An Investigation into the Robustness of Algorithms Designed for Efficient Protein Structure Alignment across Databases

Allen Holder, Jacqueline Simon, Jonathon Strauser, and Yosi Shibberu

Rose-Hulman Institute of Technology, Terre Haute, IN

Abstract. The recent literature on aligning protein structures indicates the emergence of several efficient algorithms to conduct database wide comparisons. 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 align the proteins. The algorithms have become efficient and accurate enough that it is becoming possible to study their robustness under perturbations of the data, say due to experimental error or due to protein dynamics. Our primary goal here is to promote the study of robustness as a way to assess an algorithm's stability, which will become increasingly important as the science grows to consider protein dynamics. We argue that the dynamic programming framework has a natural affinity to accommodate such studies, and we report on numerical tests to validate this observation.

Keywords: Protein Alignment, Structural Bioinformatics, Dynamic Programming

1 Introduction

The problem of aligning protein structures to infer functional similarity has a long and sustained literature, see [1, 2, 15, 18, 23, 27, 30, 34] as recent examples, and it has become a stalwart within computational biology. Within the last year there have been several publications that have promoted fast algorithms for use on database wide comparisons [1, 7, 16, 34] (see also the related work of [6, 17, 19, 20, 24, 22, 26, 29, 35]). Fast algorithms designed for database comparisons are not designed to optimize an alignment between two proteins per say, but rather, their intent is to efficiently discern if two proteins share a similar structure, which gives evidence that they may share a similar function. Since these algorithms are to be applied to large databases, they have a heightened emphasis on efficiency, only needing to be accurate enough to correctly identify biological classifications.

The speed with which the most recent algorithms can compare proteins supports a wealth of numerical work that would have been previously hamstrung by lengthy computations. Moreover, the current efficiency gives rise to the potential of studying an alignment algorithm as it accounts for a protein's dynamics, which has been overlooked to this point.

The algorithmic framework common to the most recent fast alignment methods is dynamic programming (DP), which is a polynomial-time algorithm and can be used for either global [25] or local [36] alignments. The optimality of an alignment depends on the penalties associated with introducing gaps, which means that the computed alignment is a function of the penalties. Selecting parameters that lead to a sound biological interpretation is nontrivial, and our first goal is to conduct a search of parameter space to infer the robustness of a DP based algorithm with regards to these parameters. If the volume of parameter space that achieves biological relevance is small, then we might conclude that tuning an algorithm for one dataset might not lead to success on another. However, if the volume of parameter space is large, then this gives confidence that the algorithm might work across dataset without individual tuning for each dataset.

Proteins are dynamic molecules that are vibrating, and while this fact is ignored by current alignment algorithms, it is noted in the protein database (pdb) files. There are two common methods for establishing a protein's structure, X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy. NMR experiments generate multiple descriptions of a protein in solution, and each description is a snapshot of the protein's dynamics as it vibrates through an aqueous medium. PDB files from NMR experiments include multiple 'snapshots' that are intended to help explain the protein's dynamics. The NMR technique is advantageous because a protein is in a natural environment, and hence the renderings hopefully represent actual conformations of the folded protein. The disadvantage is that only small proteins lend themselves to this type of experiment.

X-ray crystallography requires that the protein first be crystallized. This means that a solution must become saturated enough to create a crystalline structure of the protein, which 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 images are not of the protein in aqueous solutions, and hence the images are not actual snapshots of the protein in a natural setting. The dynamics of the protein are estimated, at least to some degree, by B-factors that are reported in the pdb files.

We propose a test of an alignment algorithm's ability to correctly identify family relationships as the dynamics of the proteins are considered, i.e how robust is an algorithm as proteins vibrate. To the authors' awareness no such computational evaluation has been proposed, which is not surprising since earlier alignment algorithms required lengthy computations. However, an experimental interpretation of the dynamics is included in the pdb files, and the speed of the new algorithms now supports the repeated database wide comparisons needed to undertake a simulation of the dynamics. Our goal here is meager as compared to the substantial numerical work that can, and should, be undertaken since we only test one of the recent algorithms designed for database wide comparisons. A thorough presentation comparing the many fast algorithms recently proposed along with their abilities to account for dynamics would exceed this publication's

limitations. Instead, we establish the computational test and vet one algorithm's ability to correctly identify protein families as the database is simulated to account for structural variations due to the inherit protein dynamics.

