W334–W339 *Nucleic Acids Research, 2012, Vol. 40, Web Server issue* *Published online 25 May 2012*
doi:10.1093/nar/gks436

Super: a web server to rapidly screen superposable oligopeptide fragments from the protein data bank

James H. Collier[1], Arthur M. Lesk[2], Maria Garcia de la Banda[1,*] and Arun S. Konagurthu[1,*]

[1]Faculty of Information Technology, Monash University, Clayton, Victoria, 3800, Australia and [2]Department of Biochemistry and Molecular Biology, Huck Institutes of Genomics, Proteomics and Bioinformatics, Pennsylvania State University, University Park, PA, 16802, USA

Received February 25, 2012; Revised April 20, 2012; Accepted April 25, 2012

ABSTRACT

Searching for well-fitting 3D oligopeptide fragments within a large collection of protein structures is an important task central to many analyses involving protein structures. This article reports a new web server, `Super`, dedicated to the task of rapidly screening the protein data bank (PDB) to identify all fragments that superpose with a query under a prespecified threshold of root-mean-square deviation (RMSD). `Super` relies on efficiently computing a mathematical bound on the commonly used structural similarity measure, RMSD of superposition. This allows the server to filter out a large proportion of fragments that are unrelated to the query; >99% of the total number of fragments in some cases. For a typical query, `Super` scans the current PDB containing over 80 500 structures (with ~40 million potential oligopeptide fragments to match) in under a minute. `Super` web server is freely accessible from: http://lcb.infotech.monash.edu.au/super.

INTRODUCTION

Searching for occurrences of 3D oligopeptide 'fragments' within a large collection of protein structures is an important and widely used activity in structural biology. In particular, such searches are used to understand the general principles of protein architecture, reveal the mechanism of structural changes as species evolve, determine structural motifs that are reused in the protein universe and infer missing pieces within partially interpreted 3D protein structural models (1,2).

The computational problem of screening for well-fitting (or 'superposable') structural fragments within a database of structures can be seen as a 3D counterpart to the more familiar problem of searching for 1D patterns in a large collection of reference texts (3). However, the 3D version of the problem is significantly more complex and computationally demanding than its 1D analog. Typically, for large collections of structures, such as the protein data bank (PDB) (4), the task involves considering millions of candidate oligopeptide fragments to identify those that are structurally similar to a given query. As the PDB grows rapidly, there is an urgent need for a freely available web resource that can efficiently search through large databases, while guaranteeing that all relevant fragments are found.

The commonly used measure to establish similarity between two oligopeptide fragments (with a given one-to-one correspondence between their peptide residues) is the root-mean-square deviation (RMSD). RMSD gives the 'least' sum-of-square distance between corresponding residues after rigid-body transformation (i.e. rotation and translation) of one fragment over the other. The optimization problem of finding such a spatial transformation has come to be known in the field as the 'superposition problem' (5).

McLachlan (6), Kabsch (7,8) and Kearsley (9), amongst others, gave analytical solutions to the superposition problem. The subsequent literature on this topic focused on correcting some corner cases where the analytical methods fail, and improving the computational efficiency of finding the RMSD of the 'best transformation' (or superposition). Yet, all current methods to find the RMSD of superposition involve the computationally expensive steps of constructing a matrix and performing matrix diagonalization, singular value decomposition, polar decomposition or inversion.

