doi.org/10.26434/chemrxiv.11294480.v1

ChemEnv : A Fast and Robust Coordination Environment Identification Tool

David Waroquiers, Janine George, Matthew Horton, Stephan Schenk, Kristin Persson, Gian-Marco Rignanese, Xavier Gonze, Geoffroy Hautier

Submitted date: 28/11/2019 • Posted date: 17/12/2019

Citation information: Waroquiers, David; George, Janine; Horton, Matthew; Schenk, Stephan; Persson, Kristin; Rignanese, Gian-Marco; et al. (2019): ChemEnv : A Fast and Robust Coordination Environment Identification Tool. ChemRxiv. Preprint. https://doi.org/10.26434/chemrxiv.11294480.v1

Coordination or local environments have been used to describe, analyze, and understand crystal structures for more than a century. Here, we present a new tool called ChemEnv, which can identify coordination environments in a fast and robust manner. In contrast to previous tools, the assessment of the coordination environments is not biased by small distortions of the crystal structure. Its robust and fast implementation enables the analysis of large databases of structures. The code is available open source within the pymatgen package and the software can as well be used through a web app available on http://crystaltoolkit.org through the Materials Project.

File list (2)

ChemEnv : A fast and robust coordination environment identification tool

David Waroquiers,[a] Janine George,[a] Matthew Horton,[b,c] Stephan Schenk,[d] Kristin A. Persson,[b,c] Gian-Marco Rignanese,[a] Xavier Gonze[a,e] and Geoffroy Hautier [a]*

[a]*Institute of Condensed Matter and Nanosciences, Université catholique de Louvain, Chemin des Étoiles 8, 1348 Louvain-la-Neuve, Belgium,* [b]*Energy Technologies Area, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA,* [c]*Department of Materials Science and Engineering, University of California, Berkeley, CA 94720, USA,* [d]*BASF SE, Digitalization of R&D, Carl-Bosch-Str. 38, 67056 Ludwigshafen, Germany, and* [e]*Skolkovo Institute of Science and Technology, Skolkovo Innovation Center, Nobel St. 3, Moscow, 143026, Russia. E-mail: geoffroy.hautier@uclouvain.be*

Abstract

Coordination or local environments have been used to describe, analyze, and understand crystal structures for more than a century. Here, we present a new tool called *ChemEnv*, which can identify coordination environments in a fast and robust manner. In contrast to previous tools, the assessment of the coordination environments is not biased by small distortions of the crystal structure. Its robust and fast implementation enables the analysis of large databases of structures. The code is available open source within the *pymatgen* package and the software can as well be used through a web app available on http://crystaltoolkit.org through the Materials Project.

1. Introduction

Inorganic crystal structures are typically described by their structure prototype or by a more local concept of "coordination environment".(Müller, 2007; Allmann & Hinek, 2007) Coordination environments or local environments (e.g., octahedral, tetrahedral, etc...) are often used in structure visualisation as they clarify the crystal arrangement. These environments can also be used to understand crystal structures and their properties. P. Pfeiffer was the first to transfer this concept of coordination environments from coordination complexes to crystals to rationalize crystals as large molecules.(Pfeiffer, 1915; Pfeiffer, 1916) Very often these coordination environments are determined in a non-automatic manner by the individual researcher. Local environments play a major role in solid state chemistry and physics as well as materials science. For instance, the famous Pauling rules, that have been used to understand and rationalize crystal structures for 90 years, rely heavily on this concept.(Pauling, 1929) In the Pauling rules, the analysis of the coordination environments is used to determine the stability of a material. Electronic, optical, magnetic, and other properties of crystals have also been related to and explained by local environments.(Hoffmann, 1987; Hoffmann, 1988; Lueken, 2013; Peng *et al.*, 2015) In recent years, coordination environments have been discussed and used as structural descriptors to derive structure-property relationships via machine-learning methods.(Jain *et al.*, 2016; Zimmermann *et al.*, 2017) Some of us have analyzed the coordination environments present in oxides in a statistical manner.(Waroquiers *et al.*, 2017) Such large-scale analyses require an easily reproducible, robust and automatic determination of coordination environments. Since the transfer of the concept of coordination environments from coordination complexes to crystals, various approaches to determine coordination numbers, coordination environments, or the distortion of coordination environments have been developed.(Brunner & Schwarzenbach, 1971; Hoppe, 1979; Pinsky &

Avnir, 1998; Stoiber & Niewa, 2019) However, the methods mentioned so far are very sensitive to small structural distortions and not well suited for a robust and automatic assessment of coordination environments in very large databases consisting of several thousands of crystal structures such as the ICSD,(Bergerhoff & Brown, 1987) Pearson database,(Villars & Cenzual, 2018) or CCDC.(Groom & Allen, 2014) To fill this gap, we developed *ChemEnv* – a fast and robust tool to identify coordination environments. It has already been applied in the study of coordination environments of oxides that was mentioned above.(Waroquiers *et al.*, 2017) It is embedded in *pymatgen* – a Python library for materials analysis which is part of the Materials Project that aims at the accelerated design of new materials.(Ong *et al.*, 2013; Jain *et al.*, 2013) Our approach relies on the similarity of such distorted polyhedron present in the crystal structure to ideal reference polyhedra. After a neighbor analysis, we identify potential local environments and compare them through a distance metric to a database of perfect local environments. Different algorithms called strategies are then used to decide on a local environment assignment and the final result can present a unique environment or a mixture of several environments. This approach which is robust to distortion will be described in details in this paper.

2. Method/Algorithm

2.1. Aspects of coordination environments identification

In the process of identifying coordination environments of a given atom, two main questions have to be considered :

1. What are the neighbors of this atom ?

2. What is the overall arrangement of these neighbors around this atom ?

The first question refers to what is called the coordination number while the second

corresponds to the coordination or local environment. The answer to these questions is very clear when the local structure of the atom is close to a perfect environment. However, when relatively large distortions are present, the identification can be much more difficult. In particular, a given local environment can be identified as a *mix* of two or more coordination environments (which can be of the same coordination number or not).

2.2. Voronoï analysis

The neighbors of a given atom in a given structure are determined using a modified Voronoï approach similar to what was proposed by O'Keefe (O'Keefe, 1979). The Voronoï analysis allows for the splitting of the space into regions that are closer to one atom than to any other one. In the standard Voronoï approach for determining the neighbors of a given atom X, all the atoms $\{Y_1, \ldots, Y_n\}$ whose regions are contiguous to the region of atom X are considered as coordinated to atom X. The distances between atom X and each of its neighbors are written $\{d^X_{Y_1}, \ldots d^X_{Y_n}\}$. The common faces $\{f^X_{Y_1}, \ldots f^X_{Y_n}\}$ between the region of atom X and each of the regions of atoms $\{Y_1, \ldots, Y_n\}$ define solid angles $\{\Omega^X_{Y_1}, \ldots \Omega^X_{Y_n}\}$ subtended by these faces at atom X.

The Voronoï regions are easily understood by drawing the perpendicular area bisectors for each pair of atoms X and Y. Figure 1 illustrates the concept in two dimensions (in which area bisectors are thus replaced by line bisectors). The example shown is a slightly distorted square lattice (see Fig. 1 a) where the atoms at the corners (atoms 1, 3, 6 and 8) are displaced towards the central green atom (atom 0). The perfect square lattice is shown by the grey atoms. In Fig. 1 b, the perpendicular line bisectors (in red) are drawn for each segment from the central (green) atom and all other (black) atoms around it. The Voronoï region of the central atom corresponds to the region in light green in Fig. 1 c. Figure 1 d shows the faces $\{f^0_1, \ldots f^0_8\}$ attributed to each pair

of atoms 0-i with $i = 1 \ldots 8$. The solid angle is illustrated for neighbors 1 and 5 by Ω_1^0 and Ω_5^0 respectively.

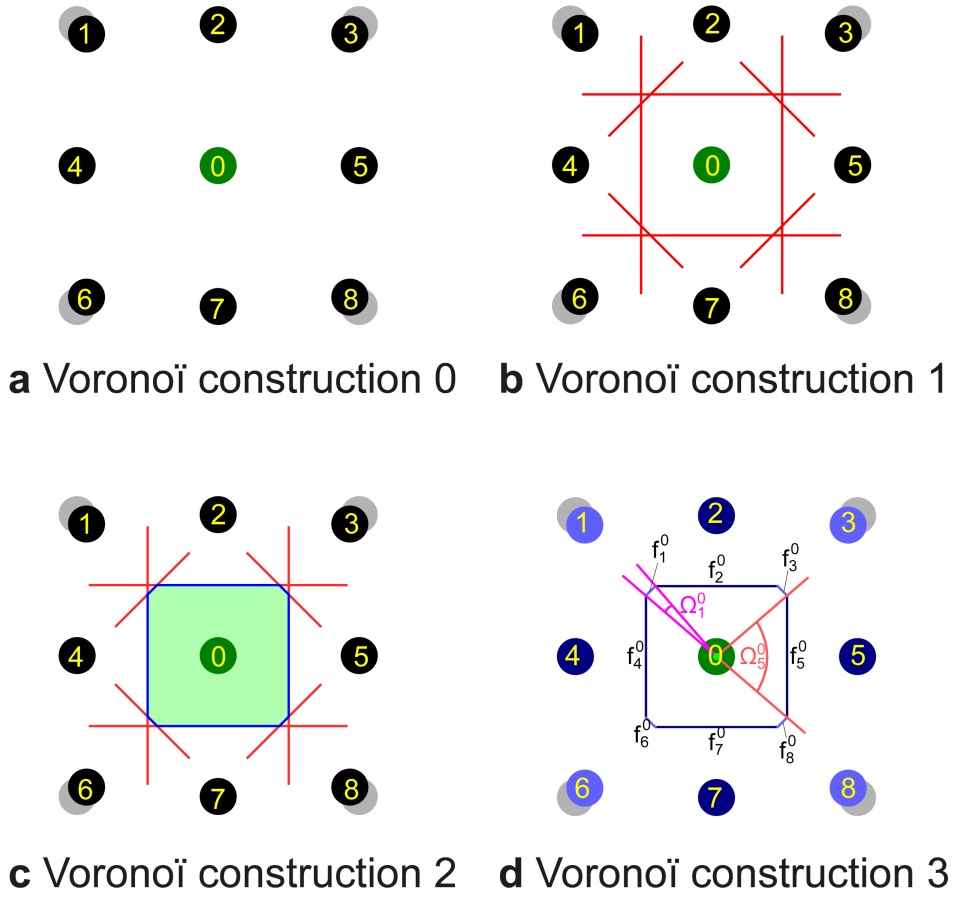


Fig. 1. Voronoï construction (see text).

In our modified approach, two additional cut-offs can be added as shown schematically in two dimensions in Fig. 2 :

1. The first cut-off excludes neighbors on the basis of the distance (Fig. 2 a). Let $d^X_{\min} = \min(\{d^X_{Y_1}, \ldots d^X_{Y_n}\})$ be the distance to the closest neighbor of atom X and $\kappa \geq 1.0$ be the distance cut-off parameter. All atoms lying inside the sphere of radius $\kappa \times d^X_{\min}$ are considered as coordinated neighbors while those lying outside

are disregarded. We define the *normalized distance* $\overline{d^X_{Y_i}}$ of each neighbor Y_i as $\frac{d^X_{Y_i}}{d^X_{\min}}$.

2. The second cut-off is based on the solid angles $\{\Omega^X_{Y_1}, \ldots \Omega^X_{Y_n}\}$ introduced before (Fig. 2 b). Let $\Omega^X_{\max} = \max(\{\Omega^X_{Y_1}, \ldots \Omega^X_{Y_n}\})$ be the biggest solid angle to a neighbor for atom X and $\gamma \leq 1.0$ be the angle cut-off parameter. All neighboring atoms with a solid angle smaller than $\gamma \times \Omega^X_{\max}$ are not considered as coordinated to atom X. We define the *normalized angle* $\overline{\Omega^X_{Y_i}}$ of each neighbor Y_i as $\frac{\Omega^X_{Y_i}}{\Omega^X_{\max}}$.

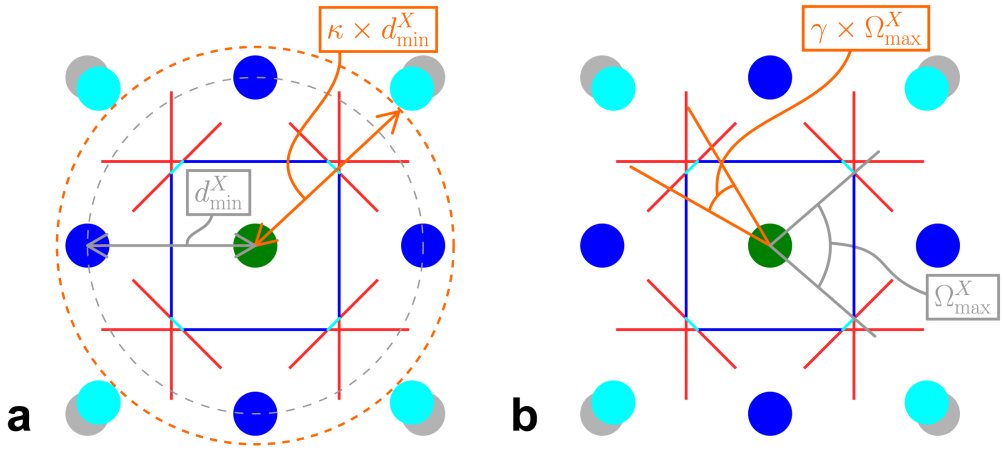


Fig. 2. Schematic representation of the cut-off parameters used in the Voronoï analysis of neighbors : (a) distance cut-off and (b) angle cut-off. (a) Distance cut-off parameter κ. $d^X_{\min}$ is the distance to the closest neighbor (one of the dark blue atoms). Any atom that lies outside the sphere of radius $\kappa \times d^X_{\min}$ (in dashed orange) is not considered as a coordinated neighbor. Atoms at the corner (in light blue) are not considered as neighbors. (b) Angle cut-off parameter γ. $\Omega^X_{\max}$ is the largest solid angle to a neighbor atom. Any atom for which the solid angle is smaller than $\gamma \times \Omega^X_{\max}$ (in orange) is not considered as a coordinated neighbor. Atoms at the corner (in light blue) are not considered as neighbors. (Adapted with permission from D. Waroquiers *et al.*, Chemistry of Materials **29**, 8346, 2017. Copyright (2017) American Chemical Society).

It is possible to use both cut-offs at the same time in which case a given atom is not considered as a coordinated neighbor if either one of the cut-offs disregards it as

a coordinated neighbor.

The modified Voronoï procedure presented above allows for the determination of the coordinated neighbors of a given atom X for a given set of distance/angle parameters. The identification of the coordinated neighbors of atom X defines the *local environment* of this atom. The identification of the model environment which this local environment resembles the most is described in the next section.

2.3. The shape *recognition problem and the continuous symmetry measure*

The *shape* recognition problem consists in the identification of the model environment to which a local and possibly distorted environment resembles the most. Figure 3 illustrates this problem. A distorted octahedron is shown in Fig. 3 a. Whether this distorted octahedron is more similar to a perfect octahedron (see Fig. 3 b) than to any other (model) shape is precisely the purpose of the shape recognition. This inherently implies that a list of model polyhedra to be compared to is known *a priori*. We stick to the list of coordination environments recommended by the IUPAC (Hartshorn *et al.*, 2007) and by the IUCr (Lima-de Faria *et al.*, 1990). This list of environments, their symbol, coordinates and additional meta-information is given as supplementary information.

