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

Guillaume Pagès, Sergei Grudinin⁎

Inria, Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, Grenoble 38000, France

ARTICLE INFO

Keywords:
Point-group symmetry
Cubic groups
Group theory
Discrete optimization
Protein structure
Continuous optimization

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 symmetric assemblies from the Protein Data Bank (PDB). It allowed us to study structural organization of symmetric 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.

1. Introduction

Symmetric protein complexes are very common in nature, as has been highlighted in our previous work on symmetry detection in cyclic protein assemblies (Pagès et al., 2018), and many of these are deposited to the Protein Data Bank (PDB) (Berman et al., 2000). As function of proteins is very often determined by their structure, it appears that complex function requires complex structures (Levy et al., 2006; Levy et al., 2008). High-order symmetries are thus essential to build large and complex protein assemblies. Dihedral and cubic groups are over-represented 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 (King et al., 2012; King et al., 2014; Bale et al., 2016; Hsia et al., 2016).

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 (Prlić et al., 2012), are massively used to display the symmetry axes on the PDB website (Berman et al., 2000).

Inspired by the quaternion arithmetic applied to the best superposition of a set of points (Horn, 1987; Diamond, 1988; Kearsley, 1989) together with our recent developments (Popov and Grudinin, 2014; Neveu et al., 2018; Pagès et al., 2018), here we propose a new symmetry measure and an analytical method to find the best symmetry axes of a symmetric 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 (Pagès et al., 2018), 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 group symmetry. 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 (Pagès et al., 2018), in this paper we will be mainly dealing with $3 \times 3$ matrices and 3-vectors. Therefore, bold upper case letters (i.e. $A$) will denote matrices, bold lower case letters (i.e. $b$) 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 $\vec{a}$.

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 $\Gamma \in \{C_n, D_n, T, O, I\}$ 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 (Pagès et al., 2018), we will express the symmetry measure of a molecular assembly using the root mean square deviation (RMSD). Given two sets of $N$ points each, $A = \{a_i\}_{i=1}^N$ and $B = \{b_i\}_{i=1}^N$, the RMSD between them is given as

$$\text{RMSD}(A, B)^2 = \frac{1}{N} \sum_{1 \leq i, j \leq N} |a_i - b_j|^2. \quad (1)$$

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 $(\Gamma, \cdot)$, where $\Gamma$ is a set of elements supplied with a group operation $\cdot$. 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 $(\Gamma_1, \cdot_1)$ and $(\Gamma_2, \cdot_2)$, a homomorphism $f$ from $\Gamma_1$ to $\Gamma_2$ satisfies the following property,

$$\forall g, g' \in \Gamma_1: f(g \cdot_2 g') = f(g) \cdot_2 f(g'). \quad (2)$$

A homomorphism is called bijective if

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

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 $\sigma$ of the set $\{1, \ldots, n\}$ will be denoted $(\sigma(1), \ldots, \sigma(n))$. 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 $\Gamma$ symmetry if the rotations in $\Gamma$ keep this set globally invariant. More precisely, any rotation operator in $\Gamma$ will displace each point that is located outside of the rotation axis, but another point will take its place. It is natural to see $\Gamma$ as a rotation group, but also as a permutation group, since a rotation applied to a set of points will permute them. For example, Fig. 1 illustrates a $C_5$ point group, which has a bijective homomorphism with a rotation group and a bijective homomorphism with a permutation group.

