An algorithm to find similar internal sequence repeats

Nirjhar Banerjee[1], N. Chidambarathanu[1], R. Sabarinathan[1], Daliah Michael[1], C. Vasuki Ranjani[1], N. Balakrishnan[2] and K. Sekar[1,]*

[1]Bioinformatics Centre (Centre of Excellence in Structural Biology and Bio-computing);
[2]Supercomputer Education and Research Centre,
Indian Institute of Science, Bangalore 560 012, India

In recent years, identification of sequence patterns has been given immense importance to understand better their significance with respect to genomic organization and evolutionary processes. To this end, an algorithm has been derived to identify all similar sequence repeats present in a protein sequence. The proposed algorithm is useful to correlate the three-dimensional structure of various similar sequence repeats available in the Protein Data Bank against the same sequence repeats present in other databases like SWISS-PROT, PIR and Genome databases.

Keywords. Amino acid substitution, evolutionary divergence, protein sequence, three-dimensional structures.

INTRAGENIC duplication, recombination events and mutation to a slight extent are thought to be the key factors responsible for the formation of sequence repeats[1]. Analysis of protein sequences aids in the discovery of significant patterns and their interpretation with respect to evolutionary processes[2]. Repetition of a small structural unit of a protein sequence confers several advantages on the protein. Variations in the number of orthologues in the protein sequences are evident from the frequent loss and gain of repeats[1,3,4]. Repeats range from single amino acid residue, three residue short tandem repeats (for example, collagen), to the repetition of homologous domains of 100 or more residues (for example, the domain of antibodies). They are further divided into two classes; namely, 'low-complexity' repeats that contain non-uniform amino acid composition and 'high-complexity' repeats that are of longer lengths with complex amino acid composition. Repeats are more common in eukaryotic than in prokaryotic organisms. The increasing complexity of cellular functions in eukaryotic organisms can be accounted from the assembly of repeats[5]. The present aim of the researchers is to see whether the repeats represent past evolutionary duplication events or have arisen due to internal sequence similarity by chance. The replacement of amino acid may lead to assignment of new functions in the protein structure. The replacement of the amino acid 'phenylalanine' with 'tyrosine' or vice-versa occurs based on structural similarity. It is also evident that such replacements are possible even in the case of other structurally similar amino acids like glutamine with glutamate, asparagine with aspartate, lysine with arginine, leucine with isoleucine, valine with threonine and serine with threonine. These substitutions preserve the physicochemical properties of the original residues. There are matrices available in the literature that is concerned with substitution of amino acids, among them PAM (Per cent Accepted Mutation) matrix[6] is the most widely used. PAM (*x*) substitution matrix is a look-up table in which scores for each amino acid substitution has been calculated based on the frequency of that substitution in closely related proteins that have experienced a certain amount (*x*) of evolutionary divergence. The amino acid substitution based on structural similarity, as stated above, is very close to the substitutions predicted based on the PAM matrix scores[6].

To the best of our knowledge, there is no algorithm available in the literature for determining similar repeats in a given sequence. Thus, we have designed an algorithm which is derived from the recent algorithm, FAIR, Finding All Identical Repeats[7]. In the present algorithm, determination of the similar repeats is based on the substitution of structurally (almost) similar amino acids. The proposed algorithm can be effectively used in the analysis of protein sequences, especially to facilitate the study of evolutionary history, structure and function of biomolecules.

The algorithm is designed to find all the similar sequence repeats in a given protein sequence. The list of similar amino acids is given below, where each pair signifies amino acids that are almost structurally similar to each other.

F ↔ Y, Q ↔ E, N ↔ D, K ↔ R, L ↔ I, V ↔ T, S ↔ T.