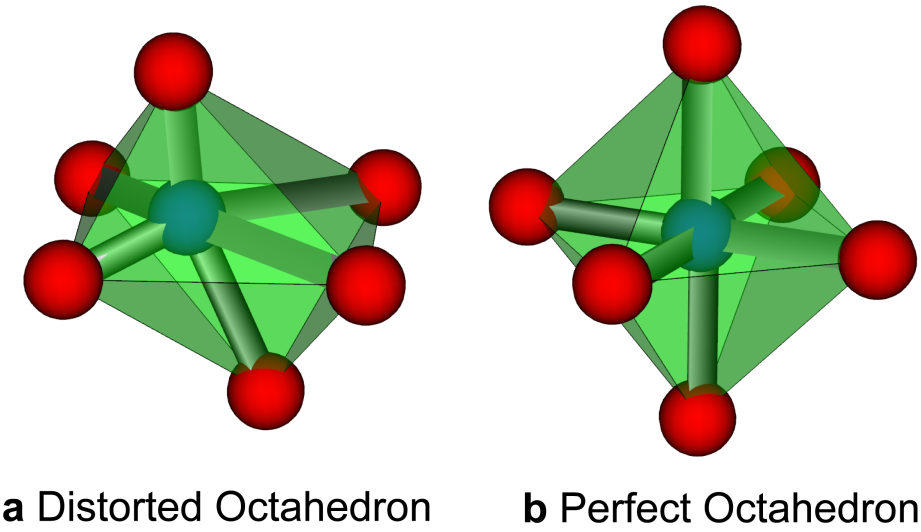


Fig. 3. The *shape* recognition problem. It consists in identifying whether the distorted octahedron in (a) is more similar to the perfect (model) octahedron in (b) than to any other model polyhedron. This presupposes that there exists a list of model polyhedra to be compared to.

In order to *measure* the closeness of a local environment to each perfect model environment, the *Continuous Symmetry Measure* (CSM) is used, as proposed by Pinsky and Avnir (Pinsky & Avnir, 1998). This CSM can be interpreted as a measure of similarity between shapes. For a given structure $\mathcal{Q}$ composed of $N = N_{\mathcal{Q}}$ atoms (vertices) with coordinates $\{\mathbf{q}_k, k = 1, 2, \ldots, N\}$, the CSM $\mathrm{S}_{\mathcal{P}}[\mathcal{Q}]$ with respect to a model polyhedron $\mathcal{P}$ with $N = N_{\mathcal{P}} = N_{\mathcal{Q}}$ vertices $\{\mathbf{p}_k, k = 1, 2, \ldots, N\}$ is defined as :

$$S_{\mathcal{P}}[\mathcal{Q}] = \min \frac{\sum_{k=1}^{N} |\mathbf{q}_k - \mathbf{p}_k|^2}{\sum_{k=1}^{N} |\mathbf{q}_k - \overline{\mathbf{q}}|^2} \times 100 \tag{1}$$

with $\overline{\mathbf{q}} = \frac{1}{N} \sum_{k=1}^{N} \mathbf{q}_k$.

With this definition, the value of the CSM is guaranteed to be in the $[0.0, 100.0]$ interval. A value of 0.0 for the CSM indicates that the two shapes are identical, i.e. the structure $\mathcal{Q}$ corresponds to the perfect structure $\mathcal{P}$. Instead, when the structure

is distorted, the value of the CSM gives a degree of distortion of the structure $\mathcal{Q}$ with respect to the perfect structure $\mathcal{P}$. As such, the CSM can be understood as one definition of a *distance to a shape*.

In Eqn. 1, the minimization has to be performed with respect to four different degrees of freedom :

1. *Translation* (see Fig. 4 a)

   This minimization is easily addressed by translating the local structures to their center of mass.

2. *Ordering* of the atoms (see Fig. 4 b)

   The simplest method is to test all possible permutations of indices. This guarantees a correct value for the CSM but the number of permutations scales as $N!$ making it time-consuming for large ($N > 6$) coordination numbers. The symmetry of the model polyhedra is used to reduce the number of independent permutations for $N \leq 6$. For larger N, a different approach is adopted (see section 2.4).

3. *Orientation* of the structure (see Fig. 5 a)

   The local (distorted) structure is rotated in order to minimize the numerator in Eqn. 1 by using an alignment procedure based on the singular value decomposition(Kabsch, 1976; Kabsch, 1978).

4. *Size* of the structure (see Fig. 5 b)

   A scaling factor is applied to the local structure to avoid size effects : the local structure is normalized to the root-mean square distance from the center of mass of the structure to all vertices.

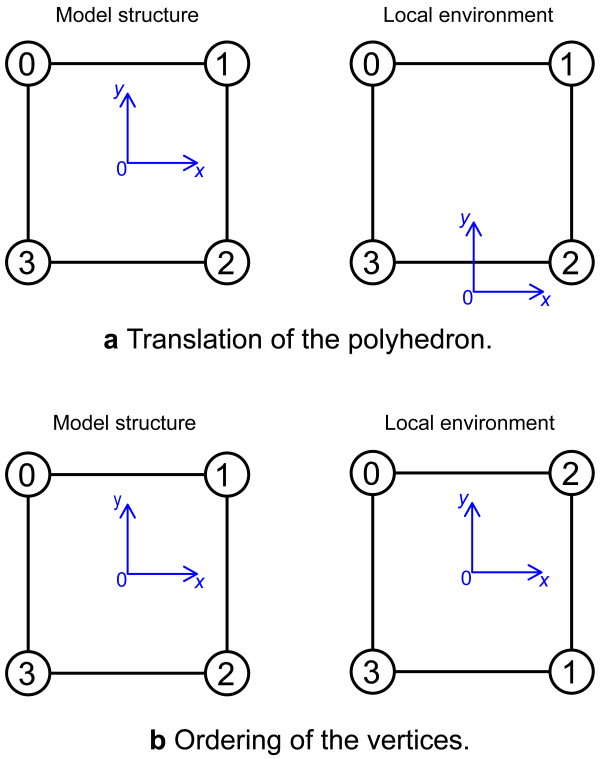


Fig. 4. Translational and ordering degrees of freedom for the minimization in Eqn. 1.

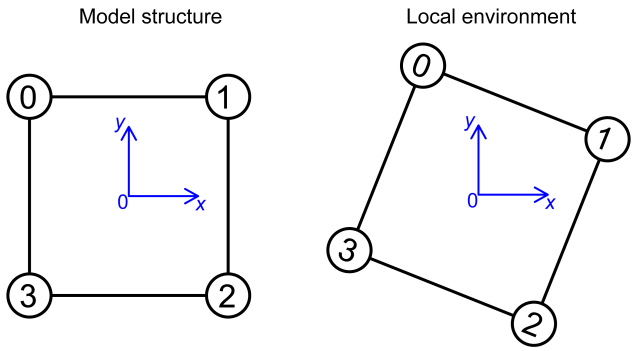


Fig. 5. Rotational and size degrees of freedom for the minimization in Eqn. 1.

The minimization process presented above is equivalent to the point set registration algorithms used in shape or pattern recognition.(François Pomerleau & Siegwart, 2015) The main challenge comes from the fact that the correspondence between points in $\mathcal{Q}$ and $\mathcal{P}$ (i.e. the ordering problem described above) is unknown. In pattern recognition in which the number of points is usually large, algorithms based on pair correlation functions combined with statistical analysis are widely used (see (Maiseli *et al.*, 2017) and references therein). On the contrary, for small number of points, a different approach has to be adopted. As briefly outlined above, the simplest solution (which is used for $N \leq 6$) is to test all possible permutations of indices (ignoring symmetrically identical ones), while for larger N the number of permutations is reduced using the *separation-plane algorithm* (see section 2.4). In any case, for a given permu-

tation of points, the CSM can be obtained thanks to algorithm 1 (see Figure 6, points in $\mathcal{Q}$ have been translated such that their center of mass coincide with that of $\mathcal{P}$). The exact CSM is then the smallest one of all the CSM computed for each permutation considered.

Algorithm 1 Computation of the CSM for a given permutation

procedure CSM($\{\mathbf{q}_k, \mathbf{p}_k\}; k = 1, 2, \ldots, N$)
  $\mathbf{R} \leftarrow$ FINDROTATION($\{\mathbf{q}_k, \mathbf{p}_k\}$) ▷ Find the rotation matrix. This is performed using a least-square-like alignment procedure based on the singular value decomposition. See references (Kabsch, 1976 ; Kabsch, 1978) for a complete description of the algorithm.
  for $k \leftarrow 1, N$ **do**
    $\mathbf{q}_k^* \leftarrow \mathbf{R}\mathbf{q}_k$ ▷ Rotate the points.
  end for
  $A \leftarrow \frac{\sum_{k=1}^{N} \mathbf{q}_k^* \cdot \mathbf{p}_k}{\sum_{k=1}^{N} \mathbf{q}_k^* \cdot \mathbf{q}_k^*}$ ▷ Find the scaling factor. See reference (Pinsky & Avnir, 1998) for a detailed derivation of the terms, as well as a proof of the interchangeability of the rotation and scaling minimizations.
  for $k \leftarrow 1, N$ **do**
    $\overline{\mathbf{q}_k^*} \leftarrow A\mathbf{q}_k^*$ ▷ Scale the points.
  end for
  for $k \leftarrow 1, N$ **do**
    $\mathbf{d}_k \leftarrow \mathbf{p}_k - \overline{\mathbf{q}_k^*}$ ▷ Get the difference between the aligned points and the perfect points.
  end for
  $M \leftarrow 100 \times \frac{\sum_{k=1}^{N} \mathbf{d}_k \cdot \mathbf{d}_k}{\sum_{k=1}^{N} \mathbf{p}_k \cdot \mathbf{p}_k}$ ▷ Compute the CSM.
  return M ▷ Return the CSM.
end procedure

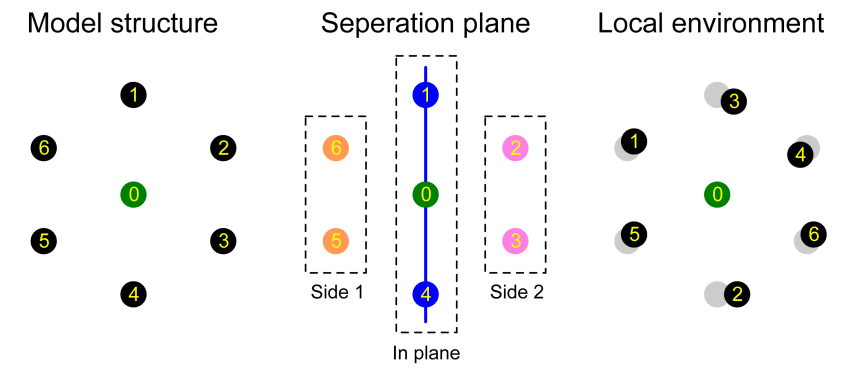
Fig. 6. Algorithm 1. Computation of the CSM for a given permutation

2.4. Separation plane algorithm

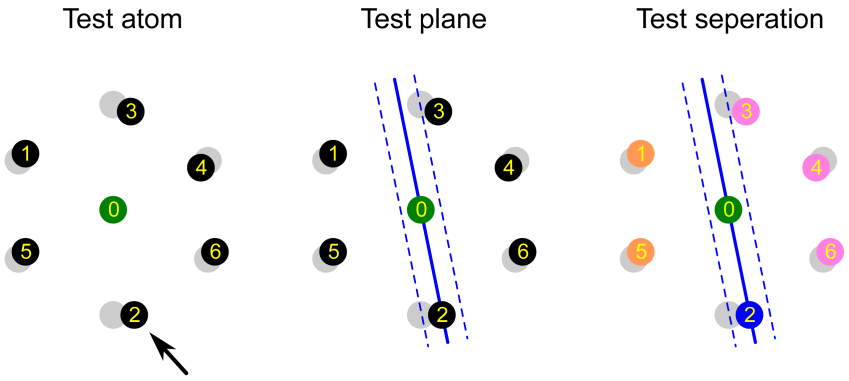
When the number N of coordinated neighbors increases, the number of permutations needed to minimize Eqn. 1 scales as $N!$. When the correspondence of vertices between the local distorted structure and the perfect model polyhedron is not known (which is usually the case for the application of the procedure to large databases of structures), this makes the computation of the CSM almost infeasible for $N > 10$ and very time-consuming for $6 > N \geq 10$ with the standard procedure (e.g. $9! = 362880$,

$12! = 479001600$).

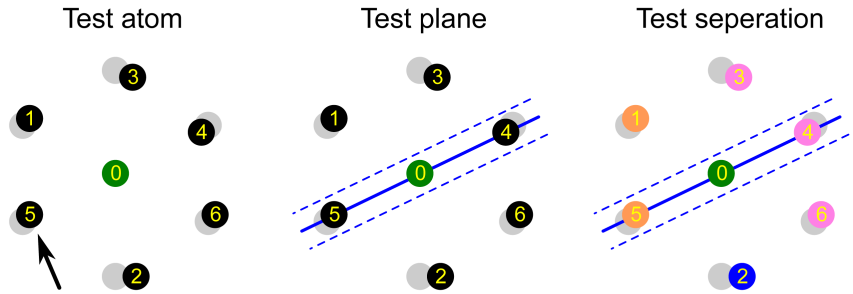
In order to overcome this difficulty, the separation plane algorithm has been devised to drastically reduce the computational time needed. The basic idea is to identify possible planes in the distorted structure that can be assigned to a plane in the model polyhedron in order to reduce the number of permutations needed to find the right correspondence between points and hence the correct CSM. This idea is illustrated in Fig. 7 for a 2-dimensional case. The points of the perfect model shape are separated into three different groups : the set of points supposed to lie within the plane and the two sets of points on either side of the plane. The permutation space is thus reduced because $N!$ is always larger than $N_1!N_2!N_3!$ if at least two of N_1, N_2, N_3 are larger than or equal to 1. For the example in Fig. 7, the number of permutations is reduced from $6! = 720$ to $2! \times 2! \times 2! = 8$. Additionally, for larger environments in which the separating plane contains more than 3 points, these can be ordered using clockwise or counterclockwise ordering, hence reducing the number of permutations even further.



a Model and local (distorted) structure.



b First trial for separation plane algorithm.



c Second trial for separation plane algorithm.

Fig. 7. Illustration of the separation plane algorithm.

A *separation* is defined by its separation plane P_{perf} passing through at least three points of the perfect polyhedron $\mathcal{P}$, and by the two separated groups of points S_{perf} and T_{perf} located on either side of the plane. The set of points in the plane is written

as $P = \{\mathbf{p}_j, j = 1, \ldots N_P\}$ while $S = \{\mathbf{s}_m, m = 1, \ldots N_S\}$ and $T = \{\mathbf{t}_n, n = 1, \ldots N_T\}$ stand for the two sets of points on either side of the plane. By construction, $\{\mathbf{q}_k\} = \{\mathbf{p}_j\} \cup \{\mathbf{s}_m\} \cup \{\mathbf{t}_n\}$ and $N = N_P + N_S + N_T$. We use $\Upsilon_{perf} = (N_S, N_P, N_T)$ as an abridged notation for the separation. For the example illustrated in Fig. 7, the separation is noted $(2, 2, 2)$. An illustration of two separation planes for the cubic and cuboctahedral environments is provided in Fig. 8.

