Journal of Molecular Graphics and Modelling 29 (2010) 93–101

Contents lists available at ScienceDirect

Journal of Molecular Graphics and Modelling

journal homepage: www.elsevier.com/locate/JMGM

LigAlign: Flexible ligand-based active site alignment and analysis

Abraham Heifets[a,b], Ryan H. Lilien[a,b,c,*]

[a] Department of Computer Science, Univ. of Toronto, Toronto, Ontario, Canada
[b] Donnelly Centre for Cellular and Biomolecular Research, Univ. of Toronto, Toronto, Ontario, Canada
[c] Banting and Best Department of Medical Research, Univ. of Toronto, Toronto, Ontario, Canada

ARTICLE INFO

Article history:
Received 12 February 2010
Accepted 7 May 2010
Available online 27 May 2010

Keywords:
Protein structure alignment
Ligand superposition
Flexible ligand analysis
Bipartite matching
Dynamic programming
Clique enumeration

ABSTRACT

Ligand-based active site alignment is a widely adopted technique for the structural analysis of protein–ligand complexes. However, existing tools for ligand alignment treat the ligands as rigid objects even though most biological ligands are flexible. We present LigAlign, an automated system for flexible ligand alignment and analysis. When performing rigid alignments, LigAlign produces results consistent with manually annotated structural motifs. In performing flexible alignments, LigAlign automatically produces biochemically reasonable ligand fragmentations and subsequently identifies conserved structural motifs that are not detected by rigid alignment.

1. Introduction

Comparison of macromolecular structures yields insights into protein function and evolutionary relationships. These comparisons are typically performed after a global alignment of the target protein structures. Unfortunately, the non-local geometric constraints imposed by a global alignment will often prevent the discovery of locally conserved active site structure. This behavior is undesirable in the context of studying protein–ligand binding as active sites are often more conserved than the global protein shape [1,2]. One solution is to restrict alignment to the active site which induces a local alignment and reveals functionally relevant conserved structures.

One established method of local alignment is the ligand-based alignment of protein active sites. In ligand-based alignment, two or more protein–ligand complexes are superimposed by computing the transformations between the bound ligand of each complex. The computed transformations are then applied to the unbound proteins. The superimposed structures can then be examined to identify conserved geometric relationships among chemically similar residues. Consistent arrangements of amino acids provide evidence of shared protein function.

We define two categories of ligand-based protein alignment: rigid and flexible. The rigid alignment of two protein complexes requires first determining a pairwise atom correspondence between the two ligands and then identifying the single transformation which minimizes the root mean square deviation (RMSD) of the corresponding atoms. Previous work in ligand-based protein alignment has used the rigid model. In contrast, this work presents the first algorithm for flexible ligand-based protein alignment. In the flexible model, each ligand is considered a group of rigid fragments connected by hinges. After automatically identifying the hinge locations, a rigid ligand-based protein alignment is performed for each rigid fragment. Conceptually, the flexible model independently aligns the subcavities surrounding corresponding rigid fragments, thereby allowing the identification of conserved structure proximal to each fragment. We propose that flexible ligand-based protein alignment should be the method of choice when comparing complexes with flexible ligands.

Rigid ligand-based alignment of proteins has been used to uncover the evolutionary history between protein families [3], to predict protein function [3], to extract common structural patterns for ligand binding [4–6], and to explain enzymatic substrate specificity [7]. These investigations considered ligands with only a single chemical moiety: heme in human cytochrome P450 CYP17 [8,9], adenine [5,4,3], and the bioactive conformation of glutathione S-transferase inhibitor analogues [7]. Their approaches consisted of two phases. In the first phase, they performed a rigid alignment of these rigid ligands to produce a single superposition of the complexes. In the second phase, a number of closely related strategies were employed to identify conserved binding motifs. For example, after alignment, Kuttner et al. discovered conserved regions by iteratively identifying and removing the most densely populated

* Corresponding author at: Department of Computer Science, Univ. of Toronto, UofT DCS Rm 3302, 10 Kings College Rd, Toronto, Ontario, Canada.
E-mail address: lilien@cs.toronto.edu (R.H. Lilien).

1093-3263/$ – see front matter.
doi:10.1016/j.jmgm.2010.05.005

1.5 Å sphere of protein atoms. As in *k*-median clustering, an exemplar atom was chosen for each group of atoms near the center of the group. In another example, Nebel computed atom clusters by iterative pairwise protein comparison and elimination of distant or chemically dissimilar atoms, similar to average-linkage hierarchical agglomerative clustering. After the clustering terminated, a consensus virtual atom denoting the cluster was generated by averaging the positions of the remaining atoms. In contrast to these examples of explicit clustering, Koehler et al. estimated and compared the electrostatic and lipophilic potentials within the aligned binding sites.

The previous work succeeded because it applied a rigid alignment to effectively rigid ligands. Flexible ligands may be effectively rigid if, in practice, they exclusively bind in highly similar poses [7]. For example, in the rigid alignment of the flexible ligand NAD, Carugo and Argos report conserved structure around the well-aligned adenine moiety but not near the disordered nicotine nucleotide [6]. In other words, conserved structure is only found where the ligands happen to assume highly similar conformations. Therefore, rigid ligand-based protein alignment and its use of a single computed transformation is not ideal for the study of flexible ligands. Unfortunately, there are a large number of ligands with at least one internal hinge, and these ligands bind in widely varying conformations [10–13].

We present a new algorithm, LigAlign, for both rigid and flexible ligand-based protein alignment and comparison. Our work is the first to use the flexible alignment of ligands to detect conserved protein structure. LigAlign automatically computes an atom correspondence, identifies hinges, determines rigid fragments, computes the transformation for each fragment, and detects clusters of conserved residues. LigAlign implements a novel method for computing the correspondence between ligand atoms using a neighborhood-aware bipartite matching. LigAlign efficiently locates hinge sites using dynamic programming and branch-and-bound search and detects conserved residue clusters via clique enumeration. Notably, the use of dynamic programming for flexible ligand alignment was previously investigated for pharmacophoric inference over unbound ligands in the Pharmagist system [14]. While Pharmagist's analysis does not involve protein structure nor does it perform a ligand-based protein alignment, its dynamic programming approach to rigid fragment assembly is similar to LigAlign's use of dynamic programming to identify hinge sites.

We first validate our ligand alignment and cluster detection on the rigid heme system analyzed by Nebel [9] and show that LigAlign can automatically generate a set of residue clusters consistent with the manually annotated profile. We then compare rigid and flexible ligand alignment of the proteins from the Carugo and Argos test set [6]. When LigAlign performs a rigid alignment, the results are consistent with previously described binding patterns. More importantly, when LigAlign performs a flexible alignment, it identifies additional clusters of conserved residues that are undetected by rigid superposition.

LigAlign's source code is freely available under the GNU LGPL at http://compbio.cs.toronto.edu/ligalign. LigAlign is implemented in Python and supported on Apple OS X, Linux, and MS Windows. Our software is implemented as an extension to the PyMOL molecular visualization software [15] permitting integration with other PyMOL-based tools and allowing the user to easily generate high quality visualizations of the alignments.

