LU TP 03-17

A Fuzzy Matching Approach to Multiple Structure Alignment of Proteins

Henrik Haraldsson and Mattias Ohlsson*

Complex Systems Division, Department of Theoretical Physics,
University of Lund, Sölvegatan 14A, SE-223 62 Lund, Sweden
`http://www.thep.lu.se/tf2/complex/`

Submitted to *Bioinformatics*

Abstract

Motivation: Comparing protein structures is important for understanding the relationships between sequence, structure and function. With the increasing number of experimentally determined protein structures, an automated algorithm that can perform structure alignment of many proteins is therefore highly advantageous.
Results: We present a novel approach for multiple structure alignment of proteins based on fuzzy pairwise alignments of each protein to a virtual consensus chain. These alignments are alternated with translations and rotations of the proteins onto the consensus structure, and with updating each consensus atom by moving it to the middle of the protein atoms aligned to it. The pairwise alignments use mean-field annealing optimization of fuzzy alignment variables, based on a cost expressed in terms of distances between aligned atoms and of gaps. No initialization in terms of all-to-all protein alignments is needed, and the only information required is the 3-D coordinates of the C_α atoms. The CPU consumption is modest, and scales approximately linearly with the number of proteins to align. Our approach is tested against a set of protein families from the HOMSTRAD database, and against a multiple structure alignment algorithm based on Monte Carlo techniques, with good results.

Key words: protein structure alignment; multiple structure comparison; fuzzy assignment; mean field annealing

*{henrikh,mattias}@thep.lu.se

1 Introduction

Comparative analysis of protein structures is a subject of utmost relevance. It enables the study of functional relationships between proteins and is very important for homology and threading methods in structure prediction. Multiple structure alignment of proteins is needed in order to group proteins into families, which enables a subsequent analysis of evolutionary issues. It is also important in order to build a consensus structure that encapsulates common re-occurring substructures among the structures. However, multiple structure alignment is not an easy task.

There exists many methods for pairwise structure alignment (see e.g. Gerstein and Levitt, 1996; Holm and Sander, 1993; Shindyalov and Bourne, 1998; Szustakowski and Weng, 2000). Few of these have been further developed for the alignment of multiple structures. The ones that exist fall into two broad groups. Methods in the first group perform all *pairwise* structure alignments and define the structure most similar to the others. Multiple structure alignment is then achieved by starting with an initial alignment of all structures to this selected one. Methods in the second group do not rely on all-to-all pairwise alignments as a starting point, instead they try to solve the multiple alignment problem in a more direct fashion.

Methods that fall within the first group can be found in the work of Gerstein and Levitt (1996) and Guda *et al.* (2001). The latter uses Monte Carlo optimization techniques to refine the initial alignment found by pairwise structure alignment using the Combinatorial Extension algorithm (Shindyalov and Bourne, 1998). Orengo and Taylor (1996) use double dynamic programming in their SSAPM algorithm, where they form an initial consensus structure from the two protein structures found to match each other the closest. Then they tentatively align all remaining structures to this consensus, select the one with the closest match and merge it into the consensus. This matching and merging is repeated until all protein structures have been merged into the consensus structure. The COMPARER algorithm developed by Šali and Blundell (1990) also performs an all-to-all pairwise alignment of all proteins as a first step in their multiple alignment procedure.

A method that does not rely on an initial all-to-all pairwise alignment is used in Leibowitz *et al.* (2001). This algorithm finds geometric substructures common for all structures to be aligned. Structural alignment of these cores then induces alignments of the full molecules. The starting point for their algorithm is only the 3-D coordinates for the C_α-atoms defining the structures to be aligned.

In this paper we have developed an approach for multiple structure alignment, that falls within the second group, with two main ingredients: (i) the construction of a *virtual consensus chain* that each individual protein is aligned to, and (ii) the use of an efficient fuzzy pairwise structure alignment method. The alignment is carried out in an iterative procedure. In each iteration step, the proteins are rotated and translated towards the consensus chain, guided by the fuzzy pairwise alignment method. Also, in each step a new consensus chain is computed from the alignments of the individual proteins.

