Biology **2013**, *4*, 1296-1310; doi:10.3390/biology2041296

OPEN ACCESS

biology

ISSN 2079-7737

www.mdpi.com/journal/biology

Article

Dynamic Programming Used to Align Protein Structures with a Spectrum Is Robust

Allen Holder [1,*], Jacqueline Simon [1], Jonathon Strauser [2], Jonathan Taylor [1] and Yosi Shibberu [1]

[1] Department of Mathematics, Rose-Hulman Institute of Technology, Terre Haute, IN 47803, USA; E-Mails: simonjc@rose-hulman.edu (J.S.); taylorj7@rose-hulman.edu (J.T.); shibberu@rose-hulman.edu (Y.S.)

[2] Roche Diagnostics, Indianapolis, IN 46256, USA; E-Mail: strauserj@upmc.edu

* Author to whom correspondence should be addressed; E-Mail: holder@rose-hulman.edu; Tel.: 812-877-8682.

Received: 3 October 2013; in revised form: 23 October 2013 / Accepted: 8 November 2013 / Published: 20 November 2013

Abstract: Several efficient algorithms to conduct pairwise comparisons among large databases of protein structures have emerged in the recent literature. The central theme is the design of a measure between the C_α atoms of two protein chains, from which dynamic programming is used to compute an alignment. The efficiency and efficacy of these algorithms allows large-scale computational studies that would have been previously impractical. The computational study herein shows that the structural alignment algorithm eigen-decomposition alignment with the spectrum (EIGAs) is robust against both parametric and structural variation.

Keywords: protein alignment; structural bioinformatics; dynamic programming

1. Introduction

The problem of aligning protein structures to infer functional similarity is a stalwart within the arena of computational biology; see [1–8] as recent examples. Over the last couple of years, there have been several publications that have promoted fast algorithms for database-wide comparisons [1,8–10] (see the related work in [11–19]). These algorithms are designed to efficiently discern if two proteins share a similar molecular structure. Since structural similarity often aligns with functional similarity, these

algorithms promote making pairwise comparisons among large databases with the intent being to identify functionally similar groups.

Classical structure alignment algorithms like Dali [20], CE [21], TM-align [22] and SSAP [23] provide accurate alignments, but are typically too slow for efficient all-against-all structural comparisons or database searches. One method of accelerating an algorithm is to modify it to run on a GPU (graphics card). Pang *et al* [24] report a 36-fold speedup for TM-align modified for GPUs. Another approach is to reduce the search space by pruning unlikely matches through a process that clusters structures and then compares cluster representatives [25]. However, such comparisons are typically between structures with low similarity, and the work of Pascual-Garcia *et al.* [26] provides evidence that low structural similarity comparisons are better represented by a network of similarities rather than discrete fold clusters. The issue is that low structural similarity can lead to the loss of transitivity, *i.e.*, protein A may be similar to protein B and protein B similar to protein C, and yet A may not be similar to C. A third approach is to combine a fast algorithm to filter comparisons with a slower algorithm to increase accuracy. For example, CATHEDRAL [27] combines a fast secondary structure-based algorithm with a slower double-dynamic programming algorithm.

The modern cohort of fast structural alignment algorithms sacrifices accuracy with regard to the pairwise comparisons in exchange for increased speed. Many of the fast algorithms (so-called 1D algorithms [3]) have a single application of dynamic programming (DP) at their core. However, so called 0D, or "finger-print," algorithms [3] have emerged that are, in principle, even faster than DP-based 1D algorithms. FragBag [28] is one such 0D algorithm. FragBag represents protein structures as histograms of backbone fragments, and its computational speed is better than DP, due to the efficiency of comparing histograms.

The speed with which the most recent algorithms can compare proteins prompts a wealth of numerical studies that would have been previously difficult, if not impossible. We harness this efficiency to test if the structural alignment algorithm eigen-decomposition with the spectrum (EIGAs) is robust with regards to parametric variation and modeling uncertainty. Like many of the recent, efficient algorithms, EIGAs uses dynamic programming (DP) as its underlying computational framework. DP depends on parameters that generally need to be tuned to the application and, in some cases, to the particular problem instance. This begs the question of how sensitive the recent DP-based algorithms are to selecting quality parameters. If parameters have to be selected somewhat precisely to obtain accuracy and/or have to be tuned per dataset, then the technique would lack sufficient robustness to instill confidence on much larger datasets, which would preclude individualized parametric tuning. We show that EIGAs is robust to parameter selection by showing that it remains highly effective at identifying structurally similar proteins over a breadth of parametric values.

The optimization model solved by DP depends on a measure of the structural similarity between two proteins. However, the coordinates of a residue are not known with perfect certainty, and the structural similarity measure and, subsequently, the alignments themselves are subject to possible adjustments, as a protein's 3D structural description varies within appropriate tolerances. As with parametric variation, if an algorithm is sensitive to small variations in structure, then it will likely perform poorly, since real datasets always have a level of uncertainty. We show that EIGAs is robust against such uncertainty by showing that it identifies a preponderance of structurally similar proteins as the coordinates of the C_α

atoms are perturbed randomly. We account for uncertainty by assuming that the atomic coordinates of the C_α atoms are probability distributions, whose standard deviations are scaled B-factors.

The next section discusses how DP is used to efficiently align protein structures. This discussion points to why DP might have a natural affinity for accounting for parametric variation and structural uncertainty. Sections 3 and 4 specify EIGAs' DP model and detail the computational environment used to conduct our numerical experiments. Section 5 reports on our numerical tests showing that EIGAs is robust against parametric adjustment. Section 6 reports on the robustness of EIGAs as coordinate uncertainty is considered.

2. Dynamic Programming and Robustness

Dynamic programming originated with Bellman [29] and was first used to align biological sequences by Needleman and Wunsch [30]. The discrete setting of structural alignment means that the iterative results of the DP recursion are decided by selecting a best value from among a few possibilities. This immediately points to the fact that these numbers can adjust over nearby values without altering the optimal solution, and it is this observation that we promote.

