IMPROVEMENT IN PROTEIN SEQUENCE-STRUCTURE ALIGNMENT USING INSERTION/DELETION FREQUENCY ARRAYS

Kyle Ellrott, Jun-tao Guo, Victor Olman, Ying Xu.
Computational Systems Biology Laboratory,
Department of Biochemistry and Molecular Biology and Institute of Bioinformatics,
The University of Georgia, Athens, Georgia 30602

As a protein evolves, not every part of the amino acid sequence has an equal probability of being deleted or for allowing insertions, because not every amino acid plays an equally important role in maintaining the protein structure. However the most prevalent models in fold recognition methods treat every amino acid deletion and insertion as equally probable events. We have analyzed the alignment patterns for homologous and analogous sequences to determine patterns of insertion and deletions, and used that information to determine the statistics of insertions and deletions for different amino acids of a target sequence. We define these patterns as Insertion/Deletion (Indel) Frequency Arrays (IFA). By applying IFA to the protein threading problem, we have been able to improve the alignment accuracy, especially for proteins with low sequence identity. Contact: xyn@bmb.uga.edu

1. INTRODUCTION

Protein threading, a technique for sequence-structure alignment, has played a key role in predicting protein structures in the past decade. Most of the details in a threading model deal with how well an amino acid from a target sequence is aligned to a particular residue position on a known protein structure. For example, the energy functions used in our threading program PROSPECT include mutation, singleton, secondary structure match, and two-body interaction energies[9]. These sets of energy function primarily concentrate on the positive space of the alignment, i.e. rewarding amino acid alignment. Deletion penalties, on the other hand, are a set of terms that describe how to penalize an alignment when gaps are introduced. There are two primary changes during protein evolution: mutation and insertion/deletion. A mutation event is the result of changing one amino acid to another and is evaluated by mutation energy matrices, such as PAM[3] and Blosum [7].

Another event in protein evolution is the insertion and deletion of amino acids. These evolutionary changes are evaluated with gap penalty models in protein threading algorithms. While several gap penalty models have been proposed, the most widely used model for gap penalty is the simple affine model [17]. In this model there is a large penalty for opening a gap, or starting a deletion, and a smaller constant penalty for continuing that insertion/deletion. This can be viewed as a simple linear function, $G = W_{open} + W_{const} * len$, where len is the length of the gap, W_{open} is the penalty for opening a gap, and every residue deleted is penalized by a constant penalty W_{const}.

This simple linear function can be easily implemented in a dynamic programming based alignment program such as the Smith-Waterman method, with a running time of $O(NM)$ where N is the length of the target sequence, and M is the length of the structural template. In addition to this linear penalty model, there are more sophisticated methods that have been used. These typically attempt to formulate the penalty as non-linear functions [16, 6], or use monotonic functions to avoid over-penalizing large gaps [13]. However, these non-linear gap functions cannot be optimized using traditional Smith-Waterman, and require more advanced algorithms for sequence-structure alignment optimization [4, 11].

Nonetheless, within the framework of the Smith-Waterman algorithm, it is possible to use a nonlinear gap function, if the function is only dependent on local sequence/structure alignments. The penalty of a gap can be dependent on the probability of an amino acid being deleted or being inserted. Given these conditions a set of local optimal decisions can still be aggregated to achieve the global optimality. Therefore, dynamic programming can still be used.

To find useful statistical information about deletion and insertion probabilities, we use a technique similar to the one for generating Position Specific Score Matrices (PSSM) [1]. PSSMs have had a significant impact on secondary structure prediction and protein fold recognition [21]. A PSSM is generated by finding homologous sequences in a non-

redundant (NR) sequence database and aligning those sequences. The amino acid mutation patterns are used to create residue specific replacement scores.

The patterns of insertions and deletions can be studied in a similar way. Using statistical analysis of alignments from a 'PsiBlast' search against the NR database, we can construct penalty functions that are based on insertion/deletion patterns specific to a protein family and the different portions of the sequence. These scores are not simply based on a global constant. For every residue, the percentage of times that it is deleted, or has an insertion before or after it can be measured against a known sequence database. We call this information the Indel Frequency Arrays (IFA). It should be pointed out that this type of energy, unlike some of other previously mentioned gap models, is only dependent on local sequence alignment and thus can be run in the same computational time as Smith Waterman algorithm. There has been similar position specific gap penalties suggested previously, such as the work by Lesk at al. [10]. However, their work was based on different scoring values for differently assigned secondary structure values, and was not specific to protein families.