The method, which requires no initial alignment from e.g. multiple sequence alignment, uses only 3-D coordinates of the C_α-atoms along the backbone and optionally secondary structure information for each amino acid. The proposed method, which is very general, has some appealing properties:

- **Scaling.** The method works by aligning each of the proteins to a common consensus chain and no initial alignment taken from an all-to-all pairwise alignment is needed. Hence, the CPU time needed grows only (approximately) linearly with the number of structures to align.

- **Generality of formulation.** Almost arbitrary additional constraints are easily incorporated into the formalism including, for example, different functional forms of gap penalties and sequence matching preferences.

The proposed method was tested on structures taken from the HOMSTRAD database (Mizuguchi *et al.*, 1998) of protein structure alignments for homologous families. Furthermore the results of our method were compared to those produced by the Monte Carlo method of Guda *et al.* (2001). The selected datasets were chosen to contain sets of proteins of various lengths, different structural classes and different average percentages of identities. For most cases our algorithm produced alignments in good agreement with the HOMSTRAD database and comparable to results obtained by the algorithm of Guda *et al.* (2001).

2 Methods

2.1 Multiple Alignment procedure

Consider K proteins that are to be structurally aligned to each other. In our approach this will be accomplished by aligning each of the K structures to a common *consensus* chain. Thus, the multiple alignment problem has been reduced to the construction of the common consensus chain and to perform K pairwise structure alignments. These two steps are not performed independently of each other; rather, we use an iterative 3-step procedure as follows:

- Fuzzy pairwise structure alignment of each of the K chains to the common consensus chain.
- Weighted rigid body rotations and translations for each of the K chains.
- Calculation of a new consensus chain.

The first two steps are based on the work by Blankenbecler *et al.* (2003) and will only be reviewed briefly.

In what follows we denote by $\mathbf{x}_i^{(k)}$ $(i = 1, ..., N_k)$ the atom coordinates of protein k, with length N_k. The phrase "atom" is here used in a generic sense – it could represent individual atoms but also groups of atoms. In this paper it will mean C_α-atoms along the backbone. Let $\mathbf{y}_j$ $(j = 1, ..., M)$ denote the atoms (hypothetical C_α positions) of the consensus chain (of length M). We use a square distance metric between the atoms in the proteins and atoms in the consensus chain,

$$d_{i,j}^{(k)} = |\mathbf{x}_i^{(k)} - \mathbf{y}_j|^2 \tag{1}$$

but the formalism is not confined to this choice.

2.2 Weighted rigid body rotations and translations

The objective of the rigid body rotation and translation of the proteins is to minimize the distance between the matched atoms for each of the K proteins and the consensus chain. Let $\tilde{\mathbf{x}}_i^{(k)}$ be the coordinates of the translated and rotated protein k, i.e., $\tilde{\mathbf{x}}_i^{(k)} = \mathbf{a}^{(k)} + \mathcal{R}^{(k)}\mathbf{x}_i^{(k)}$. Based on the fuzzy assignment matrix $\mathcal{W}^{(k)}$ we determine the translation vector $\mathbf{a}^{(k)}$ and the rotation matrix $\mathcal{R}^{(k)}$, by minimizing the following chain error function,

$$E_{\text{chain}}^{(k)} = \sum_{i=1}^{N_k}\sum_{j=1}^{M} \mathcal{W}_{i,j}^{(k)} \left(\mathbf{a}^{(k)} + \mathcal{R}^{(k)}\mathbf{x}_i^{(k)} - \mathbf{y}_j\right)^2 \ . \tag{2}$$

Matched atoms for protein k and the consensus chain are given by the elements $\mathcal{W}_{i,j}^{(k)}$, of the fuzzy assignment matrix. These elements are not restricted to be integers, leading to an interpretation of fuzzy assignments between atoms. The minimization of $E_{\text{chain}}^{(k)}$ can be solved exactly (Neumann, 1937), with closed-form expressions for $\mathcal{R}^{(k)}$ and $\mathbf{a}^{(k)}$. It should be noted that this solution is rotationally invariant (independent of $\mathcal{R}^{(k)}$) for the special case when the atoms in the two chains match each other with the same weight, i.e. when $\mathcal{W}_{i,j}^{(k)}$ is constant for all i and j.

