2

Protein Structure Comparison Using SAP

William R. Taylor

1. Introduction

In contrast to DNA, proteins exhibit an apparently unlimited variety of structure. This is a necessary requirement of the vast array of differing functions that they perform in the maintainance of life, again, in contrast to the relatively static archival function of DNA. Not only do we observe a bewildering variety of form but even within a common structure, there is variation in the lengths and orientation substructures. Such variation is both a reflection on the very long time periods over which some structures have diverged and also a consequence of the fact that proteins cannot be completely rigid bodies but must have flexibility to accommodate the structural changes that are almost always necessary for them to perform their functions. These aspects make comparing structure and finding structural similarity over long divergence times very difficult. Indeed, computationally, the problem of recognizing similarity is one of three-dimensional pattern recognition, which is a notoriously difficult problem for computers to perform. In this chapter, guidance is provided on the use of a flexible structure comparison method that overcomes many of the problems of comparing protein structures that may exhibit only weak similarity.

1.1. Structural Hierarchy

The aspect of protein structure that makes the comparison problem inherently tractable, is that protein structure is organized in a hierarchy of structural levels, beginning with the basic unit of an amino acid, short stretches of these can adopt one of two semiregular local structures referred to as α and β, being, respectively, helical and extended in nature. The simplicity of having only two secondary-structures (as they are jointly known) is that there are only three (pairwise) combinations of them that can be used to construct proteins, thus

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

giving the three major structural classes: (1) α with α, (2) α with β, and (3) β with β. Various attempts have been made to order and classify the proteins within these groups. One early attempt called a Structural Classification of Proteins (SCOP), is based mainly on visual assessment (`http://scop.mrc-lmb.cam.ac.uk/scop/`), whereas a later classification, called CATH, is based on a more automatic classification, using an earlier version of the program to be described in this chapter. CATH, which stands for the four major levels in this hierarchy — Class, Architecture, Topology (fold family), and Homologous superfamily — also contains a considerable degree of expert added information (`http://www.biochem.ucl.ac.uk/bsm/cath/`). The third main classification is Dali, which is more oriented toward searching for structural similarity using a fast, but rough, similarity method (`http://www2.ebi.ac.uk/dali/`). The resulting similarities are ordered by a variety of measures but it is sometimes difficult to draw the line between true and chance

1.1.1. All-α Proteins

The all-α protein class is dominated by small folds, many of which form a simple bundle with helices running up and down. The interactions between helices are not discrete (in the way that hydrogen bonds in a β-sheet are either there or not), which makes their classification more difficult. Set against this, however, the size of the α-helix (which is generally larger than a β-strand) gives more interatomic contacts with its neighbors (relative to the a β-strand), allowing interactions to be more clearly defined.

1.1.2. All-β Proteins

The all-β proteins are often classified by the number of β-sheets in the structure and the number and direction of β-strands in the sheet. This leads to a fairly rigid classification scheme that can be sensitive to the exact definition of hydrogen-bonds and β-strands. Because they are less rigid than an α-helix, the β-sheets in two proteins can be relatively distorted — often with differing degrees of twist of fragmented or extra strands on the edges of the sheet — making comparisons difficult.

1.1.3. α–β Proteins

The α–β protein class can be subdivided roughly into proteins that exhibit a mainly alternating arrangement of α-helix and β-strands along the sequence and those that have more segregated secondary-structures. The former class includes some large and very regular arrangements of structure (in which a central β-sheet formed of parallel β-strands is covered on both sides by α-helices. Often it is not clear whether this dominance is an evolutionary relic or simply a stable (and so favored) arrangement of secondary-structures. If the latter, then any evolutionary implications based on finding similar substructures must be weak.

1.2. Comparison Methods

The simplest approach to compare two proteins is to move the coordinate set of one structure (as a rigid body) over the other and look for equivalent atoms. This can only be done easily for relatively similar structures and any large scale movement of equivalent substructure can quickly obscure similarities.To avoid this problem, one structure can be broken into fragments; however, this can lead to a series of local comparisons in which the overall global "picture" might be missed.