The two primary outcomes of this article are 1) that DP based alignment algorithms are likely to be stable over a wide range of parameters and 2) the initiation of new computational tests that measure how well an alignment algorithm performs as protein dynamics are considered. The next section discusses DP, both generally and specifically, as it is being used to align protein structures. This discussion points to why DP may have a natural affinity for handling protein dynamics. Section 3 reports on a numerical test showing that the DP based algorithm EIGAs does indeed exhibit stability over parameter space. Section 4 simulates the protein's dynamics and investigates the robustness of EIGAs. Again, our intent is not to promote EIGAs particularly but to introduce new computational experiments that can be used to assess an algorithm's ability to accommodate protein dynamics.

2 Dynamic Programming and Protein Alignment

Over the last 18 months four algorithms have been proposed for efficient protein structure alignment across databases, those being 1) Eigen-decomposition Alignment with the spectrum (EIGAs) [33, 34], 2) A_purva [1], 3) GlObal Structure Superposition of Proteins (GOSSIP) [16], and 4) Laplacian Norm Alignment (LNA) [7] (see [8] for a succinct review of the first three). The algorithms differ in how they measure the similarity between the residues of different proteins, but they all use DP to make alignments. Notationally, if we let S_{ij} be a value that represents the similarity between residue i of one protein and residue j of another, then each algorithm scores this relationship differently.

The algorithms A_purva, EIGAs and LNA compute S_{ij} from a measure of the intra-similarity of the C_α atoms within each protein. Let d_{ij} be the distance between residues i and j in a single protein. Then each of A_purva, EIGAs and LNA uses d_{ij} differently to measure the relationship between residues i and j. The different measures are respectively,

$$C_{ij} = \begin{cases} 1, & \text{if } \eta < d_{ij} \leq \kappa \\ 0, & \text{otherwise,} \end{cases} \qquad C'_{ij} = \begin{cases} 1 - d_{ij}/\kappa, & \text{if } d_{ij} \leq \kappa \\ 0, & \text{otherwise,} \end{cases}$$

$$\text{and} \quad C''_{ij} = \begin{cases} e^{-(d_{ij}/\sigma)^2} & \text{if } i \neq j \\ 0, & \text{otherwise.} \end{cases}$$

The parameter κ defines the largest distance at which the first two models permit a nonzero relationship between residues i and j. In C residues are either in contact, with $C_{ij} = 1$, or they are not, with $C_{ij} = 0$. The lower bound η is nonnegative, and C_{ii} is always 0. In A_purva the value of η also ensures that the contact between neighboring residues is zero, i.e. $C_{i,i+1} = 0$. In C' pairs of residues receive a linearly scaled value of d_{ij} as long as the distance does not exceed κ. Note that $C'_{ii} = 1$ since $d_{ii} = 0$. This property implies that C' is

positive definite for appropriately selected κ [33], and hence, the smooth-contact matrix C' permits a Cholesky factorization, i.e. there is a matrix R so that $C' = R^T R$. In C'' the distance is an exponential decay of $(d_{ij}/\sigma)^2$. Each of C_{ij}, C'_{ij} and $C''ij$ are small if d_{ij} is large.

In EIGAs the similarity between residues i and j of different proteins is $S_{ij} = 2|\lambda_i - \lambda_j|/|\lambda_i + \lambda_j|$, where λ_i and λ_j are the eigenvalues associated with the two residues. Specifically, residue i is assigned λ_k if $|R_{ki}| = \max_t |R_{ti}|$, see [34] for an algebraic and geometric description. In LNA $S_{ij} = e^{-\nu\tau_{ij}}$, where ν a parameter and τ_{ij} is a measure of the difference between the C_α coordinates after being transformed with a Laplacian [7]. In A_purva, as well as other contact map overlap (CMO) approaches, pairs of possible matches are scored instead of single pairs, and the tuple (i, j, k, l) is assigned a value of 1 if $C_{ij} = 1$ in the first protein and $C_{kl} = 1$ in the second. Otherwise the tuple receives a value of 0. This assignment leads to the well studied CMO model, and what the authors of A_purva show is that the Lagrangian relaxation of the CMO problem can be solved by coupling two DP algorithms. A description of how the scoring matrices S are created from the Lagrangian relaxation is found in [1]. In GOSSIP residues are assigned an eight dimensional characteristic vector, and the score S_{ij} is determined by comparing these two vectors [16].

Dynamic programming was first introduced by Bellman [5], and it was first used to align biological sequences in [25]. A common recursion that defines an alignment is

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

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.