2. METHODS

Alignment Strategy

We use our threading program, PROSPECT, as an example, in which the optimal alignment is calculated by finding an alignment A with the total alignment score E_{tot} defined as:

$$\begin{aligned} E_{tot} = min_t(&W_{mut}E_{mut}(A) \\ &+ W_{singleton}E_{singleton}(A) \\ &+ W_{secstruct}E_{secstruct}(A) \\ &+ W_{open}E_{open}(A) \\ &+ W_{const}E_{const}(A)) \end{aligned} \tag{1}$$

where E_{mut} is the mutation energy, $E_{singleton}$ is the Singleton energy, $E_{secstruct}$ is secondary structure match energy; E_{open} and E_{const} represent the two aspects of the affine gap function, the gap opening penalty, and the constant deletion for each residue removed; the set of Ws represent the weight parameters.

Optimal alignments can be found with dynamic programming by finding iterative solution of the values for $S_{i,j}$, with i and j both going from zero to the length of the target (l) and template (m) respectively. $S_{i,j}$ is sub sequence alignment of the target residues from $0-i$ and the template residues from $0-j$. Thus the total alignment is expressed as $E_{tot} = S_{l,m}$. The value of $S_{l,m}$ can be iteratively calculated with the formula:

$$S_{i,j} = min \begin{cases} S_{i-1,j-1} + E_{i,j} & \text{Match} \\ S_{i-1,j} + INS(i,j) & \text{Insertion}, (i,j) \Rightarrow I \\ S_{i,j-1} + DEL(i,j) & \text{Deletion}, (i,j) \Rightarrow D \end{cases} \tag{2}$$

The energy $E_{i,j}$ for aligning a target position i to a template position j is defined as:

$$\begin{aligned} E_{i,j} = &W_{mut}E_{mut}(i,j) \\ &+ W_{singleton}E_{singleton}(i,j) \\ &+ W_{secstruct}E_{secstruct}(i,j) \end{aligned} \tag{3}$$

In the original model, the INS and DEL values where calculated as such:

$$INS(i,j) = \begin{cases} \text{If } (i-1,j) \in I & W_{const}E_{const} \\ \text{If } (i-1,j) \notin I & W_{open}E_{open} \\ & + W_{const}E_{const} \end{cases} \tag{4}$$

$$DEL(i,j) = \begin{cases} \text{If } (i,j-1) \in D & W_{const}E_{const} \\ \text{If } (i,j-1) \notin D & W_{open}E_{open} \\ & + W_{const}E_{const} \end{cases} \tag{5}$$

$$S_{0,0} = 0, \tag{6}$$

where I is the set of insertion operations, and D is the set of deletion operations. These equations refer to operations on the template, i.e. an insertion operation is an insertion on the template.

New Gap Energy Model

Deletion is the inverse operation of insertion. A deletion in the target is equivalent to an insertion to the template, and visa-versa. However, how these operations are handled is different. The deletion occurs at a specific point, while an insertion occurs in between two residues.

Our study has shown that deletion and insertion probabilities are not equally distributed across the entire sequence, and are unlikely to be similar for different protein sequences. A sample distribution