Both global and local aspects are important and were combined in a number of approaches that used local environments (or views) of the structure to produce an overall equivalence *(1,2)*. These methods determine an alignment of one protein sequence on the other (but based on structure not generic sequence similarity) that may then be used as a set of equivalences to produce a three-dimensional superposition of the structural coordinate sets. Both methods embody the constraint that the structures maintain a linear equivalence, and although this is usually a firm basis for evolutionary relationship, other methods can identify similarity without this constraint. The constraint of the linear ordering of structure is sometimes neglected simply for computational convenience but sometimes through a specific wish to find non topological relationships in structures *(3)*. Although these might elucidate structural principles — such as the mode of packing of an α-helix on a β-sheet (regardless of the β-strand ordering in the sheet) — their application to problems of evolutionary relationships would not be recommended. A major use for such methods, however, is in the identification of local arrangements of groups that constitute an active site or binding pocket, which might well have arisen independently. One of these algorithms based on a geometric hashing algorithm *(1)* is shown in **Fig. 1**.

1.3. Statistical Significance

The statistical significance of structure comparison results is not easily assessed. This is largely because there is no simple model of a random protein (in the same way that random sequences can be simply generated). The approach often taken (e.g., in Dali), is to generate a "random" background distribution from miss-hits on other proteins in the protein structure databank. This suffers from the problem that some of this background might contain unrecognized nonrandom similarities. However, it is a reasonable assumption to assume that these are relatively few.

