Regression Analysis of Multiple Protein Structures

Thomas D. Wu[1] Trevor Hastie[2] Scott C. Schmidler[1,3]
Douglas L. Brutlag[1]
Departments of [1]Biochemistry and [2]Statistics
[3]Section on Medical Informatics
Stanford University, Stanford, CA 94305, USA

Abstract

A general framework is presented for analyzing multiple protein structures. A family of related protein structures may be analyzed using statistical regression methods. The analysis requires alternating steps of finding correspondences among the protein structures and superimposing the corresponding landmarks. The superposition step may be performed using either affine or orthogonal transformations, thereby allowing protein structures to undergo either pure rotations or rotation plus shear operations. Regression analysis permits a separate weight for each position, allowing one to emphasize particular segments of a protein structure or to compensate for variances that differ at various positions in a structure.

In addition, a novel method is introduced for finding an initial correspondence, based on matching discrete curvatures along the protein backbone. Another novel method is introduced for obtaining gap functions that adapt to the given data, thereby making dynamic programming methods more robust. An example is presented involving a set of seven globin structures. Regression analysis, using both affine and orthogonal transformations, reveals that globins are most strongly conserved structurally in the mid-regions of the E and G helices.

1 Introduction

A recurring issue in biology is the study of variability. Evolutionary and population biologists have long studied the variability of organisms and species. In the past few decades, with the advent of crystallography, molecular biologists have also been able to study variability, by comparing and contrasting protein structures. The amount of variability at each position suggests their relative importance and possible functions.

Variability has also been a central concern in statistics. It would seem natural, then, to apply statistical methods to study structural variability in protein structures. In this paper, we undertake such an approach. We use the most classic field of statistical analysis—regression—to analyze a family of multiple protein structures. We assume that variations in protein structure can be represented by a statistical model. A statistical model can be obtained using techniques from regression analysis to obtain a "template" structure that describes the entire family. Such templates are useful in understanding the structure and function of a given family [5].

Our approach is more general than existing methods, which generally rotate protein structures to superimpose corresponding atoms. If we permit each protein structure to undergo an affine transformation, which may include both rotation and shear components, we obtain an elegant and efficient method for superposition. Our method essentially finds the eigensolution to a least-squares problem. In the least-squares formulation, we find a regression model $\overline{\mathbf{M}}$ and affine transformation matrices $\mathbf{B}_j$ that minimize the objective function

$$\sum_{j=1}^{J} ||\mathbf{M}_j \mathbf{B}_j - \overline{\mathbf{M}}||^2 \tag{1}$$

where each $\mathbf{M}_j$ is a coordinate matrix of corresponding atoms in the J protein structures. It can be shown that, under certain assumptions, the model found is the maximum likelihood estimate of the average protein structure [7]. Therefore, we view the analysis of protein families not merely as a superposition problem, but as a problem in modeling.

The modeling approach is more general than superposition, and can be extended to handle different assumptions about how protein structures vary. For instance, in this paper, we extend our approach to handle heteroscedasticity, or unequal variances. In general, least-square methods are affected most by those positions that have the greatest variability. This is often undesirable, because intuitively those positions should contribute proportionately less to the final result. We therefore show in this paper how multiple protein structures may be analyzed using a weighted least-square approach, where the weights may be chosen to downweight highly variable positions or to emphasize particular segments of the protein structure.

Our statistical model allows each protein structure to undergo a general affine transformation, which permits not only rotations, as in existing approaches, but also shear in three dimensions. This generalization allows us to obtain the regression solution quickly, using a single eigendecomposition [9]. In contrast, when only rotations are allowed, the optimal superposition of multiple structures requires iterative or stochastic techniques [4, 11, 13, 3]. The rotation constraint essentially requires that each transformation matrix be orthogonal:

$$\mathbf{B}_j^T \mathbf{B}_j = \mathbf{I} \tag{2}$$

Solving equation 1 subject to this constraint requires an iterative algorithm, whereas the optimal regression solution can be obtained in a single step.

Nevertheless, we have developed a general framework that allows us to superimpose multiple protein structures, using either affine or orthogonal transformation matrices. The prerequisite for either method is a correspondence among atoms in the different protein structures. Unfortunately, this correspondence is typically unknown *a priori*, but may be inferred once we have superimposed the protein structures. Hence, analysis of multiple protein structures is inherently circular, alternating between finding a correspondence and superimposing the corresponding atoms. The latter step may be performed using either pure rotations, which require an iterative algorithm, or affine transformations, which require only an eigendecomposition.

To begin the cycle, we need an initial correspondence. In this paper, we also present a novel method for finding an initial correspondence quickly and relatively accurately by measuring and matching discrete curvatures. We have found that the discrete curvature along the backbone of a protein structure serves as a useful "signature" that highlights the main features of the structure, including helices, strands, and loops. Moreover, discrete curvature is invariant to rotation and location. Therefore, discrete curvature allows us to make a preliminary correspondence among protein structures, without having to superimpose the structures beforehand. The method of matching discrete curvatures provides an alternative to existing methods for finding correspondences. Because discrete curvature is a scalar function, it simplifies the dynamic programming procedure compared with matching vector sets [16, 15], distance matrices [10], or properties [17].

