Calculating and scoring high quality multiple flexible protein structure alignments

David Ritchie

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Structural bioinformatics

Calculating and scoring high quality multiple flexible protein structure alignments

David W. Ritchie*

Inria Nancy – Grand Est, 615 Rue du Jardin Botanique, 54600 Villers-lès-Nancy, France

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received XXXX X, XXXX; Revised XXXX X, XXXX; Accepted XXXX X, XXXX

Abstract

Motivation: Calculating multiple protein structure alignments (MSAs) is important for understanding functional and evolutionary relationships between protein families, and for modeling protein structures by homology. While incorporating backbone flexibility promises to circumvent many of the limitations of rigid MSA algorithms, very few flexible MSA algorithms exist today. This article describes several novel improvements to the Kpax algorithm which allow high quality flexible MSAs to be calculated. This article also introduces a new Gaussian-based MSA quality measure called "M-score", which circumvents the pitfalls of RMSD-based quality measures.

Results: As well as calculating flexible MSAs, the new version of Kpax can also score MSAs from other aligners and from previously aligned reference datasets. Results are presented for a large-scale evaluation of the Homstrad, SABmark, and SISY benchmark sets using Kpax and Matt as examples of state-of-the-art flexible aligners and 3DCOMB as an example of a state-of-the-art rigid aligner. These results demonstrate the utility of the M-score as a measure of MSA quality and show that high quality MSAs may be achieved when structural flexibility is properly taken into account.

Availability: Kpax 5.0 may be downloaded for academic use at *http://kpax.loria.fr/*.

Contact: dave.ritchie@inria.fr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Calculating multiple structure alignments (MSAs) of proteins is an important step towards understanding the evolutionary and functional relationships between different proteins and protein families (Sierk and Kleywegt, 2004; Hasegawa and Holm, 2009). Calculating high quality structural alignments can be crucial for the accurate modelling of three-dimensional (3D) protein structures by homology (Madhusudhan *et al.*, 2009; Braberg *et al.*, 2012), and can improve the generation of MSA-based sequence profiles for protein fold prediction (Pei *et al.*, 2008; Ghouzam *et al.*, 2015). Many algorithms for calculating protein structure alignments (SAs) have been described and reviewed in recent years (Sierk and Kleywegt, 2004; Kolodny *et al.*, 2005; Hasegawa and Holm, 2009; Ma and Wang, 2014), and several of these can calculate MSAs. Some examples of multiple structural aligners are SSAP (Taylor, 1994), MUSTA (Leibowitz *et al.*, 2001), CE-MC (Guda *et al.*, 2004), MultiProt (Shatsky *et al.*, 2004b), POSA (Ye and Godzik, 2005), MAMMOTH-multi (Lupyan *et al.*, 2005), Vorolign (Birzele *et al.*, 2006) MUSTANG (Konagurthu *et al.*, 2006), PROMALS3D (Pei *et al.*, 2008), BLOMAPS (Wang and Zheng, 2009), SALIGN (Madhusudhan *et al.*, 2009), MISTRAL (Micheletti and Orland, 2009), 3DCOMB (Wang *et al.*, 2011), MAPSCI (Ilinkin *et al.*, 2010), Smolign (Sun *et al.*, 2012), msTALI (Shealy and Valafar, 2012), and mulPBA (Joseph *et al.*, 2012).

Most of the above approaches first use dynamic programming (DP) to align pairs of structures, and they then progressively merge the pair-wise alignments to make a MSA. A few approaches subsequently optimise the resulting MSA using distance-based (e.g. CE-MC and MAMMOTH-multi) or energy-based (e.g. MISTRAL) minimisation techniques. However, when aligning many structures, the progressive alignment approaches can become computationally expensive because they require an initial set of all-against-all pair-wise alignments to build a guide tree, which means that the computational cost scales as $O(N^2)$ in the number of structures, N. In order to avoid such computational costs, the MAPSCI and mulPBA algorithms build a multiple alignment with respect to a given "pivot" structure, which leads to a computational cost of

$O(N)$. But this requires prior knowledge of which structure should serve as the pivot. An effective alternative to conventional "row-first" multiple alignment algorithms is 3DCOMB (Wang *et al.*, 2011), which uses a "column-first" approach to identify "highly similar fragment blocks" to guide the calculation of MSAs. While 3DCOMB has been shown to give excellent MSAs (Wang *et al.*, 2011), the complexity of its column-first algorithm also appears to scale as $O(N^2)$ in the number of structures.

Because it is difficult to optimise simultaneously the number of aligned residues and their C_α root mean squared deviation (RMSD) (Zemla, 2003), different structure aligners make different number/RMSD trade-offs which can lead to producing locally different alignments, even within proteins that share common motifs. Thus, current SA algorithms can introduce errors or inconsistencies even in pair-wise alignments (Sadowski and Taylor, 2012), and these errors can accumulate considerably in MSA algorithms that progressively merge multiple pair-wise alignments. Furthermore, such effects can confound efforts to compare different MSA algorithms against reference SAs, because the programs used to calculate the reference SAs are themselves subject to the same number/RMSD trade-off problem.

Incorporating backbone flexibility offers one way to tackle such problems. Although some flexible SA algorithms have been described, such as FlexProt (Shatsky *et al.*, 2002, 2004a), FATCAT (Ye and Godzik, 2003), POSA (Ye and Godzik, 2005; Li *et al.*, 2014), FlexSnap (Salem *et al.*, 2010), Matt (Menke *et al.*, 2008), and RAPIDO (Mosca *et al.*, 2008), still only a very few algorithms can flexibly align multiple structures. Furthermore, other "flexible" aligners such as Vorolign (Birzele *et al.*, 2006) and GR-Align (Malod-Dognin and Pržulj, 2014) can align pairs of structures that cannot be completely superposed rigidly, but they do not perform superpositions.

To my knowledge, the only aligners that can calculate multiple superpositions flexibly are POSA (Ye and Godzik, 2005), Matt (Menke *et al.*, 2008), and Smoalign (Sun *et al.*, 2012). POSA essentially extends the pair-wise FATCAT approach to MSAs. Both POSA and Smolign calculate partial-order alignment graphs, i.e. lists of locally aligned segments which are common to all structures but which are not necessarily sequential in the primary protein sequences. In contrast, both Matt and Kpax calculate strictly sequential MSAs which do not allow non-sequential permutations of residues. Unfortunately, POSA is publicly available only as a web server (Li *et al.*, 2014), which makes it infeasible for large-scale studies, and Smolign is no longer available for download.

While the original Kpax pair-wise alignment algorithm is sufficiently fast to be able to search large 3D structural domain databases in just a few seconds, the first version was limited to performing only rigid-body superpositions of entire protein structures. This article describes several extensions to the earlier Kpax algorithm (Ritchie *et al.*, 2012) which allow pairs of proteins of arbitrary size (including multi-domain proteins) to be aligned flexibly, and which allow multiple proteins to be assembled into a MSA using a pivot-based approach.

It seems obvious that a pre-requisite for calculating high quality MSAs is to begin with an algorithm that can calculate high quality pair-wise SAs. However, there does not yet exist an accepted standard for assessing the quality of pair-wise alignments or MSAs. 3DCOMB (Wang *et al.*, 2011), calculates its MSA quality measure from the pair-wise TM-score (Zhang and Skolnick, 2005) taken over all fully aligned columns. On the other hand, for columns with an occupancy of 70% or more, mulPBA uses a quality measure based on a weighted average of the numbers of residues whose pair-wise distances fall within certain given distance ranges (Joseph *et al.*, 2012). This may be considered as an extension of the Local-Global Alignment (LGA) scoring method which calculates a weighted average over several number/RMSD bins of a pair-wise alignment (Zemla, 2003). More recently, the STOVCA program (Slater *et al.*, 2013) has been proposed to compare the quality of structural superpositions calculated by other pair-wise aligners, but this program is limited to evaluating pair-wise superpositions using its own internal pair-wise alignment algorithm. Collier *et al.* (2014) proposed a method for measuring SA quality based on information compression, but this approach is also limited to pair-wise alignments. Thus, there exists a need for a simple and general way to compare the quality of both the alignments and the corresponding superpositions of different MSA algorithms. To try to address the problem of comparing different SA algorithms, this article proposes a novel atomic Gaussian based MSA scoring function, called "M-score", which circumvents the number/RMSD trade-off problem.