If the penalties for opening and continuing a gap are different, then an affine penalty is being used. Several computational studies in biology indicate a benefit to affine gaping [14, 21]. Affine models, however, require an additional parameter over their non-affine counterparts, which complicates tuning DP for the application at hand. Our numerical work below uses an affine model, and we let ρ_o and ρ_c be the penalties for initiating and continuing a gap, respectively.

GOSSIP, EIGAs and LNA apply DP directly to their respective S matrices, and hence, these algorithms align the backbones as if they were linear sequences. Such methods are sometimes called 1-dimensional [15] since they align 3D proteins by pairing residues along the 1D backbones. A_purva's use of DP is more complicated, and the double application of DP that solves the Lagrangian relaxation of CMO must be repeated as the Lagrange variables are updated. LNA has two adaptations of the basic recursion stated above depending on whether a local or global alignment is desired [7].

Independent of the particular application of DP, we note that if the coordinates of a protein's atoms are modeled dynamically, say as probability distribu-

tions, then sampling the distributions adjusts the coordinates of the protein's atoms. Altering the coordinates adjusts the S matrices, albeit differently for each algorithm. However, each step of the DP recursion is stable for sufficiently small changes in either S_{ij} or the gaping penalty since V_{ij} depends on the comparison of three values that are likely to be distinct. So for each pair of proteins the alignment from DP is likely to be stable for sufficiently small changes in either the parameters (κ, ρ_o, ρ_c) or changes in the coordinates of the residues.

The observation that DP is stable for sufficiently small perturbations suggests two favorable characteristics with regard to database applications:

1. That for any scoring matrix S, DP will likely return the same optimal alignment over a sufficiently small range of (affine) gap penalties.
2. DP might be less sensitive to data perturbations than other alignment methods, and it is reasonable that DP should be able to efficiently calculate alignments that are stable as S is altered to account for experimental error and protein dynamics.

These observations suggests that tuning the gaping penalties over a large dataset should result in parameters that correctly assess the value of pairing residue i with residue j. These tuned gaping values applied to another dataset maintain the same physical interpretation, and hence, even if the scores are not perfect, there is a reason to believe that the alignments on the new dataset should be reasonable. Moreover, as the coordinates of the C_α atoms are adjusted to account for a protein's dynamics, DP should remain somewhat stable. So algorithms like EIGAs, LNA, GOSSIP and A_purva should be robust over sufficiently small coordinate perturbations due to experimental error or small vibrations that explain protein dynamics.

In the next two sections we report on experiments that begin to explain the robustness of DP for database applications. The next section explains a search of gaping penalties for EIGAs, and as suggested by the first observation, DP is largely stable over a wide range of parameter values. Section 4 studies how EIGAs behaves as protein dynamics are simulated, and these results show that DP can indeed support studies of classification as proteins are modeled dynamically.

While the observations above depict DP favorably, we would be remiss to ignore some of its downsides. All uses of DP for database applications only consider sequential alignments. Nonsequential alignments are important in some, and indeed possibly many, cases [10, 31, 32]. Adapting DP to accommodate for nonsequential 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 nonsequential alignments should also be investigated [28].

3 Parameter Robustness

We tested the robustness of EIGAs over the large Proteus300 dataset [1] to query it's stability as parameters were adjusted. The Proteus300 dataset consists

of 300 proteins belonging to 30 known families as identified by the Structural Classification of Proteins (SCOP) database [3, 4, 12], which is a classification of proteins with a known structure. The EIGAs code is implemented through a combination of Matlab, C, and python code. Biopython [11] is used to parse the pdb files, and Matlab is used to factor the C' matrices and to construct the scoring matrices. The DP implementation was written in C and linked to Matlab as a mex file. The earlier numerical work in [34] used a non-affine gaping model and the DP algorithm had been coded in Matlab. The previous numerical work showed that as κ ranged from 6 to 20 Å, EIGAs could correctly discern the majority of the SCOP family classifications, the best results being for κ values between 8 and 16 Å. However, even the best outcome misclassified 11 proteins. This numerical work is improved here in two ways. First, the large speedup from the C implementation of DP permits a far wider range of parameter investigations. Second, the earlier model used a simple non-affine gaping penalty of 1 along with the un-normalized scoring value of $S_{ij} = |\lambda_i - \lambda_j|$. As mentioned above, we normalize this score by the average of the eigenvalues in the numerical work here, i.e. $S_{ij} = 2|\lambda_i - \lambda_j|/|\lambda_i + \lambda_j|$, which ensures that the score is always between 0 and 2.

