol.* **261**, 289-300 (1996).

64. Gerstein, M. et al. Domain Closure in Lactoferrin: Two Hinges produce a See-saw Motion between Alternative Close-Packed Interfaces. *J. Mol. Biol.* **234**, 357-372 (1993).

65. Gerstein, M. & Chothia, C. Packing at the Protein-Water Interface. *Proc. Natl. Acad. Sci. USA* **93**, 10167-10172 (1996).

66. Ponder, J. W. & Richards, F. M. Tertiary templates for proteins: use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.* **193**, 775-791 (1987).

67. Lawson, C. L. et al. Flexibility of the DNA-Binding Domains of the *trp* Repressor. *Proteins* **3**, 18-31 (1988).

68. McPhalen, C. A. et al. Domain closure in mitochondrial aspartate aminotransferase. *J. Mol. Biol.* **227**, 197-213 (1992).

69. Olson, A. J., Bricogne, G. & Harrison, S. C. Structure of Tomato Bushy Stunt Virus: The Virus Particle at 2.9 Å Resolution. *J. Mol. Biol.* **171**, 61 (1983).

70. Anderson, B. F., Baker, H. M., Norris, G. E., Rumball, S. V. & Baker, E. N. Apolactoferrin structure demonstrates ligand-induced conformational change in transferrins. *Nature* **344**, 784-787 (1990).

71. Gerstein, M. & Chothia, C. H. Analysis of Protein Loop Closure: Two Types of Hinges Produce One Motion in Lactate Dehydrogenase. *J. Mol. Biol.* **220**, 133-149 (1991).

72. Lesk, A. M. & Chothia, C. Elbow Motion in the immunoglobulins involves a molecular ball and socket joint. *Nature* **335**, 188-190 (1988).

73. Stein, P. & Chothia, C. Serpin tertiary structure transformation. *J. Mol. Biol.* **221**, 615-621 (1991).

74. Bullough, P. A., Hughson, F. M., Skehel, J. J. & Wiley, D. C. Structure of influenza haemagglutinin at the pH of membrane fusion [see comments]. *Nature* **371**, 37-43 (1994).

75. Chasman, D. I., Flaherty, K. M., Sharp, P. A. & Kornberg, R. D. Crystal Structure of Yeast TATA-Binding Protein and Model for Interaction with DNA. *Proc. Natl. Acad. Sci.* **90**, 8174-8178 (1993).

76. Kim, Y., Geiger, J. H., Hahn, S. & Sigler, P. B. Crystal structure of a yeast TBP/TATA-box complex. *Nature* **365**, 512-520 (1993).

77. Newman, M., Strzelecka, T., Dorner, L. F., Schildkraut, I. & Aggarwal, A. K. Structure of Bam HI endonuclease bound to DNA: partial folding and unfolding on DNA binding. *Science (Washington D C)* **269**, 656-663 (1995).

78. Lewis, M. et al. Crystal structure of the lactose operon repressor and its complexes with DNA and inducer. *Science* **271**, 1247-1254 (1996).

79. Chuprina, V. P. et al. Structure of the complex of lac repressor headpiece and an 11 base-pair half-operator determined by nuclear magnetic resonance spectroscopy and restrained molecular dynamics. *J Mol Biol* **234**, 446-462 (1993).

80. Shon, K. J., Kim, Y., Colnago, L. A. & Opella, S. J. NMR studies of the structure and dynamics of membrane-bound bacteriophage Pf1 coat protein. *Science* **252**, 1303-5 (1991).

81. Subramaniam, S., Gerstein, M., Oesterhelt, D. & Henderson, R. H. Electron diffraction analysis of structural changes in the photocycle of bacteriorhodopsin. *EMBO J.* **12**, 1-8 (1993).

82. Schlichting, I. et al. Time-resolved X-ray crystallographic study of the conformational change in Ha-Ras p21 protein on GTP hydrolysis. *Nature* **345**, 309 (1990).

83. Genick, U. K. et al. Structure of a protein photocycle intermediate by millisecond time- resolved crystallography. *Science* **275**, 1471-5 (1997).

84. Lesk, A. M. & Chothia, C. Mechanisms of Domain Closure in Proteins. *J. Mol. Biol.* **174**, 175-91 (1984).

85. Flaherty, K. M., McKay, D. B., Kabsch, W. & Holmes, K. C. Similarity of the three-dimensional structures of actin and the ATPase fragment of a 70-kDa heat shock cognate protein. *Proc. Natl. Acad. Sci. USA* **88**, 5041-5045 (1991).

86. Chik, J. K., Lindberg, U. & Schutt, C. E. The structure of an open state of beta-actin at 2.65 A resolution. *J Mol Biol* **263**, 607-23 (1996).

87. Harlos, K., Vas, M. & Blake, C. F. Crystal Structure of the Binary Complex of Pig Muscle Phosphoglycerate Kinase and Its Substrate 3-Phospho-D-Glycerate. *Proteins: Struc. Func. Genet.* **12**, 133-144 (1992).

88. Blake, C. C. F., Rice, D. W. & Cohen, F. E. A "helix-scissors" mechanism for the hinge-bending conformational change in phosphoglycerate kinase. *Int. J. Peptide Protein Res.* **27**, 443-448 (1986).

89. Rayment, I. et al. Three-dimensional Structure of Myosin Subfragment-1: A Molecular Motor. *Science* **261**, 50-58 (1993).

90. Mangel, W. F., Lin, B. & Ramakrishnan, V. Characterization of an extremely large, ligand-induced conformational change in plasminogen. *Science (Washington D C)* **248**, 69-73 (1990).

91. Gilson, M. K. et al. Open "Back Door" in a Molecular Dynamics Simulation of Acetylcholinesterase. *Science* **263**, 1276-1278 (1994).

92. Hughes, D. mini-SQL program., http://Hughes.com.au (1996).

93. Stallman, R. *GNU Emacs Manual* (Free Software Foundation Inc., Cambridge, MA, 1986).

94. Korth, H. & Silberschatz, A. *Database system concepts, 2nd edition* (McGraw-Hill, New York, 1991).

95. Stonebraker, M. R. & Rowe, L. A. in *Proc. 1986 ACM-ACM-SIGMOD Conf. on Management of Data Int. Conf. on Mgt. of Data* (1986).