2.4. Problem definition

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

$$\text{Loss}^2 = \frac{1}{|\Gamma| N_a N_b} \min_{\sigma, \tau} \sum_{g \in \Gamma} \sum_{i=1}^{N_a} \sum_{j=1}^{N_b} (\sigma(a_{g(i)} \tau^{-1}) - \sigma(a_{g(j)}))^2. \quad (4)$$

such that $g \mapsto \sigma$ and $g \mapsto \tau$ are the bijective homomorphisms from $\Gamma$ to subsets of $\Sigma_n$ 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 $\Gamma$. 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 $\tau$ 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 $\sigma$ 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
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 (Pagès et al., 2018). 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 $\Gamma$ 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 symmetric 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 symmetric 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 $\Gamma$-symmetric assembly. Then, the subunits are defined by appropriately choosing one chain from each of these sets as shown in Fig. 2. All the sets are $\Gamma$-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

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

where

- $d_{\text{seq}}(i, j)$ is the minus BLOSUM62 score (Henikoff and Henikoff, 1992) of the sequence alignment between chains $i$ and $j$, computed using the MUSCLE package (Edgar, 2004).
- $d_{\text{struct}}(i, j)$ 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 symmetric assembly, only a few values of these angles are possible. More precisely, if a group $\Gamma$ contains multiple symmetry axes $n_1, n_2, \ldots, n_k$ of orders $k_j, \ldots, k_j$, correspondingly, then the pairs of chains will be mutually rotated by angles $2\pi m/l$, where $1 \leq m < l$ with $l = j \ldots k_l$. We define $d_{\text{angle}}(i, j)$ as the absolute value of the difference between $\alpha(i, j)$ and the closest listed angle.
- $d_{\text{COM}}(i, j)$ 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, $d_{\text{max}}$ 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 $d_{\text{max}}$ to make the values of denominators sufficiently large.

Algorithm 1: Clustering algorithm.

```
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
```

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 Algorithm 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

![Fig. 2. Examples of finding subunits in a C5 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 drawn 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), and 1 and 9 in (C)). The other subunits are created using the correspondences between the chains in the sets. Note that the assembly 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.](image-url)
subunits. To do so, we first compute for each of the sets the symmetry axes corresponding to the group $\Gamma$. 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 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 $g \mapsto c_1$ from $\Gamma$ to subsets of $\mathcal{E}_n$ grows exponentially with the size of $\Gamma$, 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 symmetric system consisting of $n$ subunits, with the center of the symmetry located at the origin. Let $\{c_1|c_1 \in \mathcal{E}_n\}$ be the COMs of the subunits. Remark that $c_1$ are located on a sphere. The convex hull of $\{c_1|c_1 \in \mathcal{E}_n\}$, 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 the symmetry axes of order $s > 2$ passing through them. The topology of this graph does not depend on the choice of the selected perfectly symmetric system (see Fig. 3). For 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 $P$ (Grünaum et al., 2003) 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 (Ullmann, 1976). 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 $\Gamma$. This way, we ensure that the function $g \mapsto c_1$ 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 $r_1$ being a rotation about a 3-fold axis $v_3$ by an angle $\frac{2\pi}{3}$ for cubic groups, (respectively $r_2$ being a rotation about a n-fold axis $v_n$ by an angle $\frac{2\pi}{n}$ for dihedral groups) and the other $r_2$ being a rotation about a 2-fold axis $v_2$ by an angle $\pi$. 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 as vertices in a Cailey graph (Cayley, 1878), whose edges correspond to the two types of the generator rotations. A group element here can be seen as a certain rotation of the symmetric 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 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 $r_1$ 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 Fig. 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 (Popov and Grudinin, 2014). 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 (Ritchie and Grudinin, 2016). 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 Fig. 5. It is more convenient, however, to use cosine of the angle $\alpha = \cos(a)$, such that

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

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

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

It is interesting to note that the dihedral symmetry group $D_{2n}$ can also be rigorously described with two generator axes. One is a n-fold axis $v_n$ defining the $C_n$ symmetry and the second is a 2-fold axis $\tilde{v}_2$ perpendicular to it, such that $\alpha_{\text{Dihedral}} = 0$ and $\alpha_{\text{Icosahedral}} = 90^\circ$. For any 3D rotation $r_2$, its rotation axis can be expressed in a basis spanned by $3$ axes $\tilde{v}_2, \tilde{v}_2$, and $v_3 \times \tilde{v}_2$ (respectively $\tilde{v}_2, \tilde{v}_2$, and $v_3 \times \tilde{v}_2$ for a dihedral group $D_{2n}$), such that the associated rotation quaternions $\tilde{Q}_2$ is written as