Finally, in this paper, we make a contribution to dynamic programming methods by introducing a concept called adaptive dynamic programming. Dynamic programming methods depend heavily upon particular parameters called gap penalties, which must be chosen carefully for each problem. In an earlier version of this work [19], we chose the gap penalties by trial and error. Since then, we have developed a more flexible approach to selecting the gap penalties, by allowing the gap penalties to adapt to the difficulty of matching two functions.

2 Methods

2.1 Formulation

The regression algorithm takes as input a set of J protein structures, with arbitrary orientations and shifts relative to the origin. Each protein structure is represented by a *structure matrix* $\mathbf{S}_j$, which is an $N_j \times 3$ matrix of coordinates. Each row in the matrix contains the x, y, and z coordinates for an atom in the protein structure. In this paper, we consider only the backbone, represented by the Cα atoms, ordered from the N- to the C-terminus. However, it is possible to extend our approach to consider other atoms as well.

We adopt a statistical model for the observed protein structures. We assume that the given family may be described by a template or *regression model* $\overline{\mathbf{M}}$, which is an $N \times 3$ matrix of landmarks. A *landmark* is a point that is assumed to be common to all structures in the given family. A set of landmarks describes a *correspondence* among the protein structures. The corresponding landmarks for each protein structure $\mathbf{S}_j$ are represented by a *landmark matrix* $\mathbf{M}_j$, which contains a subset of the atoms in $\mathbf{S}_j$. Our model assumes that each landmark matrix $\mathbf{M}_j$ differs from the regression model $\overline{\mathbf{M}}$, both globally and locally, by the following relationship:

$$\mathbf{M}_j = \overline{\mathbf{M}}\mathbf{B}_j^{-1} + \mathbf{1}\boldsymbol{\mu}_j^T + \boldsymbol{\varepsilon}_j \tag{3}$$

where $\mathbf{B}_j$ is a 3×3 affine *transformation matrix*, $\boldsymbol{\mu}_j$ is a 3×1 *offset vector*, and $\boldsymbol{\varepsilon}_j$ is an $N \times 3$ *error matrix*. The offset vector describes the global translation difference between each structure and the regression model; the transformation matrix describes the global rotational and shear difference; and the error matrix represents local differences for each landmark.

Our algorithm consists of three steps:

1. **Reference-based curvature matching:** Compute the discrete curvature κ_j for each protein structure $\mathbf{S}_j$. Find corresponding landmarks $\mathbf{M}_j^{(1)}$ by matching discrete curvatures to a reference structure, and obtain the regression model $\overline{\mathbf{M}}^{(1)}$ and transformation matrices $\mathbf{B}_j^{(1)}$.

2. **Reference-based coordinate matching:** Find corresponding landmarks $\mathbf{M}_j^{(2)}$ by matching coordinates to a reference structure, and obtain the regression model $\overline{\mathbf{M}}^{(2)}$ and transformation matrices $\mathbf{B}_j^{(2)}$.

3. **Model-based coordinate matching:** Find corresponding landmarks $\mathbf{M}_j$, by matching coordinates iteratively to the evolving regression model, and obtain the regression model $\overline{\mathbf{M}}$ and transformation matrices $\mathbf{B}_j$.

The third step of the algorithm may require several iterations, because each new set of correspondences results in a slightly different superposition, which may in turn result in a different set of correspondences. The algorithm stops when the set of correspondences remains constant.

2.2 Discrete Curvature

Our algorithm begins by computing the discrete curvature at each Cα carbon along the backbone of each protein structure. Curvature is essentially the rate of change of a tangent vector along a curve. We measure discrete curvature, which uses the discrete positions at Cα atoms to measure the curvature. This measure is intended to distinguish α-helices from β-strands, because alternating Cα atoms in β-strands should yield a low discrete curvature.

Let us label each Cα atom in protein $\mathbf{S}_j$ by a *sequence index* $s = 1, \ldots, N_j$. Since the series of peptide bond lengths between adjacent Cα carbons has relatively constant length, the index s also serves as an arc length parameter along the backbone. Let the position of Cα at index s be $\mathbf{p}_j^T(s) = [x_j(s)\ y_j(s)\ z_j(s)]$, where $x_j(s)$, $y_j(s)$, and $z_j(s)$ are coordinates from structure matrix $\mathbf{S}_j$.

Then we perform two rounds of numerical differentiation:

$$\Delta\mathbf{p}_j(s) = [\mathbf{p}_j(s+1) - \mathbf{p}_j(s-1)]/2 \tag{4}$$

$$\mathbf{t}_j(s) = \frac{\Delta\mathbf{p}_j(s)}{\|\Delta\mathbf{p}_j(s)\|} \tag{5}$$

$$\frac{d\mathbf{t}_j(s)}{ds} = [\mathbf{t}_j(s+1) - \mathbf{t}_j(s-1)]/2 \tag{6}$$

$$\kappa_j(s) = \left\|\frac{d\mathbf{t}_j(s)}{ds}\right\| \tag{7}$$

where $\mathbf{t}_j(s)$ is the unit tangent vector and $\kappa_j(s)$ is the discrete curvature at $s = 3, \ldots, N_j - 2$.

