Structure motif discovery and mining the PDB

Inge Jonassen1,∗, Ingvar Eidhammer1, Darrell Conklin2 and William R. Taylor1, 3

1Department of Informatics, University of Bergen, HIB, N5020 Bergen, Norway, 2ZymoGenetics Inc., 1201 Eastlake Avenue East, Seattle, WA 98102, USA and 3Division of Mathematical Biology, National Institute of Medical Research, Mill Hill, London, UK

Received on March 2, 2001; revised and accepted on September 6, 2001

ABSTRACT

Motivation: Many of the most interesting functional and evolutionary relationships among proteins are so ancient that they cannot be reliably detected through sequence analysis and are apparent only through a comparison of the tertiary structures. The conserved features can often be described as structural motifs consisting of a few single residues or Secondary Structure (SS) elements. Confidence in such motifs is greatly boosted when they are found in more than a pair of proteins.

Results: We describe an algorithm for the automatic discovery of recurring patterns in protein structures. The patterns consist of individual residues having a defined order along the protein’s backbone that come close together in the structure and whose spatial conformations are similar. The residues in a pattern need not be close in the protein’s sequence. The work described in this paper builds on an earlier reported algorithm for motif discovery. This paper describes a significant improvement of the algorithm which makes it very efficient. The improved efficiency allows us to use it for doing unsupervised learning of patterns occurring in small subsets in a large set of structures, a non-redundant subset of the Protein Data Bank (PDB) database of all known protein structures.

Availability: The program is freely available to academia, requests can be sent to Inge.Jonassen@ii.uib.no.

Contact: Inge.Jonassen@ii.uib.no

INTRODUCTION

Structural similarities consisting of a few Secondary Structures (SSs) or residues can define structurally or functionally important elements of the proteins. The relationships are subtle and do not always appear significant when found in pairwise structure comparisons. Standard approaches for analysis of protein structures builds on pairwise comparisons where the pairwise comparisons are done independently, (e.g. Sali and Blundell, 1990; Russel and Barton, 1992; Ding et al., 1994; Taylor et al., 1994; Gerstein and Levitt, 1998), for a comprehensive review see Eidhammer et al. (2000). We have earlier described an approach which allows information from multiple structures to be used simultaneously (Jonassen et al., 1999). In this approach all structures are compared to an external model, a pattern, obviating the need for all against all pairwise comparisons. The approach was implemented in a program called SPPratt. In this paper we follow the same approach and describe a more efficient algorithm, implemented in the program SPPratt2, which allows more challenging discovery problems to be tackled. The new method is able to discover automatically and in an entirely unsupervised fashion patterns shared by as few as two structures in a non-redundant subset of Protein Data Bank (PDB) (Berman et al., 2000). The method produces large number of patterns compliant with the user-definable constraints, and methods are described for removing redundancy in the output pattern set to facilitate the user’s analysis of the output data.

METHODS

Definitions

The local neighbourhood of each residue $r$ in each structure is represented as a string $N S_r$, called a neighbour string. The string encodes all residues in the structure that are within a distance of $d$ Ångstrom from $r$ (typically $d = 10$), including $r$ itself. The residue $r$ is named the anchor of $N S_r$. The residues are encoded by their amino acid type, their SS type (three types are used), and a coordinate set $(x, y, z)$ calculated as the mean of the residue’s side chain atoms. The residues’ order in the neighborhood string is defined by their order along the protein’s backbone. In this paper when giving examples of neighbour strings we write the single letter amino acid code for each of the residues with the anchor underlined.

We then define a packing pattern against which a neighbour string can be matched. A packing pattern consists of a list of elements where each element defines

∗To whom correspondence should be addressed.
a match set (set of allowed amino acids), a set of allowed SS types, and one set of coordinates. Each packing pattern has one unique anchor element. We will write a pattern as the string of single letter amino acid codes (enclosed in brackets for elements where more than one amino acid is allowed) underlining the anchor residue.