$$\tilde{Q}_2 \equiv \{s_2, q_2 \} = \{s_2, q_2v_3 + b_2v_2 + c_2v_3 \times v_2\}$$

for cubic groups and

$$\tilde{Q}_2 \equiv \{s_2, q_2 \} = \{s_2, q_2v_3 + b_2v_2 + c_2v_3 \times v_2\}$$

for dihedral groups. The rotation quaternions $\tilde{Q}_2, \tilde{Q}_2$ obtained by applying
Fig. 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 symmetric assembly. The directed edges are the generator rotations applied to the group elements. The blue edges are the $r_1$ 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.

$r_2$ after $r_1$ has the following coefficients,

\[ s = -(a_0 + b_y), \]
\[ a = -(c_y), \]
\[ b = s_y - (a_0), \]
\[ c = -(a_0). \]

Similarly, the rotation quaternion $\hat{Q}_r \hat{Q}_g$ obtained by applying $r_1$ after $r_2$ has the following coefficients,

\[ s = -\frac{s_y}{2} + \frac{\sqrt{3}}{2}(a_0 + ab_y), \]
\[ a = -\frac{b_y}{2} + \frac{\sqrt{3}}{2}(a_0 + ac_y), \]
\[ b = -\frac{s_y}{2} + \frac{\sqrt{3}}{2}b_y, \]
\[ c = -\frac{a_0}{2} + \frac{\sqrt{3}}{2}b_y. \]

Finally, for a dihedral group $D_8$, the rotation quaternion $\hat{Q}_r \hat{Q}_g$ has the following coefficients,

\[
\begin{bmatrix}
I_r = -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\
-\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\
-\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\
-\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2}
\end{bmatrix}
\]

\[
s = \cos\left(\frac{\pi}{2}\right)s_y + \sin\left(\frac{\pi}{2}\right)a_y, \]
\[a = \cos\left(\frac{\pi}{2}\right)a_y + \sin\left(\frac{\pi}{2}\right)b_y, \]
\[b = \cos\left(\frac{\pi}{2}\right)b_y + \sin\left(\frac{\pi}{2}\right)c_y, \]
\[c = \cos\left(\frac{\pi}{2}\right)c_y + \sin\left(\frac{\pi}{2}\right)d_y. \]

2.10. Optimization of the rotations

Here we will use the same notations as in our previous work on cyclic symmetries (Pagès et al., 2018). 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 Eq. (4). For each element $g$ of the chosen symmetry group, the contribution to the loss function is the RMSD between $r_g(A)$ and $A = [a_{qg}].$ According to the RMSD master Eq. (3) from Pagès et al., 2018 with $B = A$, we can say that $A$ and $B$ have the same COM, so the translational part of RMSD becomes null and we obtain

\[
\text{RMSD}^2(r_g(A), A_g) = \frac{4}{N}q^T I q + 4q^Tx_{sL} + x_{g}. \]

where

\[
\begin{aligned}
I_g &= \begin{bmatrix}
\sum (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)}) & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 \\
-\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & \sum (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)}) & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 \\
-\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & \sum (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)}) & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 \\
-\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & -\frac{1}{2} (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})/2 & \sum (x_{ij}x_{g(ij)} + z_{ij}z_{g(ij)})
\end{bmatrix},
\end{aligned}
\]

and

\[
\begin{aligned}
x_{sL} &= \frac{1}{2} \sum a_{ij} \times a_{ij} \times a_{ij} \times a_{ij} \\
x_{g} &= \frac{1}{2} \sum a_{ij} \times a_{ij}. \end{aligned}
\]

Our aim will be to minimize the sum of squared RMSDs over all elements $g$ of the group $\Gamma$. Let us first assume that we know the value of one of the two axes $v_1$ or $v_2$, for example, $v_2$. In practice, we first

