Journal of Cheminformatics

RESEARCH ARTICLE Open Access

Improved algorithms for quantifying the near symmetry of proteins: complete side chains analysis

Inbal Tuvi-Arad[1*] and Gil Alon[2*]

Abstract

Symmetry of proteins, an important source of their elegant structure and unique functions, is not as perfect as it may seem. In the framework of continuous symmetry, in which symmetry is no longer a binary yes/no property, such imperfections can be quantified and used as a global descriptor of the three-dimensional structure. We present an improved algorithm for calculating the continuous symmetry measure for proteins that takes into account their complete set of atoms including all side chains. Our method takes advantage of the protein sequence and the division into peptides in order to improve the accuracy and efficiency of the calculation over previous methods. The Hungarian algorithm is applied to solve the assignment problem and find the permutation that defines the symmetry operation. Analysis of the symmetry of several sets of protein homomers, with various degrees of rotational symmetry is presented. The new methodology lays the foundations for accurate, efficient and reliable large scale symmetry analysis of protein structure and can be used as a collective variable that describes changes of the protein geometry along various processes, both at the backbone level and for the complete protein structure.

Keywords: Protein structure, Side chains, Symmetry, Chirality, RMSD, Hungarian algorithm, Permutations, Molecular descriptors

Introduction

Symmetry offers several advantages for the evolution, oligomerization and function of proteins [1–3]. It has been shown that symmetry leads to a reduction of errors in the process of protein synthesis, especially when long peptide chains are involved [1, 4]. Symmetry tends to increase the effectiveness of allosteric regulation, e.g., by the Monod–Wyman–Changeux model of allostery (also referred to as the symmetry model) [5]. Synthesizing a symmetric structure requires less information for coding the protein, therefore may lead to faster processes [1]. Usually closed symmetric systems tend to have lower energy than asymmetric ones as the interactions between the subunits are maximized due to the symmetry. Consequently symmetry could make proteins more stable and minimize unwanted excessive aggregation [6]. Indeed, symmetry is a characteristic of many protein structures [7]. Searching the Research Collaboratory for Structural Bioinformatics Protein Data Bank (RCSB PDB) [8, 9] for symmetric structures reveals that within ca. 145,000 structures, around 40% are symmetric, ~30% have cyclic symmetry with one rotation axis, and the other 10% have higher symmetry (e.g., dihedral, octahedral, icosahedral, etc.). Moreover, out of ~55,000 homooligomers in the database, 96% are symmetric.

Nevertheless, research shows time and again that protein symmetry is far from being perfect. Such imperfections have been related to several factors, among which are the function of the protein, thermodynamic considerations, and experimental conditions [10–12]. Quantification of these imperfections is still in its very beginning. The pioneering work of Zabrodsky Peleg and Avnir [13] introduced the Continuous Symmetry

*Correspondence: inbaltu@openu.ac.il; gilal@openu.ac.il
[1] Department of Natural Sciences, The Open University of Israel, 4353701 Raanana, Israel
[2] Department of Mathematics and Computer Science, The Open University of Israel, 4353701 Raanana, Israel

Measure (CSM)—a structural descriptor that translates the full collection of geometrical parameters that define a molecular structure into a single number that measures the distance of that structure from its nearest symmetric counterpart with respect to a given cyclic point group *G*. The group *G* can be generated either by a proper or an improper rotation. The latter case allows a calculation of the Continuous Chirality Measure (CCM) [14], as an important feature of the general CSM framework. The task of computing the CSM requires the solution of an optimization problem, in which the parameters are both a permutation of the molecule's atoms and a three dimensional direction vector [15]. For macromolecules, the huge number of possible permutations makes the calculation (and even approximation) of the CSM a nontrivial algorithmic problem.

The first step towards an efficient method for calculating the CSM was the work of Pinsky et al. [15] who developed an efficient method for finding the optimal direction vector, for any given permutation. They also introduced a division of the molecule's atom into certain equivalence classes, based on the atoms' type and the connectivity map of the molecule [16]. This division narrows down the realm of possibilities for the atom permutation, but the number of permutations remains too large to enumerate, except for very small molecules.

A partial solution to this problem was given in the work of Dryzun et al. [17] who were the first to introduce an approximate method for the calculation of the CSM. Starting with a reasonable guess of the direction vector, a sequence of iterative steps is performed where in each step a new atom permutation is calculated from the direction vector, and a new direction vector is calculated from that permutation, until convergence is reached. While the method of calculating the vector from the permutation (similar to the strategy of Pinsky et al. [15]) is both efficient and exact, as we recently proved [16], the method of calculating the permutation from the direction vector uses a greedy algorithm to find the permutation, which is relatively fast, yet generally provides only a crude approximation as compared with the exact algorithm. Nevertheless, the method provided the first practical tool for estimating the CSM of proteins, macromolecules and nanomaterials [17].

In a somewhat different direction, we have recently developed an algorithm for an exact calculation of the CSM [16], which is feasible for small to medium sized molecules. This algorithm takes into account all the information from the connectivity map of the molecule, and enumerates only the permutations that preserve the molecule's bonding structure.

All of the methods described above are insufficient for estimating the symmetry of proteins. For the exact method [15, 16], the number of atoms, and consequently the number of permissible permutations are simply too large for the algorithms to complete the calculation in a reasonable time. The approximate method of Dryzun et al. [17] is also ineffective for this case, because of accumulated inaccuracies due to the approximate scheme and the use of the greedy algorithm. Furthermore, disregarding the peptide structure produces symmetric structures that mix atoms from different peptides and violate chemical rules. Nevertheless, assuming the permutation is dictated by the serial numbers, that is, each atom is permuted to an atom from another peptide with the same sequence number, provides a good estimation of the CSM of protein homomers. Recently, Bonjack-Shterengartz and Avnir applied this strategy and showed the effectiveness of using the CSM to describe the near symmetry of proteins [10, 11]. As will be shown here, this method limits the precision of the CSM calculation, especially when the number of peptides increases.

Other methods for the estimation of the protein symmetry are discussed in the literature [18–27]. These are generally based on quaternary structure alignment algorithms that involve the superposition of two peptides one over each other, while estimating their alignment by either root mean square deviation (RMSD) of matching α-carbons, or by a related scoring formula (e.g., the combinatorial extension (CE) score [28] or the template modeling (TM) score [18, 29]). However, these methods generally ignore the geometry of the side chains and therefore do not attempt to find the true permutation of the atoms.