Each alignment was scored by an extension of the CMO score [34] called the smooth-contact score,

$$sCMO_{AB} = \frac{2 \sum\limits_{i>j+2} [C']^A_{ij}[C']^B_{ij}}{\sum\limits_{i>j+2} [C']^A_{ij} + \sum\limits_{i>j+2} [C']^B_{ij}},$$

where $[C']^A$ and $[C']^B$ are the C' matrices of the two proteins being compared. There are three parameters that can be changed to affect the DP alignment. These are the cutoff value, κ, the gap opening penalty ρ_o, and the gap continuation penalty ρ_c. We incremented κ from 6 to 20 Å with a step size of 1 Å and incremented each of the gap penalties from 0 to 1 with a step size of 0.1. For each possible triple (κ, ρ_o, ρ_c), we randomly selected 10 of the 30 families in Proteus300 and used EIGAs to find the nearest neighbor of each protein in the sampled dataset, nearest as measured by the $sCMO$ score. Since we sampled on families and not individual protein chains, we knew that the nearest neighbor of each protein in any sample was indeed included in the sample. We repeated this test 10 times for each triple (κ, ρ_o, ρ_c). The intent of this sampling procedure was to evaluate how well tuning on a smaller dataset might lead to gaping penalties that would work on larger datasets.

A heatmap was generated to show the regions of parameter space that gave perfect, or near perfect, alignments. Regions in which all sampled proteins were correctly classified with their SCOP families are depicted in Figure 1 on the left. The image on the right shows the larger regions for which EIGAs correctly identified the SCOP family at least 90% of the time after sampling, i.e the DP based EIGAs correctly identified the SCOP family for at least 900 of the 1000 protein chains over the repeated sampling. From these figures we see that a DP

based alignment algorithm can have large regions of stability over the space of parameters.

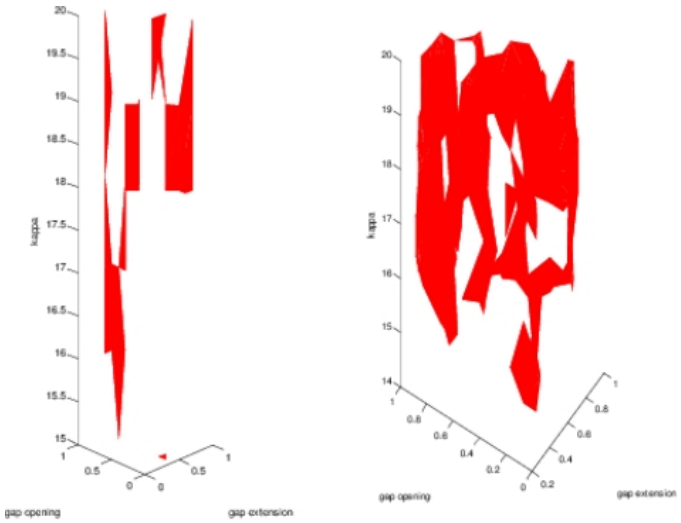

Fig. 1. The figure on the left shows the regions of parameter space over which the DP based EIGAs correctly classified the SCOP family over all samples. The figure on the right shows the regions of parameter space over which EIGAs was 90% correct over all samples. The axis on the lower left is ρ_o; on the lower right is ρ_c, and the vertical axis is κ.

The graph on the left shows that if ρ_o was between 0.7 and 0.9 and if ρ_c was approximately 0.4, then EIGAs' DP algorithm correctly identified the SCOP families for a wide range of κ values. With this information we searched for a κ value whose cross section contained a large region containing these gap penalties. The largest such region had a κ value of 17Å in the graph on the right. The center of this region had $\rho_o = 0.9$, $\rho_c = 0.4$ and $\kappa = 17$. Testing these values on the entire Proteus300 dataset resulted in perfect accuracy, which gave us some confidence that tuning DP on multiple smaller datasets could lead to gap penalties that were appropriate for larger datasets.