A neighbour string \( r_1 \ldots r_n \) is said to match a packing pattern \( P = p_1 \ldots p_n \) if it contains a subsequence \( r_{i_1} \ldots r_{i_k} \) so that the residues have amino acid and SS types included in the match sets of the corresponding pattern elements and so that the anchor residue of the neighbour string is aligned with the pattern’s anchor. For example, \( \text{ACEWGGTGEA} \) matches the packing pattern \( \text{CWGT} \). Also, \( N_Sr \) is said to structurally match \( P \) within \( \phi \) if it is possible to superpose the coordinates of \( N_Sr \) onto the coordinates of \( P \) with a root mean square deviation (RMSd, see Kabsch (1978)) of maximum \( \phi \). A neighbour string that structurally matches a packing pattern within a threshold \( \phi \) describes an occurrence of the pattern. Finally, a pattern which have occurrences in \( k \) structures is said to have support \( k \).

When presenting discovered patterns a sequence pattern can be given consisting of the residues of the packing pattern separated by spacers whose lengths are determined by the sequence separation of the residues involved in the matches. We use the PROSITE (Hofman et al., 1999) notation. See the Section Results for examples.

**ALGORITHMS**

Given a set of \( N \) structures we want to find packing patterns with occurrences in at least \( k \) of the structures, i.e. patterns with support at least \( k \). Rather than devising a method for generating all possible packing patterns, the patterns will be generated as generalizations of neighbour strings from the structures. For example, the neighbour string \( \text{ACEWGGTGEA} \) can be generalized to a large number of (matching) packing patterns, for example \( \text{G}, \text{GG}, \text{WG}, \) and \( \text{CWGT} \). If packing patterns are allowed to have amino acid match sets, the amino acids in the neighbour string can be generalized to match sets. If SS information is to be used, the pattern inherits the types from the neighbour string. The packing pattern derived from a neighbour string will inherit the neighbour string’s coordinate sets.

Geometrical constraints are used to limit the lengths of the neighbour strings while keeping in the strings the potentially most interesting neighbour residues. For residue \( r \) and a neighbour residue \( s \) (within \( d \) Ångstrom), it is calculated whether the side chains ‘face’ each other by calculating a half sphere in the residue’s direction for each of \( r \) and \( s \) and only including \( s \) in the neighbour string of \( r \) if \( s \) is in \( r \)’s half sphere and vice versa. The reasoning behind the constraint is that residues whose side chains are pointing roughly towards each other are more likely to be interacting or to take part in the same active site, and are therefore more likely to be part of the same conserved motif. Neighbour strings with fewer than 4 elements are discarded. See Table 1 for some statistics on the length of neighbour strings depending on \( d \) and use of the half sphere rule.

Starting with one neighbour string (called the probe) a simple depth first search algorithm can be used to find all generalizations of the probe that have occurrences in at least \( k \) structures. The simplest generalization of the probe only contains the probe’s anchor and matches all neighbour strings whose anchor’s amino acid type matches that of the probe’s anchor. Let us call such a pattern \( P \) and its matches \( M_P \). This pattern can be extended (specialized) to form another generalization of the probe by appending a residue \( a \) from the probe, where \( a \) is to the right of the probe’s anchor, forming \( P \cdot a \). For example, if the probe is \( \text{ACEWGGTGEA} \), the first pattern considered is \( \text{G} \) which matches all neighbour strings with glycine as anchor. This pattern can be extended with any of the probe residues to the right of the anchor to form \( \text{GT, GG, GE, or GA} \). The matches to \( P \) are analyzed to see if they can be extended to matches of \( P \cdot a \), and it is checked whether \( P \cdot a \) has sufficient support. If it does, it is again extended in all possible ways by appending elements defined from residues to the right of \( a \) in the probe. In the example, if the pattern \( \text{GE} \) has sufficient support, the (only possible) extension to the right \( \text{GEA} \) would be analyzed next.

At each point in this exploration, all possible pattern extensions to the left are explored. That is, each pattern \( P \) is extended to \( a \cdot P \) in all possible ways where \( a \) is a residue in the probe to the left of the leftmost probe residue already included in the pattern. For example, the anchor pattern \( \text{G} \) could be extended with any residue to the left of the anchor in the probe, i.e. to \( \text{GG, WG, EG, CG, and AG} \). If the pattern \( \text{EG} \) has sufficient support it will be extended further to \( \text{CEG} \) and \( \text{AEG} \). To avoid analyzing the same pattern (the same subsequence of probe residues) twice, once a pattern has been extended to the left, all further extensions will be to the left.