In previous studies, Wang *et al.* (2011) showed that 3DCOMB gives highly competitive rigid MSAs compared to MAPSCI, MAMOTH, Matt, BLOMAPS, MUSTANG, and MultiProt when applied to the large Homstrad (Mizuguchi *et al.*, 1998; Stebbings and Mizuguchi, 2004) and SABmark (Van Walle *et al.*, 2005) benchmark sets, and Berbalk *et al.* (2009) showed that Matt gave the best overall performance on a sub-set of their challenging SISY benchmark. Therefore, a similar large-scale comparison using these datasets is presented here to compare Kpax with 3DCOMB as an example of a state-of-the-art rigid multiple aligner, and with Matt as the best available flexible multiple aligner.

2 Methods

2.1 Normalised Atomic Gaussian Overlap SAs

Kpax pair-wise SAs are calculated using two C_α-based scoring functions, both of which are based on Gaussian density distributions. In the main scoring function, each C_α atom is represented as a normalised 3D atomic Gaussian shape-density distribution

$$\phi(\underline{x}) = \left(\frac{1}{\pi\sigma^2}\right)^{3/4} e^{-r^2/2\sigma^2}, \tag{1}$$

where $r = |\underline{x}|$ is the distance from the nucleus and σ is the atomic radius. I put $\sigma = 1.4$ Å, which is slightly less than the van der Waals radius of 1.5 Å often used in molecular mechanics force fields. A similar 3D Gaussian representation has been shown to provided an effective model of molecular shape for small-molecule shape comparison and virtual drug screening (Grant *et al.*, 1996).

It is well known that Gaussian functions have special mathematical properties. For example, it has long been known that the overlap between a product of two Gaussian functions is also a Gaussian (see Equation 10 in Boys (1950)). In particular, it can be shown that the 3D overlap between two of the above normalised 3D Gaussian functions on atoms i and j located at $\underline{x}_i$ and $\underline{x}_j$ and separated by a distance $R_{ij} = |\underline{x}_i - \underline{x}_j|$ is given by

$$G_{ij} = \int \phi(\underline{x}_i)\phi(\underline{x}_j)\mathrm{d}\underline{x} = e^{-R_{ij}^2/4\sigma^2}. \tag{2}$$

This overlap expression has the nice property that it gives a value of one unit when two C_α atoms are perfectly superposed because $e^0 = 1$. Furthermore, when the distance between atoms i and j exceeds the sum, S, of their C_α radii, we have $S = 2\sigma$ and e^{-R^2/S^2} tends to zero very rapidly when $R > S$. Thus, while other Gaussian scoring functions have been used to provide probabilistic or "elastic" scoring functions in previous SA algorithms, notably STAMP (Russell and Barton, 1992) and DALI (Holm and Sander, 1993), it can be seen that the form of Equation 2 is rather special, and is certainly not arbitrary. Using the above *physical model of protein shape*, a SA between two proteins may be scored by summing pairs of Gaussian overlaps to give a global alignment score, or "G-score"

$$G = \sum_{i,j} \mu_{ij} G_{ij}, \tag{3}$$

where the coefficient $\mu_{ij} = 1$ when residue i of the first protein is aligned to residue j of the second protein, and zero otherwise.

However, the above scoring function requires that a pair-wise alignment and 3D superposition are already known, which is initially not the case. Therefore, in order to identify candidate aligned fragment pairs (AFPs) for least-squares fitting and superposition, Kpax calculates an initial pose-invariant similarity score for all possible residue pairs using a scoring function based on a sliding window of 7 residues on each chain (i.e. 6 C_α pairs, around the pair of interest), and using products of normalised Gaussian overlaps at each position, which I call "K-scores", as described previously (Ritchie *et al.*, 2012). Products, rather than sums, of local Gaussian overlap scores are used to calculate an initial alignment in order that a pair-wise K-score tends rapidly to zero whenever any of its component distances are large. Thus, only those residue pairs that share highly similar local environments will have large K-scores. Applying dynamic programming (DP) to a matrix of such K-scores, produces an initial seed alignment, or "K-alignment", from which runs of 4 or more pairs (AFPs) having large K-scores may be identified and used as 3D superposition seeds.

The K-alignment of a pair of similar protein domains typically contains up to about 10 AFPs, whereas a pair of large domains with many similar secondary structure elements may give up to around 50 AFPs. For each candidate 3D superposition seed proposed by a K-alignment AFP, an all-against all matrix of C_α G-scores (the "G-matrix") is calculated, from which a global alignment may be calculated by DP with no gap penalties. The residue pairs extracted from the resulting "G-alignment" are then used to define a new 3D least-squares fitting transformation, and the cycle is iterated until convergence (typically just 3 or 4 iterations are required). The G-alignment that gives the best global G-score is then retained as the the "optimal" global alignment. From my experiments with the examples presented here, using any value of σ that differs much from 1.4 Å generally gives poorer alignments (details not shown). Hence, putting $\sigma = 1.4$Å appears to be numerically "optimal" in practice, as well as theory.

2.2 Detecting Conserved Motifs using Tiled DP

Although Kpax uses the very fast and robust quaternion minimisation least squares fitting method of Liu *et al.* (2010), and only a relatively small number of least-squares fits are necessary, the overall computational cost of the above procedure scales as $O(N^2)$ in the number of residues to be aligned. This makes it is relatively expensive to align large multi-domain proteins by iterating on multiple candidate G-matrices. Furthermore, it is possible that the optimal DP trace-back from the K-matrix might not contain any 3D fitting seeds that correspond to an optimal 3D G-alignment. Therefore, when one or both proteins has more that 120 residues, a series of small "tiled" SAs is calculated, in which each protein is divided into overlapping strips of 72 residues, and a local SA is calculated between each strip of one protein and the whole of the other protein using DP on the tiles of the K and G matrices. For example, when aligning two proteins each having 144 residues, a total of 6 tiled alignments are calculated using the residue ranges (1:71,1:144), (37:108,1:144), (73:144, 1:144), (1:144,1:71), (1:144, 37:108), (1:144,73:144). The 3D transformation from the tile that gives the greatest G-score is retained and used in a final DP iteration on the the full G-matrix. This approach is easily generalised to the case where the number of residues is not a whole multiple of the tile size. For example, a protein of 200 residues would be divided into strips of (1:64), (65:136), (137:200), (1:28), (29:100), (101:172), and (173:200).

For large proteins, tiling reduces the computational cost from $O(FN^2)$ to $O(Tfn^2)$ where F is the number of AFPs in the global alignment, f is the average number of fragments in a tile, T is the number of tiles of size n, and where $n < N$ and $T << N$. Furthermore, tiling allows the possibility of finding good local 3D seeds in the K-alignment which might not appear in the (globally optimal) trace-back path of the full K-matrix. In other words, small conserved motifs may be detected in a tiled alignment whereas they might be completely missed in a global alignment.

Since it is not known in advance whether a given pair of proteins will align well globally or whether a tiled alignment should be applied, Kpax first attempts a global alignment and if the global alignment M-score (see below) is less than 0.5, it applies a tiled alignment in order to find the best seed for a global alignment.