A string of amino acids is said to be similar to another, if there exists one or more residue(s) in the first string, which is similar (according to the above list) to the corresponding residue(s) in the second string. Thus, for the sequence repeat, KLN, the similar sequence repeats are RLN, RIN, RID, RLD, KID, KIN and KLD. If KLN is generated as a similar repeat for RID, all occurrences of RID will be generated (which are identical repeats) along with every occurrence of the repeat KLN. Thus, in certain cases, the algorithm will also show identical repeats. As stated above, the proposed algorithm is derived from the known algorithm, FAIR[7] and the necessary modifications to the algorithm FAIR are explained in subsequent sections.

Like in the algorithm FAIR, initially the protein sequence is uploaded and stored in a string a1. Then, another string a2 is created which is 'similar' to a1. To be precise, the constituent amino acids of a1 are left

*For correspondence. (e-mail: sekar@physics.iisc.ernet.in) unaltered if they do not have a similar residue but are changed to their similar residues if they have one. In a unique case, ‘T’ is similar to both ‘V’ and ‘S’. In order to address the situation, ‘T’ is replaced by a common letter ‘B’ in the ‘similar’ string created in the vector a2. Subsequently, the algorithm follows the same approach in finding repeats as implemented in FAIR. Only in the case of ‘S’ and ‘V’, the algorithm looks for ‘B’ (which is the common letter assigned earlier) instead of a perfect match. Given here is the code developed to execute the above operation

```
if((a1[i]==a2[j])||((a1[i]=='S')&&(a2[j]=='B'))
||((a1[i]=='V')&&(a2[j]=='B')))
current[j]=previous[j-1] + 1;
```

The above step assigns the ‘current’ length of the repeat to the *j*th element of the vector ‘current’. While performing the next iteration, the above step is repeated by assigning the value of vector ‘current’ to the vector ‘previous’. For example, let the string ‘KLN’ have a similar repeat ‘RID’ such that ‘KLN’ occurs from positions 6 to 8 and ‘RID’ from positions 11 to 13 (see Figure 1 for details). It is noteworthy that the vectors ‘current’ and ‘previous’ start from zero. Hence, after substitution the vectors will be:

a1=.....KLN..RID.....

a2=.....RID..KLN.....

The modifications in the required elements of both the vectors are shown below (as only the upper half of the main diagonal is required, the positions 10 to 12 are shown)

1. Initially: current = 0 and previous = 0;
2. After it finds the first match (K): current[10] = 1 and previous = 0;
3. Then the value of previous is assigned the value of current: previous[10] = 1; current[10] = 1 and rest all = 0;
4. When it finds the character L, current[11] = 2 and the others remain the same;
5. Similarly, after finding the character N, the value of current[12] = 3. Thus, we find that 3 is the length of the repeat and 12th position is the ‘end-point’.

From the given explanation, it is clear that both the position and the length of the repeat are stored by the current vector and the step of pushing the repeat sequences and their positions into vector ‘vsparse’ is exactly similar as implemented in the algorithm, FAIR.

After completion of part A, the ‘end-points’ as well as the length of the repeats are stored and this part can be explained with the help of Figure 1, where the same string ‘KLN’ is taken as an example. As shown in Figure 1, the vector ‘startd’ corresponds to the positions of the starting point of the ‘first sequence’ and the ‘second sequence’. Similarly, the vector ‘endd’ corresponds to the positions of the end points of the two sequences a1 and a2. If a string without a similar component (e.g. ‘AAPPA’) is repeated number of times, the algorithm takes it as an entry to the vector ‘vsparse’. Thus, to eliminate such cases, both the ‘first’ and the ‘second’ sequences are checked whether they are identical using the following code:

```
for(int m=startd[firstseq];
  m<=endd[firstseq]; m++)
  tmp1 += a1[m];
for(int m=startd[secondseq];
  m<=endd[secondseq]; m++)
  tmp2 += a1[m];
if (tmp1==tmp2) goto NIR;
```

NIR takes the control to the beginning of the loop. The manner in which the algorithm stores the repeat sequence and the starting and end points in the vector ‘vsubseq’ is identical to FAIR. Finally, sorting the vector and removing identical entries are performed using the method implemented in the algorithm, FAIR[7]. The output is shown in such a way that beside every repeat position, the corresponding repeat is also shown. The contents of the vectors ‘previous’ and ‘current’ are de-allocated.