Fig. 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 $\alpha$ between the generator symmetry axes is highlighted in yellow. These two axes are sufficient to describe the entire symmetry group. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

189
compute $v_i$ axis as a cyclic axis using the method from Pagès et al., 2018, then we alternate the computations of $v_i$ and $v_j$ 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 $v_2$ as follows,

$$
\sum_{g \in G} \text{RMSD}^2(g) = v_2^T \left( \sum_{g \in G} b^2 \frac{1}{N_g} I_g + 2 \sum_{g \in G} b^2 c^2 \frac{1}{N_g} I_g \right) v_2 + \sum_{g \in G} c^2 \left( v_2^T I_g v_2 \right)
$$

(17)

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

$$
\min_{v_2} v_2^T A v_2 + b^T v_2 + c
$$

s. t. $v_2^T v_2 = \alpha$.

(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,

$$
A = \sum_{g \in G} b^2 + b^2 \frac{1}{N_g} I_g + 2 \sum_{g \in G} b^2 c^2 \frac{1}{N_g} I_g \right)
$$

$$
b^T = 2 \sum_{g \in G} a_g b_g v_2^T I_g + 2 \sum_{g \in G} a_g c_g v_2^T v_g + 4 \sum_{g \in G} I_g b_g v_2^T v_g
$$

$$
c = \sum_{g \in G} a^2 v_2^T I_g v_2 + 4 \sum_{g \in G} a^2 v_2^T v_g + \sum_{g \in G} x_m^2.
$$

(19)

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

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 choose an orthonormal basis ($v_x$, $v_y$, $v_z$) and rewrite the vector $v_2$ in this basis as

$$
v_2 = \alpha v_x + x v_y + y v_z.
$$

(20)

Then, the optimization problem (18) reduces to

$$
\min_{x, y} x^2 (v_y^T A v_x) + y^2 (v_3^T A v_y) + x y (v_x^T A v_y)
$$

$$
+ x (2 v_x^T A v_y + b^T v_y) + y (2 v_y^T A v_y + b^T v_y) + c
$$

s. t. $x^2 + y^2 = 1 - \alpha^2$.

(21)

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

$$
L(x, y, \lambda) = k x^2 + b x y + m y^2 + 2 p x + 2 q y + \lambda \left( x^2 + y^2 - 1 + \alpha^2 \right).
$$

(22)

with the following coefficients

$$
k = v_y^T A v_x, 
$$

$$
l = v_y^T A v_y,
$$

$$
m = v_y^T A v_y
$$

$$
p = \alpha v_x^T A v_y + \frac{1}{2} b^T v_y,
$$

$$
q = \alpha v_y^T A v_y + \frac{1}{2} b^T v_y.
$$

(23)

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

$$
\begin{align*}
(kx + ly + p + \lambda x &= 0, \\
(lx + my + q + 2 \lambda y &= 0, \\
x^2 + y^2 &= 1 - \alpha^2.
\end{align*}
$$

(24)

After eliminating $\lambda$ we obtain

$$
\begin{align*}
1x^2 + (m-k)xy - ly^2 + qx - py &= 0, \\
x^2 + y^2 &= 1 - \alpha^2.
\end{align*}
$$

(25)

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

$$
x = 2\sqrt{1-\alpha^2} - \frac{1}{1 + t^2},
$$

$$
y = \frac{t}{1 + t^2}.
$$

(26)

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

$$
- l(1-\alpha^2) + pt(1-\alpha^2) t^2
$$

$$
+ 2(1-\alpha^2) (k-m) + t(1-\alpha^2) t
$$

$$
+ 6(1-\alpha^2) t
$$

$$
+ 2(1-\alpha^2)(k-m) + \sqrt{1-\alpha^2} t
$$

$$
- (1-\alpha^2) t - p = 0.
$$

(27)