2.3 Flexible Alignments using Multiple DP Boxes

Calculating flexible alignments may be considered as a natural extension of the above tiled approach, in which the tiles no longer have a pre-determined size. For Kpax flexible alignments, all residues are initially labelled as belonging to a default segment (Segment 0), and after the optimal rigid superposition has been calculated as described above, all pairs of residues that fall within a given distance threshold (3 Å, by default) are assigned to a new segment (Segment 1). The remaining Segment 0 residues of each structure are then considered as candidates for flexible refinement. More specifically, the first DP problem corresponds to the entire DP matrix, and consecutive runs of four or more residue pairs belonging to Segment 0 are identified and superposed by least-squares fitting. The initial DP matrix is then divided into three or more "boxes", where the first box spans the first and last residue pairs of segment 1, thus leaving one N-terminal and one C-terminal box of the DP matrix, each of which contains candidate residues for flexible refinement. It is also possible for one or more DP boxes to be found within the interior of a superposed segment (i.e. segments do not necessarily contain continuous runs of superposed residues). In any case, each new DP box may be considered as a new SA sub-problem, which may be treated exactly as described above. Thus, by maintaining a list of DP boxes to be processed on a stack, the flexible alignment procedure continues recursively, and it terminates when no more DP boxes remain in the stack.

The evolution of the DP matrix for the first few iterations of this procedure is illustrated schematically in Supplementary Figure 1. Each residue in the moving structure is transformed in 3D space at most two times: once during the rigid body superposition, and possibly once again if it belongs to a DP box that produces a superposition segment. Note, when superposing a moving segment to a rigid template structure, the coordinates of the neighbouring residues relative to the moving segment are taken into account whenever possible. Thus, the final flexibly aligned structure has few obvious gaps or tears in its backbone atom coordinates.

2.4 Pivot-First Method of Calculating MSAs

In Kpax, MSAs are calculated using a rigid pivot structure onto which the remaining structures are either rigidly or flexibly aligned and superposed. By default, in both rigid and flexible MSAs, each structure is used in turn as the pivot structure, and the MSA that yields the best overall M-score is retained as the pivot. Thus, for N structures, there will be N-1 alignments between the selected pivot and each of the remaining structures. A MSA may then be induced from the N-1 pair-wise alignments by introducing additional gaps as necessary to re-align the N-2 gapped pivot structures with the pivot of the first alignment. While this induced MSA (IMSA) is not necessarily optimal, it provides a fast way to identify highly conserved columns in the alignment. Any column of the IMSA which is occupied with 60% or more residues is noted as a highly conserved column.

The IMSA is used to provide a look-up table for building a final MSA. Kpax first sorts the N-1 other structures by their G-score similarity to the pivot. It then takes the most similar structure as a seed alignment, and it builds the final MSA by incrementally aligning the remaining N-2 structures in a "pile-up" manner onto this seed. Each in-coming conserved residue is biased to appear close to its position in the initial IMSA by calculating column-wise G-scores for a window of $\pm$ 16 residues around

the corresponding IMSA position. Thus, the alignment of conserved regions may be further optimised by DP, and in-coming non-conserved residues have the opportunity to be re-aligned against the preceding structures. Consequently, the Kpax MSA algorithm may be considered as a "pivot-first" hybrid between the row-first and column-first approaches.

Although it is conceivable that one could use a guide-tree to align multiple pairs of flexibly aligned structures, this could lead to many false AFPs and many tears in the moving structures. The above pile-up approach helps to ensure that the number of deformations to the flexible structures is small, and that each non-rigid structure is deformed only once. A pile-up approach also allows for easy parallel execution on multiple processors.

2.5 What is Wrong with RMSD?

There are two fundamental problems with RMSD as a quality measure. Firstly, it is non-linear in the number of distances concerned. Secondly, RMSDs can hide many sins. Consider one column of a MSA of 10 structures, in which 9 of the C_α atoms superpose perfectly, but one is displaced by 5 Å and therefore probably belongs to an adjacent column (recall that the distance between consecutive C_α backbone atoms is normally 3.8 Å). Suppose we calculate the RMSD with respect the average column coordinate (which for convenience can be assumed to be at the coordinate origin, and where the displacement is along one of the principal axes). This gives $((9*0.5^2+1*4.5^2)/10)^{1/2} = 1.50$ Å. Now consider one column of a MSA of 20 structures, with one C_α again displaced by 5 Å. This gives $((19*0.25^2+1*4.75^2)/20)^{1/2} = 1.08$ Å. In other words, increasing the number of rows has spread the displacement over more samples and hence apparently reduced the RMSD by about 30%.

But one should not forget that RMSD has units of "distance" / ("number-of-pairs")$^{1/2}$. So it would be more precise to label each RMSD with the number of pairs involved, e.g. $\text{RMSD}_{10} = 1.50$ Å, and $\text{RMSD}_{20} = 1.08$ Å. This reminds us that it is incorrect to compare or combine RMSD_{10} and RMSD_{20} directly because they were calculated using different distance scales. Consequently, a fairer way to compare these two cases is to extract the average deviations as $\text{RMSD}_{10}/\sqrt{10} = 0.47$ Å and $\text{RMSD}_{20}/\sqrt{20} = 0.24$ Å. But this obviously amounts to not using RMSD in the first place. If we wish to score and compare SAs reliably, it seems much more desirable to call a gap for the displaced residue, and to award scores of 9 out of 10 (say) for the first example and 19 out of 20 for second. This is what the M-score aims to achieve.

2.6 MSA Quality Assessment Using the M-score

The M-score extends the pair-wise C_α Gaussian overlap model to MSAs of arbitrary dimensions. Firstly, let N be the total number of rows (i.e. structures) in a MSA, C be the total number of columns, T be the total number of residues over all structures, and L be the number of residues in the longest structure. Thus, an "ideal" MSA may be considered to have at least one row of L residues and the remaining $(T-L)$ residues from the other structures would each align perfectly with one of the residues of the longest structure. This implies than many of the MSA columns must contain 2 or more completely overlapping C_α atoms.

A simple way to estimate this mutual overlap is to calculate an average coordinate for each column, and then to calculate the degree of overlap between this average coordinate and each of the column members. Using these ideas, the column overlap score C_j with respect to the average 3D coordinate, $\underline{c}_j$, for the j^{th} column may be defined as:

$$C_j = \sum_{i=1}^{N} \mu_{ij} e^{-(\underline{x}_{ij}-\underline{c}_j)^2/4\sigma^2}, \tag{4}$$

where μ_{ij} is either 1 or 0 according to whether the position at row i and column j contains a residue or a gap, respectively, and $\underline{x}_{ij}$ is the coordinate of the C_α atom at row i and columns j. Note that one wrongly placed C_α atom will reduce this score by at most one unit. On the other hand, if all of the C_α coordinates in column j coincide with the average coordinate, then C_j will be numerically equal to the number of residues in that column. Thus, a global overlap score may be calculated for the MSA as

$$M' = (T-L) + C - \sum_{j=1}^{C} \max(C_j, 1), \tag{5}$$

where the max function ensures that at least one unit is subtracted for each column, and thus deals with columns that contain only a few very poorly superposed C_α atoms, which may occur at the termini of an alignment or in the loop regions around an AFP. If all of the C_α atoms in each column coincide completely with the average column coordinate, then $C = L$ and $\sum_{j=1}^{C} \max(C_j, 1) = T$, and hence $M' = 0$. On the other hand, in the worst possible case of no aligned residue pairs, we have $\underline{c}_j = \underline{x}_{ij}$ for all i and hence $C_j = 1$ for all j. Therefore, for any MSA, M' will have the range $0 \leq M' \leq (T-L)$.