96. Rowe, L. A. & Stonebraker, M. R. in *Proceedings of the Thirteenth International Conference on Very Large Data Bases: 1987 13th VLDB* (eds. Stocker, P. M., Kent, W. & Hammersley, P.) 83-96 (Morgan Kaufmann, Los Altos, CA, USA, 1987).

97. Bourne, P. E. et al. The Macromolecular Crystallographic Information File (mmCIF). *Meth. Enz.* **277**, 571-590 (1997).

98. Vonrhein, C., Schlauderer, G. J. & Schulz, G. E. Movie of the structural changes during a catalytic cycle of nucleoside monophosphate kinases. *Structure* **3**, 483-490 (1995).

99. Joseph, D., Petsko, G. A. & Karplus, M. Anatomy of Conformational Change: Hinged 'Lid' Motion of the Triosephosphate Isomerase Loop. *Science* **249**, 1425-1428 (1990).

100. Wade, R. C., Davis, M. E., Luty, B. A., Madura, J. D. & McCammon, J. A. Gating of the active site of triose phosphate isomerase: Brownian dynamics simulations of flexible peptide loops in the enzyme. *Biophys. J.* **64**, 9-15 (1993).

101. McCammon, J. A. & Harvey, S. C. *Dynamics of Proteins and Nucleic Acids* (Cambridge UP, 1987).

102. Brünger, A. T. *X-PLOR 3.1, A System for X-ray Crystallography and NMR* (Yale University Press, New Haven, 1993).

103. Krebs, W., Gerstein, M. The morph server: a standardized system for analyzing and visualizing macromolecular motions in a database framework. *Nucleic Acids Res.* **28**, 1665-1675 (2000).

104. Chothia, C. Proteins — 1000 families for the molecular biologist. *Nature* **357**, 543-544 (1992).

105. Murzin, A. G., Brenner, S. E., Hubbard, T. & Chothia, C. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol* **247**, 536-40. (1995).

106. Schmidt, R. B., Gerstein, M., Altman, R.B. LPFC: an Internet library of protein family core structures. *Protein Science* **6**, 246-248 (1997).

107. Schuler, G. D., Epstein, J. A., Ohkawa, H. & Kans, J. A. Entrez: molecular biology database and retrieval system. *Methods Enzymol* **266**, 141-62 (1996).

108. Altman, R. B., Abernethy, N. F. & Chen, R. O. Standardized representations of the literature: combining diverse sources of ribosomal data. *Ismb* **5**, 15-24 (1997).

109. Bairoch, A., Bucher, P. & Hofmann, K. The prosite database, its status in 1995. *Nucleic Acids Research* **24**, 189-196 (1996).

110. Chen, R. O., Felciano, R. & Altman, R. B. RIBOWEB: linking structural computations to a knowledge base of published experimental data. *Ismb* **5**, 84-7 (1997).

111. Holm, L. & Sander, C. The FSSP database of structurally aligned protein fold families. *Nucleic Acids Res* **22**, 3600-9. (1994).

112. Orengo, C. A., Jones, D. T. & Thornton, J. M. Protein superfamilies and domain superfolds. *Nature* **372**, 631-4. (1994).

113. Meador, W. E., Means, A. R. & Quiocho, F. A. Target enzyme recognition by Calmodulin: 2.4 Å structure of a Calmodulin-Peptide Complex. *Science* **257**, 1251-1255 (1992).

114. Silicon-Graphics. VRML 2 Specification., http://webspace.sgi.com/moving-worlds/Design.html (1996).

115. Povey, S., White, J., Nahmias, J. & Wain, H. Problems of nomenclature. *Nature* **390**, 329. (1997).

116. Kraulis, P. J. MOLSCRIPT - A program to produce both detailed and schematic plots of protein structures. *J. Appl. Cryst.* **24**, 946-950 (1991).

117. Sayle, R. RasMol., (1994).

118. Debrunner, P. G. & Frauenfelder, H. Dynamics of proteins. *Annual Rev. of Phys. Chem.* **33**, 283 (1982).

119. Lipscomb, W. N. Acceleration of reactions by enzymes. *Accounts Chem. Res.* **15**, 232 (1982).

120. Vonrhein, C., Bonisch, H., Schafer, G. & Schulz, G. E. The structure of a trimeric archaeal adenylate kinase. *J Mol Biol* **282**, 167-79 (1998).

121. Sawaya, M. R., Prasad, R., Wilson, S. H., Kraut, J. & Pelletier, H. Crystal structures of human DNA polymerase beta complexed with gapped and nicked DNA: evidence for an induced fit mechanism. *Biochemistry* **36**, 11205-15 (1997).

122. Faerman, C. et al. Site-directed mutants designed to test back-door hypotheses of acetylcholinesterase function. *FEBS Lett* **386**, 65-71 (1996).

123. Ripoll, D. R., Faerman, C. H., Axelsen, P. H., Silman, I. & Sussman, J. L. An electrostatic mechanism for substrate guidance down the aromatic gorge of acetylcholinesterase. *Proc Natl Acad Sci U S A* **90**, 5128-32 (1993).

124. Pande, V. S. & Rokhsar, D. S. Molecular dynamics simulations of unfolding and refolding of a beta- hairpin fragment of protein G [In Process Citation]. *Proc Natl Acad Sci U S A* **96**, 9062-7 (1999).

125. Pande, V. S. & Rokhsar, D. S. Folding pathway of a lattice model for proteins. *Proc Natl Acad Sci U S A* **96**, 1273-8 (1999).

126. Xu, Z., Horwich, A. L. & Sigler, P. B. The crystal structure of the asymmetric GroEL-GroES-(ADP)7 chaperonin complex. *Nature* **388**, 741-750 (1997).

127. Rye, H. et al. Distinct actions of cis and trans ATP within the double ring of the chaperonin GroEL. *Nature* **388**, 792-798 (1997).

128. Xu, Z. & Sigler, P. B. GroEL/GroES: structure and function of a two-stroke folding machine. *J Struct Biol* **124**, 129-41 (1998).

129. Sigler, P. B. et al. Structure and function in GroEL-mediated protein folding. *Annu Rev Biochem* **67**, 581-608 (1998).