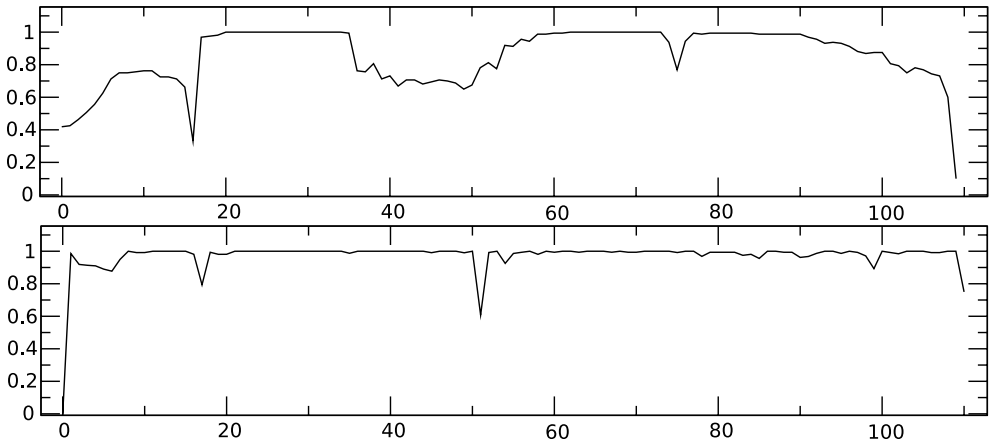

Fig. 1. IFA information associated with the SCOP identifier 'd1qhoa2'. The top graph is the IFA information for deleting any of the residues. The bottom graph represents the IFA information for having an insertion before a given residue. A value of zero means that the indel can occur without penalty.

can be seen in Figure 1. The probability of deleting a residue is not directly related to the probability of inserting a residue immediately before or after it, suggesting a simple deletion operation could actually encompass four different energies, insertion and deletion for both sequences being aligned.

As a result, the penalty feature previously described as E_{const} which was a constant penalty for every insertion/deletion can be replaced with a set of four features: E^q_{ins}, E^q_{del}, E^t_{ins}, and E^t_{del}, where t stands for template, and q stands for query or target.

In order to maintain the W coefficients each of the new penalty values needs to be scaled between 0 and 1, and multiplied by W_{const}.

Under this model we redefine the $INS(i,j)$ and $DEL(i,j)$ functions as 7 and 8.

Calculating Indel Profiles

The insertion/deletion profiles used to create the IFA are derived from alignments using 'PsiBlast' against the NR database. Deletion energies are determined by observing the number of times a residue is deleted, or a residue being inserted before it.

In order to easily access indel information, we translate the standard two line text alignment returned back by 'PsiBlast' into a number array as shown in Figure 2. In this format array a represents the indices of the aligned subject sequence for each amino acid in the target sequence. If $a[i] = j$ then the ith amino acid in the target sequence is aligned to the jth amino acid in the template. Positions that are not aligned to any residue are represented by '-1'. After the above conversion, finding the insertions/deletions becomes a matter of referring to one array, rather than parsing two text arrays. To find the deletions, one simply scans the a array looking for the '-1' entries, which represent the non-aligned residues. To find the insertions, one looks at all the non '-1' entries in the array. For the ith residue that is followed by the next non '-1' residue j, if $a[i] + 1 \neq a[j]$ then there is a gap. If the first non '-1' entry is not '1', there is a pre-sequence insertion. Similarly, if the last non '-1' entry is not aligned to the last residue of the subject, there is a post-sequence insertion. This information is then summed for each individual residue position, and devided by the number of aligned sequences. This provides the percentage of times a residue is deleteted or allows an insertion. These arrays of percentages are referred to as B_{ins} and B_{del}. They are then formulated as energy functions as such: $E_{ins}(i) = 1 - B_{ins}(i)$ and $E_{del}(i) = 1 - B_{del}(i)$.

```
    Target:-ABC--DEFGI----
  Template:AABCAA--FGIZZZZ

TargetAlign:2,3,4,-1,-1,7,8,9
TemplateAlign:-1,1,2,3,-1,-1,6,7,8,-1,-1,-1,-1
```

Fig. 2. This example shows the conversion from the sequence based model used to represent alignments, typically as outputted by Blast. The bottom shows the same alignment in an easier to use hash table format. Each position represents the number of the position of the aligned residue in the opposite sequence. Deletions are represented as a -1.

$$INS(i,j) = \begin{cases} \text{If } (i-1,j) \in I & W_{const}(E^t_{ins}(i-1) + E^q_{del}(j)) \\ \text{If } (i-1,j) \notin I & W_{open}E_{open} + W_{const}E^t_{ins}(i-1) + E^q_{del}(j)) \end{cases} \quad (7)$$

$$DEL(i,j) = \begin{cases} \text{If } (i,j-1) \in D & W_{const}(E^t_{del}(i) + E^q_{ins}(j-1)) \\ \text{If } (i,j-1) \notin D & W_{open}E_{open} + W_{const}(E^t_{del}(i) + E^q_{ins}(j-1)) \end{cases} \quad (8)$$

