Kimberly Noonan
David O'Brien
Jack Snoeyink

University of North Carolina
Chapel Hill, NC 27599, USA
snoeyink@cs.unc.edu

Probik: Protein Backbone Motion by Inverse Kinematics

Abstract

To investigate the parameters of the protein design problem that we are exploring in collaboration with biochemists, we have developed a tool that uses inverse kinematics to support moving small fragments of protein backbone, while respecting biochemists' desires to "remain in favorable regions of the Ramachandran plot" and "preserve ideal geometry". By presenting estimates of derivatives in response to motion, we are able to refine these qualitative desires as we work with our collaborators. We then explore low-dimensional bases to parametrize the space of backbone motions.

KEY WORDS—protein design, backbone geometry, six revolute joint manipulator, Denavit-Hartenberg frames

1. Introduction

Proteins are fascinating polymers. Their various structures at the molecular level create the different macroscopic properties like those of hair, skin, shell, and bone. Protein structure is a critical determinant of their function as recognizers, cleavers, splicers, motors, pumps, and many more. Many of the models of proteins (hard sphere, kinematic chain with rotatable bonds, HP-lattice; see Leach 1996) are essentially geometric, so it is no surprise that computer scientists and roboticists have been tackling problems such as protein folding (Song and Amato 2001; Zhang and Kavraki 2002), docking (Teodoro, Phillips, and Kavraki 2001), rigidity and flexibility (Teodoro, Phillips, and Kavraki 2003), and computing volume and surface area (Edelsbrunner and Koehl 2003) and normal modes (Amadei et al. 1996; de Groot et al. 1996; Kobayashi, Yamato, and Go 1997; Teodoro, Phillips, and Kavraki 2003).

Because the biological sciences have a history of being descriptive, there is a gap to be bridged when trying to apply mathematical and engineering techniques to their models. Biologists tell stories about their favorite proteins, reactions, or mechanisms. One consequence is that their models are often overly simplified—they are chosen as good verbal descriptors or analogies, and not for their mathematical completeness. When working with experienced biologists, there are always unstated qualifiers or exceptions: backbone bonds are rigid, except that all the atoms are in motion; peptide bonds are planar, plus or minus 5 degrees; atom interpenetration does not happen, except when forced by strain or improper modeling in some other area of the protein. The only way to bridge this gap is by close collaboration that makes the biologist part of the solution.

In this paper we focus on backbone adjustment within the processes of crystallographic refinement and protein design. In these processes a biochemist seeks to modify a protein's backbone to fit experimentally observed data, or to present the appropriate chemistry to perform some function. We give a brief introduction to the geometry of protein backbones in Section 2. Current approaches usually generate many substructures (loop conformations) satisfying kinematic constraints, which are then evaluated using sterics (collisions) and various energetic potentials, and sometimes clustered and presented to a user for selection. We review some of these methods, and review exact inverse kinematics solutions for six degrees of freedom (DOF) rotational manipulators in Section 3.

Our contribution (Section 4) is to use exact inverse kinematics to communicate not just a single solution, but families and spaces of solutions to our biochemistry collaborators. We can then learn why they would consider one family of solutions preferable to another, and refine our models and solution criteria. The final validation will have to come from new biological hypotheses under test, which we hope will be published in the biological literature (even though the mathematics and computation will be barely sketched in the "Methods" section—another gap). Since we believe that our methods that fill gaps have merit, we use this paper to describe the computational steps in our tool, Probik. Finally, (Section 5) we explore how few parameters are sufficient to cover the space of backbone motions for protein design tasks.

The International Journal of Robotics Research
Vol. 24, No. 11, November 2005, pp. 971-982,
DOI: 10.1177/0278364905059108

Figures 1–3 appear in color online: http://ijr.sagepub.com

2. Background: Protein Geometry and Backbone Adjustment

In this section, we provide basic definitions and terminology of protein backbone geometry and discuss the ideal versus observed geometric parameters. The reader who wishes to explore more than this basic geometry of proteins is encouraged to see Branden and Tooze (1999) or Petsko and Ringe (2004) for in-depth introductions to protein structure from a biological perspective.

We also provide an overview of where the biochemist is still an essential part of the crystallographic refinement and protein design processes, even with sophisticated computational tools such as Dezymer (Looger et al. 2003; Dwyer, Looger, and Hellinga 2004).

2.1. Protein Geometry

Proteins are built from amino acids that are chemically linked by peptide bonds to form a polypeptide chain. Each amino acid contributes five atoms to the backbone, including a chain of repeating –N–C^α–C– units (nitrogen–carbon–carbon). The carbonyl carbon C is double bonded to an oxygen atom and the nitrogen atom N is bonded to a hydrogen atom. The side chain attached to the C^α is one of 20 different organic compounds that determines the amino acid type. In this paper we focus on the protein backbone, so we treat each residue as an alanine, whose side chain is a methyl group bonded to the C^α atom, namely, a C^β carbon and three hydrogen atoms. It is common to suppress hydrogens in depictions of protein structure.

The backbone can also be partitioned into peptide units, C_i^α–C–N–C_{i+1}^α (Figure 1). The bonds of the peptide units are generally considered to be planar and of fixed length. The rotations of the peptide planes about the bonds formed at the C^α atom, however, exhibit large variation within proteins, allowing the peptide chain to assume a variety of secondary structures such as alpha helices and beta sheets. By convention, the rotation around N–C^α and C^α–C bonds are denoted by ϕ and ψ (Figure 1). Many combinations of ϕ and ψ are infeasible due to interatomic collisions, called sterics in chemistry. The ϕ, ψ distribution of data points for the non-glycine and non-proline residues of the “top500” known protein structures are plotted against one another in a Ramachandran plot (Figure 2). The “top500” proteins were selected from the Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB; http://www.rcsb.org/pdb/) by the Richardson Laboratory (http://kinemage.biochem.duke.edu/) based on the resolution and accuracy of the structures.

Although the peptide units are often considered to have a fixed geometry, variation of the bond lengths and bond angles is observed in known protein structures; Table 1 includes statistics obtained from 331 of the “top500” proteins. (169 were not considered due to sequence gaps or other inconsistencies.) For each of the 331 structures, we calculated the average value and standard deviation of the backbone bond angles and bond lengths. Displayed in Table 1 are the corresponding average values and average standard deviations of these parameters across the 331 structures. The average standard deviation represents the expected variation of the parameter value for a given protein structure and a given amino acid in the protein. Table 1 also includes the bond parameter variations allowed in Probik and the standard ideal bond geometries from Engh and Huber (1991).

2.2. Backbone Adjustment in Crystallography and Protein Design

Protein design has enormous potential application for the development of drugs and biosensors. Hellinga has used his computational tool, Dezymer, to redesign the binding site for a protein ligand complex to create a proteins with new function. Among other results, his laboratory has successfully redesigned ribose-binding protein to bind trinitrotoluene (TNT; Looger et al. 2003) and designed an enzyme (Dwyer, Looger, and Hellinga 2004).