The proposed algorithm generates a complete and comprehensive output of all possible similar repeats in a given protein sequence. It is noteworthy that, in the proposed algorithm, the minimum number of residues in a given repeat is defined by the user, thus, adding flexibility to the algorithm. However, keeping the time com-

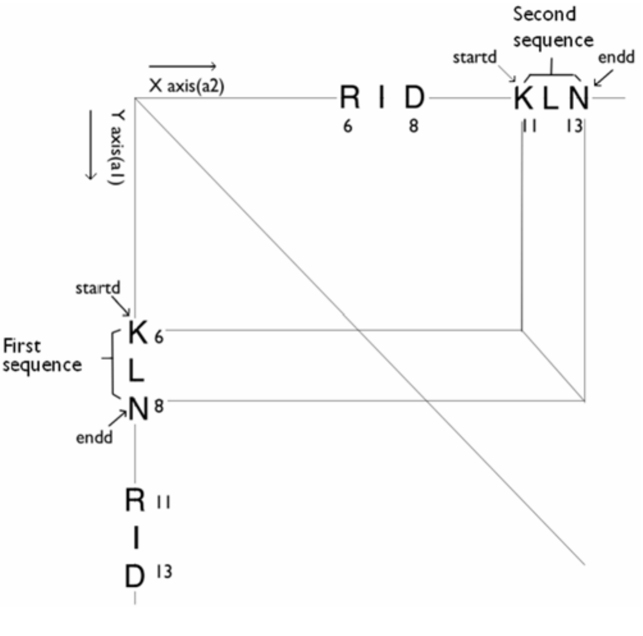

Figure 1. Alignment of subsequences (KLN and RID) to detect similar amino acid sequence repeats, where a1 and a2 are two vectors containing different substrings KLN and RID. ‘Startd’ and ‘endd’ are starting and ending positions of the substring.

plexity in mind, the algorithm performs best with a minimum length of three amino acid residues in a given similar sequence repeat. Interestingly, none of the generated sets of similar repeats are a subset of another. To illustrate this point, suppose the sequence ‘KLNQF’ has repeats such as ‘RIDEY’, it also means that the sequence ‘KLNQ’ has a similar repeat of ‘RIDE’. However, the second repeat will not be shown unless there is an independent repeat of ‘KLNQ’. Thus, the algorithm is designed in such a way that it shows only the non-redundant repeats.

Vectors are used in all instances to store the repeat sequences and their locations. Due to dynamic allocation, the memory required to store the repeats is less, and hence, there is no wastage of space, thereby making the algorithm more efficient in dealing with sequences having large number of amino acid residues. The proposed algorithm follows O(N^2) time complexity in the generalized case, where N is the number of amino acids present in the input sequence.

(1) The algorithm requires the input in FASTA format and the minimum number of amino acids in a similar repeat to be identified. The sample output shown below is for the input protein sequence taken from *Mus musculus* (hypothetical protein). The total number of amino acid residues present in the input sequence is 186. The minimum number of amino acid residues in a similar repeat is set as 150 and the algorithm identifies a significant similar repeat consisting of 157 amino acid residues. It is interesting to note that in a protein sequence with 186 amino acids residues, the algorithm detects two similar repeats of length 157 amino acids. Further, it is evident that these two similar repeats are overlapping each other (residues 10 to 166 and residues 20 to 176).

```
>gi|94370471|ref|XP_996358.1| PREDICTED: hypothetical protein [Mus
musculus]
MHRPLYGGHEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALIAQEDGNTAL
LAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDG
DTALLAQEDGDTALLAQEDGDTALLAQEDGDTALIAQEDGDTALVAVYLGKSCL

Total number of residues present in the input sequence = 186
Number of residues in the repeat = 157

EDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALIAQ
EDGNTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQ
EDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQ
EDGDTAL                                               [ 10 to 166 ]
EDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALIAQEDGNTALLAQ
EDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQ
EDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALLAQEDGDTALIAQ
EDGDTAL                                               [ 20 to 176 ]

Number of sequences uploaded = 1
Number of similar repeats identified = 1
Maximum number of amino acid residues in the repeat = 157
Minimum number of amino acid residues in the repeat = 157
```