Training and Testing

We use two methods in order to evaluate the alignment performance inprovements that the IFA method provies. First, we compared the alignment results with the output from FAST [22], a structural comparison program. We chose FAST because of its efficient and accurate performance. FAST can correctly align 96% of the residue pairs in aligned regions of the 1033 protein alignments in the HOMESTRAD database[12, 22]. As a common practice alignment is considered to be correct if the residue was aligned within 4 residues of the FAST-based structure-structure alignment position[18]. The reported percentage accuracy is the percentage of residues placed within 4 residues of the correct postion out of the total possible residue placements. The next method of evaluation is the MAMMOTH program[15]. MAMMOTH determines the statistical significance of the backbone structure created by predictive tools against the actual backbone structure of the target. We report the $-ln(E)$ score, for which a value greater then 4 is statistically significant.

Our training set is comprised of 300 SCOP [14] domain entries from the ASTRAL 25 list [2], in which no entries would have higher than 25% sequence identity. Based on the results from FAST, the average sequence identity for the aligned pairs of this dataset is 9.5%. Using the SCOP identifiers, we then compiled a list of all proteins that occurred in the same fold, super families and families. To find the optimal set of weights for each of the gap penalty permutation, we have applied 10 cycles of the Violated Inequality Minimization (VIM) [23] method of optimization. The training set is used to find a set of optimal weights for our original threading approach which uses traditional affine gap penalty. The same set of weights was used in the variable deletion model.

For the testing we used a set comprised of 724 proteins also derived from ASTRAL 25 that did not overlap with the original training set. We used the same SCOP table to determine relationship, how ever this time, relationship were filterd by the FAST SN score. The SN score determines significance of the structural alignment created by FAST. Pairs with scores lower then 2 were removed so that bad alignments would not create noise when analyzing the performance of threading results against the structural alignments. This left a testing set comprised of 3058 pair relationships. This set was made of 344 family, 1265 super family and 1449 fold level relationships.

Statistical Analysis

We describe our statistical model related to comparison of two methods as an experiment with multinomial distribution having 3 possible outcomes: Method 1 (IFA method) is better than Method 2 (original method) (probability P_{+1}), the two Methods are equal in their power (probability P_0), and Method 2 is better than Method 1 (probability P_{-1}), $P_0 + P_{-1} + P_{+1} = 1$. Our goal is to check hypothesis $H_0 : P_{+1} = P_{-1}$ against the alternative $H_1 : P_{+1} > P_{-1}$. For hypothesis checking we use Pearson's χ^2 test, i.e. $\chi^2 = \frac{(K_{+1}/N - P_{+1})^2}{P^2_{+1}} + \frac{(K_{-1}/N - P_{-1})^2}{P^2_{-1}} + \frac{(K_0/N - P_0)^2}{P^2_0}$, where N is the number of comparisons, K_{+1} the number of times Method 1 worked better, and K_{-1} the number of times Method 2 worked better. If H_0 is true then $P_{+1} = P_{-1} = 0.5*(1-P_0)$, and replacing P_0 with its maximum likelihood estimator K_0/N, we get $\chi^2 = \frac{(K_{+1}/N - p)^2}{p^2} + \frac{(K_{-1}/N - p)^2}{p^2}, p = 0.5 * (1 - P_0)$. The asymptotic distribution of the χ^2 in our case is a chi-square distribution with one degree of freedom.

3. RESULTS

Comparison with Constant Gap Penalty

The overall average improvement for alignment accuracy is shown at different alignment levels in Table 1. The more distant two proteins are evolutionarily, the

Table 1. A comparison using different gap function.

	Fast			MAMMOTH			
SCOP Alignment	Original	IFA	Increase	Original	IFA	Increase	Set Size
Fold	42.6	46.2	8.5%	11.5	13.2	14.8%	1449
SuperFamily	55.3	57.1	3.3%	13.9	15.2	9.4%	1265
Family	70.6	71.4	1.1%	14.5	15.5	6.9%	344
	Fast			MAMMOTH			
Sequence Identity	Original	IFA	Increase	Original	IFA	Increase	Set Size
0% – 5%	36.2	38.3	5.8%	13.4	15.3	14.2%	909
5% – 10%	51.8	55.2	6.6%	12.5	14.0	12%	1527
10% – 15%	68.4	70.4	2.9%	12.6	13.5	7.1%	487
15% – 20%	76.4	80.1	4.8%	11.8	12.5	5.9%	107
20% – 100%	91.4	93.9	2.7%	13.4	13.7	2.2%	28

bigger improvement of the IFA method. At the fold level, alignment accuracy increase from 50% to 55%. Once two proteins are in the same family, the amount of improvement decreases to 2.5%. A similar trend is observed if we seperate protein pairs by their percentage of aligned amino acids. The lower the identity, the larger improvement that the IFA model provides. This trend can also be seen in Figure 3. The top section of the figure represents binned averages across different levels of sequence identity. The bottom section represents a segmented linear regression.