If SS information is used, each residue in the pattern will

<table>
<thead>
<tr>
<th>( d )</th>
<th>Average NS length</th>
<th>Average NS length using constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>18</td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>29</td>
<td>11</td>
</tr>
<tr>
<td>14</td>
<td>42</td>
<td>14</td>
</tr>
<tr>
<td>20</td>
<td>95</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 1. Average neighbour string lengths for different radii (\( d \)-values) and depending on whether the half-sphere constraint (see text) is used. The numbers are calculated for the structures in PDB as defined in the Section Results.
inhibit the SS type of the corresponding probe residue. When matching a pattern, all residues in each match are required to have the same SS types as those of the pattern. As the search proceeds all patterns satisfying the constraints given by the user are output and they are postprocessed by separate programs.

For each pattern, the list of matching neighbour strings is stored. When a pattern $P$ is extended to $P'$ ($P' = P \cdot a$ or $P' = a \cdot P$), each match to $P$ is analyzed to see if it can be extended to match $P'$. For $P \cdot a$ each match to $P$ is extended to include the first, if any, $a$ after the residues matching $P$. The matches to $a \cdot P$ are found in an analogous way. Alternative alignments between the pattern and the neighbour strings are not explored since this could be computationally expensive.

To ensure that any pattern with minimum support potentially can be found in the search, all neighbour strings from the $N - k + 1$ smallest (fewest residues) structures are used as probes. Any pattern with minimum support will have an occurrence in at least one of any subset of $N - k + 1$ structures and the smallest ones are used for efficiency reasons. The search procedure used makes it likely to find the same pattern multiple times since several of the matching neighbour strings may be used as probes. Therefore simple checksums are generated for each identified pattern and when a new pattern is found, its checksum is compared to those of all previously discovered patterns before it is output.

In the search, the structural similarity of each match and the pattern is assessed by calculating the distance based RMSd. Distance based RMSd calculation was used in the search to save computations. Matches whose structural similarity is above the threshold are discarded. When patterns are output, the structural similarity of each pair of matches is calculated using superposition based RMSd using McClellan’s algorithm (1979), and reported together with a description of the matches.

As in Jonassen et al. (1999), for each pattern a score is calculated as the pattern’s information content divided by the maximum RMSd obtained when superposing each pair of matches. As the patterns used here do not contain restrictions on the spacing (in the neighbour strings) between the residues, the information content is only affected by the constraints on amino acid types for each of the residues involved. Since neighbour string matches that do not superpose well (within $\phi$) are discarded already during the search and before the patterns are reported, the matches to patterns found in SPratt2 are more likely to superpose well onto each other than are patterns found in SPratt (Jonassen et al., 1999) where all pattern matches were kept.

The algorithm is guaranteed to generate all patterns having minimum support, when the requirement of structural similarity is removed. When structural similarity is required, the heuristic of only including one alignment between a pattern and a matching neighbour string means that, potentially patterns can be discarded because a misalignment caused the structural similarity to be too low.

**IMPLEMENTATION**

The algorithms described above have been implemented in a C program SPratt2. The program takes as input a file containing PDB IDs and chain identifiers and reads the corresponding (local) PDB files and DSSP output files which are used to assign SS types to each residue (Kabsch and Sander, 1983). The program sorts the structures by size, constructs all the neighbour strings, and uses all neighbour strings from the $N - k + 1$ smallest structures as probes. As the search progresses all identified patterns are output. Separate programs have been implemented to read the output from SPratt2 and to generate representative subsets of the identified patterns, retrieve the SCOP family identifier for each, etc., and to present the results in the form of an html file that can be inspected using a web browser. Subsets of patterns are found by using a greedy cover strategy similar to that used in Brazma et al. (1996). First the pattern having maximum score (or maximum support) is chosen, and all patterns having matches to any of the structures matched by the first pattern are removed. Among the remaining patterns, the best one is chosen using the same criteria, and patterns are chosen in this way until no patterns remain. In this way we obtain a smaller set of patterns with maximum scores or with high support values such that no two patterns match the same structure.