A traditional approach to the screening of large structural databases (such as the PDB) involves the expensive procedure of finding the RMSD of superposition of the query with 'every' fragment derived by sliding along each

*To whom correspondence should be addressed. Tel: +61 3 9905 3227; Fax: +61 3 9905 5159; Email: arun.konagurthu@monash.edu
Correspondence may also be addressed to Maria Garcia de la Banda. Tel: +61 3 9903 1058; Fax: +61 3 9903 1077; Email: maria.garciadelabanda@monash.edu.

and every structure in the collection. When the task is to identify all superposable fragments within a 'prespecified threshold' of RMSD—which is typically the case in structural analyses—the traditional approach entails a large number of unnecessary RMSD computations. To eliminate many of these unnecessary computations, Lesk and colleagues (10,11) described a procedure to find a mathematical lower bound on the RMSD of the superposition of two fragments that can be computed efficiently, without expensive matrix computations. If the computed lower bound exceeds some prespecified threshold, no more computations for those fragments are required, since it can be concluded that the RMSD for those two fragments can 'never' be within the stated threshold, regardless of the spatial transformation applied. Therefore, this procedure is well-suited for searching large databases.

Despite the importance of identifying superposable fragments in many of the analyses undertaken on protein structures, to the best of our knowledge there are no functional web services dedicated solely to the problem of fragment screening. It is important to distinguish between superposition and structural alignment. Structural alignment (12–14) is in fact a more complex problem that involves not only assigning residue-residue correspondences between structures, but also computing the superposition of the aligned substructures. In contrast, for the superposition problem between two fragments, the correspondences are already known and what is sought is only the RMSD of the best superposition under those correspondences.

This article presents Super, a web server to search rapidly through the entire PDB for 'all' superposable fragments whose RMSD of superposition with the query falls within a user-specified threshold. The program driving the server uses a lower bound on RMSD defined by Lesk and colleagues (10,11), and benefits from other mathematical and engineering techniques to drastically reduce the search time, while maintaining the ability to identify all fragments that superpose under the threshold. The combination of a variety of methods applied to the problem has resulted in a significant decrease in search times, often up to two orders of magnitude compared to the traditional approach.

The server offers two useful modes of screening the PDB. In the first, the user provides a contiguous oligopeptide fragment of arbitrary length as query and asks Super to find all fragments in the PDB that superpose with it within a prespecified threshold of RMSD. In the second, the user provides an oligopeptide fragment with a fixed length gap of residues within it. Treating the subfragments on either side of the gap as 'stumps', Super returns all fragments in the database where only the stumps (of the query and database fragments) fit within a RMSD threshold. The latter type of queries are useful to bridge gaps in the incomplete models used by homology-based structure prediction methods.

The interface to Super is simple and interactive. It supports visualizations of the query and database structures, in addition to providing other supporting information useful for structural analyses of proteins.

SUPER WEB SERVER COMPONENTS

Database

Super uses the entire PDB (4) as the default database to screen for superposable fragments. As of April 2012, the PDB is composed of the coordinates of 80 850 structures (containing >200 000 protein chains). Super automatically synchronizes its database with the wwPDB server (http://www.wwpdb.org), on a weekly basis, to ensure it is always up-to-date. Alternatively, the user can choose to screen a smaller non-redundant set of 12 146 structures within the PDB, for which no two structures share >65% sequence similarity.

Super relies solely on the coordinates of C_α atoms defined in the order of residues from N- to C-terminus of each structure (or fragment), ignoring the rest of the coordinate information. Therefore, the PDB is preprocessed and serialized into a compact binary data structure that facilitates efficient searching. (See 'Implementation', section for details.)

Algorithm

Super relies on efficiently computing a lower bound on RMSD of superposition. For two corresponding fragments $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ containing n coordinates each, Lesk and colleagues (10,11) describe a mathematically guaranteed lower bound on the RMSD of superposition as:

$$\mathrm{LB}_{\mathrm{rmsd}}(U, V) = \sqrt{\left(\frac{\sum_{i=1}^{n}(|u_i| - |v_i|)^2}{n}\right)} \leq \mathrm{RMSD}(U, V) \quad (1)$$

where |.| indicates the Euclidean norm of a vector. This lower bound involves basic arithmetic operations that grow linearly in the number of coordinates, and can thus be computed efficiently.