(2) Further, to test the efficiency of the proposed algorithm, we have used a protein sequence containing more than 20 times the number of amino acids than that of the sequence used in the given case study. The sample output shown here is for the input protein sequence taken from *paratuberculosis K*-10, a subspecies of *Mycobacterium avium*. The number of amino acid residues present in the sequence is 4170. The minimum number of amino acids in a given similar repeat is set as 10 or more. The proposed algorithm detects nine similar sequence repeats, out of which one significant similar repeat consists of 88 amino acid residues. This repeat is not overlapping unlike the one described here.

```
>gi|41407894|ref|NP_960730.1| Pks12 [Mycobacterium avium subsp.
paratuberculosis K-10]
MVDQLQHATEALRKALVQVERLKRTNRALLERSSEPIAIVGMSCRFPGGVDSPEALWQMVAEGRDV
ISEFPTDRGWDLAALYDPDPDARHKCYVNTGGFVDNVADFDPAFFGIAPSEALAMDPQQRMFLELS
WEALERAG....................................DLVNAALLDDDDE

Total number of residues present in the input sequence = 4170
Number of residues in the repeat = 17

AVSLELADGLGLPVLSV                                 [ 1176 to 1192 ]
AVSIELADGLGLPVLSV                                 [ 3193 to 3209 ]
------------------------------------------------------------------
Number of residues in the repeat = 25
KPGQRVLVHAAAGGVGMAAVQLARH                         [ 1502 to 1526 ]
RPGQRVLVHAAAGGVGMAAVQLARH                         [ 3539 to 3563 ]
------------------------------------------------------------------
Number of residues in the repeat = 20
LATGEPQVLLRDGTVYTARV                              [ 1332 to 1351 ]
LAVGEPQTLLRNGTVYTARV                              [ 3368 to 3387 ]
------------------------------------------------------------------
Number of residues in the repeat = 12
MGFDDDHIGDSR                                      [ 1546 to 1557 ]
MGFDDDHLGDSR                                      [ 3583 to 3594 ]
------------------------------------------------------------------
Number of residues in the repeat = 23
PDKAFQELGFDSLTAVEMRNRLK                           [ 1998 to 2020 ]
PDRAFQELGFDSLTAVEMRNRLK                           [ 4037 to 4059 ]
------------------------------------------------------------------
Number of residues in the repeat = 44
PIAIVGMSCRFPGGVDSPEALWQMVAEGRDVISEFPTDRGWDLA      [ 36 to 79 ]
PIAIVGMSCRFPGGVDSPEALWQMVAEGRDVLSEFPTDRGWDLA      [ 2069 to 2112 ]
------------------------------------------------------------------
Number of residues in the repeat = 12
PLSGVIHAAGVL                                      [ 1771 to 1782 ]
PLTGVIHAAGVL                                      [ 3808 to 3819 ]
------------------------------------------------------------------
Number of residues in the repeat = 88
TSSVASGRVSYVLGLEGPAVSVDTACSSSLVALHMAVQSLRSGE
CDLALAGGATVNATPTVFVEFSRHRGLAPDGRCKAYAGAADGVG      [ 178 to 265 ]
SSSVASGRVSYVLGLEGPAVSVDTACSSSLVALHMAVQSLRSGE
CDLALAGGATVNATPTVFVEFSRHRGLAPDGRCKAYAGAADGTG      [ 2212 to 2299 ]
------------------------------------------------------------------
Number of residues in the repeat = 12
VNASLRLVAPGG                                      [ 1586 to 1597 ]
VDASLRLVAPGG                                      [ 3623 to 3634 ]

Number of sequences uploaded = 1
Number of similar repeats identified = 9
Maximum number of amino acid residues in the repeat = 88
Minimum number of amino acid residues in the repeat = 12
```