2. System and methods

Flexible ligand-based protein alignment and analysis (which we will now simply refer to as flexible alignment) in LigAlign consists of three stages (Fig. 1). Stage 1, ligand correspondence (Section 2.1), computes a mapping between the atoms of the bound ligands in each input structure. Stage 2, ligand alignment (Section 2.2), automatically detects hinges and aligns the identified rigid fragments of each ligand. Finally, Stage 3, cluster detection (Section 2.3), identifies structurally and chemically conserved residues.

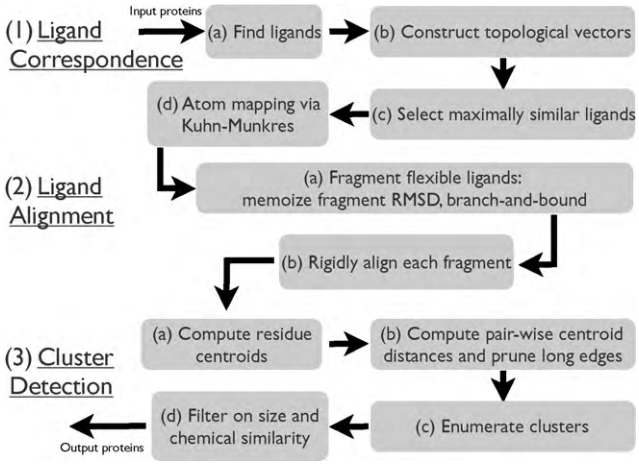


Fig. 1. LigAlign flowchart. The three stages of ligand-based active site alignment, namely ligand correspondence, ligand alignment, and cluster detection, are shown along with the techniques used to accomplish each stage.

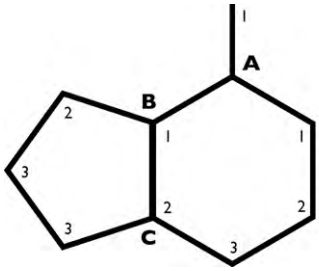
2.1. Correspondence of bound ligands

The goal of the ligand correspondence stage (Fig. 1) is to identify a set of ligands (one per protein complex) and an associated mapping between the corresponding atoms of each ligand. While the selected ligands are not required to be identical, they should be similar enough that when superimposed, they induce a clear alignment on their bound proteins and facilitate the identification of conserved protein structure. An automated system should select the most sensible single ligand from the multiple molecules frequently present in each Protein Data Bank (PDB) file. In practice, we find that LigAlign identifies the desired ligand from each protein complex, avoiding manual intervention.

2.1.1. Ligand extraction

The LigAlign system extracts the set of ligands that is most self-consistent across the submitted structures. This ligand extraction problem may be solved by reduction to weighted maximal-clique where nodes represent ligands and edge weights are proportional to the computed ligand similarity. Unfortunately, solving the NP-complete maximal-clique problem is prohibitively expensive when aligning multiple proteins with many ligands.

Instead of an exhaustive search, our ligand extraction algorithm identifies a *pivot* ligand in the first protein complex (the pivot protein) and the ligand from each non-pivot protein that is most similar to this pivot. We first eliminate a number of distracting molecules by assuming that ligands of interest must contain at least six heavy (*i.e.*, non-hydrogen) atoms. Ligands with fewer than six heavy atoms, such as lone metal ions or water, carry too little information to unambiguously align the proteins. To identify the pivot ligand, we compute a distance score (described in Section 2.1.2) between each ligand in the pivot protein and every ligand in each non-pivot protein. The pivot ligand is chosen as the ligand in the pivot protein that minimizes the sum of the distance scores between itself and the most similar ligand in each non-pivot protein. Ties are broken in favor of larger ligands.



	Minimal bond distance		
	1	2	3
A	3	3	3
B	3	5	1
C	3	4	2

Fig. 2. Topological vectors encode atom counts as a function of bond distance from a selected atom. For example, the numbers next to the atoms depict the minimal bond distance from atom A. The topology vector for B is [3, 5, 1] indicating three atoms at distance 1, five atoms at distance 2, and one atom at distance 3.

2.1.2. Atomic and molecular similarity

Our algorithms for atom correspondence and ligand extraction require distance measures that are both accurate and robust to minor structural variations. LigAlign derives its measures of atomic and molecular distance from a novel graph-based neighborhood comparison. For every atom, we generate a topology vector encoding the molecular branching and connectivity of its local neighborhood. Specifically, we count the number of atoms at each bond distance from the starting atom (Fig. 2). This count can be computed efficiently using the Floyd–Warshall all-pairs distance algorithm [16]. The molecular topology vector is defined as the average of the constituent atomic fingerprints.

The atomic and molecular distances are defined as the weighted L_1 distance over these topology vectors. We take the difference of the atom counts at each bond distance and sum the weighted difference, with the weight, w, decreasing linearly with the bond distance, where $w = (\text{maximum bond distance} - \text{atom pair distance})$. This ensures that differences in two atoms' local neighborhoods have a greater impact on their correspondence than remote differences. When mapped atoms are identified as the same element, we provide a bonus and reduce the distance score by the largest computed L_1 weight (*i.e.*, the maximum bond distance in the molecule), thereby improving the similarity. The molecular distance score is used in ligand extraction (Section 2.1.1) and the atomic distance score is used in ligand atom mapping (Section 2.1.3).

2.1.3. Ligand atom mapping

Once the ligands have been extracted, a mapping or correspondence is computed between the atoms of the pivot ligand and the atoms of each non-pivot, or *query*, ligand. Ideally, one could determine a correspondence directly from the atom names in the PDB files. Unfortunately, the oftentimes arbitrary naming of equivalent atoms [17,13,18] and the desire to compare similar but not identical ligands (*e.g.*, analogs from SAR studies [7]) makes this approach problematic.

LigAlign computes an atomic correspondence using the neighborhood-based atomic topology vectors described in Section 2.1.2. We perform a weighted bipartite matching between the atoms in the query ligand and the atoms in the pivot ligand with edge weights equal to the atomic distance score. We solve for the minimal-cost matching in polynomial time with the Kuhn–Munkres algorithm [19]. The minimal-cost matching defines an atom-to-atom correspondence, or mapping, for the atoms in the query ligand to the atoms in the pivot ligand. The computed mappings are injective but may be partial when not all atoms of the query or pivot ligands have a corresponding atom in the other molecule.

This neighborhood-based scoring function can yield ties in symmetric molecular regions, *e.g.* the ortho and meta positions of six-membered para-substituted rings, which can produce multiple equally consistent correspondences. Although the neighborhood connectivity and the minimal-cost matchings are identical for symmetric regions, these alternative atom mappings will yield different ligand alignments. We break these ties by selecting the mapping that generates the smallest global RMSD after rigid alignment.