**RESULTS**

SPratt2 was applied to a non-redundant subset of PDB called culledPDB† where the maximum pairwise sequence identity is 30% and only structures with resolution 2.0 or better are included. The set used was generated on 18th May 2000 and contained 779 chains, in the following it is referred to as PDB*. The parameters of SPratt2 were set to let it discover patterns matching at least $k$ chains in PDB*, for $k$ we tried all values between 2 and 20. The radius used was $d = 10$ and the half sphere constraint (see Section Algorithms) was applied. Furthermore, all matches were required to superpose onto the pattern with a (distance based) RMSd of maximum 1.0 Å. For each $k$ value two runs were done, one using SS information and one without. When not using SS information, the computations took between 4 and 7 h on a Sun Ultra 30 workstation with 512 Mb of memory. Including SS information, running times dropped to below 3 h for all values of $k$. Figure 1 shows how the

† see http://www.fccc.edu/research/labs/dunbrack/culledpdb.html
running time and the number of produced patterns depend on the value of \( k \).

As Figure 1b shows, the SPratt2 runs produce large numbers of patterns. Semi-manual analysis of some of these have been carried out. For example, for each pattern the classification of the matching structures in the SCOP database (Murzin et al., 1995) was retrieved automatically. It was found that most of the highest scoring patterns match structures from within the same family or superfamiliy in SCOP. For example, a large number of patterns having matches within the immunoglobulin and serine protease families. While this confirms that the algorithm is able to recover known relationships in PDB, we also wanted to see if SPratt2 is able to find relationships between even more remote structures. More comprehensive analysis of the produced patterns will be performed and presented elsewhere. Here we give some details about two patterns that span different classes (alpha helical packing pattern) and different folds (cystine scaffold) in SCOP. These were among the patterns having highest scores produced in the run performed with minimum support 5 and ignoring SS information.

When SS information is included, the number of discovered patterns is much lower (see Figure 1b). For most of the patterns, all matches are within one superfamiliy. Some patterns do cross superfamilies and these often consist of arrangements of residues having amino acid types commonly found in protein cores such as Isoleucines, Leucines, and Valines. Information about the patterns discovered in the experiments described here are available at the web address http://www.ii.uib.no/~inge/spratt2/.

A small cystine scaffold

Small solvent-exposed beta domains are often held into a structural framework by a network of disulfide bonds, without which they would not be stable in solution. These include small proteinase inhibitors, snake toxins, as well as small extracellular binding domains of receptors.

The packing pattern represented by the sequence motif C-x(4,19)-C-x(5,9)-C-x(4,17)-C was discovered and found to match a small two-disulfide framework within several small beta domains of diverse functions (see Table 2). Though the four cystine residues have a variety of spacings between them within the structural occurrences, they are found to superpose within one Ångstrom RMSd. The global topologies of the matching structures fall into two classes. The first class (e.g. 1bte, 3ebx, 9wga) comprises structures known as cyclic cystine knots (Craik et al., 1999). These are four beta strands held together by three disulfide bonds. The packing pattern captures the bonds between strands 1–3 and 2–4 (strands numbered sequentially from N-terminus). Within this first class is wheat germ agglutinin (9wga), which has four occurrences of the cyclic cystine knot. The algorithm was able to find all four occurrences. The second class (1fl; serine proteinase inhibitor) has a different beta strand topology and the packing pattern connects two strands, and one of these to a loop.

Alpha helical packing pattern

Another high scoring pattern found by the algorithm is represented by the motif V-x(9,223)-L-x(2)-A-x(3)-A (see Table 3). In contrast to the flexible cystine motif, two
The SPratt2 algorithm is significantly more efficient than the SPratt algorithm that in practice was limited to the analysis of relatively small sets of proteins (up to, say, 50) and requiring high support. The increased efficiency is due to several factors. Firstly, in SPratt the discovery of neighbour string patterns was performed using the tool Pratt (Jonassen et al., 1995; Jonassen, 1997). Pratt takes as input sets of unaligned sequences and discovers patterns of the type used in the PROSITE database (Hofman et al., 1999). In SPratt2 the patterns are effectively describing common subsequences (no restrictions on the sequence distance between residues matching pattern elements) which results in SPratt2 exploring a smaller solution space. Also, in SPratt2 the search algorithm has been tailored to the particular application while Pratt is a general tool. Secondly, in SPratt2 the structural similarity of each match and the pattern can be assessed and used to reject matches. This was not possible in SPratt since Pratt is given only the neighbour strings themselves. Thirdly, we have introduced the half sphere constraint which reduces the lengths of the neighbour strings to be analyzed. Finally, SPratt2 has been implemented as one program and its memory usage has been minimized to facilitate larger scale analyses.

