Analytical symmetry detection in protein assemblies. II. Dihedral and Cubic symmetries

Guillaume Pagès, Sergei Grudinin

Analytical symmetry detection in protein assemblies. II. Dihedral and Cubic symmetries

Guillaume Pagès[a], Sergei Grudinin[a]

[a]*Inria, Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, Grenoble, 38000, France*

Abstract

Protein assemblies are often symmetric, as this organization has many advantages compared to individual proteins. Complex protein structures thus very often possess high-order symmetries. Detection and analysis of these symmetries has been a challenging problem and no efficient algorithms have been developed so far. This paper presents the extension of our cyclic symmetry detection method for higher-order symmetries with multiple symmetry axes. These include dihedral and cubic, i.e., tetrahedral, octahedral, and icosahedral, groups. Our method assesses the quality of a particular symmetry group and also determines all of its symmetry axes with a machine precision. The method comprises discrete and continuous optimization steps and is applicable to assemblies with multiple chains in the asymmetric subunits or to those with pseudo-symmetry.

We implemented the method in C++ and exhaustively tested it on all 51,358 symmetrical assemblies from the Protein Data Bank (PDB). It allowed us to study structural organization of symmetrical assemblies solved by X-ray crystallography, and also to assess the symmetry annotation in the PDB. For example, in 1.6% of the cases we detected a higher symmetry group compared to the PDB annotation, and we also detected several cases with incorrect annotation. The method is available at http://team.inria.fr/nano-d/software/ananas. The graphical user interface of the method built for the SAMSON platform is available at http://samson-connect.net.

Keywords: Point-Group Symmetry, Cubic Groups, Group Theory, Discrete Optimization, Protein Structure, Continuous Optimization

Email address: sergei.grudinin@inria.fr (Sergei Grudinin)

1. Introduction

Symmetrical protein complexes are very common in nature, as has been highlighted in our previous work on symmetry detection in cyclic protein assemblies [1], and many of these are deposited to the Protein Data Bank (PDB) [2]. As function of proteins is very often determined by their structure, it appears that complex function requires complex structures [3, 4]. High-order symmetries are thus essential to build large and complex protein assemblies. Dihedral and cubic groups are overrepresented among large protein assemblies with some specific structural functions, for example those of viral capsids. Also, high-order symmetry drastically reduces the complexity of *de novo* design of self-assembling nanomaterials [5, 6, 7, 8].

To assess the quality of symmetry for such assemblies, a cyclic symmetry measure is necessary, as the cyclic axes constitute the basic bricks from which one can reconstruct high-order symmetry groups. However, considering each symmetry axis separately would result in a globally incorrect assessment, as there are strict geometrical constraints between different axes of symmetry in high-order symmetry groups. This motivated us to develop a symmetry detection method specifically suited for dihedral and cubic groups. Indeed, the need for this symmetry detection method exists, as some approximate methods, i.e. those from BioJava [9], are massively used to display the symmetry axes on the PDB website [2].

Inspired by the quaternion arithmetic applied to the best superposition of a set of points [10, 11, 12] together with our recent developments [13, 14, 1], here we propose a new symmetry measure and an analytical method to find the best symmetry axes of a symmetrical assembly possessing multiple symmetry axes. The method guaranties that the detected axes are consistent with the symmetry constraints. Similar to the case of cyclic symmetry detection [1], our method produces results with a machine precision, its cost function is solely based on 3D Euclidean geometry, and most of the operations are performed analytically. This makes it extremely fast and particularly suitable for exhaustive analysis of PDB data. Below we provide details about the high-order symmetry measure and the computation of the symmetry axes for an assembly possessing any point symmetry group. The method first perceives the topology between different chains, and is able to deal with complex subunits that are composed of multiple chains. Then it iteratively

solves a constrained quadratic optimization problem using a set of analytical solutions.

2. Methods

2.1. Notations

Similarly to the analysis of the cyclic groups [1], in this paper we will be mainly dealing with 3×3 matrices and 3-vectors. Therefore, bold upper case letters (i.e. $\mathbf{A}$) will denote matrices, bold lower case letters (i.e. $\mathbf{b}$) will denote vectors, and normal weight lower case letters (i.e. c) will denote scalars. For trigonometric operations and illustrations we will also use an arrow notation for 3-vectors, such as $\vec{v}$.

All amino acids, except glycine, are chiral. Hence, symmetry groups that can be present in protein assemblies cannot contain any reflection, inversion, or improper rotation. The only remaining finite point groups are the cyclic (C_n for the cyclic group of order n), dihedral (D_n for the dihedral group of order n), tetrahedral, octahedral and icosahedral (respectively T, O and I), the three cubic groups. Symbol $\Gamma \in \{C_n\}_{n>1} \bigcup \{D_n\}_{n>1} \bigcup \{T, O, I\}$ will denote one of these point groups. Its cardinality, i.e. the number of its elements, will be denoted as $|\Gamma|$.

2.2. Root mean square deviation

As in our previous work on cyclic groups [1], we will express the *symmetry measure* of a molecular assembly using the root mean square deviation (RMSD). Given two sets of N points each, $A = \{\mathbf{a}_i\}_N$ and $B = \{\mathbf{b}_i\}_N$, the RMSD between them is given as

$$\mathrm{RMSD}(A, B)^2 = \frac{1}{N} \sum_{1 \le i \le N} |\mathbf{a}_i - \mathbf{b}_i|^2 . \tag{1}$$

2.3. Group Theory

Firstly, we will give a brief introduction to the group theory used in the paper. A group is a set of elements equipped with an operation that combines any two elements to form a third element. Formally, it can be written in the form of $(\Gamma, \star)$, where Γ is a set of elements supplied with a group operation $\star$. A *homomorphism* is a function that takes a group element as input and returns an element of another group as output, preserving the

group structure. Given two groups $(\Gamma_1, \star)$ and $(\Gamma_2, *)$, a homomorphism f from Γ_1 to Γ_2 satisfies the following property,

$$\forall g, g' \in \Gamma_1 : f(g \star g') = f(g) * f(g'). \tag{2}$$

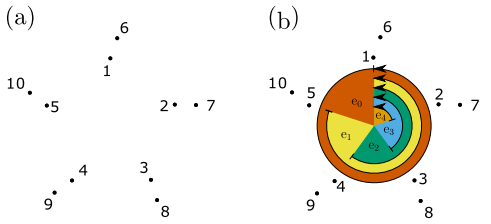
A homomorphism is called *bijective* if

$$\forall g_2 \in \Gamma_2 \exists! g_1 \in \Gamma_1 \text{ such that } f(g_1) = g_2, \tag{3}$$

meaning that it provides a one-to-one association between the elements of the two groups. In this paper, we will consider three different types of groups :

- *Rotation groups*, where the elements are rotations and the group operation is the composition of rotations. The group of all rotations in 3 dimensions will be noted $SE(3)$, and rotations will be noted r.
- *Permutation groups*, where the elements are permutations and the group operation is the composition of permutations. A permutation σ of the set $\{1, .., n\}$ will be noted $(\sigma(1), ..., \sigma(n))$. The group of all permutations of n elements will be noted $\mathfrak{S}_n$.
- *Point groups* Γ that will be composed of abstract elements. We know how these elements are combined (meaning that we know the results of composition of any two elements), and we will take the freedom to use the elements either as permutations or as rotations, thanks to the bijective homomorphisms that will be computed.

The considered point groups are only those that can be described with rotation operators (no reflections or inversions). We say that a set of points (a protein assembly will be represented with a set of points) has a Γ symmetry if the rotations in Γ keep this set globally invariant. More precisely, any rotation operator in Γ will displace each point that is located outside of the rotation axis, but another point will take its place. It is natural to see Γ as a rotation group, but also as a permutation group, since a rotation applied to a set of points will permute them. For example, Figure 1 illustrates a C_5 point group, which has a bijective homomorphism with a rotation group and a bijective homomorphism with a permutation group.



Element	Permutation	Angle
e_0	(1,2,3,4,5,6,7,8,9,10)	2π
e_1	(2,3,4,5,1,7,8,9,10,6)	$8\pi/5$
e_2	(3,4,5,1,2,8,9,10,6,7)	$6\pi/5$
e_3	(4,5,1,2,3,9,10,6,7,8)	$4\pi/5$
e_4	(5,1,2,3,4,10,6,7,8,9)	$2\pi/5$

Figure 1: Point group C_5 is illustrated. (a) Set of points with a cyclic symmetry of order 5; (b) All the rotations that keep this set of points fixed; (c) Associations between the elements of C_5, permutations of the points and rotations.