Such large similar repeats present in a particular protein sequence could have formed due to substitution of structurally similar amino acids after duplication during the course of evolution. Thus, analysis of these similar repeats would shed light into their biological significance and further enlighten their function and mechanism of formation.

(3) The third case study is also performed to see whether the three-dimensional structures adopted by similar sequence repeats are similar. Thus, a sequence of a known three-dimensional protein molecule is used to identify the similar repeats. The sequence of lymphocyte receptor B protein from *Eptatretus burgeri* [PDB-id: 2o6s][8] is used and the number of residues in the sequence is 208. The minimum number of amino acids in a given similar repeat is set as 10 and above. The proposed algorithm produces three similar repeats of lengths 13, 11 and 19 respectively and the details of the output are shown here.

```
>2o6s_A mol: protein length:208  Variable lymphocyte receptor B
CPSRCSCSGTTVECYSQGRTSVPTGIPAQTTYLDLETNSLKSLPNGVFDELTSLTQLYLGGNKL
QSLPNGVFNKLTSLTYLNLSTNQLQSLPNGVFDKLTQLKELALNTNQLQSLPDGVFDKLTQLKD
LRLYQNQLKSVPDGVFDRLTSLQYIWLHDNPWDCTCPGIRYLSEWINKHSGVVRNSAGSVAPDS
AKCSGSGKPVRSIICP
Total number of residues present in the input sequence = 208

Number of residues in the repeat = 13
LQSLPNGVFNKLT           [ 64 to 76 ]
```

```
LQSLPNGVFDKLT          [ 88 to 100 ]
LQSLPDGVFDKLT          [ 112 to 124 ]
--------------------------------------------------------------
Number of residues in the repeat = 11
PNGVFNKLTSL            [ 68 to 78 ]
PDGVFDRLTSL            [ 140 to 150 ]
--------------------------------------------------------------
Number of residues in the repeat = 19
TNQLQSLPNGVFDKLTQLK    [ 85 to 103 ]
TNQLQSLPDGVFDKLTQLK    [ 109 to 127 ]

Number of sequences uploaded = 1
Number of similar repeats identified = 3
Maximum number of amino acid residues in the repeat = 19
Minimum number of amino acid residues in the repeat = 11
```

Further, the corresponding three-dimensional structures of the above similar repeats are superposed using a web-based program 3d-SS[9]. It is interesting to note that the three-dimensional structures adopted by these similar repeats are almost identical and the results are shown in Figures 2–4. The results reveal a high degree of linkage between the similar sequence repeats and their corresponding three-dimensional structures. However, it is difficult to arrive at a conclusion that all similar sequence repeats available in the known protein structures will have similar three-dimensional structures.

We described here an algorithm to find all similar amino acid sequence repeats present in a given protein