2.3 Fuzzy pairwise structure alignment

Our approach for multiple structure alignment is an iterative procedure involving pairwise alignments to a common consensus chain. Each pairwise alignment is obtained using the fuzzy alignment method developed by Blankenbecler *et al.* (2003). This method is similar to the dynamical programming method for global sequence alignment (Needleman and Wunsch, 1970), but with two important differences. First, instead of using a score between aligned atoms, a cost formulation is used. This cost, which depends on the distances between the atoms and on the number of gaps and their locations, is changing throughout the alignment procedure. Second, in the original Needleman–Wunsch algorithm (Needleman and Wunsch, 1970) an optimal alignment path is calculated, whereas fuzzy alignment paths are computed in the method by Blankenbecler *et al.* (2003). The rest of this section gives a small review of the fuzzy alignment method. For readability we have dropped the (k) superscript that indicates one of the k proteins to align.

The structure alignment of two proteins is carried out in an annealing procedure, controlled by a *temperature* parameter T. Let $\mathcal{D}_{i,j}$ denote a fuzzy generalization of the optimal alignment cost at node (i, j) in the *dot-matrix* (cf. (Blankenbecler *et al.*, 2003)), used to represent all possible alignments of two proteins. We have

$$\mathcal{D}_{i,j} = \sum_{l=1}^{3} v_{i,j;\,l}\, \widetilde{\mathcal{D}}_{i,j;\,l} \ , \tag{3}$$

where $\widetilde{\mathcal{D}}_{i,j;\,l}$ is the corresponding generalized fuzzy alignment cost if the alignment path is forced to pass through the preceeding node given by l. In the Needleman–Wunsch algorithm only the optimal direction l is used, which implies that $v_{i,j;\,l}$ are integers and $\sum_l v_{i,j;\,l} = 1$. This restriction is relaxed in the fuzzy alignment method where $v_{i,j;\,l} \in [0, 1]$, but still sum up to unity. These so called *mean field* variables are calculated according to,

$$v_{i,j;\,l} = \frac{e^{-\widetilde{\mathcal{D}}_{i,j;\,l}/T}}{\sum_{l'} e^{-\widetilde{\mathcal{D}}_{i,j;\,l'}/T}} \ . \tag{4}$$

The generalized fuzzy alignment costs $\widetilde{\mathcal{D}}_{i,j;\,l}$ are calculated using the following recursive relation,

$$\begin{aligned}
\widetilde{\mathcal{D}}_{i,j;\,1} &= \mathcal{D}_{i,j-1} + \lambda_j^{(2)}(1 - v_{i,j-1;\,1}) + \lambda_{\text{ext}} v_{i,j-1;\,1} \ , \\
\widetilde{\mathcal{D}}_{i,j;\,2} &= \mathcal{D}_{i-1,j-1} + d_{i,j} \ , \\
\widetilde{\mathcal{D}}_{i,j;\,3} &= \mathcal{D}_{i-1,j} + \lambda_i^{(1)}(1 - v_{i-1,j;\,3}) + \lambda_{\text{ext}} v_{i-1,j;\,3} \ .
\end{aligned} \tag{5}$$

Here, $\lambda_a^{(n)}$ is the penalty for matching atom a in chain n to a gap and λ_{ext} is the gap extension penalty.

At each iteration in the annealing procedure of lowering T, a fuzzy assignment matrix $\mathcal{W}_{i,j}$ is calculated as

$$\mathcal{W}_{i,j} = P_{i,j} v_{i,j;\,2} \ , \tag{6}$$

where $P_{i,j}$ is the probability that node (i, j) is part of the optimal path and $v_{i,j;\ 2}$ is the probability that atoms i and j are locally matched. $P_{i,j}$ can be calculated with a similar recursive relation as for $\mathcal{D}_{i,j}$. With the obvious initial value $P_{M,N} = 1$, one has

$$\begin{aligned} P_{i,j} &= v_{i,j+1;\ 1} P_{i,j+1} \\ &+ v_{i+1,j+1;\ 2} P_{i+1,j+1} \\ &+ v_{i+1,j;\ 3} P_{i+1,j} \ . \end{aligned} \tag{7}$$

The fuzzy assignment matrix $\mathcal{W}_{i,j}$ is used when moving one of the proteins onto the consensus chain according to the minimum of Eq. 2.