130. Verschelde, J. L. et al. Analysis of three human interleukin 5 structures suggests a possible receptor binding mechanism. *FEBS Lett* **424**, 121-6 (1998).

131. Crofts, A. R. et al. Pathways for proton release during ubihydroquinone oxidation by the bc(1) complex. *Proc Natl Acad Sci U S A* **96**, 10021-10026 (1999).

132. Crofts, A. R. & Berry, E. A. Structure and function of the cytochrome bc1 complex of mitochondria and photosynthetic bacteria. *Curr Opin Struct Biol* **8**, 501-9 (1998).

133. Bystrom, C. E., Pettigrew, D. W., Branchaud, B. P., P, O. B. & Remington, S. J. Crystal structures of Escherichia coli glycerol kinase variant S58-->W in complex with nonhydrolyzable ATP analogues reveal a putative active conformation of the enzyme as a result of domain motion. *Biochemistry* **38**, 3508-18 (1999).

134. Feese, M. D., Faber, H. R., Bystrom, C. E., Pettigrew, D. W. & Remington, S. J. Glycerol kinase from Escherichia coli and an Ala65-->Thr mutant: the crystal structures reveal conformational changes with implications for allosteric regulation. *Structure* **6**, 1407-18 (1998).

135. Thompson, A. B. et al. Aerosolized beclomethasone in chronic bronchitis. Improved pulmonary function and diminished airway inflammation. *Am Rev Respir Dis* **146**, 389-95 (1992).

136. Sykes, J. A., Thomas, M. J., Goldie, D. J. & Turner, G. M. Plasma lactoferrin levels in pregnancy and cystic fibrosis. *Clin Chim Acta* **122**, 385-93 (1982).

137. Martz, E. (URL: http://www.umass.edu/microbio/chime/explorer/index.htm, 1999).

138. Schnecke, V., Swanson, C. A., Getzoff, E. D., Tainer, J. A. & Kuhn, L. A. Screening a peptidyl database for potential ligands to proteins with side-chain flexibility. *Proteins* **33**, 74-87 (1998).

139. Gerstein, M. & Krebs, W. A Database of Macromolecular Movements. *Nucl. Acids Res* **26**, 4280 (1998).

140. Gerstein, M. B., Jansen, R., Johnson, T., Park, B. & Krebs, W. in *Rigidity theory and applications* (eds. Thorpe, M. F. & Duxbury, P. M.) 401-442 (Kluwer Academic/Plenum press, 1999).

141. Gerstein, M. A Structural Census of Genomes: Comparing Bacterial, Eukaryotic, and Archaeal Genomes in terms of Protein Structure. *J. Mol. Biol.*, (in press) (1997).

142. Gerstein, M. & Wilson, C. Assessing Annotation Transfer for Genomics: Quantifying the relations between protein sequence, structure, and function through traditional and probabilistic scores. *J Mol Biol* **(in press)** (2000).

143. Gerstein, M. & Levitt, M. Comprehensive Assessment of Automatic Structural Alignment against a Manual Standard, the Scop Classification of Proteins. *Protein Science* **7**, 445-456 (1998).

144. Levitt, M. & Gerstein, M. A Unified Statistical Framework for Sequence Comparison and Structure Comparison. *Proceedings of the National Academy of Sciences USA* **95**, 5913-5920 (1998).

145. Barton, G. J. & Sternberg, M. J. E. A strategy for the rapid multiple alignment of protein sequences: Confidence levels from tertiary structure comparisons. *J Mol Biol* **198**, 327-338 (1987).

146. Barton, G. J. & Sternberg, M. J. E. Flexible protein sequence patterns: A sensitive method to detect weak structural similarities. *J Mol Biol* **212**, 389-402 (1990).

147. Barton, G. J. *Methods in Enzymology* **183**, 403-428 (1990).

148. Lesk, A. M. *Protein Architecture: A Practical Approach* (IRL Press, Oxford, 1991).

149. Gerstein, M. & Altman, R. Average core structures and variability measures for protein families: Application to the immunoglobulins. *J. Mol. Biol.* **251**, 161-175 (1995).

150. Gerstein, M., Lesk, A. & Chothia, C. in *Protein Motions* (ed. Subbiah, S.) (in press) (R G Landes, Austin, TX, 1995).

151. Wriggers, W. & Schulten, K. Protein domain movements: detection of rigid domains and visualization of hinges in comparisons of atomic coordinates. *Proteins* **29**, 1-14 (1997).

152. Maiorov, V. & Abagyan, R. A new method for modeling large-scale rearrangements of protein domains. *Proteins* **27**, 410-24 (1997).

153. Ohkawa, H., Ostell, J. & Bryant, S. MMDB: an ASN.1 specification for macromolecular structure. *Ismb* **3**, 259-67 (1995).

154. Kleywegt, G. J. & Jones, T. A. Where freedom is given, liberties are taken. *Structure* **3**, 535-40 (1995).

155. Kleywegt, G. J. & Jones, T. A. Phi/psi-chology: Ramachandran revisited. *Structure* **4**, 1395-400 (1996).

156. Moffat, K. Time-resolved macromolecular crystallography. *Annu. Rev. Biophys. Biophys. Chem.* **18**, 309-32 (1989).

157. Pesce, M. *VRML* (New Riders, Indianapolis, IN, 1995).

158. in *Description of Portable Document Format* (Adobe Corporation, URL: http://www.adobe.com/supportservice/custsupport/SOLUTIONS/ac76.htm).

159. Brooks, B. R. et al. CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics Calculations. *J. Comp. Chem.* **4**, 187-217 (1983).

160. Colonna-Cesari, F. et al. Interdomain motion in liver alcohol dehydrogenase: structural and energetic analysis of the hinge bending mode. *J. Biol. Chem.* **261**, 15273-15280 (1986).

161. Eklund, H. et al. Structure of a triclinic ternary complex of horse liver alcohol dehydrogenase at 2.9 A resolution. *J. Mol. Biol.* **146**, 561-587 (1981).

162. Ames, J. B. et al. Molecular mechanics of calcium-myristoyl switches. *Nature* **389**, 198-202 (1997).

163. Tanaka, T., Ames, J. B., Harvey, T. S., Stryer, L. & Ikura, M. Sequestration of the membrane-targeting myristoyl group of recoverin in the calcium-free state. *Nature* **376**, 444-7 (1995).