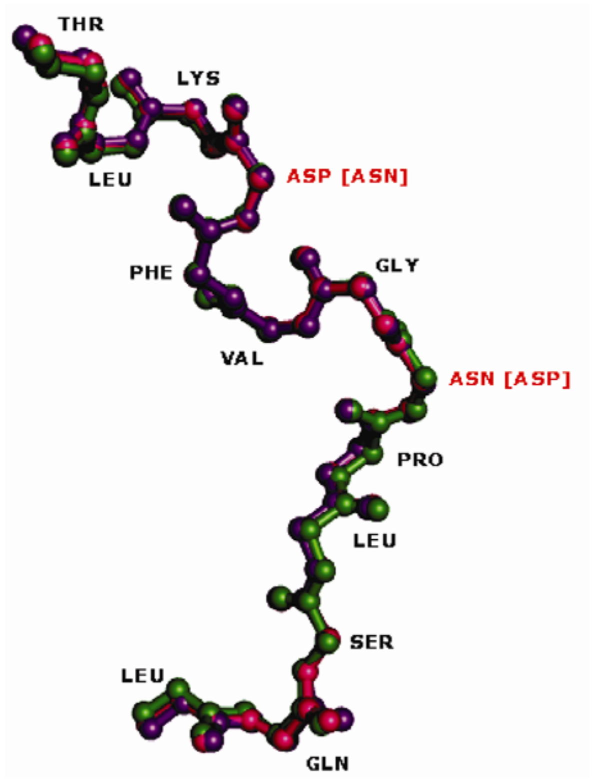

Figure 2. Superposed structures of three similar repeats [‘LQSLPNGVFNKLT’ (64 to 76), ‘LQSLPNGVFDKLT’ (88 to 100) and ‘LQSLPDGVFDKLT’ (112 to 124)] from the structure of lymphocyte receptor B protein (PDB-id: 2o6s). The fixed molecule during superposition is shown in green and the other mobile molecules are shown in red and purple colours respectively. The root mean-square deviations are 0.253 Å and 0.322 Å respectively.

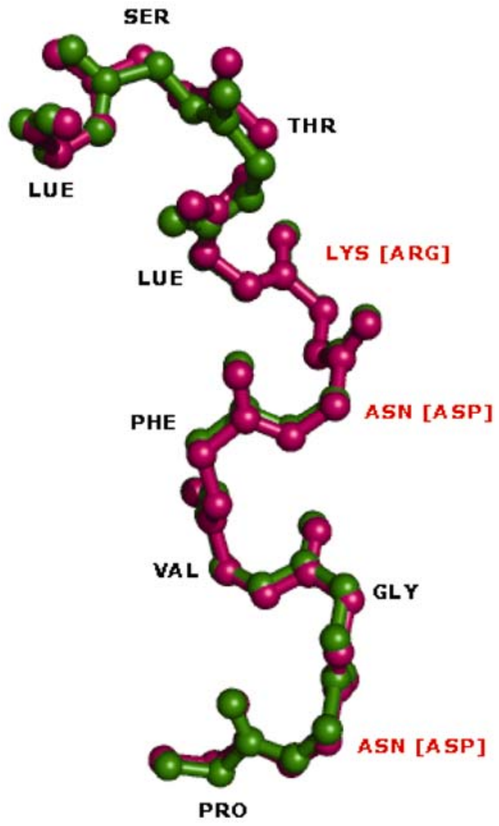

Figure 3. Superposed structures of two similar repeats [‘PNG VFNKLTSL’ (68 to 78) and ‘PDGVFDRLTSL’ (140 to 150)] (PDB-id: 2o6s). The fixed molecule is shown in green and the mobile molecule is shown in red. The root mean-square deviation is 0.544 Å.