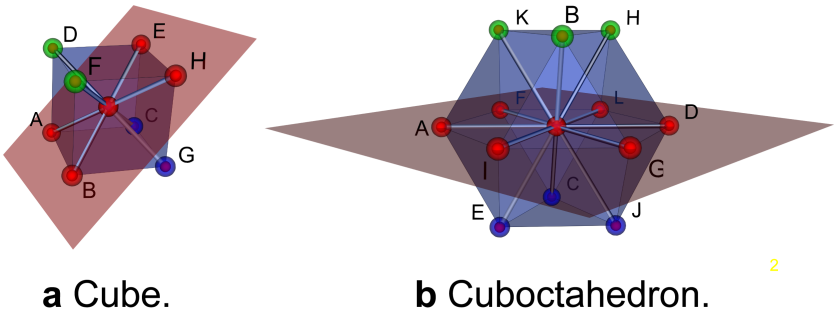


Fig. 8. Examples of separation planes. (a) Separation (2, 4, 2) in the cubic environment: points A, B, H and E (in red) belong to the plane that separates points D and F (in green) from points C and G (in blue). (b) Separation (3, 6, 3) in a cuboctahedron : points A, I, G, D, L and F (in red) belong to the plane that separates points C, E and J (in blue) from points H, K and B (in green).

The procedure for the computation of the CSM of environments with more than six atoms is described in algorithm 2 which is shown in Fig. 9. Separation planes have been defined for all the perfect model environments above six atoms. Usually, more than one separation plane can be defined in a given model polyhedron. In practice, the overall algorithm tests all the available separation planes that have been defined for the polyhedron under consideration. The list of separation planes for each coordination environment is available as SI and is also easily viewable with a script provided in the *ChemEnv* subpackage of *pymatgen*.

Algorithm 2 The separation plane algorithm

▷ For a given perfect environment $\mathcal{P}$, a separation plane P_{perf} is used to reduce the number of permutations. Such a plane has to be defined for each perfect environment, as well as its abridged notation Υ_{perf} (see in the SI and in the pymatgen package for the description of the planes used for each environment). The algorithm searches for similar planes in the local (probably distorted) environment $\mathcal{Q}$.

procedure CSMSeparationPlane($\mathcal{Q}, \mathcal{P}, P_{perf} \| \Upsilon_{perf}$)
  $M_{min} = 100.0$ ▷ Initialize the CSM to its maximum possible value.
  for $\mathbf{q}_a, \mathbf{q}_b \leftarrow$ Combinations($\{\mathbf{q}_k\}, 2$) **do** ▷ Loop on all possible combinations of 2 points.
    $P_{trial} \leftarrow$ SetupPlane($\mathbf{c}, \mathbf{q}_a, \mathbf{q}_b$) ▷ Set up the test plane based on the center $\mathbf{c}$ and the two points $\mathbf{q}_a$ and $\mathbf{q}_b$.
    $\Upsilon_{trial} \leftarrow$ GetSeparation(P_{trial}, δ) ▷ Get the separation for the local environment $\mathcal{Q}$. Points are considered *in* the plane if their distance to the plane is less than δ.
    if $\Upsilon_{trial} \neq \Upsilon_{perf}$ **then**
      continue ▷ Skip separation planes in $\mathcal{Q}$ that do not correspond to the separation plane in $\mathcal{P}$.
    end if
    ▷ Multiple loop on all combinations of the separation :
    for each permutation σ_S of the points in S_{trial} **do**
      for each permutation σ_P of the points in P_{trial} **do**
        for each permutation σ_T of the points in T_{trial} **do**
          $\hat{\sigma} \leftarrow$ concatenation of permutations σ_S, σ_P and σ_T
          $\mathbf{q}_k^{\hat{\sigma}} \leftarrow \hat{\sigma}$-permuted $\mathbf{q}_k$ points
          ▷ Get the CSM for this permutation :
          $M \leftarrow \mathrm{CSM}(\{\mathbf{q}_k^{\hat{\sigma}}, \mathbf{p}_k\})$
          ▷ Update value of the minimum CSM if applicable :
          $M_{min} = min(M_{min}, M)$
        end for
      end for
    end for
  end for
  return M_{min} ▷ Return the CSM.
end procedure

Fig. 9. Algorithm 2. The separation plane algorithm.

The algorithm has been optimized by ordering the points of the separation plane in a clockwise or counterclockwise direction whenever possible. This makes it possible

to reduce the number of permutations related to the separation plane. For example, for the separation (3, 6, 3) of the cuboctahedron shown in Fig. 8 b, the number of permutations of the points in the plane is $6! = 720$. Ordering the points in the perfect and local environments makes it possible to reduce the number of trials to six. A similar optimization is also possible for the two separated groups of points for the separations in which these groups contain a sufficient number of points (e.g. in the icosahedral environment, the separation plane contains four points and splits the other points into two groups of four points each).

2.5. Neighbor sets and distance/angle parameters maps

The distance and angle parameters defined in Sect. 2.2 are very sensitive parameters for the determination of the neighbors of a given atom. Indeed, a very slight change in one of the parameters can change the atoms considered as neighbors and hence the coordination. Each neighbor set of atom A with coordination N is denoted by $\Xi_{\mathrm{N},j}(A)$. The j index comes from the fact that two different neighbor sets can have the same coordination N. A two-dimensional example of such a case is illustrated in Fig. 10 in which two sets of distance and angle cut-off parameters result in two different neighbor sets of the same coordination.

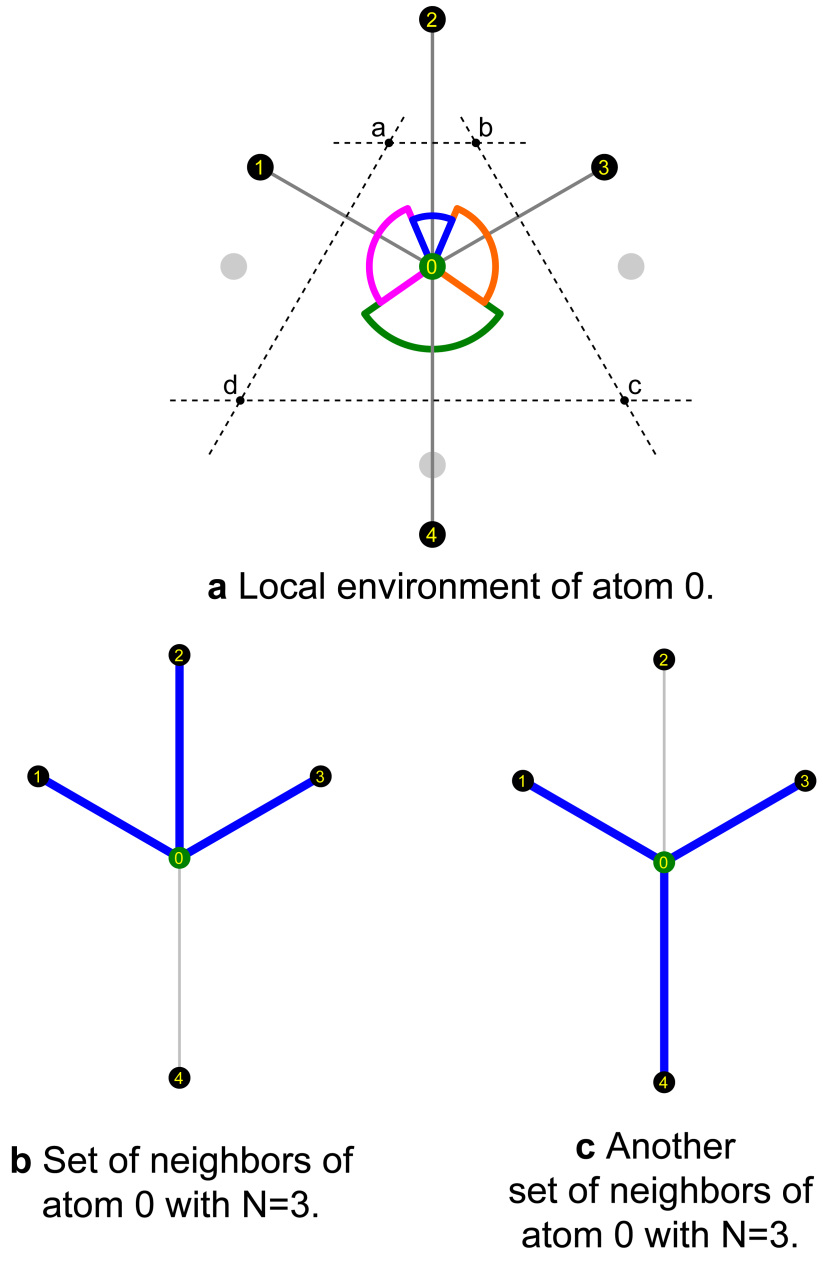


Fig. 10. Illustration in two dimensions of two sets of neighbors having the same coordination number. (a) Local environment of atom 0. Normalized distances to neighbors 1, 2, 3 and 4 are $\overline{d_1^0} = \overline{d_3^0} = 1.0$, $\overline{d_2^0} = 1.15$ and $\overline{d_4^0} = 1.35$. Normalized angles to neighbors 1, 2, 3 and 4 are $\overline{\Omega_4^0} = 1.0$, $\overline{\Omega_1^0} = \overline{\Omega_3^0} \approx 0.924$ and $\overline{\Omega_2^0} \approx 0.42$. (b) Set of neighbors (1, 2 and 3) of atom 0 with N=3. This set of neighbors is obtained with e.g. $\kappa = 1.25$ and $\gamma = 0.3$ cut-offs. (c) Another set of neighbors (1, 3 and 4) of atom 0 with N=3. This set of neighbors is obtained with e.g. $\kappa = 1.4$ and $\gamma = 0.5$ cut-offs.

In order to ensure robustness with respect to the distance and angle cut-off parameters, the identification procedure is performed in two steps. First, all sets of neighbors $\Xi_{\mathrm{N},j}(A)$ are obtained for all possible distance/angle parameters in the Voronoï analysis. For each neighbor set, CSMs are computed with respect to each model polyhedron of the same coordination. This can be represented by a *map* of distance/angle parameters with regions defined for each neighbor set (see Fig. 11 for examples of such maps for Si and O sites in SiO_2 as well as for Cr and Te sites in $Cr_2Te_4O_{11}$). The second step allows one to test the sensitivity of the distance/angle parameters by means of *strategies* (see Sect. 2.6). While for the three first cases in Fig. 11, the "correct" environment is reasonably clear by just looking at the figure (assigning tetrahedral (T:4), angular (A:2) and octahedral (O:6) environments, respectively, to Si in SiO_2, O in SiO_2 and Cr in $Cr_2Te_4O_{11}$), the situation is more complex and the identification is not so evident for Te in $Cr_2Te_4O_{11}$. In this case, the environment could be seen as an *intermediate* between two different environments. The use of *strategies* can clarify such ambiguous cases.

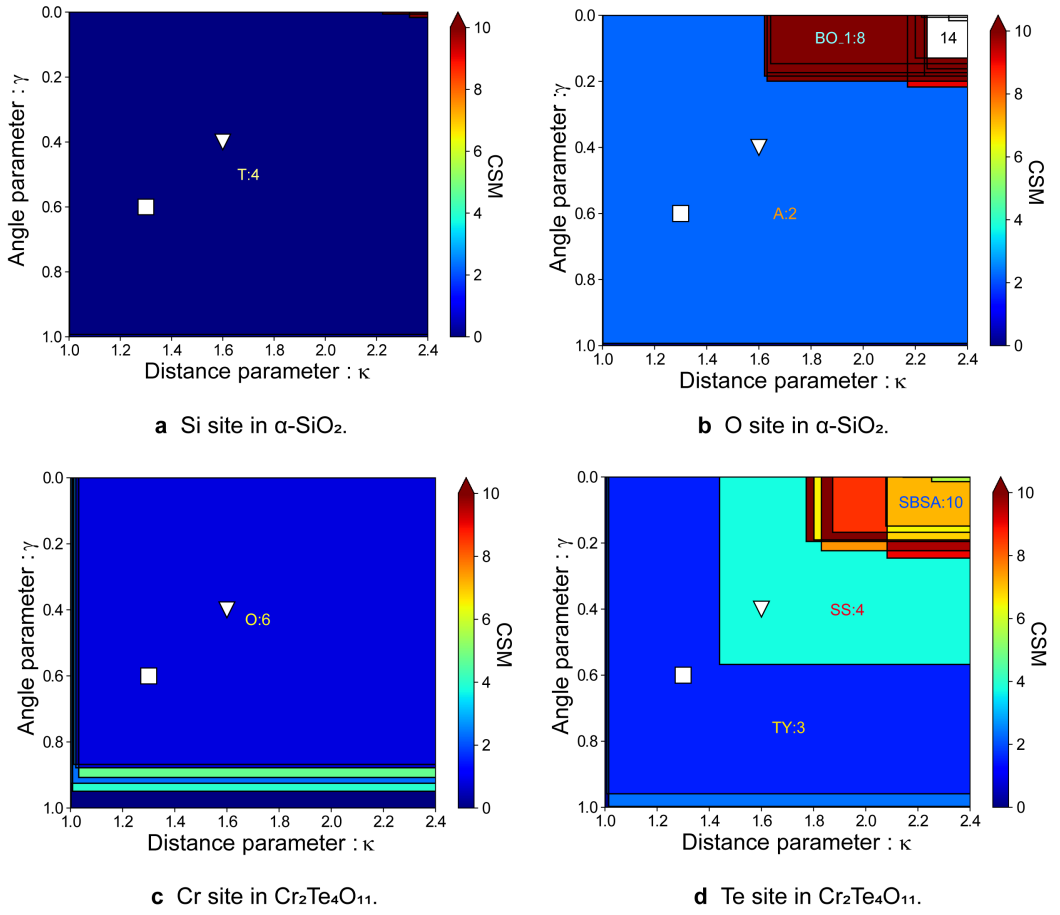


Fig. 11. Examples of distance/angle parameters maps for Si and O in α-SiO_2 (Materials Project id : mp-7000) and Cr and Te in $Cr_2Te_4O_{11}$ (Materials Project id : mp-540537). Each neighbor set corresponds to a region in which any distance/angle parameters combination result in the same set. The color level of each region gives an indication of the CSM value of the model polyhedron to which the corresponding neighbor set resembles the most (i.e. for which the CSM is the lowest). For the larger regions, this model polyhedron is indicated by its symbol. The square and triangle symbols correspond to fixed distance and angle parameters respectively of 1.3/0.6 and 1.6/0.4, showing a clear ambiguity for the Te site in $Cr_2Te_4O_{11}$ (see Sect. 2.6 on how to clarify such cases).

The neighbors in each set, the CSMs for each model polyhedron in each set, and other data related to each neighbor set are stored in a so-called *StructureEnviron-*

ments (see also Sect. 3) or *SE* (hereafter also symbolized by Φ_A for atom A) object. As exemplified in Fig. 11, this SE is not very useful as such as it contains a lot of information that is difficult to interpret directly. In the second step presented below, *strategies* are used to analyze the SE and extract usable and valuable information from the SE.

2.6. Strategies

For the final step of the identification procedure, *strategies* are used to reliably analyze the SE object and extract a useful and usable result. Reliability refers to the robustness of our algorithm in which the sensitivity of the identification to the distance/angle parameters is tested and challenged. Hence, the local environments can be interpreted as one unique environment or as an intermediate between two (or more) coordination environments, each of which being attributed a fraction or percentage. Different strategies can be used depending on the goals, needs and constraints required by the user. This flexibility provided by the strategies is one of the strengths of our identification procedure. For visualisation purposes, a strategy resulting in the identification of a single coordination environment for each site has to be used while reviewing the most commonly observed environments can be done with a strategy allowing for multiple environments for the same site. One can also favour specific or larger/smaller environments depending on the project. In the following, two strategies are developed further.