2.3 Corresponding Landmarks

We compute corresponding landmarks using dynamic programming, which is also known as dynamic time-warping in other literature [12]. Dynamic programming finds a correspondence between two structures that minimizes the overall distance between the structures. Let r and s be the sequence indices of atoms in structure matrices $\mathbf{S}_i$ and $\mathbf{S}_j$, respectively. Let $d(r,s)$ be some distance metric between atoms r and s. Then we would like to find two collinear sequences of atoms $1 \le r_{(1)} < r_{(2)} < \ldots < r_{(m)} \le N_r$ and $1 \le s_{(1)} < s_{(2)} < \ldots < s_{(m)} \le N_s$ that minimize the function

$$\sum_{i=1}^{m} d(r_{(i)}, s_{(i)}) + g(0, r_{(1)}) + \sum_{i=1}^{m-1} h(r_{(i)}, r_{(i+1)}) + g(r_{(m)}, N_r + 1) + g(0, s_{(1)}) + \sum_{i=1}^{m-1} h(s_{(i)}, s_{(i+1)}) + g(s_{(m)}, N_s + 1)$$

where $g(x,y)$ is the *gap penalty* for skipping from x to y at the end of either sequence, and $h(x,y)$ is the gap penalty for skipping from x to y in the middle of either sequence. Gap penalty functions may be arbitrarily complex, but when they are linear functions, the time complexity is reduced from $O(n^3)$ to $O(n^2)$ [8]. Hence, we use $g(x,y) = \alpha_0 + \beta_0(y-x)$, and $h(x,y) = \alpha_1 + \beta_1(y-x)$, except we require the gap penalty be zero when $y - x = 1$. The parameters α are opening penalties, and the parameters β are extension penalties. We might typically choose smaller penalties for g, because protein sequences often have variable-length ends that do not correspond well to other sequences.

The three steps apply dynamic programming with different distance metrics:

$$d(r,s) = \begin{cases} (\kappa_i(r) - \kappa_j(s))^2 & \text{for step 1} \\ \|\mathbf{p}_i(r) - \mathbf{p}_j(s)\|^2 & \text{for step 2} \\ \|\mathbf{p}_{\overline{\mathbf{M}}'}(r) - \mathbf{p}_j(s)\|^2 & \text{for step 3} \end{cases} \tag{8}$$

In steps 1 and 2, we compute distances relative to a reference structure $\mathbf{S}_i$. The reference for step 1 is simply the longest protein. The reference for step 2 is the protein structure closest to the initial regression model obtained in step 1. In step 3, we transform the regression model $\overline{\mathbf{M}}$ into a structure $\overline{\mathbf{M}}'$ in the space of $\mathbf{S}_j$, and then measure distances relative to the transformed model.

2.4 Adaptive Dynamic Programming

A major problem with dynamic programming methods is that their behavior depends greatly on the particular gap penalties chosen. If the gap penalties are too small, the solution will contain too many gaps. Conversely, if the gap penalties are too large, the solution may contain no gaps, even though the biologically correct solution contains them. If we choose gap penalties before we have seen the data, the dynamic programming algorithm may introduce too many gaps for some data and too few for other data.

The problem of gap penalties is particularly troublesome for superimposing protein structures, because some sets of structures superimpose tightly, while other sets superimpose loosely. To handle this problem, we suggest that gap penalties for structural superposition be chosen adaptively. In other words, we propose gap penalties that are data-dependent.

Our method for performing adaptive dynamic programming requires two passes of dynamic programming, each with a different set of gap penalties. The first set of gap penalties is matrix-based; the second set is path-based. Suppose we have a distance metric $d(r,s)$ defined for $1 \le r \le N_r$ and $1 \le s \le N_s$, such as those in equation 8. This metric can be represented as a matrix of size $N_r \times N_s$. Then we choose the gap penalties for the first pass to be some function of the mean μ and standard deviation σ of $d(r,s)$ over the entire matrix. In preliminary tests, we have found good results by letting $\alpha_0 = \beta_0 = \alpha_1 = \beta_1 = \mu + \sigma$. The first dynamic programming pass will find an initial alignment path, typically without gaps, because the gap penalties in the first pass are relatively high.

For the second pass, we compute the mean μ' and standard deviation σ' of $d(r,s)$ over the alignment path found in the first pass. This allows us to find gap penalties that are smaller and dependent on the data. Again, we have found good results by letting all gap penalty parameters in the second pass equal $\mu' + \sigma'$. The second dynamic programming pass introduces gaps into the alignment.

2.5 Regression Model using Affine Transformations

For each round of corresponding landmarks, our algorithm computes a regression model and set of transformation matrices. In this section, we discuss the solution for affine transformations. In section 2.9, we provide a solution for orthogonal transformations. Let us assume that each landmark matrix $\mathbf{M}_j$ is centered by subtracting $\mathbf{1}\boldsymbol{\mu}_j$, where the offset vector $\boldsymbol{\mu}_j$ contains the mean x, y, and z coordinates of $\mathbf{M}_j$. Our algorithm stores $\boldsymbol{\mu}_j$ at each step, for use in later superpositions.