164. Ames, J. B., Tanaka, T., Ikura, M. & Stryer, L. Nuclear magnetic resonance evidence for Ca(2+)-induced extrusion of the myristoyl group of recoverin. *J Biol Chem* **270**, 30909-13 (1995).

165. Sawaya, M. R., Pelletier, H., Kumar, A., Wilson, S. H. & Kraut, J. Crystal structure of rat DNA polymerase beta: evidence for a common polymerase mechanism. *Science* **264**, 1930-5 (1994).

166. Tung, C. S., Harvey, S. C. & McCammon, J. A. Large-amplitude bending motions in phenylalaine transfer RNA. *Biopolymers* **23**, 2173 (1984).

167. Bennett, M. J., Schlunegger, M. P. & Eisenberg, D. 3D domain swapping: a mechanism for oligomer assembly. *Protein Sci* **4**, 2455-68 (1995).

168. Bennett, M. J., Choe, S. & Eisenberg, D. Domain swapping: entangling alliances between proteins. *Proc Natl Acad Sci U S A* **91**, 3127-31 (1994).

169. Gerstein, M. A Protein Motions Database. *Protein Data Bank Quarterly Newsletter* **73**, 2 (July) (1995).

170. Krebs, W. G. & Gerstein, M. The morph server: a standardized system for analyzing and visualizing macromolecular motions in a database framework. *Nucleic Acids Res* **28**, 1665-1675 (2000).

171. Berman, H., M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E. The Protein Data Bank. *Nucleic Acids Res.* **28**, 235-242 (2000).

172. Isralewitz, B., Gao, M. & Schulten, K. Steered molecular dynamics and mechanical functions of proteins. *Curr Opin Struct Biol* **11**, 224-30. (2001).

173. Young, M., Kirshenbaum, K., Dill, K. A. & Highsmith, S. Predicting conformational switches in proteins. *Protein Sci* **8**, 1752-64. (1999).

174. Shaknovich, R., Shue, G. & Kohtz, D. S. Conformational activation of a basic helix-loop-helix protein (MyoD1) by the C-terminal region of murine HSP90 (HSP84). *Mol Cell Biol* **12**, 5059-68. (1992).

175. Dixon, M. M., Nicholson, H., Shewchuk, L., Baase, W. A. & Matthews, B. W. Structure of a hinge-bending bacteriophage T4 lysozyme mutant, Ile3-- >Pro. *J Mol Biol* **227**, 917-33. (1992).

176. Oka, T. et al. Time-resolved x-ray diffraction reveals multiple conformations in the M- N transition of the bacteriorhodopsin photocycle. *Proc Natl Acad Sci U S A* **97**, 14278-82. (2000).

177. Volkman, B. F., Lipson, D., Wemmer, D. E. & Kern, D. Two-state allosteric behavior in a single-domain signaling protein. *Science* **291**, 2429-33. (2001).

178. Tsai, J., Levitt, M. & Baker, D. Hierarchy of structure loss in MD simulations of src SH3 domain unfolding. *J Mol Biol* **291**, 215-25. (1999).

179. Tang, Y. Z., Chen, W. Z. & Wang, C. X. Molecular dynamics simulations of the gramicidin A- dimyristoylphosphatidylcholine system with an ion in the channel pore region. *Eur Biophys J* **29**, 523-34 (2000).

180. Van Belle, D., De Maria, L., Iurcu, G. & Wodak, S. J. Pathways of ligand clearance in acetylcholinesterase by multiple copy sampling. *J Mol Biol* **298**, 705-26. (2000).

181. Wlodek, S. T., Shen, T. & McCammon, J. A. Electrostatic steering of substrate to acetylcholinesterase: analysis of field fluctuations. *Biopolymers* **53**, 265-71. (2000).

182. Daggett, V. & Levitt, M. Realistic simulations of native-protein dynamics in solution and beyond. *Annu Rev Biophys Biomol Struct* **22**, 353-80 (1993).

183. Berneche, S. & Roux, B. Molecular dynamics of the KcsA K(+) channel in a bilayer membrane. *Biophys J* **78**, 2900-17. (2000).

184. Wriggers, W. & Schulten, K. Investigating a back door mechanism of actin phosphate release by steered molecular dynamics. *Proteins* **35**, 262-73. (1999).

185. Wilson, E. B., Decius, J. C. & Cross, P. C. *Molecular Vibrations* (McGraw-Hill, New York, 1955).

186. Levy, R. M. Computer simulations of macromolecular dynamics: models for vibrational spectroscopy and X-ray refinement. *Ann N Y Acad Sci* **482**, 24-43 (1986).

187. Levitt, M., Sander, C. & Stern, P. S. Protein normal-mode dynamics: trypsin inhibitor, crambin, ribonuclease and lysozyme. *J Mol Biol* **181**, 423-47. (1985).

188. van der Spoel, D., de Groot, B. L., Hayward, S., Berendsen, H. J. & Vogel, H. J. Bending of the calmodulin central helix: a theoretical study. *Protein Sci* **5**, 2044-53. (1996).

189. Ma, J., Sigler, P. B., Xu, Z. & Karplus, M. A dynamic model for the allosteric mechanism of GroEL. *J Mol Biol* **302**, 303-13. (2000).

190. Brooks, B. & Karplus, M. Normal modes for specific motions of macromolecules: Application to the hinge-bending mode of lysozyme. *Proc. Natl. Acad. Sci. USA* **82**, 4995-4999 (1985).

191. Duncan, B. S. & Olson, A. J. Approximation and visualization of large-scale motion of protein surfaces. *J Mol Graph* **13**, 250-7. (1995).

192. Hinsen, K., Thomas, A. & Field, M. J. Analysis of domain motions in large proteins. *Proteins* **34**, 369-82. (1999).

193. Miller, D. W. & Agard, D. A. Enzyme specificity under dynamic control: a normal mode analysis of alpha-lytic protease. *J Mol Biol* **286**, 267-78 (1999).

194. Thomas, A., Hinsen, K., Field, M. J. & Perahia, D. Tertiary and quaternary conformational changes in aspartate transcarbamylase: a normal mode study. *Proteins* **34**, 96-112 (1999).