Suppose we want to align two sequences, one indexed by i and the other by j. Allowing S_{ij} to be the similarity between element i of the first sequence and element j of the second, we have that a common recursion defining an optimal alignment is:

$$V_{ij} = \min \begin{cases} V_{i-1,j} + \rho \\ V_{i,j-1} + \rho \\ V_{i-1,j-1} + S_{ij}, \end{cases} \tag{1}$$

where ρ is a penalty for inserting or continuing a gap. Calculating the recursion over i and j produces the optimal value over all possible alignments, and an optimal alignment itself can be calculated by backtracing the steps used to construct the optimal value. As an illustrative example, assume the similarity values are tabulated so that:

$$\begin{array}{c} \\ i=1 \\ i=2 \\ i=3 \\ i=4 \end{array} \begin{array}{c} \begin{array}{ccccc} j=1 & j=2 & j=3 & j=4 & j=5 \end{array} \\ \begin{bmatrix} 0.1 & 0.2 & 0.8 & 0.7 & 1.2 \\ 0.5 & 0.4 & 0.2 & 0.1 & 0.5 \\ 0.2 & 0.3 & 0.1 & 0.2 & 0.2 \\ 0.8 & 0.7 & 0.6 & 0.1 & 0.1 \end{bmatrix} \end{array} = S.$$

The values of V with $\rho = 0.3$ are calculated from the recursion in (1):

$$\begin{array}{c} \\ i=1 \\ i=2 \\ i=3 \\ i=4 \end{array} \begin{array}{c} \begin{array}{ccccc} j=1 & j=2 & j=3 & j=4 & j=5 \end{array} \\ \begin{bmatrix} \boxed{0} & 0.3 & 0.6 & 0.9 & 1.2 & 1.5 \\ 0.3 & \boxed{0.1} & \boxed{0.4} & 0.7 & 1.0 & 1.3 \\ 0.6 & 0.4 & 0.5 & \boxed{0.6} & 0.8 & 1.1 \\ 0.9 & 0.7 & 0.7 & 0.6 & \boxed{0.8} & 1.0 \\ 1.2 & 1.0 & 1.0 & 0.9 & 0.7 & \boxed{0.9} \end{bmatrix} \end{array} = V.$$

Each element of V, except the initial zero in the upper left, is calculated from (1). For example, the 0.4 in the $i = 1$, $j = 2$ position is:

$$0.4 = \min \begin{cases} 0.1 + 0.3 & \text{(add gap penalty from left)} \\ 0.3 + 0.2 & \text{(pair } i = 1 \text{ with } j = 2 \text{ from the upper left diagonal)} \\ 0.6 + 0.3 & \text{(add gap penalty from top).} \end{cases}$$

Once V is constructed, the score of the optimal alignment is in the bottom right element, and the alignment itself is calculated by backtracing a path that produced this score. In this case, the unique elements of the optimal path are circled, and the optimal alignment/solution is:

Protein A	1	_ _	2	3	4
Protein B	1	2	3	4	5
Pair Value	0.1	0.3	0.2	0.2	0.1.

An optimal solution depends on the gap penalty, ρ, and the individual values of the structural similarity measure, S_{ij}. However, solutions are not necessarily sensitive to changes in these values. Indeed, a simple, albeit tedious, calculation shows that the example's unique solution remains optimal over the substantial range of $0.15 < \rho < \infty$. Similar calculations show that each individual, S_{ij}, may vary with $\rho = 0.3$ over the intervals noted below:

$$\begin{bmatrix} (0.0, 0.2) & (0.1, \infty) & (0.0, \infty) & (0.0, \infty) & (0.0, \infty) \\ (0.0, \infty) & (0.3, \infty) & (0.0, 0.3) & (0.0, \infty) & (0.0, \infty) \\ (0.0, \infty) & (0.0, \infty) & (0.0, \infty) & (0.0, 0.3) & (0.0, \infty) \\ (0.0, \infty) & (0.0, \infty) & (0.0, \infty) & (0.0, \infty) & (0.0, 0.2) \end{bmatrix}.$$

This example illustrates that a solution from DP is stable as the input data is slightly, and, in many cases, robustly, perturbed. The sizes of the stability ranges depend on the particular problem, and they can be quite small. However, changes in the alignment resulting from a slight extension beyond one of these intervals can be minor, and the resulting score may still suffice as a delineating value. The fundamental question of this paper is whether or not the efficacy of EIGAs degrades as its input data is adjusted. If so, then either it is highly sensitive to changes in its algorithmic parameters, such as the gap penalty, ρ, or it is highly sensitive to minor adjustments in its model of structural similarity, S.

There are numerous variants of the simple recursion in (1). The gap penalty, ρ, in the example penalizes opening and continuing a gap identically. If the penalties for opening and continuing a gap are different, then an affine gap penalty is being used. Some studies in computational biology have indicated a benefit to an affine gap model [31,32]. However, affine models require an additional parameter over their non-affine counterparts, which complicates parameter tuning for the application at hand. Our numerical work uses an affine gap penalty, and we let ρ_o and ρ_c be the penalties for initiating and continuing a gap, respectively.

While the discussion above depicts DP favorably, we would be remiss to ignore some of its downsides. All uses of DP for database applications only consider sequential alignments. Non-sequential alignments are important in some, and, indeed, possibly many, cases [33–35]. Adapting DP to accommodate for non-sequential alignments, say by re-ordering the manner in which DP iterates along the backbone,

would be an important direction for future research. The use of bipartite graph matching in place of DP to obtain non-sequential alignments should also be investigated [36].

3. Eigen-Decomposition with the Spectrum

A protein (chain) is a linear polymer of amino acid residues that folds into a unique 3D structure. The amino acid residues are held together with strong covalent bonds, and the covalent backbone is comprised of a repeating pattern of atoms identical for each residue. However, each residue is distinguished by a side chain of atoms that is not on the backbone, but is attached to the alpha carbon backbone atom, denoted C_α, for each residue. We represent each residue by the coordinates of its C_α atom. An alignment between two protein chains is a pairing of residues between the proteins, which reduces to a pairing between the C_α atoms along the two backbones.