In this paper we describe a new method for estimating the CSM for proteins. Our method relies on the properties of the protein sequence and the division into peptides, and presents algorithmic improvements as compared with previous methods. Specifically, our strategy introduces three improvements over the method of Dryzun et al. [17]: (a) We take into consideration the information of the protein sequence. This allows us to refine the division of atoms into equivalence groups, such that only atoms of the same identity, residue type and residue sequence number can be interchanged. (b) The greedy algorithm for calculating the permutation in the iterative step is replaced with the Hungarian algorithm, which is known to guarantee to yield an optimal solution of the related assignment problem [30, 31]. (c) We take into considerations the division of the molecule into peptides. Our algorithm distinguishes between the permutation of the peptides and the permutation of the atoms in the peptides, thus making sure that the resulting permutation will not mix atoms from different peptides. The Hungarian algorithm is applied at this stage as well, to find the optimal peptide permutation. These

improvements lead to a tremendous increase in the accuracy and speed of the calculation, and turn the CSM into a robust methodology to describe protein structure, that can be used both for homomers as well as internal symmetry investigation of protein domains.

Methodology

Review of the CSM

Let us briefly review the fundamentals of the CSM methodology [13, 15, 16]. We consider a given molecule A of N atoms, and a point group symmetry G of order n, which can be of the type C_n or S_n. Let $\mathbf{Q} = \{\mathbf{Q}_k : 1 \le k \le N\}$ be the set of coordinate vectors of the molecule's atoms, and let $\mathbf{Q}_0 = \frac{1}{N}\sum_{k=1}^{N}\mathbf{Q}_k$ be its geometric center of mass. We are looking for a symmetry operation T, which generates a cyclic point group of type G. Note that T is a rotation (either proper or improper) by an angle of $360^\circ/n$. In both cases, T is determined by a 3-dimensional direction vector, which we denote by v_{sym}. The continuous symmetry measure (CSM) is defined by $S(G) = 100 \cdot M(G)/N(G)$, where

$$M(G) = \min\left[\sum_{k=1}^{N}|\mathbf{Q}_k - \mathbf{P}_k|^2\right]; \quad N(G) = \sum_{k=1}^{N}|\mathbf{Q}_k - \mathbf{Q}_0|^2 \tag{1}$$

and the minimum is over all the symmetric (i.e. T-invariant) structures $\{\mathbf{P}_k : 1 \le k \le N\}$ and all possible direction vectors v_{sym}. Each symmetric structure $\{\mathbf{P}_k\}$ induces a permutation π on the set of atoms $\{1, 2, \ldots, N\}$, defined by the relation

$$T\mathbf{P}_k = \mathbf{P}_{\pi(k)} \text{ for } 1 \le k \le N \tag{2}$$

A symmetric structure $\{\mathbf{P}_k\}$ which minimizes Eq. (1) is determined by T and the permutation π, via the relation

$$\mathbf{P}_k = \frac{1}{n}\sum_{i=1}^{n} T^{-i}\mathbf{Q}_{\pi^i(k)} \tag{3}$$

Therefore, the calculation of the CSM and the nearest symmetric structure amount to finding the vector v_{sym} and the permutation π which minimize Eq. (1), or equivalently, attain the minimum

$$M(G) = \frac{1}{2n}\min\sum_{i=1}^{n}\sum_{k=1}^{N}\left|T^i\mathbf{Q}_k - \mathbf{Q}_{\pi^i(k)}\right|^2 \tag{4}$$

The permutation π must satisfy a few requirements:

(a) Since T is a generator of the group G, and π is related to T by Eq. (2), the cycles of π can only be of size 1, 2, or n (2 is only allowed when $G = S_n$ or $G = C_2$).

(b) π must preserve the structure of the original molecule. This means that π only permutes atoms of the same type, and that π neither breaks the bonds between atoms, nor creates new ones.

A method for an efficient enumeration of the permutations satisfying these conditions was described in our previous publication [16]. However, for proteins, such an enumeration is impossible as there are just too many such permutations. In earlier implementations of the CSM calculation [15, 17], the structure preserving condition was not enforced, but rather a weaker condition was used: The molecule's atoms were divided into equivalence classes $\{\mathcal{C}_1, \mathcal{C}_2, \ldots\}$. The division was determined by an iterative process in which the initial division is deduced from the atom types, and further refinements were based on neighboring atoms in the connectivity map of the molecule. Finally, the permutation was required to preserve the equivalence classes.

Estimating the CSM for large structures

The approximate algorithm of Dryzun et al. [17] is based on an iterative calculation. One begins with a reasonable guess of the direction vector v_{sym}; and then, at each iterative step, a permutation is calculated from the current vector, and a new vector is calculated from that permutation. This is repeated until either the process converges (the value of v_{sym} stabilizes) or too many iterations have passed. The initial guess of the direction vector is found by performing linear regression on the set of centers of mass of the equivalence classes $\{\mathcal{C}_1, \mathcal{C}_2, \ldots\}$. This is reasonable because for perfectly symmetric molecules, the symmetry operation preserve these centers of mass. The permutation is calculated in each iterative step by a greedy algorithm: In each step, an atom i and a permutation value $\pi(i) = j$ are chosen, such that: (a) j is in the same equivalence class as i. (b) The atom i has not yet been assigned a permutation value and the atom j has not yet been assigned as a permutation value. (c) The distance $|T\mathbf{Q}_i - \mathbf{Q}_j|$ is minimal among such pairs i,j. The direction vector v_{sym} is calculated from the permutation π by observing that given π, the solution of Eq. (4) is a quadratic optimization problem, which can be efficiently and accurately solved using Lagrange multipliers and matrix eigenvalues [15].

The above algorithm for finding the permutation has the advantage of a reasonable running time, which makes it feasible for large molecules. However, it is not always accurate since the greedy algorithm does not take into account the interaction between the choices of permutation values for various atoms. An example of

the inaccuracy created by the greedy algorithm even in the case of two atoms is given as Additional file 1.