The general principles for the formation of specific protein/ligand complexes are often depicted as a lock and key fit, determined primarily by short-range interactions (steric contacts and hydrogen bonds). For a protein to bind a new ligand, the chemistry and geometry of the binding pocket must be changed without destabilizing the protein (i.e., preventing its fold). This change can be effected only indirectly, by changing the sequence of amino acids. The Dezymer software identifies a set of amino acids to change under the assumption that the backbone will remain fixed, then performs a large combinatorial search of ligand translations and rotations inside the pocket (approximately 10^8) and mutations of amino acid residues (typically 12–18 residues, corresponding to 10^{45} to 10^{68} mutant structures representing 10^{15} to 10^{25} sequences; Looger et al. 2003). It ranks feasible pocket designs, using a series of energy and potential functions.

We know that changing the sequence will move the backbone of the protein, too, but Dezymer’s analysis does not currently include backbone modification. In fact, attempts to modify protein backbones have always tended to destabilize proteins, and so are avoided.

One exception is the backbone manipulations by the biochemists of the Richardson Laboratory (http://kinemage.biochem.duke.edu/), which are designed to correct crystallographic misfittings in the PDB. In using their MolProbity suite (Davis et al. 2004) to analyze all-atom contacts in protein structures in the PDB, they have observed some bad clashes that can be removed with a small backbone motion, and possibly an alternative rotamer at a sidechain. They are studying alternate conformations of backbone atoms and Cβ atoms in high-resolution structures to characterize the backbone motions observable in native proteins (Davis et al. 2005).

To preserve the stability of designed proteins and to study minimal adjustments that would avoid misfittings, our collaborators wanted a tool that would support conservative

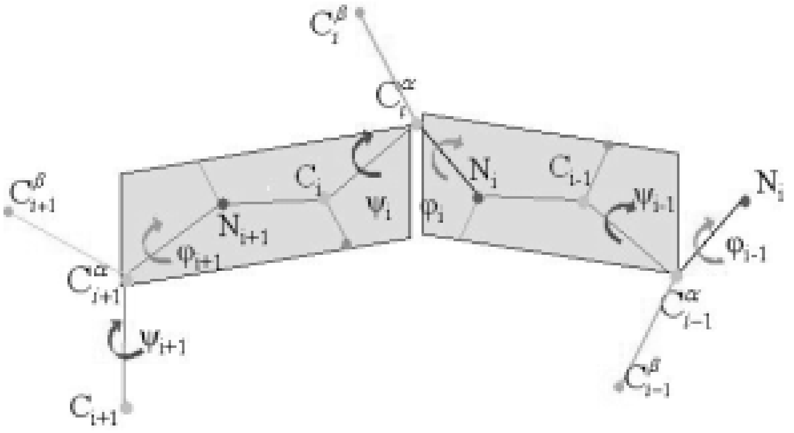

Fig. 1. Polypeptide chain.

Table 1. Bond Angles and Lengths: Ideal Values, Observed Values and Variations, and Allowed Variations in Probik

	Bond angle (°)			Bond length (Å)			Dihedral (°)
Atoms	N C_α C	C_α CN	CNC_α	N C	C_α C	C N	C_α N C C_α
Ideal value	111.2	116.2	121.7	1.47	1.53	1.33	180.0
Avg value	111.700	117.326	121.931	1.447	1.511	1.317	179.533
Avg stdev	3.066	1.581	1.815	0.011	0.012	0.010	8.172
Allowed dev	4.0	2.0	2.0	0.02	0.02	0.02	8.0

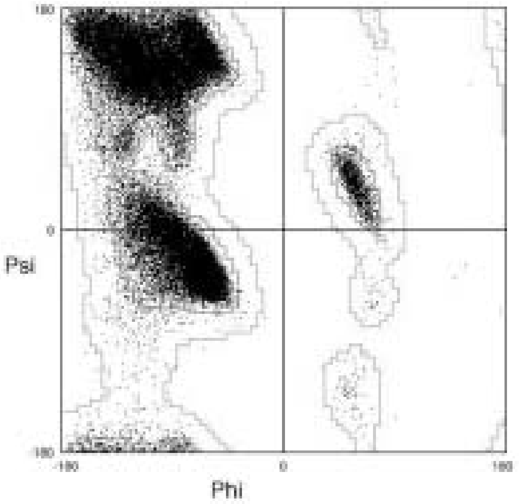

Fig. 2. Ramachandran plot of backbone ϕ, ψ angles in the top500 data set (Lovell et al. 2003; see http://kinemage.biochem.duke.edu/).

manipulation of the wild-type backbone ($<$ 1 Å), and satisfy geometric constraints. We need an accurate geometric model of local backbone motions that can be realized by rotation of ϕ, ψ angles and possibly small perturbations of other bond parameters. Probik, our computational tool, provides the biochemist with the range of motions possible for a fragment of the protein backbone as a result of variations in bond parameters.

2.3. Loop Closure Problem

The n-atom loop closure problem is, given the (fixed) position and orientation of the first and last atoms of an n-atom chain, to determine the possible positions of the $n-2$ intervening atoms that satisfy desired constraints on the bond lengths and bond angles and peptide dihedrals. Typically the dihedral angles ϕ, ψ are the free variables, and other parameters are fixed to ideal or observed values. If the first atom is brought to a canonical position, then six DOF are required to specify the position and orientation of the last atom. Thus, the loop closure problem is overconstrained for $<$ six DOF and underconstrained for $>$ six DOF.

Algorithms solving loop closure problems have been important in computational protein modeling for decades. They are used in what biochemists call "homology modeling", in which a new protein structure is predicted from a known structure of a protein with significant sequence overlap. Putative helices and sheets from the new protein are matched to corresponding structures in the old, and loop closure is used to sample possible loop conformations to connect them.

Loop closure algorithms are used similarly in crystallographic threading, where the goal is to fit a fragment of protein structure to electron density inferred from X-ray diffraction experiments. The recent work of Lotan et al. (2005) is an especially nice example: it helps fill in loops missed in crystallographic threading due to poor electron density by using kinematics to sample conformations that can then be tested for fits to the density and portions of the model already constructed.

In these applications, loop closure algorithms are applied to underconstrained loops with more than six DOF. Many algorithms sample the DOF and use minimization to achieve closure (Go and Scheraga 1970; Bruccoleri and Karplus 1985). Cyclic coordinate descent (Canutescu and Dunbrack 2003), which adjusts one dihedral angle at a time to match coordinates of three target atoms, is a popular algorithm because of its simplicity and speed. Perturbations of parameters have been studied to obtain derivatives for minimization and to validate the closures (Braun 1987; Bruccoleri and Karplus 1985; Palmer and Scheraga 1991). Some researchers build loops from fragments of known backbones structures to create more "protein-like" loops (Kolodny et al. 2005 is a recent example that contains many other references).