Let d_{kt} be the distance between residues k and t of a single protein. The smooth contact matrix for the protein is:

$$C_{kt} = \begin{cases} 1 - d_{kt}/\kappa, & \text{if } d_{ij} \leq \kappa \\ 0, & \text{otherwise.} \end{cases}$$

The parameter, κ, defines the largest distance at which the model permits a nonzero relationship between residues k and t. Residue pairs whose distance is less than κ receive a linearly-scaled value of d_{kt}. We note that $C_{kk} = 1$, since $d_{kk} = 0$. This property implies that C is positive definite for appropriately selected κ [37], and hence, C can be factored as $C = UDU^T$, where $U^T = U^{-1}$ and D is a diagonal matrix of positive eigenvalues. If we let $R = \sqrt{D}U^T$, then $C = R^T R$. The column vectors of R are called intrinsic contact vectors, and they support a geometric perspective of the alignment problem [8].

The similarity between residues i and j of different proteins is $S_{ij} = |\lambda_i - \lambda_j|/|\lambda_i + \lambda_j|$, where λ_i and λ_j are the eigenvalues associated with the two residues. Specifically, residue i is assigned λ_p if $|R_{pi}| = \max_q |R_{qi}|$; see [8] for additional details. This eigenvalue assignment is not that of principle component analysis (PCA) or normal mode analysis (NMA), as it includes many smaller eigenvalues, which are typically more sensitive to deviations in the input data. As an example, the distribution of EIGAs' eigenvalue assignments for the Proteous300 dataset is tabulated below.

	smallest									largest
size	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
assignments	0.42%	4.03%	6.36%	9.15%	11.41%	18.06%	12.18%	9.66%	3.93%	24.79%

Nearly 75% of the residues are assigned eigenvalues whose magnitude is below 90% of the largest eigenvalue, and about 31% of the residues are assigned eigenvalues whose magnitude is below 50% of the largest eigenvalue. Unlike PCA or NMA, EIGAs' assignment associates each residue with the eigenvalue of the nearest eigenspace, and as the distribution above shows, many of a protein's more minor eigenvalues are used by EIGAs.

4. The Computational Setting

The EIGAs code is implemented in Python, although portions, such as the DP core, are written in C as Python extensions via SWIG [38]. BioPython [39] is used to parse the protein database (pdb) files, and NumPy is used to factor the smooth contact matrices. All computations for this study were

completed on a 2.2 GHz chipset with 7 G of RAM, and all tests were conducted on the Proteus300 [1] dataset, which consists of 300 proteins belonging to 30 known families, as identified by the Structural Classification of Proteins (SCOP) database [40–42]. EIGAs requires about three minutes to complete the 44,850 pairwise comparisons of Proteus300 in serial (about 0.0041 seconds per comparison). The preprocessing step of encoding each of the 300 proteins as a list of eigenvalues also requires about three minutes (approximately 0.6240 seconds per protein). Using Python's multiprocessing package, the same task can be distributed to different CPUs, although we have not experienced speed-ups.

EIGAs' computational efficiency compares favorably with other published results. The Eig_7 algorithm in [12] can complete the Proteus300 dataset in about six hours, and A_purva [1] is able to finish the dataset in about 13.5 hours. The numerical results in [9] show that GOSSIP can complete the Proteus300 dataset in as little as 20 minutes and 44 seconds. Kpax [43] reports that it can accomplish 2,980 comparisons per second, which equates to about 15 seconds for Proteus300. LNA has the fastest reported results on GPUs and is capable of finishing the Proteus300 dataset in under a second [9].

The earlier numerical results of EIGAs in [8] have been improved in many ways. The earlier version of EIGAs used only a non-affine gap model, and the DP algorithm had been coded in Matlab. We gained a significant speedup with a C implementation of DP. EIGAs had previously used a non-affine gap penalty of one along with the un-normalized similarity value of $S_{ij} = |\lambda_i - \lambda_j|$. EIGAs now uses an affine gap model, and similarity values are normalized as $S_{ij} = |\lambda_i - \lambda_j|/|\lambda_i + \lambda_j|$, which ensures that the score is always between zero and one. Lastly, the previous benchmark had been a simple count of the proteins whose nearest neighbor had shared the same family classification. Assessments are now made from a receiver operating characteristic (ROC) curve, which is an improved evaluative tool that parameterizes the trade-off between positive and negative results.

5. Robustness Against Parametric Variation

Three algorithmic parameters affect alignments, those being the cutoff value, κ, the gap opening penalty, ρ_o, and the gap continuation penalty, ρ_c. We incremented κ from four to 23 Å with a step size of 1 Å and incremented each of the gap penalties from zero to one with a step size of 0.1. The 30 families of the Proteus300 dataset were randomly divided into three sub-groups of 10 families (100 proteins) each. We adapted a standard three-fold cross-validation test in an attempt to quantify how parametric tuning on a small collection of proteins might foreshadow success on a larger, and possibly disjoint, dataset. Instead of tuning on two of the three sub-groups and then validating on the third, parameters were tuned on a single dataset and then validated on the remaining two. The tuning on each sub-group was over the space of all $20 \times 10 \times 10 = 2,000$ triples (κ, ρ_o, ρ_c). Hence, each of the three tunings required $2,000 \times \binom{100}{2} = 9,900,000$ applications of DP, which, at 0.0041 seconds per solve, is about 11.5 hours.

An ROC curve is generated for each (κ, ρ_o, ρ_c), which is a parametric plot of the false positive rate (FPR) against the true positive rate (TPR), as the value deciding structural similarity varies. As an example, suppose the optimal alignments among five proteins have the optimal DP values in Table 1. The best (minimum) agreement is between proteins C and E, which are aligned by EIGAs' DP algorithm with a globally optimal value of 0.08. The worst agreement is between proteins A and E, which even when optimally aligned by EIGAs receive a score of 0.68. Let τ satisfy $0.08 \leq \tau \leq 0.68$, and for the

moment, consider $\tau = 0.20$. Only three alignments have a value below τ, those being between pairs (A,D), (C,E) and (D,E), but only the first two correctly pair proteins within the same family. There are a total of 10 possible pairs in this example, but only four correctly pair proteins of the same family. Therefore, for $\tau = 0.20$, the TPR is $2/4$, *i.e.*, we correctly identified two of the four pairings that share the same family. Since (D,E) was the only false positive out of six possible, the FPR for $\tau = 0.20$ is $1/6$. Varying τ over the interval $[0.08, 0.68]$ gives a parametric plot of FPR *versus* TPR. Note that TPR and FPR are both one for $\tau \leq 0.08$ and are both zero for $\tau > 0.68$.