The objective function in equation 1 can be solved using least-squares regression, following a method described by Hastie and colleagues [9]. To avoid degeneracies, we require that $\overline{\mathbf{M}}$ is orthogonal, so $\overline{\mathbf{M}}^T\overline{\mathbf{M}} = \mathbf{I}$. Then the optimal transformation matrix for $\mathbf{M}_j$ is $\mathbf{B}_j = (\mathbf{M}_j^T\mathbf{M}_j)^{-1}\mathbf{M}_j^T\overline{\mathbf{M}}$ and hence $\mathbf{M}_j\mathbf{B}_j = \mathbf{H}_j\overline{\mathbf{M}}$, where $\mathbf{H}_j = \mathbf{M}_j(\mathbf{M}_j^T\mathbf{M}_j)^{-1}\mathbf{M}_j^T$ is a projection operator. So at the minimum, the quantity in equation 1 equals

$$\begin{aligned}\sum_{j=1}^{J} \|(\mathbf{H}_j - \mathbf{I})\overline{\mathbf{M}}\|^2 &= \sum_{j=1}^{J} \mathrm{tr}(\overline{\mathbf{M}}^T(\mathbf{I} - \mathbf{H}_j)\overline{\mathbf{M}}) \\ &= J\mathrm{tr}(\overline{\mathbf{M}}^T(\mathbf{I} - \overline{\mathbf{H}})\overline{\mathbf{M}}) \qquad (9)\end{aligned}$$

where $\overline{\mathbf{H}}$ is the average of the projection operators $\mathbf{H}_j$. The solution for the regression model that minimizes the above quantity can be obtained by letting $\overline{\mathbf{M}}$ be the eigenvectors corresponding to the three largest eigenvalues of $\overline{\mathbf{H}}$.

In practice, to achieve better numerical stability, we perform the above computations by using QR decompositions $\mathbf{M}_j = \mathbf{Q}_j\mathbf{R}_j$, where $\mathbf{Q}_j$ is orthogonal and $\mathbf{R}_j$ is upper triangular. This decomposition then allows us to compute $\mathbf{H}_j = \mathbf{Q}_j\mathbf{Q}_j^T$ and $\mathbf{B}_j = \mathbf{R}_j^{-1}\mathbf{Q}_j^T\overline{\mathbf{M}}$.

2.6 Weighted Regression

Let us examine the error matrix $\boldsymbol{\varepsilon}_j$ in our statistical model 3. Each row $\boldsymbol{\varepsilon}_j(s)$ in the matrix represents the error at landmark s. Let $\|\boldsymbol{\varepsilon}_j(s)\|$ be the magnitude of the error. In the ordinary least-squares model, we assume that the expected error is zero, that errors have constant variance, and that they are uncorrelated:

$$\begin{aligned}E(\|\boldsymbol{\varepsilon}_j(s)\|) &= 0 \\ \mathrm{Var}(\|\boldsymbol{\varepsilon}_j(s)\|) &= \sigma^2 \\ \mathrm{Cov}(\|\boldsymbol{\varepsilon}_j(r)\|, \|\boldsymbol{\varepsilon}_j(s)\|) &= 0, \quad r \ne s\end{aligned}$$

Under these assumptions, our solution for $\mathbf{B}_j$ has minimum variance among all unbiased estimators.

However, the assumption of constant variance is unreasonable, since different positions in protein structures are conserved in varying degrees. Let us assume that errors are still uncorrelated, but that they have different variances:

$$\mathrm{Var}(\|\boldsymbol{\varepsilon}_j(s)\|) = \sigma^2/w_i \quad w_i > 0 \qquad (10)$$

A least-squares solution for this situation is equivalent to minimizing

$$\sum_{j=1}^{J}(\mathbf{M}_j\mathbf{B}_j - \overline{\mathbf{M}})^T\mathbf{W}(\mathbf{M}_j\mathbf{B}_j - \overline{\mathbf{M}}) \qquad (11)$$

where $\mathbf{W} = \mathrm{diag}(w_i)$ [18]. This weighted objective function shows that weights w_i can also be used to emphasize certain segments in the protein structure. Equivalently, positions with large variances will be given less weight.

The solution is obtained when the projection operator is $\mathbf{H}_j = \mathbf{M}_j(\mathbf{M}_j^T\mathbf{W}\mathbf{M}_j)^{-1}\mathbf{M}_j^T\mathbf{W}$. Alternatively, we can use the procedure described in the previous section if we first multiply each landmark matrix $\mathbf{M}_j$ on the left by $\mathbf{W}^{1/2}$, obtain the model as before, and then multiply that model on the left by $\mathbf{W}^{-1/2}$.

In order to obtain estimates for the variances, we might superimpose a family of protein structures using the ordinary regression solution. We could then compute the observed variability at each position as an estimate of the variance. These variances would then allow us to repeat the superposition using a weighted regression.

2.7 Affine Superpositions

Several steps of our algorithm compare one protein structure to another or to the regression model. We make these comparisons using the transformation matrices and offset vectors computed in the previous section. To superimpose the regression model $\overline{\mathbf{M}}$ onto a protein structure $\mathbf{S}_j$ or landmark matrix $\mathbf{M}_j$, we apply the inverse of the transformation matrix to obtain

$$\overline{\mathbf{M}}' = \overline{\mathbf{M}}\mathbf{B}_j^{-1} + \mathbf{1}\boldsymbol{\mu}_j^T \qquad (12)$$

where $\boldsymbol{\mu}_j$ is the offset vector for $\mathbf{M}_j$.