Finding the permutation with the Hungarian algorithm

Our first improvement of the approximate method, which is not specific for proteins, focuses on the calculation of the permutation in the iterative step. We are given a direction vector v_{sym} (determined in the previous step), and a symmetry operation T. For each equivalence class $\mathcal{C}_i$, consisting of the atoms $\{a_1,\ldots,a_k\}$ we define a $k\times k$ matrix A, whose elements, A_{ij}, are given by:

$$A_{ij}=\left|T\mathbf{Q}_{a_i}-\mathbf{Q}_{a_j}\right|^2 \quad \text{for} \quad 1\le i\le k, \quad 1\le j\le k \tag{5}$$

Our algorithm chooses the permutation π of the values $\{a_1,\ldots,a_k\}$, defined by $\pi(a_i)=a_{\mu(i)}$ (for $i=1,\ldots,k$), where μ is the permutation of $\{1,\ldots,k\}$ for which the sum $\sum_{i=1}^{k}A_{i\mu(i)}$ is minimal. Therefore, μ is the solution of *the assignment problem* for the matrix A [30]. For this problem, there is an efficient algorithm—the so-called Hungarian Algorithm, which finds the optimal solution in time $O(k^3)$ [30]. We provide further information about the assignment problem and its well-known solution in the Additional file 1.

Our revised method consists of forming the matrix A for each equivalence class, and obtaining the permutation values in each equivalence class by running the Hungarian algorithm on this matrix.

It should be noted that our algorithm does not take into account the restriction on the cycle structure of the permutation, described above. To the best of our knowledge, there is no efficient solution to the assignment problem under such cycle structure constraints. We also note that our method for finding the permutation (like the method of Pinsky et al. [32]) aims to minimize the term in Eq. (4) corresponding to $i=1$. This has proved to be a good approximation of the minimizer of the entire sum in Eq. (4).

Reducing equivalence groups: the "use sequence" algorithm

It is intuitively clear, and practically verifiable, that a crucial factor in the accuracy and efficiency of the approximate algorithm (with the Hungarian method improvement) is the size of the atom equivalence classes: The algorithm has better performance when the classes are small. For proteins, we can greatly refine the initial division into classes by using the information of the protein sequence. We assign different classes to pairs of atoms which differ in their chemical identity, residue type or sequence number (excluding the remoteness indicator). Consequently the size of the equivalence classes is determined by either the number of peptides (for protein backbone atoms and most of the side chains atoms as well), twice this number for atoms that differ only by their remoteness indicator (e.g., the two C_γ atoms of Val), or three times this number if hydrogen atoms are taken into account.

The many chains algorithm

We now describe a further improvement of the algorithm, in which we make sure that the permutation does not break the peptides, but rather, carry each peptide in its entirety to another peptide. This is achieved by performing two levels of implementation of the Hungarian algorithm. The lower level calculates the mapping of the atoms in a peptide, and the higher level calculates the permutation of the peptides.

Let us denote the peptides by $\{\mathsf{P}_1,\ldots,\mathsf{P}_\ell\}$. In the iterative step of the approximate algorithm, given the symmetry operation T, we calculate for each pair of peptides $\mathsf{P}_i,\mathsf{P}_j$, the following minimum:

$$B_{i,j}=\min_{\nu}\sum_{u=1}^{M}\left|TU_i-V_{\nu(i)}\right|^2 \tag{6}$$

where $\{U_1,\ldots,U_M\}$ are the atoms of P_i, $\{V_1,\ldots,V_M\}$ are the atoms of P_j, and the minimum is over all the permutations ν of $\{1,\ldots,M\}$ which preserve the equivalence classes. The value of $B_{i,j}$ and the minimizing permutation ν is calculated by the Hungarian algorithm as explained

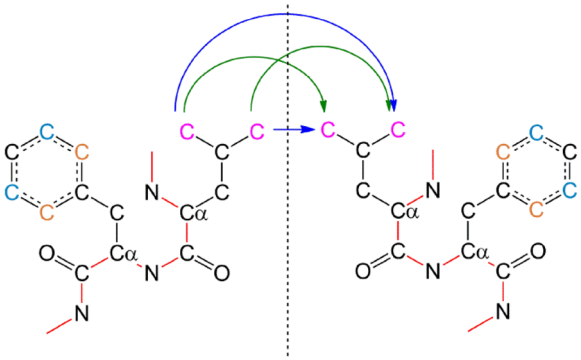

Fig. 1 Atoms' permutation for a two-residue (Leu-Phe) section of a homodimer. Black atoms form equivalence classes of two atoms each (one from each peptide). Orange, blue and magenta carbon atoms represent carbon atoms that differ only by their remoteness indicator, each couple forms an equivalence group of four atoms. Red bonds represent the protein backbone. Two choices for permuting the Leu-C_δ carbon atoms (magenta) are illustrated with blue and green arrows

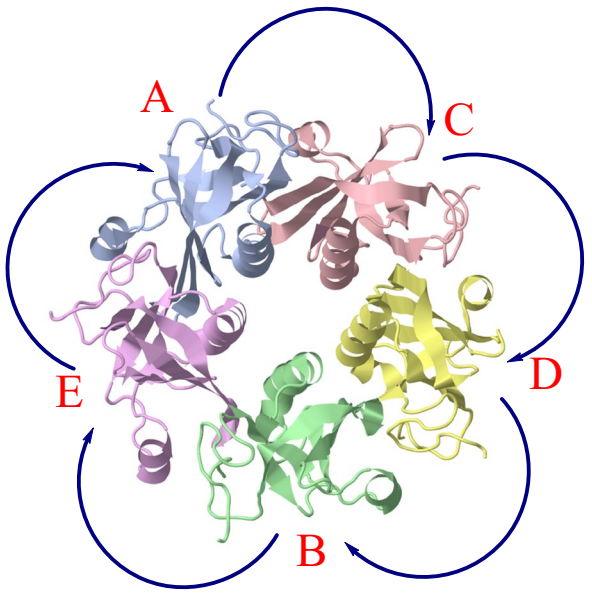

Fig. 2 Peptides' permutation in AB_5 toxin from *E. coli* (PDB ID: 3DWA) fits the peptide permutation order: A → C → D → B → E. $S(C_5) = 0.0410$. Peptides' labels are taken from the PDB file

above. We call this stage the lower level application of the Hungarian algorithm—see Fig. 1.

Applying the Hungarian algorithm to the matrix $B = (B_{i,j})$ (this is the higher level application of the Hungarian algorithm, see Fig. 2) results with the optimal permutation of the peptides. Given the permutation of the peptides, and using the permutation calculated in the lower level, we obtain the full permutation of the molecule's atoms. This permutation has the desired property of maintaining the peptides structure as well as the protein sequence, and is the optimal one among all such permutations.