195. Thomas, A., Field, M. J. & Perahia, D. Analysis of the low-frequency normal modes of the R state of aspartate transcarbamylase and a comparison with the T state modes. *J Mol Biol* **261**, 490-506 (1996).

196. Thomas, A., Field, M. J., Mouawad, L. & Perahia, D. Analysis of the low frequency normal modes of the T-state of aspartate transcarbamylase. *J Mol Biol* **257**, 1070-87 (1996).

197. Hinsen, K. Analysis of domain motions by approximate normal mode calculations. *Proteins* **33**, 417-29 (1998).

198. Marques, O. & Sanejouand, Y. H. Hinge-bending motion in citrate synthase arising from normal mode calculations. *Proteins* **23**, 557-60 (1995).

199. de Groot, B. L., Vriend, G. & Berendsen, H. J. Conformational changes in the chaperonin GroEL: new insights into the allosteric mechanism. *J Mol Biol* **286**, 1241-9. (1999).

200. Hayward, S., Kitao, A. & Berendsen, H. J. Model-free methods of analyzing domain motions in proteins from simulation: a comparison of normal mode analysis and molecular dynamics simulation of lysozyme. *Proteins* **27**, 425-37. (1997).

201. Wilson, C. A., Kreychman, J. & Gerstein, M. Assessing annotation transfer for genomics: quantifying the relations between protein sequence, structure and function through traditional and probabilistic scores [In Process Citation]. *J Mol Biol* **297**, 233-49 (2000).

202. Brenner, S., Chothia, C., Hubbard, T. J. P. & Murzin, A. G. Understanding Protein Structure: Using Scop for Fold Interpretation. *Meth. Enz.* **266**, 635-642 (1996).

203. Dubchak, I., Muchnik, I. & Kim, S. H. Protein folding class predictor for SCOP: approach based on global descriptors [In Process Citation]. *Ismb* **5**, 104-7 (1997).

204. Gerstein, M. & Levitt, M. Large-scale Application of a Structural Alignment Method to the SCOP Classification of Proteins: Objective Assessment of Alignability. *J. Mol. Biol.*, (in press) (1997).

205. Hinsen, K. The Molecular Modeling Toolkit: A New Approach to Molecular Simulations. *J. Comp. Chem.*, 79-85 (2000).

206. Ascher, D., Dubois, P. F., Hinsen, K., Hugunin, J. & Oliphant, T. (Lawrence Livermore National Laboratory, Livermore, CA 94566, 2000).

207. Wall, L., Christiansen, D. & Schwartz, R. *Programming Perl* (O'Reilly and Associates, Sebastapol, CA, 1996).

208. Press, W. H., Flannery, B. P., Teukolsky, S. A. & Vetterling, W. T. *Numerical Recipes in C* (Cambridge University Press, Cambridge, 1992).

209. Levy, R., Srinivasan, A., Olson, W. & McCammon, J. Quasi-harmonic method for studying very low frequency modes in proteins. *Biopolymers* **23** (1984).

210. Levy, R., Perahia, D. & Karplus, M. Molecular dynamics of an ff-helical polypeptide: temperature dependance and deviation from harmonic behavior. *Proc. Natl. Acad. Sci. USA* **79**, 1346-1350 (1982).

211. Ripley, B. D. *Pattern recognition and neural networks* (Cambridge University Press, Cambridge ; New York, 1996).

212. Christendat, D. et al. Structural proteomics of an archaeon. *Nat Struct Biol* **7**, 903-9. (2000).

213. Venables, W. N. & Ripley, B. D. *Modern applied statistics with S-PLUS* (Springer, New York, 1997).

214. Krause, A. & Olson, M. *The basics of S and S-Plus* (Springer, New York, 2000).

215. Qian, J. et al. PartsList: a web-based system for dynamically ranking protein folds based on disparate attributes, including whole-genome expression and interaction information. *Nucleic Acids Research* (2001).

216. Chothia, C. Proteins. One thousand families for the molecular biologist. *Nature* **357**, 543-544 (1992).

217. Brenner, S. E., Hubbard, T., Murzin, A., Chothia, C. Gene duplications in H. influenzae. *Nature* **378**, 140 (1995).

218. Wolf, Y. I., Grishin, N.V., Koonin, E.V. Estimating the number of protein folds and families from complete genome data. *J. Mol. Biol.* **299**, 897-905 (2000).

219. The_C._elegans_Sequencing_Consortium. Genome sequence of the nematode C. elegans: a platform for investigating biology. *Science* **282**, 2012-8 (1998).

220. Laskowski, R. A., Hutchinson, E.G., Michie. A.D., Wallace, A.C., Jones, M.L., Thornton, J.M. PDBsum: a Web-based database of summaries and analyses of all PDB structures. *Trends Biochem. Sci.* **22**, 488-490 (1997).

221. Wang, Y., Addess, K.J., Geer, L., Madej, T., Marchler-Bauer, A., Zimmernan, D., Bryant, S.H. MMDB: 3D structure data in Entrez. *Nucleic Acids Res.* **28**, 243-245 (2000).

222. Ball, C. A., Dolinski, K., Dwight, S.S., Harris, M.A., Issel-Tarver, L., Kasarskis, A., Scafe, C.R., Sherlock, G., Binkley, G., Jin, H., Kaloper, M., Orr, S.D., Schroeder, M., Weng, S., Zhu, Y., Botstein, D., Cherry, J.M. Integrating functional genomic information into the Saccharomyces genome database. *Nucleic Acids Res.* **28**, 77-80 (2000).

223. Frishman, D., Heumann, K., Lesk, A., Mewes, H. W. Comprehensive, comprehensible, distributed and intelligent databases: current status. *Bioinformatics* **14**, 551-561 (1998).

224. FlyBase. The FlyBase database of the Drosophila Genome Projects and community literature. *Nucleic Acids Res.* **27**, 85-88 (1999).

225. Tatusov, R. L., Galperin, M.Y., Natale, D.A., Koonin, E.V. The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucleic Acids Res.* **28**, 33-36 (2000).

226. Aach, J., Rindone, W., Church, G.M. Systematic management and analysis of yeast gene expression data. *Genome Res.* **10**, 431-445 (2000).