Table 1. Illustrative values assigned to optimal alignments from dynamic programming (DP).

	Prot.A	Prot. B	Prot. C	Prot. D	Prot. E
Prot. A	0.00	0.21	0.43	0.13	0.68
Prot. B	0.21	0.00	0.61	0.26	0.34
Prot. C	0.43	0.61	0.00	0.80	0.08
Prot. D	0.13	0.26	0.80	0.00	0.19
Prot. E	0.68	0.34	0.08	0.19	0.00
Family ID	1	1	2	1	2

A perfect classification has a TPR of one and an FPR of zero, and hence, it is desirable to have a ROC curve pass as close to this point as possible. A random classifier gives identical TPR and FPR values. Therefore, if an ROC curve is above the TPR = FPR line, then the classification method is better than random classification, but if the ROC curve is below the TPR = FPR line, then the classification is worse than random. A common metric is the area under the ROC curve (AUROC), which is at best one and at worst zero. The area of the ROC curve along with its minimum distance to a perfect classification are the metrics used to evaluate the alignments for each (κ, ρ_o, ρ_c).

Figures 1 through 3 illustrate the sensitivity of EIGAs' alignments as evaluated by AUROC. For each individual value of κ, ρ_o and ρ_c, the best possible AUROC over the other two parameters is plotted. The sensitivity is greatest with respect to κ, and while AUROC exceeded 0.9 in all cases, the sharp rise to well over 0.95 as κ increases demonstrates that κ should not be small. The best values were $\kappa = 19, 21,$ and 16 over the three sub-groups for respective best AUROCs of 0.9767, 0.9959 and 0.9801. The alignments were nearly insensitive to ρ_o, as illustrated by the nearly constant graphs in Figure 2. Indeed, each ρ_o yielded an AUROC exceeding 0.97. From Figure 3, we see that ρ_c should not be selected too small, since, if so, the alignments do not accurately predict family classification. However, quality alignments with AUROCs over 0.95 are possible, as long as $0.1 \leq \rho_c \leq 0.9$. The best values of ρ_c over the three sub-groups are 0.3, 0.3 and 0.2 with corresponding AUROCs of 0.9767, 0.9960 and 0.9801.

Our computational results show that AUROC exceed 0.95, as long as $7 \leq \kappa \leq 23$, $0 \leq \rho_o \leq 1$ and $0.1 \leq \rho_c \leq 0.9$. These parametric ranges are wide and account for over 67% of the tested parameters. We conclude that EIGAs is robust with respect to parametric tuning. This outcome suggests that it might be possible to tune parameters for general use on small datasets. To test this concept, we used the parameters with the best AUROC from each sub-group on the remaining proteins of Proteus300. The results are shown in Table 2, and they demonstrate that it is reasonable to assume that parameter tuning

on smaller datasets will result in quality parameters for larger datasets. On average, AUROC degraded from 0.9842 to 0.9742 over the three sub-group tests, although AUROC increased on the first sub-group. The TPR and FPR values listed in Table 2 are those closest to perfect classification, and they show that 90% of EIGAs' alignments correctly pair proteins of the same family, assuming that an appropriate threshold, τ, is selected. Likewise, less than 9% of family associations predicted by EIGAs are incorrect.

Figure 1. The best area under the receiver operating characteristic curve (AUROC) over all (ρ_o, ρ_c) for each κ. Each of the three curves corresponds with one of the random sub-groups of the Proteus300 dataset.

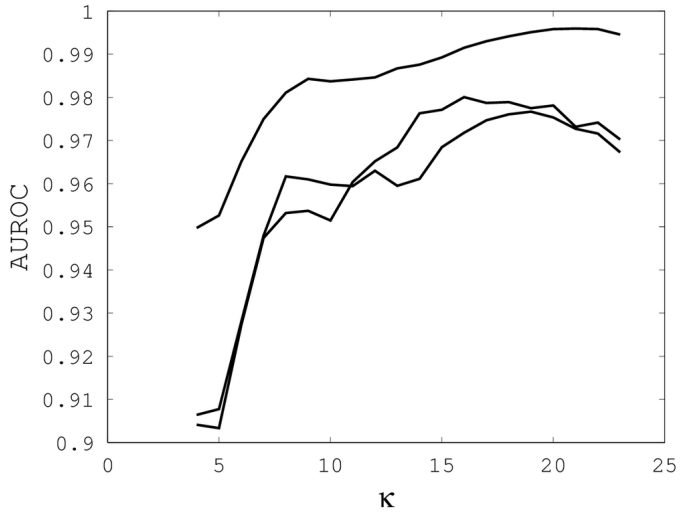
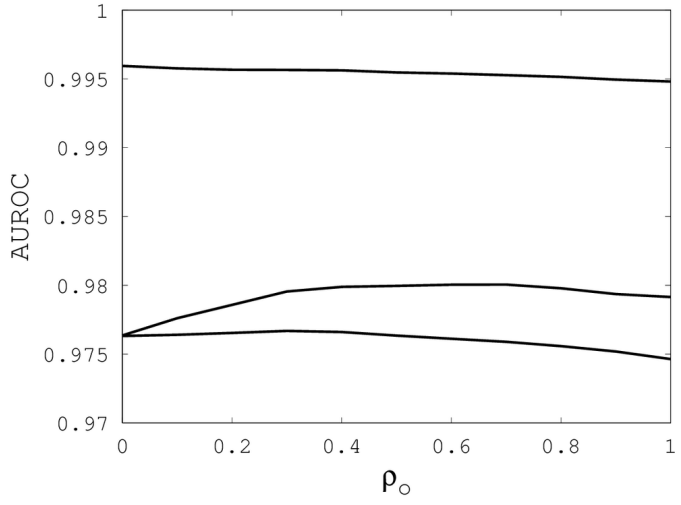


Figure 2. The best AUROC over all (κ, ρ_c) for each ρ_o. Each of the three curves corresponds with one of the random sub-groups of the Proteus300 dataset.