Testing methodology

Creating a database of the proteins: selection criteria

To test the effectiveness of our code we applied it to several sets of protein homomers. The coordinates of the proteins of each set were extracted from the RCSB website [8, 9]. In order to assure that only high quality proteins with minimal statistical bias will be selected [33, 34], several criteria for filtering the proteins were applied: (a) The experimental method was X-ray crystallography with a resolution of 2.5 Å or better, equivalent to at least a "Good" grade as defined by FirstGlance in Jmol [35]; (b) Only homomeric proteins were chosen in which the asymmetric unit contained all chains of the protein required to create a symmetric structure, that is, a biological assembly identical to the asymmetric unit exist, as defined by the transformation matrix in remark 350 of the PDB file. (c) DNA, RNA or hybrid chains were filtered out. (d) Based on R_{free} values the proteins were assigned with an R_{free} grade [36], which measures the quality of fitting a simulated diffraction pattern to the analyzed experimental diffraction pattern. Only proteins with a grade of "average at this resolution" and better were included; (e) finally, to reduce redundancy, proteins were filtered to maintain up to 70% sequence identity as defined by the RCSB website. Lists of the proteins used in this work are provided in the Additional file 1. Filtering was based on the RSCB search terms and followed by our own Python code: pdb_prep (see below). It should be noted that averaged B-factors, representing the mean square isotropic displacement of each atom [37, 38], were not required as an additional filter since for most of the protein used, the above filters naturally reduced these values to less than or equal to 40 $Å^2$. However, those with higher average B factor showed good R_{free} grades and were therefore included in this study.

The above filters were applied to all symmetric homomers found in the RCSB website with C_4, C_5 and C_6 rotational symmetry resulting with sets of 31, 51 and 16 proteins respectively. For protein homodimers with C_2 symmetry and homotrimers with C_3 symmetry, which are common in the RCSB database, we applied a randomization algorithm (by the Linux "shuf" command written by Paul Eggert) to choose 300 proteins of each type from the website. After applying the above filters these sets were reduced to 194 and 214 proteins respectively.

Preparing proteins for CSM calculations

Prior to analysis, each protein in our sets was cleaned with our python code pdb_prep to delete ligands, solvents, non-coordinates lines (e.g., ANISOU data representing anisotropic temperature factors) from the ATOM section in the PDB file [8, 9], and choose the first location in cases of alternate locations of specific residues. Hydrogen atoms, if existed, were deleted in order to unify the dataset, as most of the PDB files did not include them. In addition the code used reported data on missing residues and atoms (based on remarks 465 and 470 in the PDB file) to insure that all peptides have the same length. If a residue was missing from one or more of the peptides—it was automatically deleted from all other peptides. Similar treatment was given to missing atoms. Finally the files were checked to verify that the length of all peptides is identical.

Results

Symmetry levels of homomers

Table 1 presents descriptive statistics of our set of proteins. It should be noted that while the general scale of the CSM is between 0 and 100, typical CSM values of

Table 1 Descriptive statistics of CSMs for the sets of homomers

Set	CSM	N	Mean	Standard deviation	SE of mean	Minimum	Median	Maximum
Dimers	$S(C_2)$	194	0.1350	0.3132	0.0225	0.0001	0.0471	2.8786
Trimers	$S(C_3)$	214	0.0775	0.1698	0.0116	0.0003	0.0304	1.9519
Tetramers	$S(C_2)$	31	0.0446	0.1444	0.0259	0.0006	0.0130	0.8171
	$S(C_4)$	31	0.1001	0.2490	0.0447	0.0022	0.0311	1.0366
Pentamers	$S(C_5)$	51	0.0974	0.1872	0.0262	0.0028	0.0345	0.9505
Hexamers	$S(C_2)$	16	0.0529	0.0437	0.0109	0.0055	0.0435	0.1565
	$S(C_3)$	16	0.0714	0.0603	0.0151	0.0077	0.0555	0.2326
	S(C6)	16	0.0898	0.0715	0.0179	0.0102	0.0737	0.2699

proteins with approximate symmetry are significantly smaller than 100 [10]. For our sets of proteins the distortion levels varied between 0 and 2. This does not necessarily mean that proteins are more symmetric than small molecules. Rather this results from the definition of the symmetry measure. The denominator, $N(G)$, in Eq. (1) is the sum of distances of each atom from the center of mass of the molecule. While for small molecules this value may be at the order of the sum of deviations of each atom from its expected symmetric position (as appears in the numerator, $M(G)$), for a protein, especially for an elongated structure, the sum of these distances can be much higher than the numerator leading to a small CSM value. Table 1 should thus be used as a reference table to which CSM levels of proteins with approximate symmetry can be compared. Within this range, up to 4 orders of magnitude differences exist between the minimum CSM representing highly symmetric proteins and the maximum CSM representing highly distorted ones. It should be noted that for asymmetric proteins, the CSM can be significantly higher than the values in Table 1, and values in the range 20–30 and even higher are common. On the other hand, there are specific structures that appear to be symmetric by their CSM value, although they are classified as asymmetric in the RCSB website. We comment on this topic in the Additional file 1 and presents statistics for asymmetric homotrimers and homotetramers in Additional file 1: Table S1.

Figure 3 exemplifies the calculation for the crystal structure of the VirB8-like protein, *R. typhi* RvhB8-II homodimer (PDB ID: 4O3 V) [39]. The left structure is the original one, and the right structure is the nearest symmetric structure with $S(C_2) = 1.1261$. While the differences may appear minor with the ribbons view, one should bare in mind that these are more significant at the atoms level (see a ball and sticks view in Additional file 1: Figure S2).

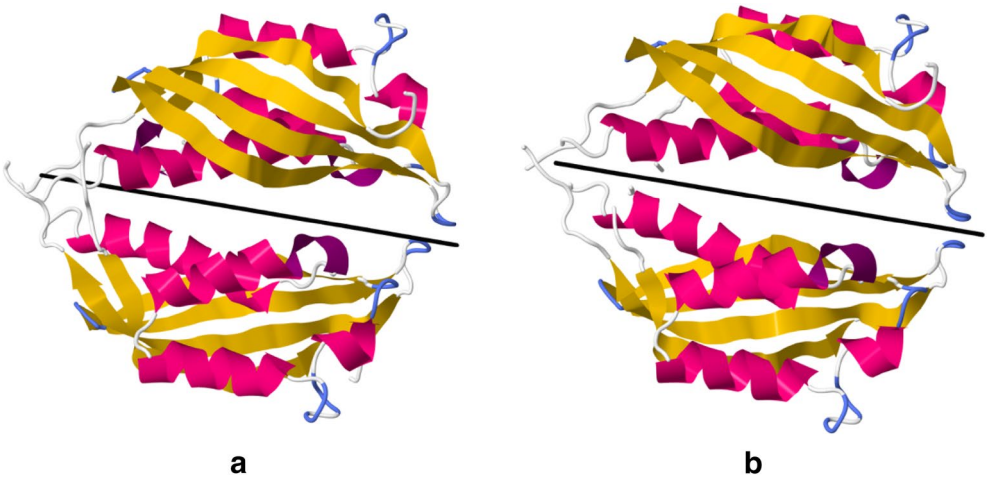

Fig. 3 The homodimer (*R. typhi* RvhB8-II) (PDB-ID: 4O3V) is characterized with $S(C_2) = 1.1261$. **a** Original structure and **b** nearest symmetric structure. The black line represents the direction of the symmetry axis for the nearest symmetric structure. See Additional file 1: Figure S2 for ball and sticks models of the structures

