LU TP 00-07
SLAC-PUB-8429
April 13, 2000

A Novel Approach to Structure Alignment

Mattias Ohlsson, Carsten Peterson, Markus Ringnér[1]

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

Richard Blankenbecler[2]

Stanford Linear Accelerator Center
P.O. Box 4349, Stanford, CA 94309, USA

Submitted to *Structure with Folding & Design*

Running Head: Structure Alignment of Proteins

Keywords: protein structure alignment; permutation; mean field annealing; fuzzy assignment; database searching

[1] {mattias,carsten,markus}@thep.lu.se
[2] rzbth@slac.stanford.edu

Abstract

Background: Aligning protein structures is a highly relevant task. It enables the study of functional and ancestry relationships between proteins and is very important for homology and threading methods in structure prediction. Existing methods typically only partially explore the space of possible alignments and being able to efficiently handle permutations efficiently is rare.

Results: A novel approach for structure alignment is presented, where the key ingredients are: (1) An error function formulation of the problem simultaneously in terms of binary (Potts) assignment variables and real-valued atomic coordinates. (2) Minimization of the error function by an iterative method, where in each iteration a mean field method is employed for the assignment variables and exact rotation/translation of atomic coordinates is performed, weighted with the corresponding assignment variables. The approach allows for extensive search of all possible alignments, including those involving arbitrary permutations. The algorithm is implemented using a C_α-representation of the backbone and explored on different protein structure categories using the Protein Data Bank (PDB) and is successfully compared with other algorithms.

Conclusions: The approach performs very well with modest CPU consumption and is robust with respect to choice of parameters. It is extremely generic and flexible and can handle additional user-prescribed constraints easily. Furthermore, it allows for a probabilistic interpretation of the results.

Introduction

Aligning protein structures is a subject of utmost relevance. It enables the study of functional relationship between proteins and is very important for homology and threading methods in structure prediction. Furthermore, by grouping protein structures into fold families and subsequent tree reconstruction, ancestry and evolutionary issues may get unraveled.

Structure alignment amounts to matching two 3D structures such that potential common substructures, e.g. α-helices, have priority. The latter is accomplished by allowing for gaps in either of the chains. Also, the possibility of permuting sites within a chain may be beneficial. At first sight, the problem may appear very similar to sequence alignment, as manifested in some of the vocabulary (gap costs etc.). However, from an algorithmic standpoint there is a major difference. Whereas sequence alignment can be solved within polynomial time using dynamical programming methods [1], this is not the case for structure alignment since rigid bodies are to be matched. Hence, for all structure alignment algorithms the scope is limited to high quality approximate solutions.

Existing methods for structure alignment fall into two broad classes, depending upon whether one (1) directly minimizes the *inter*-atomic distances between two structures or (2) minimizes the distance between substructures that are either preselected or supplied by an algorithm involving *intra*-atomic distances.

One approach within the first category is the iterative dynamical programming method [2, 3], where one first computes a distance matrix between all pairs of atoms (e.g. C_α) forming a similarity matrix, which by dynamical programming methods gives rise to an assignment matrix mimicking the sequence alignment procedure. One of the chains is then moved towards the other by minimizing the distance between assigned pairs. This method does not allow for permutations. Another inter-atomic approach is pursued in [4], where the area rather than distances between two structures is minimized.

In [5] the approach is different. Here one compares distance matrices within each of the two structures to be aligned, which provide information about similar substructures. The latter are subsequently matched. A similar framework is used in [6] and also in [7]. Not surprisingly, in [5, 6] and [7] permutations can in principle be dealt with.

There are implementation issues shared by both methodologies above. One is structure encoding (C_α and/or C_β of the chains). For many comparisons C_α appears to be sufficient, whereas in some cases C_β is needed. Also, the choice of distance metric is a subject of concern in order to avoid the influence of outliers.

The iterative dynamical programming method [3] has been extensively assessed for backbone structures [8] from the SCOP [9] database, in which protein structures have been classified by visual inspection. Some comparisons with SCOP have also been performed [10] using the method in [6].

Here we present a novel approach, which shares some of its philosophy from the iterative dynamical programming method [3]. Its key ingredients are: (A) An error function formulation of the problem simultaneously in terms of binary (Potts) assignment variables and real-valued atomic coordinates and (B) minimization of the error function by an iterative method, where each iteration contains two steps:

1. A mean field procedure for minimizing with respect to the assignment variables.
2. Exact rotation and translation of atomic coordinates weighted with the corresponding assignment variables.

The approach, which is very general, has some very appealing properties:

- Implicit complete exploration of the entire space of alignments, which allows for arbitrary permutations. To our knowledge, no other approach has this feature.
- Probabilistic interpretation of the results. This feature is present without tedious Monte Carlo estimates since the algorithm is deterministic. Among other things, this implies that the approach is less sensitive to the choice of distance metric, since the distances are weighted with fuzzy numbers.
- With its generality, almost arbitrary additional constraints are easily incorporated into the formalism including different functional forms of gap penalties.