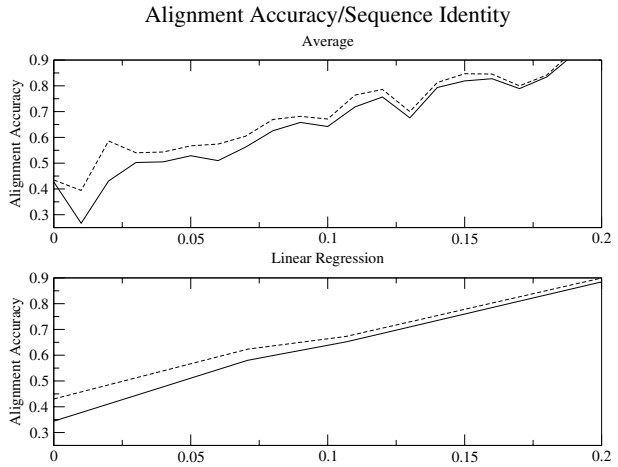

Fig. 3. The binned averages and multi-segment linear regressions of alignment accuracy for the two methods. The dotted line represent the IFA model. As the sequence identity of the aligned structures decreases, the contribution of the IFA model become more valueable.

A specific example for alignment improvement shown in Figure 4 demonstrates the potential benefits of this information source. Both proteins were classified as 'winged helix' DNA-binding domains. The original model over compensated for C-terminal deletions. This caused the second helix to be aligned to the location where the first helix should be aligned. This cascaded into a series of mid sequence deletions that smeared two helices together. Our model increases the alignment accuracy from 26.5% to 73%.

In order to show that these improvements in alignment accuracy were not the product of random statistical fluctuations we have also analyzed the comparative performance of the two methods on the same alignment pairs. We counted the number of times the new model has led to an improvement in alignment accuracy, shown in Table 2 . We have also calculated the statistical significance of these numbers. Using the statistical analysis described in the Methods section, we can calculate the p-value of the hypothesis that improvements are random. For the testing set with FAST based alignemnts, we get the values $K_{+1} = 1464$, $K_0 = 882$, and $K_{-1} = 712$, which for the first statistical testing method leads to the p-value=3.55×10^{-111}. This shows that the difference in performance of two methods can not be explained by pure chance, indicating the superiority of Method 1, our new IFA method.

Improving Fold Recognition

In many studies fold recognition techniques are trained to achieve the optimal alignment accuracy. This is based on the assumption that the closer two proteins are in terms of their fold families, the more amino acids that are likely to be correctly aligned. Therefore, the better the predicted alignment accuracy, the better the fold recognition method is likely to be. When training machine learning techniques

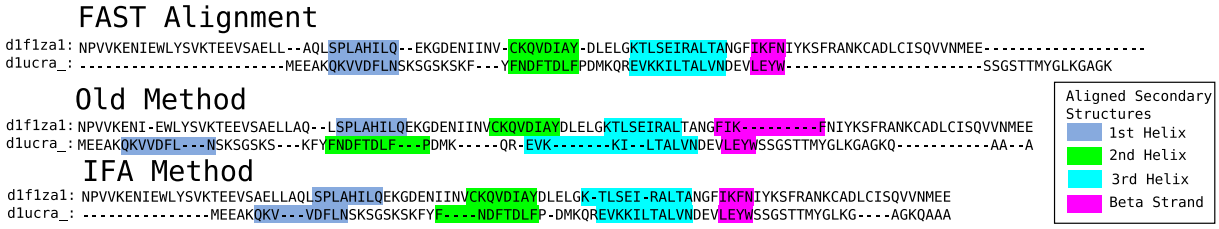

Fig. 4. This example is for the alignment between SCOP domains 'd1f1za1' and 'd1ucra_'. Each block represents an assigned secondary structure element.