2.6.1. Fixed distance/angle cut-offs strategy The simplest way to identify the environment is to use fixed distance and angle cut-off parameters. In this *SimplestChemenvStrategy*, the set of neighbors is thus unique and the environment is identified as the one for which the CSM is the lowest. The advantage of such a simple procedure is that it

makes it possible to describe a local environment by its unique corresponding model environment, which is easier to use for visualisation purposes. However, some (distorted or very distorted) local environments can be considered to be an intermediate between two or more model coordination polyhedra. In such cases, this strategy will simply "choose" one environment, depending on the distance and angle parameters. As a simple illustration, Fig. 12 shows the sudden switch from the square-pyramidal environment to the octahedral environment when the distance cut-off is increased. Similarly, for fixed distance and angle cut-offs, when an octahedron is smoothly distorted by moving away one of the atoms, the resulting environment from this simplest strategy changes abruptly from octahedral to square-pyramidal as shown in Fig. 13 (thin lines correspond to the *SimplestChemenvStrategy*). It is thus very sensitive to small changes in the positions of the atoms. Nevertheless, with decent distance and angle parameters (e.g. $\kappa = 1.4$ and $\gamma = 0.3$), the identified environment is reasonably correct in about 85% of the cases.

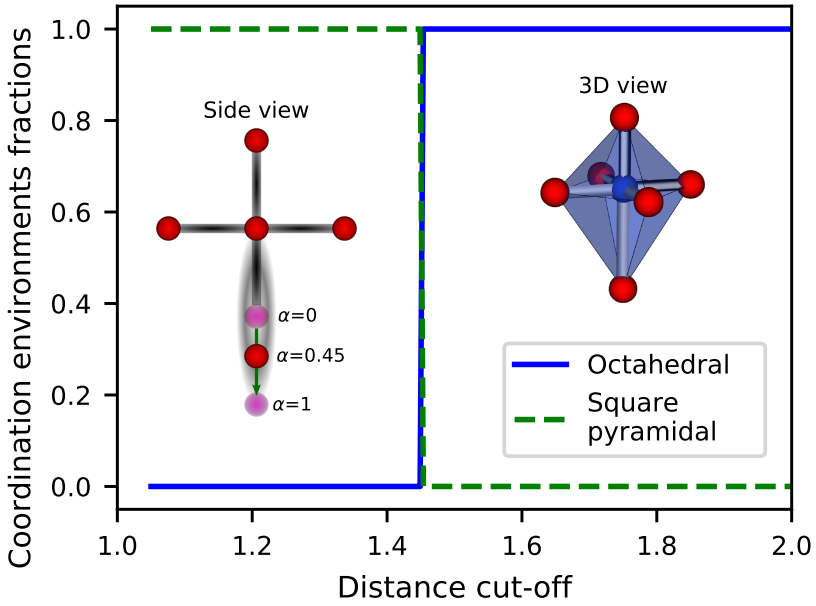


Fig. 12. Coordination environments for a distorted octahedron in which the bottom atom is at distance 1.45 times larger than the other 5 neighbors. When the distance cut-off is lower than 1.45, the bottom atom is not considered as a neighbor and the environment is identified as a square pyramid. When the distance cut-off is larger than 1.45, the bottom atom is taken into account and the environment is identified as an octahedron.

Another illustration of this strategy is shown in Fig. 14 in which a triangular prism is smoothly distorted towards an octahedron by rotating the upper and lower triangular planes in opposite directions (thin lines correspond to the *SimplestChemenvStrategy*). In this case, the number of neighbors remains the same while the actual identified environment switches abruptly from triangular prismatic to octahedral when the CSM of latter becomes smaller than that of the former. Once again, the sensitivity with respect to small changes in the positions of the atoms is critical in this strategy.

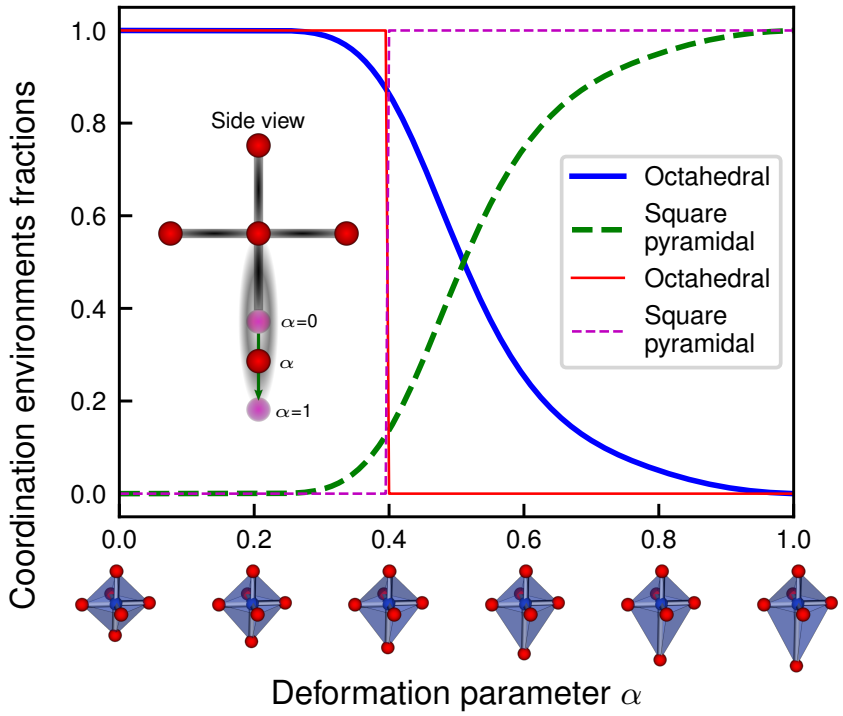


Fig. 13. Smooth distortion from octahedral to square-pyramidal environment by moving away the bottom atom. The deformation parameter $\alpha = 0$ corresponds to the perfect octahedron while for $\alpha = 1$, the bottom atom has been moved to a distance that is twice that of the distance to the other neighbors. The thin lines gives the fractions of octahedral and square-pyramidal environments obtained with the *SimplestChemenvStrategy* (with a distance cut-off of 1.4) while the thick lines correspond to the fractions obtained with the *MultiWeightsChemenvStrategy*. Octahedral and square-pyramidal are respectively shown as solid and dashed lines.

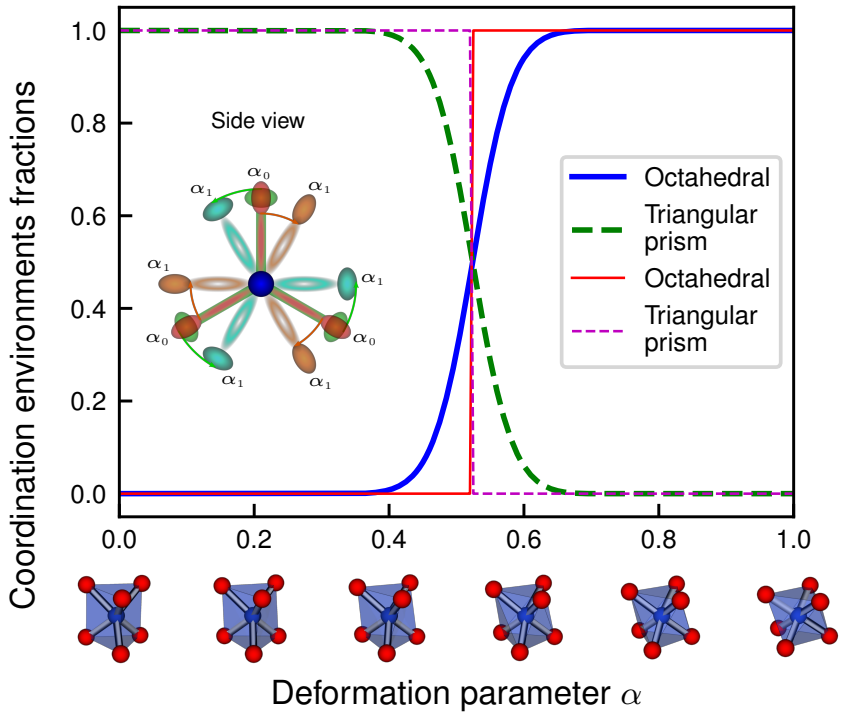


Fig. 14. Smooth distortion from triangular prismatic to octahedral environment by twisting the triangular prism around the principal axis. The deformation parameter $\alpha = 0$ corresponds to the perfect trigonal prism while for $\alpha = 1$, the upper (red $\rightarrow$ orange) and lower (green $\rightarrow$ cyan) triangles have been rotated respectively clockwise and counterclockwise by 30°, corresponding to an octahedron. The thin lines gives the fractions of triangular prismatic and octahedral environments obtained with the *SimplestChemenvStrategy* while the thick lines correspond to the fractions obtained with the *MultiWeightsChemenvStrategy*. Octahedral and triangular prismatic are respectively shown as solid and dashed lines.

2.6.2. Strategy based on multiple weights A second strategy is developed hereafter, in which special care has been taken to remove the artificial abrupt transitions observed with the *SimplestChemenvStrategy*. The idea is to smooth these transitions using a combination of smooth step functions. A given local environment can thus be identified either as one unique coordination environment if distortions are small, or as a *mix* of two or more environments for larger distortions. In practice, the local environment is described as a list of environments, each being assigned a *fraction* or *percentage*.

The percentage or fraction $f_{\mathcal{E}}(A)$ of a given model coordination environment $\mathcal{E}$ depends on the results (CSMs, Voronoï parameters, ...) for each possible set of neighbors contained in Φ_A.

$$f_{\mathcal{E}}(A) = f[\Phi_A](\mathcal{E}) \tag{2}$$

The procedure used to get the fraction of a model polyhedron $\mathcal{E}$ for a given local environment is then obtained as the product of two terms. Suppose $\mathcal{E}$ occurs in a given neighbor set Ξ. The first term results from the relative weight of the neighbor set (as compared to the other neighbor sets) displaying model environment $\mathcal{E}$. The second term comes from the relative weight of the model polyhedron $\mathcal{E}$ within that specific neighbor set.

$$f[\Phi_A](\mathcal{E}) = f^{\text{outer}}[\Phi_A] \times f^{\text{inner}}[\Xi_A^N](\mathcal{E}) \tag{3}$$

In the following, the first term is referred to as the *outer weight* (i.e. the weight that depends on other so-called *outer* neighbor sets) and the second term is referred to as the *inner weight* (i.e. the weight *inside* a specific neighbor set).

Inner weight For a given neighbor set Ξ_A^N of atom A in a given coordination N, the relative weight (and hence fraction) of each model polyhedron is not straightforward. Let Θ^N be the set of K model environments with coordination N :

$$\Theta^N = \{\mathcal{E}_1^N, \ldots, \mathcal{E}_i^N, \ldots, \mathcal{E}_K^N\} \tag{4}$$

For example, the set Θ^6 of six-coordinated model polyhedra (as reported in Refs. (Hartshorn *et al.*, 2007) and (Lima-de Faria *et al.*, 1990) and implemented in the *ChemEnv* package) is composed of the octahedron (symbolized O:6), the trigonal prism (symbolized T:6) and the pentagonal pyramid (symbolized PP:6).

For each model polyhedron $\mathcal{E}_i^N$, the CSM $S_{\mathcal{E}_i^N}[\Xi_A^N]$ with respect to the local environment Ξ_A^N is used to assign a weight to each model polyhedron thanks to the use of an adequately shaped function. Model environments with a lower CSM (i.e. more similar to the local environment) are assigned a larger weight. In particular, if one of the model environments has a CSM of 0.0 (i.e. the local environment is perfect), its weight should be infinite so that it is the only model environment identified. The function should also allow for the assignment of a zero weight to a model polyhedron for which the CSM is larger than a given maximum value $S^{\max}$. One example of such a function is the "modified" inverse function defined in Eqn. 5 and shown in Fig. 15.

$$w_{S^{\max}}(S) = \begin{cases} 1/S^{\max} \times \frac{(S-S^{\max})^2}{S} & \text{if } S \leq S^{\max}, \\ 0.0 & \text{if } S > S^{\max}. \end{cases} \tag{5}$$

in which the numerator $(S - S^{\max})^2$ ensures the continuity at $S = S^{\max}$ while the prefactor $1/S^{\max}$ arises from the normalization of the $[0, S^{\max}]$ to $[0, 1]$.

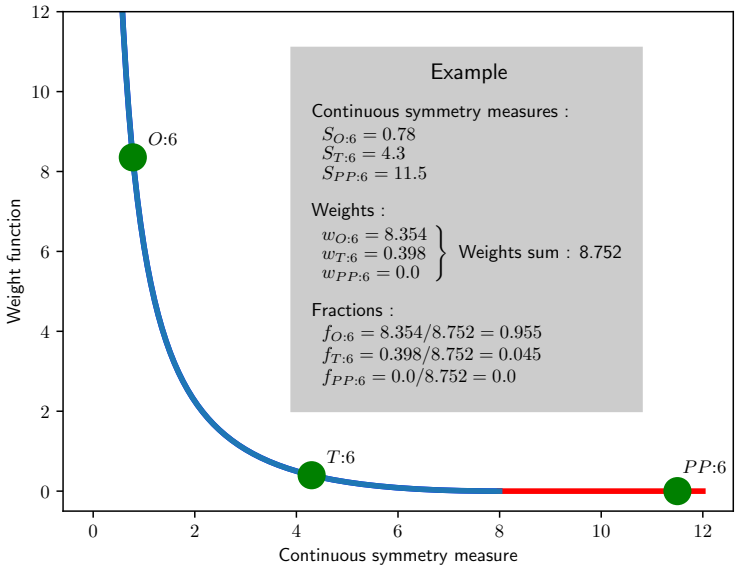


Fig. 15. Weight function for the *inner* weight of model polyhedra. In this example, $S_{\max}$ is set to 8.0, so that the weight of any model polyhedron with a CSM larger than 8.0 is zero.

Fractions of each model environment $\mathcal{E}_i$ are then obtained from these weights:

$$f^{\text{inner}}[\Xi_A^N](\mathcal{E}_i) = \frac{w_{S^{\max}}(S_{\mathcal{E}_i})}{\sum_{j=1}^{j=K} w_{S^{\max}}(S_{\mathcal{E}_j})} \tag{6}$$

A small example is also given in Fig. 15 in which CSMs for a fictitious six-coordinated case are provided.

When the coordination is clearly defined (i.e. when only one neighbor set is identified using the procedure outlined in 2.5), the fractions of each model polyhedron are solely determined by this inner weight. On the other hand, when different neighbor sets are identified, an additional complexity arises from the fact that smaller environments usually tend to be more easily recognized as similar (i.e. having smaller CSMs). The extreme case is the single neighbor which is always assigned a CSM of zero. For cases in which more than one neighbor set is present, the outer weight is used to determine the relative predominance of each of the neighbor sets (and hence their corresponding model polyhedron).

Outer weight The *outer weight* or *neighbor set weight* refers to the weight of a given neighbor set with respect to the other neighbor sets. This outer weight is defined as a product of several "partial weights" (the definition being general enough to allow for flexibility in the choice of the weights) :

$$w^{\text{outer}}[\Psi_A](\Xi_A) = \prod_{i=1}^{i=n_w} \widehat{w^i}[\Psi_A](\Xi_A) \tag{7}$$

in which n_w is the number of partial weights used.