We only tested gap penalties as large as 1, although the value of S_{ij} could have been as large as 2. Large values of S_{ij} indicate a dissimilarity between the residues, and pairing residues with large S_{ij} values does not agree with the intent of the alignment process. Figure 2 depicts a histogram of all possible S_{ij} values over all possible $44,850$ pairs of the Proteus300 dataset. From this graph we have that the preponderance of pairwise scores was below 1, which indicated that our parameter search covered the majority of alignment possibilities. However, in the

future we will complete the lengthy computation needed to extend the search to all possible gaping penalties.[1]

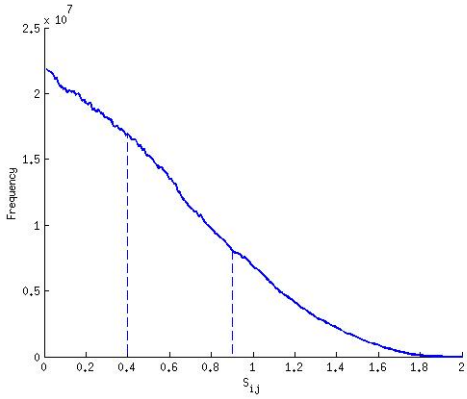

Fig. 2. A histogram of all possible S_{ij} values over all $44,850$ pairs of the Proteus300 dataset. Our parameter search indicated a large region of stability around $(\kappa, \rho_o, \rho_c) = (17, 0.9, 0.4)$. The affine gap penalties are noted by dashed lines.

A few comments about the computing environment and how our results compare to others in the literature. We parallelized the all-against-all pairwise comparisons to use 8 cores on a linux server with an Intel Core i7-860 2.8 GHz chipset. The time to complete all 44,850 pairwise comparisons of the entire Proteus300 dataset was 137 seconds, which correctly classified each of the 300 proteins with the parameter values suggested above. The Eig_7 algorithm in [17] can complete the Proteus300 dataset in about 6 hours and can correctly classify all but 3 of the proteins. A_purva [1] is able to finish the dataset in about 13.5 hours and can correctly classifies all 300 proteins. The numerical results in [7] show that GOSSIP can complete the Proteus300 dataset in as little as 20 minutes and 44 seconds and can correctly classifies 290 of the 300 proteins. LNA has the fastest reported results on GPUs and is capable of finishing the Proteus300 dataset in under a second with perfect classification [7].

4 Robustness and Protein Dynamics

Although the results of the previous section are encouraging from a computational perspective, maybe the most exciting outcome of the recent DP based algorithms is that their speed and accuracy might allow proteins to be studied as the dynamic entities that they are. If so, then our computational models

[1] Completing the search as depicted required about a week of computation.

should be able to heighten the biological significance of numerical results. In this section we use the DP based alignment algorithm EIGAs to assess how such an algorithm might behave as the Skolnick40 [9] dataset is simulated to account for protein dynamics.

To illustrate the effect of randomizing the coordinates, consider 1QMPa with each coordinate being normally distributed – the mean being the reported coordinate value and the variance being a multiple of the B-factor as described below. An image of the static protein in terms of secondary structure is shown in Figure 3 on the left (images generated by pymol [13]). A random sample of the same protein with its coordinates being drawn from normal distributions as described below is depicted in the center. The difference in the two illustrates that secondary structures can be lost as the coordinates vary to account for their random position. 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 can be lost. These observations suggest that an alignment algorithm that directly compares backbone atoms might be more stable with regards to protein dynamics than an algorithm that is dependent on secondary structure.

An X-ray crystallography experiment produces a B-factor, often called a Debye-Waller factor, that is a scaled version of the mean squared displacement of the atom. If $\bar{r}_i$ is the expected value of the position of the i-th residue, which is estimated by averaging the positions of the atom in the crystalline structure, and r_i is the random position of the atom, then the B-factor of the i-th residue is

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

where E is the expected value. So, if we assume that the coordinates of the C_α atoms are uncorrelated normals with the same variance, then the standard deviation of each coordinate is $\sqrt{B_i/24\pi^2}$.

To assess how well EIGAs could correctly identify the SCOP family classification as the proteins vibrated, we assumed that the coordinates of the C_α atoms from the X-ray crystallography experiments were uncorrelated normals:

$$x_i \sim N(\bar{x}_i, sB_i/24\pi^2), \; y_i \sim N(\bar{y}_i, sB_i/24\pi^2), \text{ and } z_i \sim N(\bar{z}_i, sB_i/24\pi^2),$$

where s was a scale factor that permitted an investigation into how EIGAs behaved as the variance increased. The reason for the scale factor is that more stable algorithms should be able to correctly identify the SCOP family classifications for larger values of s. If the family classification is correct up to a value of $s = 1$, then the algorithm is stable up to the dynamics recognized by the X-ray crystallography experiment.