Ideally, we would like loop closure algorithms that automatically find a native-like loop by determining the minimum of an energy function. Modern algorithms typically find loop conformations that have lower energies than the native state (Jacobson et al. 2004); thus, energy functions and not sampling are the limitation for automatic loop closure. This is especially a problem for the small backbone adjustments for the crystallographic and protein design operations, because the important configurations are those in which the atoms are under strain to present a certain unique chemistry or even catalyze a reaction.

The above loop closure algorithms are often used to sample a large number of configurations, which can optionally be clustered, and presented to the user. Using exact inverse kinematics methods, as we shall see, allows us to present a more continuous range of options in a way that can be tied to perturbations of the original parameters. We can then observe how these parameters are varied to capture preferences for parameter values and ranges that would otherwise be difficult to elucidate.

3. Review: Exact Inverse Kinematics for Backbone Loop Closure

For a small segment of the backbone, we want to model the free rotation of the dihedrals while maintaining fixed bond lengths, bond angles, and peptide dihedrals. Mathematically, if we fix coordinate frames for an anchor and a target atom, assume fixed bond angles, bond lengths, and peptide dihedral angles, and allow the ϕ and ψ dihedral angles to rotate freely, what are all the possible positions of the backbone? What happens if we additionally allow one bond angle or bond length parameter to vary within some tolerance range? We can answer these questions using exact analytic solutions of the loop closure problem.

In this paper, we consider loops of a protein with six major DOF, corresponding to six consecutive ϕ or ψ dihedral angles. We "wiggle" a seventh DOF to generate continuous families of solutions. Figure 3 shows three-residue loops.

3.1. Denavit–Hartenberg Local Frames

The Denavit–Hartenberg (DH) local frames are widely used for modeling molecular configurations and serial manipulators. They provide a systematic way of describing the rotational and translational relationship along a protein backbone by defining a local, right-handed coordinate system for each rotatable bond.

Suppose that we choose n rotatable bonds, $b_1, b_2, \ldots, b_n$, along the protein backbone. Let u_i be the point on the line through bond b_i that is closest to the line through bond b_{i-1}; if these lines are parallel, any point may be chosen as u_i. We define the DH local frame associated with bond b_i to be $F_i = \{x_i, y_i, z_i, u_i\}$, where z_i is the vector in the direction of the previous bond, b_{i-1}, x_i is perpendicular to bonds b_{i-1} and b_i, and y_i is perpendicular to x_i and z_i (Figure 4).

Consecutive DH local frames, F_i and F_{i+1}, are related by a series of rotations and translations with the following geometric DH parameters:

- d_i = distance along bond b_i from u_i to the closest point to u_{i+1};
- θ_i = dihedral angle = measured from x_i to x_{i+1} about z_i;
- a_i = offset = distance from the line through bond b_i to u_{i+1};
- α_i = angle from z_i to z_{i+1}.

When all atoms of the backbone are used, then u_i are the atom positions, d_i are bond lengths, α_i are bond angles, and $a_i = 0$. However, the DH parameters can also specify frames for any chosen rotatable bonds.

Let $R_x(\omega)$ denote the 4×4 homogeneous matrix for rotation about the x-axis by angle ω, and let $T_z(d)$ denote the 4×4 homogeneous matrix for translation along the z-axis by amount d. The DH frames F_i and F_{i+1} are related by $F_{i+1} = A_i F_i$ where $A_i = T_z(d) \cdot R_z(\theta) \cdot R_x(\alpha) \cdot T_x(a)$.

Let A_1 and A_{hand} denote the DH frames for the anchor atom and target atom in the chain. Then we can write the known coordinate system, A_{hand}, in terms of the unknown coordinate

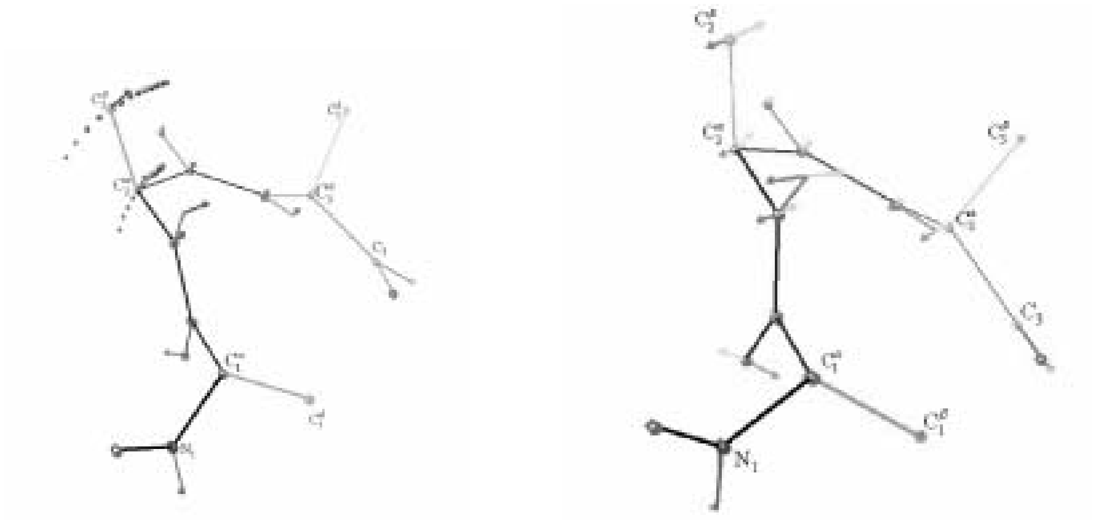

Fig. 3. A three-residue loop and corresponding derivative vectors for wiggled C_2^α bond angle (left) and wiggled C_2^α and C_2 bond angles (right).

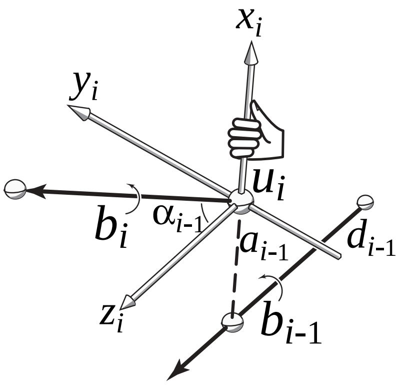

Fig. 4. Denavit–Hartenberg local frame.

systems of the intermediate atoms with a series of matrix multiplications. $A_1 A_2 \cdots A_n = A_{hand}$.