To superimpose one protein structure $\mathbf{S}_i$ onto another $\mathbf{S}_j$, we transform $\mathbf{S}_i$ into the model space and then transform it into the space of $\mathbf{S}_j$:

$$\mathbf{S}_i' = (\mathbf{S}_i - \mathbf{1}\boldsymbol{\mu}_i^T)\mathbf{B}_i\mathbf{B}_j^{-1} + \mathbf{1}\boldsymbol{\mu}_j^T \qquad (13)$$

where $\mathbf{B}_i$ and $\mathbf{B}_j$ are transformation matrices and $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ are offset vectors for $\mathbf{M}_i$ and $\mathbf{M}_j$, respectively.

2.8 Decomposition of Transformations

The affine superpositions described in the previous section may introduce shear components. The amount of shear may be determined as follows. Let us consider the superposition matrix $\mathbf{T} = \mathbf{B}_i\mathbf{B}_j$ that superimposes structure $\mathbf{S}_i$ onto $\mathbf{S}_j$. This matrix can be decomposed into a pure rotation $\mathbf{R}$ followed by a pure scaling $\mathbf{D}$, then a shear $\mathbf{Z}$:

$$\mathbf{T} = \mathbf{RDZ} \qquad (14)$$

where $\mathbf{R}$ is orthogonal, $\mathbf{D}$ is diagonal, and $\mathbf{Z}$ is upper triangular with ones along the diagonal. We can solve for the components by letting $\mathbf{G}$ be the Cholesky decomposition of $\mathbf{T}^T\mathbf{T}$ and setting $\mathbf{D}$ equal to the diagonal entries of $\mathbf{G}$. Then $\mathbf{Z} = \mathbf{D}^{-1}\mathbf{G}$ and $\mathbf{R} = \mathbf{TG}^{-1}$. The upper triangular entries of $\mathbf{Z}$ measure shear of each axis relative to other axes.

2.9 Regression Model using Orthogonal Transformations

Our framework for obtaining regression solutions can also handle orthogonal transformations. As in section 2.5, we assume that a correspondence is given and that each landmark matrix $\mathbf{M}_j$ is centered. When $\mathbf{B}_j$ is constrained to be orthogonal, computing a regression model requires an iterative algorithm. At each iteration, we use the old regression model, denoted as $\overline{\mathbf{M}}$, as a target structure, to obtain a new regression model, denoted as $\overline{\mathbf{M}}'$. For the first iteration, when there is no old model, we select one of the landmark matrices as the target.

We apply the Procrustes solution to rotate each landmark matrix onto the old model [7, 6]. Let $\mathbf{U}_j\mathbf{D}_j\mathbf{V}_j^T$ be the singular value decomposition of $\overline{\mathbf{M}}^T\mathbf{M}_j$. Then the optimal orthogonal transformation matrix of $\mathbf{M}_j$ onto $\overline{\mathbf{M}}$ is $\mathbf{B}_j = \mathbf{V}_j\mathbf{U}_j^T$. Therefore, the rotated version of the landmark matrix is $\mathbf{M}_j\mathbf{V}_j\mathbf{U}_j^T$. The new model is obtained by averaging the rotated landmark matrices and orthogonalizing the result. Orthogonalization may be performed using the QR decomposition or singular value decomposition.

We repeat the process until the difference in total squared distance between the old model $\overline{\mathbf{M}}$ and new model $\overline{\mathbf{M}}'$ is less than some tolerance criterion. In order to compare the two models, we use the same Procrustes procedure to rotate one model onto another.

3 Example

We now present a case study involving the globin family, chosen largely because it has been studied extensively in prior studies of protein structure families [1, 16, 5]. We studied the seven globin structures examined by Bashford and colleagues [1]: human deoxyhemoglobin α (PDB accession 4HHBA) and β (4HHBB), sperm whale deoxymyoglobin (5MBN), larval deoxyhemoglobin (from *Chironomous thummi*, 1ECD), sea lamprey cyanohemoglobin (2LHB), yellow lupin root nodule cyanoleghemoglobin (*Lupinus luteus*, 2LH3), and annelid worm deoxyhemoglobin (*Glycera dibranchiata*, 2HBG).

We implemented the algorithm in the statistical computing language S-Plus, except for the pairwise adaptive dynamic programming procedure, which was written in C and loaded dynamically into S-Plus. We executed the program on a Silicon Graphics O2 workstation with a 175 MHz MIPS R10000 processor. We obtained both the affine and orthogonal solutions, with the latter solution obtained using a tolerance criterion of 10^{-6}. Our algorithm required 12 CPU seconds to execute for the affine superposition and 27 CPU seconds for the orthogonal superposition.