However, because it is more intuitive to let large scores correspond to good alignments, the above expression may be scaled and shifted using $M = 1 - M'/(T-L)$ to give

$$M = \frac{\sum_{j=1}^{C} \max(C_j, 1) - C}{(T-L)}. \tag{6}$$

With this scoring function, a perfect multiple alignment of any number of structures (e.g. between any protein and one or more identical copies of itself) would give a score of unity. Furthermore, for pair-wise alignments, the factor $(T-L)$ will be equal to the number of residues in the smaller structure, and a score of unity will correspond to the complete overlap of all C_α atoms of the smaller structure with some subset of the C_α atoms of the larger structure. If desired, the M-score may be multipled by $(T-L)\sigma^3$ to give an absolute quality measure in Å^3 units.

It may be noted that the M-score contains no gap penalties and has no distance bins or distance cut-offs. Its only potentially adjustable parameter is the C_α radius, which in practice should be fixed at 1.4 Å. Furthermore, by construction, the only way to maximise M is to have as few columns as possible of which some subset must each contain two or more maximally overlapping C_α atoms. This means that the M-score smoothly circumvents the number/RMSD trade-off problem.

In cases where two or more proteins share only a small motif, or where they have long unaligned N or C termini, it can be useful to restrict the M-score to a given block of the MSA. For example, to score the alignment of a motif between columns J and K, it is straight-forward to calculate

$$M_{JK} = \frac{\sum_{j=J}^{K} \max(C_j, 1) - C_{JK}}{(T_{JK} - L_{JK})}, \tag{7}$$

where C_{JK} and T_{JK} denote the number of columns and total number of residues between columns J and K, respectively, and L_{JK} is the longest chain within that block.

2.7 Comparing Pair-wise SA Algorithms

As a first but important comparison, Table 1 shows the flexible alignment results calculated by Kpax, Matt, and FATCAT for for the 10 "difficult" pairs of structures from Fischer *et al.* (1996). Supplementary Table 1 shows the results for the corresponding rigid alignment calculated by Kpax, Matt, FATCAT, 3DCOMB, TMalign, MUSTANG, CE, and DALI. Comparison of the numbers of aligned residues and the RMSDs with the corresponding M-score in these tables shows that the M-score captures rather well the aim to reward a high number of aligned pairs with low RMSD. For example, in the first row of Table 1 (1fxi/1ubq), Kpax aligns 64 pairs with a RMSD

of 2.0 and M=0.684, whereas FATCAT aligns 63 pairs but with a worse RMSD of 3.0Å and M=0.492. On the other hand, Matt aligns a smaller number of pairs with a lower RMSD (49 pairs for 1.0Å) which yields M=0.607. Similar observations may be made for the other example pairs in both tables. In general, it can be seen that Matt tends to align fewer pairs for a lower RMSD, whereas FATCAT aligns more pairs for a higher RMSD. Of the rigid aligners, it can be seen that MUSTANG gives the poorest rigid alignment figures with an average of 125 pairs at 3.2Å yielding M=0.398. More importantly, it can be seen from these two tables that for a comparable number of aligned pairs, a lower RMSD yields a higher M-score, and for similar RMSDs a larger number of aligned pairs yields a higher M-score. This shows that the M-score captures rather well the simultaneous objective of finding a large number of aligned pairs with a low RMSD. Overall, according to the M-scores, Table 1 shows that Kpax produces better flexible alignments than both Matt and FATCAT, while Supplementary Table 1 shows that Kpax and 3DCOMB produce amongst the best rigid alignments.

Table 1. Flexible SA comparison between Kpax, Matt and FATCAT for the 10 "difficult" pairs from Fischer et al. (1996).*

PDBs†	Kpax-flex					Matt-flex				FATCAT-flex				
	S	A	R	I	M	A	R	I	M	S	A	R	I	M
1fxi/1ubq	4	64	1.6	4	**0.734**	49	1.0	5	0.607	1	63	3.0	9	0.492
1ten/3hhr	1	86	1.5	17	**0.854**	73	0.7	17	0.797	1	87	1.9	16	0.794
3hla/2rhe	4	81	1.6	11	**0.701**	58	1.0	9	0.551	3	91	3.2	10	0.497
2aza/1paz	3	88	1.5	18	**0.652**	61	0.8	11	0.489	1	89	2.8	15	0.472
1cew/1mol	3	81	1.0	16	**0.811**	69	0.6	13	0.715	1	83	2.4	14	0.641
1cid/2rhe	3	100	1.9	14	**0.739**	77	1.2	11	0.622	1	103	2.8	13	0.581
1crl/1ede	11	238	2.2	22	**0.606**	153	0.9	9	0.470	7	280	3.1	21	0.490
2sim/1nsb	12	286	1.8	25	**0.632**	218	1.1	22	0.534	1	305	3.1	30	0.429
1bge/2gmf	6	105	1.8	18	**0.740**	59	0.3	11	0.486	1	116	3.0	18	0.488
1tie/4fgf	4	117	1.8	11	**0.800**	69	0.6	7	0.543	1	120	3.2	11	0.589
Mean	5	125	1.7	16	**0.727**	89	0.8	12	0.581	2	134	2.8	16	0.547

* Listed are the numbers of segments (S; not available for Matt), aligned pairs (A), aligned residue identities (I), the alignment RMSD in Å units (R), and the M-score (M). The Matt alignments were made using Matt-1.00 (Menke *et al.*, 2008). The FATCAT alignments were made using jFATCAT (Prlić *et al.*, 2010). † The A chain is used in all PDB structures except for 3hhr (chain B), 1cew (chain I), 1nsb (chain B), and 1bge (chain B).

2.8 Flexibly Aligning Multi-Domain Proteins

To my knowledge, the FlexProt algorithm of Shatsky *et al.* (2002, 2004a) is the first structure aligner to have been developed to handle structural flexibility in large multi-domain proteins which cannot be rigidly superposed. According to the M-scores in Table 2, Kpax often produces better flexible alignments for the 18 structure pairs of Shatsky *et al.* (average Kpax M-score: 0.794) than both Matt (M=0.732) and FATCAT (M=0.689). Although the average numbers of aligned residues and residue identities are similar for Kpax and FATCAT (245 and 129 for Kpax compared to 254 and 132 for FATCAT, respectively), the lower average RMSD for Kpax (1.4 Å compared to 2.3 Å) accounts for its greater average M-score. As in Table 1, Matt produces some very low RMSD alignments, but this is at the expense of finding fewer aligned pairs and pair-wise identities. It may be noted that Matt finds sub-optimal alignments in at least three cases (2bbm/1cll, 2bbm/1top, and 1mcp/1tcr) in which the M-scores and numbers of aligned residues are significantly lower than those for Kpax and FATCAT. In contrast, Kpax finds only one sub-optimal alignment (1a21/1hwg) relative to the Matt and FATCAT alignments. Although FATCAT finds respectable alignments for all of the examples in Tables 1 and 2, in each case either Kpax or Matt produces a better alignment according to the M-score.