Figure 1 illustrates the conceptual flow of the algorithm used by the web server. Given a query and a user-specified RMSD threshold for screening, the query is scanned over a (query-sized) sliding window across each structure in the PDB. At each such step, the lower bound given in Equation (1) is computed. If the computed lower bound is larger than the threshold, the database fragment can 'never' superpose with the query with an RMSD less than or equal to the threshold. Thus, computing the lower bound allows Super to filter out a large proportion of fragments without having to invoke the expensive operation of finding the exact RMSD of the superposition. When the computed lower bound is within the stated threshold, the exact RMSD between the query and database fragments is computed using the Kearsley's method (9) to solve the superposition problem. This method frames the superposition problem as an eigenvalue decomposition problem of a 4×4 symmetric matrix expressed in the quaternion components of the fragment coordinate vectors. We use the Jacobi's diagonalization algorithm (15) since it decomposes symmetric matrices with excellent numerical stability and has

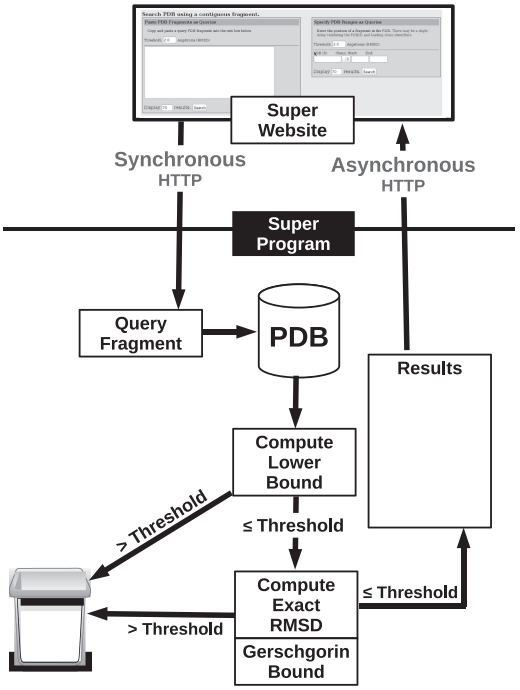

Figure 1. Super web server information flow.

potential for parallelization. To further speed up the diagonalization, we employ the Gerschgorin's circle theorem (16), which allows us to compute the bounds on the eigenvalues of square matrices. Note that the smallest eigenvalue of Kearsley's 4×4 matrix is proportional to the square of the RMSD of the best superposition of two fragments. This enables us to terminate the diagonalization of the matrix whenever the bounds on the eigenvalues exceeds the number of coordinates in the fragment times the square of the RMSD threshold.

In addition, `Super` benefits from many software engineering efforts to whittle away on the constant factors involved in the method's computational complexity. In particular, the screening of a large database requires sequential data accesses and can thus benefit tremendously from spatial locality of reference of coordinate data in the database. This is a common technique in computer science for performance optimization (17). To achieve this, the PDB is preprocessed and serialized as a large contiguous binary file stored on disk. This file is accessed sequentially as a 'memory mapped' data structure. Memory mapping is an efficient alternative to standard file Input/Output on Unix-based systems where the kernel provides an interface to map files on disk directly into the main memory. (POSIX.1 standardizes the system call `mmap()`, which almost all current Unix-based distributions implement.) The crucial advantage of memory mapping is that the user can inform the kernel in advance of how a memory-mapped file is accessed (using the `madvise()` system call). When the kernel recognizes (or is advised) that the accesses to memory mapped blocks are sequential, it automatically optimizes the read-performance by asynchronously caching prefetched (coordinate) data from disk. This engineering design allows `Super` to exploit the memory hierarchy to the maximum, thereby drastically reducing the constant factors involved in the computational complexity of our algorithm for screening large structural databases.

User interface

`Super` supports two forms of queries: (i) screening with a contiguous fragment and (ii) screening with a 'gapped' fragment (containing a stretch of missing residues).