2.2. Ligand alignment

After the ligands have been selected and mapped in the ligand correspondence stage (Section 2.1) we compute the transformations that yield the minimal RMSD between the bound ligands. We will discuss the flexible case in Section 2.2.1. In the rigid case, given the atoms of the query, V_q, and pivot, V_p, and an atom-to-atom correspondence $m : V_q \mapsto V_p$ (such as the mapping defined in Section 2.1.3), we define the RMSD of the rigid ligands as

$$\text{RMSD}(V_q, m) = \sqrt{\frac{\sum_{(v,u)\in \text{MappedAtoms}(V_q,m)} \text{distance}(v,u)^2}{|\text{MappedAtoms}(V_q, m)|}} \quad (1)$$

$$\text{MappedAtoms}(V_q, m) = \{(v, u) | v \in V_q, u = m(v)\}. \quad (2)$$

If the ligands are not identical, the mapping will be incomplete, *i.e.*, m will be undefined for some atoms in V_q. Therefore, this score is only computed over mapped atoms, as determined by Eq. (2). Given an atomic correspondence, the transformation that minimizes the RMSD between two rigid ligands can be computed in closed form [20]. Directly computing a minimal-RMSD ligand alignment, without an atomic mapping as computed in Section 2.1.2, is NP-hard [14].

2.2.1. Fragmentation of flexible ligands

Our algorithm models molecular flexibility by splitting ligands into a set of rigid fragments connected by flexible hinges. Each rigid fragment is aligned independently and is not constrained by the alignment of the other fragments. The user may either specify a limit on the number of hinges or allow LigAlign to automatically determine the number of fragments. In the second case, LigAlign iteratively increases the number of hinges until the improvement in RMSD gained by the incorporation of additional hinges is below a threshold amount.[1] Because the system lacks foreknowledge of the correct hinge placements, LigAlign performs a complete search of possible hinge placements. Similar to other flexible ligand analyses [14], this search is exhaustive; however, dynamic programming and branch-and-bound techniques typically prune the search space efficiently.

A bond connecting atoms x and y decomposes V_q into two subsets: a rigid molecular fragment, F_{x,y,V_q}, and the possibly empty remainder of the molecule, R_{x,y,V_q}. We can efficiently compute the RMSD between F_{x,y,V_q} and the corresponding part of the pivot ligand using Eq. (1) because F_{x,y,V_q} is rigid. Because the F_{x,y,V_q} and R_{x,y,V_q} subsets may be of different sizes, the contributions to the achievable RMSD for the entire molecule must be weighted by the number of atoms in each piece. If we have fewer than the maximum number of allowed hinges, we attempt to further reduce the RMSD of R_{x,y,V_q} by splitting it into smaller pieces.

$$\text{F}_{x,y,V_q} = \{q | q \in V_q, q \text{ has a path to } x \text{ which does not include } y\}. \quad (3)$$

$$\text{R}_{x,y,V_q} = V_q \setminus \text{F}_{x,y,V_q}. \quad (4)$$

[1] The default threshold is 10%.

$$\text{FlexibleRMSD}(V_q, m, bonds, k) = \begin{cases} \text{RMSD}(V_q, m) & \text{if } k = 0 \\ \min\limits_{(x,y)\in bonds} \sqrt{\frac{1}{|V_q|} \cdot \begin{pmatrix} \text{RMSD}(\text{F}_{x,y,V_q}, m)^2 \cdot |\text{F}_{x,y,V_q}| + \\ \text{FlexibleRMSD}(\text{R}_{x,y,V_q}, m, bonds, k-1)^2 \cdot |\text{R}_{x,y,V_q}| \end{pmatrix}} & \text{if } k \geq 1. \end{cases} \quad (5)$$

Eq. (5) simultaneously determines which bonds should have hinges and computes the minimal RMSD of the two ligands after flexible alignment. The recursive description presented allows caching of partial solutions for molecular fragments and necessary intermediate computations such as RMSD scoring. Memoizing the best fragmentation for a particular molecular piece yields an exponential reduction in the search space, especially in long chain or branched molecules. In the extreme case of a linear molecule, this dynamic programming formulation reduces the runtime from exponential to polynomial, analogous to CKY parsing of context-free grammars [21]. Furthermore, the alternative extreme case, a ligand composed mainly of conjugated ring systems such as heme, is also handled quickly. As a consequence of Eq. (3), rings are treated as rigid fragments and no time is spent attempting to split them.

The recursive search for the best hinges in a molecular fragment has a worst-case time exponential in the number of hinges. Fortunately, the RMSD(·) function of Eq. (5) is non-negative and can be used to compute a lower bound on the FlexibleRMSD of the current fragmentation. This allows us to implement an efficient branch-and-bound search over possible fragmentations. The current implementation of LigAlign incorporates all of the algorithmic extensions described in this section.

2.3. Residue cluster extraction via clique detection

The result of the second stage (Section 2.2) is a set of ligands split into rigid fragments and a set of transformations capable of aligning these fragments. The third stage of flexible alignment and analysis in LigAlign is the identification of common chemical or structural protein features. In the case of flexible ligand-based alignment, each rigid fragment and its accompanying protein is superimposed against the pivot ligand and the pivot protein. Common features can be found by identifying regions of the active site where several proteins have placed chemically similar amino acids. These residue clusters are identified and reported by LigAlign. The now-aligned proteins are examined for conserved residue clusters and these clusters are displayed through the PyMOL interface to the user. Although, for clarity, we restrict our discussion here to protein residues, LigAlign will report clusters of solvent molecules. An example is found in Section 3.2.

For k proteins, residue cluster detection is an instance of k-Partite 3D matching, an NP-hard problem [22]. Our algorithm for computing residue clusters, which completes quickly on typical test cases, is exact and complete, and therefore requires exponential time in the worst case. LigAlign computes the centroids of each residue in each protein, as in SPASM [17], and creates a graph with a node corresponding to each centroid. Weighted edges connect nodes from one protein to nodes from the other proteins, where the edges have weight proportional to the Euclidian distance between the centroids. Edges with a distance above a specified threshold are removed (see next paragraph). Possible clusters are then determined by enumerating cliques in this graph. Cliques are ranked so as to maximize cardinality and, secondarily, minimize total edge weight. Any clique that is a proper subset of another clique is removed.

We extend this basic algorithm in three ways. First, the requirement that every protein must contribute to every cluster is often too restrictive. Therefore, LigAlign accepts a lower bound on the fraction of input proteins that must have a residue present in a cluster for the cluster to be reported. Second, as the diameter of the cluster increases, the biological significance of the cluster is likely to decrease. Our clustering algorithm takes a bound for the maximum acceptable cluster diameter to prevent reporting of biologically meaningless large-diameter clusters. We remove edges from the centroid-distance graph if the distance between the aligned residues is more than the user-specified maximum cluster diameter. This pruning ensures that the distance between every pair of clustered residues is smaller than the maximum diameter while simultaneously improving runtime. Although clusters that are proper subsets of another cluster are removed, it is possible that reported clusters can partially overlap. For example, given a set of core residues and several outliers, multiple alternative clusters may be reported with a different outlier in each cluster. Such boundary cases can occur regardless of the chosen maximum cluster diameter. Instead of arbitrarily selecting only one of the overlapping clusters, LigAlign returns all clustering alternatives.