2.4 Calculation of the consensus chain

The coordinates $\mathbf{y}_j$ for the consensus chain are calculated using the coordinates of all the K proteins and the K fuzzy assignment matrices from each of the pairwise alignments. In our approach we construct $\mathbf{y}_j$ according to

$$\mathbf{y}_j = \sum_{k=1}^{K} \alpha_j^{(k)} \tilde{\mathbf{y}}_j^{(k)} \ , \quad \text{where} \tag{8}$$

$$\tilde{\mathbf{y}}_j^{(k)} = \sum_{i=1}^{N_k} \tilde{\mathcal{W}}_{i,j}^{(k)} \mathbf{x}_i^{(k)} \ . \tag{9}$$

The last expression (Eq. 9) computes the (weighted) average position of the atoms in protein k that are matched to atom j in the consensus chain. The weights $\tilde{\mathcal{W}}_{i,j}^{(k)}$ are normalized columns of the fuzzy assignment matrix $\mathcal{W}_{i,j}^{(k)}$:

$$\tilde{\mathcal{W}}_{i,j}^{(k)} = \frac{\mathcal{W}_{i,j}^{(k)}}{\sum_{i'=1}^{N_k} \mathcal{W}_{i',j}^{(k)}} \ . \tag{10}$$

The final position of consensus atom j is given in Eq. 8, which is a weighted sum of all $\tilde{\mathbf{y}}_j^{(k)}$; chains with a large $\alpha_j^{(k)}$ contribute more to the final consensus atom $\mathbf{y}_j$, compared to chains with a low $\alpha_j^{(k)}$. These weights are given by

$$\alpha_j^{(k)} = \frac{e^{-\beta_j^{(k)}/\psi}}{\sum_{k'=1}^{K} e^{-\beta_j^{(k')}/\psi}} \ , \tag{11}$$

where ψ is a parameter and $\beta_j^{(k)}$ is the distance from the center of mass of the positions $\tilde{\mathbf{y}}_j^{(k')}, (k' = 1, ..., K)$ to $\tilde{\mathbf{y}}_j^{(k)}$ itself. This expression (Eq. 11) gives approximately equal weights to positions $\tilde{\mathbf{y}}_j^{(k')}$ that are within the characteristic distance ψ from the center of mass, whereas positions $\tilde{\mathbf{y}}_j^{(k')}$ that are much further away receive a negligible weight. When calculating the center of mass of the positions $\tilde{\mathbf{y}}_j^{(k')}$, only proteins where the total matching $\sum_{i=1}^{N_k} \mathcal{W}_{i,j}^{(k)}$ to consensus atom j is larger than 10^{-4}, are considered.

2.5 Initialization of the consensus chain

The multiple alignment procedure starts with the definition of an initial consensus chain of length M, where M is a parameter for the algorithm, chosen to be $M = \kappa \max_k\{N_k\}$, where $\kappa \geq 1$. The consensus chain is then initiated as the longest of the K proteins to align, with random coordinates for the left-over tails (if $\kappa > 1$).

A pre-alignment step is then performed where all structures are moved towards the consensus chain using fuzzy assignment matrices $\mathcal{W}_{i,j}^{(k)}$, with a band diagonal structure. The consensus chain is then updated once using Eq. 8. It should be noted that the performance of the algorithm does not depend on the details of the initialization of the consensus chain. This is due to the fact that the fuzzy pairwise alignment procedure starts with a large value for the T-parameter (see Eq. 4), making the initial 3-D coordinates for the consensus chain less important.