Table 2. Flexible SA comparison between Kpax, Matt, and FATCAT for 18 pairs from Shatsky et al. (2002, 2004a).*

PDBs†	Kpax-flex					Matt-flex				FATCAT-flex				
	S	A	R	I	M	A	R	I	M	S	A	R	I	M
1wdn/1ggg	2	219	0.7	214	**0.970**	214	0.4	208	0.963	1	220	1.0	220	0.948
2bbm/1cll	2	140	1.8	131	**0.801**	65	0.4	31	0.448	2	144	2.1	141	0.774
2bbm/1top	7	143	1.6	65	**0.837**	65	0.4	24	0.436	4	147	2.3	77	0.743
1ake/2ak3	4	206	1.2	83	0.880	201	0.5	81	**0.925**	3	206	1.7	84	0.819
2ak3/1uke	6	184	1.3	49	**0.867**	168	0.5	52	0.859	1	188	2.5	54	0.654
1mcp/4fab	2	217	1.2	176	0.921	213	0.6	174	**0.952**	2	217	1.3	176	0.908
1mcp/1tcr	5	212	1.6	53	**0.841**	95	0.7	28	0.419	2	213	2.2	52	0.760
1lfh/1lfg	2	687	0.8	684	0.960	675	0.2	674	**0.973**	3	689	1.0	687	0.945
1tfd/1lfh	3	289	1.2	182	**0.915**	229	0.5	108	0.769	3	291	1.4	187	0.889
1qf6/1adj	7	349	1.8	59	**0.696**	299	0.8	57	0.685	2	353	2.4	61	0.589
2clr/3fru	8	264	1.3	70	**0.888**	233	0.5	66	0.853	1	263	3.0	68	0.563
1fmk/1qcf	9	431	1.0	256	0.933	418	0.4	259	**0.945**	1	434	2.3	265	0.737
1fmk/1tki	8	240	1.7	39	**0.640**	209	0.6	42	0.638	1	242	3.2	42	0.438
1a21/1hwg	2	89	1.9	18	0.392	120	0.9	17	**0.599**	2	167	2.9	20	0.490
1ntr/1dc7	8	122	1.3	122	**0.893**	90	0.8	90	0.695	1	122	3.1	122	0.565
1dce/1a17	3	103	2.3	9	0.507	117	0.6	16	**0.722**	4	150	2.7	14	0.631
2nac/1psd	4	306	1.3	80	0.736	281	0.6	76	**0.737**	1	311	3.2	76	0.441
1chm/1xgs	3	206	1.4	29	**0.622**	171	0.7	26	0.565	1	214	2.4	28	0.509
Mean	5	245	1.4	129	**0.794**	215	0.6	113	0.732	2	254	2.3	132	0.689
Time/s			7.6					55.6					92.5	

*Column heading are the same as in Table 1. †The A chain in all PDB structures is used here except for PDB code 1mcp (chain L).

2.9 Two Flexible MSA Examples

Figure 1 shows two examples of the rigid and flexible MSA superpositions produced by Kpax using the SABmark superfamily group104 ("sup-group104") and twilight-zone group202 ("twi-group202"). In this figure, the structures are coloured using a simple but informative "rainbow" colour-coding scheme, in which the residues in consecutive columns of a MSA are coloured in consecutive colours of the rainbow. These examples demonstrate the benefit of using each member structure as a candidate rigid pivot structure. For sup-group104 (consisting of the four SCOP structures: d1lgpa_, d1dmza_, d1g6ga_, and d1gxca_), the best rigid MSA pivot is found to be d1dmza_, giving M=0.626 and 68 core columns, while the worst pivot (d1lgpa_) gives M=0.541 and 64 core columns. However, for a flexible MSA with d1lgpa_ as pivot, the above quality measures increase to M=0.779 and 97 core columns. On the other hand, the best pivot changes to d1g6ga_, giving M=0.784 and 98 core columns. This shows that the best pivot for a flexible MSA cannot necessarily be predicted from rigid pair-wise SAs.

While Figure 1 suggests that calculating a MSA for sup-group104 should be relatively easy, it also shows that aligning all 3 members of twi-group202 (SCOP codes d1l2wi_, d1k8kd1, and d1jyoa_) is much more challenging. In this case, the rigid Kpax MSA gives only 14 core columns and M=0.285. In contrast, the Kpax flexible MSA (lower right in Figure 1) gives 51 core columns and M=0.750. This demonstrates the improvement that may be achieved on going from a rigid to a flexible MSA.

2.10 The Homstrad, SABmark, and SISY Benchmarks

The version of Homstrad used here (October 2014) consists of 1,032 families of homologous protein families containing a total of 3,458 structures. Each Homstrad family contains from 2 to 25 members, where no two members of any family have more than 90% sequence identity (Stebbings and Mizuguchi, 2004). The Homstrad SAs were calculated using STAMP, MNYFIT (Sutcliffe *et al.*, 1987), and COMPARER (Sali and Blundell, 1990).

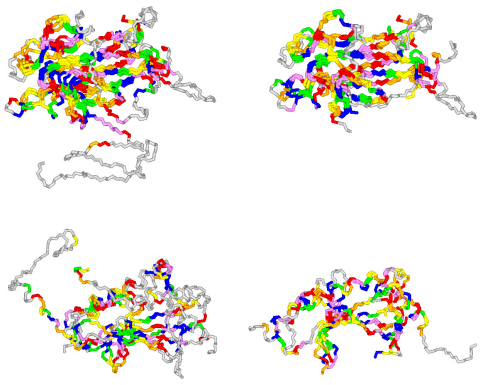

Fig. 1. Kpax rigid (left) and flexible (right) MSAs for SABmark superfamily group104 (top row) and twighlight zone group202 (bottom row), drawn using scripts generated by Kpax. The two groups of superpositions are not shown at the same scale.

The SABmark dataset (version 1.65) contains multiple alignments for 3,320 structures belonging of 425 superfamilies with low sequence identity and 1,740 structures belonging to 209 "twighlight zone" families with very low sequence identity. The SABmark dataset has only sequence-based pair-wise reference alignments. It has been shown that these alignments contain mutual inconsistencies (Edgar, 2010), hence it impossible to reconstruct MSAs from the reference pairs. Consequently, it is not possible to calculate M-scores or core columns for SABmark.

The SISY datasets were collected by Berbalk *et al.* (2009), who calculated the SISY reference alignments using CE and DALI. SISY-Families consists of 106 families of single chain SCOP domains with less than 40% sequence identity. SISY-pairs consists of 130 protein pairs with low sequence similarity drawn from the SISYPHUS database (Mayr *et al.*, 2007). SISY-RIPC (repetitions, insertions, permutations, and conformational variability) consists of 23 pairs which Berbalk *et al.* report to be difficult to align. Many of the SISY reference alignments are given not as full alignments but as "trimmed" alignments, consisting of a small list of highly conserved reference pairs or MSA columns. Several of the examples in the SISY-Pairs and SISY-RIPC datasets contain non-sequential alignments or small motifs within large unrelated folds which can be difficult to detect.

As noted by Berbalk *et al.*, some aligners can fail when presented with a PDB file that contains more than one structure, or when a PDB file contains SEQRES records that differ from the ATOM records. Therefore, all SISY structures involving more than one chain per PDB file were excluded, and the remaining PDB files were cleaned by running Kpax with the "-preprocess" option to change HETERO amino acid records to ATOM records, and to remove all other HETERO and SEQRES records. After cleaning, 3DCOMB was found to fail with a memory fault on 2 of the SISY examples and Matt gave a memory fault or failed to terminate on 6 examples. Excluding the above cases left a total of 77 SISY-Families, 111 SISY-Pairs, and 23 SISY-RIPC examples. Applying the same cleaning procedure to the Homstrad dataset detected 7 PDB files with duplicate or corrupted PDB chains, and these were also excluded. Additionally, 5 SABmark superfamily groups and 4 SABmark twighlight zone groups were eliminated because their reference alignments contained no pairs.