Table 2. The side by side comparison. The number of times each method had a better score, and the statistical significance of that ratio. The p-value is the probability that the improvement is caused by chance, the lower the better.

Level	IFA	Original	Tie	Test 1	Test 2	Test 3
FAST						
All	1464	712	882	3.55×10^{-111}	1.58×10^{-113}	6.22×10^{-26}
Fold	667	362	420	5.00×10^{-38}	1.29×10^{-38}	7.62×10^{-10}
SuperFamily	647	281	337	2.97×10^{-61}	5.60×10^{-63}	5.63×10^{-15}
Family	150	69	125	3.02×10^{-12}	1.5×10^{-12}	4.61×10^{-04}
MAMMOTH						
All	2383	282	393	0	0	1.03×10^{-157}
Fold	1170	138	141	0	0	2.57×10^{-78}
SuperFamily	973	118	174	6.72×10^{-289}	0	1.05×10^{-64}
Family	240	26	78	2.94×10^{-73}	1.29×10^{-84}	1.29×10^{-17}

Table 3. The Correlation coefficients of the gap energies, as applied to the two threading method results.

Feature	Constant Deletion	IFA
E_{open}	-0.29	-0.29
E_{const}	-0.18	-0.11
E^t_{del}	-0.12	-0.15
E^t_{ins}	-0.10	-0.14
E^q_{del}	-0.11	-0.13
E^q_{ins}	-0.10	-0.16
E_{var}	-0.17	-0.27

previous research has used a vector to represent a set of features from an alignment, such as the values from the different energy terms, and trained them for fold recognition using the alignment accuracy as a measure. Previous research have used machine learning techniques such as neural networks [20], SVMs [18], and gradient boosting [8] based functions. For these techniques, the more correlated a given feature is to the alignment accuracy, the easier it will be to train the regression function. Therefore, we can predict the benefit to fold recognition a feature will have by measuring its correlation with the alignment accuracy. We test our new energy functions by comparing the correlation coefficients of: E_{const}, E^t_{del}, E^q_{del}, E^t_{ins}, E^q_{ins} with the alignment accuracy.

We compared the correlation coefficients for each of the energies in two different threading sets in Table 3. The combined energy is $E_{var} = E^t_{del} + E^t_{ins} + E^q_{del} + E^q_{ins}$. The variable deletion energies, once combined, are very well correlated to alignment accuracy, indicating a good ability to distinguish correct alignments. The ability increases even more once they are used to optimize the alignment, as in the variable deletion threading set. First, the threading set comprised of alignments created with E_{const} and next, the set of alignments optimized using the set of variable gap penalties. As we can see in Table 3, individually each of the separate variable deletion penalties is not as correlated as E_{const}. However, once they are summed together, and used to optimize the alignment, their correlation increases greatly. This makes sense, because each of the variable deletion penalties is only 1/4th of the total deletion energy. The increase in correlation from -0.18, using the original model for alignment and fold recognition, to -0.27, for using the new model, should correspond to a greater ability to correctly differentiate correct fold from incorrect fold.

4. DISCUSSION

We have demonstrated our new deletion model in the context of protein threading. This new energy seems to work best in the context of distantly-related threading models. The variable gap penalty by Madhusudhan et al [11] concentrated on the performance improvements of variable gap penalties for protein alignments with sequence identity spanning the range of 20-40%. Our profile-based variable deletion energy has its best improvements in the low homology range, from 2-15% sequence identity, the so called 'twilight zone' where both fold recognition and threading alignment accuracies are in desperate needs for improvements. As seen in Figure 3, the lower the sequence idenity, the more the IFA can improve the accuracy of sequence alignment. But at higher sequence identity levels, where the variable deletion penalty starts to lose some of its advantage, it does not cause an increase in false positive. So it can be used safely, regardless of the level of homology. We have shown that our energy fits within the Smith-Waterman alignment framework, but it is also theortically possible to incorporate it into the algorithmic methods suggested by Madhusudhan et al[11].

Not only did Indel Frequency Arrays improve the alignment accuracy, we also have shown that it is a statistically significant improvement. Further study, with the application of SVMs, neural networks, or gradient boosting methods will be needed to see if the increased alignment accuracy correlation coefficient we detected translates to better fold recognition.