The approach is tested using C_α-representation of backbones, by comparing the results with the approaches of [3] and [5] as implemented in the YALE ALIGNMENT SERVER and DALI respectively and in one instance also with [6] (ENTREZ). In choosing protein pairs to align we followed [8] to a large extent. In [8] pairs with marginal sequence overlap but where each protein in a pair belongs to the same SCOP superfamily and therefore have a similar structure were picked for assessment. We selected pairs from a varied selection of the families used in [8] to test our algorithm:

- *Dihydrofolate Reductases* (α/β)
- *Globins* (all-α)
- *Plastocyanin/azurin* (all-β)
- *Immunoglobulins* (all-β)

In addition, we test the permutation capacity of our approach, by aligning:

- *Permuted proteins* (winged helix fold)

When assessing the algorithm, we limit ourselves to a core version, where C_β degrees of freedom are not included. Also, no post-processing of the results is done. We defer such elaborations and others to forthcoming publication. Nevertheless, the core version of our approach is already very competitive even for chains, where permutations are not needed. For the latter case, the other algorithms could not be tested using the corresponding WWW-servers. In the instances, where we have tested it for this kind of problems, it also performs well.

The algorithm is implemented in C++. Given its generality and power, the CPU demand is quite modest – it scales like the chain lengths squared and on the average requires a few seconds on a Pentium 400MHz PC.

Methods

The Algorithm

In what follows we have two proteins with N_1 and N_2 atoms to be structurally aligned. This is accomplished by a series of weighted rigid body transformations of the first chain, keeping the second chain fixed. We denote by $\mathbf{x}_i$ $(i = 1, ..., N_1)$ and $\mathbf{y}_j$ $(j = 1, ..., N_2)$ the atom coordinates of the first and second chain, respectively. The phrase "atom" will be used throughout this paper in a generic sense – it could represent individual atoms but also groups of atoms. In our applications it will mean C_α-atoms along the backbone. A square distance metric between the chain atoms is used,

$$d_{ij}^2 = |\mathbf{x}_i - \mathbf{y}_j|^2 \tag{1}$$

but the formalism is not confined to this choice.

We start by discussing the encodings and error function and then we present a method for minimizing the latter.

The Gapless Case. For pedagogical reasons, we start off with the gapless case with $N_1 = N_2$. We define binary assignment variables s_{ij} such that $s_{ij} = 1$ if atom i in one chain matches j in the other and $s_{ij} = 0$ otherwise. Since every atom in one chain must match one atom in the other, the following conditions must be fulfilled:

$$\sum_{i=1}^{N_1} s_{ij} = 1 \quad j = 1, \ldots, N_2 \tag{2}$$

$$\sum_{j=1}^{N_2} s_{ij} = 1 \quad i = 1, \ldots, N_1 \tag{3}$$

A suitable error function to minimize subject to the above constraints (Eqs. (2,3)) is

$$E_{\mathrm{chain}} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} s_{ij} d_{ij}^2 \tag{4}$$

where the spatial degrees of freedom, $\mathbf{x}_i$, are contained in the distance matrix d_{ij}^2. Thus whenever s_{ij}=1 one adds a penalty d_{ij}^2 to E_{chain}. Note that Eq. (4) is to be minimized both with respect to the binary variables s_{ij} and the real-valued coordinates $\mathbf{x}_i$.

The Gapped Case. Allowing for gaps in either of the chains is implemented by extending s_{ij} to include 0-components in a compact way; $s_{i0} = 1$ and $s_{0j} = 1$ if an atom (i or j) in one chain is matched with a gap in the other and vice versa. Hence, gap positions are not represented by individual elements in s_{ij}; rather the gap-elements correspond to common sinks. The matrix $\mathcal{S}$, with elements s_{ij}, containing gap-elements is shown in Eq. (5).

$$\mathcal{S} = \left(\begin{array}{c|cccc} & s_{01} & s_{02} & \cdots & s_{0N_2} \\ \hline s_{10} & s_{11} & s_{12} & \cdots & s_{1N_2} \\ s_{20} & s_{21} & s_{22} & \cdots & s_{2N_2} \\ . & & & & \\ . & & & & \\ . & & & & \\ s_{N_1 0} & s_{N_1 1} & s_{N_1 2} & \cdots\cdot & s_{N_1 N_2} \end{array} \right) \tag{5}$$

Some caution is needed when generalizing Eqs. (2,3) to host gaps, since the elements of the first row and column (gap-mappings containing the index 0) in Eq. (5) differ from the others in that they need not sum up to 1. Hence Eqs. (2,3) becomes

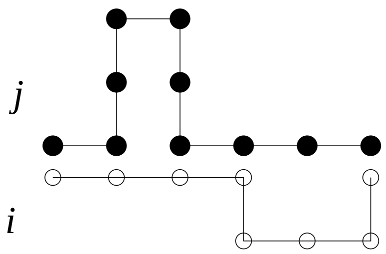
$$\begin{aligned} &\sum_{i=0}^{N_1} s_{ij} = 1; \quad j = 1, \ldots, N_2 \\ &\sum_{j=0}^{N_2} s_{ij} = 1; \quad i = 1, \ldots, N_1 \end{aligned} \tag{6}$$

where the first condition can be rewritten as

$$\sum_{i=1}^{N_1} s_{ij} = 1 \quad \text{or} \quad \sum_{i=1}^{N_1} s_{ij} = 0; \quad j = 1, \ldots, N_2 \tag{7}$$

The encoding (s_{ij}) of matches and gaps is illustrated in Fig. 1 with a simple example.

$$\mathcal{S}$$



$$\left(\begin{array}{c|cccccccccc}
 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\
\hline
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right)$$

Figure 1: A simple example of the assignment matrix $\mathcal{S}$ (right) corresponding to the matching of the two toy chains (left).