2.11 MSA Benchmark Quality Comparison

Table 3 presents results for the alignments calculated by Kpax, Matt, and 3DCOMB for the above benchmark datasets. The Homstrad dataset provides superposed structures and a PIR-format MSA alignment file. Hence it is possible to calculate several common quality measures such as the number of fully aligned core columns ("#Core" in Table 3), the number of aligned pairs ("#Pair"), the number of aligned residue identities ("#Identity"), and the average RMSD over all aligned pairs. Additionally, since such global measures do not take into account local alignment quality, it is useful calculate the number of badly aligned pairs, and the number of superposed fragment pairs ("#SFP"), where a SFP is taken to be a run of 3 or more aligned pairs in which the distance between each pair does not exceed 3.5Å, and where any pair-wise distance that exceeds 8.0 Å is counted as a distance violation ("#Viol"). The columns labeled "M_{1C}" and "M_{JK}" show the M-scores calculated for the full and trimmed MSAs, respectively, where M_{1C} ranges over all MSA columns and M_{JK} is calculated between the columns containing the first and last pair of aligned residues. The values shown for the number of pairs, identities, distance violations, SFPs, and RMSD values are normalised by the number of pairs of structures involved (i.e. for N structures, the normalisation factor is $N(N-1)/2$). Supplementary Figures 2–10 show more detailed "box-whisker" plots (Spitzer *et al.*, 2014) of the results in Table 3. Spread-sheets of numerical results are provided as Supplementary Files.

For this table, the RMSD values for the SABmark reference alignments were calculated using the pair-wise reference alignments to superpose the corresponding structures. The RMSD values for the SISY reference alignments were calculated by iteratively superposing the SISY structures onto an average coordinate for each reference column. Because several of the SISY reference alignments contain non-sequential SAs that cannot be fully aligned by sequential aligners, these datasets are grouped separately in Table 3 (denoted as "seq" and "nonseq", accordingly) because these sub-sets show rather different statistics.

Several revealing observations may be made from Table 3. Firstly, it can seen that some of the reference alignments contain many distance violations, notable the Homstrad and SISY-RIPC-seq datasets. As observed previously for multiple sequence alignments (Edgar, 2010), this means that structural reference alignments sometimes contain significant errors, and it is therefore dangerous to treat even well curated reference alignments as a gold standard against with different algorithms should be compared. In particular, for Homstrad, it can be seen that 3DCOMB, Matt, and Kpax all achieve higher M-scores with lower numbers of cores columns and residue identities than the Homstrad reference alignments, while the Homstrad reference alignments give a higher average RMSD and more distance violations than these three methods. For the SISY-RIPC-seq reference set, most of the distance violations come from just 3 pairs, with the pair d2bbma_/4clna_ having 114 distance violations, which is clearly incorrect, and the pairs d1ggga_/d1wdna_ and d1l5ba_/d1l5ea_ having 25 and 19 violations, respectively, which make them of doubtful quality.

The large numbers of core columns calculated by 3DCOMB is impressive, and no doubt reflects 3DCOMB's strategy of aiming to produce many filled columns (Wang *et al.*, 2011), but this is often at the expense of higher RMSDs and more distance violations than Kpax and Matt. In contrast, Matt-flex gives amongst the lowest RMSDs and lowest numbers of distance violations on all datasets. However, with the exception of some of the trimmed SISY reference alignments, this is always at the expense of finding fewer aligned pairs and residue identities than the other alignments.

It can be seen from Table 3 that Kpax-flex produces the largest average M-score while also finding the greatest number of residue identities in all datasets except for Homstrad and the small SISY-RIPC-nonseq group. Kpax-flex also finds the largest average trimmed M-score for the Homstrad and SABmark datasets, and it is interesting to note that average trimmed M-score for Kpax-flex is larger than that of the reference alignments for the SISY-RIPC-seq group. Furthermore, Kpax, Matt, and 3DCOMB all find more aligned pairs, residue identities, and AFPs in the SISY datasets than are present in the SISY reference alignments. While the trimmed SISY reference alignments have high M_{JK} values, which is to be expected, the rigid alignments calculated by Kpax and Matt for the full structures

Table 3. Kpax, Matt, and 3DCOMB performance on the Homstrad, SABmark, and SISY datasets.

	#Core	RMSD	#Viol	M_{1C}	M_{JK}	#Pair	#Identity	#SFP
Homstrad (1025 groups, 3.3 structures per group)								
Kpax-flex	195.1	1.26	0.2	**0.860**	**0.871**	201.6	80.2	9.0
Matt-flex	172.8	0.53	0.0	0.771	0.824	172.8	73.7	8.4
Kpax-rigid	190.3	1.73	0.4	0.769	0.783	196.8	78.3	8.7
Matt-rigid	193.5	1.91	1.5	0.746	0.762	197.9	79.2	8.3
3DCOMB	197.0	1.96	2.3	0.765	0.778	202.1	79.6	8.3
Reference	199.6	3.11	7.6	0.714	0.722	205.3	82.3	7.9
SABmark-sup (420 groups, 7.8 structures per group)								
Kpax-flex	96.3	2.79	1.1	**0.754**	**0.765**	127.8	28.1	10.4
Matt-flex	76.9	1.10	0.0	0.469	0.615	76.9	19.7	7.0
Kpax-rigid	93.3	3.93	1.8	0.637	0.649	122.7	27.4	8.9
Matt-rigid	103.6	4.50	4.0	0.571	0.589	121.7	26.9	8.1
3DCOMB	106.0	4.40	3.1	0.610	0.629	128.4	28.0	8.3
Reference	–	2.03	0.1	–	–	96.8	23.9	6.2
SABmark-twi (205, groups, 8.4 structures per group)								
Kpax-flex	56.2	3.36	1.5	**0.657**	**0.674**	93.2	11.8	9.4
Matt-flex	42.7	1.19	0.0	0.320	0.574	42.7	6.6	5.2
Kpax-rigid	56.2	4.87	2.2	0.508	0.524	88.0	11.3	7.1
Matt-rigid	66.0	5.24	5.6	0.411	0.436	83.0	10.6	6.2
3DCOMB	70.4	5.39	3.4	0.472	0.500	94.5	11.7	6.8
Reference	–	2.51	0.3	–	–	67.9	9.7	5.5
SISY-Families-seq (69 groups, 8.5 structures per group)								
Kpax-flex	124.0	2.78	1.6	**0.758**	0.772	176.6	54.6	12.5
Matt-flex	100.7	1.09	0.0	0.466	0.676	100.7	33.9	8.1
Kpax-rigid	113.0	4.13	2.5	0.638	0.654	163.5	50.9	10.5
Matt-rigid	127.7	4.40	11.8	0.548	0.571	158.3	46.0	9.0
3DCOMB	130.9	4.76	5.4	0.602	0.623	170.9	51.9	9.4
Reference	96.0	1.97	0.5	0.402	**0.820**	96.0	32.8	6.3
SISY-Pairs-seq (95 groups, 2 structures per group)								
Kpax-flex	149.4	1.26	0.0	**0.669**	0.733	149.4	26.7	10.5
Matt-flex	115.8	0.51	0.0	0.581	0.71	115.8	24.1	8.9
Kpax-rigid	137.8	1.71	0.0	0.524	0.596	137.8	24.7	8.6
Matt-rigid	139.1	1.93	2.7	0.493	0.567	139.1	26.0	7.9
3DCOMB	152.8	2.26	6.8	0.525	0.574	152.8	25.8	8.6
Reference	77.9	2.45	2.0	0.373	**0.736**	89.9	19.9	6.0
SISY-RIPC-seq (17 groups, 2 structures per group)								
Kpax-flex	145.2	1.29	0.0	**0.651**	**0.718**	145.2	47.0	9.4
Matt-flex	104.1	0.45	0.0	0.530	0.689	104.1	33.9	7.8
Kpax-rigid	115.5	1.61	0.0	0.453	0.556	115.5	30.2	6.9
Matt-rigid	129.0	2.30	4.1	0.379	0.468	129.0	36.5	7.8
3DCOMB	140.7	2.38	6.9	0.450	0.530	140.7	31.8	8.0
Reference	32.7	3.78	9.8	0.043	0.666	32.7	30.3	0.9
SISY-Families-nonseq (8 groups, 13.6 structures per group)								
Kpax-flex	44.9	4.34	3.1	**0.603**	0.613	96.5	17.3	9.5
Matt-flex	35.8	1.37	0.0	0.191	0.513	35.8	7.2	4.5
Kpax-rigid	41.9	6.06	3.0	0.499	0.523	95.1	18.6	8.7
Matt-rigid	60.5	7.60	8.3	0.371	0.393	91.5	16.7	7.1
3DCOMB	66.9	5.84	5.8	0.424	0.441	101.6	19.0	7.4
Reference	55.1	2.58	1.1	0.253	**0.742**	55.1	12.8	5.1
SISY-Pairs-nonseq (16 groups, 2 structures per group)								
Kpax-flex	90.9	1.49	0.0	**0.479**	0.621	90.9	13.9	7.8
Matt-flex	62.1	0.62	0.0	0.374	0.601	62.1	10.0	6.4
Kpax-rigid	81.9	1.86	0.0	0.351	0.516	81.9	12.1	5.7
Matt-rigid	82.9	2.07	1.2	0.328	0.489	82.9	11.0	5.8
3DCOMB	96.8	2.48	5.6	0.357	0.529	96.8	13.2	6.1
Reference	59.2	2.37	0.2	0.269	**0.715**	59.2	10.3	5.6
SISY-RIPC-nonseq (6 groups, 2 structures per group)								
Kpax-flex	101.0	1.27	0.0	**0.502**	0.707	101.0	16.2	8.8
Matt-flex	68.3	0.59	0.0	0.377	0.745	68.3	15.3	6.0
Kpax-rigid	91.2	1.84	0.0	0.367	0.572	91.2	16.2	7.0
Matt-rigid	82.5	2.28	1.7	0.283	0.517	82.5	16.7	4.7
3DCOMB	101.7	2.35	4.5	0.371	0.567	101.7	16.7	6.8
Reference	32.5	1.48	0.0	0.228	**0.858**	32.5	9.5	2.7