We tested how well EIGAs could identify SCOP families on a simulated dataset. Simulating the entire dataset permitted a higher degree of perturbation than randomizing a single protein and then comparing it to a dataset of static proteins, and while we believe that the latter test should be part of a broader numerical investigation, we began by considering the dynamics of all proteins at once. To do so we simulated all protein chains in the Skolnick40 dataset, a

smaller dataset of 40 proteins spread over 5 families. We initially varied s from 0 to 1 in small step sizes, and somewhat to our surprise, we found that EIGAs correctly identified the correct family classification even after numerous samples. Our curiosity then turned to the question of how robust EIGAs could be as protein coordinates varied beyond their stated variance. The results described below vary s from 0 to 100 with a step size of 0.25. For each value of s we generated 40 sample data sets, and on each we ran EIGAs to compare all 780 pairwise comparisons. Each protein was then assigned the SCOP family of its nearest neighbor as measured by the sCMO score. The affine gap penalties were $\rho_o = 0.9$ and $\rho_c = 0.4$ as suggested from the work in Section 3. The cutoff parameter was $\kappa = 17$Å. The reason for 40 samples was that this number nicely allowed a 95% bootstrap confidence interval by removing the highest and lowest percentage of correctly identified families.

Since NMR coordinates do not have a simple interpretation as random variables, we randomly selected one of the NMR models uniformly for each NMR file. Since the different NMR models are intended to explain the protein's dynamics, this seemed to be an appropriate way to include the NMR files. The right most image in Figure 3 depicts all models of 1NTR superimposed on each other , which gives a sense of the variation among different NMR models.

Fig. 3. An unperturbed depiction of 1QMPa is depicted on the left, and a sample from a random model of the same protein chain is shown in the center. The image on the right is all NMR renderings of 1NTR superimposed on each other.

There were three reasons we used the smaller Skolnick40 dataset. First, Skolnick40 is a widely tested dataset that is commonly used as new algorithms are developed. Since this was the first study of dynamics, we thought that using Skolnick40 was reasonable. Second, since Skolnick40 is disjoint from Proteus300, testing on Skolnick40 allowed us to test the gap penalties that had been tuned on Proteus300. Third, testing on Proteus300 would have required (much) longer computations and would have exceeded our ability to complete the numerical work for this paper.

The results from our numerical work are depicted in Figure 4, which plots the percent of correctly identified families against the scale factor. The first scale factor for which EIGAs did not correctly identify all families was 1.75, which

means that EIGAs can account for dynamic perturbations of 1.75 times greater than the experimental estimate on variability as specified by the B-factors. The largest scale factor for which all families were correctly identified was 13.25. From Figure 4 we see that the average behavior of EIGAs degrades but appears to level off at about 80%. The lowest average was 78.7%. The variability increased until the scale factor reached about 75, but beyond this value the random ability to predict families appears to have stabilized with a predictive ability of about $80 \pm 15\%$.

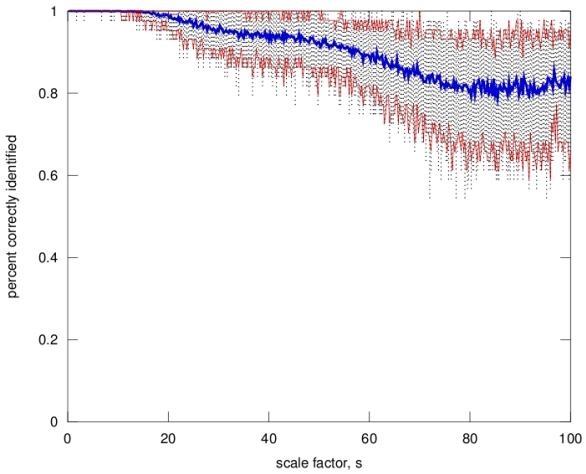

Fig. 4. The percentage of correctly identified families versus the scale factor of the variance induced by the B-factor. The gray dots show the variability observed over the entire sample set. The central curve is the mean for each scale factor, and the upper and lower curves form a 95% bootstrap confidence interval.

Figure 4 was not what the authors had expected, and the results of this experiment point to many questions. First, we don't have a substantive answer as to how EIGAs can correctly identify 80% of the SCOP families on average as the coordinates are scrambled 100-fold beyond their experimental variation. The B-factors are not uniform across the atoms, and we suspect that the smaller B-factors correlate with the hydrophobic core, which means that the protein's core is (much) more stable. We have confirmed this to some degree by studying structures with perturbed coordinates. As one would expect, it appears that most of the dynamics occurs on the surface of the protein. This means the portion of the eigen-decomposition of C' that is related to the hydrophobic core is probably stable under coordinate perturbation, and if EIGAs is aligning residues based on these eigenvalues, then this could reasonably explain the behavior.