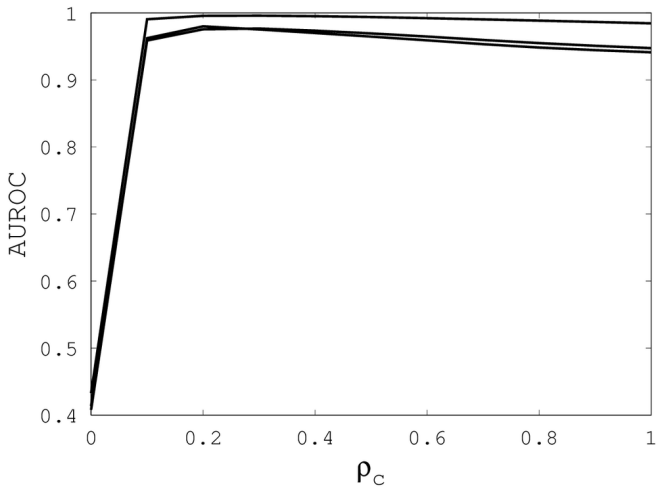


The parameter tuning suggests that $\kappa = 19$, $\rho_o = 0.5$ and $\rho_c = 0.3$ will generally result in alignments of sufficient quality to identify a high percentage of proteins with the same family classification. Using these parameters on the entire Proteus300 dataset gives an AUROC of 0.9787 and a best TPR and FPR of 0.9267 and 0.0704. The ROC curve is shown in Figure 4. As a point of observation, the EIGAs scoring method consistently resulted in the best TPR and FPR for $0.12 \leq \tau \leq 0.15$. The best TPR and FPR for the entire Proteus300 dataset occurred at $\tau = 0.1458$.

Table 2. Parameters κ, ρ_o and ρ_c were tuned to give the best possible AUROC for three randomly selected sub-groups of Proteus300. The results of using these parameters on the remaining, disjoint set of proteins are listed in the last three columns. TPR, true positive rate; FPR, false positive rate.

	Best Tuned Parameters per Sub-Group				Results on Remaining Proteins		
	AUROC	κ	ρ_o	ρ_c	AUROC	TPR	FPR
Sub-Group 1	0.9767	19	0.3	0.3	0.9801	0.9389	0.0724
Sub-Group 2	0.9959	21	0.0	0.3	0.9696	0.9067	0.0836
Sub-Group 3	0.9801	16	0.7	0.2	0.9728	0.9100	0.0848

Figure 3. The best AUROC over all (κ, ρ_o) for each ρ_c. Each of the three curves corresponds with one of the random sub-groups of the Proteus300 dataset.

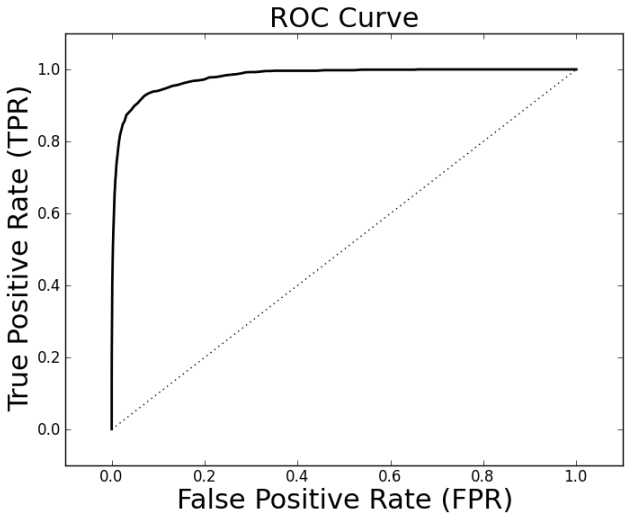


6. Robustness Against Coordinate Uncertainty

Beyond parametric variation, optimal alignments also depend on the C_α coordinates, since they decide the similarity values, S_{ij}, for a fixed value of κ. As far as we are aware, no computational evaluation has been proposed to measure an algorithm's ability to account for 3D variability, which is not surprising, since earlier alignment algorithms required lengthy computations. However, EIGAs' speed supports the repeated database-wide comparisons needed to study how it reacts as protein descriptions vary over experimental tolerances.

There are two common methods for establishing a protein's structure: X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy. Each accounts for variability differently, and these differences need to be reconciled to have a consistent study of coordinate uncertainty. We reconcile the differences probabilistically and assume that the coordinates of each C_α are random variables that can be modeled from either experiment. We then draw a sample dataset from these distributions and align them with EIGAs. Each run of EIGAs on a sample dataset is evaluated by AUROC, with $\kappa = 19$, $\rho_o = 0.5$ and $\rho_c = 0.3$, as determined in Section 5.

Figure 4. The receiver operating characteristic (ROC) curve for the entire Proteus300 dataset with $\kappa = 19$, $\rho_o = 0.5$ and $\rho_c = 0.3$.



NMR experiments generate multiple descriptions of a protein in aqueous solution, and pdb files from NMR experiments include numerous 3D models. An overlay of all 20 models from the NMR experiment of the protein structure 1NTR is shown in Figure 5. The NMR technique is advantageous because proteins are in a natural, aqueous environment, and hence, the renderings hopefully represent conformations of a folded protein as they exist naturally. The disadvantage is that only small proteins lend themselves to this type of experiment. We compute the sample mean and variance of the C_α coordinates over the different models and assume that each C_α coordinate is normally distributed as:

$$x_i \sim N(\bar{x}_i, s^2\sigma_x),\ y_i \sim N(\bar{y}_i, s^2\sigma_y),\ \text{and}\ z_i \sim N(\bar{z}_i, s^2\sigma_z),$$

where the means are sample means and the variances are scaled sample variances. The use of the scale parameter is discussed momentarily.