5. CONCLUSION

We have shown there is large amount of information inherent in the insertions and deletions during protein evolution. This information can be determined by analyzing sequence alignments with homologous sequences. Once applied, this technique can improve protein threading alignment accuracy. We have shown that this information can be applied to the Smith-Waterman sequence alignment algorithm without added complexity. These energies can also be added to more complex methods, such as integer programming [19, 5].

ACKNOWLEDGMENTS

The work is, in part, supported by National Science Foundation (DBI-0354771/ITR-IIS-0407204/CCF-0621700), and a Distinguished Cancer Scholar grant from the Georgia Cancer Coalition.

References

1. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, 1997.
2. J. Chandonia, G. Hon, N. Walker, L. L. Conte, P. Koehl, M. Levitt, and B. SE. The astral compendium in 2004. *Nucleic Acids Research*, 32:D189–D192, 2004.
3. M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model for evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 5:345–352, 1978.
4. T. G. Dewey. A sequence alignment algorithm with an arbitrary gap penalty function. *Journal of Computational Biology.*, 8(2):177–190, 2001.
5. K. Ellrott, J. tao Guo, V. Olman, and Y. Xu. A generalized threading model using integer programming with secondary structure element deletion. *Genome Informatics*, 17(2), 2006.
6. N. C. Goonesekere and B. Lee. Frequency of gaps observed in a structurally aligned protein pair database suggests a simple gap penalty function. *Nucleic Acids Research*, 32(9):2838–2843, 2004.
7. S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, 89(22):10915–9, 1992.
8. F. Jiao, J. Xu, L. Yu, and D. Schuurmans. Protein fold recognition using the gradient boost algorithm. In *Computational Systems Bioinformatics Conference*, 2006.
9. D. Kim, D. Xu, J. T. Guo, K. Ellrott, and Y. Xu. PROSPECT II: protein structure prediction program for genome-scale applications. *Protein Eng*, 16(9):641–50, 2003.
10. A. M. Lesk, M. Levitt, and C. Chothia. Alignment of the amino acid sequences of distantly related proteins using variable gap penalties. *Protein Engineering*, 1(1):77–78, 1986.
11. M. Madhusudhan, M. A. Marti-Renom, R. Sanchez, and A. Sali. Variable gap penalty for protein sequence-structure alignment. *Protein Engineering, Design, and Selection*, 19(3):129–133, 2006.
12. K. Mizuguchi, C. Deane, T. Blundell, and J. Overington. Homstrad: a database of protein structure alignments for homologous families. *Protein Science*, 7:2469–2471, 1998.
13. R. Mott. Local sequence alignments with monotonic gap penalties. *Bioinformatics*, 15(6):455–462, 1999.
14. A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol*, 247(4):536–40, 1995.

15. A. Ortiz, C. Strauss, and O. Olmea. Mammoth (matching molecular models obtained from theory): an automated method for model comparison. *Protein Sci.*, 11(11):2606–21, 2002.
16. B. Qian and R. A. Goldstein. Distribution of indel lengths. *Proteins: Structure, Function, and Genetics*, 45:102–104, 2001.
17. J. G. Reich, H. Drabsch, and A. Däumler. On the statistical assessment of similarities in dna sequences. *Nucleic Acids Res*, 12(13):5529–5543, July 1984.
18. J. Xu. Fold recognition by predicted alignment accuracy. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(2):157–165, 2005.
19. J. Xu, M. Li, D. Kim, and Y. Xu. RAPTOR: optimal protein threading by linear programming. *J. Bioinform. Comput. Biol.*, 1(1):95–117, 2003.
20. Y. Xu, D. Xu, and V. Olman. A practical method for interpretation of threading scores: An application of neural network. *Statistica Sinica*, 12:159–177, 2002.
21. G. Yona and M. Levitt. Within the twilight zone: a sensitive profile-profile comparison tool based on information theory. *J Mol Biol*, 315(5):1257–75, 2002.
22. J. Zhu and Z. Weng. FAST: A novel protein structure alignment algorithm. *Proteins*, 58:618–627, 2005.
23. A. Zien, R. Zimmer, and T. Lengauer. A simple iterative approach to parameter optimization. In *Recomb Proceedings 2000*, pages 318–327, 2000.