For the first type of query, the user has two input options to specify a query fragment: by pasting the coordinate data of the fragment in the standard Brookhaven PDB format into a text box, or by specifying a `wwPDB` identifier, chain and residue range to identify a fragment in the PDB. In both cases, the user specifies a threshold of RMSD under which the chosen database is screened, and the maximum number of results to be displayed on the results page. Note that the latter parameter merely controls the display of these results (the server always finds all superposable fragments in the PDB) and, thus, it can be changed after the results are displayed. Further, the entire search results can be downloaded as a text file from the results page.

For the second type of query, the user provides the coordinate information of two ends (or stumps) of the query in the Brookhaven PDB format, and specifies a gap length of residues between them. This enables `Super` to identify all fragments in the database where only the stumps (ignoring the gap length of residues in the database fragments) are superposed under the specified RMSD threshold.

The average search time of `Super` to screen the entire PDB is about a minute and the results are presented in a separate results page containing the information of the identified fragment, RMSD, percentage sequence identity and similarity between the query and each match. (See screenshot shown in Figure 2.) Additionally, a 'View' link is available for each identified fragment in the database. Clicking on this link will automatically generate and display the query structure superimposed over the matching PDB structure at the position where the superposition was detected. This is accompanied by a display of the sequences corresponding to the query and the database fragments.

For convenience, the results page can be bookmarked and the results of all searches are saved for later access for up to 30 days after the run. This allows the server to maintain a browser cookie to list all previous searches carried out by the user from a particular browser. Note that users are assigned a unique 26 digit identifier to prevent access to other users' results. A detailed online help page on how to use all features of the server is available at http://lcb.infotech.monash.edu.au/super/help.

Implementation

The implementation of `Super` is split into two components, the web server and the standalone program that drives it. These components are illustrated in Figure 1. `Super` is entirely implemented in the C programming language to take advantage of the low-level power and

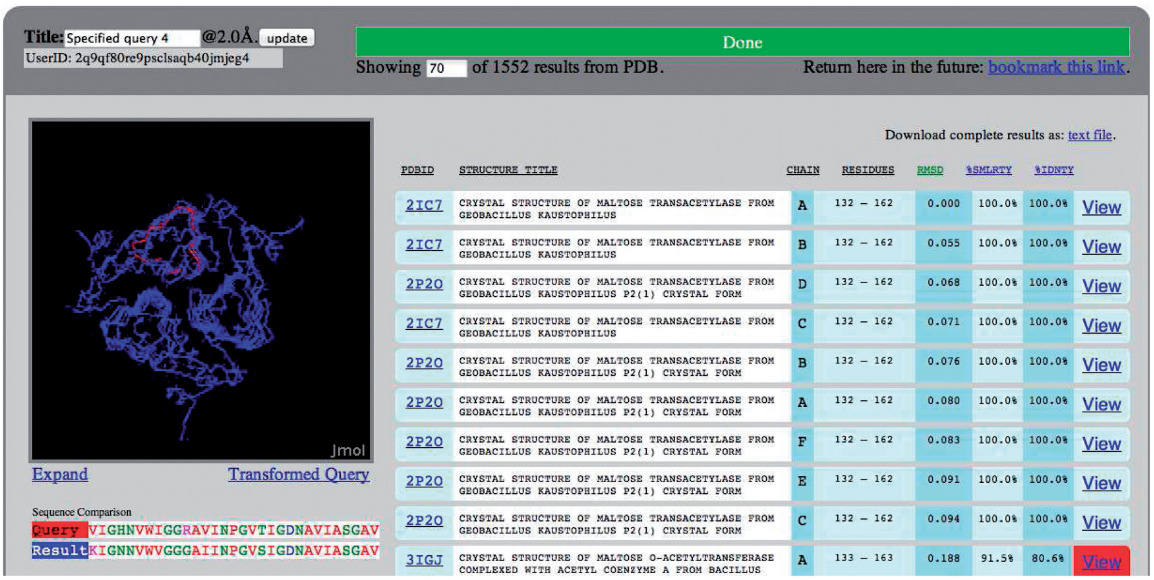

Figure 2. A screenshot of the results window showing some of the returned matches, one of which is selected for viewing in the top-left of the figure.