X-ray crystallography requires that a protein first be crystallized, and a solution must become saturated enough to create a crystalline structure that can then be imaged with X-rays. The advantage over NMR is that X-ray crystallography more easily accommodates large proteins. The disadvantage is that the 3D descriptions are not of the proteins in a natural aqueous solution. Uncertainty for X-ray crystallography experiments is expressed by a list of B-factors, each of which assesses the variation of the spacial coordinates of a specific atom. A B-factor, often called a Debye-Waller factor, is a scaled version of the mean squared displacement of an atom. If $\bar{r}_i$ is the mean of the i-th residue's C_α coordinates over the crystalline structure and r_i is the random position of the atom, then the B-factor of the i-th residue is:

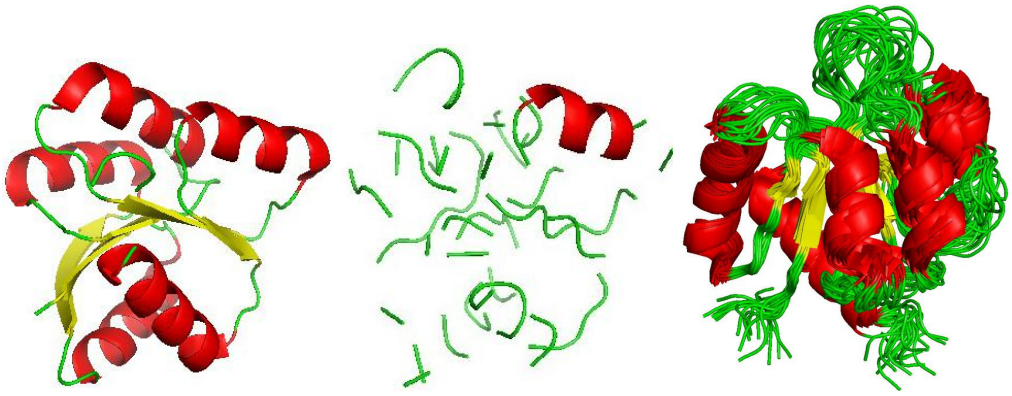
$$B_i = 8\pi^2\,\mathrm{E}((r_i - \bar{r}_i)^T(r_i - \bar{r}_i)) = 8\pi^2\,\sigma_i^2,$$

where E is the expected value and σ^2 is the variance. To assess EIGAs' sensitivity to coordinate perturbation, we assume that the coordinates of the C_α atoms are uncorrelated normals, whose variances are scalar multiples of a third of σ_i^2. Hence, for each X-ray crystallography experiment, the coordinates of the C_α atoms are the random variables:

$$x_i \sim N\left(\bar{x}_i, \frac{s^2 B_i}{24\pi^2}\right),\ y_i \sim N\left(\bar{y}_i, \frac{s^2 B_i}{24\pi^2}\right),\ \text{and } z_i \sim N\left(\bar{z}_i, \frac{s^2 B_i}{24\pi^2}\right), \tag{2}$$

where s scales the standard deviation.

Figure 5. An unperturbed depiction of 1QMPa is shown on the left, and a sample from a random model of the same protein chain is shown in the center ($s = 1$). The image on the right is an overlay of all nuclear magnetic resonance (NMR) renderings of 1NTR.



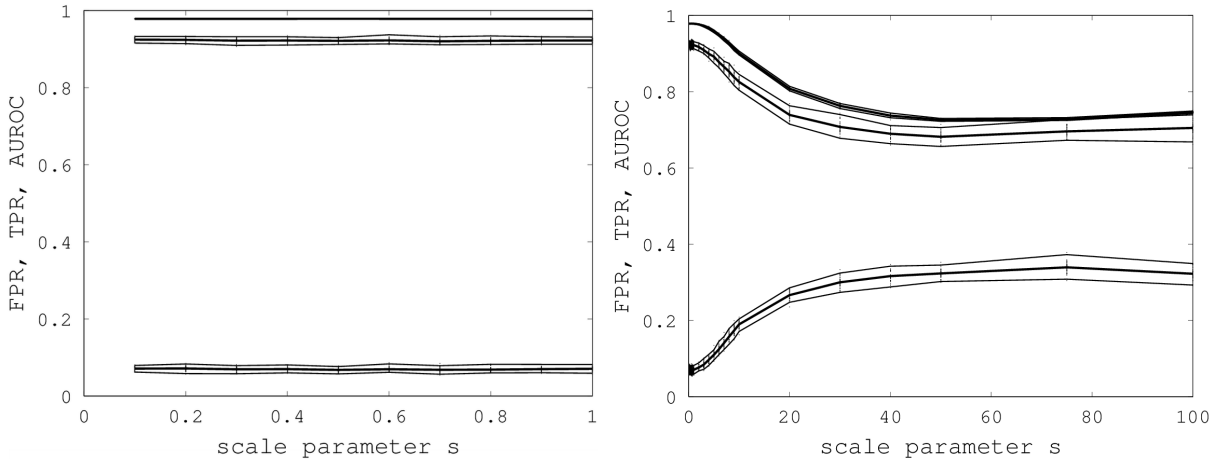
The reason for scaling the variances is that it provides a tool to adjust coordinate uncertainty, and hence, it can be used to assess an algorithm's sensitivity to coordinate uncertainty. A robust algorithm will remain effective for large values of s, whereas a less robust algorithm's efficacy will degrade for small values of s. For example, algorithms that depend on secondary structure assignments may be unduly sensitive to small coordinate variations. If an algorithm remains stable for s up to one, then the algorithm is robust with regards to the coordinate variations of the experiment.

Before continuing, we address an important criticism of using uncorrelated distributions for coordinate perturbations. Atomic coordinates are not uncorrelated, and for this reason, the uncorrelated assumption will over estimate coordinate variability. However, should an algorithm remain stable for large s, then we have some confidence that it will remain stable over the more limited perturbations that are actually possible. Hence, the test used here to assess experimental robustness is one-sided in that an algorithm may exhibit a loss of efficacy for small s but remain accurate over the range of actual variation. With only a B-factor available, the uncorrelated assumption leads to a reasonable test that is independent of a full simulation of each protein's dynamics. Such simulations are computationally costly and would negate our ability to harness the alignment efficiency of EIGAs to consider experimental robustness.

To illustrate the effect of randomizing the coordinates, we consider 1QMPa with each coordinate being distributed as in (2). An image of the static protein in terms of secondary structure is shown in Figure 5 on the left (images generated by PyMOL [44]). A random sample of the same protein with its coordinates being drawn from the normal distributions with $s = 1$ is depicted in the center. The difference in the two illustrates that secondary structures can be lost as the coordinates vary. We mention that after looking at several such comparisons, the atom (sphere) view commonly shows that the random perturbations maintain the general globular shape, even though secondary structures are lost. This observation suggests that an alignment algorithm that directly compares backbone atoms might be more stable with regards to coordinate uncertainty than an algorithm that depends on secondary structure.