from the two SISY-Pairs datasets give lower RMSDs than the RMSDs of the SISY reference alignments. Taken together, these observations indicate that a high proportion of the SISY reference alignments contain sub-optimal alignments compared to those calculated by Kpax, Matt, and 3DCOMB. The box-whisker plots in Supplementary Figures 5–10 further support this observation.

Apart from the surprisingly high number of residue identities in the Homstrad reference alignments, the Kpax flexible alignments contain the greatest number of aligned identities in all datasets except for the SISY-Families-nonseq and SISY-RIPC-nonseq groups. Additionally, the final column of Table 3 shows that the flexible Kpax alignments contain the largest average number of SFPs in all datasets. While it might be argued that flexible Kpax is "over-fitting" the 3D structures, it should be noted that the Kpax rigid alignments also contain the most SFPs for all datasets except for the SISY-RIPC-seq and SISY-Pairs-nonseq groups. Because it would seem to be very difficult to match runs of 3 or more low RMSD pairs purely by chance, these results strongly suggest that the Kpax alignments contain more conserved structural regions than the 3DCOMB and Matt alignments, and indeed more than many of the reference alignments. More certainly, the results in Table 3 show that high M-scores are highly correlated with high numbers of SFPs, core residues, aligned pairs, and low RMSDs. Therefore, these results demonstrate the utility of the M-score as a measure of MSA quality. Nonetheless, since current high quality alignments can still contain distance violations, a useful future extension could be to add a term to penalise such occurrences.

It is worth mentioning that Kpax is considerably faster than 3DCOMB and Matt. Aligning the 77 SISY-Families MSAs using Kpax takes around 13 minutes for rigid alignments and 17 minutes for flexible alignments on a workstation with eight E5410 2.3 GHz processors and 12 Gb memory. In comparison, 3DCOMB takes 49 minutes for its rigid alignments and Matt takes 200 minutes for both rigid and flexible alignments.

3 Conclusions

This article has described a novel approach for calculating flexible protein SAs by recursively dividing a DP scoring matrix into multiple boxes. Each box of the DP matrix corresponds to a rigid SA sub-problem. This allows flexible SAs to be calculated almost as easily as rigid alignments. This approach is easily extended to calculating flexible MSAs by flexibly aligning each mobile structure onto an automatically selected rigid pivot.

This article has also described the M-score, a novel Gaussian-based measure of MSA quality. The results presented for three large benchmark datasets show that high M-scores correspond to SAs with high numbers of aligned pairs with low RMSD. These results show that Kpax, Matt, and 3DCOMB generally produce SAs with more aligned pairs, residue identities, and aligned fragments than are present in the Homstrad, SABmark, and SISY reference alignments. It has also been shown that the rigid SAs from Matt and 3DCOMB, as well as several reference datasets, can contain a high proportion of pair-wise distances that exceed 8 Å, indicating the presence of significant alignment errors. While the flexible SAs produced by Matt are free of distance violations, these SAs are often much shorter than those produced by Kpax. In contrast, the full-length Kpax flexible SAs contain more residue identities and structurally aligned fragments than the Matt SAs, while still showing few distance violations and reduced RMSDs. Overall, these results show that high quality SAs may be achieved when structural flexibility is properly taken into account, and they support the utility of the M-score as a measure of MSA quality.

Funding

This work was funded by Inria Nancy – Grand Est and by the Agence Nationale de la Recherche, grant reference ANR-11-MONU-006-02.

References

Berbalk, C., Schwaiger, C. S., and Lackner, P. (2009). Accuracy analysis of multiple structure alignments. *Protein Science*, **18**, 2027–2035.

Birzele, F., Gewehr, J. E., Csaba, G., and Zimmer, R. (2006). Vorolign – fast structural alignment using Voronoi contacts. *Bioinformatics*, **23**, e205–e211.

Boys, S. F. (1950). Electronic wave functions I. A general method of calculation for the stationary states of any molecular system. *Proc. Roy. Soc.*, **A200**, 542–554.

Braberg, H., Webb, B. M., Tjioe, E., Pieper, U., Sali, A., and Madhusudhan, M. S. (2012). SALIGN: a web server for alignment of multiple proteins and structures. *Bioinformatics*, **28**, 2072–2073.

Collier, J. H., Allison, L., Lesk, A. M., Garcia de la Banda, M., and Konagurthu, A. S. (2014). A new statistical framework to assess structural alignment quality using information compression. *Bioinformatics*, **30**, i512–i518.

Edgar, R. C. (2010). Quality measures for protein alignment benchmarks. *Nucleic Acids Research*, **7**, 2145–2153.

Fischer, D., Elofsson, A., Rice, D., and Eisenberg, D. (1996). Assessing the performance of fold recognition methods by means of a comprehensive benchmark. In *Proceedings of the 1st Pacific Symposium on Biocomputing*, pages 300–318, Singapore. World Scientific Publishing Co.

Ghouzam, Y., Postic, G., de Brevern, A. G., and Gelly, J.-C. (2015). Improving protein fold recognition with hybrid profiles combining sequence and structure evolution. *Bioinformatics*, **doi**, 10.1093/bioinformatics/btv462.

Grant, J. A., Gallardo, M. A., and Pickup, B. T. (1996). A fast method of molecular shape comparison: A simple application of a Gaussian description of molecular shape. *Journal of Computational Chemistry*, **17**(14), 1653–1666.

Guda, C., Lu, S., Sceeff, E. D., Bourne, P. E., and Shindyalov, E. N. (2004). CE-MC: a multiple protein structure alignment server. *Nucleic Acids Research*, **32**, W100–W103.

Hasegawa, H. and Holm, L. (2009). Advances and pitfalls of protein structure alignment. *Current Opinion in Structural Biology*, **19**, 341–348.

Holm, L. and Sander, C. (1993). Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, **233**, 123–138.