Second, we know that assuming x, y and z to be uncorrelated normals is not the best model for protein dynamics. A protein's backbone geometry permits

only certain angles and distances between atoms, and these restrictions are not included here. Our sentiment was that if EIGAs could correctly identify families beyond the geometric limitations imposed by physics, then the algorithm should be stable with those limitation in place. We will work toward a more realistic model of the dynamics in the future, but as of our computational work here, it is clear that EIGAs is stable beyond experimental estimates.

Third, we don't know if our results are due to a well-tuned DP algorithm, our eigen-decomposition model, or just a fluke of the smaller dataset. We are preparing to conduct similar tests with GOSSIP, A_purva, and LNA to see if our results are representative of a DP based algorithm. If so, then DP and it's tuning are important. If not, then how we model similarity between proteins is probably more important than DP itself. One explanation is that the eigen-decomposition is somewhat stable due the symmetry of our random coordinate perturbations. As shown in [34], the eigenvectors have the interpretation that they are the vectors most in contact with the embedding of the residues. So if the residues are equally likely to move in any direction, then the eigenvectors may, to some degree, average away random residue perturbations. If this is the case, then our results could be the by-product of the way EIGAs is measuring the similarity between residues. We also need to conduct similar tests on larger datasets like Proteus300. Such a task is nontrivial computationally. Generating the data for Figure 4 required about 2.5 days on a modern workstation, and extending this computation from 780 to 44,850 pairwise comparisons per sample is a daunting challenge.

5 Conclusion

The work of this article initiates the numerical work that is becoming possible with the advent of efficient and accurate algorithms to align protein structures across databases. We have reviewed the most recent algorithms in this category and have provided evidence that their DP underpinnings might provide them an inherit robustness under either parametric adjustments or data perturbations. The former is important since it points to the fact that tuning DP on one dataset should lead to parameters that work on others. The latter is important because it permits proteins to be compared dynamically. Continued computational research is needed to compare different algorithms and to gain an understanding of an algorithm's robustness.

References

1. Rumen Andonov, Nol Malod-Dognin, and Nicola Yanev. Maximum contact map overlap revisited. *J Comput Biol*, 18(1):27–41, 2011.
2. Rumen Andonov, Nicola Yanev, and Noël Malod-Dognin. An efficient lagrangian relaxation for the contact map overlap problem. In *WABI '08: Proceedings of the 8th international workshop on Algorithms in Bioinformatics*, pages 162–173. Springer-Verlag, 2008.

3. Antonina Andreeva, Dave Howorth, John-Marc Chandonia, Steven E Brenner, Tim J P Hubbard, Cyrus Chothia, and Alexey G Murzin. Data growth and its impact on the scop database: new developments. *Nucleic Acids Res*, 36(Database issue):D419–D425, 2008.
4. Antonina Andreeva and Alexey G Murzin. Structural classification of proteins and structural genomics: new insights into protein folding and evolution. *Acta Crystallogr Sect F Struct Biol Cryst Commun*, 66(Pt 10):1190–1197, 2010.
5. R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.
6. Sourangshu Bhattacharya, Chiranjib Bhattacharyya, and Nagasuma R Chandra. Projections for fast protein structure retrieval. *BMC Bioinformatics*, 7 Suppl 5:S5, 2006.
7. N. Bonnel and P.F. Mareau. Lna: Fast protein classification using a laplacian characterization of tertiary structure. Technical Report hal-00639663, IRISA-UBS, Université de Bretagne Sud, France, 2001.
8. M. Brandt, A. Holder, and Y. Shibberu. Fundamentals of protein structure alignment. Work in progress.
9. Alberto Caprara, Robert Carr, Sorin Istrail, Giuseppe Lancia, and Brian Walenz. 1001 optimal pdb structure alignments: integer programming methods for finding the maximum contact map overlap. *J Comput Biol*, 11(1):27–52, 2004.
10. Luonan Chen, Ling-Yun Wu, Yong Wang, Shihua Zhang, and Xiang-Sun Zhang. Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison. *BMC Struct Biol*, 6:18, 2006.
11. Peter J A Cock, Tiago Antao, Jeffrey T Chang, Brad A Chapman, Cymon J Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J L de Hoon. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 2009.
12. L. Lo Conte, B. Ailey, T. J. Hubbard, S. E. Brenner, A. G. Murzin, and C. Chothia. Scop: a structural classification of proteins database. *Nucleic Acids Res*, 28(1):257–259, 2000.
13. W. L. Delano. The pymol molecular graphics system., 2002.
14. Nalin C W Goonesekere and Byungkook Lee. Frequency of gaps observed in a structurally aligned protein pair database suggests a simple gap penalty function. *Nucleic Acids Res*, 32(9):2838–2843, 2004.
15. Hitomi Hasegawa and Liisa Holm. Advances and pitfalls of protein structural alignment. *Curr Opin Struct Biol*, 19(3):341–348, 2009.
16. I. Kifer, R. Nussinov, and H. J. Wolfson. Gossip: A method for fast and accurate global alignment of protein structure. *Bioinformatics*, 27:925–932, 2011.
17. Pietro Di Lena, Piero Fariselli, Luciano Margara, Marco Vassura, and Rita Casadio. Fast overlapping of protein contact maps by alignment of eigenvectors. *Bioinformatics*, 26(18):2250–2258, 2010.
18. Shuai Cheng Li and Yen Kaow Ng. On protein structure alignment under distance constraint. *Theoretical Computer Science*, 412(32):4187 – 4199, 2011. Algorithms and Computation.
19. Wei Liu, Anuj Srivastava, and Jinfeng Zhang. A mathematical framework for protein structure comparison. *PLoS Comput Biol*, 7(2):e1001075, 2011.
20. Zaixin Lu, Zhiyu Zhao, and Bin Fu. Efficient protein alignment algorithm for protein search. *BMC Bioinformatics*, 11 Suppl 1:S34, 2010.