We initially varied s from zero to one in steps of 0.1, and for each s, we generated 40 random samples of the Proteus300 dataset [45]. The reason for 40 samples was that this nicely yielded a 95% bootstrap confidence interval by removing the highest and lowest AUROC. We found that EIGAs' efficacy was essentially unaffected for this range of s. A graph of AUROC and the best TPR and FPR is shown on the left in Figure 6. Over all 400 samples, as s increased from 0.1 to one, the AUROC was always above 0.9783 and below 0.9785 and, hence, was nearly constant. The best TPR (FPR) was between 0.9201 (0.0678) and 0.9244 (0.0715). We conclude that EIGAs is robust with respect to coordinate uncertainty.

Figure 6. On the left, AUROC (top) and the best TPR (middle) and FPR (bottom) are shown as s ranges from 0.1 to one. On the right, AUROC (top) and the best TPR (middle) and FPR (bottom) as s ranges from 0.1 to 100. 95% bootstrap confidence intervals are shown in both graphs for all simulations.



The success of EIGAs for $s \leq 1$ led to the question of how much coordinate variability EIGAs could accommodate before AUROC degraded significantly. To answer this question, we let s range over $\{1, 2, \ldots, 10, 20, 30, 40, 50, 75, 100\}$, again using 40 samples for each s. The results are shown on the right in Figure 6. AUROC and the best TPR and FPR degraded as s increased from one to 20, but then, these metrics flattened as s increased to 100. Indeed, from 50 to 100, AUROC is nearly constant at 0.7332, and the best TPR and FPR are nearly constant at 0.6941 and 0.3284. This is a surprising result, as it shows that EIGAs can correctly match proteins within a family with about 70% accuracy, even with a 100-fold increase in coordinate variation.

7. Conclusions and Future Research

The immediate conclusion of our numerical work is that EIGAs is robust against variations that could alter the DP algorithm. Moreover, our adapted three-fold cross-validation suggests that EIGAs can be tuned on moderate-sized datasets and then applied to larger collections. A secondary conclusion is that the modern DP-based algorithms are likely to exhibit similar robustness, since DP can be stable over wide ranges of input data.

The results for large values of s with regard to coordinate uncertainty are not what we had expected. The B-factors are not uniform across the atoms, and we suspect that the smaller B-factors correlate with

the hydrophobic core, which is likely (much) more stable. The fact that EIGAs remains effective for large values of s suggests that it is aligning structures based on their most stable elements.

A near term research question is to test several of the other DP algorithms designed for fast alignments to verify whether or not they share EIGAs' robustness. If so, then a logical conclusion would be that DP is the reason why efficient algorithms are finding success for a variety of different similarity models.

Acknowledgments

The authors are thankful for the assistance of Dr. Eric Reyes for his suggestion on adapting a three-fold cross-validation test, to Dr. Joseph Eichholz for his programming aid and to Dr. Mark Brandt for his guidance on biochemistry. We also thank three anonymous referees for their comments on an earlier version.

Conflicts of Interest

The authors declare no conflict of interest.

References and Notes

1. Andonov, R.; Malod-Dognin, N.; Yanev, N. Maximum contact map overlap revisited. *J. Comput. Biol.* **2011**, *18*, 27–41.
2. Andonov, R.; Yanev, N.; Malod-Dognin, N. An Efficient Lagrangian Relaxation for the Contact Map Overlap Problem. In Proceedings of the 8th International Workshop on Algorithms in Bioinformatics, Karlsruhe, Germany, 15–19 September 2008; Springer-Verlag: Berlin/Heidelberg, Germany, 2008; pp. 162–173.
3. Hasegawa, H.; Holm, L. Advances and pitfalls of protein structural alignment. *Curr. Opin. Struct. Biol.* **2009**, *19*, 341–348.
4. Li, S.C.; Ng, Y.K. On protein structure alignment under distance constraint. *Theor. Comput. Sci.* **2011**, *412*, 4187–4199.
5. Menke, M.; Berger, B.; Cowen, L. Matt: Local flexibility aids protein multiple structure alignment. *PLoS Comput. Biol.* **2008**, *4*, e10.
6. Poleksic, A. Algorithms for optimal protein structure alignment. *Bioinformatics* **2009**, *25*, 2751–2756.
7. Prlic, A.; Bliven, S.; Rose, P.W.; Bluhm, W.F.; Bizon, C.; Godzik, A.; Bourne, P.E. Pre-calculated protein structure alignments at the RCSB PDB website. *Bioinformatics* **2010**, *26*, 2983–2985.
8. Shibberu, Y.; Holder, A. A spectral approach to protein structure alignment. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2011**, *8*, 867–875.
9. Bonnel, N.; Mareau, P. LNA: Fast Protein Classification Using A Laplacian Characterization of Tertiary Structure. Technical Report for IRISA: Vannes, France, 2012.
10. Kifer, I.; Nussinov, R.; Wolfson, H.J. GOSSIP: A method for fast and accurate global alignment of protein structure. *Bioinformatics* **2011**, *27*, 925–932.
11. Bhattacharya, S.; Bhattacharyya, C.; Chandra, N.R. Projections for fast protein structure retrieval. *BMC Bioinform.* **2006**, *7*(Suppl. 5), S5.