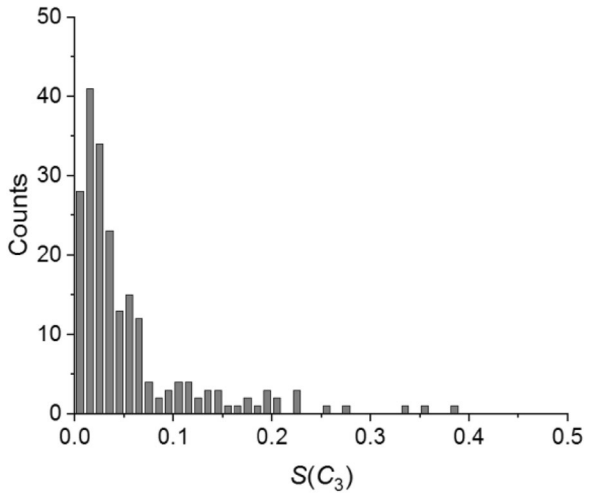

Fig. 4 $S(C_3)$ distribution for the set of 214 protein trimers. Bin size was set to 0.01. The right tail of the distribution is not shown in order to increase the visibility

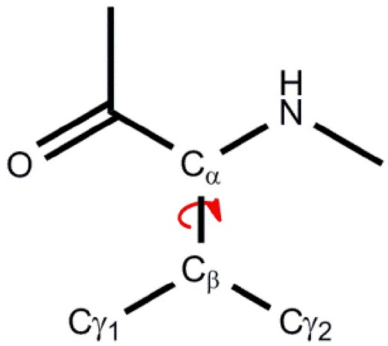

Fig. 5 Val has two C_γ atoms. The free rotation around the C_α–C_β bond, give rise to two possible permutations of these carbon atoms with those of another Val on an equivalent peptide

Returning to Table 1 we note that the standard deviation of each set is much higher than the corresponding mean value. This is resulting from the fact that the CSM is always positive and its distribution has a long tail as exemplified in Fig. 4 for our set of 214 trimers.

Another interesting point apparent from Table 1 is that for even homomers, distortion increases with the order of the rotational symmetry. For tetramers—the mean deviation from C_2 symmetry (0.0442) is smaller than the mean deviation from C_4 symmetry (0.1001). Likewise the mean distortion of the set of hexamers increases in the order: $S(C_2) < S(C_3) < S(C_6)$. The same trend is seen when looking at either the minimum, median or maximum CSM values. This is to be expected—for a hexamer to be perfectly C_6-symmetric, all peptides must be symmetrically aligned around the rotation axis. That is, all peptides must attain a symmetrically equivalent conformation. However, to obtain a C_2-symmetry, it is enough that three of the peptides will be symmetrically equivalent to the other three. That is, the structure has more degrees of freedom because the peptides at each triplet need not be perfectly symmetric with respect to each other.

Finding the correct permutation

As we have described above, our algorithm guarantees that the atom's permutation does not break the peptides: Each peptide is carried in its entirety to another peptide, and therefore we have two levels of permutations: The permutation of equivalence classes of atoms (Fig. 1), and the permutation of the peptides (Fig. 2). Let us look at these permutations more closely. For most of the atoms, their order in the sequence determines their permutation. That is, a C_α of Gly can only be interchanged with a C_α of Gly that has the same sequence number on the equivalent peptide. However, Val for example, has two C_γ atoms (see Fig. 5). These give rise to two possible permutations: either $C_{\gamma1}(A) \rightarrow C_{\gamma1}(B)$ or $C_{\gamma1}(A) \rightarrow C_{\gamma2}$ (B) where A and B are the equivalent peptides. Other possibilities for such permutations are the ring carbons of Phe and Tyr, The C_δ's of Leu, the two nitrogen atoms at the tail of Arg and the two oxygen atoms at the tail of Asp and Glu. If hydrogen atoms exist in the PDB file, larger equivalence groups will result (e.g., for methyl groups at the edge of the side chains). The atoms permutation is kept as a separate list for each combination of two peptides. After the peptides permutation is found, the final total permutation is constructed by linking the peptides permutation with the relevant atoms permutations. In what follows we describe the possibilities for peptide permutations, and the differences between them.

Finding the permutation of the peptides follows Eq. (6) described above. For dimers and trimers, finding the permutation of the peptides is straightforward: $A \rightarrow B$ for dimers and $A \rightarrow B \rightarrow C$ for trimers (note that $A \rightarrow C \rightarrow B$ is an equivalent permutation). However as the number of peptides in a protein increases, it is not clear a priori which permutation will lead to a smaller CSM. A peptide permutation that follows the order of the peptides in the PDB file was found for 45% of the tetramers, 69% of the pentamers and 81% of the hexamers. In other words, relying on the PDB file order of peptides can lead to an error of up to 55% of the tetramers, 31% of the pentamers and 19% of the hexamers. Nevertheless, larger data sets may alter these numbers. Additional file 1: Tables S2–S4 present the specific permutations and their frequencies for our sets of tetramers, pentamers and hexamers

We continue by testing the differences between the best permutation of the peptides and atoms, to the sequence-ordered permutation of the atoms, in which

the atoms are interchanged according to their serial numbers in the PDB file, and to the permutation found by the greedy algorithm. In both cases all possible permutations of the peptides were taken into account. These differences, although important, do not affect the permutation of the peptides. Starting with the sequence-ordered permutation of the atoms, we found that it generally leads to higher CSM values as compared with the ones found by the Hungarian algorithm. That is, it finds the protein to be less symmetric than it really is. Table 2 presents the results of these comparisons in terms of the relative error defined by:

$$\text{Relative Error} = 100 \cdot \frac{\left|CSM_{full} - CSM_{sequence-ordered}\right|}{CSM_{full}} \tag{7}$$

As is evident, using the sequence-ordered permutation adds a median relative error of 4–10%, but the maximum error is much higher and can be as high as 49%. Two comments are in place here. First, the higher errors are more abundant when the measure itself is low, that is, the protein is highly symmetric. This is resulting from the definition of the relative error in Eq. (7). As an example, Fig. 6 presents the relative error as a function of $S(C_3)$ for the set of trimers. Second, in few cases the sequence-ordered permutation does provide the same CSM value as with the Hungarian algorithm. This was obtained for 11 out of 214 trimers (5%) and 1 out of 31 tetramers (3%) in the calculation of $S(C_4)$. Note that the minimum relative error of $S(C_2)$ for the same set of tetramers is not zero.