Some of the partial neighbor set weights compare the CSMs of this neighbor set with the ones for the other neighbor sets. The simplest approach is to take the smallest CSM for each of the neighbor sets. In practice, to ensure continuity, an *effective* CSM is defined. The effective CSM of a given neighbor set Ξ_A^N, denoted $S_{\text{eff}}(\Xi_A^N)$, is obtained

from a weighted average using the "modified" inverse function defined in Eqn. 5.

$$S_{\text{eff}}(\Xi_A^N) = \frac{\sum_{\mathcal{E}\in\Theta^N} w_{S^{\max}}(S_\mathcal{E}) \times S_\mathcal{E}}{\sum_{\mathcal{E}\in\Theta^N} w_{S^{\max}}(S_\mathcal{E})} \tag{8}$$

in which $S_\mathcal{E}$ is a short form for $S_\mathcal{E}(\Xi_A^N)$, i.e. the CSM of the neighbor set with respect to the perfect environment $\mathcal{E}$.

Partial weights In the following, the partial weights used in the "default" multi-weights strategy (used in a previous publication (Waroquiers *et al.*, 2017)) are described. The strategy with these default parameters is easily obtained with the following class method (see examples in the tutorials provided in the supplementary material):

`MultiWeightsChemenvStrategy.stats_article_weights_parameters()`

Other weights have also been implemented in the *ChemEnv* package in *pymatgen*.

"Distance-angle area" weight. The idea is to restrict the neighbor sets to those originating from a specific range of values for the distance and angle cutoffs. For example, one might only consider distance cutoffs between 1.2 and 1.8. One might also consider that the Voronoï angle towards a neighbor should always be between 0.3 and 0.8. In practice, a special *area* of distance-angle parameters is defined such as the one shown in Fig. 16. Indeed, there is not much sense to allow for neighbors with a small angle parameter and a small distance parameter or with a large angle parameter and a large distance parameter. If the region of a given neighbor set (as defined in Sect. 2.5) is crossing the above mentioned area, the weight of this neighbor set is 1.0 (indicated in white on Fig. 16), otherwise it is set to 0.0. An extension of this weight could be to ensure it is continuous.

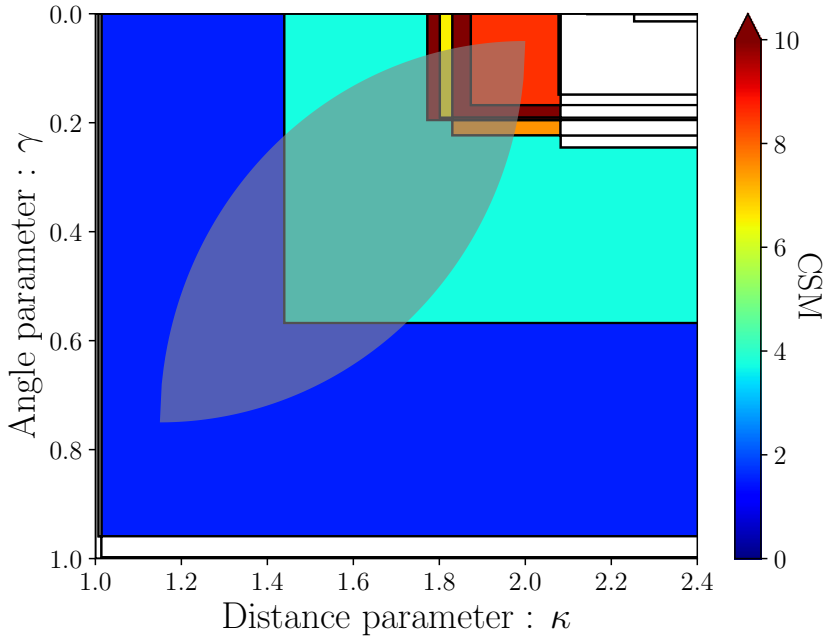


Fig. 16. Schematic of the distance-angle area weight. The shaded area is used to determine which neighbor sets are considered. If the region of a given neighbor set is crossing the shaded area, the set is assigned a "distance-angle area" weight of 1.0. In the opposite case, the set is assigned a weight of 0.0 (white regions).

"Self CSM" weight. This weight makes use of the effective CSM S_{eff} of each neighbor set defined in Eqn. 8. Each neighbor set is assigned a weight depending on the value of this effective S_{eff}. The idea is to disfavour neighbor sets that are *globally* more distorted than others. One example function used to estimate this weight is defined in Eqn. 9 and shown in Fig. 17.

$$w_{S^{\max},\lambda}(S_{\text{eff}}) = \begin{cases} (\overline{S_{\text{eff}}} - 1.0)^2 \times \mathrm{e}^{-\lambda \overline{S_{\text{eff}}}} & \text{if } \overline{S_{\text{eff}}} \leq 1.0, \\ 0.0 & \text{if } \overline{S_{\text{eff}}} > 1.0. \end{cases} \tag{9}$$

where $\overline{S_{\text{eff}}}$ is the normalized effective CSM defined as $\frac{S_{\text{eff}}}{S_{\max}}$.

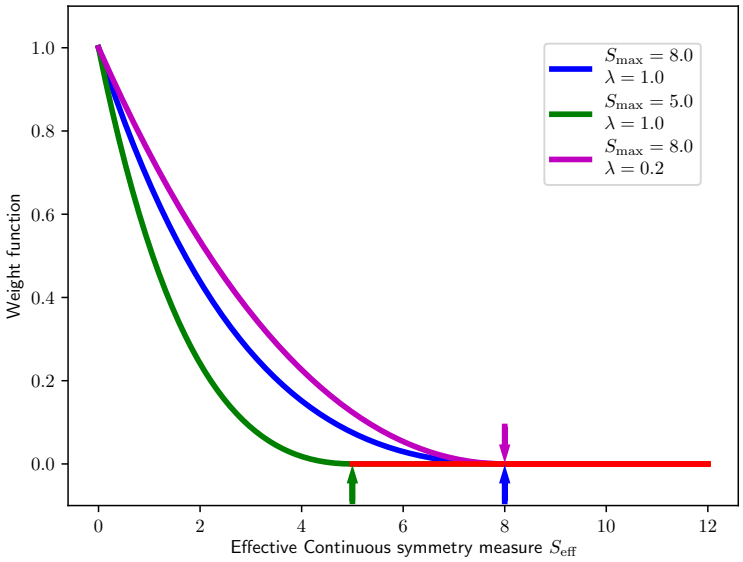


Fig. 17. Weight function for the Self-CSM *outer* weight of neighbor sets as defined in Eqn. 9. The default parameters for this weight are shown as blue while the green and purple curves illustrate other parameters. Arrows indicate thresholds above which values (i.e. $S_{\max}$) of the effective CSM S_{eff} each of the weight functions are set to zero.

"Delta CSM" weight. The goal of this neighbor set weight is to reduce the importance of a given neighbor set Ξ^{N_1} if another neighbor set Ξ^{N_2} of larger coordination number $N_2 > N_1$ is present and not too distorted with respect to the first one. In practice, this weight depends on the difference ΔS_{eff} between the effective CSMs (as defined in Eqn. 8) of the neighbor sets Ξ^{N_2} and Ξ^{N_1} :

$$\Delta S_{\text{eff}}(\Xi^{N_1}, \Xi^{N_2}) = S_{\text{eff}}(\Xi^{N_2}) - S_{\text{eff}}(\Xi^{N_1}) \tag{10}$$

The Delta CSM weight is defined as :

$$w^{delta}_{\chi;\Delta^{min},\Delta^{max}}[\Phi_A](\Xi_A) = \min_{\substack{\Xi_i \in \Phi_A; \\ N(\Xi_i) > N(\Xi_A)}} \chi_{\Delta^{min},\Delta^{max}}\left(\Delta S_{\text{eff}}(\Xi_A, \Xi_i)\right) \tag{11}$$

in which χ is a sigmoid-like function (e.g. a smooth step or smoother step function), $N(\Xi)$ is the coordination of neighbor set Ξ and Δ_{min}, Δ_{max} are the edges used in the χ function.

An example of a χ function is the smoother step function shown in Fig. 18 and defined as :

$$\chi_{a,b}^{smootherstep}(x) = \begin{cases} 0.0 & \text{if } x \leq a, \\ 6\overline{x}^5 - 15\overline{x}^4 + 10\overline{x}^3 & \text{if } a < x < b, \\ 1.0 & \text{if } x \geq b. \end{cases} \tag{12}$$

in which $\overline{x} = (x - a)/(b - a)$ is the scaled value of x mapping the $[a, b]$ interval to the $[0, 1]$ interval.

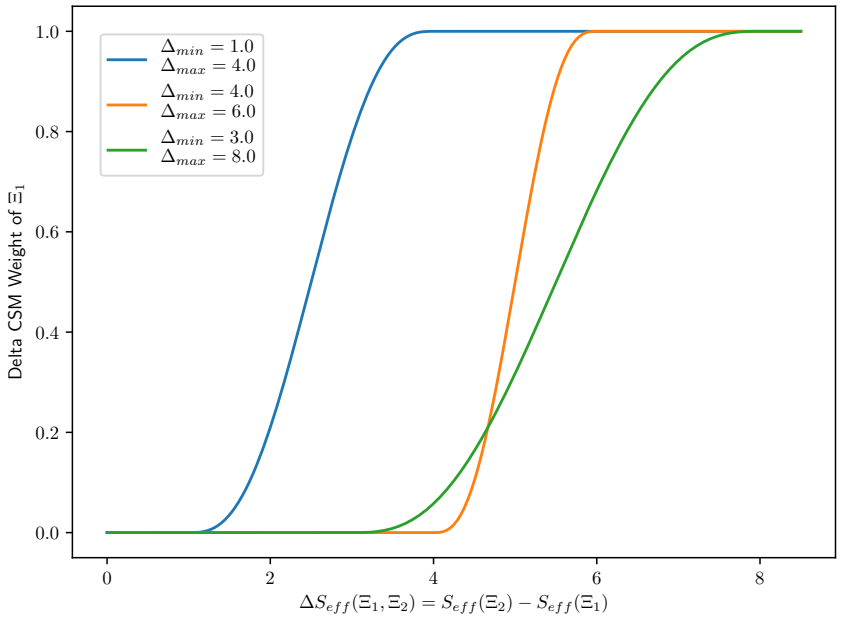


Fig. 18. Smoother step function used in the Delta CSM weight. The "Delta CSM" weight assigned to the Ξ_1 neighbor set is equal to 0.0 if the difference $\Delta S_{\text{eff}}(\Xi_1, \Xi_2)$ between the effective CSM $S_{\text{eff}}(\Xi_2)$ of the Ξ_2 neighbor set and its own effective CSM $S_{\text{eff}}(\Xi_1)$ is lower than Δ_{min}. If the difference $\Delta S_{\text{eff}}(\Xi_1, \Xi_2)$ is larger than Δ_{min}, the Ξ_1 set is assigned a weight of 1.0. The smoother step function is used between these two extremes. The Δ_{min} and Δ_{max} values can be changed if needed and examples of smoother step functions for different values are shown.

Choice of partial weights The default list of outer weights consists of the three above-mentioned partial weights. As an example and in particular to illustrate the need to use both the Self CSM weight and the Delta CSM weight, Fig. 19 shows the fractions of environments obtained for different choices of weights in the case of the smooth distortion from octahedral to square-pyramidal environment (see Fig. 13).

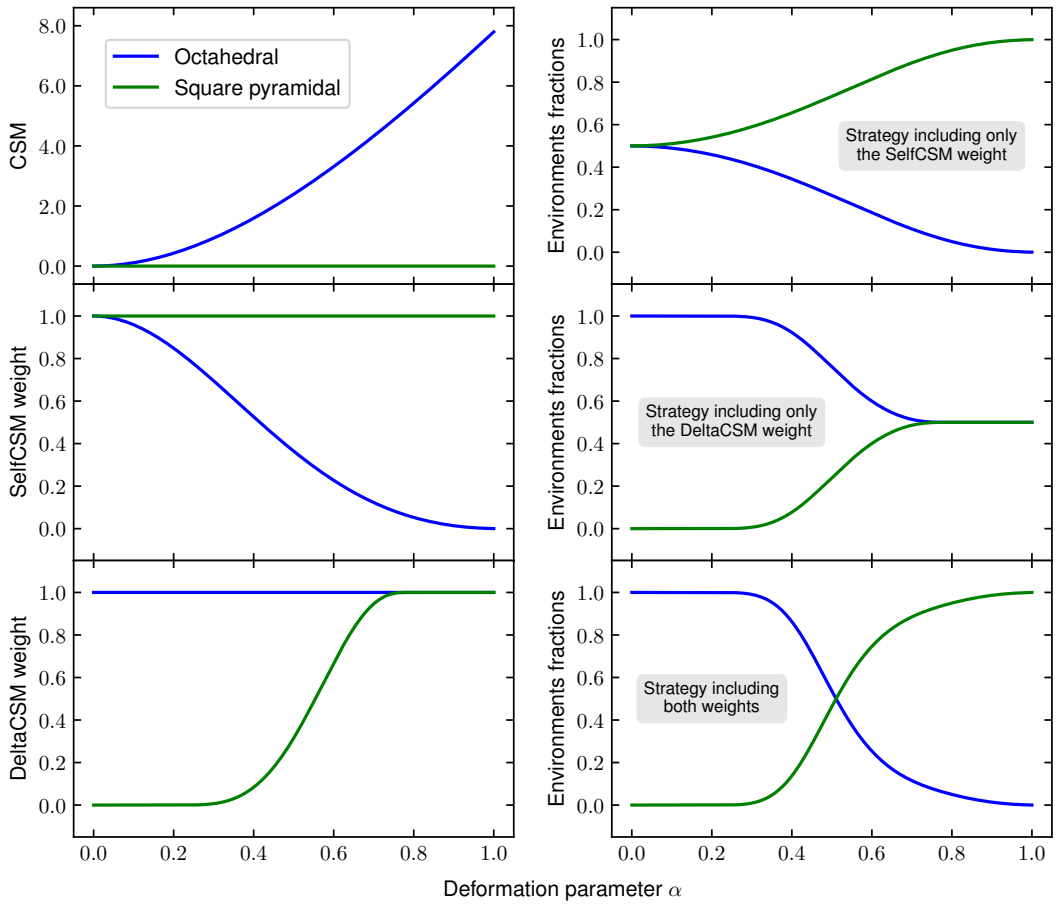


Fig. 19. Choice of partial weights: comparison and combination of Self CSM and Delta CSM weights in the case of the smooth distortion from octahedral to square-pyramidal environment. Curves in blue (green) correspond to the octahedral (square-pyramidal) environment. See text for details.

The upper left panel shows the CSM of the octahedral (increasing with the distortion) and square-pyramidal (always equal to 0.0). The middle left and lower left

panels show the Self CSM and Delta CSM weights for both environments. The Self CSM weight for the square-pyramidal environment is always 1.0 as its CSM is always 0.0. Conversely, the Delta CSM weight for the octahedral environment is always 1.0 as there is no larger neighbor set to be compared to. As shown in the upper right panel, when the sole Self CSM weight is included, the fractions obtained are 50% octahedral and 50% square-pyramidal when no or little distortion is applied (while one would expect to have 100% octahedral and 0% square-pyramidal). Indeed, for both environments, the value of the CSM is 0.0 and hence the Self CSM weight is 1.0. At variance, the middle right panel illustrates the fractions obtained when the sole Delta CSM weight is included. In that case, for large distortions, the fractions obtained are also 50% for each environment. Indeed, when the distortion is large, the Delta CSM weight for the square-pyramidal environment reaches 1.0 as the larger environment is too distorted to disfavour the square-pyramidal environment. The lower right panel illustrates the case when both the Self CSM and Delta CSM weights are included.

3. Description of the package

The *ChemEnv* module is written in Python and can be found in the *pymatgen* package (Ong *et al.*, 2013) as part of the `analysis` submodule. The organisation of the package is shown diagrammatically in Fig. 20. The description of each of the objects referenced as circled numbers in this figure is given hereafter :

① *LocalGeometryFinder*

Main class used to identify the local environments in a structure.