2.6 Summary of the method

As stated in the beginning, the multiple alignment of K protein structures is accomplished by iteratively aligning each of the K proteins to a consensus chain using fuzzy assignment matrices, followed by re-computation of the consensus chain. The T-parameter that controls the degree of fuzziness of the assignment matrices is *annealed* during the iterative procedure from a large value to a small value. This ensures the necessary transition from fuzzy assignment matrices to binary ones, where uniquely matched atoms to the consensus chain are identified. The major algorithmic steps are summarized in Figure 1.

The final multiple alignment of the K proteins is defined entirely by what is matched to the virtual consensus atoms in the K pairwise alignments, but the consensus atoms themselves are not considered part of the final alignment. Protein atoms that are matched to one and the same consensus atom, are considered to be matched to each other and form an *alignment column.* If a consensus atom is matched to a gap in one of the pairwise alignments, this protein will naturally have a gap at this position in the multiple alignment as well. Any protein atom matched to a gap in the consensus chain, and any protein atom that is matched to a consensus atom without any other protein atoms matched to it, is considered to be unaligned. Consensus atoms that are entirely unmatched do not contribute to the final multiple alignment.

3 Results and Discussion

3.1 Data sets used

To test the quality of our alignment algorithm, we have compared alignments on a set of protein families from the HOMSTRAD database (Mizuguchi *et al.*, 1998). Here, the goal was not a full investigation of all families, but rather to explore a limited set with representative variation. Table 1 lists the protein families from HOMSTRAD used in our comparison. We denote proteins by their PDB (Berman *et al.*, 2000) identifier.[†]

3.2 Heuristic post-processing

The atoms in the different protein chains are matched through the mediation of the virtual consensus atoms. If there are too few consensus atoms at a given position, protein atoms will be unaligned there. Conversely, if there are many consensus atoms close to a given position, protein atoms which really should be aligned will split up between the consensus atoms, leading to unnecessary gaps in the assignment.

[†]PDB, Protein Data Bank, located at http://www.rcsb.org/pdb/ .

1. Initialization.
    - (a) Move all proteins to their common center of mass and rescale coordinates such that the largest distance between atoms within the chains is unity.
    - (b) Initiate the consensus chain.
    - (c) Initiate the temperature, $T = 10$.
2. Alignment procedure.
    - (a) For each protein k, proceed row by row $i \rightarrow i+1$, $i = 1, \ldots, M$, and in each row column by column, $j \rightarrow j+1$, $j = 1, \ldots, N_k$, and calculate in the given order:
        - i. $v^{(k)}_{i,j;\ l}$ from Eq. 4.
        - ii. $\widetilde{\mathcal{D}}^{(k)}_{i,j;\ l}$ from Eq. 5.
        - iii. $\mathcal{D}^{(k)}_{i,j}$ from Eq. 3.
    - (b) Compute the fuzzy matching matrices from Eq. 6.
    - (c) For each protein k, minimize Eq. 2 by rotation and translation of the protein to the consensus chain.
    - (d) Repeat items 2(a)-2(c) 3 times, or until $\max_{i,j}(|v_{i,j;\ 2} - v^{(\text{old})}_{i,j;\ 2}|) < 0.05$, whichever comes first. $v^{(\text{old})}_{i,j;\ 2}$ are variables from the previous iteration step.
    - (e) Update the consensus chain according to Eq. 8
    - (f) Lower the T-parameter, e.g. $T \rightarrow \epsilon * T$.
    - (g) Repeat items 2(a)-2(f) until $(1/K)\sum_k \left(\sum_{i,j}(\mathcal{W}^{(k)}_{i,j})^2 / \sum_{i,j}\mathcal{W}^{(k)}_{i,j}\right) < 1 - 10^{-10}$.
3. Extract the multiple structure alignment by examining the alignment of each protein to the consensus chain.

Figure 1: The essential algorithmic steps in the multiple structure alignment algorithm.