2.4. Problem definition

For a molecular point group $\Gamma \in \{C_n\}_{n>1} \bigcup \{D_n\}_{n>1} \bigcup \{T, O, I\}$, and an assembly $A = \{\mathbf{a}_{i,j}\}_{N_s,N_a}$ consisting of N_s subunits, each composed of N_a atoms, we want to minimize the following loss function,

$$\text{Loss}^2 = \frac{1}{|\Gamma| N_s N_a} \min_{\sigma_g, r_g} \sum_{g \in \Gamma} \sum_{i=1}^{N_s} \sum_{j=1}^{N_a} \left(\mathbf{a}_{\sigma_g(i),j} - r_g \left(\mathbf{a}_{i,j} \right) \right)^2, \tag{4}$$

such that $g \mapsto \sigma_g$ and $g \mapsto r_g$ are the bijective homomorphisms from Γ to subsets of $\mathfrak{S}_n$ and $SE(3)$. The loss function is the sum of RMSDs between the original assembly and the rotated assemblies for every rotation in the group Γ. We should mention that this loss function is very natural, since it is only based on Euclidean 3D distances, no adjustable parameters are required and all the rotations r_g have equal importance.

2.5. Workflow

Minimization of the loss function 4 requires optimization over the group of rotations, which is a *continuous optimization*, and over the group of permutations, which is a *discrete optimization*. In practice, we do not know how to do both simultaneously, so we first apply a heuristic approach to determine the correspondences σ_g between the subunits, and then we optimize the rotations. Overall, we solve the optimization problem in three steps,

1. Subunit definition
2. Estimation of the permutations (5)
3. Optimization of the rotations

The third step is an analytical continuous minimization. It gives the expected result with a machine precision, as we have already demonstrated for cyclic symmetries [1]. The first two steps are heuristics that assume the assembly to be symmetric enough, which allows to estimate the best correspondences between the subunits. This problem of estimating the best correspondence is discrete. Any error during the estimation of the correspondences typically leads to the solution of optimization problem 4 with the Loss comparable to the distance between the center of masses (COMs) of the subunits. It is therefore straightforward to verify whether the result of our discrete optimization is correct. Without loss of generality, we assume that the molecular point group called Γ is given. If it is not the case, we can exhaustively search over three cubic groups and also try all the dihedral groups below a certain maximum order. Below we discuss the individual steps of our optimization algorithms in more detail.

2.6. Finding the subunits

We call subunit a minimum part of the symmetrical assembly, from which the entire assembly can be reconstructed by replicating the subunits according to the symmetry operator. Note that the total number of subunits in a complete symmetrical assembly has to be equal to $|\Gamma|$. In most of the practical cases, subunits are actually the individual chains of the molecular assembly. In this case, it is straightforward to define them. In some cases, however, subunits can be composed of several chains. To find these multi-chain subunits, we create several sets of chains, each forming a Γ symmetrical assembly. Then, the subunits are defined by appropriately choosing one chain from from each of these sets as shown in Fig. 2. All the sets are Γ-symmetric assemblies, and thus contain precisely $|\Gamma|$ chains. Since we know the initial number of chains in the assembly, we also know that the number of sets has to be equal to the number of chains divided by $|\Gamma|$. These sets are constructed using a penalty function defined for a pair of chains with four contributions. These are obtained by computing a pairwise sequence alignment followed by a structural superposition for all the pairs of chains. For two chains i and j in an assembly composed of n chains, we define their penalty function as

$$d(i,j) = \frac{d_{seq}(i,j)}{d_{seq}^{max}+40} + \frac{d_{struct}(i,j)}{d_{struct}^{max}+3\text{Å}} + \frac{d_{angle}(i,j)}{d_{angle}^{max}+0.05rad} + \frac{d_{center}(i,j)}{d_{center}^{max}+3\text{Å}}, \quad (6)$$

where

- $d_{seq}(i,j)$ is the minus BLOSUM62 score [15] of the sequence alignment between chains i and j, computed using the MUSCLE package [16].

- $d_{struct}(i,j)$ is the RMSD between two chains i and j, after the best superposition of the corresponding alpha-carbons.

- Let us define an angle $\alpha(i,j)$ between two chains i and j as the angle returned by the superposition procedure, i.e. the rotation angle between them after the COMs were superposed. In a perfectly symmetrical assembly, only a few values of these angles are possible. More precisely, if a group Γ contains multiple symmetry axes $n_j...n_k$ of orders $j...k$, correspondingly, then the pairs of chains will be mutually rotated by angles $2m\pi/l$, where $1 \leq m < l$ with $l = j...k$. We define $d_{angle}(i,j)$ as the absolute value of the difference between $\alpha(i,j)$ and the closest listed angle.

- $d_{center}(i,j)$ is the distance between the initial COM of the whole assembly and its position after applying the rigid-body transformation that superposes chain i with chain j.

In all the contributions in the equation above, d^{max} stands for the largest value of the corresponding contribution. This way, we ensure that all the terms in the above equation have weights of approximately the same magnitude, and none of these terms are bigger than 1. To improve the discrimination of the penalty function for the assemblies with nearly perfect symmetry, we also add constants to d^{max} to make the values of denominators sufficiently large.

After having computed all the pairwise penalties $d(i,j)$, we apply an algorithm to cluster the chains into several sets, as it is listed in Alg. 1. To merge two sets, we define a new set containing all the chains from the two other sets. The clustering algorithm is based on the computation of pairwise distances between the sets, where each distance is defined as an average of the penalties between all pairs of chains, such that one chain belongs to the first set and another chain belongs to the second set.

Once the sets are computed, we proceed with the construction of the subunits. To do so, we first compute for each of the sets the symmetry axes corresponding to the group Γ. The detailed axes computation procedure is explained below. Then, we group chains from different sets to construct the subunits. We first match axes computed for all the sets, then we choose

```
foreach chain do
    Create a new set;
    Add this chain to the set;
end
while Number of sets is bigger than expected do
    Find the two closest sets whose sum of sizes is smaller or equal to
    the expected set size;
    if Such a pair of sets is found then
        Merge the two sets;
    else
        Fail;
    end
end
```

Algorithm 1: Clustering algorithm.

one arbitrary reference chain from each set and assemble them to create the first subunit. For all the other subunits, we choose a group operator and assemble together all the chains from the different sets that correspond to the reference chain to which this operator is applied. The correspondence estimation method is detailed below.

2.7. Correspondences between subunits after a rotation

The number of bijective homomorphisms $g \mapsto \sigma_g$ from Γ to subsets of $\mathfrak{S}_n$ grows exponentially with the size of Γ, it is thus not feasible to do an exhaustive search for $|\Gamma|$ bigger than 10. Therefore, to estimate the correspondence between the subunits we set up heuristics.

Let us start by imagining a perfectly symmetrical system consisting of n subunits, with the center of the symmetry located at the origin. Let $\{c_i\}_{\{1\leq i\leq n\}}$ be the COMs of the subunits. Remark that c_i are located on a sphere. The convex hull of $\{c_i\}_{\{1\leq i\leq n\}}$, which is a polyhedron with n vertices, possesses the following properties: each axis of symmetry of order $s > 2$ crosses two faces of the convex hull, which are the regular polygons with s vertices, and each axis of symmetry of order 2 crosses an edge of the convex hull. We first create a *reference graph* from the edges of the convex hull that belong to the regular polygons with one of symmetry axes of order $s > 2$ passing through them. The topology of this graph does not depend on the choice of the selected perfectly symmetrical system (see Figure 3). For