Similarly to the analysis presented above, a comparison with the greedy algorithm was conducted. In all cases but one, the greedy algorithm found a higher CSM value as compared with the Hungarian algorithm, that is a permutation that leads to a less symmetric structure. Table 3 presents the comparison of the Hungarian algorithm and the greedy algorithm. Here the relative errors are lower, with a median error of 1% and up to 3% and a maximum error of 3% and up to 10%. This makes sense as the greedy algorithm does attempt to find a better permutation than the sequence-ordered permutation, although it does not succeed in all of the cases. A zero relative error has been obtained in higher percentages as compared with the sequence-ordered permutation: 9% of the dimers, 12% of the trimers, 26% for $S(C_2)$ of the tetramers and 19% for $S(C_4)$, 6% of the pentamers, none for $S(C_2)$ and $S(C_3)$ of the hexamers and 13% for $S(C_6)$ of the hexamers. Altogether, we can estimate that the greedy algorithm for finding the permutation of the atoms is equivalent to the Hungarian algorithm in up to 26% of the calculations.

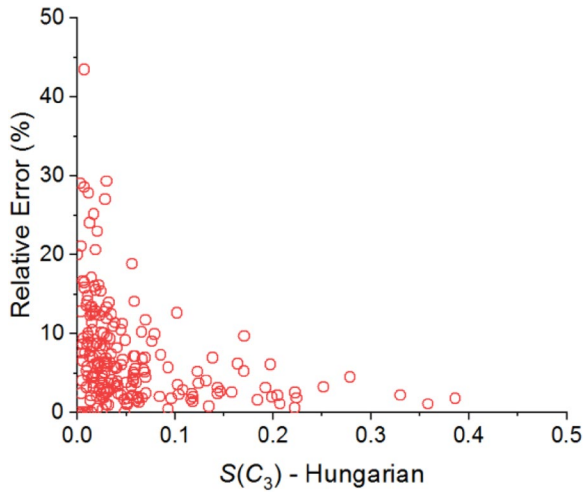

Fig. 6 Relative error of $S(C_3)$ calculated for the dataset of 214 protein trimers resulting from using the sequence-ordered permutation as opposed to the permutation found by the Hungrian algorithm

Table 2 Descriptive statistics of the relative deviation of the CSM resulting from the sequence-ordered permutation of the atoms as compared with the Hungarian algorithm

Set	CSM	N	Mean (%)	Standard deviation (%)	SE of mean (%)	Minimum (%)	Median (%)	Maximum (%)
Dimers[a]	$S(C_2)$	193	6.7	6.7	0.5	0.1	4.5	37.3
Trimers	$S(C_3)$	214	6.9	6.6	0.5	0.0	5.1	43.5
Tetramers	$S(C_2)$	31	12.8	10.6	1.9	0.4	10.0	49.3
	$S(C_4)$	31	9.9	10.1	1.8	0.0	6.6	42.4
Pentamers	$S(C_5)$	51	5.7	5.1	0.7	0.1	4.6	21.5
Hexamers	$S(C_2)$	16	9.0	6.9	1.7	2.4	6.6	25.0
	$S(C_3)$	16	7.7	5.2	1.3	2.2	6.4	20.8
	$S(C_6)$	16	7.1	4.9	1.2	1.9	6.1	21.0

[a] Statistical analysis was done on 193 out of 194 dimers. One dimer, with PDB-ID 2AJQ, is highly symmetric with $S(C_2) = 0.0001$ for the Hungarian algorithm and 0.0004 for the sequence-ordered permutation, led to an error of 300%. It was therefore considered as an outlier and excluded from this calculation

Table 3 Descriptive statistics of the relative deviation of the CSM resulting from a greedy algorithm of the atoms as compared with the Hungarian algorithm

Set	CSM	N	Mean (%)	Standard deviation (%)	SE of mean (%)	Minimum (%)	Median (%)	Maximum (%)
Dimers	$S(C_2)$	194	1.5	1.3	0.1	0.0	1.2	7.1
Trimers	$S(C_3)$	214	1.5	1.1	0.1	0.0	1.3	5.6
Tetramers	$S(C_2)$	31	2.0	1.9	0.3	0.0	1.7	6.8
	$S(C_4)$	31	1.1	1.0	0.2	0.0	1.0	3.3
Pentamers	$S(C_5)$	51	2.1	1.9	0.3	0.0	1.6	9.8
Hexamers	$S(C_2)$	16	2.9	2.1	0.5	0.5	2.2	7.9
	$S(C_3)$	16	1.6	1.2	0.3	0.3	1.3	3.9
	$S(C_6)$	16	2.5	1.6	0.4	0.0	2.6	5.1

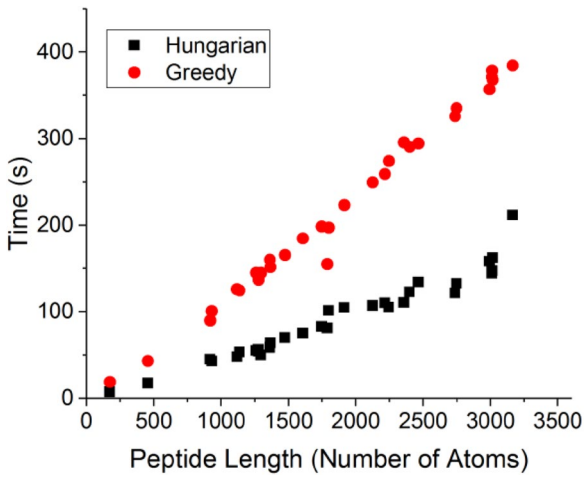

Fig. 7 Calculation time of $S(C_4)$ for the tetramers dataset using the many-chains algorithm. Red circles: greedy algorithm. Black square: Hungarian algorithm