② *AllCoordinationGeometries*

Class containing the list of all the available model coordination geometries (as CoordinationGeometry objects, see ③).

③ *CoordinationGeometry*

Generic class for the description of all the model coordination geometries. An instance of this class is created for each model environment (from the json files stored in the `coordination_geometry_files` directory). It contains information about its perfect coordinates as well as its edges and faces, name(s), symbol(s), technical details for the identification procedure, ...

④ *StructureEnvironments*

Class containing the information (CSMs, neighbors, ...) on all possible neighbor sets for all sites in the structure as introduced in Sect. 2.5. This object is meant to be post-processed with a strategy in order to get relevant and usable data about the local environments of the structure.

⑤ *LightStructureEnvironments*

Class containing the processed data from the *StructureEnvironments* class using one strategy. This object lists the environment(s) and their corresponding fractions (in case of a strategy allowing for mixtures of environments) for each site of the structure.

⑥ *DetailedVoronoiContainer*

Class containing the information on the Voronoï analysis (see Sect. 2.2) performed at the beginning of the identification procedure in order to define the different possible neighbor sets.

⑦ *SimplestChemenvStrategy*

Class used to apply the fixed distance/angle cutoff strategy introduced in Sect. 2.6.1.

⑧ *MultiWeightsChemenvStrategy*

Class used to apply the strategy based on multiple weights as introduced in Sect. 2.6.2.

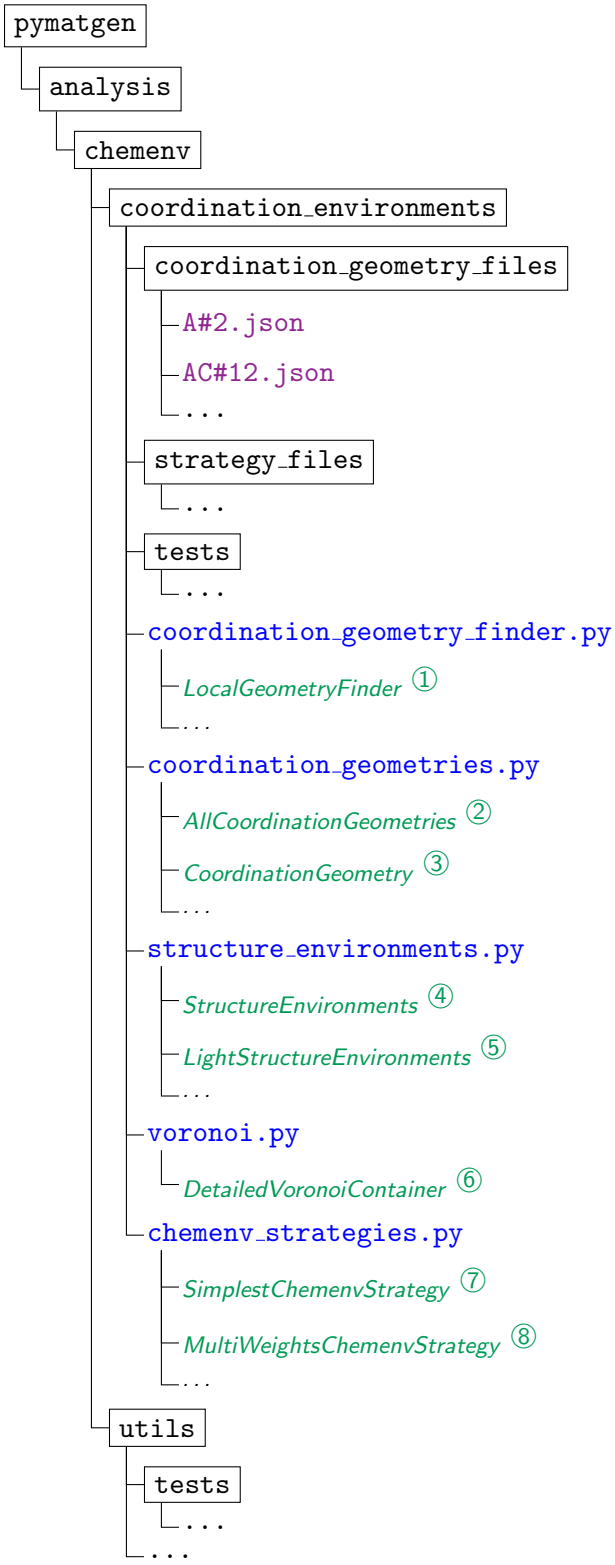


Fig. 20. Organisation of the *ChemEnv* package. Directories are indicated in black and surrounded by a rectangle. Files are indicated in typewriter (blue for python files, purple for other files). The most important python objects are indicated in italic (green). See text for more information.

The most relevant objects needed for the user of *ChemEnv* package are illustrated in Fig. 21.

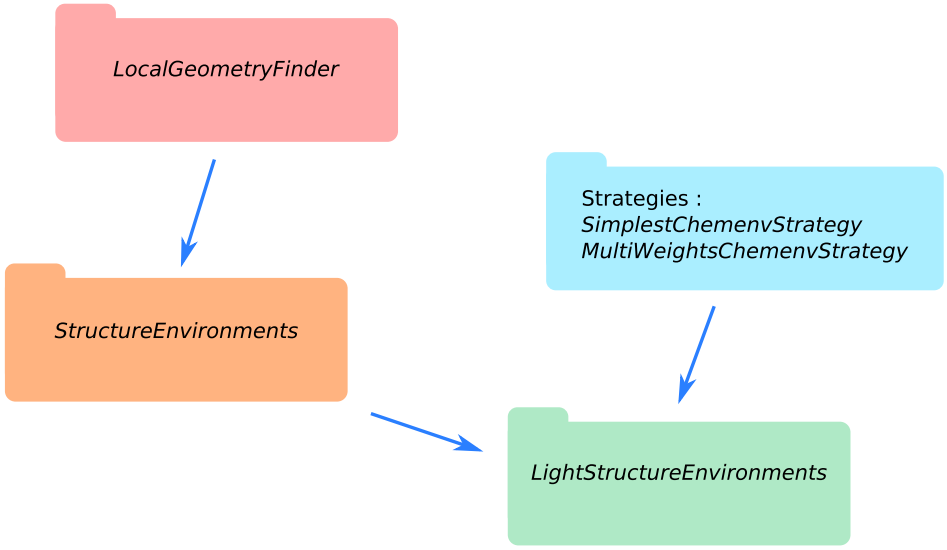


Fig. 21. Main objects of the *ChemEnv* package.

The *LocalGeometryFinder* object is the main class used to initialize and set up the structure as well as to compute the *StructureEnvironments* object (containing the raw coordination environments data as introduced in Sect. 2.5). Combining this *StructureEnvironments* object with a strategy (e.g. *SimplestChemenvStrategy* or *MultiWeightsChemenvStrategy*) leads to the *LightStructureEnvironments* object. This latter object contains the usable information about the environments in a structure, i.e. the environment or *mix* of environments (with their corresponding fractions) that is identified for each site.

4. Interactive Web App

An interactive web app has been developed to improve accessibility of the *ChemEnv* algorithms as part of the Materials Project's Crystal Toolkit platform. While the

Python interface is intuitive and well-documented, not all scientists are Python users, and the web app enables use of *ChemEnv* by any user without installing custom software. The web app supports uploading of any file format supported by the *pymatgen* code, including Crystallographic Information Format (CIF). Alternatively, structures can be loaded directly from the Materials Project database containing more than 100,000 inorganic materials.

The web app is designed to offer one-to-one equivalent functionality to *ChemEnv* by directly calling the corresponding *pymatgen* interface, specifically using the *LightStructureEnvironments* and *SimplestChemenvStrategy*, and allowing the user full interactive control over the distance and angle cut-offs. Each symmetrically distinct chemical environment is shown in 3D using a custom atomic visualizer, along with Wyckoff label, IUPAC symbol, CSM, and human-readable environment label. Oxidation states will be used in the analysis if atoms are appropriately annotated in the uploaded file or, if these are not supplied, oxidation states can be guessed on-the-fly using *pymatgen*'s bond valence analysis algorithms. It will be hosted by the Materials Project, and is available at http://crystaltoolkit.org .

5. Conclusion

We have developed a tool that can analye coordination or local environments of large numbers of crystal structures in a fast and robust manner. The analysis of the neighboring atoms relies on a modified Voronoï approach on a grid of distance and angle cutoffs. On this grid of different distance and angle cutoffs, the coordination environments are determined with the help of a similtarity metric to the shape of ideal polyhedra. Two different strategies are implemented to arrive at the final assignment of the coordination environments. One of these strategies is especially robust against small distortions of the crystal structures making the algorithm particularly useful for

automatic, unsupervised, local environment assignment. This new tool can be used as part of the open-source Python library (*pymatgen*) and within an interactive web app available on http://crystaltoolkit.org through the Materials Project.

6. Supplementary Information

A tutorial for the *ChemEnv* package both in pdf- and jupyter-notebook format is available in the Supplementary Information. The list of all environments as well as some details about the implementation are also available in the Supplementary Information.

J. G. acknowledges funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 837910. Integration with the Materials Project infrastructure was supported by by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, Materials Sciences and Engineering Division under Contract No. DE-AC02-05CH11231 (Materials Project program KC23MP).

References

Allmann, R. & Hinek, R. (2007). *Acta Cryst. A*, **63**(5), 412–417.
URL: *https://doi.org/10.1107/S0108767307038081*

Bergerhoff, G. & Brown, I. D. (1987). *Crystallographic databases*, vol. 360, pp. 77–95.

Brunner, G. O. & Schwarzenbach, D. (1971). *Z. Kristallogr. - Cryst. Mater.* **133**(1-6), 127.
URL: *https://www.degruyter.com/view/j/zkri.1971.133.issue-1-6/zkri.1971.133.16.127/zkri.1971.133.16.127.xml*

Lima-de Faria, J., Hellner, E., Liebau, F., Makovicky, E. & Parthé, E. (1990). *Acta Cryst. A*, **46**(1), 1–11.
URL: *http://dx.doi.org/10.1107/S0108767389008834*

François Pomerleau, F. C. & Siegwart, R. (2015). *A Review of Point Cloud Registration Algorithms for Mobile Robotics.*

Groom, C. R. & Allen, F. H. (2014). *Angew. Chem. Int. Ed.* **53**(3), 662–671.
URL: *http://dx.doi.org/10.1002/anie.201306438*

Hartshorn, R. M., Hey-Hawkins, E., Kalio, R. & Leigh, G. J. (2007). *Pure Appl. Chem.* **79**(10), 1779–1799.
URL: *http://iupac.org/publications/pac/79/10/1779/*

Hoffmann, R. (1987). *Angew. Chem. Int. Ed.* **26**(9), 846.

Hoffmann, R. (1988). *Solids and surfaces: a chemist's view of bonding in extended structures.*

Hoppe, R. (1979). *Z. Kristallogr. - Cryst. Mater.* **150**(1-4), 23.
URL: *https://www.degruyter.com/view/j/zkri.1979.150.issue-1-4/zkri.1979.150.14.23/zkri.1979.150.14.23.xml*

Jain, A., Hautier, G., Ong, S. P. & Persson, K. (2016). *J. Mater. Res.* **31**(8), 977.

Jain, A., Ong, S. P., Hautier, G., Chen, W., Richards, W. D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G. & Persson, K. a. (2013). *APL Mater.* **1**(1), 011002.
URL: *http://link.aip.org/link/AMPADS/v1/i1/p011002/s1&Agg=doi*

Kabsch, W. (1976). *Acta Cryst.* A**32**(5), 922–923.

Kabsch, W. (1978). *Acta Cryst.* A**34**(5), 827–828.

Lueken, H. (2013). *Magnetochemie: Eine Einführung in Theorie und Anwendung*. Teubner Studienbücher Chemie. Vieweg+Teubner Verlag.
URL: *https://books.google.be/books?id=JMT1BQAAQBAJ*

Maiseli, B., Gu, Y. & Gao, H. (2017). *J. Visual Commun. Image Represent.* **46**(Supplement C), 95 – 106.
URL: *http://www.sciencedirect.com/science/article/pii/S1047320317300743*

Müller, U. (2007). *Inorganic Structural Chemistry*. Wiley.
URL: *https://books.google.de/books?id=s3KlfXCY11sC*

O'Keefe, M. (1979). *Acta Cryst. A*, **35**(1978), 772–775.

Ong, S. P., Richards, W. D., Jain, A., Hautier, G., Kocher, M., Cholia, S., Gunter, D., Chevrier, V. L., Persson, K. A. & Ceder, G. (2013). *Comput. Mater. Sci.* **68**(0), 314–319.
URL: *http://www.sciencedirect.com/science/article/pii/S0927025612006295*

Pauling, L. (1929). *J. Am. Chem. Soc.* **51**(4), 1010.
URL: *https://doi.org/10.1021/ja01379a006*

Peng, H., Ndione, P. F., Ginley, D. S., Zakutayev, A. & Lany, S. (2015). *Phys. Rev. X*, **5**, 021016.
URL: *https://link.aps.org/doi/10.1103/PhysRevX.5.021016*

Pfeiffer, P. (1915). *Z. Anorg. Allg. Chem.* **92**(1), 376–380.
URL: *https://doi.org/10.1002/zaac.19150920126*

Pfeiffer, P. (1916). *Z. Anorg. Allg. Chem.* **97**(1), 161–174.
URL: *https://doi.org/10.1002/zaac.19160970112*

Pinsky, M. & Avnir, D. (1998). *Inorg. Chem.* **37**(21), 5575–5582.
URL: *http://pubs.acs.org/doi/abs/10.1021/ic9804925*

Stoiber, D. & Niewa, R. (2019). *Z. Kristallogr. - Cryst. Mater.* **0**(0).
URL: *https://www.degruyter.com/view/j/zkri.ahead-of-print/zkri-2018-2115/zkri-2018-2115.xml*

Villars, P. & Cenzual, K. (2018). *Pearson's Crystal Data: Crystal Structure Database for Inorganic Compounds (on DVD), Release 2018/19*. ASM International, Materials Park, Ohio, USA.

Waroquiers, D., Gonze, X., Rignanese, G.-M., Welker-Nieuwoudt, C., Rosowski, F., Göbel, M., Schenk, S., Degelmann, P., André, R., Glaum, R. & Hautier, G. (2017). *Chem. Mat.* **29**(19), 8346.
URL: *https://doi.org/10.1021/acs.chemmater.7b02766*

Zimmermann, N. E. R., Horton, M. K., Jain, A. & Haranczyk, M. (2017). *Front. Mater.* **4**, 34.
URL: *https://www.frontiersin.org/article/10.3389/fmats.2017.00034*

Synopsis

We present a new tool called *ChemEnv*, which can identify coordination environments in a fast and robust manner.

Acta Crystallographica Section A
Foundations of Crystallography
ISSN 0108-7673

Supporting Information for "*ChemEnv* : A fast and robust coordination environment identification tool"

David Waroquiers,[a] Janine George,[a] Matthew Horton,[b,c] Stephan Schenk,[d] Kristin A. Persson,[b,c] Gian-Marco Rignanese,[a] Xavier Gonze[a,e] and Geoffroy Hautier [a]*

[a]Institute of Condensed Matter and Nanosciences, Université catholique de Louvain, Chemin des Étoiles 8, 1348 Louvain-la-Neuve, Belgium, [b]Energy Technologies Area, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA, [c]Department of Materials Science and Engineering, University of California, Berkeley, CA 94720, USA, [d]BASF SE, Digitalization of R&D, Carl-Bosch-Str. 38, 67056 Ludwigshafen, Germany, and [e]Skolkovo Institute of Science and Technology, Skolkovo Innovation Center, Nobel St. 3, Moscow, 143026, Russia. Correspondence e-mail: geoffroy.hautier@uclouvain.be

This supplementary information describes the different coordination environments identified by ChemEnv and provides technical details about the identification procedure.