At high temperatures, the consensus atoms move to their approximate positions, but as the temperature decreases, there is a tendency that some positions end up with too few or to many consensus atoms. For this situation to be remedied, a whole range of consensus atoms has to be moved one or more steps along the chain, in order to shift extra consensus atoms into or out of the interior of the alignment. There is no force in the alignment procedure that makes this happen. The pairwise protein-consensus alignments are optimized with respect to the *current* position of the consensus atoms, and the consensus atoms are likewise moved to *currently* matched protein atoms. These two steps happen sequentially rather than simultaneously. Therefore, the algorithm does not “know” that a globally superior alignment would be achieved if the consensus atoms were shifted along the sequence.

The above situation is taken care of by a series of heuristic procedures, that insert or delete consensus atoms near the end of the annealing process. The following criteria are checked for:

1. Consider two consecutive consensus atoms. If these are matched to at least one gap in each of the K proteins, then the two consensus atoms are merged by translating one of them to the midpoint

Table 1: Protein families from the HOMSTRAD database used in our comparison.

Family	PDB entries for the proteins	Class	No. of proteins	Average length	Average identity (%)
ghf7	1cel, 1egl, 2ovw, 2a39	all β	4	399	47
tim	1amk, 5timA, 1htiA, 1timA, 1ypiA, 1treA, 1ydvA, 1aw2A, 2btmA, 1tcdA	α/β barrel	10	249	47
cyt3	2cdv, 2cym, 1wad, 3cyr, 2cy3, 1age	all α	6	110	40
cytc	1yea, 1ycc, 2pcbB, 5cytR, 1ccr, 1cry, 1hroA, 1cxc, 1c2rA, 155c, 2c2c	all α	11	111	44
intb	1afcA, 2afgA, 2fgf, 2mib, 1i1b, 1iraX	all β	6	137	30
ricin	1apa, 1qciA, 1abrA, 1fmp, 1mrj, 1mrg, 1cf5A	α plus β	7	254	38
asprs	1aszA, 1lylA, 1b8aA	multi domain	3	470	28

between the original consensus atom positions, and deleting the other one. (Only consensus atoms actually matched to something are considered here. There is often a pool of unmatched consensus atoms, but these are excluded in this analysis.)

2. If an atom in one of the proteins is unaligned, a consensus atom is inserted at this position, unless there is another consensus atom within the distance 1.5 Å.

3. If two consensus atoms are within the distance 1.0 Å from each other, one of them is removed. This deletion is performed so as to make the average distance between the remaining consensus atoms as close to 3.8 Å as possible.

The above heuristics are performed every fifth iteration when $\Sigma > 0.99$, and before the final iteration. The saturation Σ is a measure of how close to zero or unity the elements in the assignment matrices $\mathcal{W}_{i,j}^{(k)}$ are, and is defined as

$$\Sigma = \frac{1}{K} \sum_k \left(\frac{\sum_{i,j} (\mathcal{W}_{i,j}^{(k)})^2}{\sum_{i,j} \mathcal{W}_{i,j}^{(k)}} \right). \tag{12}$$

3.3 Comparing with HOMSTRAD alignments

Besides HOMSTRAD (Mizuguchi *et al.*, 1998), we have compared our result with the Monte Carlo method of Guda *et al.* (2001) which is available through a WWW-server.[‡] When comparing either of the two methods against HOMSTRAD, we have counted the number of aligned columns that exactly matched that of the manual alignment; here the columns with only one atom are also counted. For each method we also present the number of aligned columns with two atoms or more, and the average column distance in the final geometrical configuration. The column distance is defined as the average of all pairwise geometric

[‡]http://cl.sdsc.edu/mc/mc.html

distances between atoms in a column. Columns with only one atom obviously have no column distance, and are not included in the average.

Table 2: Summary of the results from the comparison.