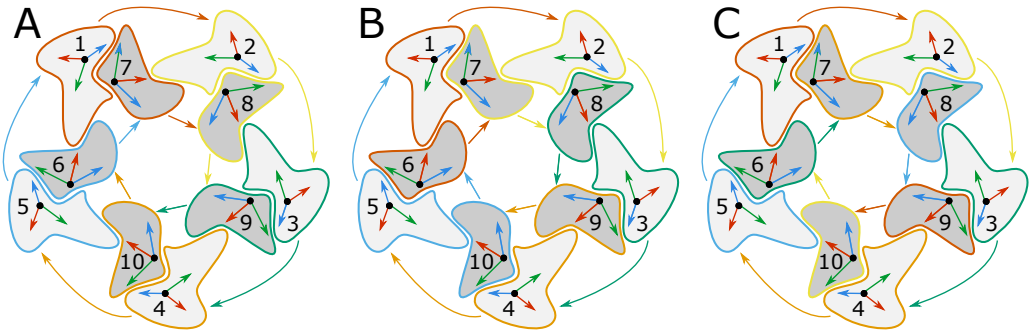


Figure 2: Examples of finding subunits in a C_5 complex containing 10 chains, which leads to 2 sets and 5 subunits. Each chain is shown as a shape, with a local coordinate frame computed with a structural superposition. A correct clustering in this example should determine a set with the chains 1,2,3,4,5 (light grey) and another one with the chains 6,7,8,9,10 (dark grey). The correspondence between the subunits after a rotation is represented by the colored arrows. Chains belonging to the same subunits are drown with the same outline colors. After we define the two sets, the first subunit is created by taking arbitrary chains from each cluster (1 and 7 in (A), 1 and 6 in (B), 1 and 9 in (C)). The other subunits are created using the correspondences between the chains in the sets. Note that the assembling of subunits is not unique. In (A) and (B) the subunits seem to be assembled more naturally, as they contain chains that are spatially proximate to each other. However (C) is a perfectly valid assembly too. All the three subunit assemblies give exactly the same result in terms of symmetry axes and the RMSD loss function.

the dihedral assemblies we also include into the graph the edges crossing the 2-fold axes. The main idea behind the correspondence estimation method is to fit our non-perfect system to a perfect canonical example, and then to use the known correspondence from this canonical example to deduce the correspondence of our system. To create such a fitting, the COMs of the subunits are projected on a sphere centered in the COM of the assembly. Then the *polyhedral graph* P [17] of the convex hull is computed, and we seek for subgraphs of P that are isomorphic to the *reference graph*. This problem is generally known as the subgraph isomorphism problem, which has been well studied in literature [18]. Since our example is not perfect, it happens that we obtain either zero or several matches. If we get zero matches, we connect the two most spatially proximate yet unconnected vertices in our graph with an edge and restart the subgraph isomorphism procedure until we obtain some result. If we get several results, we use two geometric criteria to select the best one. More precisely, the first criterion is the variance of the lengths of the graph edges that belong to the same regular polyhedra. The second criterion is the difference between the reference subunits' angles $\alpha(i, j)$ and the observed angles.

Once the *polyhedral graph* computed from the input structure has been mapped to the *reference graph*, we use the precomputed correspondences of the reference graph to map them on our graph and obtain the correspondences between subunits for each element of Γ. This way, we ensure that the function $g \mapsto \sigma_g$ is, by construction, a bijective homomorphism.

2.8. Graph representation of the group generation

A dual representation of a symmetry group given as a set of permutations, will be a set of rotations. All of these can be obtained as a combination of two *generator* rotations, one r_3 being a rotation about a 3-fold axis $\vec{v}_3$ by an angle $\frac{2\pi}{3}$ for cubic groups, (respectively r_n being a rotation about a n-fold axis $\vec{v}_n$ by an angle $\frac{2\pi}{n}$ for dihedral groups) and the other r_2 being a rotation about a 2-fold axis $\vec{v}_2$ by an angle π. We should emphasize that these two rotations are present in the 3 possible cubic groups. Then, we may represent all the elements of a point group symmetry as vertices in a *Cailey graph* [19], whose edges correspond to the two types of the *generator* rotations. A group element here can be seen as a certain rotation of the symmetrical assembly. For example, a tetrahedral symmetry would have 12 elements (or rotations), an octahedral symmetry would have 24 elements, and an icosahedral symmetry would have 60 elements. The same representation

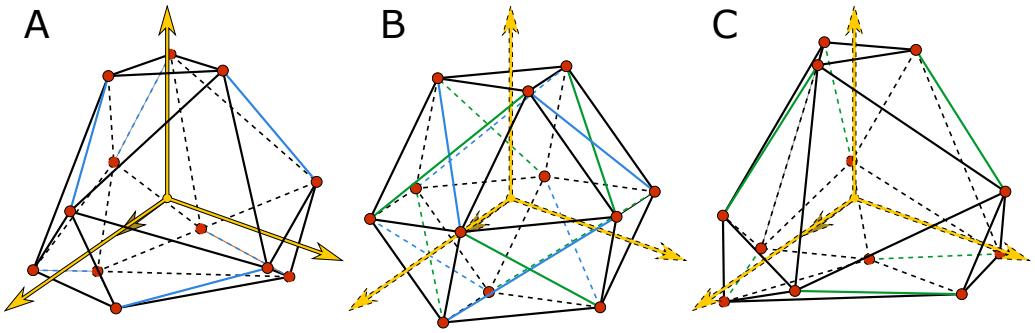


Figure 3: Three convex hulls of COMs of perfectly symmetrical tetrahedral assemblies. The black edges correspond to the maximum common topology between all the possible tetrahedral configurations. The green and blue edges are present only in some convex hull topologies. (A) and (C) show two typical topologies, while (B) shows a degenerated example, where some of the convex hull faces are rectangular instead of being triangular. By applying some random noise to the assembly shown in (B), we can obtain a convex hull with randomly chosen green and blue edges.

holds for dihedral symmetry groups. These, however, will have one generator being r_n, a rotation about the n-fold axis by an angle $\frac{2\pi}{n}$ instead of r_3 as the first generator. The number of elements is dependent on their order, $2n$ elements for D_n. It is easy to demonstrate that all the elements in the groups can be obtained from any initial element by successively applying a combination of the two *generator* rotations, such that the Cailey graphs are connected, as it is shown in Figure 4. We can also see that a combination of the two *generator* rotations only produces the group elements, such that the graphs are finite.

2.9. Geometry of multiple axes of symmetry

A rotation operator can be uniquely represented by an axis and an angle of rotation [13]. In our case, to determine the rotation operators, it is convenient to only work with their axes, since the angles are already constrained by the symmetry group. These axes are always placed in an identical configuration with respect to each other, and the position of two axes is sufficient to determine the positions of all other axes [20]. Thus we construct our basis with the axes of the two *generator* rotations defined above. The angle a between the two basis axes is uniquely defined by the type of symmetry group, as it is shown in Figure 5. It is more convenient, however, to use cosine of

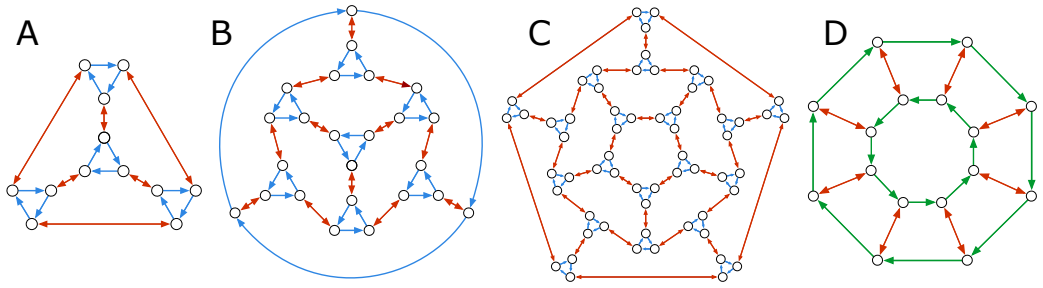


Figure 4: Cailey graph representation of the different cubic groups, (A) the tetrahedral, (B) the octahedral, (C) the icosahedral, and (D) the dihedral D_8. Each vertex represents a group element, which is also a rotation of the symmetrical assembly. The directed edges are the *generator* rotations applied to the group elements. The blue edges are the r_3 *generator* rotations, the red edges are the r_2 *generator* rotations, and the green edges are the r_n *generator* rotations. Each vertex has *indegree* and *outdegree* of two, such that the graph is balanced.