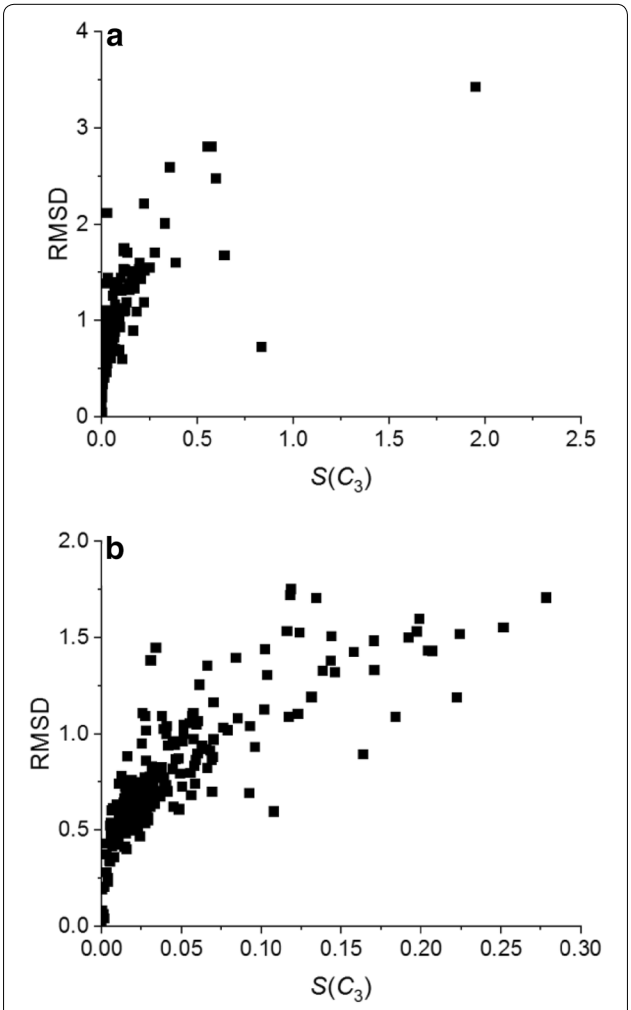

Fig. 8 RMSD versus $S(C_3)$ for the set of trimers. **a** Full range and **b** inset

Calculation time

As an estimation for the speed of the calculation we present in Fig. 7 the real time for calculating $S(C_4)$ for our set of tetramers as a function of the number of atoms in each peptide. Generally, for short proteins, time increases linearly with size. As the number of atoms increases, the time dependency deviates from linearity. The time range was 9 s for the shortest protein with 174 atoms in each peptide, and up to ca. 5 min for a protein with 3166 atoms in each peptide. We note that the number of iterations the code performs in order to find the best permutation is typically small, between 2 and 4. All calculations were performed using one core of an 8-cores Linux machine with 64 GB RAM memory.

CSM and RMSD

Symmetry breaking is commonly analyzed in the literature by superposing the protein subunits and assessing the root mean square deviation (RMSD) between them [26, 27]. The RMSD, like the CSM is zero for perfect symmetry and increases as the distortion increases. As

seen in Fig. 8a for our set of trimers, both the RMSD (calculated with MOE [40] for all atoms) and the CSM increases with the deviation from C_3 symmetry, but are not quantitatively correlated. For highly symmetric structures the correlation improves though it remains qualitative (Fig. 8b). Similar results were obtained for the other sets of homomers.

Conclusions

Continuous symmetry and chirality measures determine the distortion level of a structure by searching for the nearest symmetric (or achiral) structure and calculating the distance between the two structures. The approximate algorithm presented here provides significant improvements over pervious codes in terms of accuracy, speed of the calculation, and the scope of molecular structure complexity it can handle. As has been shown here it can be used as a robust and versatile molecular descriptor of protein structure. Symmetry is an important advantageous for protein structure, yet not trivial to achieve. With an accurate and efficient tool to estimate this symmetry one opens the door to understand where and why nature fails to achieve perfect symmetry and what functions do imperfection serve. Applications of the methods include characterization of the three-dimensional structure of proteins in the solid state or in solution, analysis of conformational changes during dynamical processes and exploration of quantitative structure–activity relationships. Modification of the methods for a robust analysis of non-biological large molecules as well as nanomaterials and molecular clusters is currently in progress.

Additional file

Additional file 1. Supplementary material.

Abbreviations

CSM: continuous symmetry measure; RMSD: root mean square deviation.

Acknowledgements

We are extremely grateful to Itay Zandbank and Devora Witty (The scientific software company, Israel) for their help in programming the new CSM code, to Sagiv Barhoom (The Open University) for his help in programming and developing the pdb_prep code, and to Yaffa Shalit (The Open University) for her help in testing the codes. Special thanks are reserved to Prof. David Avnir (The Hebrew University of Jerusalem) for numerous fruitful discussions.

Authors' contributions

The research was conducted by mutual contributions of both authors. Both authors read and approved the final manuscript.

Funding

Supported by the Israel Science Foundation (Grant 411/15).

Availability of data and materials

A list of PDB-IDs used in this study is given in the Additional file 1. An online CSM calculator is available at: http://csm.ouproj.org.il. Project name: proteincsm. Project home page: https://github.com/continuous-symmetry/proteincsm. Archived version: 1.0.1. Operating system(s): Linux, Windows. Programming language: Python, c++. Other requirements: unzipping package like bzip2, OpenBabel (requires X11 libraries), c++ compiler, conda, numpy. License: GNU-GPL version 2. Any restrictions to use by non-academics: Not applicable. Project name: pdb_prep. Project home page: https://sagivba.github.io/pdb_prep/. Archived version: 0.0.8.4. Operating system(s): Linux. Programming language: Python. Other requirements: click. License: BSD 2-Clause. Any restrictions to use by non-academics: Not applicable.

Competing interests

The authors declare no competing financial interest.