3.2. Implementation and Exact Analytic Solutions

Raghavan and Roth (1989) consider the six revolute manipulator problem, analogous to the six DOF atom loop problem. The chain consists of six revolute joints connected by six links, where each link contributes one free torsion angle. The previous equation becomes $A_1A_2A_3A_4A_5A_6 = A_{hand}$. The parameters α_i, d_i, and a_i $(1 \le i \le 6)$ are fixed and the torsion angles $\theta_1, \ldots, \theta_6$ are free. Thus, the matrix equation gives 12 multivariable equations, only six of which are independent because the coordinate frame vectors are orthogonal.

To reduce the degree and complexity of the system of six equations, they rewrite $A_3A_4A_5 = A_2^{-1}A_1^{-1}A_{hand}A_3^{-1}$. The variables appearing in the matrix entries are

$$\begin{pmatrix} (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) \\ (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) \\ (\theta_4,\theta_5) & (\theta_4,\theta_5) & (\theta_4,\theta_5) & (\theta_4,\theta_5) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) \\ (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) \\ (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

Raghavan and Roth proceed with the six independent equations from the third and fourth columns, sequentially eliminating four of the five variables by dialytic elimination. After symbolically expanding the determinant of the resulting 12×12 matrix, they obtain a 16 degree polynomial in $\tan(\theta_3/2)$. Mathematically, this method provides exact analytic solutions for the general six revolute joint manipulator problem; however, computationally it is subject to significant numerical errors as discussed in Manocha and Canny (1994).

Manocha and Canny (1994) replace the root-finding of Raghavan and Roth with more robust eigenvalue computations in a 24×24 matrix. Eight of the 24 eigenvalues are constants and the remaining 16 eigenvalues correspond to the undetermined dihedrals angles. There can be at most 16 real solutions and empirically $<$ eight real solutions are observed. An implementation is reported by Manocha and Zhu (1994)

As input, their IK solver requires (1) the DH frame for the target, A_{hand}, and (2) the known DH parameters, $\{\alpha_i$, d_i,

$a_i : 1 \le i \le 6\}$, for each of the six links in the chain. As output, the IK solver returns a finite number of solution sets for the free DH parameters, $\theta_1, \ldots, \theta_6$. We call each feasible set of dihedrals, $\theta_1, \ldots, \theta_6$, a “solution pose”.

Their solver uses symbolic preprocessing, rank computations, and dialytic elimination to obtain a 12×12 matrix whose entries are quadratic polynomials in terms of $\tan(\theta)$. This provides us an efficient and numerically stable solution to the six-revolute manipulator problem.

Wedemeyer and Scheraga (1999) present exact, elegant solutions to several loop closure problems including the tripeptide loop. They used spherical geometry and dialytic elimination to derive a 16 degree polynomial in the tangent of the half angle of one of the dihedral angles. Their method assumes that peptides are planar, so does not apply to all backbone chains. We have compared results from Manocha and Canny (1994) with our independent implementation of Wedemeyer and Scheraga (1999) to verify that they do agree on backbones with planar peptides.

Recently, Coutsais et al. (2004) generalized the solution of Wedemeyer and Scheraga (1999), obtaining a method that is similar to ours. They have use steepest descent minimization to vary bond angles or bond lengths to close loops when the initial conditions do not admit a solution in ϕ and ψ. An appendix shows some graphs of how perturbations change the solutions of the polynomial, but they do not exploit this as a tool for elucidating perturbation preferences as we have done. They have applied their method in Monte Carlo minimization for longer loops, and demonstrated improved efficiency on an eight-residue loop.

Finally, our method can be seen as a simple example of a continuation method, for which the mathematics has been developed by Wampler and co-workers (Wampler and Morgan 1993; Sommese and Wampler 2005). A continuation method takes a general solution and studies the behavior of numerical solutions under a homotopy of the parameters. We thank one of the reviewers for pointing out this connection, which we plan to explore further, starting with the work of Sommese and Wampler (2005).

4. Probik: Backbone Manipulation Algorithm

Probik is the computational tool we have developed for modeling local protein backbone motion. The PDB file for a given protein contains a list of atoms with residue numbers, atom numbers, types, and $\mathbf{R}^3$ coordinates. For a region of the backbone with six specified DOF (usually three- or four-residue loop) Probik computes derivatives of each atom position with respect to variation in the bond parameters in the region. In this way, Probik provides a numeric and geometric indication of the range of local backbone motions available by varying geometric parameters. In this section, we describe the four steps of the algorithm: calculate wiggled loops, obtain exact solutions, classify poses, and estimate derivatives. We begin by defining notation.

4.1. Protein Backbone Representations

Given the PDB file for a protein and a specified region of the backbone, Probik can iterate through all six DOF loops in the region. In this paper we consider one such loop, which we call the 6DOF loop, and we fix one bond parameter to vary, which we call the fuzz parameter.

Probik extracts, from the PDB file, the coordinates and types of all backbone atoms (N–H, C^α–C^β, C–O) in the 6DOF chain. We call the list of main chain coordinates (N, C^α, C, ...) the “AA chain” for the 6DOF loop. In order to use the IK solver discussed in Section 3.2, we compute another representation of the 6DOF loop with only six links, one for each free dihedral angle ϕ and ψ. For each full peptide, C^α_1–C–N–C^α_2, in the AA chain, let l_1 and l_2 be the lines through the atoms C^α_1–C and N–C^α_2, respectively. Let P_1 be the point on l_1 closest to line l_2. Define P_2 to be the corresponding point on l_2. We can represent the peptide by the three joints, C^α_1, P_2, and C^α_2, where the center joint P_2 is offset from the joint C^α_1 by the distance between P_1 and P_2. We call the resulting list of coordinates, C^α, P_2, C^α, P_2, ..., the “IK chain”.

For both AA and IK chain representations of the 6DOF loop, we calculate corresponding sets of DH parameters $\{\theta_i, \alpha_i, d_i, a_i\}$, and denote them AA parameters and IK parameters, respectively. (Note that in the AA parameters offset $a_i = 0$ for all i, whereas in the IK parameters a_i is the distance between P_1 and P_2 for the C^α_1–P link and $a_i = 0$ for the P–C^α_2 link.) Probik uses both representations: modifying AA parameters to wiggle or fuzz an angle parameter, sending IK chain parameters to the IK solver, and applying returned θ_i parameters to the AA chain to compute new amino acid coordinates.

4.2. Calculate Wiggled Loops

The first step in the algorithm is to compute the geometric representations and parameters of the 6DOF loop and to generate an array of values for the fuzz parameter. From the PDB file, we obtain the AA chain coordinates for the 6DOF loop and calculate the corresponding AA parameters, or actual DH parameters.

Using the actual value, the ideal value, and the standard deviation of the fuzz parameter, we generate an array of values for the parameter. We choose a range of values that includes at least one standard deviation around the actual value and (2/3) of a standard deviation about the ideal value. This ensures that we sample fuzz parameter values all around the actual and ideal values. The actual AA parameters, combined with this array of values, yield our array of wiggled AA parameters.

