An algorithm to find similar internal sequence repeats

Nirjhar Banerjee1, N. Chidambarathanu1, R. Sabarinathan1, Daliah Michael1, C. Vasuki Ranjani1, N. Balakrishnan2 and K. Sekar1,*,

1Bioinformatics Centre (Centre of Excellence in Structural Biology and Bio-computing); 2Supercomputer 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 repeats1. Analysis of protein sequences aids in the discovery of significant patterns and their interpretation with respect to evolutionary processes2. 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 repeats1,3,4. 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 repeats5. 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 amino acid may lead to assignment of new functions in the protein structure.

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 R,L,N, R,D,N, R,D,I,K,D, K,N and K,L,D. If K,L,N is generated as a similar repeat for R,D, all occurrences of R,D will be generated (which are identical repeats) along with every occurrence of the repeat K,L,N. Thus, in certain cases, the algorithm will also show identical repeats. As stated above, the proposed algorithm is derived from the known algorithm, FAIR7 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

```c
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 jth 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 current is assigned the value of previous: 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 ‘vsubseq’ 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 ‘vsubseq’. Thus, to eliminate such cases, both the ‘first’ and the ‘second’ sequences are checked whether they are identical using the following code:

```c
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)
go to 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, FAIR5. 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.](image-url)
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 \( 'KLNQ' \) 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 numbers of amino acid residues. The proposed algorithm follows \( O(N^2) \) time complexity in the general 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 \textit{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 acid 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).

(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 \textit{paratuberculosis K-10}, a subspecies of \textit{Mycobacterium avium}. The number of amino acid residues present in the sequence is 4170. The minimum number of amino acid residues 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.

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 \textit{Eptatretus burgeri} [PDB-id: 2068] 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.
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...
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).


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

Comparative studies on species richness, diversity and composition of Anogeissus latifolius mixed forests in Phakot and Pathri Rao watersheds of Garhwal Himalaya

Prerna Pokhriyal, Varsha Naithani, Sabyasachi Dasgupta and N. P. Todaria*

Department of Forestry, P.O. Box 59, H.N.B. Garhwal University, Srinagar-Garhwal 246 174, India

The floral diversity is fascinating because of species richness and diverse community structure. Species richness, diversity and composition of plant species were examined in Anogeissus latifolius mixed forests of Pathri Rao and Phakot watersheds in Garhwal Himalaya. Both the watersheds have their own diverse characteristics. A part of Pathri Rao is fully protected as it is part of Rajaji National Park situated in the Siwalik Forest Division, whereas forests in Phakot watershed are reserve forests. Various land-use categories such as cultivated land, scrubland and orchards under fruit trees are available within Phakot watershed.

In this study, a total of 87 species were recorded in Pathri Rao among which 27 were trees, 21 shrubs and 39 herbs whereas a total of 92 species, with 24 trees, 23 shrubs and 45 herbs were present in Phakot watershed. The tree species richness was slightly higher in Pathri Rao whereas shrub and herb diversity was higher in Phakot watershed. Poaceae and Fabaceae were found to be the dominant families in Pathri Rao whereas Poaceae and Asteraceae were the dominant families in Phakot watershed. The study revealed that distribution and species richness pattern in Phakot and Pathri Rao watersheds were more or less similar.

Keywords: Anogeissus latifolius, biodiversity, Himalayan watersheds, species richness.

HIGH biodiversity favours ecological stability, whereas accelerating species-loss could lead to collapse of the ecosystem. Biodiversity is essential for human survival and economic well-being and for the ecosystem function and stability. Human domination of earth’s ecosystem, which is markedly reducing the diversity of species in many habitats worldwide is also accelerating species extinction. The Himalayan ecosystems are under constant threat of mass wasting and erosion caused by depletion of forest cover, unscientific agronomic practices, and degradation of land which have created an overall adverse impact, disturbance and imbalance in the ecosystem. It is to be emphasized that 70–80% of the hill population

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