Finally, the third extension allows the user to specify chemical constraints on the membership of residue clusters. Depending on the biological system, residues with different chemical properties may or may not be considered meaningful evidence of a conserved template. LigAlign includes three measures of residue similarity for filtering returned clusters. The least restrictive measure is geometric, which requires no chemical similarity. A second measure categorizes amino acids into chemical classes and requires that all residues in a cluster belong to the same group. We use the same seven non-exclusive chemical classes as Nebel: acidic (D, E), basic (R, H, K), amidic (N, Q), nonpolar (L, V, A, G, I, M, P), aromatic (F, W, Y), hydroxyl (S, R, Y), and sulphide bond forming (C) [9]. The third measure of similarity requires all residues in a cluster to be the same amino acid type. These requirements are applied during clique generation to prune prospective clusters and further speed cluster detection.

3. Results and discussion

We performed a series of experiments using LigAlign in both its rigid and flexible alignment modes to study the structures of two protein families. We first examined the cytochrome P450 active site and its rigid heme cofactor. As validation, we directly compared the structural motifs found by LigAlign to those previously described [9]. We then examined LigAlign's performance on the more difficult case of the NAD-binding proteins originally investigated by Carugo and Argos [6]. Although NAD is a flexible ligand, we first performed a rigid ligand-based protein alignment to allow comparison to previous work. We then performed a flexible ligand alignment on the NAD-binding protein family and compared the flexible results to those obtained via rigid alignment.

3.1. Heme

We demonstrate LigAlign's ability to extract biologically significant residue motifs from the structure of human cytochrome P450 CYP17 by performing an alignment of the rigid heme moiety in PDB files 1BU7, 1H5Z, 1N97, 1PQ2, and 1W0G. We validate LigAlign's results by comparing the detected clusters to results previously reported by Nebel [9]. The patterns discussed by Nebel are

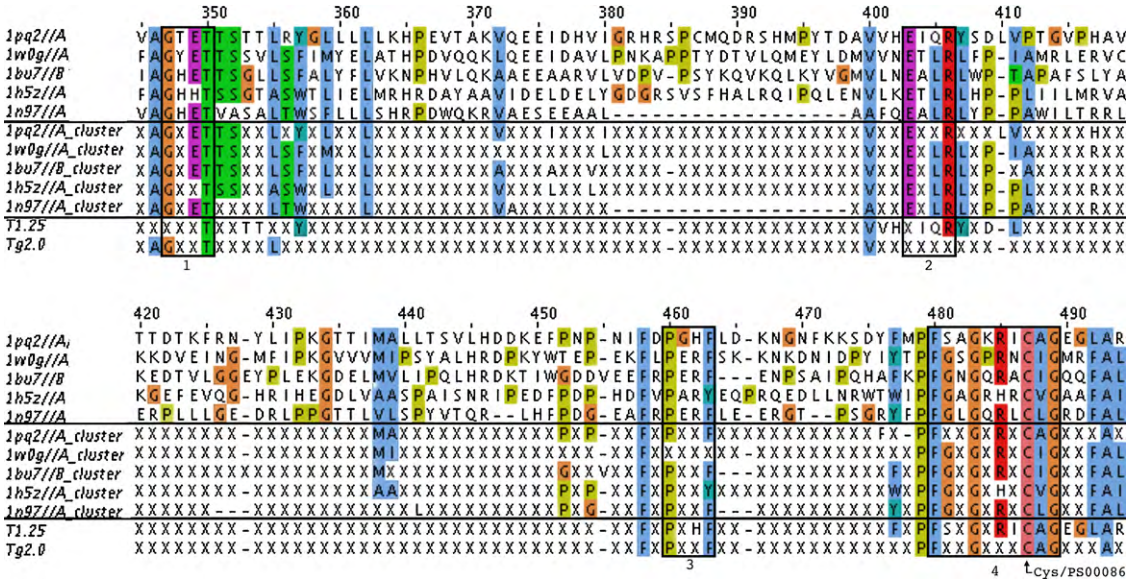


Fig. 3. A sequence alignment of P450 homologs and extracted patterns, as computed by MUSCLE [24] using default settings. The colors correspond to the default Clustal coloring scheme. Four biologically significant regions [9] are boxed and numbered and the conserved cysteine residue from PROSITE pattern PS00086 is marked. The first five rows comprise the input proteins; the next five rows show the subset of residues marked as conserved by LigAlign; the last two rows are the conserved residues in the results reported by Nebel. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

the P450 conserved tetrapeptide G-x-[DEH]-T, the E-x-x-R and P-E-R-F motifs, and the P450 cysteine heme-iron ligand signature [FW]-[SGNH]-x-[GD]-{F}-[RKHPT]-{P}-C-[LIVMFAP]-[GAD]. These patterns are highlighted with boxes and labeled 1, 2, 3, and 4, respectively, in Fig. 3. Pattern 4 is PROSITE database pattern PS00086 [23], and its conserved cysteine residue is marked with an arrow.

Fig. 3 shows our results for this system. The first five rows of Fig. 3 show a sequence alignment of 1BU7, 1H5Z, 1N97, 1PQ2, and 1W0G, as computed by MUSCLE [24] using default settings. The residues are colored by MUSCLE according to the standard Clustal coloring rules. LigAlign's clusters are shown in the middle five rows. These are the residues selected from each protein by LigAlign via ligand alignment and cluster detection.[2] The acceptable cluster diameter was set at 2.5 Å to permit direct comparison with previously reported results, as described below. LigAlign pruned any clusters with less than 80% representation from the proteins (*i.e.*, fewer than four out of the five proteins considered). LigAlign was set to reject clusters of residues that were not of the same chemical class. This experiment completed in approximately 2 min on our test system, a 2.53GHz MacBook Pro laptop running OS X 10.6.2.

LigAlign detects clusters that match the entirety of the G-x-[DEH]-T and E-x-x-R motifs and the P450 cysteine heme-iron ligand signature. For patterns 1 and 2, LigAlign's reported patterns include the known motif. Ligalign reports G-x-E-T for pattern 1, additionally specifying a glutamic acid, and E-x-L-R for pattern 2, additionally specifying a leucine. These extensions to the known patterns reflect the conservation of the residues present in our five selected proteins. As seen by examining the first five rows of Fig. 3, this particular set of P450 homologous proteins shows consistent clusters of glutamic acid and leucine. A different set of input proteins would be required to discover patterns as permissive as the known motifs.

On pattern 3, LigAlign's performance matches Nebel's results, with both systems finding the proline and phenylalanine of the P-E-R-F motif. LigAlign does not detect the glutamic acid or arginine amino acids of the pattern. These residues turn out to be less spatially conserved than the proline and phenylalanine. The glutamic acid residues form a cluster with a diameter of 4.32 Å and the arginine amino acids form a cluster with a diameter of 2.69 Å. LigAlign detects these clusters only when run with a larger acceptable diameter. Under the set cutoff of 2.5 Å, LigAlign correctly prunes both the glutamic acid and arginine residues of pattern 3.