1. Model coordination environments and separation planes

The following lists the model coordination environments for each coordination number. For each model coordination environment, the symbol used in *ChemEnv*, a descriptive name, the coordinates of the points, the IUCr and IUPAC symbols as well as technical details about the algorithm used for the identification are provided.

Coordination 1

- **S:1 → Single neighbor**
  IUCr symbol : [1l]
  IUPAC symbol : None
  Points :

A	0.0000	0.0000	1.0000

  Explicit permutations algorithm

Coordination 2

- **L:2 → Linear**
  IUCr symbol : [2l]
  IUPAC symbol : L-2
  Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000

  Explicit permutations algorithm

- **A:2 → Angular**
  IUCr symbol : [2n]
  IUPAC symbol : A-2
  Points :

A	1.0000	0.0000	0.0000
B	−0.5000	0.8660	0.0000

  Explicit permutations algorithm

Coordination 3

- **TL:3 → Trigonal plane**
  IUCr symbol : [3l]
  IUPAC symbol : TP-3
  Points :

A	0.0000	1.0000	0.0000
B	0.8660	−0.5000	0.0000
C	−0.8660	−0.5000	0.0000

  Explicit permutations algorithm

- **TY:3 → Triangular non-coplanar**
  IUCr symbol : [3n]
  IUPAC symbol : TPY-3
  Points :

A	0.5774	−0.5774	−0.5774
B	−0.5774	0.5774	−0.5774
C	−0.5774	−0.5774	0.5774

  Explicit permutations algorithm

- **TS:3 → T-shaped**
  IUCr symbol : None
  IUPAC symbol : TS-3
  Points :

A	−1.0000	0.0000	0.0000
B	1.0000	0.0000	0.0000
C	0.0000	0.0000	1.0000

  Explicit permutations algorithm

Coordination 4

- **T:4 → Tetrahedron**
  IUCr symbol : [4t]
  IUPAC symbol : T-4
  Points :

A	0.5774	−0.5774	−0.5774
B	−0.5774	0.5774	−0.5774
C	−0.5774	−0.5774	0.5774
D	0.5774	0.5774	0.5774

  Explicit permutations algorithm

- **S:4 → Square plane**
  IUCr symbol : [4l]
  IUPAC symbol : SP-4
  Points :

A	1.0000	0.0000	0.0000
B	−1.0000	0.0000	0.0000
C	0.0000	1.0000	0.0000
D	0.0000	−1.0000	0.0000

  Explicit permutations algorithm

- **SY:4 → Square non-coplanar**
  IUCr symbol : [4n]
  IUPAC symbol : SPY-4
  Points :

A	0.9258	0.0000	0.3780
B	−0.9258	0.0000	0.3780
C	0.0000	0.9258	0.3780
D	0.0000	−0.9258	0.3780

  Explicit permutations algorithm

- **SS:4 → See-saw**
  IUCr symbol : None
  IUPAC symbol : SS-4
  Points :

A	1.0000	0.0000	0.0000
B	0.0000	0.8660	0.5000
C	0.0000	0.0000	−1.0000
D	−1.0000	0.0000	0.0000

  Explicit permutations algorithm

Coordination 5

- **PP:5 → Pentagonal plane**
  IUCr symbol : [5l]
  IUPAC symbol : PP-5
  Points :

A	1.0000	0.0000	0.0000
B	0.3090	0.9511	0.0000
C	−0.8090	0.5878	0.0000
D	−0.8090	−0.5878	0.0000
E	0.3090	−0.9511	0.0000

  Explicit permutations algorithm

- **S:5 → Square pyramid**
  IUCr symbol : [5y]
  IUPAC symbol : SPY-5
  Points :

A	1.0000	0.0000	0.0000
B	−1.0000	0.0000	0.0000
C	0.0000	1.0000	0.0000
D	0.0000	−1.0000	0.0000
E	0.0000	0.0000	1.0000

  Explicit permutations algorithm

- **T:5 → Trigonal bipyramid**
  IUCr symbol : [5by]
  IUPAC symbol : TBPY-5
  Points :

A	0.0000	1.0000	0.0000
B	0.8660	−0.5000	0.0000
C	−0.8660	−0.5000	0.0000
D	0.0000	0.0000	1.0000
E	0.0000	0.0000	−1.0000

  Explicit permutations algorithm

Coordination 6

- **O:6 → Octahedron**
  IUCr symbol : [6o]
  IUPAC symbol : OC-6
  Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000

  Separation plane algorithms :
  ➔ E / ACBD / F
  ➔ ∅ / ACE / FBD

- **T:6 → Trigonal prism**
  IUCr symbol : [6p]
  IUPAC symbol : TPR-6
  Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547

  Separation plane algorithms :
  ➔ ∅ / ABED / CF
  ➔ ∅ / ABC / DEF

- **PP:6 → Pentagonal pyramid**
  IUCr symbol : None
  IUPAC symbol : PPY-6
  Points :

A	1.0000	0.0000	0.0000
B	0.3090	0.9511	0.0000
C	−0.8090	0.5878	0.0000
D	−0.8090	−0.5878	0.0000
E	0.3090	−0.9511	0.0000
F	0.0000	0.0000	1.0000

  Separation plane algorithms :
  ➔ ∅ / ABCDE / F
  ➔ BC / AF / ED

Coordination 7

- **PB:7 → Pentagonal bipyramid**
  IUCr symbol : [7by]
  IUPAC symbol : PBPY-7
  Points :

A	1.0000	0.0000	0.0000
B	0.3090	0.9511	0.0000
C	−0.8090	0.5878	0.0000
D	−0.8090	−0.5878	0.0000
E	0.3090	−0.9511	0.0000
F	0.0000	0.0000	1.0000
G	0.0000	0.0000	−1.0000

  Separation plane algorithms :
  ➔ F / ABCDE / G
  ➔ DE / AFG / CB
- **ST:7 → Square-face capped trigonal prism**
  IUCr symbol : [6p1c]
  IUPAC symbol : TPRS-7
  Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.0000	−1.0000	0.0000

  Separation plane algorithms :
  ➔ G / ABED / CF
  ➔ AD / CFG / BE
- **ET:7 → End-trigonal-face capped trigonal prism**
  IUCr symbol : None
  IUPAC symbol : TPRT-7
  Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.0000	0.0000	1.0000

  Separation plane algorithms :
  ➔ G / ABC / DEF
  ➔ BE / AGD / CF
- **FO:7 → Face-capped octahedron**
  IUCr symbol : None
  IUPAC symbol : OCF-7
  Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774

  Separation plane algorithms :
  ➔ G / ACE / BDF
  ➔ D / AEBF / GC

Coordination 8

- **C:8 → Cube**
  IUCr symbol : [8cb]
  IUPAC symbol : CU-8
  Points :

A	−0.5774	−0.5774	−0.5774
B	0.5774	−0.5774	−0.5774
C	−0.5774	0.5774	−0.5774
D	−0.5774	−0.5774	0.5774
E	−0.5774	0.5774	0.5774
F	0.5774	−0.5774	0.5774
G	0.5774	0.5774	−0.5774
H	0.5774	0.5774	0.5774

  Separation plane algorithms :
  ➔ ∅ / ABGC / DFHE
  ➔ FD / ABHE / GC

- **SA:8 → Square antiprism**
  IUCr symbol : [8acb]
  IUPAC symbol : SAPR-8
  Points :

A	0.0000	0.8595	0.5111
B	0.0000	−0.8595	0.5111
C	0.8595	0.0000	0.5111
D	−0.8595	0.0000	0.5111
E	0.6078	0.6078	−0.5111
F	0.6078	−0.6078	−0.5111
G	−0.6078	0.6078	−0.5111
H	−0.6078	−0.6078	−0.5111

  Separation plane algorithms :
  ➔ (∅ / ACBD / EFHG)
  ➔ (E / ACFG / DBH)
- **SBT:8 → Square-face bicapped trigonal prism**
  IUCr symbol : None
  IUPAC symbol : TPRS-8
  Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.8660	0.5000	0.0000
H	−0.8660	0.5000	0.0000

  Separation plane algorithms :
  ➔ (∅ / ABED / CGFH)
  ➔ (H / ACFD / BGE)
- **TBT:8 → Triangular-face bicapped trigonal prism**
  IUCr symbol : [6p2c]
  IUPAC symbol : TPRT-8
  Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.0000	0.0000	1.0000
H	0.0000	0.0000	−1.0000

  Separation plane algorithm :
  ➔ (AD / CFHG / BE)
- **DD:8 → Dodecahedron with triangular faces**
  IUCr symbol : [8do]
  IUPAC symbol : DD-8
  Points :

A	−0.5000	0.0000	−0.7839
B	0.5000	0.0000	−0.7839
C	0.0000	−0.6446	−0.2056
D	0.0000	0.6446	−0.2056
E	−0.6446	0.0000	0.2056
F	0.6446	0.0000	0.2056
G	0.0000	−0.5000	0.7839
H	0.0000	0.5000	0.7839

  Separation plane algorithm :
  ➔ (CG / ABFE / DH)
- **DDPN:8 → Dodecahedron with triangular faces - p2345 plane normalized**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	−0.5000	0.0000	−0.7839
B	0.5000	0.0000	−0.7839
C	0.0000	−0.9068	−0.2056
D	0.0000	0.9068	−0.2056
E	−0.9068	0.0000	0.2056
F	0.9068	0.0000	0.2056
G	0.0000	−0.5000	0.7839
H	0.0000	0.5000	0.7839

  Separation plane algorithm :
  ➔ (CG / ABFE / DH)
- **HB:8 → Hexagonal bipyramid**
  IUCr symbol : [8by]
  IUPAC symbol : HBPY-8
  Points :

A	1.0000	0.0000	0.0000
B	0.5000	0.8660	0.0000
C	−0.5000	0.8660	0.0000
D	−1.0000	0.0000	0.0000
E	−0.5000	−0.8660	0.0000
F	0.5000	−0.8660	0.0000
G	0.0000	0.0000	1.0000
H	0.0000	0.0000	−1.0000

  Separation plane algorithms :
  ➔ (G / ABCDEF / H)
  ➔ (FE / AHDG / BC)

- **BO_1:8 → Bicapped octahedron (opposed cap faces)**
IUCr symbol : None
IUPAC symbol : OCT-8
Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774
H	−0.5774	−0.5774	−0.5774

Separation plane algorithms :
➔ AD / EGFH / CB
➔ FH / ACBD / GE

- **BO_2:8 → Bicapped octahedron (cap faces with one atom in common)**
IUCr symbol : None
IUPAC symbol : OCT-8
Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774
H	0.5774	−0.5774	−0.5774

Separation plane algorithms :
➔ BE / CGDH / FA
➔ AG / CEDF / HB
➔ D / AEBF / CGH

- **BO_3:8 → Bicapped octahedron (cap faces with one edge in common)**
IUCr symbol : None
IUPAC symbol : OCT-8
Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	−0.5774	0.5774
H	0.5774	−0.5774	−0.5774

Separation plane algorithms :
➔ CE / AGHB / FD
➔ AG / CEDF / BH
➔ E / ACBD / GHF

Coordination 9

- **TC:9 → Triangular cupola**
IUCr symbol : None
IUPAC symbol : TCA-9
Points :

A	1.0000	0.0000	0.0000
B	0.5000	0.8660	0.0000
C	−0.5000	0.8660	0.0000
D	−1.0000	0.0000	0.0000
E	−0.5000	−0.8660	0.0000
F	0.5000	−0.8660	0.0000
G	0.0000	0.5774	0.8165
H	−0.5000	−0.2887	0.8165
I	0.5000	−0.2887	0.8165

Separation plane algorithms :
➔ ∅ / ABCDEF / GHI
➔ EF / ADHI / BCG
➔ BC / ADG / FEHI

- **TT_1:9 → Tricapped triangular prism (three square-face caps)**
IUCr symbol : [6p3c]
IUPAC symbol : TPRS-9
Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.8660	0.5000	0.0000
H	−0.8660	0.5000	0.0000
I	0.0000	−1.0000	0.0000

Separation plane algorithms :
➔ ABC / GHI / DEF
➔ AID / BEH / CGF
➔ I / ABED / CGFH

- **TT_2:9 → Tricapped triangular prism (two square-face caps and one triangular-face cap)**
IUCr symbol : [6p3c]
IUPAC symbol : TPRS-9
Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.8660	0.5000	0.0000
H	−0.8660	0.5000	0.0000
I	0.0000	0.0000	1.0000

Separation plane algorithms :

➔ AD / HIBE / CGF

➔ GEB / CFI / HDA

- **TT_3:9 → Tricapped triangular prism (one square-face cap and two triangular-face caps)**
IUCr symbol : [6p3c]
IUPAC symbol : TPRS-9
Points :

A	−0.6547	−0.3780	0.6547
B	0.6547	−0.3780	0.6547
C	0.0000	0.7559	0.6547
D	−0.6547	−0.3780	−0.6547
E	0.6547	−0.3780	−0.6547
F	0.0000	0.7559	−0.6547
G	0.0000	−1.0000	0.0000
H	0.0000	0.0000	−1.0000
I	0.0000	0.0000	1.0000

Separation plane algorithms :

➔ AD / IGHFC / BE

➔ CF / BEHI / AGD

- **HD:9 → Heptagonal dipyramid**
IUCr symbol : None
IUPAC symbol : HBPY-9
Points :

A	1.0000	0.0000	0.0000
B	0.6235	0.7818	0.0000
C	−0.2225	0.9749	0.0000
D	−0.9010	0.4339	0.0000
E	−0.9010	−0.4339	0.0000
F	−0.2225	−0.9749	0.0000
G	0.6235	−0.7818	0.0000
H	0.0000	0.0000	1.0000
I	0.0000	0.0000	−1.0000

Separation plane algorithm :

➔ H / ABCDEFG / I

- **TI:9 → Tridiminished icosahedron**
IUCr symbol : None
IUPAC symbol : None
Points :

A	0.0000	0.5257	0.8507
B	0.0000	0.5257	−0.8507
C	0.0000	−0.5257	−0.8507
D	0.5257	0.8507	0.0000
E	0.5257	−0.8507	0.0000
F	−0.5257	−0.8507	0.0000
G	0.8507	0.0000	0.5257
H	−0.8507	0.0000	0.5257
I	−0.8507	0.0000	−0.5257

Separation plane algorithms :

➔ GE / AFCD / HIB

➔ B / CDI / EGAHF

- **SMA:9 → Square-face monocapped antiprism**
IUCr symbol : None
IUPAC symbol : SAPRS-9
Points :

A	0.0000	0.8595	0.5111
B	0.0000	−0.8595	0.5111
C	0.8595	0.0000	0.5111
D	−0.8595	0.0000	0.5111
E	0.6078	0.6078	−0.5111
F	0.6078	−0.6078	−0.5111
G	−0.6078	0.6078	−0.5111
H	−0.6078	−0.6078	−0.5111
I	0.0000	0.0000	−1.0000

Separation plane algorithms :

➔ AGE / CDI / BHF

➔ I / EFHG / CBDA

➔ CBF / EHI / ADG

- **SS:9 → Square-face capped square prism**
IUCr symbol : None
IUPAC symbol : CUS-9
Points :