Assuming a constant penalty per inserted gap one has the error function

$$E = E_{\text{chain}} + \sum_{i=1}^{N_1} \lambda_i^{(1)} s_{i0} + \sum_{j=1}^{N_2} \lambda_j^{(2)} s_{0j} \tag{8}$$

where $\lambda_i^{(1)}$is the cost for matching atom i in the first chain with a gap in the second chain, and similarly for $\lambda_j^{(2)}$. The position dependence of the gap costs, $\lambda_i^{(1)}$ and $\lambda_j^{(2)}$, originates from the fact that it is desirable not to break α-helix and β-strand structures.

In Eq. (8) the gap penalties are proportional to gap lengths. In sequence alignment it is conjectured that gap penalties consist of two parts; a penalty for opening a gap and then a penalty proportional to the gap length. As in [3], we will for structure alignment here adopt the same gap cost philosophy, i.e. $\lambda_i^{(1)}$ and $\lambda_j^{(2)}$ for opening a gap and a position-independent δ per consecutive gap. Hence, Eq. (8) generalizes to

$$\begin{aligned} E &= E_{\text{chain}} + \sum_{i=1}^{N_1} \lambda_i^{(1)} s_{i0} + \sum_{j=1}^{N_2} \lambda_j^{(2)} s_{0j} \\ &+ \sum_{i=2}^{N_1} \left(\delta - \lambda_i^{(1)}\right) s_{i-1,0} s_{i0} + \sum_{j=2}^{N_2} \left(\delta - \lambda_j^{(2)}\right) s_{0,j-1} s_{0j} \end{aligned} \tag{9}$$

where products like $s_{i-1,0}s_{i0}$ are 1 if two adjacent atoms are matched to gaps.

Minimization. Next we need an efficient procedure for minimizing E with respect to both s_{ij} and $\mathbf{x}_i$ subject to the constraints in Eqs. (6,7). As mentioned above, this minimization problem is non-trivial due to the rigid body constraint. A similar problem in terms of fitting structures with relevance factors was probed in [11] for track finding problems with a template approach using the mean field approximation. Here we will adopt a similar approach.

In our formulation, the inherent optimization difficulty resides in the binary part (s_{ij}) of the problem. Hence, minimizing Eq. (9) using a simple updating rule for s_{ij} will very likely yield poor solutions due to local minima. Well known stochastic procedures such as simulated annealing (**SA**) [12] for avoiding this are too costly from a computational standpoint. In the *mean field* (**MF**) approach [13], the philosophy behind SA is retained, but the tedious simulations are replaced by an efficient deterministic process. The binary variables s_{ij} are then replaced by continuous *mean field* variables $v_{ij} \in [0, 1]$, with a dynamics given by iteratively solving the MF equations for a decreasing set of temperatures T down to T_0, where most of the v_{ij} approach either 1 or 0. These continuous MF variables can evolve in a space not accessible to the original intermediate variables. The intermediate configurations at non-zero T have a natural probabilistic interpretation.

For s_{ij} satisfying Eq. (6), the MF equations for the corresponding v_{ij} read

$$v_{ij} = \frac{e^{u_{ij}/T}}{\sum_{k=0}^{N_2} e^{u_{ik}/T}}; \quad i = 1, ..., N_1 \tag{10}$$

where the force u_{ij} is given by

$$u_{ij} = -\frac{\partial E}{\partial v_{ij}} \tag{11}$$

and is computed by substituting s_{ij} with v_{ij} in E (Eq. (9)). Note that the desired normalization condition, Eq. (6),

$$\sum_{j=0}^{N_2} v_{ij} = 1; \quad i = 1, ..., N_1 \tag{12}$$

is fulfilled automatically in Eq. (10). The other condition (Eq.(7)) is enforced by adding a penalty term

$$\begin{aligned} E_\gamma &= \gamma \sum_{j=1}^{N_2} [(\sum_{i=1}^{N_1} v_{ij})(\sum_{k=1}^{N_1} v_{kj} - 1)] \\ &= \gamma \sum_{i=1}^{N_1} \sum_{k \neq i}^{N_1} \sum_{j=1}^{N_2} v_{ij} v_{kj} \end{aligned} \tag{13}$$

where γ is a parameter and the last equality follows from the fact that $v_{ij}^2 = v_{ij}$ for T=0.