We detect a number of residue clusters that fall outside of the four previously discussed patterns. These clusters are colored in light grey in Fig. 4. Although the residues are not contained in the four discussed patterns, examination of Fig. 4 shows that such clusters have good spatial and chemical agreement. These clusters are unsurprising, given that this set of proteins shows consistent sequence alignments outside of the four patterns (Fig. 3).

The last two rows of Fig. 3 are reproduced from Nebel's best template extraction results. `T1.25` is determined from clusters formed from the atoms of the active site by merging the nearest atoms of the same elemental type, as long as these atoms are within 1.25 Å. `T1.25` lists all residues containing any such clustered atom. Our acceptable cluster diameter of 2.5 Å was selected to correspond to `T1.25`, since LigAlign defines cluster size in terms of maximum diameter and Nebel's agglomerative clustering bound determines a radius. `Tg2.0` reports the clusters Nebel generated from residue centroids, rather than atoms, with a distance bound of 2.0 Å.

Nebel performs cluster detection on atoms rather than amino acids and, for sequence representation, chooses a single exemplar from the set of amino acids that contributed atoms to the cluster. LigAlign reports the residues of the original proteins, thereby avoiding the need to choose a single (possibly misleading) residue to represent clusters that may contain several different amino acids. This explicit presentation of residues makes the variation within clusters clear and can be helpful in determining if clusters are missing representative residues due to variation in chemical class (such

[2] The simplicity of using LigAlign is demonstrated by the fact that this experiment required only two commands. Ligand alignment was performed with the command "`ligalign 1pq2, 1w0g, 1bu7, 1h5z, 1n97`" Clusters were computed and reported with the command "`ligalign_find_template 2.5, 0.8, chemical, 1pq2, 1w0g, 1bu7, 1h5z, 1n97`".

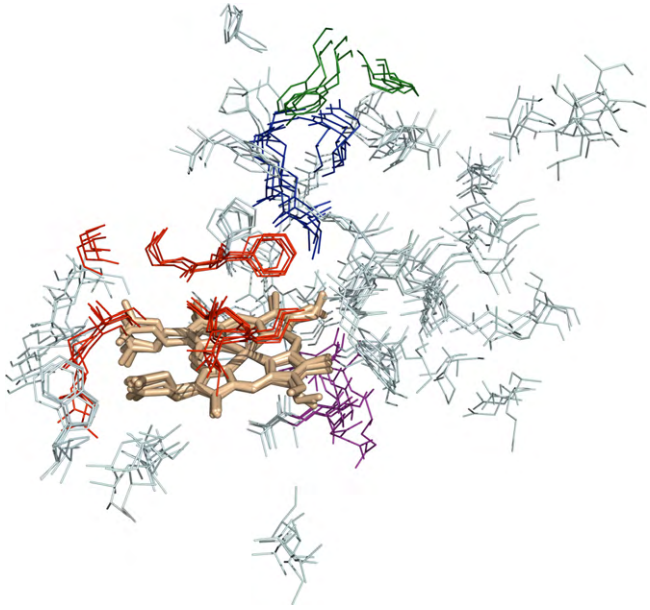


Fig. 4. A ligand-based alignment of P450 homologs showing the shared heme moiety (colored tan) surrounded by the residue clusters detected by LigAlign. The patterns labelled in Fig. 3 are distinguished by color. Pattern 1 is shown in purple; pattern 2 is dark blue; pattern 3 is green; and pattern 4 is presented in red. Clustered residues which are not members of one of these patterns are shown in light grey.

as the serine in pattern 4 of 1PQ2) or diverged spatial placement (as with pattern 3 of 1W0G).

3.2. Nicotinamide adenine dinucleotide

NAD is a ubiquitous ligand and is known to adopt multiple conformations [13]. To demonstrate the impact of flexible alignment, we compared the results of LigAlign's flexible and rigid alignments of NAD. Before this comparison, we first validated the clusters reported under rigid alignment against previously known binding patterns. Carugo and Argos describe nicotinamide adenine dinucleotide binding motifs present in a wide variety of NAD-dependent proteins [6]. They perform a rigid minimal-RMSD alignment of 21 NAD cofactors and visually determine consistent patches of enzyme residues and solvent atoms that fall within 4.5 Å of any NAD atom. These results are summarized in NAD-binding sequence patterns such as G-x-G-x-x-G or G-x-x-G-x-x-G [25], where the 'x' represents any amino acid and reflects the variability exhibited in the residue sites.

3.2.1. Rigid NAD alignment

We use LigAlign to rigidly align the NAD ligand of the 21 proteins examined by Carugo and Argos, which are listed in Table 1, and automatically extract clusters of conserved residues. Like Carugo and Argos, we specify 1HDX as the pivot ligand conformation. We choose a maximum cluster diameter of 3.0 Å and prune clusters that lack residues from at least half of the proteins. We also keep clusters only when the residues are in the same chemical class. The ligand alignment and cluster detection for the 21 NAD-binding proteins required just under 24 min of runtime on our test system. The aligned NAD ligands and extracted clusters are shown in Fig. 5. The disorder in the alignment of the nicotine and nicotine ribose moieties is clearly visible, as noted by Carugo and Argos. Groups of water molecules, which are visible near the diphosphate and adenine moieties, are also detected by LigAlign. These are structurally conserved solvent molecules responsible for mediating hydrogen bonding [25,6].

An alternative presentation of the extracted residue clusters is shown in Table 1. Table 1 lists 17 clusters comprising the residues extracted as conserved by LigAlign. Cluster numbers are assigned such that the residue positions in the pivot protein are non-decreasing. For each cluster, the amino acid type and sequence position of the participating residues are reported. These clusters are generated by post-processing the original clusters reported by LigAlign. Similar to Carugo and Argos, any cluster consisting of residues further than 4.5 Å from the aligned ligands is discarded. When a cluster lies on the boundary, with some residues further than 4.5 Å, Carugo and Argos report the distant residues; we highlight such residues in grey in Table 1.

As discussed in Section 2.3, LigAlign reports alternative clusterings of outlier residues which, when visualized, produce the appearance of a single larger cluster. Merging the overlapping clusters produces a list of clusters that more closely matches the

Table 1
Merged clusters as detected after rigid superposition of bound NAD ligands. Diameter is the maximum pairwise distance of residue centroids in a cluster in Angstroms. Residues of type '?' are either water or small-molecule ligands. Entries in grey are further than 4.5 Å from any of their protein's NAD atoms. The last row shows the correspondence of LigAlign's clusters with the conserved interacting positions reported by Carugo and Argos [6]. Cluster 15 has a representative from each protein and a cluster diameter of 0 Å because it is the bound NAD ligand. Cluster 16 is a water molecule also reported by Carugo and Argos (see Fig. 11 [6], *ibid.*). Cluster 17 corresponds to a structurally conserved water molecule described by Bottoms et al. [25].