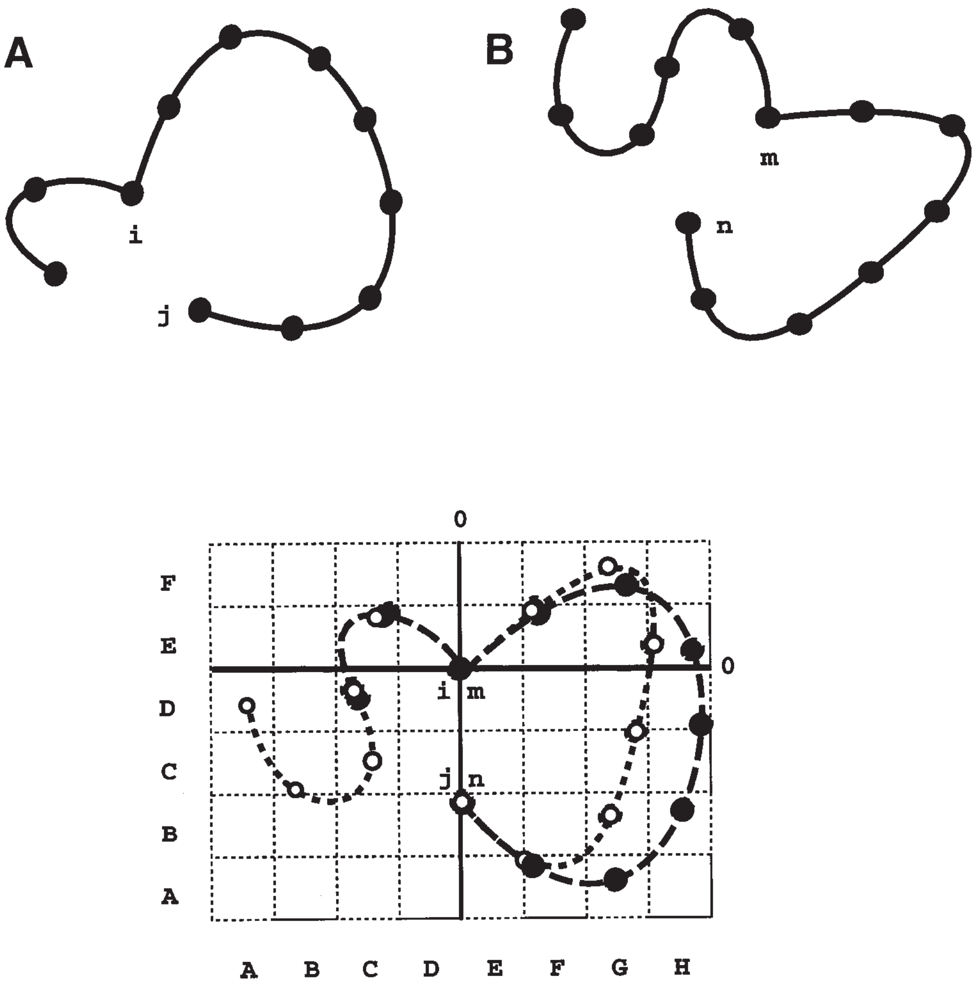

Fig. 1. Geometric hashing algorithm. Two protein structures **(A)** and **(B)** are shown schematically. Two pairs of positions (*i*, *j*) in **(A)** and *m*, *n* in **(B)** are selected. Both structures are centered on the origin of a grid **(C)** at *i* and *m* and orientated by placing a second atom in each structure (*j* and *n*) on the vertical axis which is (coincidentally) the terminal atom of each structure. (In three dimensions, three atoms are required to define a unique orientation.) Atoms in both structures (open and filled circles) are assigned an identifier that is unique to the cell in which they lie (the *hash* key). For simplicity, this is shown as the concatenation of two letters associatedwith the ordinate with the abscissa (XY). For example, atoms in structure **(B)** are assigned identifiers AD, BC, CC, CD, etc. The number of common identifiers between the structures provides a score of similarity. In this example, these are CD, CE, FE, GF, HE, and FA (not counting *i*, *j* and *m*, *n*) giving a score of 6. The process is repeated for all pairs of pairs, or in three dimensions, all triples of triples and the results pooled.

When one is dealing only with unconnected secondary-structure segments, better theoretical distributions can be deduced, allowing very fast filtering of potentially significant similarities ***(5)***. This is the basis underlying the vector alignment search tool (VAST) structure comparison and search method (`http://www.ncbi.nlm.nih.gov/Structure/VAST.vast.html`).

A hybrid approach adopted in the program SAP (described in **Subheading 2.**) in which the protein structure is reversed to form a random model (as this program only uses α-carbons, the secondary-structure remains virtually unaltered under reversal). Further variation is generated by random reconnections of secondary-structure and randomization in the selection phase of the comparison algorithm ***(6)***.

2. `SAP`

The program described here is called SAP (for Structure Alignment Program) and was derived from a related program SSAP, which forms the basis of the CATH classification and was one of the earlier methods based on the use of a local structural view to make an alignment ***(1,7)***. The current version is largely a simplification of its predecessor but is also based on a refined iterative algorithm.

2.1. Structure Alignment Algorithm

The core comparison algorithm underlying both SAP (as well as SSAP, and also some sequence/structure comparison methods ***[8,9]***) is based on the same algorithm as is used to compare protein sequences ***(10)***. As such, insertions and deletions can be easily incorporated, allowing the full range of variation that would be expected between distantly related proteins. When comparing just sequences, one amino acid is (from the point of view of the algorithm) just like any other amino acid of the same type, and as such can be assigned a generic score when matched up (aligned) with another residue. This is not the situation in structure comparison where an amino acid in the core of the protein is fundamentally different from an amino acid on the surface of the protein — even if they are the same amino acid type. This difference in situation can be embodied in a measure of the local structural environment of each residue that can then form the basis of a similarity measure between positions and so allow an alignment algorithm to be applied.

2.1.1. Double Dynamic Programming

The simplest comparison approach would be to have a measure based only on the secondary-structure state and degree of burial of the two residues in the two proteins being compared. Such a simplistic measure, however, could not distinguish two adjacent β-strands, both of which were buried in the core of