So far we have only looked at the assignment part when minimizing the error function. When updating the mean field variables v_{ij}, using the MF equations, the distance measure d_{ij}^2 is a fixed quantity. This corresponds to having the chains at fixed positions. However, we also want to minimize the distance between the two chains. Based on the *probabilistic* nature of the mean field variables we propose to update the chain positions using the (fuzzy) assignment matrix $\mathcal{V}$, with elements v_{ij}. This is done simultaneously with the updating of v_{ij}. Explicitly, one of the chains will be moved in order to minimize the chain error function E_{chain} (Eq. (4)).

The distance measure d_{ij}^2 depends on the translation vector $\mathbf{a}$ and the rotation matrix $\mathcal{R}$, making a total of six independent variables. Let $\mathbf{x}_i'$ be the coordinates of the translated and rotated protein, i.e. $\mathbf{x}_i' = \mathbf{a} + \mathcal{R}\mathbf{x}_i$, then

$$E_{\text{chain}} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} v_{ij} \left(\mathbf{a} + \mathcal{R}\mathbf{x}_i - \mathbf{y}_j\right)^2 \tag{14}$$

This minimization problem can be solved exactly with closed-form expressions for $\mathcal{R}$ and $\mathbf{a}$ that minimizes E_{chain} [14]. It should be noted that this solution is rotationally invariant (independent of $\mathcal{R}$) for the special case when the atoms in the two chains matches each other with the same weight, i.e. when v_{ij} = constant for all i and j, which is the case for high T.

In summary, for a decreasing set of temperatures T, one iterates until convergence:

1. The MF equations (Eq. (10)).
2. Exact translation and rotation of the chain (Eq. (14)).

We stress again that step 2 is done with the fuzzy MF assignment variables v_{ij} and not with the binary ones, s_{ij}. After convergence, v_{ij} are rounded off to 0 or 1 and *rms* (root-mean-square-distance) is computed for the matching pairs. Algorithmic details can be found in the next subsection.

The forces u_{ij} entering Eq. (10) are proportional to d_{ij}^2 (Eqs. (4,11)). It is the ratio d_{ij}^2/T that counts. Hence, for large temperatures T, v_{ij} are fairly insensitive to d_{ij} and many potential matching pairs (i, j) contribute fairly evenly. As the temperature is decreased, a few pairs (the ones with small d_{ij}) are singled out and finally at the lowest T only one winner remains. One can view the situation as that around each atom i one has a Gaussian domain of attraction, which initially (large T) has a large width, but gradually shrinks to a small finite value.

The fuzziness of the approach is illustrated in Fig. 2, where the evolution of v_{ij}, as T is lowered, is shown for parts of the first helices of 1ECD and 1MBD (see next section) together with snap-shots of the corresponding chain sections. At high T all v_{ij} are similar; all potential matches have equal probability. At lower T, several v_{ij} have approached 0 or 1 and the movable chain is moving in the right direction. At yet lower T, note that a few v_{ij} converge later than the majority. These are in this example related to the matching of the last atom in one of the chains. This atom has two potential candidates to match resulting in a number of v_{ij} that converge last.