The discrete curvature functions are shown in Figure 1. The figure also shows the location of the α-helices for each globin, as defined by Bashford and colleagues. Each discrete curvature function has discrete regions of relatively high and constant curvature—corresponding to the helices—separated by regions of lower, more variable curvature—corresponding to the loops. The discrete curvature is unusually variable in the C and D helices, and only 4HHBB, 5MBN, and 2LHB have clearly

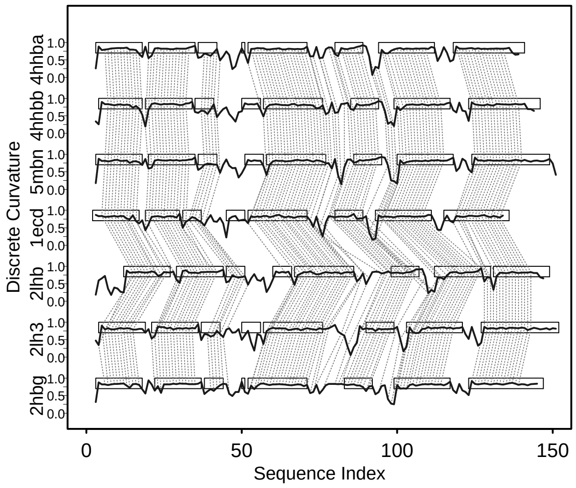

Figure 1: Discrete curvature functions and correspondence for globins. The discrete curvature function for each globin is drawn in heavy lines. The location of helices for each globin are denoted by rectangles. Corresponding landmarks are shown as dotted lines between discrete curvature functions.

defined D helices. Between helices F and G, all structures have a sequence of 2 to 3 amino acids where the curvature drops sharply. The common element to all sequences appears to be a small hydrophobic residue—valine or isoleucine—surrounded by one or more charged amino acids.

The relationship between discrete curvature and secondary structure is illustrated clearly in Figure 2, which maps discrete curvature onto the three-dimensional structures of a globin and an immunoglobulin. The high discrete curvature for α-helices in globins contrasts sharply with the low discrete curvature for β-strands in immunoglobulins. Loops generally have intermediate curvature.

In step 1, our algorithm found an initial correspondence of 71 landmarks by matching discrete curvatures to the reference structure 5MBN, chosen because it is longest. The pairwise dynamic programming procedure for matching discrete curvature is demonstrated in Figure 3 (left), which shows the match between 5MBN and 1ECD. The resulting set of landmarks for all structures is shown as dashed lines in Figure 1. Landmarks were found primarily in the helices, and virtually none in the regions between helices.

In step 2, the affine algorithm found 103 corresponding landmarks by matching coordinates to the reference structure 5MBN, chosen because it was closest to the regression model obtained in step 1. The coordinate-based pairwise dynamic programming procedure is illustrated in Figure 3 (middle). The orthogonal algorithm found 105 corresponding landmarks in its step

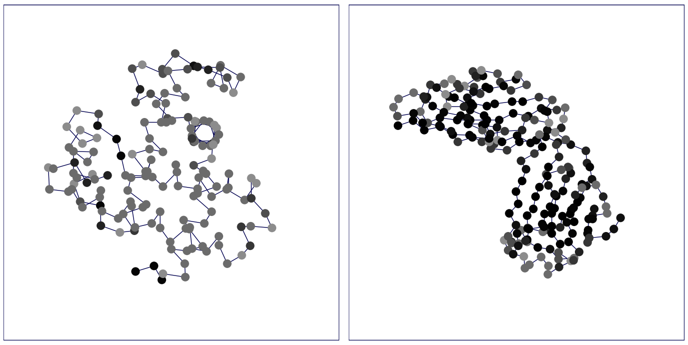

Figure 2: Relationship between discrete curvature and secondary structure. The figure on the left is sperm whale myoglobin (5MBN); on the right is the variable light chain of immunoglobulin NC41 (1NCA). Each Cα carbon is shaded according to its discrete curvature, from black (curvature of 0.0) to white (1.0).

2.

Step 3 of the affine algorithm refined the sets of landmarks in four iterations, retaining 103 landmarks at each step. Step 3 of the orthogonal algorithm required 11 iterations, retaining 105 landmarks at each step. The affine and orthogonal algorithms produced essentially the same corresponding landmarks, differing in only 10 positions. The orthogonal algorithm found an additional two landmarks just before the first landmark found by the affine algorithm. The two methods differed slightly in the correspondence for 2LH3 at three positions at the beginning of the B helix. Finally, the two methods had different correspondences in five positions between the E and F helices.

The final superposition using the affine method is shown in Figure 4, which shows each of the original protein structures superimposed onto the space of 5MBN.

The amount of structural variation can be quantified by measuring residual deviations from the model. We compute the residuals by superimposing the regression model onto each landmark matrix and measuring the differences between corresponding coordinates. The resulting residual matrix $\mathbf{e}_j$ provides an estimate of the error matrix ε_j in equation 3. If we assume that ε_j is isotropic, the unbiased estimate of the variance at each landmark s is the mean square error

$$\sum_{j=1}^{J} \|\mathbf{e}_j(s)\|^2/(J-1)$$

Figure 5 shows the variability across all landmarks, using both the affine and orthogonal methods. The two variability estimates are very similar, with the affine method giving a slightly better fit at the end of the E helix. Our analysis shows that the mid-regions of helices E and G are conserved the most, which makes