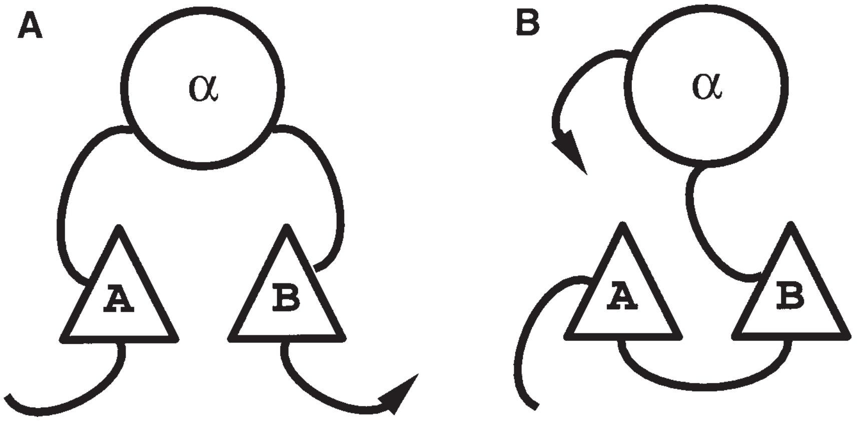

Fig. 2. Two β-strands, A and B, are shown schematically as triangles packing against an α-helix (circle) in two distinct structural fragments, **(A)** (βαβ) and **(B)** (ββα). The packing in the two fragments could be identical but a comparison method that takes account of the topology (or connectivity) of the units would not detect any great similarity.

both proteins. For this, a description of environment is required that can capture the true three-dimensional relationship between residues (referred to as their topological relationship). This is a difficult computational problem and might best be appreciated by the following simple example. Consider two β-strands — A and B — found in both proteins being compared and lying in that order both in the sequence of the two proteins and also in their respective β-sheets. If both pack against an α-helix then, in both proteins, a point on A would be buried by a β-strand to the right and an α-helix above, and would be considered to be in similar environments. If, however, in one protein, the α-helix lay between strand A and B, while in the other protein it lay after strand B, then the two arrangements would not be topologically equivalent (**Fig. 2**).

To discount the contribution of the α-helix in the foregoing example, one must know before assessing the environments of the β-strands that the two helices are not equivalent. Were this known beforehand (for all such elements), then the comparison problem would be solved before the first step was taken. To break this circularity, the following computational device was used: given the assumption (retaining the foregoing example) that strand A in both proteins are equivalent, then how similar can their environments be made to appear while still retaining topological equivalence? If, in the foregoing example, only the B strands could be equivalenced and, consequently, the assumption that the two A strands are equivalent would not be supported strongly. If, on the other hand, the two helices were also equivalent (say both proteins had a βαβ structure), then the equivalence of the A strands would be scored more highly. These scores themselves can be calculated between all pairs of residues and taken to form the basis of a score matrix, from which the "best-of-the-best" set of equivalences can be extracted while still retaining topological equivalence.

The basic alignment (or Dynamic Programming) algorithm is applied at two distinct levels: a low level to find the best score given that residue *i* is equivalent to *j*, and at a high level to select which of all possible pairs form the best alignment. This double level (combined with the basic algorithm) gave rise to the name *"Double Dynamic Programming."* Although previously discussed in terms of secondary-structures, the algorithm operates at the level of individual residues and the environments that are compared consist of interatomic vector sets. To convey some impression of these data, a simplified (two-dimensional) example is shown in **Fig. 3**, in which the construction of the low-level matrix is demonstrated.

2.1.2. Selection and Iteration

The Double Dynamic Programming algorithm described earlier, requires a computation time proportional to the fourth power of the sequence length (for two proteins of equal length) as it performs an alignment for all residue pairs. To circumvent this severe requirement, some simple heuristics were devised based on the principle that comparing the environment of all residue pairs is not necessary. Based on local structure and environment, many residue (indeed most) pairs can be neglected. This selection is based on secondary-structure state (one would not normally want to compare an α-helix with a β-strand) and burial (those with a similar degree of burial are most similar) but a component based on the amino acid identity can also be used, giving any sequence similarity a chance to contribute.

The basic algorithm was implemented, as previously ***(11)***, in an iterative form using the heuristics on the first cycle to make a selection of potentially similar residue pairs. On subsequent cycles, the results of the comparison based on this selection are used to refine the next selection. Previously, a large number of potentially equivalent residue pairs were selected for the initial comparison, and after this only 20 were taken. In the reformulated algorithm, this trend is reversed and an initially small selection (typically 20–30) pairs are selected and gradually increased with each iteration. This initial sparse sampling can, however (just by chance), be unrepresentative of the truly equivalent pairs. To avoid this problem, continuity through the early sparse cycles was maintained in the current algorithm by using the initial rough similarity score matrix (referred to as the *bias* matrix) as a base for incremental revision. As the cycles progress, the selection of pairs becomes increasingly determined by the dominant alignment, approaching (or attaining), by the final cycle, a self-consistent