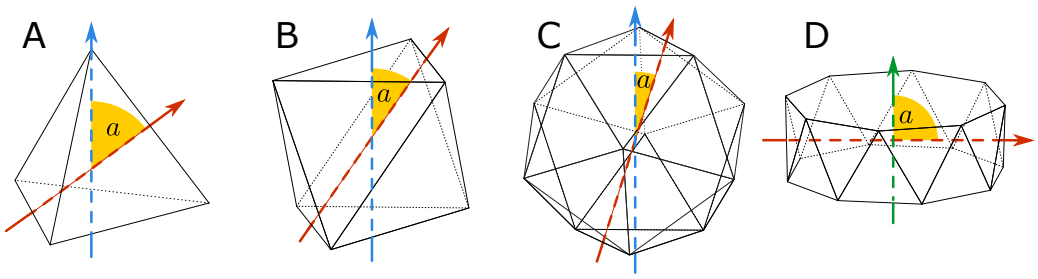


Figure 5: Four high-order point symmetry groups, (a) tetrahedral, (b) octahedral, (c) icosahedral and (d) dihedral group D_8. Two *generator* axes for each of the groups are shown. The 3-fold axis is colored in blue, the 2-fold is colored in red, and the 8-fold axis of the D_8 group is colored in green. The angle a between the *generator* symmetry axes is highlighted in yellow. These two axes are sufficient to describe the entire symmetry group.

the angle $\alpha = \cos(a)$, such that

$$\alpha_{\text{Tetrahedral}} = \frac{1}{\sqrt{3}} \qquad \alpha_{\text{Octahedral}} = \sqrt{\frac{2}{3}} \qquad \alpha_{\text{Icosahedral}} = \frac{\phi}{\sqrt{3}}, \tag{7}$$

where ϕ is the golden number $(\sqrt{5}+1)/2$. The angles a will be then

$$a_{\text{Tetrahedral}} \approx 54.7^\circ \qquad a_{\text{Octahedral}} \approx 35.3^\circ \qquad a_{\text{Icosahedral}} \approx 20.9^\circ. \tag{8}$$

It is interesting to note that the dihedral symmetry group D_n can also be rigorously described with two *generator* axes. One is a n-fold axis $\vec{v}_n$ defining

the C_n symmetry and the second is a 2 fold axis $\vec{v}_2$ perpendicular to it, such that $\alpha_{\text{Dihedral}} = 0$ and $a_{\text{Dihedral}} = 90°$. For any 3D rotation r_g, its rotation axis can be expressed in a basis spanned by 3 axes $\vec{v}_2$, $\vec{v}_3$, and $\vec{v}_3 \times \vec{v}_2$ (respectively $\vec{v}_2$, $\vec{v}_n$, and $\vec{v}_n \times \vec{v}_2$ for a dihedral group D_n), such that the associated rotation quaternion $\hat{Q}_g$ is written as

$$\hat{Q}_g \equiv [s_g, \mathbf{q}_g] = [s_g, a_g \mathbf{v}_3 + b_g \mathbf{v}_2 + c_g \mathbf{v}_3 \times \mathbf{v}_2] \tag{9}$$

for cubic groups and

$$\hat{Q}_g \equiv [s_g, \mathbf{q}_g] = [s_g, a_g \mathbf{v}_n + b_g \mathbf{v}_2 + c_g \mathbf{v}_n \times \mathbf{v}_2] \tag{10}$$

for dihedral groups. The rotation quaternion $\hat{Q}_2 \hat{Q}_g$ obtained by applying r_2 after r_g has the following coefficients,

$$\begin{aligned} s &= -(\alpha a_g + b_g) \\ a &= -(c_g) \\ b &= s_g - (\alpha c_g) \\ c &= -(a_g). \end{aligned} \tag{11}$$

Similarly, the rotation quaternion $\hat{Q}_3 \hat{Q}_g$ obtained by applying r_3 after r_g has the following coefficients,

$$\begin{aligned} s &= -\frac{s_g}{2} + \frac{\sqrt{3}}{2}(a_g + \alpha b_g) \\ a &= -\frac{a_g}{2} + \frac{\sqrt{3}}{2}(s_g + \alpha c_g) \\ b &= -\frac{b_g}{2} + \frac{\sqrt{3}}{2} c_g \\ c &= -\frac{c_g}{2} + \frac{\sqrt{3}}{2} b_g. \end{aligned} \tag{12}$$

Finally, for a dihedral group D_n, the rotation quaternion $\hat{Q}_n\hat{Q}_g$ has the following coefficients,

$$\begin{aligned} s &= \cos\left(\frac{2\pi}{n}\right) s_g + \sin\left(\frac{2\pi}{n}\right) a_g \\ a &= \cos\left(\frac{2\pi}{n}\right) a_g + \sin\left(\frac{2\pi}{n}\right) s_g \\ b &= \cos\left(\frac{2\pi}{n}\right) b_g + \sin\left(\frac{2\pi}{n}\right) c_g \\ c &= \cos\left(\frac{2\pi}{n}\right) c_g + \sin\left(\frac{2\pi}{n}\right) b_g. \end{aligned} \tag{13}$$

2.10. Optimization of the rotations

Here we will use the same notations as in our previous work on cyclic symmetries [1] . From now on, for simplicity, we will only write equations for the cubic group. Indeed, the equations for the dihedral group are obtained by substituting the index n for the index 3. Our goal is to minimize the loss function defined in equation 4. For each element g of the chosen symmetry group, the contribution to the loss function is the RMSD between $r_g(A)$ and $A_g = \{\mathbf{a}_{\sigma_g(i),j}\}$. According to the RMSD master equation (3) from [1] with $B = A_g$, we can say that A and B have the same COM, so the translational part of RMSD becomes null and we obtain

$$\mathrm{RMSD}^2(r_g(A), A_g) = \frac{4}{N}\mathbf{q}^T\mathbf{I}_g\mathbf{q} + 4s\mathbf{q}^T\mathbf{x}_{g\perp} + x_{gs}, \tag{14}$$

where

$$\mathbf{I}_g = \begin{pmatrix} \sum(y_{i,j}y_{\sigma_g(i),j} + z_{i,j}z_{\sigma_g(i),j}) & -\sum(x_{\sigma_g(i),j}y_{i,j} + x_{i,j}y_{\sigma_g(i),j})/2 & -\sum(x_{\sigma_g(i),j}z_{i,j} + x_{i,j}z_{\sigma_g(i),j})/2 \\ -\sum(x_{i,j}y_{\sigma_g(i),j} + x_{\sigma_g(i),j}y_{i,j})/2 & \sum(x_{i,j}x_{\sigma_g(i),j} + z_{i,j}z_{\sigma_g(i),j}) & -\sum(y_{\sigma_g(i),j}z_{i,j} + y_{i,j}z_{\sigma_g(i),j})/2 \\ -\sum(x_{i,j}z_{\sigma_g(i),j} + x_{\sigma_g(i),j}z_{i,j})/2 & -\sum(y_{i,j}z_{\sigma_g(i),j} + y_{\sigma_g(i),j}z_{i,j})/2 & \sum(x_{i,j}x_{\sigma_g(i),j} + y_{i,j}y_{\sigma_g(i),j}) \end{pmatrix}, \tag{15}$$

and

$$\begin{aligned} \mathbf{x}_{g\perp} &= \sum_{i,j} \mathbf{a}_{\sigma_g(i),j} \times \mathbf{a}_{i,j}/N \\ x_{gs} &= \sum_{i,j} (\mathbf{a}_{i,j} - \mathbf{a}_{\sigma_g(i),j})^2/N. \end{aligned} \tag{16}$$

Our aim will be to minimize the sum of squared RMSDs over all elements g of the group Γ. Let us first assume that we know the value of one of the two axes $\mathbf{v}_3$ or $\mathbf{v}_2$, for example, $\mathbf{v}_3$. In practice, we first compute $\mathbf{v}_3$ axis as a cyclic axis using the method from [1], then we alternate the computations of $\mathbf{v}_2$ and $\mathbf{v}_3$ considering the other axis as known. This method converges to machine precision in about 10 iterations. Thanks to the RMSD master equation, we can write the loss function as a function of the axis $\mathbf{v}_2$ as follows,

$$\begin{aligned}
&\sum_{g\in\Gamma} \mathrm{RMSD}_g^2(\mathbf{v_2}) = \\
&\mathbf{v}_2^T \left(\sum_{g\in\Gamma} b_g^2 \frac{4}{N}\mathbf{I}_g + 2\sum_{g\in\Gamma} b_g c_g \frac{4}{N}\mathbf{I}_g[\mathbf{v}_3]_\times + \sum_{g\in\Gamma} c_g^2 {[\mathbf{v}_3]_\times}^T \frac{4}{N}\mathbf{I}_g[\mathbf{v}_3]_\times \right) \mathbf{v}_2 \\
&+ \left(2\sum_{g\in\Gamma} a_g b_g \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g + 2\sum_{g\in\Gamma} a_g c_g \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g + 4\sum_{g\in\Gamma} s_g b_g \mathbf{x}_{g\perp}^T \right) \mathbf{v}_2 \\
&+ \sum_{g\in\Gamma} a_g^2 \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g \mathbf{v}_3 + 4\sum_{g\in\Gamma} s_g b_g \mathbf{x}_{g\perp}^T \mathbf{v}_3 + \sum_{g\in\Gamma} x_{gs}.
\end{aligned} \tag{17}$$