advanced optimization techniques supported by modern C compilers. The task of preprocessing the PDB is undertaken using a script written in the Python scripting language, using the ProDy (18) package for parsing PDB files. The web server uses a combination of HTML with Javascript on the client-side to support the user interface in the web browser, and PHP on the server-side to process user requests. The visualization of results is enabled through the Jmol (19) Java applet. Some degree of browser platform independence is achieved by conformance to HTML markup standards and the use of the cross-platform Javascript library, Dojo (20). The entire source code of `Super` is freely available for download under the liberal GPLv3 license. The web server source code is available for download under the Affero GPLv3 license.

In the short term, the current version of the server is hosted on a virtual machine at Monash University with a single CPU core and 2 GB of main memory. The operating system running on the machine is Linux 2.6.18 x86_64. As the project evolves, the server will be ported (with the same URL) to a more powerful machine with multiple cores to exploit the program's ability to run in parallel.

COMPLETE PDB SCREENING

Test data sets

The `Super` web server has been extensively tested for accuracy and performance using different fragments from the PDB as queries. In particular, we report on the results obtained for the following two sets of fragments, all of which have lengths varying from 5 to 50 amino acids:

- Set 1: a set of 10 000 frequently occurring structural fragments, mainly composed of standard secondary structures observed in proteins. This set captures the worst case scenario for the program, measuring how well it performs under unfavourable conditions (i.e. when the query is similar to a large number of fragments within the database of structures).
- Set 2: a set of 10 000 query fragments from random positions from the PDB. This set captures the average case scenario, measuring how well the program performs under normal conditions.

The graphs in Figure 3 show the mean, median, maximum and minimum run times obtained for screening the PDB (under varying RMSD thresholds) using all the fragments in each of the above two sets as queries. We observe that the average run time for a query across the two data sets ranges from ~40 s when screening with smaller RMSD thresholds, to between 1 and 2 min for larger RMSD thresholds. In contrast, a non-optimized screening (that is, without any lower bound computations or other optimizations) takes, on average, >75 min per query to scan through the entire PDB (Figure 4). We note that all runs were performed on a single machine, with each of the 10 000 queries in a set running sequentially without exploiting any of the straightforward parallelism that can be achieved in our approach. (Indeed the C program driving `Super` can exploit multiple cores on the CPU and, therefore, can run in parallel mode, reducing the runtime almost in proportion to the number of CPU cores available.) This clearly suggests that the search time for a query is within a reasonable limit for the user to receive the screening results in real time.

Across the two test sets, it can be seen that the time to search the PDB grows as the RMSD threshold increases. At higher values of the threshold, the method permits more fragments to pass through to the computationally expensive step that computes the exact RMSD of superposition between fragments. Whereas at smaller values of the threshold, the method is able to filter out a large proportion (>99%) of fragments without having to compute