The half sphere constraint reduces the average length of neighbour strings to less than half (Table 1) which results in greatly reduced running times (for example from 31 h to less than 3 h for minimum coverage 6 in the experiments above). However, by requiring residues to ‘face’ each other in order to be considered for inclusion in a pattern, there is a possibility that some biologically interesting patterns will not be found. When a user executes SPratt2 he can choose whether to enforce the constraint at all and how strict it should be (angle threshold). In this way the user can decide his/her own balance between computational efficiency and sensitivity.

The efficiency of the SPratt2 algorithm enables mining of the complete set of known protein structures (represented by a non-redundant subset) in an exhaustive and fully automatic manner. In addition to being able to recover known relationships between proteins within families and superfamilies, it has also discovered packing patterns that occur in diverse folds and topologies. The cysteine pattern and the helical packing pattern would be very difficult to induce from sequence information alone. However, in combination with structural information they are revealed from the data in an unsupervised fashion using the algorithm described in this paper.

While encouraging, the results also indicate further exploration, development and refinement of the method. Patterns spanning superfamilies and even fold classes are the most interesting, as these are unlikely to be found by sequence based methods. However, the two patterns presented here do not appear to preserve topology or of the distances in this pattern are exactly conserved.

Inspection of the occurrences of this pattern revealed that the subpattern comprising L, A, A is in all occurrences on the buried face of a helix. The side chain of the Valine in the first position of the pattern faces this helix and is on a beta strand (e.g. in 1fwa) or on another helix (e.g. in 2gdm). Within proteins of similar topology, the position of the Valine is not topologically conserved: it occurs on a beta strand (e.g. in 1fwa) or on another helix (e.g. in 366).

### Table 3. The matches to the pattern represented by the sequence motif V-x(9,223)-L-x(2)-A-x(3)-A

<table>
<thead>
<tr>
<th>Protein</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>L</td>
</tr>
<tr>
<td>2dbm</td>
<td>V11</td>
</tr>
<tr>
<td>1fua</td>
<td>V91</td>
</tr>
<tr>
<td>1gdoA</td>
<td>V131</td>
</tr>
<tr>
<td>1dciA</td>
<td>V105</td>
</tr>
<tr>
<td>1iow</td>
<td>V143</td>
</tr>
<tr>
<td>1qusA</td>
<td>V127</td>
</tr>
<tr>
<td>4pgaA</td>
<td>V95</td>
</tr>
<tr>
<td>1bw9A</td>
<td>V62</td>
</tr>
<tr>
<td>1lam</td>
<td>V248</td>
</tr>
</tbody>
</table>

DISCUSSION

The SPratt2 algorithm presented here together with the previously reported SPratt algorithm (Jonassen et al., 1999) represents a novel approach to discover protein structure motifs. The algorithms are able to discover motifs consisting of single residues, and their primary usage is therefore finding motifs for ligand binding and drug design. For fold recognition it is generally more efficient to work at the SS level. A number of methods have been proposed for comparing structures at this level (e.g. Koch et al., 1996; Holm and Sander, 1995).
even SS. The diverse topologies of the cystine pattern occurrences and the non-conserved SS of the Valine in the helix pattern occurrences clearly illustrate this point. Due to these features, these patterns cannot be used effectively for structural alignment, as can more specific patterns discovered in a supervised fashion (Jonassen et al., 1999). Furthermore, it is clear that a four amino acid pattern, though conserved, is too short and its connecting regions too degenerate to be used for local tertiary structure prediction. Future research could address these specificity issues. A possible avenue towards the sequence specificity problem is to combine short packing patterns into wider conjunctions of patterns. One might also consider weakening the restriction of absolute residue conservation (considering the use of residue sets) but insisting on much longer patterns spanning a larger volume of 3D space. In conjunction with this, the RMSd tolerance could be relaxed for wider conjunctions of patterns. One might also consider specializing our algorithm to deal specifically with cystine packing motifs.

ACKNOWLEDGEMENT

I.J. was supported by grants from the Norwegian Research Council.

REFERENCES