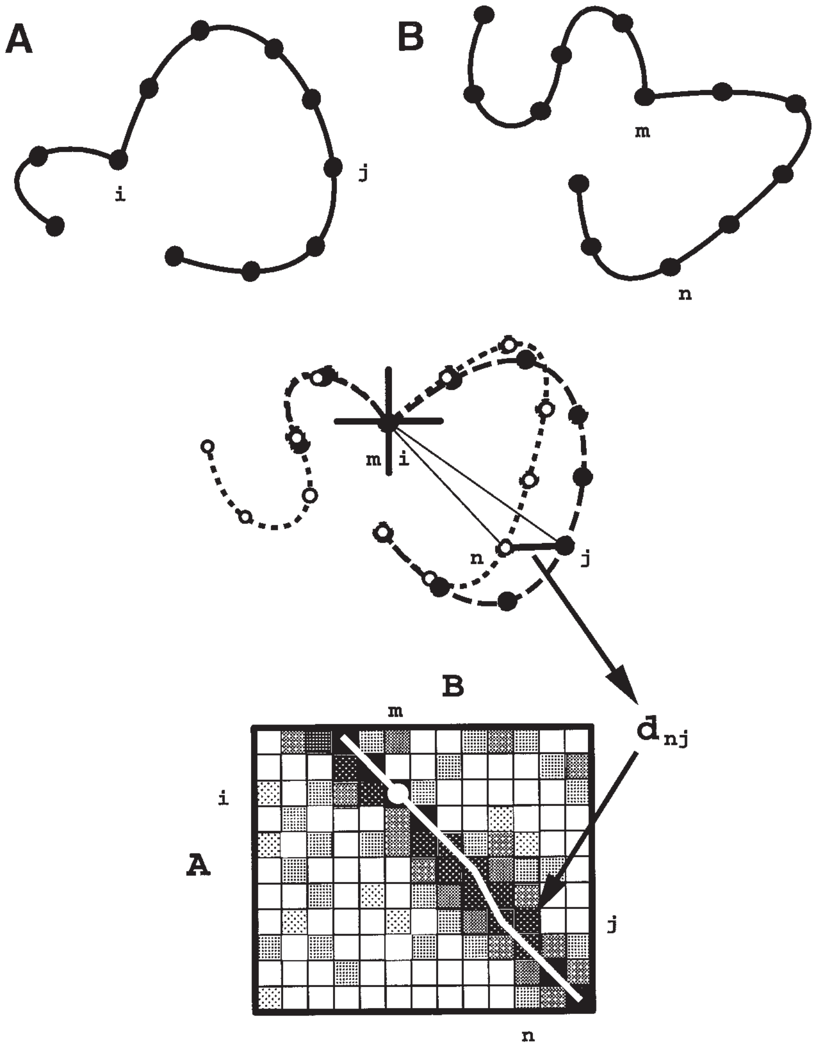

Fig. 3. Two protein structures **(A)** and **(B)** are shown schematically. A pair of positions (*i* in **(A)** and *m* in B) is selected. Both structures are centered on *i* and *m* and orientated by a local measure, such as the α-carbons geometry (indicated by the large cross). In this superposition the relationship between all pairs of atoms (e.g., *n* and *j*) is quantified, either as a simple distance (d_{nj}) or by some more complex function. All pair values are stored in a matrix and an alignment (white trace) found by the dynamic programming algorithm. The arbitrary choice of equating *i* and *m* is circumvented by repeating the process for all possible *i,m* super positions and pooling the results. In the SSAP algorithm, a final alignment is extracted from the pooled results by a second dynamic programming step.

state in which the alignment has been calculated predominantly (or completely) from pairs of residues that lie on the alignment

2.2. Multiple Structure Comparison

The problem of multiple structure comparison, which is often problematic when dealing with structure superposition falls naturally into the structure alignment algorithm described earlier. The approach follows that described for the multiple alignment of protein sequences by MULTAL (*see* Chapter 1). The only difference is that the idea of comparing a point in one subalignment with a point in another requires a measure that is more complex than simply the average pairwise amino acid similarity used in MULTAL.