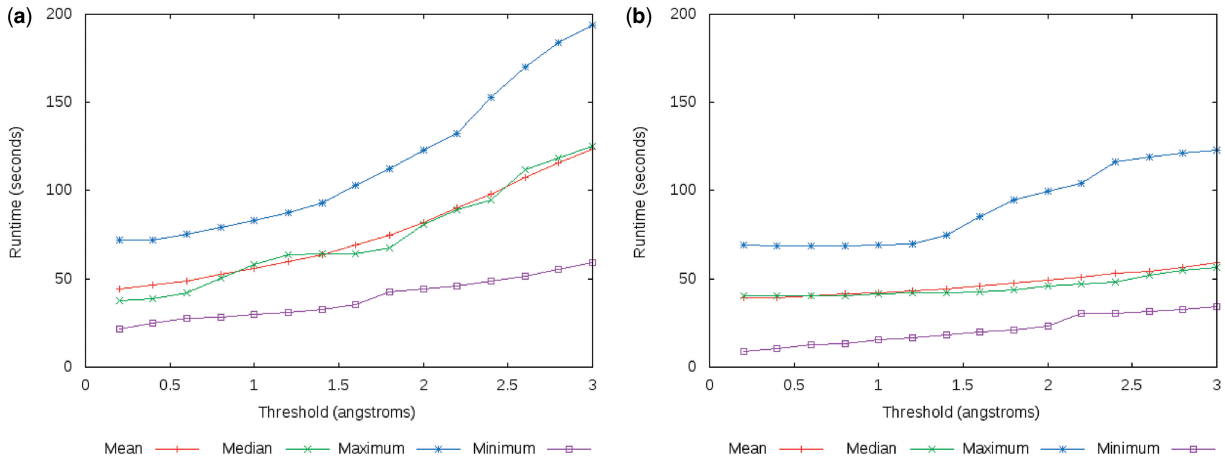

Figure 3. Timing results of screening the entire PDB using `Super` across two data sets containing 10 000 query fragments each (**a**) Set 1 (**b**) Set 2. The graphs show the RMSD threshold on the X-axis and time on the Y-axis, giving the mean, the median, the maximum and the minimum time taken over the 10 000 queries for each set.

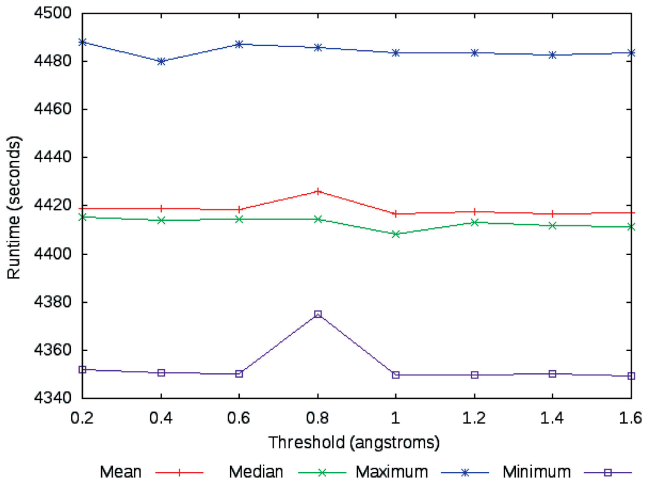

Figure 4. Timing results of screening the entire PDB using 10 000 randomly selected query fragments, running Super without RMSD lower bound filtering and other engineering optimizations. X-axis gives the RMSD threshold and Y-axis gives the runtime.

the exact RMSD. However, even at the large RMSD threshold of 3Å, the average search time is ~120 s for the frequent set (Set 1) and ~60 s for the random set (Set 2). These times are still very small, when considering the effort required to screen the entire PDB.

We emphasize that `Super` guarantees to find 'all' superposable fragments (under a specified threshold) for any given query. To ensure the correctness of the server, `Super` has been compared with exhaustive methods where the exact RMSD is computed for each and every fragment in the PDB.

CONCLUSION

We have implemented a web server, `Super`, to undertake rapid screening of the entire PDB to find all well-fitting fragments under a prespecified threshold of RMSD. To the best of our knowledge, `Super` is the only functional web server dedicated to the task of rapidly screening large structural databases. The server is fast, easy to use and supported by an interactive visualization, giving useful information for the structural analyses of proteins. The source code of the C program driving the web server is available for download from http://gitorious.org/super.

ACKNOWLEDGEMENTS

This web server is the result of J.H.C.'s undergraduate Honours thesis work. The authors acknowledge Roger Gray, Ben Doubleday, Annie Sio and Torsten Seemann for their help in hosting this web service.