21. M. S. Madhusudhan, Marc A Marti-Renom, Roberto Sanchez, and Andrej Sali. Variable gap penalty for protein sequence-structure alignment. *Protein Eng Des Sel*, 19(3):129–133, 2006.
22. Lazaros Mavridis and David W Ritchie. 3d-blast: 3d protein structure alignment, comparison, and classification using spherical polar fourier correlations. *Pac Symp Biocomput*, pages 281–292, 2010.
23. Matthew Menke, Bonnie Berger, and Lenore Cowen. Matt: local flexibility aids protein multiple structure alignment. *PLoS Comput Biol*, 4(1):e10, 2008.
24. Georgina Mirceva, Ivana Cingovska, Zoran Dimov, and Danco Davcev. Efficient approaches for retrieving protein tertiary structures. *IEEE/ACM Trans Comput Biol Bioinform*, 2011.
25. S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, 1970.
26. Tom Novosd, Vclav Snel, Ajith Abraham, and Jack Y Yang. Searching protein 3-d structures for optimal structure alignment using intelligent algorithms and data structures. *IEEE Trans Inf Technol Biomed*, 14(6):1378–1386, 2010.
27. Aleksandar Poleksic. Algorithms for optimal protein structure alignment. *Bioinformatics*, 25(21):2751–2756, 2009.
28. Aleksandar Poleksic. On complexity of protein structure alignment problem under distance constraint. *IEEE/ACM Trans Comput Biol Bioinform*, 2011.
29. Aleksandar Poleksic. Optimal pairwise alignment of fixed protein structures in subquadratic time. *J Bioinform Comput Biol*, 9(3):367–382, 2011.
30. Andreas Prlic, Spencer Bliven, Peter W Rose, Wolfgang F Bluhm, Chris Bizon, Adam Godzik, and Philip E Bourne. Pre-calculated protein structure alignments at the rcsb pdb website. *Bioinformatics*, 26(23):2983–2985, 2010.
31. Saeed Salem, Mohammed J Zaki, and Chris Bystroff. Flexsnap: flexible non-sequential protein structure alignment. *Algorithms Mol Biol*, 5:12, 2010.
32. Tobias Schmidt-Goenner, Aysam Guerler, Bjoern Kolbeck, and Ernst Walter Knapp. Circular permuted proteins in the universe of protein folds. *Proteins*, 78(7):1618–1630, 2010.
33. Y. Shibberu, A. Holder, and K. Lutz. Fast protein structure alignment. In *LNCS (LNBI)*, volume 6053, pages 152–165. Springer-Verlag, 2010.
34. Yosi Shibberu and Allen Holder. A spectral approach to protein structure alignment. *IEEE/ACM Trans Comput Biol Bioinform*, 2011.
35. Tetsuo Shibuya, Jesper Jansson, and Kunihiko Sadakane. Linear-time protein 3-d structure searching with insertions and deletions. *Algorithms Mol Biol*, 5:7, 2010.
36. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197, 1981.