In SAP the internal description of the structural environments is captured as interatomic vector sets. In a multiple version, in which the positions in two proteins have been equivalenced (say, *i* and *j*), these vector sets are combined to produce an averaged set in which the multiple vector is an average of the two (or more) contributions. Importantly, the coherence of the vector is recorded. If the combined vectors form a tight bunch, then the position is conserved and given a high weight (one for identical vectors), whereas if the vectors point in opposite directions their weight is zero ***(12)***.

The multiple version of SAP is currently being developed for use on a paralyzed Web server called PHASE (funded by the European Union Esprit program) and the state of availability can be checked at the following Web site: `http://mathbio.nimr.mrc.ac.uk/`.

2.3. Treatment of Domains

In most comparison problems, the problem of domains is avoided by dividing the proteins into different domains before any comparison is made. This is also done in the various classification databases (CATH, SCOP, etc.) and also in specialized domain databases such as DDBASE (`http://www-cryst.bioc.cam.ac.uk/~ddbase/`) or 3Dee (`http://circinus.ebi.ac.uk:8080/3Dee/help/help\_intro.html`).

The approach in SAP is to iteratively define domains as the comparison of the structures progresses. This approach is experimental and is not yet generally implemented in the publicly available program (*see* **Subheading 3.**) except for a limited facility that looks for internal domain duplication within a structure. If the same structure is presented twice to SAP, rather than return the obvious, the trivial solution (on the diagonal of the comparison matrix) is masked out in the program, and the ensuing selection of pairs with similar local structure directs the search toward off-diagonal solutions that correspond to internal duplications.

3. Installation and Operation

3.1. Installation

SAP can be downloaded by `ftp` from `http://mathbio.nimr.mrc.ac.uk/`. It is currently specific to Silicon Graphics computers.

1. In the Internet location `http://mathbio.nimr.mrc.ac.uk/`, click on the SAP–FTP name to go to the `sap` directory. Here, two files will be found: `README.txt` and `sap.tar.gz`.
2. Click on `sap.tar.gz` and provide a local directory name into which it canbe copied.
3. Unpack the file in the local directory by typing `gunzip -c sap.tar.gz | tar xvof -`. This will create a director called `sap` containing the program and a subdirectory `data` containing an amino acid similarity matrix.
4. `SAP` can be run by typing the line `sap file1.pdb file2.pdb`. The program can read the full PDB (Protein DataBank) files but needs only the α-carbons

3.2. Operation

A good example on which to test SAP is the two small β/α proteins flavodoxin and the chemotaxis-Y (PDB codes: `4fxn` and `3chy`, respectively). These two proteins have the same fold but no specific sequence similarity. After 10 cycles, SAP should find a solution in which 102 common α-carbons are equivalenced, at which point 84.21% of the selected residue pairs lay on the alignment — in other words, convergence was not complete. This is reported in the output as "`Percents sel on aln`." Of the 102 residues in the alignment, 62.75% of them had been selected as pairs for comparison (reported as "`Percent aln in sel`"). These percentages are a guide to the quality of the comparison but should not be expected to reach 100%. However, if either (or both) fall far below 50%, then caution should be exercised in the interpretation of the results.

The alignment is presented in vertical format with the numbered sequences on either side. Inserted or deleted segments are not printed; these are only apparent from breaks in the residue numbering.* Between the sequences is a numeric value that reflects the degree of similarity between the two local environments (big is more similar). Thus the similar portions are immediately apparent (having values over 100, whereas the dissimilar regions will have values below 10). These numbers are normalized and applied as weights to produce a weighted rigid body superposition of the two structures ***(13)***, for