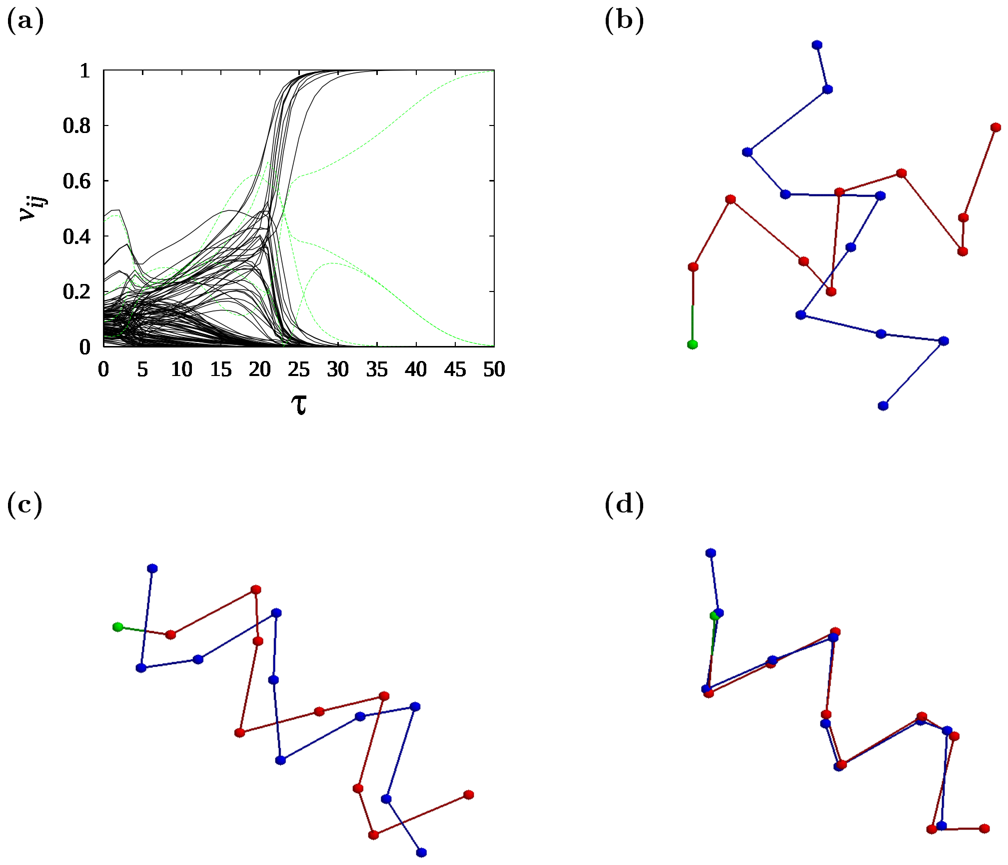


Figure 2: Illustration of the fuzziness of the approach. The alignment shown is for 10 atoms in the first helices in the proteins 1ECD (blue) and 1MBD (red). **(a)** Evolution of all the 120 v_{ij} as a function of iteration time τ (T is lowered with τ). **(b)** Positions of the atoms at $\tau = 1$. For high T every atom in a protein feels all the atoms in the other protein and the problem is rotationally invariant. **(c)** $\tau = 12$; most of the relevant matchings are forcing the system to move in the right direction. **(d)** $\tau = 50$; the final assignments are done. The different snapshots are presented using different projections. Some v_{ij} approach 0 or 1 rather late and they are coloured green. These v_{ij} are related to the atom at the end of the 1MBD segment, which also is coloured green, and as can be seen in **(c)** the difficulty is whether to align this atom to the last or second last atom in the 1ECD segment.

Implementation

Here we give a very condensed, but yet self-contained and detailed description of the algorithm and the parameters involved, such that the results of this paper are reproducible.

Parameters. Two kind of parameters are used; the ones related to encoding of the problem (γ) and iteration dynamics (ϵ), where ϵ governs the annealing schedule (see below), and the ones specifying gap costs (λ, δ). The same set of parameters can be used for most of the pairs (see Table 1); the algorithm is remarkably stable.

Protein Family	ϵ	γ	λ	λ_{sheet}	λ_{helix}	δ
α/β, all-α	0.8	0.065	0.10	1.5λ	1.5λ	$\lambda/2$
Plastocyanin/azurin	0.8	0.035	0.10	2.0λ	2.0λ	$\lambda/5$
Immunoglobulins	0.8	0.040	0.15	2.0λ	2.0λ	$\lambda/5$
Winged helix fold	0.8	0.070	0.20	2.0λ	2.0λ	$\lambda/5$

Table 1: Parameters used in the algorithm. The first family involves 27 pairs, whereas the others one each.

Initialization. An initialization of the chains is made prior to the mean field alignment. First both chains are moved to their common center of mass. For the *random* initialization, this move is then followed by a random rotation of one of the chains. Most of the times, however, a *sequential* initialization is used that consists of minimizing Eq. (4) using a band-diagonal assignment matrix $\mathcal{S}$. This corresponds to a situation where, on the average, atom i in one of the chains is matched to atom i in the other. If not explicitly mentioned, sequential initialization is used for all the protein pairs in this paper.

Iteration Steps. The shortest chain is always chosen as the one that is moved ($\mathbf{x}_i$). The mean field variables v_{ij} are updated according to Eq. (10) where, in order to improve convergence, the derivatives in Eq. (11) are replaced by finite differences (see e.g. [15]). This update equation accounts for all mean field variables except for the first row of $\mathcal{V}$, which is updated according to

$$v_{0j} = 1 - \sum_{i=1}^{N_1} v_{ij}; \quad j = 1, ..., N_2 \tag{15}$$

The algorithmic steps are shown in Fig. 3. After convergence, no post processing is applied for the results in the next section.

Results

To test the quality of our alignment algorithm, we have compared alignments of protein pairs with results from other automatic procedures. For most of the tested pairs, each protein belongs to the same SCOP superfamily. The goal here is not a full investigation of all families but rather to explore a limited set with representative variation. Pairs were picked from a selection of the families investigated in [8]. Our choice of pairs is essentially

1. Initialization.
2. Rescale coordinates such that the largest distance between atoms within the chains is unity.
3. Initiate all v_{ij} close to $1/\max(N_1, N_2)$ (randomly).
4. Initiate the temperature (e.g. $T = 2$).
5. Randomly (without replacement) select one row, say row k.
6. Update all $v_{kj}, j = 0, ..., N_2$ according to Eq. (10).
7. Repeat items $5 - 6$ N_1 times (such that all rows have been updated once).
8. Repeat items $5 - 7$ until no changes occur (defined e.g. by $1/(N_1 N_2) \sum_{ij} |v_{ij} - v_{ij}^{(\text{old})}| \leq 0.0001$).
9. Rotation and translation of the shortest chain using the fuzzy assignment matrix $\mathcal{V}$.
10. Decrease the temperature, $T \to \epsilon T$.
11. Repeat items $5 - 10$ until all v_{ij} are close to 1 or 0 (defined e.g. by $1/N_1 \sum_{ij} v_{ij}^2 \geq 0.99$).
12. Finally, the mean field solution is given by the integer limit of v_{ij}, i.e. for each row i, $i = 1, ..., N_1$ select the column j^* such that v_{ij^*} is the largest element for this row. Let $s_{ij^*} = 1$ and all other $s_{ij} = 0$ for this row.