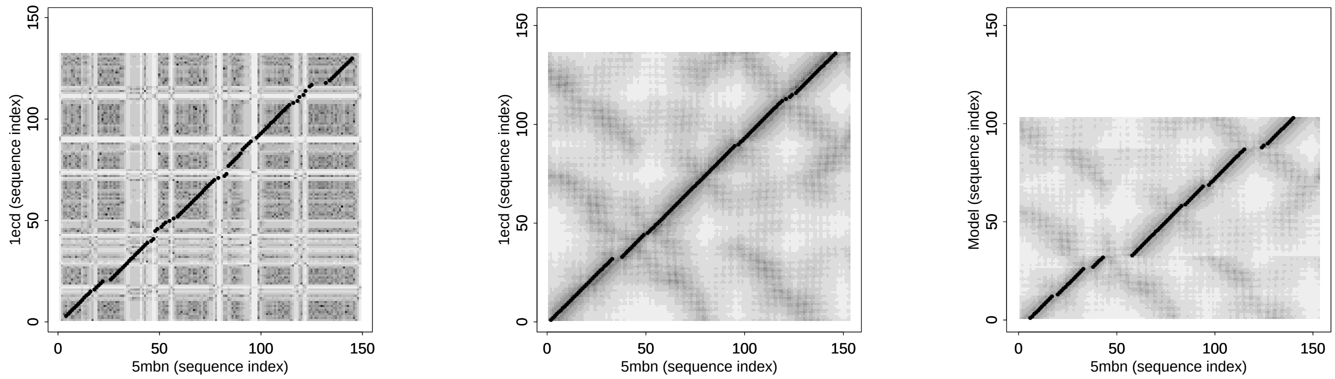

Figure 3: Dynamic programming method for finding correspondences, using discrete curvature (left), reference-based coordinate distance (middle), and model-based coordinate distance (right). Distance metrics are plotted as images, with darker intensity representing smaller distance. Optimal solutions are plotted as points on each graph.

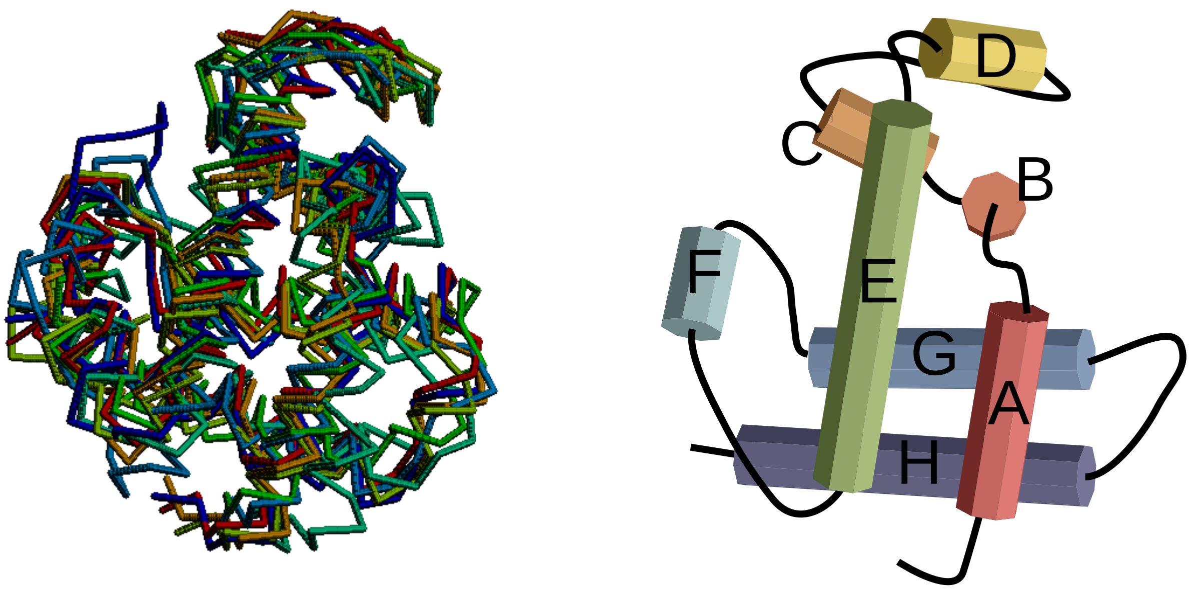

Figure 4: Affine superposition of globins. The superposition is shown on the left, with a schematic on the right. The seven globins are represented by line segments connecting their Cαatoms.

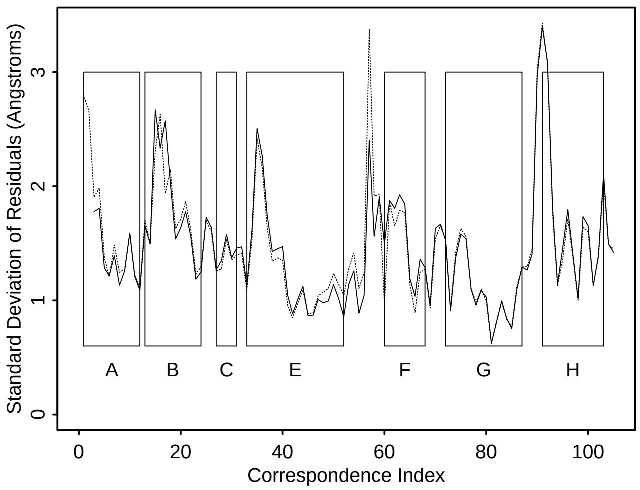

Figure 5: Structural variability of the globins. Residual standard deviations are plotted versus correspondence index. The variability estimate using the affine method is shown as a solid line, whereas the estimate using the orthogonal method is shown as a dashed line. The curve for the affine method is shifted two positions to the right, so that positions correspond. Helices are marked by solid rectangles and labeled from A to H.

physiologic sense, because both helices make close contact with the heme group.