227. Bader, G. D., Hogue, C.W. BIND--a data specification for storing and describing biomolecular interactions, molecular complexes and pathways. *Bioinformatics* **16**, 465-477 (2000).

228. Xenarios, I., Rice, D.W., Salwinski, L., Baron, M.K., Marcotte, E.M., Eisenberg, D. DIP: the database of interacting proteins. *Nucleic Acids Res.* **28**, 289-291 (2000).

229. Benson, D. A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Rapp, B.A., Wheeler, D.L. GenBank. *Nucleic Acids Res.* **28**, 15-18 (2000).

230. Konopka, A. K. & Martindale, C. Noncoding DNA, Zipf's law, and language [letter]. *Science* **268**, 789 (1995).

231. Flam, F. Hints of a language in junk DNA [news]. *Science* **266**, 1320 (1994).

232. Bornberg-Bauer, E. How are model protein structures distributed in sequence space? *Biophys. J.* **73**, 2393-2403 (1997).

233. Gerstein, M. Patterns of protein-fold usage in eight microbial genomes: a comprehensive structural census. *Proteins* **33**, 518-534 (1998).

234. Gerstein, M. A Structural Census of Genomes: Comparing Eukaryotic, Bacterial and Archaeal Genomes in terms of Protein Structure. *J. Mol. Biol.* **274**, 562-576 (1997).

235. Jeong, H., Tombor, B., Albert, R., Oltvai, Z.N., Barabasi, A.L. The large-scale organization of metabolic networks. *Nature* **407**, 651-654 (2000).

236. Amaral, L. A. N., Scala, A., Barthelemy, M., Stanley, H.E. Classes of small-world networks. *Proc. Natl. Acad. Sci. USA* **97**, 11149-11152 (2000).

237. Orengo, C. A. et al. CATH--a hierarchic classification of protein domain structures. *Structure* **5**, 1093-108 (1997).

238. Holm, L. & Sander, C. Mapping the protein universe. *Science* **273**, 595-603. (1996).

239. Gibrat, J. F., Madej, T. & Bryant, S. H. Surprising similarities in structure comparison. *Curr Opin Struct Biol* **6**, 377-85 (1996).

240. Madej, T., Gibrat, J-F., Bryant, S.H. Threading a database of protein cores. *Proteins* **23** (1995).

241. Bateman, A., Birney, E., Durbin, R., Eddy, S.R., Finn, R.D., Sonnhammer, E.L.L. The Pfam protein families database. *Nucleic Acids Res.* **27**, 260-262 (1999).

242. Henikoff, J. G., Greene, E.A., Pietrokovski, S., Henikoff, S. Increased coverage of protein families with the blocks database servers. *Nucleic Acids Res.* **28**, 228-230 (2000).

243. Schultz, J., Milpetz, F., Bork, P., and Ponting, C.P. SMART, a simple modular architecture research tool: identification of signaling domains. *Proc. Natl. Acad. Sci. USA* **95**, 5857-5864 (1998).

244. Brenner, S. E., Koehl, P., Levitt, M. The ASTRAL compendium for protein structure and sequence analysis. *Nucleic Acids Res.* **28**, 254-256 (2000).

245. Lipman, D. J. & Pearson, W. R. Rapid and sensitive protein similarity searches. *Science* **227**, 1435-1441 (1985).

246. Altschul, S. F., Koonin, E.V. Iterated profile searches with PSI-BLAST--a tool for discovery in protein databases. *Trends Biochem. Sci.* **23**, 444-447 (1998).

247. Brenner, S., Chothia, C. & Hubbard, T. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci. USA* **95**, 6073-6078 (1998).

248. Hegyi, H., Lin, J., Gerstein, M. *submitted* (2000).

249. Gerstein, M., Levitt, M. A structural census of the current population of protein sequences. *Proc. Natl. Acad. Sci. USA* **94**, 11911-11916 (1997).

250. Teichmann, S., Chothia, C., Gerstein, M. Advances in structural genomics. *Curr. Opin. Struc. Biol.* **9**, 390-399 (1999).

251. Gerstein, M., Lin, J. & Hegyi, H. Protein folds in the worm genome. *Pac Symp Biocomput*, 30-41. (2000).

252. Lin, J., Gerstein, M. Whole-genome trees based on the occurrence of folds and orthologs: implications for comparing genomes on different levels. *Genome Res.* **10**, 808-818 (2000).

253. Gerstein, M. How representative are the known structures of the proteins in a complete genome? A comprehensive structural census. *Folding & Design* **3**, 497-512 (1998).

254. Brown, P. O. & Botstein, D. Exploring the new world of the genome with DNA microarrays. *Nat Genet* **21**, 33-7 (1999).

255. Lipshutz, R. J., Fodor, S. P., Gingeras, T. R. & Lockhart, D. J. High density synthetic oligonucleotide arrays. *Nat Genet* **21**, 20-4. (1999).

256. Velculescu, V. E. et al. Characterization of the yeast transcriptome. *Cell* **88**, 243-51 (1997).

257. Jansen, R., Gerstein, M. Analysis of the yeast transcriptome with structural and functional categories: characterizing highly expressed proteins. *Nucleic Acids Res.* **28**, 1481-1488 (2000).

258. Gerstein, M., Jansen, R. The current excitement in bioinformatics-analysis of whole-genome expression data: how does it relate to protein structure and function. *Curr. Opin. Struc. Biol.* (2000).

259. Holstege, F. C., Jennings, E.G., Wyrick, J.J., Lee, T.I., Hengartner, C. J., Green, M.R., Golub, T.R., Lander, E.S., Young, R.A. Dissecting the regulatory circuitry of a eukaryotic genome. *Cell* **95**, 717-728 (1998).

260. Roth, F. P., Hughes, J. D., Estep, P.W., Church, G. M. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotechnology* **16**, 939-945 (1998).

261. Jelinsky, S. A., Samson, L.D. Global response of Saccharomyces cerevisiae to an alkylating agent. *Proc. Natl. Acad. USA.* **96**, 1486-1491 (1999).

262. Park, J., Lappe, M., Teichmann, S.A. Mapping Protein Family Interactions: Intra- and Intermolecular Interactions Repertoires are Distinct. *J. Mol. Biol.*, (in press) (2000).