Figure 3: Algorithmic steps.

based on two criteria. First, the pairs should have diverse structures, and in particular include all-α, all-β, and α/β proteins. Second, in [8] some families are considered to be *very easy*, *easy* and *difficult* to align, respectively, and we included pairs from all these categories. In addition we have tested the algorithm on cases where permutations are needed.

Our results are compared with the Yale Alignment Server (http://bioinfo.mbb.yale.edu/align/) and Dali (http://www2.ebi.ac.uk/dali/). The Yale server applies post processing to its alignments by removing aligned pairs with too large root-mean-square-distance (*rms*) in an iterative manner subject to a termination criteria. A similar procedure is of course possible in our approach, but we have chosen at this stage to keep the algorithm clean. In the comparisons below we have for the Yale server quoted results both before and after the post-processing.

Unless otherwise stated, proteins are in what follows denoted by their Pdb [16] identifier, and in the case of chains or parts of chains with their Scop domain label. A summary of the results in terms of rms and the number of aligned atoms (N) is shown in Table 2

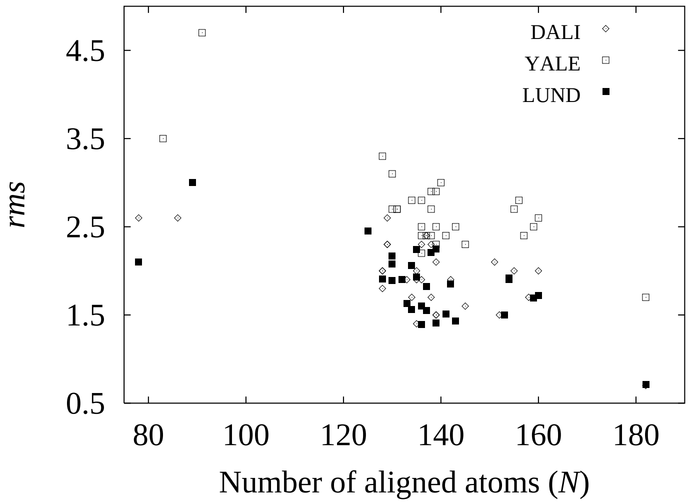


Figure 4: rms and N corresponding to Table 2. The YALE data correspond to no post processing (see text).

and in Fig. 4. Detailed comments upon these results and some additional ones can be found below.

With regard to the general performance one must keep in mind that it is not straightforward to assess alignment algorithms in terms of e.g. rms and N, since there are no obvious figure-of-merits. It is interesting to notice though that when inspecting aligned core regions in detail, we are close to the YALE alignments but in general with a lower rms. However, in such comparisons, we differ more from DALI. The YALE algorithm has been subject to comparison with SCOP classifications using a multiple alignment procedure [8], giving its and our alignments a higher credibility.

Dihydrofolate Reductases (α/β). These proteins belong to the SCOP class α/β, which contains α- and β-proteins that have mainly parallel beta sheets. They are considered very easy to align [8]. If we compare alignments of core structure parts using the three methods we find that they all essentially agree. However, one notes that the YALE results are very sensitive to the post processing.

Globins (all-α). In the all-α SCOP class we particularly study a set of globin proteins. In general, we get lower rms than the other algorithms for the same number of aligned residues. When comparing alignments from the three algorithms we find that an important aspect of our algorithm is manifested – allowing for permutation of individual atoms. The reason for this is that to optimally align secondary structures it is often beneficial to have a few permuted residues in loops between the secondary structures.