	Our method			Guda *et al.*		
Family	No. aligned columns	Ave. column distance (Å)	No. correct columns	No. aligned columns	Ave. column distance (Å)	No. correct columns
ghf7	403	1.30	402	402	1.30	406
tim	261	1.08	236	249	1.04	229
cyt3	118	1.81	98	111	1.69	89
cytc	132	1.41	102	124	1.20	94
intb	158	1.63	126	155	1.83	121
ricin	279	1.29	238	251	1.20	221
asprs	457	2.22	400	446	2.24	377

Table 2 shows the result obtained for the families listed in Table 1. Our method produces alignments in good agreement with the manual alignment from Homstrad as can be seen from the number of matched columns or by visual inspection of the alignments. Compared to the Monte Carlo method by Guda et al. we usually have more correctly aligned columns. The average column distance for the final geometric configuration is, on the average, somewhat higher for our method. This can be explained by the fact that our method generally aligns more columns. As more columns are included in the alignment, some of these are likely to be hard-to-align ones with a higher column distance, compared to the case when the method aligns fewer columns and then presumably chooses regions that match each other more closely.

Columns with large column distances may also be a result of the heuristic post-processing (cf. section 3.2). Often there are too few consensus atoms at various places; therefore unaligned protein atoms are given a consensus atom in the heuristic post-processing, unless there is a consensus atom close by already. This, however, has an unfortunate side effect: some protein atoms that really should be unaligned get a consensus atom that then aligns itself to some other unaligned atom within a distance of a few Ångströms. This causes a false alignment between the two proteins. These false alignments mainly occur in loop regions and do not disrupt the overall alignment, but they can imply a high column distance for some columns. Thus, there appears to be room for improvement in terms of finding better heuristic post-processing methods; such refinements might lead to fewer false alignments, and to a lower average column distance.

The result of the multiple alignment of the triosephosphate isomerase (tim) family (Lolis *et al.*, 1990) is shown in Figure 2. This family consists of 10 proteins belonging to the α/β-barrel structure class. Comparing to the manual alignment from Homstrad, we align 236 of the total 260 columns correctly. A detailed investigation shows that all α-helix and β-sheet structures are correctly aligned. Misalignments occurs in loop regions between α-helix and β-sheets.

Our multiple alignment algorithm depends on a number of parameters for its operation. Table 3 shows the values, used in this paper, for important parameters. We have used the same set of parameters for all the families tested.

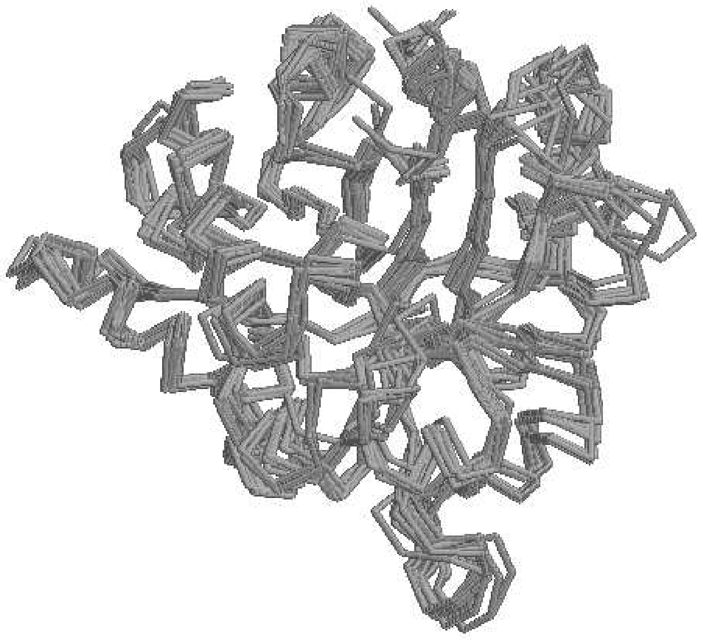

Figure 2: Alignment of the 10 proteins in the triose phosphate isomerase (tim) family from HOMSTRAD using our method. Each protein is optimally aligned to the consensus chain (which is not shown). The proteins are: 1amk, 5timA, 1htiA, 1timA,1ypiA, 1treA, 1ydvA, 1aw2A, 2btmA and 1tcdA.

Table 3: Parameters used in our multiple alignment method.

Parameter	Value	Legend
λ	7.0	The gap cost in the pairwise alignment.
λ_{ext}	0.7	The gap extension cost in the pairwise alignment.
κ	1.1	The multiplicative factor used when defining the length of the consensus chain (cf. section 2.5).
ϵ	0.7	The annealing rate of T (see Figure 1).
ψ	0.9	The width used in the weighting of the different chains when updating consensus atoms (see Eq. 11).