Received: 1 October 2018 Accepted: 28 May 2019
Published online: 06 June 2019

References

1. Goodsell DS, Olson AJ (2000) Structural symmetry and protein function. Annu Rev Biophys Biomol Struct 29:105–153
2. Kojic-Prodic B, Stefanic Z (2010) Symmetry versus asymmetry in the molecules of life: homomeric protein assemblies. Symmetry-Basel 2(2):884–906
3. Blundell TL, Srinivasan N (1996) Symmetry, stability, and dynamics of multidomain and multicomponent protein systems. Proc Natl Acad Sci USA 93(25):14243–14248
4. Andre I, Strauss CEM, Kaplan DB, Bradley P, Baker D (2008) Emergence of symmetry in homooligomeric biological assemblies. Proc Natl Acad Sci USA 105(42):16148–16152
5. Changeux JP (2012) Allostery and the Monod–Wyman–Changeux model after 50 years. Annu Rev Biophys 41:103–133
6. Schulze B, Sljoka A (2008) Whiteley W (2014) How does symmetry impact the flexibility of proteins? Philos Trans R Soc A Math Phys Eng Sci 372:20120041
7. Levy ED, Pereira-Leal JB, Chothia C, Teichmann SA (2006) 3D complex: a structural classification of protein complexes. PLoS Comput Biol 2(11):1395–1406
8. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN et al (2000) The protein data bank. Nucleic Acids Res 28(1):235–242
9. Berman H, Henrick K, Nakamura H (2003) Announcing the worldwide Protein Data Bank. Nat Struct Biol 10(12):980
10. Bonjack-Shterengartz M, Avnir D (2015) The near-symmetry of proteins. Proteins Struct Funct Bioinform 83(4):722–734
11. Bonjack-Shterengartz M, Avnir D (2017) The enigma of the near-symmetry of proteins: domain swapping. PLoS ONE 12(7):e0180030
12. Levy Y, Cho SS, Shen T, Onuchic JN, Wolynes PG (2005) Symmetry and frustration in protein energy landscapes: a near degeneracy resolves the Rop dimer-folding mystery. Proc Natl Acad Sci USA 102(7):2373–2378
13. Zabrodsky H, Peleg S, Avnir D (1992) Continuous symmetry measures. J Am Chem Soc 114(20):7843–7851
14. Zabrodsky H, Avnir D (1995) Continuous symmetry measures. 4. Chirality. J Am Chem Soc 117(1):462–473
15. Pinsky M, Dryzun C, Casanova D, Alemany P, Avnir D (2008) Analytical methods for calculating continuous symmetry measures and the chirality measure. J Comput Chem 29(16):2712–2721
16. Alon G, Tuvi-Arad I (2018) Improved algorithms for symmetry analysis: structure preserving permutations. J Math Chem 56(1):193–212
17. Dryzun C, Zait A, Avnir D (2011) Quantitative symmetry and chirality—a fast computational algorithm for large structures: proteins, macromolecules, nanotubes, and unit cells. J Comput Chem 32(12):2526–2538
18. Zhang Y, Skolnick J (2005) TM-align: a protein structure alignment algorithm based on the TM-score. Nucleic Acids Res 33(7):2302–2309
19. Tai CH, Paul R, Dukka KC, Shilling JD, Lee B (2014) SymD webserver: a platform for detecting internally symmetric protein structures. Nucleic Acids Res 42(W1):W296–W300

20. Myers-Turnbull D, Bliven SE, Rose PW, Aziz ZK, Youkharibache P et al (2014) Systematic detection of internal symmetry in proteins using CE-Symm. J Mol Biol 426(11):2255–2268
21. Mukherjee S, Zhang Y (2009) MM-align: a quick algorithm for aligning multiple-chain protein complex structures using iterative dynamic programming. Nucleic Acids Res 37(11):e83
22. Madej T, Lanczycki CJ, Zhang DC, Thiessen PA, Geer RC et al (2014) MMDB and VAST+: tracking structural similarities between macromolecular complexes. Nucleic Acids Res 42(D1):D297–D303
23. Lafita A, Bliven S, Kryshtafovych A, Bertoni M, Monastyrskyy B et al (2018) Assessment of protein assembly prediction in CASP12. Proteins Struct Funct Bioinform 86:247–256
24. Kim C, Basner J, Lee B (2010) Detecting internally symmetric protein structures. BMC Bioinform 11:303
25. Do Viet P, Roche DB, Kajava AV (2015) TAPO: a combined method for the identification of tandem repeats in protein structures. FEBS Lett 589(19):2611–2619
26. Pagès G, Kinzina E, Grudinin S (2018) Analytical symmetry detection in protein assemblies. I. Cyclic symmetries. J Struct Biol 203(2):142–148
27. Pagès G, Grudinin S (2018) Analytical symmetry detection in protein assemblies. II. Dihedral and cubic symmetries. J Struct Biol 203(3):185–194
28. Shindyalov IN, Bourne PE (1998) Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Eng 11(9):739–747
29. Zhang Y, Skolnick J (2004) Scoring function for automated assessment of protein structure template quality. Proteins Struct Funct Bioinform 57(4):702–710
30. Munkres J (1957) Algorithms for the Assignment and Transportation Problems. J Soc Ind Appl Math 5(1):32–38
31. Allen WJ, Rizzo RC (2014) Implementation of the Hungarian algorithm to account for ligand symmetry and similarity in structure-based design. J Chem Inf Model 54(2):518–529
32. Pinsky M, Casanova D, Alemany P, Alvarez S, Avnir D et al (2008) Symmetry operation measures. J Comput Chem 29(2):190–197
33. Carugo O, Eisenhaber F (eds) (2016) Data mining techniques for the life sciences, 2nd edn. Springer, New York
34. Lamb AL, Kappock TJ, Silvaggi NR (2015) You are lost without a map: Navigating the sea of protein structures. Biochim Biophys Acta Proteins Proteom 1854(4):258–268
35. Martz E. FirstGlance in Jmol. Version 2.51. https://bioinformatics.org/firstglance/fgij
36. Kleywegt GJ, Jones TA (1997) Model building and refinement practice. Macromol Crystallogr B 277:208–230
37. Trueblood KN, Burgi HB, Burzlaff H, Dunitz JD, Gramaccioli CM et al (1996) Atomic displacement parameter nomenclature—report of a subcommittee on atomic displacement parameter nomenclature. Acta Crystallogr Sect A 52:770–781
38. Read RJ, Adams PD, Arendall WB, Brunger AT, Emsley P et al (2011) A new generation of crystallographic validation tools for the protein data bank. Structure 19(10):1395–1412
39. Gillespie JJ, Phan IQH, Scheib H, Subramanian S, Edwards TE et al (2015) Structural insight into how bacteria prevent interference between multiple divergent type IV secretion systems. mBio 6(6):e01867-15
40. Molecular Operating Environment (MOE) (2013) Chemical Computing Group Inc. https://www.chemcomp.com

Publisher's Note