We can rewrite this equation as the following minimization problem with respect to $\mathbf{v_2}$,

$$\begin{aligned}
\arg\min_{\mathbf{v}_2} \quad & \mathbf{v}_2^T \mathbf{A} \mathbf{v}_2 + \mathbf{b}^T \mathbf{v}_2 + c \\
\text{s.t.} \quad & \begin{cases} \mathbf{v}_2^T \mathbf{v}_2 = 1 \\ \mathbf{v}_3^T \mathbf{v}_2 = \alpha. \end{cases}
\end{aligned} \tag{18}$$

The two constraints come from the unit norm of the rotation axes and the geometry of the generator axes. The above equations has the following coefficients,

$$\begin{aligned}
\mathbf{A} &= \sum_{g\in\Gamma} b_g^2 \frac{4}{N}\mathbf{I}_g + 2\sum_{g\in\Gamma} b_g c_g \frac{4}{N}\mathbf{I}_g[\mathbf{v}_3]_\times + \sum_{g\in\Gamma} c_g^2 {[\mathbf{v}_3]_\times}^T \frac{4}{N}\mathbf{I}_g[\mathbf{v}_3]_\times \\
\mathbf{b}^T &= 2\sum_{g\in\Gamma} a_g b_g \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g + 2\sum_{g\in\Gamma} a_g c_g \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g + 4\sum_{g\in\Gamma} s_g b_g \mathbf{x}_{g\perp}^T \\
c &= \sum_{g\in\Gamma} a_g^2 \mathbf{v}_3^T \frac{4}{N}\mathbf{I}_g \mathbf{v}_3 + 4\sum_{g\in\Gamma} s_g b_g \mathbf{x}_{g\perp}^T \mathbf{v}_3 + \sum_{g\in\Gamma} x_{gs}.
\end{aligned} \tag{19}$$

Similar equations can be written for the optimization of the loss function with respect to $\mathbf{v_3}$.

2.11. 2D trust-region optimization problem

The optimization problem (18) can be efficiently solved by reducing it to the standard form of the *trust-region subproblem*. However, in our particular case, we can use one of the constraints in eq. (18) to project the optimization problem to a two-dimensional subspace. This allows us to solve it analytically, as we explain below.

First of all, it is convenient to chose an orthonormal basis $(\mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_3)$ and rewrite the vector $\mathbf{v}_2$ in this basis as

$$\mathbf{v}_2 = \alpha\mathbf{v}_3 + x\mathbf{v}_x + y\mathbf{v}_y. \tag{20}$$

Then, the optimization problem (18) reduces to

$$\begin{aligned} \arg\min_{x,y} \quad & x^2\left(\mathbf{v}_x^T\mathbf{A}\mathbf{v}_x\right) + 2xy\left(\mathbf{v}_x^T\mathbf{A}\mathbf{v}_y\right) + y^2\left(\mathbf{v}_y^T\mathbf{A}\mathbf{v}_y\right) \\ & + x\left(2\alpha\mathbf{v}_x^T\mathbf{A}\mathbf{v}_3 + \mathbf{b}^T\mathbf{v}_x\right) + y\left(2\alpha\mathbf{v}_y^T\mathbf{A}\mathbf{v}_3 + \mathbf{b}^T\mathbf{v}_y\right) \\ & + \alpha^2\mathbf{v}_3^T\mathbf{A}\mathbf{v}_3 + \mathbf{b}^T\mathbf{v}_3 + c \\ \text{s.t.} \quad & x^2 + y^2 = 1 - \alpha^2. \end{aligned} \tag{21}$$

To solve it, we find stationary points of the corresponding Lagrangian $L(x, y, \lambda)$,

$$L(x, y, \lambda) = kx^2 + 2lxy + my^2 + 2px + 2qy + \lambda(x^2 + y^2 - 1 + \alpha^2), \tag{22}$$

with the following coefficients

$$\begin{aligned} k &= \mathbf{v}_x^T\mathbf{A}\mathbf{v}_x \\ l &= \mathbf{v}_x^T\mathbf{A}\mathbf{v}_y \\ m &= \mathbf{v}_y^T\mathbf{A}\mathbf{v}_y \\ p &= \alpha\mathbf{v}_x^T\mathbf{A}\mathbf{v}_3 + \frac{1}{2}\mathbf{b}^T\mathbf{v}_x \\ q &= \alpha\mathbf{v}_y^T\mathbf{A}\mathbf{v}_3 + \frac{1}{2}\mathbf{b}^T\mathbf{v}_y. \end{aligned} \tag{23}$$

Assigning the partial derivatives of the Lagrangian to zeros, we arrive to the following system of equations,

$$\begin{cases} kx + ly + p + \lambda x = 0 \\ lx + my + q + \lambda y = 0 \\ x^2 + y^2 = 1 - \alpha^2. \end{cases} \tag{24}$$

After eliminating λ we obtain

$$\begin{cases} lx^2 + (m-k)xy - ly^2 + qx - py = 0 \\ x^2 + y^2 = 1 - \alpha^2. \end{cases} \tag{25}$$

Finally, we exclude the last equation by changing the variables and introducing the new optimization variable t,

$$x = \frac{2t\sqrt{1-\alpha^2}}{1+t^2}; \quad y = \frac{(1-t^2)\sqrt{1-\alpha^2}}{1+t^2}. \tag{26}$$

Then, making the change of variables and multiplying the first equation by non-zero $(1+t^2)^2$ we obtain,

$$\begin{aligned} &\left(-l(1-\alpha^2) + pt\sqrt{1-\alpha^2}\right) t^4 + 2\left((1-\alpha^2)(k-m) + t\sqrt{1-\alpha^2}q\right) t^3 \\ &+ 6(1-\alpha^2)lt^2 + 2\left(1-\alpha^2)(-k+m) + \sqrt{1-\alpha^2}q\right) t - (1-\alpha^2)l - p = 0. \end{aligned} \tag{27}$$

This is our final fourth-order algebraic equation, whose roots can be found analytically [21]. After finding all of its roots, we discard the complex ones, then compute the corresponding values of x and y, substitute them in the original quadratic function (21) and choose the pair of x and y that gives the smallest value. We also additionally test the case of $y = -\sqrt{1-\alpha^2}$ and $x = 0$ that has been excluded during the change of variables in eq. (26).

3. Results and Discussion

3.1. Examples

Figure 6 presents an example of symmetry axes detection for each of the cubic groups, i.e. tetrahedral, octahedral and icosahedral, and for a dihedral group of order 6. These assemblies do not possess any particular computational difficulty. Indeed, their asymmetric subunits are composed of a single chain, which makes the first step of optimization problem 5 trivial.

Some assemblies contain more chains than the number of asymmetric subunits expected from their point group symmetry. Each subunit thus must be composed of several chains. For example, Figure 7 shows the 5t0v structure, which is an octahedral assembly with 48 chains and a stoichiometry of A24B24. This example demonstrates that our method determines symmetry axes in assemblies where the asymmetric subunits are composed of

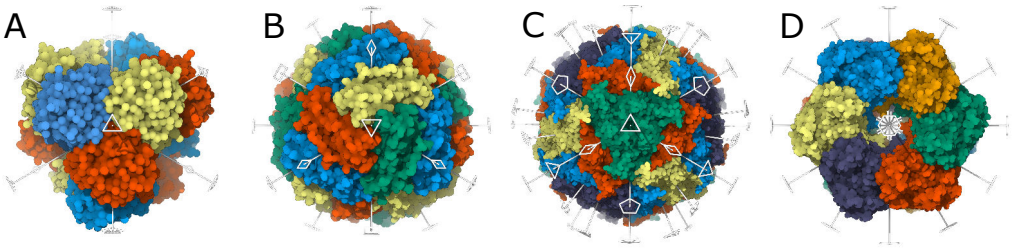


Figure 6: Four examples of symmetrical assemblies with their axes. All of these are seen from a 3-fold axis except for the last one, seen from a 6 fold axis. The order n of each axis is represented with a regular n-gone, except of order 2 represented with a rhombus. A – A tetrahedral assembly (1d0i) with the RMSD loss of 0.36 Å. B – An octahedral assembly (1bfr) with the RMSD loss of 0.22 Å. C – A perfect icosahedral assembly (1stm) with the RMSD loss of 0.0 Å. D – A dihedral D_6 assembly (1f52) with the RMSD loss of 0.20 Å. This illustration and all the illustrations below were produced in SAMSON (www.samson-connect.net).

multiple chains. We should also note that in this case it is important to rigorously take into account all the chains, since the angular difference in the axis determination can be as large as 1° if only chains A or B are considered.