The corresponding array of wiggled IK parameter set is calculated as follows. For each set of wiggled AA parameters, we rebuild the AA chain, convert to its IK chain representation, and compute the corresponding IK parameters. Note that a change in a bond angle in the AA chain corresponds to a change in all the DH parameters (θ, α, d, a) of the corresponding adjacent links in the IK chain representation. The result is an array of wiggled IK parameters.

4.3. Get Exact Solutions—Using Manocha and Canny

The second step in the algorithm is to generate all possible configurations that close the 6DOF loop for small variations in the fuzz parameter. We use the IK Solver (Section 3.2; Manocha and Canny 1994) to obtain the solution poses for each set of wiggled IK parameters. Recall that there are at most 16 feasible configurations to close the 6DOF loop for a set of known DH parameters, $\{\alpha_i, d_i, a_i\}_{1\leq i\leq 6}$. We call the loop closure configurations corresponding to the original IK parameters the original solution poses. Of these, the actual pose is the solution pose that corresponds to the initial AA coordinates.

Typically, the IK solver returns between two and six sets of DH dihedral angles. We denote these solutions by Θ_1, $\Theta_2, \ldots, \Theta_s$. For each solution, $\Theta_i = \{\theta_1, \ldots, \theta_6\}$, we apply the corresponding DH transformation matrices to rebuild the IK chain. We discard any solutions whose final DH frame, A_{end}, does not hit the target frame within a specified tolerance ($\|A_{target} - A_{end}\|_2 > 10^{-4}$) due to numerical errors in the solver. For each solution that closes the loop, we convert the IK chain to the AA chain and order the resulting poses in increasing distance from the actual pose.

4.4. Classify Families of Poses

The next step is to organize the solution poses obtained from the IK solver to determine the range of motion allowed by varying the fuzz parameter. Recall that each solution pose corresponds to a zero of the underlying 16 degree polynomial for the actual IK parameter set. A small variation in the fuzz parameter corresponds to a small perturbation of the polynomial, which generally yields a small change in the roots by properties of transversality. As we vary the fuzz parameter and obtain the associated solution poses, we are sampling along the zero level set of a manifold in R^3. The solution poses over the range of fuzz values correspond to branches of the solution space. These branches move at different speeds and may merge into or appear from a complex solution over the range of fuzz values, which can result in a decrease or increase in the number of solution poses from one fuzz value to the next.

To classify solution branches, we consider solution poses ordered by increasing fuzz value. For each pose, we compute the distance using the vector norm to each of the poses for the next fuzz value and store the minimum distance in the corresponding entry of an adjacency matrix. We align the solution poses for the next fuzz value with the original poses according to the adjacency matrix. We continue in this way, aligning the solution poses for the increasing fuzz values, and then repeat the process for decreasing fuzz values.

Ambiguity arises when the number of solutions increases or decreases from one fuzz value to the next. We use the adjacency matrix and the second closest solution pose to classify the branches as continuing, appearing, or disappearing. After processing all the solutions for increasing and then decreasing fuzz values, we obtain the feasible range of motion for each of the original branches of solution.

4.5. Estimate the Derivative

The final step in the algorithm is to estimate the derivative of motion for each atom in the 6DOF chain motion with respect to the fuzz parameter. The geometric constraints within the loop force the atoms to move along smooth curves. For each branch, we model the motion of each atom in the loop with a second-order polynomial. The coefficients of the quadratic polynomial are determined by computing the least-squares fit on the coordinate points for each atom in the loop. The approximate magnitude and direction of motion is obtained for each atom (N–H, C^α–C^β, C–O) in the 6DOF loop, for each of the original solution poses.

4.6. Preliminary Results

Figure 3 shows example results of Probik in Mage, the protein viewer from the Richardson Laboratory (http://kinemage.biochem.duke.edu/). Probik can be used to study a specific motion in detail, with the derivatives and atom position dots corresponding to one wiggle parameter (left subfigure), or to offer choices of motion, by displaying derivatives corresponding to two (or more) bond parameter variations (right subfigure). Probik allows us to compare the motion resulting from varying different parameters, whether this be to choose one automatically, or to present the options to a biochemist and to see and understand why she makes the choice that she does.

In collaboration with biochemists, we have used Probik to reduce clashes in crystallographic refinement and to increase the space of rotamers allowed for residue changes in Dezymer. These have suggested new sequence designs to increase the stability for the TNT-binding protein, which are being created in the wet lab.

5. Covering the Space of Protein Backbone Motions

The algorithm described above allows us to move a protein backbone loop by varying one or more of the individual bond angle, bond length, or peptide dihedral parameters. In this section, we use simple, linear principal component analysis to find combinations of parameters that can act as control handles to move the backbone. A small number of control handles can move the backbone throughout the same range of motion as that created by perturbing all individual parameters. This reduction in the number of controls reduces the complexity of protein backbone manipulation and design. Our results show that four control handles are sufficient to move a loop to within 0.1 Å cRMSD of any wiggled conformation.

5.1. Principal Component Analysis on Backbone Parameters

Principal component analysis (PCA) reduces the dimensions of a set of inter-related variables, producing a set of derived variables known as the principal components (Jolliffe 2002). We use the most common PCA technique, which is to use singular value decomposition (SVD) to successively find and remove the best linear approximation to the residuals. Although our protein model is not linear, our data suggest that it is a sufficient method to test our hypothesis that the flexibility of protein backbones can be represented to a high degree of accuracy with just four DOF. We can reconstruct the data from the projections on all principal components, but we can also use the projections on the first few principal components as a lower-dimensional representation of the data, since these components are chosen to greedily preserve as much of the original data variation as possible.

For this work, we apply PCA to a randomly generated cloud set consisting of a large number of wiggled loops centered around a three-peptide loop extracted from a PDB file. We collect eight bond angle parameters, two peptide dihedral parameters, and six ϕ, ψ parameters, from the cloud set and store their deviations from the mean values in a matrix $\vec{M}$. The SVD procedure calculates matrices $\vec{U}$, $\vec{S}$, and $\vec{V}$ such that $\vec{M} = \vec{U}\vec{S}\vec{V}^{\mathrm{T}}$. The columns of $\vec{V}$ are the principal components of $\vec{M}$ and represent a new set of orthogonal basis vectors for the loop parameters. The diagonal entries of $\vec{S}$ represent the relative weights of the principal components. Matrix $\vec{U}$ contains the original loop parameter deviations transformed into the new basis.