This is our final fourth-order algebraic equation, whose roots can be found analytically (Neumark, 2014). 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

Fig. 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 assembly $(1stm)$ with the RMSD loss of 0.0 Å, D – A dihedral $D_3$ 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).
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, Fig. 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 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, Fig. 8 shows the 1ocw structure, which is a perfect Cs assembly with a stoichiometry of A4B4 and the RMSD loss of 0 Å. Our algorithm also detects a D4 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. Fig. 9 shows two of such examples. One is the 4itv protein labelled as C2 (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 C5 (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 (Pagès et al., 2018). More precisely, we compared it to the results published by David Avnir and colleagues (Dryzun et al., 2011; Pinsky et al., 2013). We will refer to it as to CSM (Continuous Symmetry Measure). We also compared our method to the one from Emmanuel Levy (Levy et al., 2006), and will refer to it as to Levy. Please refer to the first part of our paper (Pagès et al., 2018) for more details.

![Fig. 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.](image)

![Fig. 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 – The axes are computed for the full assembly, with the RMSD loss of 2.83 Å.](image)

![Fig. 9. A – The 4itv protein classified in PDB as C2 (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 C5 (RMSD loss of 0.68 Å), also has an icosahedral symmetry with the RMSD loss = 0.56 Å.](image)

### Table 1

<table>
<thead>
<tr>
<th>PDB Code</th>
<th>Group</th>
<th>RMSD (AnAnaS)</th>
<th>RMSD (CSM)</th>
<th>RMSD (Levy)</th>
<th>AnAnaS Time(^a)</th>
<th>CSM Time(^b)</th>
<th>Levy Time(^c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1mos(^d)</td>
<td>D1</td>
<td>1.36 Å</td>
<td>–</td>
<td>1.39 Å</td>
<td>0.13 s</td>
<td>3.7 s</td>
<td>0.49 s</td>
</tr>
<tr>
<td>2hbb</td>
<td>D2</td>
<td>1.64 Å</td>
<td>2.43 Å</td>
<td>1.64 Å</td>
<td>0.05 s</td>
<td>12.2 s</td>
<td>0.28 s</td>
</tr>
<tr>
<td>3kwc</td>
<td>D1</td>
<td>0.81 Å</td>
<td>–</td>
<td>0.89 Å</td>
<td>0.63 s</td>
<td>3950 s</td>
<td>2.3 s</td>
</tr>
<tr>
<td>2pzw</td>
<td>D1</td>
<td>0.34 Å</td>
<td>0.39 Å</td>
<td>0.47 Å</td>
<td>0.23 s</td>
<td>–</td>
<td>1.8 s</td>
</tr>
<tr>
<td>1ogi</td>
<td>D1</td>
<td>0.35 Å</td>
<td>0.50 Å</td>
<td>0.47 Å</td>
<td>0.14 s</td>
<td>–</td>
<td>1.5 s</td>
</tr>
<tr>
<td>1fs2</td>
<td>D2</td>
<td>0.19 Å</td>
<td>0.15 Å</td>
<td>0.54 Å</td>
<td>1.21 s</td>
<td>–</td>
<td>16.6 s</td>
</tr>
<tr>
<td>5x47</td>
<td>T</td>
<td>0.85 Å</td>
<td>–</td>
<td>1.02 Å</td>
<td>0.32 s</td>
<td>–</td>
<td>5.62 s</td>
</tr>
<tr>
<td>4p18</td>
<td>O</td>
<td>0.19 Å</td>
<td>–</td>
<td>2.13 Å</td>
<td>3.1 s</td>
<td>–</td>
<td>131 s</td>
</tr>
<tr>
<td>4ozr(^e)</td>
<td>I</td>
<td>1.05 Å</td>
<td>–</td>
<td>2.38 Å</td>
<td>18.8 s</td>
<td>–</td>
<td>1118 s</td>
</tr>
</tbody>
</table>

\(^a\) 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 Dryzun et al., 2011 and Pinsky et al., 2013 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.