FUNDING

Australian Postgraduate Award (Doctoral Research to J.H.C.). A.S.K.'s research is supported by Monash Larkins Fellowship. Funding for open access charge: Monash University.

Conflict of interest statement. None declared.

REFERENCES

1. Rustici,M. and Lesk,A.M. (1994) Three-dimensional searching for recurrent structural motifs in data bases of protein structures. *J. Comput. Biol.*, **1**, 121–132.
2. Lesk,A.M. (1997) Extraction of well-fitting substructures: root-mean-square deviation and the difference distance matrix. *Fold. Des.*, **2**, 12–14.
3. Lesk,A.M. (1979) Detection of three-dimensional patterns of atoms in chemical structures. *Commun. ACM*, **22**, 219–224.
4. Berman,H.M., Westbrook,J., Feng,Z., Gilliland,G., Bhat,T.N., Weissig,H., Shindyalov,I.N. and Bourne,P.E. (2000) The Protein Data Bank. *Nucleic Acids Res.*, **28**, 235–242.
5. Gu,J. and Bourne,P.E. (2009) *Structural Bioinformatics*. Wiley-Blackwell, NJ, USA.

6. McLachlan,A.D. (1972) A mathematical procedure for superimposing atomic coordinates of proteins. *Acta Crystallogr.*, **28**, 656–657.
7. Kabsch,W. (1976) A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr.*, **32**, 922–923.
8. Kabsch,W. (1978) A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. Sect. A*, **34**, 827–828.
9. Kearsley,S.K. (1989) On the orthogonal transformation used for structural comparisons. *Acta Crystallogr. Sect. A*, **45**, 208–210.
10. Tramontano,A. and Lesk,A.M. (1992) Common features of the conformations of antigen-binding loops in immunoglobulins and application to modeling loop conformations. *Proteins*, **13**, 231–245.
11. Lesk,A.M. (1995) Three-dimensional pattern matching in protein structure analysis. In: Zvi,G. and Esko,U. (eds), *Combinatorial Pattern Matching*. Springer, Berlin/Heidelberg, pp. 248–260.
12. Maiti,R., Domselaar,G.H.V., Zhang,H. and Wishart,D.S. (2004) SuperPose: a simple server for sophisticated structural superposition. *Nucleic Acids Res.*, **32**, W590–W594.
13. Sumathi,K., Ananthalakshmi,P., Roshan,M.N. and Sekar,K. (2006) 3dSS: 3D structural superposition. *Nucleic Acids Res.*, **34**, W128–W132.
14. Bauer,R.A., Bourne,P.E., Formella,A., Frömmel,C., Gille,C., Goede,A., Guerler,A., Hoppe,A., Knapp,E.-W., Pöschel,T. *et al.* (2008) Superimposé: a 3D structural superposition server. *Nucleic Acids Res.*, **36**, W47–W54.
15. Slapničar,I. (2007) Symmetric matrix eigenvalue techniques. In: Hogben,L. (ed.), *Handbook of Linear Algebra*. Chapman & Hall/CRC, Boca Raton, USA, pp. 42-1–42-23.
16. Scott,D.S. (1985) On the accuracy of the Gerschgorin circle theorem for bounding the spread of a real symmetric matrix. *Linear Algebra Appl.*, **65**, 147–155.
17. Robinson,J. (2004) *Software Design for Engineers and Scientists*. Newnes, Oxford, USA.
18. Bakan,A., Meireles,L.M. and Bahar,I. (2011) ProDy: protein dynamics inferred from theory and experiments. *Bioinformatics*, **27**, 1575–1577.
19. Hanson,R.M. (2010) Jmol – a paradigm shift in crystallographic visualization. *J. Appl. Crystallogr.*, **43**, 1250–1260.
20. Holzner,S. (2008) *The Dojo Toolkit: Visual Quickstart Guide*. Peachpit Press, Berkeley, USA.