3.4 Scaling properties

Our algorithm for multiple alignment of protein structures was implemented in C++ on a PC running the Linux operating system. The running times ranged from 11 seconds for the cytochrome-c3 (cyt3) family to 140 seconds for the glycosyl hydrolase family 7 (ghf7) on a 2.4 GHz processor (Intel). The required CPU time depends on the lengths of the proteins and the number of structures to align. The pairwise alignment algorithm scales like $N * M$ for two structures of length N and M. However, for proteins structures with equal lengths, the required CPU time for the multiple alignment algorithm grows approximately linearly with the number of structures to align. This is due to the consensus chain that all structures are aligned to, and that no initial all-to-all pairwise alignment is required.

4 Conclusions

In summary, we have presented a novel method for multiple alignment of proteins structures. It has been tested on several protein families with encouraging results. Our approach has appealing properties, such as:

- It is fully automatic, requiring (in principle) only the 3-D coordinates of the C_α-atoms as input and no initialization using all-to-all pairwise alignments.
- With the definition of a consensus chain to which all structures are aligned, it gives rise to a scaling property, where the CPU consumption grows (approximately) linearly with the number of structures to align.
- With the underlying fuzzy pairwise alignment, the method can easily be extended to handle more detailed chain representations (e.g. side chain orientation) and additional user-provided constraints of almost any kind.

5 Acknowledgments

We thank Carsten Peterson for fruitful discussions. This work was in part supported by the Swedish Research Council.

References

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The protein data bank. *Nucleic Acids Res.*, **28**, 235–242.

Blankenbecler, R., Ohlsson, M., Peterson, C., and Ringnér, M. (2003). *Matching protein structures with fuzzy alignments.* (Lund University preprint LU TP 02-39, submitted to *Proc. Natl. Acad. Sci. USA*)

Gerstein, M., and Levitt, M. (1996). Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. In D. J. States, P. Agarwal, T. Gaasterland, L. Hunter, and R. F. Smith (Eds.), *Proceedings of the fourth international conference on intelligent systems in molecular biology* (p. 59-67). Menlo Park, CA: AAAI Press.

Guda, C., Scheeff, E., Bourne, P., and Shindyalov, I. (2001). A new algorithm for the alignment of multiple protein structures using Monte Carlo optimization. In *Proceedings of the pacific symposium on biocomputing* (Vol. 6, p. 275-286).

Holm, L., and Sander, C. (1993). Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, **233**, 123–138.

Leibowitz, N., Fligelman, Z. Y., Nussinov, R., and Wolfson, H. J. (2001). Automated multiple structure alignment and detection of a common substructural motif. *Proteins*, *43*, 235-245.

Lolis, E., Alber, T., Davenport, R., Rose, D., Hartman, F., and Petsko, G. (1990). Structure of yeast triosephosphate isomerase at 1.9-A resolution. *Biochemistry*, *29*, 6609-6618.

Mizuguchi, K., Deane, C., Blundell, T., and Overington, J. (1998). HOMSTRAD: a database of protein structure alignments for homologous families. *Protein Sci*, *7*, 2469-2471.

Needleman, S. B., and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453.

Neumann, J. von. (1937). Some matrix-inequalities and metrization of matric-space. *Tomsk Univ. Rev.*, **1**, 286–300.

Orengo, C., and Taylor, W. (1996). SSAP: sequential structure alignment program for protein structure comparison. *Methods Enzymol*, *266*, 617-635.

Shindyalov, I. N., and Bourne, P. E. (1998). Protein structure by incremental combinatorial extension of the optimal path. *Protein Eng.*, *11*, 739-747.

Szustakowski, J., and Weng, Z. (2000). Protein structure alignment using a genetic algorithm. *Proteins*, *38*, 428-440.

Šali, A., and Blundell, T. (1990). Definition of general topological equivalence in protein structures: A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J. Mol. Biol.*, *212*, 403-428.