For the comparison, we have selected all dihedral assemblies presented in the original CSM publications (Dryzun et al., 2011; Pinsky et al., 2013). 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 4ozr 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 (Pagès et al., 2018), it clearly shows that our method scales with the size of the input assembly much better than the other two 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...
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 by Dryzun et al., 2011; Pinsky et al., 2013). 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 symmetric 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. \(C_t\) is a subgroup of \(D_t\)). 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. \(C_3\) and \(D_7\)). 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 \(C_2\) are 4aod and 4aoe, for which the biological assemblies are indeed \(C_2\). The 11 other cases have uncertainties between \(C_2\) and \(C_5\) 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 \(C_5\) and \(C_2\) axes are detected, and is actually a part of a \(D_7\) assembly. The other is 3qqg, which is annotated as \(C_3\) 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 asymmetric.
- Some structures have missing or additional chains that are not supported by our program. For example, 2zl2 has a \(D_7\) symmetry but 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 assemblies, we consider them symmetric only if the corresponding RMSD measure is smaller than half of the radius of gyration of the assembly.
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 with graphical user interface for the SAMSON software platform. We can also provide the source code upon request.

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 h 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 $D_4$ 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 (Pagès et al., 2018) 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 Pagès et al., 2018 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 our 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/.

Funding

This work has been supported by L’Agence Nationale de la Recherche (Grant No. ANR-15-CE11-0029-03).

Acknowledgements

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/

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 $C_1$ in the PDB for which we found a $C_1$ symmetry (i.e. no symmetry).

<table>
<thead>
<tr>
<th>PDB</th>
<th>AnAnaS</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
<th>$C_5$</th>
<th>$C_6$</th>
<th>$C_7$</th>
<th>$C_8$</th>
<th>$D_2$</th>
<th>$D_3$</th>
<th>$D_4$</th>
<th>$D_5$</th>
<th>$D_6$</th>
<th>$D_7$</th>
<th>$D_8$</th>
<th>$T$</th>
<th>$O$</th>
<th>$I$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_2$</td>
<td>54</td>
<td>33091</td>
<td>8</td>
<td>23</td>
<td>6</td>
<td>470</td>
<td>15</td>
<td>7</td>
<td>1</td>
<td>205</td>
<td>2</td>
<td>33883</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_3$</td>
<td>2</td>
<td>3</td>
<td>4188</td>
<td>16</td>
<td>60</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4269</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_4$</td>
<td>1</td>
<td>2</td>
<td>1046</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>1060</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_5$</td>
<td>6</td>
<td>561</td>
<td>1</td>
<td>1</td>
<td>568</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_6$</td>
<td>2</td>
<td>2</td>
<td>411</td>
<td>1</td>
<td>416</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_7$</td>
<td>6</td>
<td>104</td>
<td>6</td>
<td>110</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C_8$</td>
<td>8</td>
<td>34</td>
<td>37</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_2$</td>
<td>3</td>
<td>26</td>
<td>6571</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>6609</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_3$</td>
<td>1</td>
<td>5</td>
<td>1939</td>
<td>1</td>
<td>2</td>
<td>1952</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_4$</td>
<td>1</td>
<td>1</td>
<td>654</td>
<td>5</td>
<td>661</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_5$</td>
<td>1</td>
<td>1</td>
<td>236</td>
<td>5</td>
<td>237</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_6$</td>
<td>1</td>
<td>1</td>
<td>106</td>
<td>99</td>
<td>106</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_7$</td>
<td>1</td>
<td>1</td>
<td>34</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D_8$</td>
<td>2</td>
<td>359</td>
<td>3</td>
<td>364</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T$</td>
<td>6</td>
<td>329</td>
<td>329</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$O$</td>
<td>2</td>
<td>617</td>
<td>625</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The authors also thank Elvira Kinzina and Andrei Kazennov from MIPT Moscow for their support at the initial stage of the project.

References