PDB ID	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17
1HDX	–	–	G199	L200	G201	G202	V203	G204	V222	D223	I224	I269	–	V292	?377	–	?385
1BMD	A132	A245	G10	A11	G13	–	I15	G16	L40	E41	I42	G87	A88	V128	?334	–	?342
1DHR	A133	–	G13	G14	G16	A17	L18	–	I36	D37	V38	A83	G84	–	?241	?245	?248
1EMD	V121	M227	G7	A8	G10	G11	I12	G13	–	D34	I35	A77	G78	I117	?314	?317	?316
1GD1	A120	–	G7	–	G9	–	I11	G12	–	D32	L33	–	G97	–	?336	?380	?352
1GEU	–	–	G174	–	–	L203	–	–	–	E197	M198	A260	G262	I178	?452	–	–
1GGA	A134	–	G8	–	G10	–	I12	–	V36	D37	M38	–	G111	–	?361	–	–
1HDG	A120	–	G7	–	G9	–	I11	G12	–	D32	L33	–	G97	–	?336	?422	?369
1HDR	A136	–	G16	G17	G19	A20	L21	L21	V39	D40	V41	A86	G87	–	?244	?247	?249
1HLP	V140	P249	G27	A27A	G29	–	V31	G32	V51	D52	I53	A96	G97	–	?330	–	–
1LDM	V140	I249	G27	V28	G29	A30	V31	G32	V51	D52	V53	A96	G97	V136	?330	?373	–
1LDN	V140	I251	G27	A28	G29	–	V31	G32	I51	D52	A53	A96	G97	A136	?352	?364	?356
1LLD	V127	I240	G14	A15	G16	A17	V18	G19	–	D39	I40	A83	G84	I123	?320	–	?324
1PSD	V265	V112	G158	–	G160	–	I162	G163	–	D181	I182	V211	P212	A238	?450	–	?566
1LVL	P268	–	A263	–	G180	–	–	I182	–	–	–	V264	–	–	?460	–	?583
2HSD	–	I217	G13	G14	–	G17	L18	G19	A36	D37	V38	A88	G89	I137	?256	–	–
2NAD	V309	V150	A198	A199	G200	–	I202	G203	–	D221	–	–	P256	A283	?394	?578	?409
2NPX	–	–	G156	–	G158	L185	–	I162	–	D179	–	–	–	–	?818	–	?1147
2OHX	L309	–	G199	L200	G201	G202	V203	G204	V222	D223	I224	I269	–	V292	?403	?616	?477
4MDH	A132	A245	G10	A11	G13	–	I15	A16	L40	D41	I42	G87	–	V128	?335	?396	?362
9LDT	V142	I250	G28	V29	G30	A31	V32	G33	V52	D53	V54	A98	G99	V138	?401	?442	?424
Diameter	3.99	2.59	3.00	2.29	4.09	2.98	3.76	4.41	2.86	3.45	3.65	3.18	3.61	3.69	0.00	2.91	3.38
Carugo et al.	–	–	S1	S2,3	S4	S5	S6	S7	S8	S9	S10	S11,15	S12,16	–	–	–	–

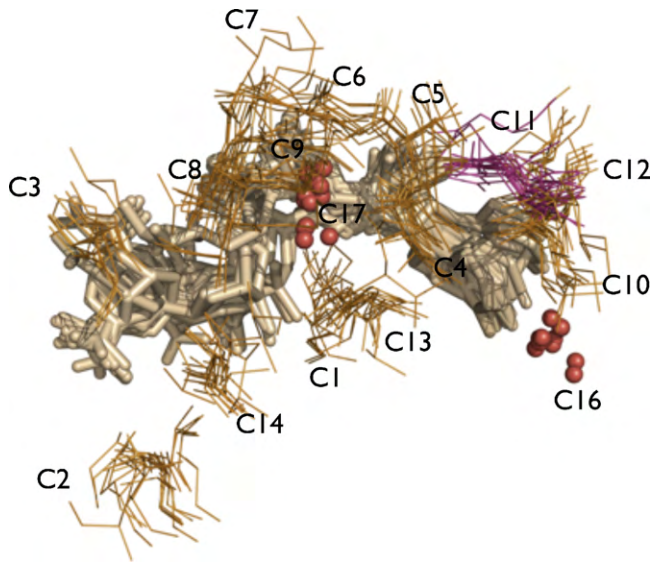


Fig. 5. A rigid minimal-RMSD ligand-based alignment of the NAD ligand from the 21 NAD-binding proteins listed in Table 1. The shared NAD ligand is colored tan. Consistent clusters of residues are shown surrounding the ligand. Hydrophobic residues (LVAGIMP) are depicted as orange lines; acidic residues (DE) are shown as pink lines. Red spheres correspond to the location of structurally conserved water molecules. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

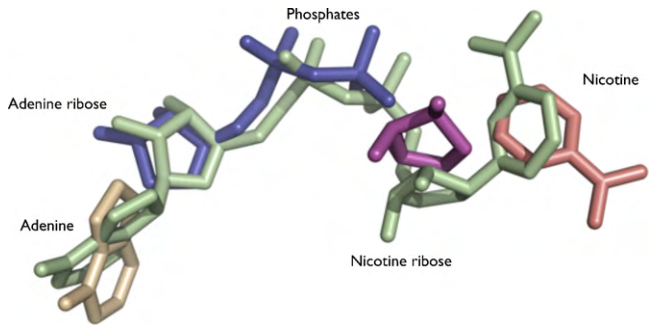


Fig. 6. Fragmentation of the bound NAD ligand from 1HDR (each rigid fragment is shown in a different color), after fragmentation but before alignment to the green pivot NAD ligand in 1HDX. The NAD moieties of nicotine, nicotine ribose, nicotine phosphate, adenine phosphate, adenine ribose, and adenine are labeled. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

presentation shown in Fig. 5. In the case when overlapping clusters cannot be unambiguously merged (*i.e.*, when one protein contributes different amino acids to each of the different clusters) we report the cluster of larger cardinality, breaking ties to favor the cluster with smaller diameter. The diameters of the merged clusters are listed in Angstroms in the second to last row of Table 1, and the original clusters may be inspected in Table 1 through 22 of the supplementary material.

The final row of Table 1 shows the correspondence between LigAlign's clusters and the sequence positions reported by Carugo and Argos. Carugo and Argos manually identify 17 residue sites, labelled S1 through S17, as interacting with NAD. The clusters reported by LigAlign include interacting positions manually identified by Carugo and Argos. In particular, columns C5, C8, and C13 of Table 1 correspond to the glycines in the binding motif of G-x-G-x-x-G. Carugo and Argos' positions S13, S14, and S17 were not identified by LigAlign as these clusters contained residues from fewer than half of the proteins considered.

LigAlign also reports several clusters, C1, C2, and C14, which were not described by Carugo and Argos. Although we cannot use Carugo and Argos's results to validate these clusters, Fig. 5 shows they are as spatially and chemically consistent as the other detected clusters.

3.2.2. Flexible NAD alignment

The rigid alignment results of Section 3.2.1 are consistent with those previously reported in the literature. Of course, a rigid alignment of a flexible ligand, such as NAD, has inherent limitations. Having verified our clustering methodology for rigid molecules, we next used LigAlign to perform a fragment-based flexible alignment of NAD.