4 Discussion

We have developed a general framework for analyzing families of protein structures. Our approach generalizes the superposition problem to apply methods from statistical regression analysis. Our method can find regression models using either affine or orthogonal transformation matrices. Affine transformations may add flexibility to the comparison of different structures, which may allow us to see similarities among protein structures that more constrained methods may miss. For instance, by relaxing the rotational constraint, Diamond [2] was able to superimpose oxy- and deoxyhemoglobin appropriately.

Our method is also able to handle robust least squares regression using weights, so that positions with high variability influence the result less than positions with less variability. We have not yet explored the use of weights, and the globin example would not be particularly illustrative because the globins exhibit relatively low variability throughout.

We have introduced a new method for finding gap penalties, based on the data. Adaptive methods may perhaps be useful in other applications of dynamic programming, which is used widely in computational biology. Adaptive gap penalties are particularly useful when we wish to align two functions as completely as possible, even though the overall alignment may be relatively poor. Moreover, the magnitudes of the resulting gap penalties give an indication of how well the two functions align. There is room for further work in this area.

The speed and simplicity of our approach, especially when using affine transformations, also creates new opportunities for further study. Affine transformations not only speed up the superposition step, but in the case of the globins, required much fewer iterations to converge on a set of corresponding landmarks. Our method generates regression models in a matter of seconds, and this speed may permit other types of investigations, such as cluster analyses of protein structures.

Many methods exist for finding initial correspondences, such as secondary structure methods [14]. One advantage of the discrete curvature method presented here is that it does not require prior definitions of secondary structure. Moreover, discrete curvature may be useful in other applications. Discrete curvature reveals secondary structure elements readily, and matches of discrete curvature may show quickly whether two structures are similar, even without performing a superposition. Hence, discrete curvature may be useful for scanning the structural database quickly.

Studies of curvature may also provide insights into protein structure. Currently, local conformations of amino acids are characterized by ϕ-ψ angles, which represent torsional angles between adjacent residues. Because curvature represents local conformation differently, further studies of curvature may enhance our understanding of the sequence-structure relationship in proteins.

Acknowledgments

TDW is a Howard Hughes Medical Institute Physician Postdoctoral Fellow. TH is supported by NSF grant DMS-9504495 and NIH grant ROI-CA-72028. SCS is supported by NLM training grant LM-07033. DLB is supported by NLM grant LM-05716.

References

[1] Donald Bashford, Cyrus Chothia, and Arthur M. Lesk. Determinants of a protein fold: Unique features of the globin amino acid sequences. *J. Mol. Biol.*, 196:199–216, 1987.

[2] R. Diamond. On the comparison of conformations using linear and quadratic transformations. *Acta Cryst.*, A32:1–10, 1976.

[3] R. Diamond. On the multiple simultaneous superposition of molecular structures by rigid body transformations. *Protein Sci.*, 1:1279–1287, 1992.

[4] Paul R. Gerber and Klaus Müller. Superimposing several sets of atomic coordinates. *Acta Cryst.*, A43:426–428, 1987.

[5] Mark Gerstein and Russ B. Altman. Using a measure of structural variation to define a core for the globins. *Comp. Appl. Biosci.*, 11(6):633–644, 1995.

[6] Gene H. Golub. *Matrix Computations*, third edition. Johns Hopkins University Press, 1996.

[7] Colin Goodall. Procrustes methods in the statistical analysis of shape. *J. Royal Stat. Soc. B*, 53(2):285–339, 1991.

[8] Osamu Gotoh. An improved algorithm for matching biological sequences. *J. Mol. Biol.*, 162:705–708, 1982.

[9] Trevor Hastie, Eyal Kishon, Malcolm Clark, and Jason Fan. A model for signature verification. Technical report, AT&T Bell Laboratories, 1992.

[10] Liisa Holm and Chris Sander. Protein structure alignment by alignment of distance matrices. *J. Mol. Biol.*, 233:123–138, 1993.

[11] Simon K. Kearsley. An algorithm for the simultaneous superposition of a structural series. *J. Comput. Chem.*, 11(10):1187–1192, 1990.

[12] D. Sankoff and J. B. Kruskal. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison.* Addison-Wesley, 1983.

[13] A. Shapiro, J. D. Botha, A. Pastore, and A. M. Lesk. A method for multiple superposition of structures. *Acta Cryst.*, A48:11–14, 1992.

[14] Amit P. Singh and Douglas L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations. In *Proceedings, Intl. Conf. on Intelligent Systems in Molecular Biology*, pages 284–293, 1997.

[15] William R. Taylor, Tomas P. Flores, and Christine A. Orengo. Multiple protein structure alignment. *Protein Sci.*, 3:1858–1870, 1994.

[16] William R. Taylor and Christine A. Orengo. Protein structure alignment. *J. Mol. Biol.*, 208:1–22, 1989.

[17] Andrej Šali and Tom L. Blundell. Definition of general topological equivalence in protein structures: A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J. Mol. Biol.*, 212:403–428, 1990.

[18] Sanford Weisberg. *Applied Linear Regression*, second edition. John Wiley and Sons, 1985.

[19] Thomas D. Wu, Scott C. Schmidler, Trevor Hastie, and Douglas L. Brutlag. In *Pacific Symposium on Biocomputing*, 3:507–518, 1998.