*The numbering is the sequential numbering in the files as presented and not the attached (PDB) residue number.

```
 file1 = /pdb/brk/3chy.brk
 prot1->Compound = CHE*Y
 file2 = /pdb/brk/4fxn.brk
 prot2->Compound = FLAVODOXIN (SEMIQUINONE FORM)
Mutation Data Matrix (120 PAMs)
ARNDCQEGHILKMFPSTWYVBZX
matrix constant = 8
Cycle 1, 16 residues selected
Cycle 2, 23 residues selected
Cycle 3, 29 residues selected
Cycle 4, 36 residues selected
Cycle 5, 43 residues selected
Cycle 6, 49 residues selected
Cycle 7, 56 residues selected
Cycle 8, 63 residues selected
Cycle 9, 69 residues selected
Cycle 10, 76 residues selected

 score = 2555.236572

Percent sel on aln =   84.21
Percent aln in sel =   62.75

**M    1 105.4   7 F*
**K    2  93.4   8 L**
**I    3 120.6   9 V**
 *V    4 102.9  10 V*
 *Y    5  87.7  11 D**
 *W    6  49.6  12 D*
**T   12   3.9  13 F
 *E   13  27.7  14 S
 *K   14  20.8  15 T
**M   15  22.9  16 M
**A   16  36.4  17 R**
  E   17  23.8  18 R
 *L   18  17.6  19 I
**I   19  27.8  20 V*
 *A   20  34.8  21 R*
  K   21  17.2  22 N
  :                :
 *I  116  16.9 107 V*
 *V  117   2.2 108 K**
  Q  126   0.1 109 P
 *D  127   4.9 110 F*
 *C  128   1.5 111 T*
**I  129   6.2 112 A*
  E  130   6.2 113 A
 *F  131   7.3 114 T
**G  132  15.4 115 L*
 *K  133  14.5 116 E*
 *K  134   8.7 117 E
 *I  135   4.9 118 K*
**A  136   3.4 119 L
  N  137   5.2 120 N
  I  138   0.1 121 K
Weighted RMSd =  2.498 (over 102 atoms)
Un-weighted RMSd =  2.328 over best 43 atoms
Un-weighted RMSd =  4.068 over all matched atoms (102)
```

Fig. 4. Text output from SAP. Two small proteins were compared (`4fxn` and `3chy`). In each alignment, the sequences run vertically and the intervening numeric value is a measure of the strength of each equivalence in the alignment. Solvent exposure is also indicated as "**" = very buried and "*" = partly buried.