Principal component analysis gives us a lower-dimensional approximation of the 16 backbone parameters for each loop in the cloud set. To measure accuracy, we could easily compare the approximation to the original loop parameters. In design applications, however, accuracy is better measured by how well atom positions can be reconstructed by the IK solver (respecting the closure constraints). For this second measure, we calculate the coordinate root-mean-squared deviation error (cRMSD, a common measure in biochemistry) of the reconstructed loop from the original. We consider two loops with a cRMSD of less than 0.1 Å to be similar.

When reducing the dimension of the space of loops, it is more important that the space remain accessible with fewer handles than that the parametrization remain the same. Therefore, we calculate a third measure of how well the first few principal components approximate the full range of motion. We build a probe set of loops by varying the test loop parameters along the first few principal components and then measure how well the probe set corresponds to the cloud set. We find that four principal components are sufficient to construct a probe loop similar to each loop in the cloud set.

5.2. Experimental Setup

To perform PCA on a given loop, we first generate a large number of perturbed loops. We wiggle each of the starting loop's bond angles and peptide dihedral angles in a normal distribution around their starting values. Normal sampling ensures that few of the chains have the majority of their parameters far from ideal values; thus, we obtain a cloud that covers the space of motions close to the starting loop. Because our starting loops come from real PDB files, many contain parameters that are beyond the allowable ranges. To compensate, we extend the ranges by 1.5 degrees in each direction. After the inverse kinematics solver generates a set of solutions from the new parameters, we keep all branches that are within the extended ranges and whose ϕ, ψ angles meet the additional requirements described below. This is repeated until we have 5,000 perturbed loops that form a cloud around the starting loop.

We must calculate the mean value of each of the parameters in the cloud set to perform PCA. Calculating the mean of the peptide unit dihedrals and the bond angles is straightforward, because they are confined to small ranges around their ideal values. The ϕ, ψ dihedral angles do not have a specified range and can lie anywhere on the unit circle. Because it is not clear how to average angles that are distributed across the whole unit circle, we filter out all loops that have any ϕ, ψ angles that differ by more 90 degrees from the starting values. This allows us to calculate a well-defined mean value. This restriction should not be significant, because in tightly packed molecules, interatom collisions should restrict the movement of ϕ, ψ dihedrals to within a similar range.

We perform the above calculations on 150 test loops from the following PDB files: 1tre.pdb, 1r69.pdb, 1byi.pdb, 2ci2.pdb, and 2mhu.pdb. We use the tripeptide loop and wiggle the eight bond angles and the two peptide units. The inverse kinematics solver calculates the remaining six ϕ, ψ angles, for a total of 16 parameters. These are used to create the $5{,}000 \times 16$ matrix for the PCA calculation described in the previous section.

5.3. Direct Reconstruction of the Cloud Set

A first test of how well a PCA-derived lower-dimensional approximation can reconstruct the cloud set is to measure how well each individual loop can be reconstructed from the approximation of its parameters. For each loop in the cloud set, we approximate the parameters using one to 16 principal

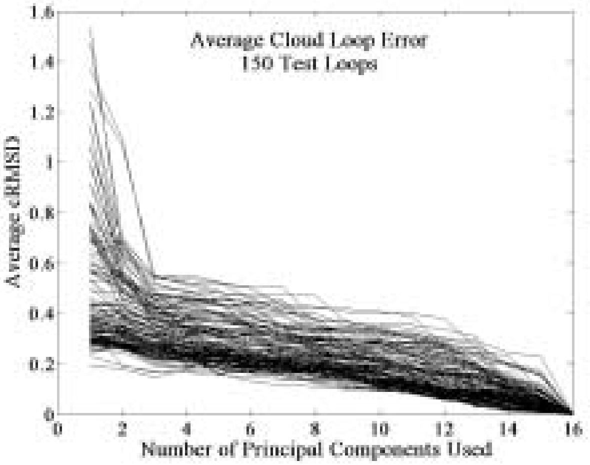

Fig. 5. Average error for 150 test loops.

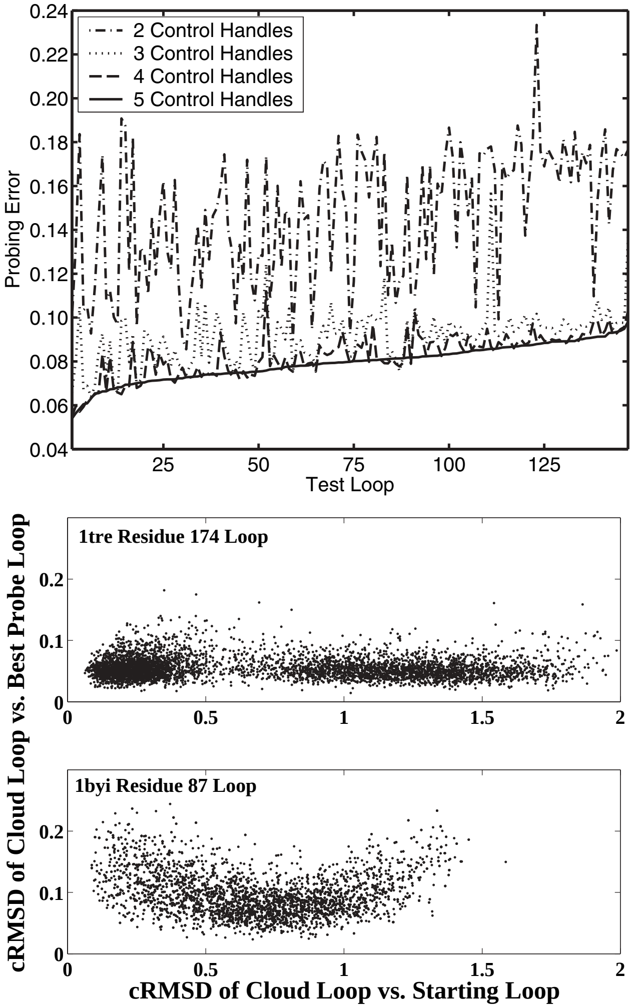

Fig. 6. Left: average probing error for 150 test loops. Right: relationship between a cloud loop's deviations from the original and its closest probe loop.

components and then use them to rebuild atom positions. Because the loop end positions and orientations will have changed, we use the inverse kinematics solver to close the reconstructions, then calculate the cRMSD of each reconstruction against the original atom positions. Figure 5 plots for 150 test loops the average cRMSD error of the 5,000 cloud set loops, as a function of the number of principal components used to approximate the individual loop parameters.

The plot shows that many more than five principal components are necessary for accurate reconstruction. The average cRMSD for most of the loops does not reach 0.2 Å until eight principal components are used, which is too many for interactive control of backbone motion. The error is high because closing the loop with the IK solver changes ϕ and ψ angles and coordinate positions, making many of the reconstructed loops miss their original targets. Because it is not necessary to keep the same parametrization of the cloud, in the next section we describe a better measure.