As in the rigid case, we took 1HDX as the pivot conformation, used a maximum cluster diameter of 3.0 Å, and pruned clusters of disparate chemical type or with residue representation from fewer than half of proteins. We used the default minimum fragment size of six heavy atoms. Rather than specify a number of fragments for partitioning NAD, we used LigAlign's default behavior to iteratively increase the number of hinges inserted until the incremental RMSD improvement from additional hinges was less than 10%. Given the set parameters, LigAlign automatically determines a fragmentation that yields a minimal-RMSD alignment against the pivot ligand conformation. The fragmentation for the NAD ligand of 1HDR is shown in Fig. 6. In some cases, such as 9LDT or 1LDN, the bound conformation of the ligand is sufficiently similar to the conformation of NAD in 1HDX, that the ligand is not fragmented. In this situation, alignment proceeds identically to the rigid case. In the proteins under consideration, the largest number of fragments required is 4. Each of these fragments can be independently aligned to the pivot. The fragment alignments induce a fragment-based superposition of the bound proteins, which can be searched for clusters of amino acids.

Once we have fragmented the ligands, we generate a protein alignment for each moiety by superimposing the fragment containing the moiety under consideration against the pivot. The consistent residues around each moiety can be determined by examining each alignment in turn. Among common natural ligands, NAD is particularly flexible and contains six distinct moieties (nicotine, nicotine ribose, nicotine phosphate, adenine phosphate, adenine ribose, and adenine). The large number of rigid fragments creates a significant amount of clustering data. Table 5 through 22 in the supplementary material list the overlapping clusters discovered when the superposition of each moiety is independently considered. As expected, the fragment-based alignments of the adenine, adenine ribose, and phosphate moieties find clusters consistent with those discovered through rigid alignment, since a rigid alignment achieves a close superposition of these moieties [6]. A fragment-based alignment of the nicotine ribose also finds clusters consistent with the rigid alignment.

In the case of the nicotine fragment, however, our flexible results diverge from the rigid analysis. Carugo and Argos note a high variability in the orientation of this moiety and the shape of the surrounding active site pocket. Using rigid alignment, they find no consistent positions near the nicotine. Using LigAlign's fragment-based alignment, we can avoid the disorder induced by rigid ligand alignment on the position and orientation of the nicotine moiety. After flexible alignment, LigAlign suggests a new structurally conserved group corresponding to cluster C5 in Table 2, shown in Fig. 7. The amino acids in cluster C5 are close enough to be considered part of the active site and Carugo and Argos group the arginine residues with their S5 position. However, when the nicotines are precisely aligned, a structurally and chemically consistent cluster of basic residues becomes apparent.

As demonstrated by NAD, a rigid alignment of a complete ligand gives no guarantee that constituent ligand moieties will be meaningfully aligned. In contrast, flexible ligand-based alignment does

Table 2
Merged clusters (see discussion in Sections 2.3 and 3.2.1) as detected after rigid superposition of fragments containing the nicotine moiety. Entries in grey are further than 4.5 Å from any of their protein's nicotine fragment's atoms. Residues of type '?' in cluster C6 are conserved waters.

PDB ID	C1	C2	C3	C4	C5	C6
1HDX	G199	G201	V203	I269	–	?385
1BMD_1	G10	G13	I15	G87	H186	?342
1DHR_3	–	A149	–	G84	–	–
1EMD_4	I97	–	I75	–	–	–
1GD1_3	–	–	–	–	R10	–
1GEU_1	–	–	–	–	–	?478
1GGA_4	–	–	–	–	R11	–
1HDG_3	–	–	–	–	R10	–
1HDR_1	–	A152	–	A86	–	–
1HLP_2	–	A34	A250	–	–	–
1LDM_1	G27	G29	V31	A96	H193	–
1LDN_1	G27	G29	V31	A96	H193	?356
1LLD_1	G14	G16	V18	A83	H180	?324
1PSD_2	G158	G160	I162	–	H292	?566
1LVL_4	–	A313	–	–	–	–
2HSD_3	V107	–	–	G89	–	–
2NAD_2	A198	G200	I202	–	H332	?409
2NPX_2	–	–	–	–	–	?900
2OHX_1	G199	G201	V203	I269	–	?477
4MDH_1	G10	G13	I15	G87	H186	?362
9LDT_1	G28	G30	V32	A98	H195	?424
Diameter	2.96	3.55	3.01	2.93	2.90	2.73

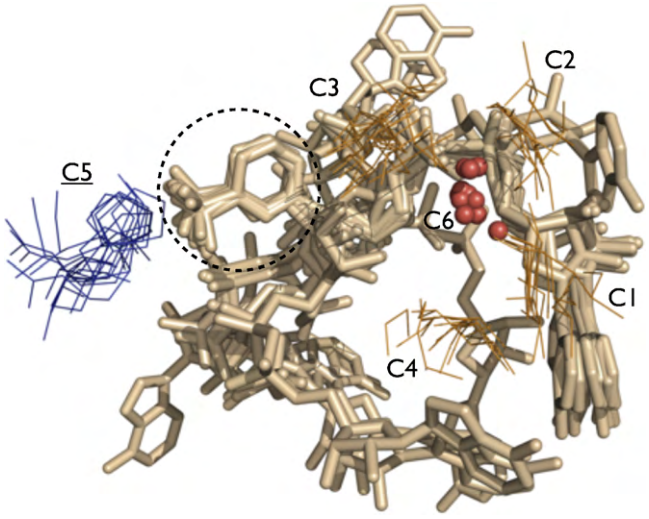


Fig. 7. A protein alignment using only the nicotine moiety of the NAD ligand (shown in a dashed circle). The shared NAD ligand is colored tan. The misalignment induced by nicotine alignment on the non-nicotine moieties is apparent. Consistent clusters of residues, as listed in Table 2, are shown surrounding the ligand. The underlined C5 cluster was identified by flexible alignment but was not detected by rigid alignment. Hydrophobic residues (LVAGIMP) are depicted as orange lines; basic residues (RHK) are blue lines. Red spheres correspond to the location of structurally conserved water molecules. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

not suffer from this disadvantage. By separately aligning each moiety, additional conserved structural relationships can be identified.

4. Conclusion and future work

The structural analysis of protein families relies on the ability to accurately align corresponding regions of individual structures. Ligand-based protein alignment has emerged as one solution for this alignment problem. Unfortunately, while state-of-the-art methods succeed for rigid ligands, their performance falls short for ligands which bind in multiple conformations. For flexible ligands, a rigid ligand-based protein alignment can induce an apparent disorder in the residues surrounding each ligand moiety causing clustering to fail.

In this paper, we presented LigAlign, a new software system for the structural analysis of protein active sites via ligand-based protein alignment. When performing rigid alignments, LigAlign produces automatically generated results consistent with manually annotated, biologically relevant structural motifs. When performing flexible alignments, LigAlign automatically produces biochemically reasonable ligand fragmentations and identifies conserved structural motifs that are not detected by the rigid alignment. This scenario was demonstrated for the flexible ligand NAD. The adoption of LigAlign as a tool for structural biologists could uncover similar examples of non-obvious amino acid structural conservation.