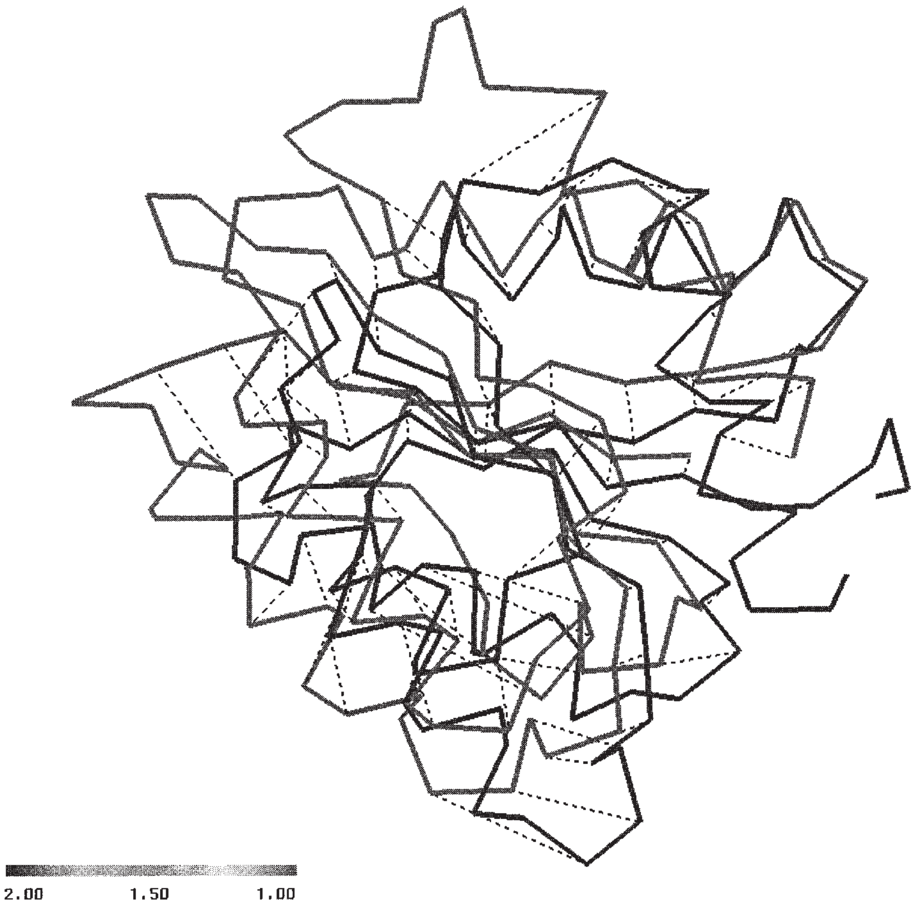

Fig. 5. PROTDRAW display of `4fxn` (gray) on `3chy` (black).The dashed lines connect residues (α-carbons positions) that have been aligned by the SAP program as described in the text.

which the root-mean-square deviation (RMSD) is quoted as "`Weighted RMSD = 2.498 (over 102 atoms)`." Two further values are quoted, which are the unweighted RMSD based on the highest scoring (most locally similar) residue pairs and an unweighted RMSD over all the matched atoms (*see* **Fig. 4**, previous page).

3.3. Visualization

SAP uses the alignment of the two structures and the local similarity values to perform a weighted rotation of one coordinate set onto the other. The result of this transformation is saved in the file `super.pdb` (which is written with each run of the program). Only the α-carbons are saved, but the format is in standard PDB form with each structure separated by a "TER" record and the local similarity score written to the B-value field. Any visualization program (such as RASMOL) can be used to view the results, however, a simple viewing program called PROTDRAW (András Aszódi, unpublished software) is provided in the FTP-file (and should automatically appear in the local directory) (*see* **Fig. 5**). The `README.txt` file should be consulted for a full description of this program.

PROTDRAW has various options (which can be reached by pressing the right mouse button), the most useful of which is to color the structures by B-value and so illuminate their most similar regions. These appear as red with gradations through yellow and green to blue for the least similar parts of the structure. The darkest blue is reserved for unaligned portions of the structures. A second useful feature to visualize the equivalence between the structures is to connect the equivalent residues. SAP does this by writing "fake" hydrogen-bond records to the PDB file and when these are turned-on in PROTDRAW, a white dashed line links equivalent atoms

References

1. Taylor, W. R. and Orengo C. A. (1989) Protein structure alignment. *J. Mol. Biol.* **208,** 1–22.
2. Sali, A. and Blundell T. L. (1990) Definition of general topological equivalence in protein structures: a procedure involving comparison of properties and relationship through simulated annealing and dynamic programming. *J. Mol. Biol.* **212,** 403–428.
3. Holm, L. and Sander, C. (1993) Protein-structure comparison by alignment of distance matrices. *J. Mol. Biol.* **233,** 123–138.
4. Nussinov, R. and Wolfson, H. J. (1991) Efficient detection of 3-dimensional structure motis in biological macromolecules by computer vision techniques. *Proc. Natl. Acad. Sci. USA* **88,** 10,495–10,499.
5. Gibrat, J. F., Madej, T., Spouge, J. L., and Bryant S. H. (1997) The VAST protein structure comparison method. *Biophys. J.* **72,** MP298.
6. Taylor, W. R. (1997) Random models for double dynamic score normalization. *J. Mol. Evol.* **44,** S174-S180. (Special issue in memory of Kimura.)
7. Taylor, W. F. and Orengo, C. A. (1989) A holistic approach to protein structure comparison. *Prot. Eng.* **2,** 505–519.
8. Jones, D. T., Taylor, W. R., and Thornton J. M. (1992) A new approach to protein fold recognition. *Nature* **358,** 86–89.
9. Taylor, W. R. (1997) Multiple sequence threading: an analysis of alignment quality and stability. *J. Mol. Biol.* **269,** 902–943
10. Neeleman, S. B. and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48,** 443–453.

11. Orengo, C. A. and Taylor, W. R. (1990) A rapid method for protein structure alignment. *J. Theor. Biol.* **147,** 517–551.
12. Taylor, W. R., Flores, T. P., and Orengo, C. A. (1994) Multiple protein structure alignment. *Protein Sci.* **3,** 1858–1870.
13. Rippmann, F. and Taylor, W. R. (1991) Visualization of structural similarity in proteins. *J. Mol. Graph.* **9,** 3–16