While scanning the PDB, we found several assemblies that are classified with a low-order symmetry group, but can alternatively possess a higher symmetry group. For example, Figure 8 shows the 1ocw structure, which is a perfect C_4 assembly with a stoichiometry of A4B4 and the RMSD loss of 0 Å. Our algorithm also detects a D_4 pseudo-symmetry with the RMSD loss of 2.68 Å, which is rather low. The visual inspection of this protein confirms this possibility (see Fig. 8). Similarly, we also discovered some assemblies with cubic symmetries that were labelled as cyclic in the PDB database. Figure 9 shows two of such examples. One is the 4itv protein labelled as C_2 (RMSD loss of 4.44 Å), but also possessing a tetrahedral symmetry with the RMSD loss of 10.94 Å. The other is the 5hpn protein labelled as C_5 (RMSD loss of 0.68 Å), but also possessing an icosahedral symmetry with the RMSD loss of 0.56 Å.

3.2. Comparison with other methods

We compared our approach with two other published methods following the comparison strategy from our previous work on symmetry detection in cyclic protein assemblies [1]. More precisely, we compared it to the results published by David Avnir and colleagues [22, 23]. We will refer to it as to

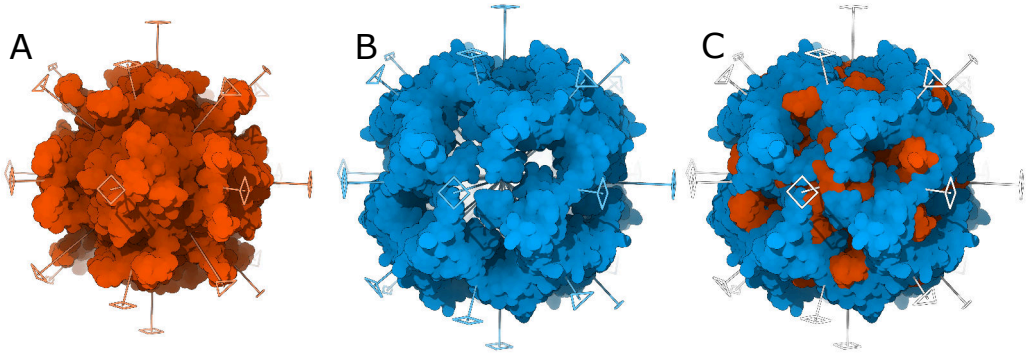


Figure 7: The 5t0v octahedral assembly. The homologous chains are colored with the same color. A – The chains of the first type form an octahedral assembly with the RMSD loss of 2.94 Å. B – The chains of the second type also form an octahedral assembly with the RMSD loss of 2.67 Å. The axes are slightly different from the first assembly, with about 1° of difference. C –y The axes are computed for the full assembly, with the RMSD loss of 2.83 Å.

CSM (Continuous Symmetry Measure). We also compared our method to the the one from Emmanuel Levy [3], and will refer to it as to Levy. Please refer to the first part of our paper [1] for more details.

For the comparison, we have selected all dihedral assemblies presented in the original CSM publications [22, 23]. These are listed in Table 1. We have also complemented these assemblies with three examples of cubic groups, 5x47 with tetrahedral symmetry, 4p18 with octahedral symmetry, and 4zor with icosahedral symmetry.

Table 1 lists the execution time and the symmetry measure (RMSD value) for the three tested methods. As in the cyclic case [1], it clearly shows that our method scales with the size of the input assembly much better than the two other methods. This is especially notecable for large assemblies. Regarding the accuracy of the obtained results, it is typically much better than in the Levy method for high-order symmetries. As we have mentioned in the first part of this work, comparison to CSM is trickier because this method considers more atoms than we do. Therefore, the additional atoms add more freedom to the CSM method when it chooses the correspondences between these, which can explain small differences in the computed RMSD values. For example, in the 1f52 case CSM reports a smaller RMSD measure than we do (0.15 Å vs. 0.19 Å).

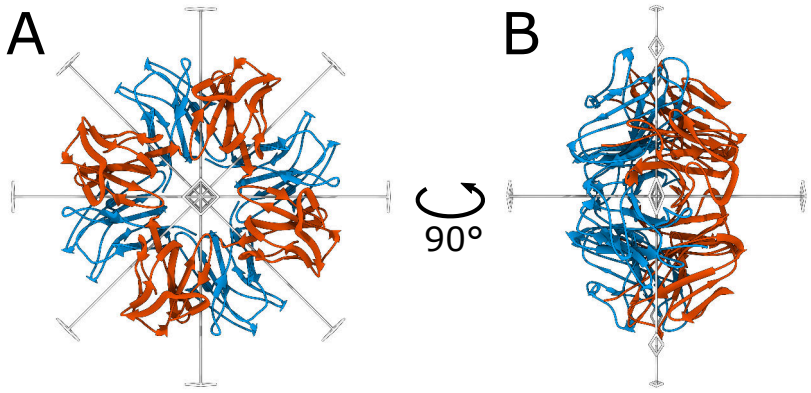


Figure 8: The 1ocw protein colored in blue for the A chains and red for the B chains. A – as seen from the 4-fold axis. B – as seen from a 2-fold axis computed with our method.

3.3. Exhaustive analysis of symmetrical structures in the PDB

To demonstrate the efficiency of our approach, we exhaustively analyzed all the structures labelled as symmetrical in the PDB. To do so, we downloaded their biological assemblies (about 40,800 cyclic, 9,800 dihedral and 1,300 cubic examples as for January 2018) and assessed the symmetry for each of these. Figure 10 plots the distribution of the RMSD symmetry measures for assemblies with different types of symmetry. We should note that there are many structures with a very low RMSD value (< 0.001Å), which is the precision of the pdb format. These are typically obtained by replicating subunits with crystallographic symmetry or BIOMT transforms, so they have a perfect symmetry. Regarding all other structures, we can see that all the three distributions of cyclic, dihedral, and cubic groups follow the same law in the log-log scale. The maxima of the distributions belong to the range of 0.2-0.5 Å, and there are no noticeable differences between the shapes of all of these.

Another interesting question we are able to answer using our tool is whether the degree of asymmetry is related to the size of the assembly under consideration. In other words, we can study if the RMSD symmetry measure is related with the radius of gyration of the symmetrical assemblies. A geometrical intuition would suggest that as the angular uncertainty should stay constant with the size of the assembly, and of protein assembly grows larger, the imperfections of its symmetry become more pronounceable. Visually, we would expect a linear correlation between the RMSD symmetry

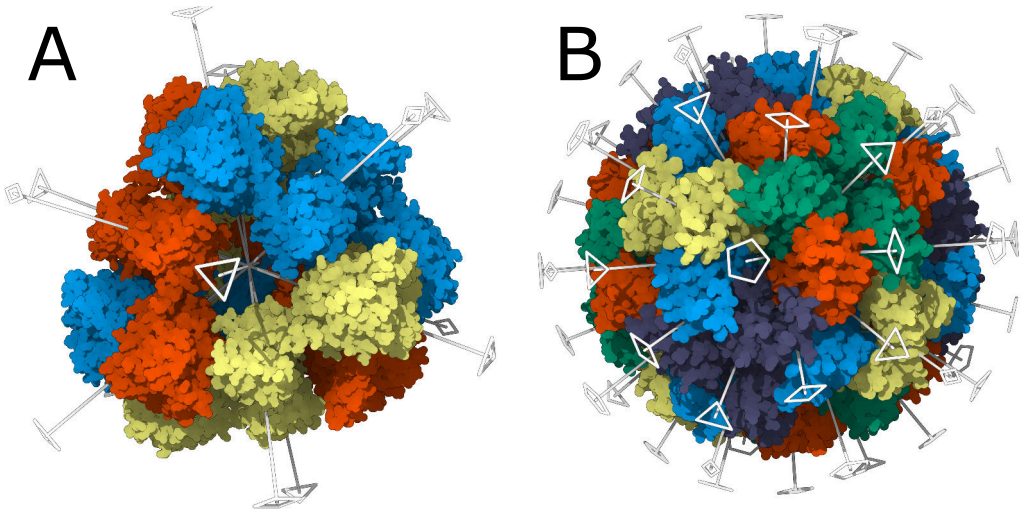


Figure 9: A – The 4itv protein classified in PDB as C_2 (RMSD loss of 4.44 Å), also has a tetrahedral symmetry with the RMSD loss = 10.94 Å. B – The 5hpn protein classified in PDB as C_5 (RMSD loss of 0.68Å), also has an icosahedral symmetry with the RMSD loss = 0.56 Å.