5.4. Exploring the Range of Motion of Control Handles

A control handle adjusts a loop's angle parameters in the direction of its associated principal component. To test how well we can cover the space of the original loop cloud, we randomly adjust the control handles to create a probe set of 10,000 loops. Starting from the average angle parameters, we successively add the product of a principal component with a random scalar chosen so that no individual angle parameter moves beyond the extended ranges. For this experiment, we add two, three, four, and five control handle movements, and run the inverse kinematics solver to enforce closure constraints.

We then calculate, for each loop in the cloud set, the lowest cRMSD to our set of 10,000 probe loops. The average value across the 5,000 cloud loops is our estimate of probing error. Probing error decreases to a limit as the number of probes increases; In our experiments, we found that the probing error decreased only a few hundredths of an Angstrom when the number of probes increased from 10,000 to 20,000. Figure 6 (left) plots the probing error for each of the 150 test loops. Each line on the plot shows the results for a different number of control handles. The y-axis shows the probing error for each of 150 test loops along the x-axis. The loops on the x-axis are sorted by their probing error when using five control handles. For most loops, three control handles produce average cRMSD error below 0.1 Å. Overall, the low errors demonstrate that PCA-based control handles can reproduce the space of allowed motions.

A final check shows that probe loops land close to cloud loops that are both near to and far from the starting position. Figure 6 (right) shows scatter plots for two examples: the upper, 1tre.pdb residue 174, has one of the smallest probing errors and the lower, 1byi residue 87, has one of the greatest. For each cloud loop we plot a point whose x-coordinate is the cRMSD to the original and y-coordinate is the cRMSD to the closest probe loop constructed using four control handles. The lower, 1byi, shows a slight curve, probably because its initial pose is strained so that perturbations take it toward a more relaxed position. Since the range of x is 10 times the range of y, both are essentially flat—we cover all areas of the space.

5.5. Control Handles for Loops in α-Helices

In the previous section we have demonstrated that we can construct control handles for most loops that provide a good range of motion. Ideally we should be able to pre-calculate sets of control handles that will work well for common loop conformations and store these in a library. We examined the principal components of the 71 loops from the previous section that were from α-helix structures. These loops have similar bond and peptide dihedral angle parameters. We have found that the principal components of these parameters are also similar. Figure 7 (left) plots the normalized values of the first three principal components for the α-helix loops. Each principal component vector has two peptide dihedral angles and eight bond angles.

We ran the probing experiment from the previous section on these 71 loops using the average of their principal components as control handles. Figure 7 (right) shows the results, which were comparable to the original probing experiment in which each loop used its own individual control handles. The first four average control handles produce mean cRMSD errors of less than 0.1 Å for almost all loops. The error is slightly higher using average control handles, but still within our goal of reaching all cloud set loops to within 0.1 Å using four or fewer control handles. Furthermore, we no longer need to construct a large cloud set or perform PCA to find good control handles for α-helix loops.

6. Conclusion

Geometric modeling of protein structure is fundamental to the task of computational protein design, crystalographic structure refinement, and computational protein folding. Solutions to inverse kinematic problems help to provide better geometric models for molecular chains. In general, the solution to an inverse kinematic problem is complex and computationally expensive. Probik, our computational tool, models local backbone motion by combining the exact kinematic solutions for a segment of a protein backbone with the observed variation of bond angles, lengths, and dihedrals.

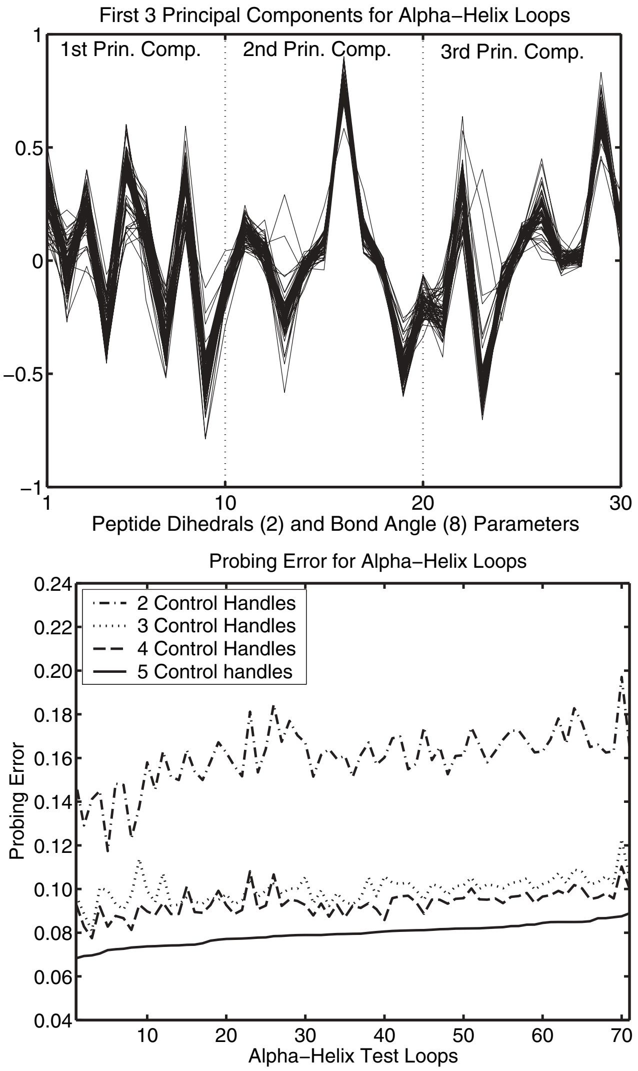

Fig. 7. First three principal components for helix loops.

To obtain the exact solutions to the 6DOF loop closure problem, we use the Manocha and Canny (1994) IK solver. This computational tool is an efficient and numerically robust 6DOF IK solver. Our technique avoids many geometric assumptions, such as planar peptides, which are often made to reduce the complexity of the inverse kinematics problem. We combine the actual geometric measures of the given backbone segment with controlled variation of a bond parameter, to model the natural motion of a protein backbone.

By estimating the derivative of motion with respect to variable bond parameters, we provide a concise numeric and geometric notion of the range of feasible motions. Our algorithm generates a Kinemage file, roughly the size of the original PDB file, containing the three-dimensional representation of the protein and the derivative vectors, which allows interactive computer display. The computation of and file generation for the motions resulting from one wiggle parameter in a given protein structure takes approximately 3–5 s. The geometric model produced by Probik allows biochemists to investigate, understand, and predict the motion of a protein backbone.

Acknowledgments

We are grateful to our collaborators: to Professors David and Jane Richardson, and student Ian Davis for sharing their expertise on what proteins do, and sharing their Kinemage suite of programs (Richardson and Richardson 2001; see http://kinemage.biochem.duke.edu/); to Professor Homme Hellinga and students Loren Looger and Yaji Tian for many ideas and discussions, and for access to Dezymer. We thank the three referees for their suggestions and references. This research has been partially supported by grants from the National Science Foundation ITR (0076984) and DARPA.