Protein family	Protein Pair		YALE		DALI		LUND	
			rms	N	*rms*	N	*rms*	N
Dihydrofolate Reductases	1DHFA	- 8DFR	1.7 (0.7)	182 (182)	0.7	182	0.7	182
	1DHFA	- 4DFRA	2.7 (1.2)	155 (130)	2.0	155	1.9	154
	1DHFA	- 3DFR	2.5 (1.2)	159 (143)	1.7	158	1.7	159
	8DFR	- 4DFRA	2.8 (1.3)	156 (131)	2.1	151	1.9	154
	8DFR	- 3DFR	2.6 (1.3)	160 (146)	2.0	160	1.7	160
	4DFRA	- 3DFR	2.4 (1.1)	157 (140)	1.5	152	1.5	153
Globins	2HHBA	- 2HHBB	2.3 (1.2)	139 (129)	1.5	139	1.4	139
	2HHBA	- 2LHB	2.7 (1.6)	131 (123)	1.8	128	1.9	130
	2HHBA	- 1MBD	2.4 (1.5)	141 (138)	1.5	139	1.5	141
	2HHBA	- 2HBG	2.4 (0.8)	138 (105)	1.7	138	1.6	137
	2HHBA	- 1MBA	2.9 (2.2)	138 (134)	2.3	136	2.2	138
	2HHBA	- 1ECD	3.1 (2.2)	130 (126)	2.3	129	2.2	130
	2HHBB	- 2LHB	2.5 (1.3)	136 (126)	1.7	134	1.6	134
	2HHBB	- 1MBD	2.3 (1.4)	145 (138)	1.6	145	1.4	143
	2HHBB	- 2HBG	2.4 (1.4)	136 (125)	2.0	135	1.6	133
	2HHBB	- 1MBA	3.0 (2.2)	140 (137)	2.3	138	2.2	139
	2HHBB	- 1ECD	2.8 (2.2)	136 (134)	2.3	129	2.1	134
	2LHB	- 1MBD	2.4 (1.0)	137 (121)	1.4	135	1.4	136
	2LHB	- 2HBG	2.7 (1.5)	131 (119)	2.0	128	2.1	130
	2LHB	- 1MBA	2.7 (1.8)	138 (130)	1.9	135	1.9	132
	2LHB	- 1ECD	2.7 (1.9)	130 (127)	2.0	128	1.9	128
	1MBD	- 2HBG	2.5 (1.6)	139 (130)	2.1	139	1.8	137
	1MBD	- 1MBA	2.5 (1.7)	143 (137)	1.9	142	1.8	142
	1MBD	- 1ECD	2.2 (1.6)	136 (134)	1.9	136	1.6	136
	2HBG	- 1MBA	2.9 (2.2)	139 (136)	2.4	137	2.2	135
	2HBG	- 1ECD	3.3 (2.5)	128 (125)	2.6	129	2.4	125
	1MBA	- 1ECD	2.8 (1.7)	134 (125)	1.9	133	1.9	135
Plastocyanin/azurin	1PLC	- 1AZU	4.7 (2.9)	91 (85)	2.6	86	2.1	78
Immunoglobulins	7FABL2	- 1REIA	3.5 (2.6)	83 (79)	2.6	78	3.0	89

Table 2: The root-mean-square-distance (*rms*) and the number of aligned residues (N) from the alignment of different protein pairs. The results are presented for several automatic alignment procedures; LUND refers to this work. For YALE the numbers within parenthesis refer to after post processing (see text).

If we again compare the core parts of the alignments from the three algorithms we find that they agree on a large fraction of the parts.

Plastocyanin/azurin (all-β). All-β proteins are difficult to align if one only takes backbone coordinates (C_α or C_β) into account, even though using C_β instead of C_α coordinates, in general, improves the results. As an initial example of all-β proteins we have looked at plastocyanin versus azurin. Even though this alignment is slightly more difficult than the previous cases, all three methods give similar *rms* and N and they all agree on the alignment of a majority of the core parts. For this example several restarts were performed with random initialization.

Immunoglobulins (all-β). A more difficult example of all-β proteins is immunoglobulins. We align the domain 7FABL2 with the chain 1REIA and find that we can find alignments with low *rms* that look good. However, if we investigate the alignment in detail we find that atoms in all core regions, except one, are misaligned. This is also the case in [8], where the same alignment is investigated. To get the core regions correctly aligned in [8] they improve their method and take side chain orientation into account. We expect that this is the case for our method too. When aligning strands using only C_α coordinates, strands in the two proteins are often matched satisfyingly to one another while the individual atoms are aligned such that one strand is translated with respect to the other. It is therefore obvious that side chain orientation is very important when aligning strands. For this example several restarts were performed with random initialization.

Permuted proteins – winged helix fold. Finally we look at permuted versions of similar folds. We compare two DNA binding domains related to transcription regulation. The compared domains both have the winged helix fold but one of them has the secondary structures in a circularly permuted order. This is a case where iterative dynamical programming algorithms will fail. We look at 1LEA and compare it to the ENTREZ-MMDB [17] structural domain 4 in chain B of 1XGN. This part of 1XGN is classified as a circularly permuted winged helix fold in SCOP. In the ENTREZ-MMDB database, which uses VAST (http://www.ncbi.nlm.nih.gov/Structure/VAST/) for alignments, 1XGNB4 is listed as a low priority structural neighbour to 1LEA, even though VAST does not allow for permutations of secondary structure. If one looks at the actual alignment one finds that the permuted secondary structures are not aligned. In Figure 5 we compare our alignment with VAST. We show the sequential parts of our alignment and in particular all parts with secondary structure are shown. VAST aligns only 39 residues in this comparison, while we align 60. We note that we get all the 39 of the aligned residues of VAST but that we in addition align the sheet at the end of 1LEA with the sheet at the beginning of the domain in 1XGNB. This demonstrates the importance of having a procedure that takes permutations into account, which our method does. Otherwise, important similarities between protein structures will not be found. For this example several restarts were performed with random initialization.