PDB Code	Group	RMSD(AnAnaS)	RMSD(CSM)	RMSD(Levy)	AnAnaS Time[a]	CSM Time[b]	Levy Time[a]
1mso[c]	D_3	1.36 Å	-	1.39 Å	0.13 s	3.7 s	0.49 s
2hhb	D_2	1.64 Å	2.43 Å	1.64 Å	0.05 s	12.2 s	0.28 s
2nwc	D_7	0.81 Å	-	0.89 Å	0.63 s	3950 s	2.3 s
2rgw	D_3	0.34 Å	0.39 Å	0.47 Å	0.23 s	-	1.8 s
1odi	D_3	0.35 Å	0.50 Å	0.47 Å	0.14 s	-	1.5 s
1f52	D_6	0.19 Å	0.15 Å	0.54 Å	1.21 s	-	16.6 s
5x47	T	0.85 Å	-	1.02 Å	0.32 s	-	5.62 s
4p18	O	0.19 Å	-	2.13 Å	3.1 s	-	131 s
4zor[c]	I	1.05 Å	-	2.38 Å	18.8 s	-	1118 s

[a] AnAnaS and Levy times were measured on a Windows laptop equipped with an Intel i7 @ 3.1 GHz.
[b] CSM times and CSM symmetry measures were taken from [22] and [23] with a different, 7 year older, CPU. However, we believe that the order of magnitude of these timings is still correct.
[c] For these structures, the biological assembly was used.

Table 1: Comparative results between AnAnaS, CSM and Levy methods for dihedral and cubic molecular assemblies.

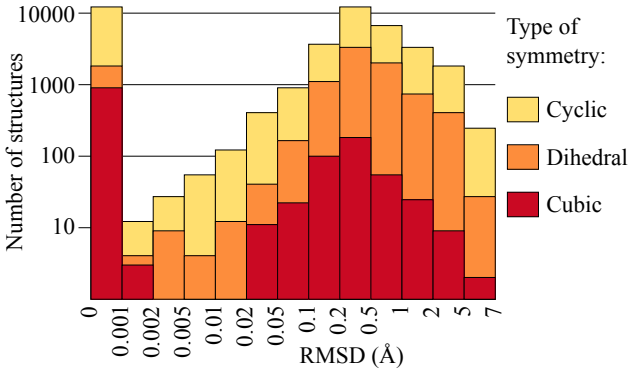


Figure 10: Distribution of the RMSD symmetry measure for different types of symmetry shown in a log-log scale.

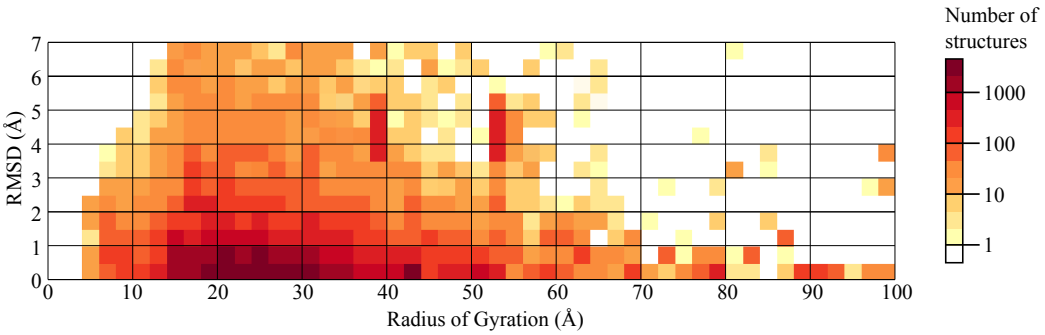


Figure 11: Distribution of the RMSD symmetry measure with respect to the radius of gyration for non-perfectly symmetrical assemblies from PDB.

measure and the radius of gyration of the assemblies. However, it is not the case in reality. Indeed, Figure 11 does not demonstrate any clear relation between the size and the imperfection of the PDB assemblies, and the correlation between these two variables is only about 0.1. Interestingly enough, large assemblies are very well organized with sufficiently small values of the RMSD measure. This is one of the reasons behind our choice of RMSD as the symmetry measure instead of its normalization by the size of the structure (as it is often done in other methods [22, 23]). We should specifically add that in the case of very small assemblies, we consider them symmetric only if the corresponding RMSD measure is smaller than half of the radius of gyration of the assembly.

3.4. How good are symmetry annotations in the PDB?

Our tool also allows to assess the overall quality of annotations of symmetrical assemblies in the PDB. More precisely, we compared the highest symmetry group suggested by our method with the group provided in the PDB. If these two groups are different, there are two types of possible errors. First, one of the two groups can be a subgroup of the other one (e.g. C_4 is a subgroup of D_4). This type of errors simply results from a difference of sensibility between the annotation methods. We call the groups *compatible*. Second, the two groups may also be *incompatible* (e.g. C_4 and D_5). This case means that one of the two results is wrong and a careful visual inspection is generally required.

Table 2 lists the results for 51,358 PDB structures. In 50,378 cases (98.1% of all the cases), the symmetry group annotated by the PDB is the one found by our method. These cases are located at the green diagonal of the table. Red cells show the incompatible groups, while white cells show the compatible groups. Our method is generally more sensitive compared to the PDB annotation. Indeed, there are 845 structures (1.6%) for which it finds a higher order compatible group, while only in 125 cases (0.2%) the PDB annotated compatible group has a higher order. Finally, there are only 13 cases (0.03%) that present incompatible groups. We have visually inspected all of these structures. The two of these annotated as T and detected as C_5 are 4aod and 4aoe, for which the biological assemblies are indeed C_5. The 11 other cases have uncertainties between C_2 and C_3 annotation. In all of these cases, both symmetries are detected by our method, and the difference of RMSD between the two symmetries is smaller than 1 Å. Moreover, some of these examples have less than 5 amino acids in each chain, and are at the limit of the usability of the annotation techniques. We can also mention two particular cases. One is 3alz, for which both perfect C_3 and C_2 axes are detected, and is actually a part of a D_3 assembly. The other is 3aqq, which is annotated as C_2 in the PDB, but looks much more like a partial C_3 assembly.

The first column of Table 2 lists 75 structures for which AnAnaS was not able to detect symmetry. There are 4 reasons that explain this:

- For the 6 icosahedral structures, we ran out of memory at the discrete optimization step. Thus, no results were outputted and we considered these cases as assymetric.

- Some structures have missing or additional chains that are not supported by our program. For example, 2zl2 has a D_7 symmetry but

PDB \ AnAnaS	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	D_2	D_3	D_4	D_5	D_6	D_7	D_8	T	O	I	Total
C_2	54	33091	8	23		6			470	15	7	1	1	205	2				33883
C_3	2	3	4188			16				60									4269
C_4	1	2		1046				7			4								1060
C_5	6				561							1							568
C_6		2	2			411							1						416
C_7							104							6					110
C_8								34							3				37
D_2	3	26							6571		6		1			2			6609
D_3		8	5							1939									1952
D_4	1	1									654				5				661
D_5		1										236							237
D_6													106						106
D_7	1						1							99					101
D_8															34				34
T					2											359	3		364
O																	329		329
I	6															2		617	625

Table 2: Summary of the symmetry groups annotated in the PDB (rows) against the ones discovered by AnAnaS (columns). Red cells mark incompatible groups, while white cells mark compatible groups and green cells mark identical groups. For example, first cell shows that there are 54 structures annotated as C_2 in the PDB for which we found a C_1 symmetry (i.e. no symmetry).

contains 24 chains, 10 of them being very small peptides. AnAnaS expects a multiple of 14 chains as input to test a D_7 symmetry and, therefore, does not test it. However, if we remove these small peptides, we detect a D_7 symmetry with an RMSD of 0.35 Å.