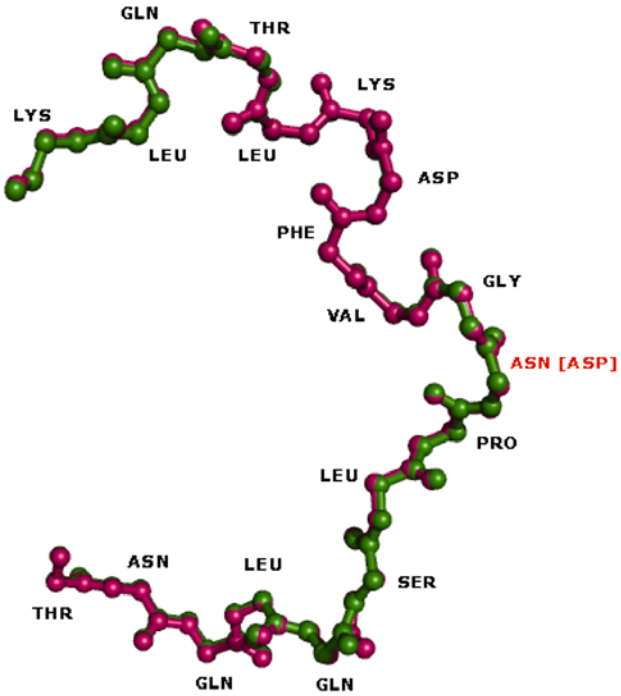

Figure 4. Superposed structures of two similar repeats [‘TNQ LQSLPNGVFDKLTQLK’ (85 to 103) and ‘TNQLQSLPDGVFDKLT QLK’ (109 to 127)] (PDB-id: 2o6s) are shown as green and red colours respectively. The root mean-square deviation is 0.241 Å.

sequence. The algorithm is designed in such a way that the user can upload a single protein sequence or all the protein sequences of a particular gene. The present study reveals that the three-dimensional structures are similar in all three similar sequence repeats identified in a particular protein structure. In order to understand better the sequence–structure relationship, a detailed data-mining study is planned to identify and correlate similar sequence repeats and their three-dimensional structures in all 90% non-homologous protein structures. Such a study would be of use to structural biologists and those who are interested in molecular modelling. In addition, we plan to construct an integrated knowledgebase of similar sequence repeats available in various sequence databases (SWISS-PROT, PIR and Genome database).

1. Andrade, M. A., Perez-Iratxeta, C. and Ponting, C., Protein repeats: structures, functions and evolution. *J. Struct. Biol.*, 2001, **134**, 117–131.
2. Samuel, K., Statistical significance of sequence patterns in proteins. *Curr. Opin. Struct. Biol.*, 1995, **5**, 360–371.
3. Kruglyak, S., Durvett, R. T., Schug, M. D. and Aquadro, C. F., Equilibrium distribution of microsatellite repeat length resulting from a balance between slippage events and point mutations. *Proc. Natl. Acad. Sci.*, 1998, **95**, 10774–10778.
4. Buard, J. and Vergnad, G., Complex recombination events at the hypermutable minisatellite CEB1 (D2S90). *EMBO J.*, 1997, **13**, 3203–3210.
5. Marcotte, E. M., Pellegrini, M., Yeates, T. O. and Eisenberg, D. A., Census of protein repeats. *J. Mol. Biol.*, 1999, **293**, 151–160.
6. Dayhoff, M. O., Schwartz, R. M. and Orcutt, B. C., A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 1978, **5**, 345–352.
7. Banerjee, N., Chidambarathanu, N., Daliah, M., Balakrishnan, N. and Sekar, K., An algorithm to find all identical internal sequence repeats. *Curr. Sci.*, 2008, **95**, 188–195.
8. Kim, H. M., Oh, S. C., Lim, K. J., Kasamatsu, J., Heo, J. Y., Park, B. S., Lee, H., Yoo, O. J., Kasahara, M. and Lee, J. O., Structural diversity of the hagfish variable lymphocyte receptors. *J. Biol. Chem.*, 2007, **282**, 6726–6732.
9. Sumathi, K., Ananthalakshmi, P., Md. Roshan, M. N. A. and Sekar, K., 3dSS: 3-dimensional structural superposition. *Nucleic Acids Res.*, 2006, **34**, W128–W134.

ACKNOWLEDGEMENTS. We gratefully acknowledge the use of the Bioinformatics Centre (DIC) and the Interactive Graphics Based Molecular Modelling (IGBMM) facility. The work is fully supported and funded by the Department of Information Technology (DIT), New Delhi.

Received 17 November 2008; revised accepted 15 September 2009

*For correspondence. (e-mail: nptfd@yahoo.com)