Ilinkin, I., Ye, J., and Janardan, R. (2010). Multiple structure alignment and consensus identification for proteins. *BMC Bioinformatics*, **11**, 71.

Joseph, A. P., Srinivasan, N., and de Brevern, A. G. (2012). Progressive structure-based alignment of homologous proteins: Adopting sequence comparison strategies. *Biochimie*, **94**, 2025–2034.

Kolodny, R., Koehl, P., and Levitt, M. (2005). Comprehensive evaluation of protein structure alignment methods: Scoring by geometric measures. *Journal of Molecular Biology*, **346**(4), 1173–1188.

Konagurthu, A. S., Whisstock, J. C., Stuckley, P. J., and Lesk, A. M. (2006). MUSTANG: a multiple structural alignment algorithm. *Proteins: Structure, Function, Bioinformatics*, **64**, 559–574.

Leibowitz, N., Nussinov, R., and Wolfson, H. J. (2001). MUSTA – a general, efficient, automated method for multiple structure alignment and detection of common motifs: Application to proteins. *Journal of Computational Biology*, **8**, 93–121.

Li, Z., Natarajan, P., Ye, Y., Hrabe, T., and Godzik, A. (2014). POSA: a user-driven, interactive multiple protein structure alignment server. *Nucleic Acids Research*, **42**, W240–W245.

Liu, P., Agrafiotis, D. K., and Theobald, D. L. (2010). Fast determination of the optimal rotational matrix for macromolecular superpositions. *Journal of Computational Chemistry*, **31**, 1561–1563.

Lupyan, D., Leo-Macias, A., and Ortiz, A. R. (2005). A new progressive-iterative algorithm for multiple structure alignment. *Bioinformatics*, **21**, 3255–3263.

Ma, J. and Wang, S. (2014). Algorithms, applications, and challenges of protein structure alignment. *Advances in Protein Chemistry and Structural Biology*, **94**, 121–175.

Madhusudhan, M. S., Webb, B. M., Marti-Renom, M. A., Eswar, N., and Sali, A. (2009). Alignment of multiple protein structures based on sequence and structure features. *Protein Engineering, Design & Selection*, **32**, W100–W103.

Malod-Dognin, N. and Pržulj, N. (2014). GR-Align: fast and flexible alignment of protein 3d structures using graphlet degree similarity. *Bioinformatics*, **30**, 1259–1265.

Mayr, G., Dominques, F. S., and Lackner, P. (2007). Comparative analysis of protein structure alignments. *BMC Structural Biology*, **7**, 50.

Menke, M., Berger, B., and Cowen, L. (2008). Matt: local flexibility aids protein multiple structure alignment. *PLoS Computational Biology*, **4**, e10.

Micheletti, C. and Orland, H. (2009). MISTRAL: a tool for energy-mbased multiple structure alignment of proteins. *Bioinformatics*, **25**, 2663–2669.

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

Mosca, R., Brannetti, B., and Schneider, T. R. (2008). Alignment of protein structures in the presence of domain motions. *BMC Bioinformatics*, **9**, 352.

Pei, J., Kim, B.-H., and Grishin, N. V. (2008). PROMALS3D: a tool for multiple protein sequence and structure alignments. *Nucleic Acids Research*, **36**, 2295–2300.

Prlić, A., Bliven, S., Rose, P. W., Bluhm, W. F., Bizon, C., Godzik, A., and Bourne, P. E. (2010). Pre-calculated protein structure alignments at the RCSB PDB website. *Bioinformatics*, **26**, 2983–2985.

Ritchie, D. W., Mavridis, A. G. L., and Venkatraman, V. (2012). Fast protein structure alignment using Gaussian overlap scoring of backbone peptide fragment similarity. *Bioinformatics*, **28**, 3274–3281.

Russell, R. B. and Barton, G. J. (1992). Multiple sequence alignment from tertiary structure comparison: Assignment of global and residue confidence levels. *Proteins: Structure, Function, Genetics*, **14**, 309–323.

Sadowski, M. I. and Taylor, W. R. (2012). Evolutionary innacuracy of pairwise structural alignments. *Bioinformatics*, **28**, 1209–1215.

Salem, S., Zaki, M., and Bystoff, C. (2010). FlexSnap: flexible non-sequential protein structure alignment. *Algorithms for Molecular Biology*, **5**, article no. 12.

Sali, A. and Blundell, T. L. (1990). Definition of general topological equivalence in protein structures: A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *Journal of Molecular Biology*, **212**, 403–428.

Shatsky, M., Nussinov, R., and Wolfson, H. J. (2002). Flexible protein alignment and hinge detection. *Proteins: Structure, Function, Genetics.*, **48**, 242–256.

Shatsky, M., Nussinov, R., and Wolfson, H. J. (2004a). FlexProt: alignment of flexible protein structures without a predifinition of hinge regions. *Journal of Computational Biology*, **11**, 83–106.

Shatsky, M., Nussinov, R., and Wolfson, H. J. (2004b). A method for simultaneous alignment of multiple protein structures. *Proteins: Structure, Function, Bioinformatics*, **56**, 143–156.

Shealy, P. and Valafar, H. (2012). Multiple structure alignment with msTALI. *BMC Bioinformatics*, **13**, 105.

Sierk, M. L. and Kleywegt, G. J. (2004). Déjà vu all over again: Finding and analyzing protein structure similarities. *Structure*, **12**, 2103–2111.

Slater, A. W., Castellanos, J. I., Sippl, M. J., and Melo, F. (2013). Towards the development of standardized methods for comparison, ranking and evaluation of structure alignments. *Bioinformatics*, **29**, 47–53.

Spitzer, M., Wildenhain, J., Rappsilber, J., and Tyers, M. (2014). BoxPlotR: a web tool for generation of box plots. *Nature Methods*, **11**, 121–122.

Stebbings, L. A. and Mizuguchi, K. (2004). HOMSTRAD: recent developments of the homologous protein structure alignment database. *Nucleic Acids Research*, **32**, D203–D207.

Sun, H., Sacan, A., Ferhatsomanoglu, H., and Wang, Y. (2012). Smolign: a spatial motifs based protein multiple structural alignment method. *IEEE Transactions on Computational Biology and Bioinformatics*, **9**, 249–261.

Sutcliffe, M. J., Haneef, I., Carney, D., and Blundell, T. L. (1987). Knowledge based modelling of homologous proteins, part i: three-dimensional frameworks derived from the simultaneous superposition of multiple structures. *Protein Engineering*, **1**, 377–384.

Taylor, W. R. (1994). Multiple protein structure alignment. *Protein Science*, **3**, 1858–1870.

Van Walle, I., Lasters, I., and Wyns, L. (2005). SABmark–a benchmark for sequence alignment that covers the entire known fold space. *Bioinformatics*, **21**, 1267–1268.

Wang, S. and Zheng, W.-M. (2009). Fast multiple alignment of protein structures using conformational letter blocks. *Open Bioinformatics Journal*, **3**, 69–83.

Wang, S., Peng, J., and Xu, J. (2011). Alignment of distantly related protein structures: algorithm, bound and implications to homology modeling. *Bioinformatics*, **27**, 2537–2545.

Ye, T. and Godzik, A. (2003). Flexible structure alignment by chained aligned fragment pairs allowing twists. *Bioinformatics*, **19**(Suppl. 2), ii246–ii255.

Ye, T. and Godzik, A. (2005). Multiple flexible structure alignment using partial order graphs. *Bioinformatics*, **21**, 2362–2369.

Zemla, A. (2003). LGA a method for finding 3D similarities in protein structures. *Nucleic Acids Research*, **31**, 3370–3374.

Zhang, Y. and Skolnick, J. (2005). TM-align: a protein structure alignment algorithm based on TM-score. *Nucleic Acids Research*, **33**(7), 2302–2309.