Several subproblems of flexible ligand-based protein alignment (such as ligand correspondence, unmapped ligand alignment, and cluster detection) are NP-hard and therefore likely to have a worst-case exponential runtime. In LigAlign, a series of optimizations including branch-and-bound search, dynamic programming, and memoization allow the typical execution to complete quickly. Looking forward, several improvements and design alternatives may be possible. For example, LigAlign currently computes ligand correspondences using a maximal bipartite matching based on neighborhood similarity. While this technique is robust to structural and naming discrepancies, it is not guaranteed to properly map ligands that share only small fragments. A ligand mapping technique based on maximal common substructure detection could address this limitation. Unfortunately, maximal common substructure detection is NP-complete and it is unknown if new molecularly optimized algorithms [26] will complete sufficiently quickly on large ligands, such as heme.

It may also be possible to improve the clustering step. LigAlign finds clusters of residue and solvent centroids based on geometric proximity and filtered by chemical similarity. These tests are currently binary, where a residue will be either accepted into a cluster or rejected as inconsistent. An alternative would be to use a soft assignment of residue membership to clusters, producing a real-valued consistency score. For example, clusters could be judged based on BLOSUM scores [17] producing a non-binary measure of cluster acceptability.

We believe LigAlign is a useful tool for the structural biology community and we encourage structural biologists to use it. LigAlign runs under the PyMOL molecular viewing program. This integration allows the user to easily generate and interact with the types of figures found in this manuscript. LigAlign is supported on Apple OS X, Linux, and MS Windows. The source code is freely available under the GNU LGPL. LigAlign is available for download at http://compbio.cs.toronto.edu/ligalign.

Acknowledgements

We thank Ms. Maria Safi, Mr. Izhar Wallach, and all members of the Lilien lab for helpful discussions and comments on drafts.

This work is supported by grants to R.H.L. from the Bill and Melinda Gates Foundation (Grand Challenges Explorations) and the Natural Sciences and Engineering Research Council of Canada (Discovery).

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.jmgm.2010.05.005.

References

[1] K.A. Denessiouk, J.V. Lehtonen, M.S. Johnson, Enzyme–mononucleotide interactions: three different folds share common structural elements for ATP recognition, Protein Sci. 7 (1998) 1768–1771.

[2] N. Kobayashi, N. Go, ATP binding proteins with different folds share a common ATP-binding structural motif, Nat. Struct. Biol. 4 (1997) 6–7.
[3] J.-C. Nebel, P. Herzyk, D.R. Gilbert, Automatic generation of 3D motifs for classification of protein binding sites, BMC Bioinformatics 8 (2007) 321.
[4] Y.Y. Kuttner, V. Sobolev, A. Raskind, M. Edelman, A consensus-binding structure for adenine at the atomic level permits searching for the ligand site in a wide spectrum of adeninecontaining complexes, Proteins 52 (2003) 400–411.
[5] K.A. Denessiouk, V.V. Rantanen, M.S. Johnson, Adenine recognition: a motif present in ATP-, CoA-, NAD-, NADP-, and FAD-dependent proteins, Proteins 44 (2001) 282–291.
[6] O. Carugo, P. Argos, NADP-dependent enzymes. I: conserved stereochemistry of cofactor binding, Proteins 28 (1997) 10–28.
[7] R.T. Koehler, H.O. Villar, K.E. Bauer, D.L. Higgins, Ligand-based protein alignment and isozyme specificity of glutathione S-transferase inhibitors, Proteins 28 (1997) 202–216.
[8] J.-C. Nebel, Modelling of P450 active site based on consensus 3D structures, in: Int. Conf. Biomed. Eng., 2005.
[9] J.-C. Nebel, Generation of 3D templates of active sites of proteins with rigid prosthetic groups, Bioinformatics 22 (2006) 1183–1189.
[10] J.A. Erickson, M. Jalaie, D.H. Robertson, R.A. Lewis, M. Vieth, Lessons in molecular recognition: the effects of ligand and protein flexibility on molecular docking accuracy, J. Med. Chem. 47 (2004) 45–55.
[11] R. Najmanovich, N. Kurbatova, J. Thornton, Detection of 3D atomic similarities and their use in the discrimination of small molecule protein–binding sites, Bioinformatics 24 (2008) i105–i111.
[12] A. Kahraman, R.J. Morris, R.A. Laskowski, J.M. Thornton, Shape variation in protein binding pockets and their ligands, J. Mol. Biol. 368 (2007) 283–301.
[13] G.R. Stockwell, J.M. Thornton, Conformational diversity of ligands bound to proteins, J. Mol. Biol. 356 (2006) 928–944.
[14] D. Schneidman-Duhovny, O. Dror, Y. Inbar, R. Nussinov, H.J. Wolfson, Deterministic pharmacophore detection via multiple flexible alignment of drug-like molecules, J. Comput. Biol. 15 (2008) 737–754.
[15] W.L. DeLano, The PyMOL Molecular Graphics System, 2002, http://www.pymol.org.
[16] R.W. Floyd, Algorithm 97: shortest path, Commun. ACM 5 (1962) 345.
[17] G.J. Kleywegt, Recognition of spatial motifs in protein structures, J. Mol. Biol. 285 (1999) 1887–1897.
[18] C.A. Bottoms, D. Xu, Wanted: unique names for unique atom positions. PDB-wide analysis of diastereotopic atom names of small molecules containing diphosphate, BMC Bioinformatics 9 (Suppl. 9) (2008) 16.
[19] J. Munkres, Algorithms for the assignment and transportation problems, J. Soc. Ind. Appl. Math. 5 (1957) 32–38.
[20] W. Kabsch, A discussion of the solution for the best rotation to relate two sets of vectors, Acta Crystallogr. A 34 (1978) 827–828.
[21] D. Kozen, Automata and Computability, Springer, New York, 1997.
[22] M. Shatsky, A. Shulman-Peleg, R. Nussinov, H.J. Wolfson, The multiple common point set problem and its application to molecule binding pattern detection, J. Comput. Biol. 13 (2006) 407–428.
[23] N. Hulo, A. Bairoch, V. Bulliard, L. Cerutti, B.A. Cuche, E. de Castro, C. Lachaize, P.S. Langendijk-Genevaux, C.J.A. Sigrist, The 20 years of PROSITE, Nucleic Acids Res. 36 (2008) D245–D249.
[24] R.C. Edgar, MUSCLE: a multiple sequence alignment method with reduced time and space complexity, BMC Bioinformatics 5 (2004) 113.
[25] C.A. Bottoms, P.E. Smith, J.J. Tanner, A structurally conserved water molecule in Rossmann dinucleotide-binding domains, Protein Sci. 11 (2002) 2125–2137.
[26] Y. Cao, T. Jiang, T. Girke, A maximum common substructure-based algorithm for searching and predicting drug-like compounds, Bioinformatics 24 (2008) i366–i374.