- Some structures are at the edge of the threshold that we set up for the assemblies to be symmetric. More precisely, as we explain it below, RMSD must be smaller than 7 Å and also smaller than half of the radius of gyration.

- Finally, some structures do not possess the symmetry annotated in the PDB. For example, 2ol9 is the structure of two identical peptides translated with respect to each other, and these are annotated as C_2, while a C_2 symmetry would necessarily require a rotation between the two peptides.

3.5. Computational details

We implemented the AnAnaS method using the C++ programming language. It is available as a standalone executable and also as a module with graphical user interface for the SAMSON software platform. We can also provide the source code upon request.

We have exhaustively assessed our program with all the structures labeled as symmetric in the PDB. This demonstrates the reliability and robustness of our method overall, and its heuristic for the discrete optimization steps in particular. Running the tests on all of these structures took us about 10 hours on a Windows laptop equipped with an Intel Core i7 @ 3.1 GHz CPU. For all the examples we tested, the running time was largely dominated by the multiple sequence alignment, which is required to compare the relevant alpha carbons in different subunits. Only in one case (2qzv) with a D_{48} symmetry, the computational bottleneck turned out to be the graph matching step.

We should also say that if no symmetry group is specified by a user, then the program exhaustively tests all the symmetry groups that are consistent with the number of chains in the input assembly. Also, we label an assembly as symmetric only if the corresponding RMSD measure is smaller than 7 Å and smaller than half of its radius of gyration. The second condition is added to filter out very small asymmetric assemblies.

4. Conclusions

This work extends our previous cyclic symmetry detection method [1] for high-order point groups. It required to develop a robust heuristic algorithm that perceives the correspondence between asymmetric subunits, and also to extend the constrained quadratic optimization problem from [1] to multiple symmetry axes with mutual constraints. Using the quaternion arithmetic, we expressed the constrained optimization problem as a 2D trust-region subproblem and found its solution analytically. We have compared out method with two other published techniques that can detect symmetry in high-order symmetrical assemblies and demonstrated that it is generally much more robust and efficient.

We have demonstrated the efficiency of our method on all the structures marked as symmetric in the PDB, including those with multiple chains per asymmetric subunit or with pseudo-symmetry. It allowed us to verify symmetry annotations in the PDB and detect several inconsistencies in the

annotations. For example, in 1.6 % of the cases, we detected a higher symmetry group compared to those provided in the PDB. We have also compared structural organization of protein assemblies with different point group symmetries and concluded that these follow the same distribution laws. Finally, we have detected that the angular impurity in symmetry does not scale with the size of the assemblies. More precisely, very often these are the largest and high-order symmetry systems that are organized the most regularly.

The method is available at https://team.inria.fr/nano-d/software/ananas/. The SAMSON GUI-assisted module is available at http://samson-connect.net/.

5. Aknowledgements

The authors thank Nikolay Mayorov for his 2D trust-region algorithm developed during Google Summer of Code 2015, and Sergei Khashin from Ivanovo State University for his forth order polynomial solver available at http://math.ivanovo.ac.ru/dalgebra/Khashin/poly/index.html. The authors also thank Elvira Kinzina and Andrei Kazennov from MIPT Moscow for their support at the initial stage of the project.

Funding

This work has been supported by L'Agence Nationale de la Recherche (grant number ANR-15-CE11-0029-03).

References

[1] G. Pagès, E. Kinzina, S. Grudinin, Analytical symmetry detection in cyclic protein assemblies, J Struc Biol (2018) In Press.

[2] P. W. Rose, A. Prlić, A. Altunkaya, C. Bi, A. R. Bradley, C. H. Christie, L. D. Costanzo, J. M. Duarte, S. Dutta, Z. Feng, et al., The rcsb protein data bank: integrative view of protein, gene and 3d structural information, Nucleic Acids Res (2016) gkw1000.

[3] E. D. Levy, J. B. Pereira-Leal, C. Chothia, S. A. Teichmann, 3d complex: a structural classification of protein complexes, PLoS Comput Biol 2 (2006) e155.

[4] E. D. Levy, E. B. Erba, C. V. Robinson, S. A. Teichmann, Assembly reflects evolution of protein complexes, Nature 453 (2008) 1262–1265.

[5] N. P. King, W. Sheffler, M. R. Sawaya, B. S. Vollmar, J. P. Sumida, I. André, T. Gonen, T. O. Yeates, D. Baker, Computational design of self-assembling protein nanomaterials with atomic level accuracy, Science 336 (2012) 1171–1174.

[6] N. P. King, J. B. Bale, W. Sheffler, D. E. McNamara, S. Gonen, T. Gonen, T. O. Yeates, D. Baker, Accurate design of co-assembling multi-component protein nanomaterials, Nature 510 (2014) 103–108.

[7] J. B. Bale, S. Gonen, Y. Liu, W. Sheffler, D. Ellis, C. Thomas, D. Cascio, T. O. Yeates, T. Gonen, N. P. King, et al., Accurate design of megadalton-scale two-component icosahedral protein complexes, Science 353 (2016) 389–394.

[8] Y. Hsia, J. B. Bale, S. Gonen, D. Shi, W. Sheffler, K. K. Fong, U. Nattermann, C. Xu, P.-S. Huang, R. Ravichandran, et al., Design of a hyperstable 60-subunit protein icosahedron, Nature 535 (2016) 136–139.

[9] A. Prlić, A. Yates, S. E. Bliven, P. W. Rose, J. Jacobsen, P. V. Troshin, M. Chapman, J. Gao, C. H. Koh, S. Foisy, et al., Biojava: an open-source framework for bioinformatics in 2012, Bioinformatics 28 (2012) 2693–2695.

[10] B. K. Horn, Closed-form solution of absolute orientation using unit quaternions, J Opt Soc Am A 4 (1987) 629–642.

[11] R. Diamond, A note on the rotational superposition problem, Acta Crystallogr A 44 (1988) 211–216.

[12] S. K. Kearsley, On the orthogonal transformation used for structural comparisons, Acta Crystallogr A 45 (1989) 208–210.

[13] P. Popov, S. Grudinin, Rapid determination of rmsds corresponding to macromolecular rigid body motions, J Comput Chem 35 (2014) 950–956.

[14] E. Neveu, P. Popov, A. Hoffmann, A. Migliosi, X. Besseron, G. Danoy, P. Bouvry, S. Grudinin, RapidRMSD : Rapid determination of RMSDs corresponding to motions of flexible molecules, 2018. Unpublished.

[15] S. Henikoff, J. G. Henikoff, Amino acid substitution matrices from protein blocks, Proc Natl Acad Sci USA 89 (1992) 10915–10919.

[16] R. C. Edgar, Muscle: multiple sequence alignment with high accuracy and high throughput, Nucleic Acids Res 32 (2004) 1792–1797.

[17] B. Grünbaum, Convex polytopes: Prepared by volker kaibel, victor klee, and günter ziegler (graduate texts in mathematics) (2003).

[18] J. R. Ullmann, An algorithm for subgraph isomorphism, J ACM 23 (1976) 31–42.

[19] P. Cayley, Desiderata and suggestions: No. 2. the theory of groups: graphical representation, Am J Math 1 (1878) 174–176.

[20] D. W. Ritchie, S. Grudinin, Spherical polar fourier assembly of protein complexes with arbitrary point group symmetry, J Appl Crystallogr 49 (2016) 158–167.

[21] S. Neumark, Solution of cubic and quartic equations, Elsevier, 2014.

[22] C. Dryzun, A. Zait, D. Avnir, Quantitative symmetry and chirality—a fast computational algorithm for large structures: Proteins, macromolecules, nanotubes, and unit cells, J Comput Chem 32 (2011) 2526–2538.

[23] M. Pinsky, A. Zait, M. Bonjack, D. Avnir, Continuous symmetry analyses: Cnv and dn measures of molecules, complexes, and proteins, J Comput Chem 34 (2013) 2–9.