References

Amadei, A. et al. 1996. An efficient method for sampling the essential subspace of proteins. *Journal of Biomolecular Structure Dynamics* 13(4):615–625.

Branden C. and Tooze J. 1999. *Introduction to Protein Structure*, 2nd edition, Garland Publishing, New York.

Braun, W. 1987. Local deformation studies of chain molecules: Differential conditions for changes of dihedral angles. *Biopolymers* 26(10):1691–1704

Bruccoleri, R. E. and Karplus, M. 1985. Chain closure with bond angle variations. *Macromolecules* 18:2767–2773.

Canutescu, A. A. and Dunbrack, R. L. Jr. 2003. Cyclic coordinate descent: A robotics algorithm for protein loop closure. *Protein Science* 12:963–972.

Coutsais, E. A., Seok, C., Jacobson, M. P., and Dill, K. A. 2004. A kinematic view of loop closure. *Journal of Computational Chemistry* 25:510–528.

Davis, I. W., Murray, L. W., Richardson, J. S., and Richardson, D. C. 2004. MolProbity: structure validation and all-atom contact analysis for nucleic acids and their complexes. *Nucleic Acids Research* 32(Web Server Issue):W615–619.

Davis, I. W., Arendall, W. B. III, Richardson, J. S., and Richardson, D. C. 2005. The Backrub: A gentle but rotamer-coupled mode of local backbone movement in proteins. In preparation.

de Groot, B. L. et al. 1996. Toward an exhaustive sampling of the configurational spaces of the two forms of the peptide hormone guanylin. *Journal of Biomolecular Structure Dynamics* 13(5):741–751.

Dwyer, M., Looger, L., and Hellinga, H. 2004. Computational design of a biologically active enzyme. *Science* 304:1967–1971.

Edelsbrunner H. and Koehl P. 2003. The weighted volume derivative of a space-filling diagram. *Proceedings of the National Academy of Science* 100:2203–2208.

Engh R. A. and Huber, R. 1991. Accurate bond and angle parameters for X-ray protein structure refinement. *Acta Crystallographica* A47:392–400.

Go, N. and Scheraga, H. A. 1970. Ring closure and local conformational deformations of chain molecules. *Macromolecules* 3:178–186.

Jacobson, M. P., Pincus, D. L., Rapp, C. S., Day, T. J. F., Honig, B., Shaw, D. E., and Friesner, R. A. 2004. A hierarchical approach to all-atom protein loop prediction. *Proteins: Structure, Function, and Bioinformatics* 55(2):351–367

Jolliffe, I. T. 2002. *Principal Component Analysis*, Springer-Verlag, New York.

Kobayashi, N., Yamato, T., and Go, N. 1997. Mechanical property of a TIM-barrel protein. *Proteins* 28(1):109–116.

Kolodny, R., Guibas, L., Levitt, M., and Koehl, P. 2005. Inverse kinematics in biology: The protein loop closure problem. *International Journal of Robotics Research* 24(2–3):151–163.

Leach, A. R. 1996. *Molecular Modelling: Principles and Applications*, Addison-Wesley Longman, Reading, MA.

Looger, L., Dwyer, M., Smith, J., and Hellinga, H. 2003. Computational design of receptor and sensor proteins with novel functions. *Nature* 423(6936):185–190.

Lotan, I., van den Bedem, H., Deacon, A. M., and Latombe, J.-C. 2005. Computing protein structures from electron density maps: The missing fragment problem. *Algorithmic Foundations of Robotics VI, STAR 17,* M. Erdmann, D. Hsu, M. Overmars, F. van der Stappen, editors, Springer-Verlag, Berlin.

Lovell, S. et al. 2003. Structure validation by Cα Geometry: ϕ, ψ and Cβ deviation. *Proteins: Structure, Function, and Genetics* 50:437–450. See http://kinemage.biochem.duke.edu/validation/model.html.

Manocha, D. and Canny, J. 1992. Real-time inverse kinematics for general 6R manipulators. *Proceedings of the IEEE International Conference on Robotics and Automation*, Nice, France, pp. 383–389.

Manocha, D. and Canny, J. 1994. Efficient inverse kinematics of general 6R manipulators. *IEEE Transactions on Robotics and Automation* 10(5):648–657.

Manocha, D. and Zhu, Y. 1994. A fast algorithm and system for inverse kinematics of general serial manipulators. *Proceedings of the IEEE Conference on Robotics and Automation*, San Diego, CA, pp. 3348–3354.

Palmer, K. A. and Scheraga, H. A. 1991. Standard-geometry chains fitted to X-ray derived structures: Validation of the rigid-geometry approximation. I. Chain closure through a limited search of "loop" conformations. *Journal of Computational Chemistry* 12(4):505–526.

Petsko, G. A. and Ringe, D. 2004. *Protein Structure and Function*, New Science Press, Oxford.

Raghavan, M. and Roth, B. 1989. Kinematic analysis of the 6R manipulator of general geometry. *International Symposium on Robotics Research*, Tokyo, Japan, pp. 314–320.

Richardson, D. C. and Richardson, J. S. 2001. MAGE, PROBE, and Kinemages. *International Tables for Crystallography*, M. G. Rossmann and E. Arnold, editors, Kluwer, Dordrecht, Vol. F, Chapter 25.2.8., pp. 727–730.

Sommese, A. J. and Wampler, C. W. II. 2005. *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*, World Scientific, Singapore.

Song, G. and Amato, N. M. 2001. A motion planning approach to folding: From paper craft to protein folding. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, May 21–26, pp. 948–953.

Teodoro M. L., Phillips, G. N. Jr., and Kavraki, L. E. 2001. Molecular docking: A problem with thousands of degrees of freedom. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, May 21–26, pp. 960–965.

Teodoro M. L., Phillips, G. N. Jr., and Kavraki, L. E. 2003. Understanding protein flexibility through dimensionality reduction. *Journal of Computational Biology* 10(3–4):617–634.

Wampler, C. W. II and Morgan, A. P. 1993. Solving the kinematics of general 6R manipulators using polynomial continuation. *Robotics: Applied Mathematics and Computational Aspects*, K. Warwick, editor, Clarendon, Oxford, pp. 57–69.

Wedemeyer, W. and Scheraga, H. 1999. Exact analytical loop closure in proteins using polynomial equations. *Journal of Computational Chemistry* 20:819–844.

Zhang, M. and Kavraki, L. E. 2002. Solving molecular inverse kinematics problems for protein folding and drug design *Currents in Computational Molecular Biology, RECOMB ’02*, Washington, DC, USA, pp. 214–215.