A	−0.5774	−0.5774	−0.5774
B	0.5774	−0.5774	−0.5774
C	−0.5774	0.5774	−0.5774
D	−0.5774	−0.5774	0.5774
E	−0.5774	0.5774	0.5774
F	0.5774	−0.5774	0.5774
G	0.5774	0.5774	−0.5774
H	0.5774	0.5774	0.5774
I	0.0000	0.0000	1.0000

Separation plane algorithms :

➔ BF / ADIHG / CE

➔ I / DEHF / ACGB

➔ BG / ACHF / DEI

- **TO_1:9 → Tricapped octahedron (all 3 cap faces share one atom)**
IUCr symbol : None
IUPAC symbol : TOCT-9
Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774
H	0.5774	−0.5774	0.5774
I	0.5774	−0.5774	−0.5774

Separation plane algorithms :

➔ IBF / CDH / GEA

➔ BE / CGDI / AFH

➔ DF / AHIB / GCE

- **TO_2:9 → Tricapped octahedron (cap faces are aligned)**
  IUCr symbol : None
  IUPAC symbol : TOCT-9
  Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774
H	0.5774	−0.5774	0.5774
I	−0.5774	−0.5774	−0.5774

  Separation plane algorithms :

  ➔ CB / EGHFI / AD

  ➔ EB / CGDI / HAF

- **TO_3:9 → Tricapped octahedron (all 3 cap faces are sharingone edge of a face)**
  IUCr symbol : None
  IUPAC symbol : TOCT-9
  Points :

A	0.0000	0.0000	1.0000
B	0.0000	0.0000	−1.0000
C	1.0000	0.0000	0.0000
D	−1.0000	0.0000	0.0000
E	0.0000	1.0000	0.0000
F	0.0000	−1.0000	0.0000
G	0.5774	0.5774	0.5774
H	−0.5774	0.5774	−0.5774
I	0.5774	−0.5774	−0.5774

  Separation plane algorithms :

  ➔ CGA / FIE / BHD

  ➔ AF / DGCI / HEB

Coordination 10

- **PP:10 → Pentagonal prism**
  IUCr symbol : None
  IUPAC symbol : PPR-10
  Points :

A	1.0000	0.0000	−0.5878
B	0.3090	0.9511	−0.5878
C	−0.8090	0.5878	−0.5878
D	−0.8090	−0.5878	−0.5878
E	0.3090	−0.9511	−0.5878
F	1.0000	0.0000	0.5878
G	0.3090	0.9511	0.5878
H	−0.8090	0.5878	0.5878
I	−0.8090	−0.5878	0.5878
J	0.3090	−0.9511	0.5878

  Separation plane algorithms :

  ➔ ∅ / ABCDE / FGHIJ

  ➔ BG / ACHF / EDIJ

- **PA:10 → Pentagonal antiprism**
  IUCr symbol : None
  IUPAC symbol : PAPR-10
  Points :

A	1.0000	0.0000	−0.4253
B	0.3090	0.9511	−0.4253
C	−0.8090	0.5878	−0.4253
D	−0.8090	−0.5878	−0.4253
E	0.3090	−0.9511	−0.4253
F	0.8090	0.5878	0.4253
G	−0.3090	0.9511	0.4253
H	−1.0000	0.0000	0.4253
I	−0.3090	−0.9511	0.4253
J	0.8090	−0.5878	0.4253

  Separation plane algorithms :

  ➔ ∅ / ABCDE / FGHIJ

  ➔ DIH / CEJG / BAF

- **SBSA:10 → Square-face bicapped square antiprism**
  IUCr symbol : None
  IUPAC symbol : SAPRS-10
  Points :

A	0.0000	0.8595	0.5111
B	0.0000	−0.8595	0.5111
C	0.8595	0.0000	0.5111
D	−0.8595	0.0000	0.5111
E	0.6078	0.6078	−0.5111
F	0.6078	−0.6078	−0.5111
G	−0.6078	0.6078	−0.5111
H	−0.6078	−0.6078	−0.5111
I	0.0000	0.0000	−1.0000
J	0.0000	0.0000	1.0000

  Separation plane algorithms :

➔ CFE / AJBI / DHG

➔ JBC / ADF / GHIE

- **MI:10 → Metabidiminished icosahedron**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	0.0000	0.5257	0.8507
B	0.0000	0.5257	−0.8507
C	0.0000	−0.5257	−0.8507
D	0.5257	0.8507	0.0000
E	0.5257	−0.8507	0.0000
F	−0.5257	−0.8507	0.0000
G	0.8507	0.0000	0.5257
H	−0.8507	0.0000	0.5257
I	−0.8507	0.0000	−0.5257
J	0.8507	0.0000	−0.5257

  Separation plane algorithms :

  ➔ FGE / AJCH / IDB

  ➔ AH / GDIF / JBCE

- **BS_1:10 → Bicapped square prism (opposite faces)**
  IUCr symbol : None
  IUPAC symbol : CUS-10
  Points :

A	−0.5774	−0.5774	−0.5774
B	0.5774	−0.5774	−0.5774
C	−0.5774	0.5774	−0.5774
D	−0.5774	−0.5774	0.5774
E	−0.5774	0.5774	0.5774
F	0.5774	−0.5774	0.5774
G	0.5774	0.5774	−0.5774
H	0.5774	0.5774	0.5774
I	0.0000	0.0000	−1.0000
J	0.0000	0.0000	1.0000

  Separation plane algorithms :

  ➔ FB / ADJHGI / EC

  ➔ ∅ / ABFD / CIGHJE

- **BS_2:10 → Bicapped square prism (adjacent faces)**
  IUCr symbol : None
  IUPAC symbol : CUS-10
  Points :

A	−0.5774	−0.5774	−0.5774
B	0.5774	−0.5774	−0.5774
C	−0.5774	0.5774	−0.5774
D	−0.5774	−0.5774	0.5774
E	−0.5774	0.5774	0.5774
F	0.5774	−0.5774	0.5774
G	0.5774	0.5774	−0.5774
H	0.5774	0.5774	0.5774
I	1.0000	0.0000	0.0000
J	0.0000	1.0000	0.0000

  Separation plane algorithms :

  ➔ BFI / ADHG / CEJ

  ➔ AD / EFBC / JHIG

  ➔ I / BFHG / ADEJC

- **TBSA:10 → Trigonal-face bicapped square antiprism**
  IUCr symbol : None
  IUPAC symbol : SAPRT-10
  Points :

A	0.0000	0.8595	0.5111
B	0.0000	−0.8595	0.5111
C	0.8595	0.0000	0.5111
D	−0.8595	0.0000	0.5111
E	0.6078	0.6078	−0.5111
F	0.6078	−0.6078	−0.5111
G	−0.6078	0.6078	−0.5111
H	−0.6078	−0.6078	−0.5111
I	0.0000	0.9710	−0.2391
J	0.0000	−0.9710	−0.2391

  Separation plane algorithms :

  ➔ DHG / ABJI / CFE

  ➔ ∅ / ACBD / IEFJHG

Coordination 11

- **PCPA:11 → Pentagonal-face capped pentagonal antiprism**
  IUCr symbol : None
  IUPAC symbol : PPRP-11
  Points :

A	1.0000	0.0000	−0.5878
B	0.3090	0.9511	−0.5878
C	−0.8090	0.5878	−0.5878
D	−0.8090	−0.5878	−0.5878
E	0.3090	−0.9511	−0.5878
F	1.0000	0.0000	0.5878
G	0.3090	0.9511	0.5878
H	−0.8090	0.5878	0.5878
I	−0.8090	−0.5878	0.5878
J	0.3090	−0.9511	0.5878
K	0.0000	0.0000	1.0000

  Separation plane algorithms :

  ➔ K / FGHIJ / ABCDE

  ➔ BGHC / AFK / EJID

- **H:11 → Hendecahedron**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	0.0000	0.0000	2.0000
B	2.0000	1.0000	1.0000
C	0.0000	−1.0000	1.0000
D	−2.0000	1.0000	1.0000
E	0.0000	2.0000	0.0000
F	1.0000	−1.0000	0.0000
G	−1.0000	−1.0000	0.0000
H	2.0000	1.0000	−1.0000
I	0.0000	−1.0000	−1.0000
J	−2.0000	1.0000	−1.0000
K	0.0000	0.0000	−2.0000

  Separation plane algorithm :
  ➔ (DGJ / EACIK / BFH)
- **DI:11 → Diminished icosahedron**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	0.0000	−1.0000	−1.6180
B	0.0000	1.0000	−1.6180
C	0.0000	−1.0000	1.6180
D	0.0000	1.0000	1.6180
E	−1.0000	−1.6180	0.0000
F	1.0000	−1.6180	0.0000
G	−1.0000	1.6180	0.0000
H	1.0000	1.6180	0.0000
I	−1.6180	0.0000	−1.0000
J	−1.6180	0.0000	1.0000
K	1.6180	0.0000	−1.0000

  Separation plane algorithms :
  ➔ (I / GJEAB / DCFKH)
  ➔ (FHK / ACDB / EJGI)

Coordination 12

- **I:12 → Icosahedron**
  IUCr symbol : [12i]
  IUPAC symbol : IC-12
  Points :

A	0.0000	−1.0000	−1.6180
B	0.0000	1.0000	−1.6180
C	0.0000	−1.0000	1.6180
D	0.0000	1.0000	1.6180
E	−1.0000	−1.6180	0.0000
F	1.0000	−1.6180	0.0000
G	−1.0000	1.6180	0.0000
H	1.0000	1.6180	0.0000
I	−1.6180	0.0000	−1.0000
J	−1.6180	0.0000	1.0000
K	1.6180	0.0000	−1.0000
L	1.6180	0.0000	1.0000

  Separation plane algorithm :
  ➔ (EIGJ / ABDC / FKHL)
- **PBP:12 → Pentagonal-face bicapped pentagonal prism**
  IUCr symbol : None
  IUPAC symbol : PPRP-12
  Points :

A	1.0000	0.0000	−0.5878
B	0.3090	0.9511	−0.5878
C	−0.8090	0.5878	−0.5878
D	−0.8090	−0.5878	−0.5878
E	0.3090	−0.9511	−0.5878
F	1.0000	0.0000	0.5878
G	0.3090	0.9511	0.5878
H	−0.8090	0.5878	0.5878
I	−0.8090	−0.5878	0.5878
J	0.3090	−0.9511	0.5878
K	0.0000	0.0000	−1.0000
L	0.0000	0.0000	1.0000

  Separation plane algorithm :
  ➔ (EJID / AFLK / BGHC)
- **TT:12 → Truncated tetrahedron**
  IUCr symbol : [12tt]
  IUPAC symbol : None
  Points :

A	−0.5774	0.5774	−1.7321
B	0.5774	−0.5774	−1.7321
C	−0.5774	−1.7321	0.5774
D	0.5774	−1.7321	−0.5774
E	1.7321	0.5774	0.5774
F	1.7321	−0.5774	−0.5774
G	−1.7321	−0.5774	0.5774
H	−1.7321	0.5774	−0.5774
I	0.5774	1.7321	0.5774
J	−0.5774	1.7321	−0.5774
K	0.5774	0.5774	1.7321
L	−0.5774	−0.5774	1.7321

Separation plane algorithms :

➔ CD / GLFB / KIEAJH

➔ BDF / ACE / HGLKIJ

- **C:12 → Cuboctahedron**
  IUCr symbol : [12co]
  IUPAC symbol : None
  Points :

A	0.0000	−1.0000	−1.0000
B	0.0000	1.0000	−1.0000
C	0.0000	−1.0000	1.0000
D	0.0000	1.0000	1.0000
E	−1.0000	−1.0000	0.0000
F	1.0000	−1.0000	0.0000
G	−1.0000	1.0000	0.0000
H	1.0000	1.0000	0.0000
I	−1.0000	0.0000	−1.0000
J	−1.0000	0.0000	1.0000
K	1.0000	0.0000	−1.0000
L	1.0000	0.0000	1.0000

  Separation plane algorithms :

  ➔ BKH / AFLDGI / ECJ

  ➔ IGJE / ABDC / KHLF

- **AC:12 → Anticuboctahedron**
  IUCr symbol : [12aco]
  IUPAC symbol : None
  Points :

A	1.0000	0.0000	0.0000
B	0.5000	0.8660	0.0000
C	−0.5000	0.8660	0.0000
D	−1.0000	0.0000	0.0000
E	−0.5000	−0.8660	0.0000
F	0.5000	−0.8660	0.0000
G	0.5000	0.2887	−0.8165
H	−0.5000	0.2887	−0.8165
I	0.0000	−0.5774	−0.8165
J	0.5000	0.2887	0.8165
K	−0.5000	0.2887	0.8165
L	0.0000	−0.5774	0.8165

  Separation plane algorithm :

  ➔ GHI / ABCDEF / JKL

- **SC:12 → Square cupola**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	0.9239	0.3827	0.0000
B	0.3827	0.9239	0.0000
C	−0.3827	0.9239	0.0000
D	−0.9239	0.3827	0.0000
E	−0.9239	−0.3827	0.0000
F	−0.3827	−0.9239	0.0000
G	0.3827	−0.9239	0.0000
H	0.9239	−0.3827	0.0000
I	0.5054	0.0000	0.8409
J	0.0000	0.5054	0.8409
K	−0.5054	0.0000	0.8409
L	0.0000	−0.5054	0.8409

  Separation plane algorithm :

  ➔ ∅ / ABCDEFGH / IJKL

- **HP:12 → Hexagonal prism**
  IUCr symbol : [12p]
  IUPAC symbol : HPR-12
  Points :

A	1.0000	0.0000	−0.5000
B	0.5000	0.8660	−0.5000
C	−0.5000	0.8660	−0.5000
D	−1.0000	0.0000	−0.5000
E	−0.5000	−0.8660	−0.5000
F	0.5000	−0.8660	−0.5000
G	1.0000	0.0000	0.5000
H	0.5000	0.8660	0.5000
I	−0.5000	0.8660	0.5000
J	−1.0000	0.0000	0.5000
K	−0.5000	−0.8660	0.5000
L	0.5000	−0.8660	0.5000

  Separation plane algorithms :

  ➔ ∅ / ABCDEF / GHIJKL

  ➔ FEKL / ADJG / BCIH

- **HA:12 → Hexagonal antiprism**
  IUCr symbol : None
  IUPAC symbol : HAPR-12
  Points :

A	1.0000	0.0000	−0.4278
B	0.5000	0.8660	−0.4278
C	−0.5000	0.8660	−0.4278
D	−1.0000	0.0000	−0.4278
E	−0.5000	−0.8660	−0.4278
F	0.5000	−0.8660	−0.4278
G	0.8660	0.5000	0.4278
H	0.0000	1.0000	0.4278
I	−0.8660	0.5000	0.4278
J	−0.8660	−0.5000	0.4278
K	0.0000	−1.0000	0.4278
L	0.8660	−0.5000	0.4278

  Separation plane algorithm :

  ➔ ∅ / ABCDEF / GHIJKL

Coordination 13

- **SH:13 → Square-face capped hexagonal prism**
  IUCr symbol : None
  IUPAC symbol : None
  Points :

A	1.0000	0.0000	−0.5000
B	0.5000	0.8660	−0.5000
C	−0.5000	0.8660	−0.5000
D	−1.0000	0.0000	−0.5000
E	−0.5000	−0.8660	−0.5000
F	0.5000	−0.8660	−0.5000
G	1.0000	0.0000	0.5000
H	0.5000	0.8660	0.5000
I	−0.5000	0.8660	0.5000
J	−1.0000	0.0000	0.5000
K	−0.5000	−0.8660	0.5000
L	0.5000	−0.8660	0.5000
M	0.9682	−0.5590	0.0000

  Separation plane algorithm :

  ➔ ∅ / ABCDEF / GHIJKLM