Discussion

A new approach to structure alignment has been presented and explored. It is based upon an error function encoding in terms of both binary assignment variables and real-valued atom coordinates. The encoding allows for an extensive search through all possible

```
              **************      *******        ******************
              5           18    26                               59     65    71
              |            |     |                                |      |     |
1LEA          TARQQEVFDLIRDH     PTRAEIAQRLGFRSPNAAEEHLKALARKGVIEIV     -GIRLLQE
1XGNB4        VAQARFLLAKIKRE     FAYRWLQN-D-M-PEGQLKLALKTLEKAGAIYGY     IYMYVRDV
              |            |     |                                |      |     |
            216          229   235                              265    206   212
```

Figure 5: Alignment of 1LEA against the ENTREZ-MMDB domain 1XGNB4. The '*' denotes atoms also aligned by VAST. 1XGNB4 is a circularly permuted version of 1LEA and our method finds this and aligns the sheet at the end of 1LEA with the sheet at the beginning of the domain in 1XGNB.

alignments, including the ones involving arbitrary permutations.

The error function is efficiently minimized using a mean field approximation of the assignment variables and exact translation/rotation of the atom coordinates. As a by-product of this approximation, a probabilistic interpretation of the result is available without tedious stochastic simulations. The approach is not sensitive to the choice of distance metric, and hence to a large extent ignores outliers.

Despite some conceptual similarities with the iterative dynamical programming method [3], our approach is probabilistic and more general. Also, and maybe more importantly, it is quite different since permutations are allowed from the outset. For the latter reason, the algorithm in [3] cannot be derived as a special case in any limit.

The method is readily extended to handle more detailed chain representations (e.g. side-chain orientation) and user-provided constraints of almost any kind.

The approach is evaluated using pairs of protein chains chosen to represent a wide variety of situations and the resulting alignments are successfully compared with other methods that are available on WWW-servers. This evaluation is done using C_α-representations of the chains.

Despite being very flexible, generic and covering the entire space of alignments the method is on the average as fast as [6], slightly slower than [3] and significantly faster than [5]. Also, it is very robust with respect to the algorithmic parameters used with a few exceptions. Once side-chains are included, the latter will disappear.

Biological Implications

There is a strong need for efficient protein structure alignment algorithms. Aligning proteins forms the basis for studying functional relationships among proteins and construction of phylogenetic trees. It is also very important for structure prediction.

Acknowledgments

We thank Bo Söderberg for valuable suggestions and Guoguang Lu for fruitful discussions. This work was in part supported by the Swedish Natural Science Research Council and the Swedish Foundation for Strategic Research. One of us (CP) thanks the Theory Group at SLAC, where this work was initiated, for its hospitality.

References

[1] 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**:443–453, 1971.

[2] D. V. Laurents, S. Subbiah, and M. Levitt. Structural similarity of dna-binding domains of bacteriophage repressors and the globin core. *J. Mol. Biol.*, **3**:141–148, 1993.

[3] M. Gerstein and M. Levitt. Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. In *Proceedings of the Fourth International Conference on Intelligent Systems in Molecular Biology*, Menlo Park, CA, 1996. AAAI Press.

[4] A. Falicov and F. E. Cohen. A surface of minimum area metric for the structural comparison of proteins. *J. Mol. Biol.*, **258**:871–892, 1996.

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

[6] J-F. Gibrat, T. Madej, and S. H. Bryant. Surprising similarities in structure comparison. *Curr. Opin. Struct. Biol.*, **6**:377–385, 1996.

[7] G. Lu. Top: A new method for protein structure and similarity searches. *J. Appl. Cryst.*, **33**:176–183, 2000.

[8] M. Gerstein and M. Levitt. Comprehensive assessment of automatic structural alignment against a manual standard, the SCOP classification of proteins. *Prot. Sci.*, **7**(2):445–456, 1998.

[9] T. J. Hubbard, A. G. Murzin, S. E. Brenner, and C. Chothia. SCOP: A structural classification of proteins database. *Nucl. Acid. Res.*, **25**:236–239, 1997.

[10] Y. Matsuo and S. H. Bryant. Identification of homologous core structures. *Proteins*, **35**:70–79, 1999.

[11] M. Ohlsson, C. Peterson, and A. L. Yuille. Track finding with deformable templates - the elastic arms approach. *Comp. Phys. Comm.*, **71**:77–98, 1992.

[12] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, **220**:671–680, 1983.

[13] C. Peterson and B. Söderberg. A new method for mapping optimization problems onto neural networks. *Int. J. Neural. Syst.*, **1**:3–22, 1989.

[14] J. von Neumann. Some matrix-inequalities and metrization of matric-space. *Tomsk Univ. Rev.*, **1**:286–300, 1937.

[15] M. Ohlsson and H. Pi. A study of the mean field approach to knapsack problems. *Neur. Netw.*, **10**:263–271, 1997.

[16] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucl. Acid. Res.*, **28**:235–242, 2000.

[17] A. Marchler-Bauer, K. J. Addess, C. Chappey, L. Geer, T. Madej, Y. Matsuo, Y. Wang, and S. H. Bryant. MMDB: ENTREZ's 3d structure database. *Nucl. Acid. Res.*, **27**:240–243, 1999.