263. Teichmann, S. A., Park, J., Chothia, C. Structural assignments to the Mycoplasma genitalium proteins show extensive gene duplications and domain rearrangements. *Proc. Natl. Acad. Sci. USA* **95**, 14658-14663 (1998).

264. Teichmann, S., Chothia, C., Church, G., Park, J. Fast assignment of protein structures to sequences using the intermediate sequence library PDB-ISL. *Bioinformatics* **16**, 117-124 (2000).

265. Uetz, P. et al. A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. *Nature* **403**, 623-7. (2000).

266. Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Nishizawa, M., Yamamoto, K., Kuhara, S., Sakaki, Y. Toward a protein-protein interaction map of the budding yeast: A comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proc. Natl. Acad. Sci. USA* **97**, 1143-1147 (2000).

267. Ross-Macdonald, P. C., P.S.R., Roemer, T., Agarwal, S., Kumar, A., Jansen, R., Cheung, K., Sheehan, A., Symoniatis, D., Umansky, L., Heidtman, M., Nelson, F.K., Iwasaki, H., Hager, K., Gerstein, M., Miller, P., Roeder, G.S., Snyder, M. Large-scale analysis of the yeast genome by transposon tagging and gene disruption. *Nature* **402**, 413-418 (1999).

268. Harrison, P., Echols, N., Gerstein, M. Digging for Dead Genes: An Analysis of the Characteristics of the Pseudogene Population in the C. elegans Genome. *Nucleic Acids Res.* **29**, 818-830 (2001).

269. Hegyi, H. & Gerstein, M. The relationship between protein structure and function: a comprehensive survey with application to the yeast genome. *J Mol Biol* **288**, 147-64. (1999).

270. Schwikowski, B., Uetz, P. & Fields, S. A network of protein-protein interactions in yeast. *Nat Biotechnol* **18**, 1257-61. (2000).

271. Knuth, D. *The Art of Computer Programming: vol 3, Sorting and Searching* (Addison-Wesley, Reading, MA, 1973).

272. Bairoch, A. The ENZYME data bank. *Nucleic Acids Res.* **21**, 3155-3156 (1993).

273. Riley, M., Labedan, B. in *Escherichia coli and Salmonella: Cellular and Molecular Biology* (ed. Neidhardt, F., Curtiss, III, R., Lin, E.C.C., Ingraham, J., Low, K.B., Magasanik, B., Reznikoff, W., Riley, M., Schaechter, M., Umbarger, H.E.) 2118-2202 (ASM Press, Washington D.C., 1996).

274. Ashburner, M. et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet* **25**, 25-9 (2000).

275. Schmidt, R., Gerstein, M. & Altman, R. LPFC: An Internet Library of Protein Family Core Structures. *Prot. Sci.* **6**, 246-248 (1997).

276. Weber, D. J., Serpersu, E. H., Gittis, A. G., Lattman, E. E. & Mildvan, A. S. NMR docking of the competitive inhibitor thymidine 3',5'-diphosphate into the X-ray structure of staphylococcal nuclease. *Proteins* **17**, 20-35. (1993).

277. Kombo, D. C., Young, M. A. & Beveridge, D. L. One nanosecond molecular dynamics simulation of the N-terminal domain of the lambda repressor protein. *Biopolymers* **53**, 596-605. (2000).

278. Dinner, A. R., Sali, A., Smith, L. J., Dobson, C. M. & Karplus, M. Understanding protein folding via free-energy surfaces from theory and experiment. *Trends Biochem Sci* **25**, 331-9. (2000).

279. Liwo, A. et al. Comparison of the low energy conformations of an oncogenic and a non- oncogenic p21 protein, neither of which binds GTP or GDP. *J Protein Chem* **13**, 237-51. (1994).

280. Drawid, A., Jansen, R., Gerstein, M. Genome-wide analysis relating expression level with protein subcellular localization. *Trends Genet.* **16**, 426-429 (2000).

281. Park, J. et al. Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J Mol Biol* **284**, 1201-10. (1998).

282. Spellman, P. T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K. Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell* **9**, 3273-3297 (1998).

283. DeRisi, J. L., Iyer, V.R., and Brown P.O. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* **278**, 680-686 (1997).

284. Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P.O., Herskowitz, I. The transcriptional program of sporulation in budding yeast. *Science* **282**, 699-705 (1998).

285. Richmond, C. S., Glasner, J.D., Mau, R., Jin, H., Blattner, F.R. Genome-wide expression profiling in Escherichia coli K-12. *Nucleic Acids Res.* **27**, 3821-3835 (1999).

286. Wixon, J., Blaxter, M., Hope, I., Barstead, R., Kim, S. Caenorhabditis elegans. *Yeast* **17**, 37-42 (2000).

287. Holm, L. & Sander, C. The FSSP database of structurally aligned protein fold families. *Nuc. Acid Res.* **22**, 3600-3609 (1994).

288. Voronoi, G. F. Nouveles applications des paramétres continus à la théorie des formes quadratiques. *J. Reine Angew. Math.* **134**, 198-287 (1908).

289. Bernal, J. D. & Finney, J. L. Random close-packed hard-sphere model II. Geometry of random packing of hard spheres. *Disc. Faraday Soc.* **43**, 62-69 (1967).

290. Richards, F. M. The Interpretation of Protein Structures: Total Volume, Group Volume Distributions and Packing Density. *J. Mol. Biol.* **82**, 1-14 (1974).

291. Kleywegt, G. J. & Jones, T. A. Detection, Delineation, Measurement and Display of Cavities in Macromolecular Structures. *Acta Cryst.* **D50**, 178-185 (1994).

292. Chothia, C. Structural invariants in protein folding. *Nature* **254**, 304-308 (1975).

293. Janin, J. & Chothia, C. The Structure of Protein-Protein Recognition Sites. *J. Biol. Chem.* **265**, 16027-16030 (1990).

294. Finney, J. L. Volume Occupation, Environment and Accessibility in Proteins. The Problem of the Protein Surface. *J. Mol. Biol.* **96**, 721-732 (1975).