12. Lena, P.D.; Fariselli, P.; Margara, L.; Vassura, M.; Casadio, R. Fast overlapping of protein contact maps by alignment of eigenvectors. *Bioinformatics* **2010**, *26*, 2250–2258.
13. Liu, W.; Srivastava, A.; Zhang, J. A mathematical framework for protein structure comparison. *PLoS Comput. Biol.* **2011**, *7*, e1001075.
14. Lu, Z.; Zhao, Z.; Fu, B. Efficient protein alignment algorithm for protein search. *BMC Bioinform.* **2010**, *11*(Suppl. 1), S34.
15. Mavridis, L.; Ritchie, D.W. 3d-blast: 3d protein structure alignment, comparison, and classification using spherical polar fourier correlations. *Pac Symp. Biocomput.* **2010**, doi:10.1142/9789814295291_0030.
16. Mirceva, G.; Cingovska, I.; Dimov, Z.; Davcev, D. Efficient approaches for retrieving protein tertiary structures. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2012**, *9*, 1166–1179.
17. Novosd, T.; Snel, V.; Abraham, A.; Yang, J.Y. Searching protein 3-D structures for optimal structure alignment using intelligent algorithms and data structures. *IEEE Trans. Inf. Technol. Biomed.* **2010**, *14*, 1378–1386.
18. Poleksic, A. Optimal pairwise alignment of fixed protein structures in subquadratic time. *J. Bioinform. Comput. Biol.* **2011**, *9*, 367–382.
19. Shibuya, T.; Jansson, J.; Sadakane, K. Linear-time protein 3-D structure searching with insertions and deletions. *Algorithms Mol. Biol.* **2010**, *5*, 7.
20. Holm, L.; Sander, C. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.* **1993**, *233*, 123–138.
21. Shindyalov, I.N.; Bourne, P.E. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.* **1998**, *11*, 739–747.
22. Zhang, Y.; Skolnick, J. TM-align: A protein structure alignment algorithm based on the TM-score. *Nucleic Acids Res.* **2005**, *33*, 2302–2309.
23. Orengo, C.A.; Taylor, W.R. SSAP: Sequential structure alignment program for protein structure comparison. *Meth. Enzymol.* **1996**, *266*, 617–635.
24. Pang, B.; Zhao, N.; Becchi, M.; Korkin, D.; Shyu, C.R. Accelerating large-scale protein structure alignments with graphics processing units. *BMC Res. Notes* **2012**, *5*, 116.
25. Holm, L.; Kriinen, S.; Rosenstrm, P.; Schenkel, A. Searching protein structure databases with DaliLite v.3. *Bioinformatics* **2008**, *24*, 2780–2781.
26. Pascual-Garca, A.; Abia, D.; Ortiz, A.R.; Bastolla, U. Cross-over between discrete and continuous protein structure space: Insights into automatic classification and networks of protein structures. *PLoS Comput. Biol.* **2009**, *5*, e1000331.
27. Redfern, O.C.; Harrison, A.; Dallman, T.; Pearl, F.M.G.; Orengo, C.A. CATHEDRAL: A fast and effective algorithm to predict folds and domain boundaries from multidomain protein structures. *PLoS Comput. Biol.* **2007**, *3*, e232.
28. Budowski-Tal, I.; Nov, Y.; Kolodny, R. FragBag, an accurate representation of protein structure, retrieves structural neighbors from the entire PDB quickly and accurately. *Proc. Natl. Acad. Sci. USA* **2010**, *107*, 3481–3486.
29. Bellman, R. *Dynamic Programming*; Princeton University Press: Princeton, NJ, USA, 1957.

30. Needleman, S.B.; Wunsch, C.D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **1970**, *48*, 443–453.
31. Goonesekere, N.C.W.; Lee, B. Frequency of gaps observed in a structurally aligned protein pair database suggests a simple gap penalty function. *Nucleic Acids Res.* **2004**, *32*, 2838–2843.
32. Madhusudhan, M.S.; Marti-Renom, M.A.; Sanchez, R.; Sali, A. Variable gap penalty for protein sequence-structure alignment. *Protein Eng. Des. Sel.* **2006**, *19*, 129–133.
33. Chen, L.; Wu, L.Y.; Wang, Y.; Zhang, S.; Zhang, X.S. Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison. *BMC Struct. Biol.* **2006**, *6*, 18.
34. Salem, S.; Zaki, M.J.; Bystroff, C. FlexSnap: Flexible non-sequential protein structure alignment. *Algorithms Mol. Biol.* **2010**, *5*, 12.
35. Schmidt-Goenner, T.; Guerler, A.; Kolbeck, B.; Knapp, E.W. Circular permuted proteins in the universe of protein folds. *Proteins* **2010**, *78*, 1618–1630.
36. Poleksic, A. On complexity of protein structure alignment problem under distance constraint. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2012**, *9*, 511–516.
37. Shibberu, Y.; Holder, A.; Lutz, K. *Fast Protein Structure Alignment*; LNCS (LNBI). Springer-Verlag: Berlin/Heidelberg, Germany, 2010; Volume 6053, pp. 152–165.
38. Homepage of SWIG. Available online: www.swig.org (accessed on 22 October 2013).
39. Cock, P.J.A.; Antao, T.; Chang, J.T.; Chapman, B.A.; Cox, C.J.; Dalke, A.; Friedberg, I.; Hamelryck, T.; Kauff, F.; Wilczynski, B.; de Hoon, M.J.L. Biopython: Freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **2009**, *25*, 1422–1423.
40. Andreeva, A.; Howorth, D.; Chandonia, J.M.; Brenner, S.E.; Hubbard, T.J.P.; Chothia, C.; Murzin, A.G. Data growth and its impact on the SCOP database: New developments. *Nucleic Acids Res.* **2008**, *36*, D419–D425.
41. Andreeva, A.; Murzin, A.G. Structural classification of proteins and structural genomics: New insights into protein folding and evolution. *Acta Crystallogr. Sect. F Struct. Biol. Cryst. Commun.* **2010**, *66*, 1190–1197.
42. Conte, L.L.; Ailey, B.; Hubbard, T.J.; Brenner, S.E.; Murzin, A.G.; Chothia, C. SCOP: A structural classification of proteins database. *Nucleic Acids Res.* **2000**, *28*, 257–259.
43. Ritchie, D.W.; Ghoorah, A.W.; Mavridis, L.; Venkatraman, V. Fast protein structure alignment using Gaussian overlap scoring of backbone peptide fragment similarity. *Bioinformatics* **2012**, *28*, 3274–3281.
44. Delano, W.L. *The PyMOL Molecular Graphics System*; DeLano Scientific: San Carlos, CA, USA, 2002.
45. The chain d1fvqa, which is copper-transporting ATPase, has a single model and no B-factor. Hence, this chain was constant throughout the study.