295. Finney, J. L., Gellatly, B. J., Golton, I. C. & Goodfellow, J. Solvent Effects and Polar Interactions in the Structural Stability and Dynamics of Globular Proteins. *Biophys. J.* **32**, 17-33 (1980).

296. Gerstein, M., Tsai, J. & Levitt, M. The volume of atoms on the protein surface: Calculated from simulation, using Voronoi polyhedra. *J. Mol. Biol.* **249**, 955-966 (1995).

297. Tsai, J., Gerstein, M. & Levitt, M. Keeping the shape but changing the charges: A simulation study of urea and its iso-steric analogues. *J. Chem. Phys.* **104**, 9417-9430 (1996).

298. Tsai, J., Gerstein, M. & Levitt, M. Estimating the size of the minimal hydrophobic core. *Protein Science*, (in press) (1997).

299. Finney, J. L. Volume Occupation, Environment, and Accessibility in Proteins. Environment and Molecular Area of RNase-S. *J. Mol. Biol.* **119**, 415-441 (1978).

300. David, C. W. Voronoi Polyhedra as Structure Probes in Large Molecular Systems. *Biopolymers* **27**, 339-344 (1988).

301. Shih, J. P., Sheu, S. Y. & Mou, C. Y. A Voronoi Polyhedra Analysis of Structures of Liquid Water. *J. Chem. Phys.* **100**, 2202-2212 (1994).

302. Sibbald, P. R. & Argos, P. Weighting Aligned Protein or Nucleic Acid Sequences to Correct for Unequal Representation. *J. Mol. Biol.* **216**, 813-818 (1990).

303. O'Rourke, J. *Computational Geometry in C* (Cambridge UP, Cambridge, 1994).

304. Procacci, P. & Scateni, R. A General Algorithm for Computing Voronoi Volumes: Application to the Hydrated Crystal of Myoglobin. *Int. J. Quant. Chem.* **42**, 151-1528 (1992).

305. Gellatly, B. J. & Finney, J. L. Calculation of Protein Volumes: An Alternative to the Voronoi Procedure. *J. Mol. Biol.* **161**, 305-322 (1982).

306. Branden, C. & Tooze, J. *Introduction to Protein Structure* (Garland Publishing Incorporated, New York, 1991).

307. Thompson, J. D., Higgins, D. G. & Gibson, T. J. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. *Nuc. Acid. Res.* **22**, 4673-4680 (1994).

308. Higgins, D. G., Thompson, J. D. & Gibson, T. J. Using CLUSTAL for multiple sequence alignments. *Methods Enzymol* **266**, 383-402 (1996).

309. Sonnhammer, E. L., Eddy, S. R., Birney, E., Bateman, A. & Durbin, R. Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Res* **26**, 320-2 (1998).

310. Krogh, A., Brown, M., Mian, I. S., Sjölander, K. & Haussler, D. Hidden Markov Models in Computational Biology: Applications to Protein Modelling. *J. Mol. Biol.* **235**, 1501-1531 (1994).

311. Eddy, S. R. Hidden Markov models. *Curr. Opin. Struc. Biol.* **6**, 361-365 (1996).

312. Eddy, S. R., Mitchison, G. & Durbin, R. Maximum Discrimination Hidden Markov Models of Sequence Consensus. *J. Comp. Bio.* **9**, 9-23 (1994).

313. Baldi, P., Chauvin, Y. & Hunkapiller, T. Hidden Markov Models of Biological Primary Sequence Information. *Proc. Natl. Acad. Sci.* **91** (1994).

314. Krebs, W. G. GNU Queue. *Linux Journal* **79** (2000).

315. Greve, G. in *Linux Magazin (Germany)* (2000).

316. Krebs, W. *The GNU Queue Manual* (The Free Software Foundation, Inc., Boston, 2000).

317. Baker, M., Fox, G. & Yau, H. (Syracuse University, Syracuse, NY, 1995).

318. Platform Computing, I.

319. Herbert, S. (The University of Sheffield, Sheffield, UK).

320. Henderson, R. & Tweten, D. (NASA Ames Research Center, Ames, IA, 1995).

321. Becker, D. et al. (NASA, 1995).

322. Livny, M. The Condor Distributed Processing System. *Dr Dobbs Journal*, 40-48 (1995).

323. James, H. in *Computer Science* (University of Adelaide, Adelaide, 1999).

324. Ramme, F. & Kremer, K. in *IEEE Int. Symp. on High-Performance Distributed Computing* 106-113 (San Francsico, 1994, 1994).

325. Dierks, T. & Allen, C. (The Internet Engineer Task Force, 1999).

326. Stevens, W. *Unix Network Programming*.

327. NIST. (N.I.S.T, 1980).

328. Rivest, R. (The Internet Engineering Task Force, 1992).

329. Thayer, R. & Kaukonen, K. (RSA Data Encryption, Inc.).

330. StJohns, M. (The Internet Engineering Task Force, 1985).

331. Franks, J. (The Internet Engineering Task Force, 1999).

332. NIST. (N.I.S.T., Washington , D.C., 1995).

333. Kaliski, B. & Robshaw, M. in *CryptoBytes* (RSA, Inc., 1995).

334. Cantor, B. (The Internet Engineering Task Force, 1991).

335. Garfinkel & Spafford. *Practical Unix Security*.

336. Bourne, P. E., Murray-Rust, J. & Lakey, J. H. Lipids membrane proteins engineering and design. *Curr Opin Struct Biol* **9**, 423-4. (1999).

337. Anonymous. in *Science* 871 (1999).

338. Gerstein, M., Schulz, G. & Chothia, C. Domain Closure in Adenylate Kinase: Joints on Either Side of Two Helices Close Like Neighboring Fingers. *J. Mol. Biol.* **229**, 494-501 (1993).

339. Wierenga, R. K., Noble, M. E. M. & Davenport, R. C. Comparison of the Refined Crystal Structures of Liganded and Unliganded Chicken, Yeast and Trypanosomal Triosephosphate Isomerase. *J. Mol. Biol.* **224**, 1115-1126 (1992).

340. Krebs, W. & Gerstein, M. in *Polymer and Cell Dynamics: Multiscale Modelling and Numerical Simulations* (eds. Alt, W., Chaplain, M., Griebel, M. & Lenz, J.) (Bonn, Germany, 2000).