Dalton Transactions

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Cite this: *Dalton Trans.*, 2021, **50**, 1086

OctaDist: a tool for calculating distortion parameters in spin crossover and coordination complexes†

Rangsiman Ketkaew, [a] Yuthana Tantirungrotechai, [a] Phimphaka Harding, [b] Guillaume Chastanet, [c] Philippe Guionneau, [c] Mathieu Marchivie *[c] and David J. Harding *[b]

Received 21st November 2020,
Accepted 17th December 2020

DOI: 10.1039/d0dt03988h

rsc.li/dalton

OctaDist is an interactive and visual program for determination of structural distortion in octahedral coordination complexes such as spin crossover complexes, single-ion magnets, perovskites or metal–organic frameworks. OctaDist computes the octahedral distortion parameters initially designed in the context of the spin-crossover phenomenon and denoted ζ, Σ, and Θ from standard structural files. The program also provides additional tools for molecular analyses and visualization. It emphasizes performance, flexibility, ease of use, application programming interface (API) consistency, and clear documentation. The modules and classes in OctaDist can be easily customized to include new algorithms or analytical tools. OctaDist is cross-platform supported for modern operating systems and is available as open-source distributed under the GNU General Public License version 3.

Introduction

Many transition metal complexes adopt octahedral geometries, with six donor atoms forming bond angles of *ca.* 90° about the central atom with adjacent ligands. Nevertheless, among the numerous natural or synthetic complexes the geometries around the metal ion can significantly deviate from a perfect octahedron depending on the nature of the ligands and the surrounding environment in the solid state. The distortion from octahedral geometry can be strongly correlated to their physical properties including directly observable ones like color to more specific properties like magnetism. Among magnetic compounds, the design of property-controllable spin crossover (SCO) materials remains a key goal of SCO research in part because of their potential in a range of new technologies.[1–4] Many molecular, 1D, 2D and 3D SCO systems, often based on iron, have been synthesized in recent decades.[5,6] These studies have provided new principles for the design, not only of SCO compounds, but also iron-based catalysts where the reactivity is often spin-state dependent.[7,8] Spin crossover involves switching between low-spin (LS) and high-spin (HS) states and is determined by the free energy difference and depends on the oxidation state of the central metal ion and the ligand donors. In the LS state, the bonding t_{2g} orbitals are preferentially occupied while in the HS state some electrons are promoted in to the antibonding e_g orbitals. Consequently, the LS state has shorter metal–ligand bond lengths and thus smaller vibrational entropy. This in turn results in a less distorted geometry at the metal centre.[9] Extensive structural studies have shown that the degree of SCO and even its abruptness may be related to the type and extent of the geometric distortion at the metal centre.[10] It follows that the determination of the degree of structural distortion is a key consideration in the rationalization of SCO behaviour, though all physical scales must be investigated to get the complete picture.[11] The role of octahedral distortion, for example, has been identified as a key parameter to obtain long lifetimes in photo-induced systems.[12] Spin crossover from LS to HS in octahedral iron(II) complexes results in a large increase in the six Fe(II)–N bond lengths.[13,14]

One of the simplest parameters to track this change is d_{mean} which refers to the average metal–ligand distances in the octahedral coordination sphere. However, in order to comprehensively represent the distortion of the octahedron, the accompanying parameters ζ,[15] Σ,[16,17] and Θ[18] have been proposed and used for considering stretching, angular, and torsional distortions, respectively. ζ and Σ are general deviations of the metal ion complex from an ideal octahedral structure, whilst Θ represents a distortion from a perfect octahedral (O_h)

[a]*Computational Chemistry Research Unit, Department of Chemistry, Faculty of Science and Technology, Thammasat University, Pathum Thani, 12120 Thailand*
[b]*Functional Materials and Nanotechnology Center of Excellence, Walailak University, Thasala, Nakhon Si Thammarat, 80160, Thailand. E-mail: hdavid@mail.wu.ac.th*
[c]*CNRS, Univ. Bordeaux, Bordeaux INP, ICMCB, UMR 5026, 87 av. Dr A. Schweitzer, F-33600 Pessac, France. E-mail: mathieu.marchivie@icmcb.cnrs.fr*

†Electronic supplementary information (ESI) available. See DOI: 10.1039/d0dt03988h

to a trigonal prismatic (D_{3h}) geometry. The mathematical expressions of ζ, Σ, and Θ parameters are given by the following equations:[19]

$$\zeta = \sum_{i=1}^{6} |d_i - d_{mean}|$$

$$\Sigma = \sum_{i=1}^{6} |\phi_i - 90|$$

$$\Theta = \sum_{i=1}^{6} |\theta_i - 60|$$

The calculation of the ζ and Σ parameters is straightforward, providing that the structural data are meaningful. The latter implies, in the case of SCO compounds, the use of accurate enough data determined without any bias due to an incomplete and ongoing transition or to other structural events such as twinning and symmetry breaking. The parameter ζ is the average of the sum of the deviation of 6 unique metal–ligand bond lengths around the central metal atom (d_i) from the average value (d_{mean}).[15] The parameter Σ is the sum of the deviation of 12 unique *cis* ligand–metal–ligand angles (ϕ_i) from 90°.[16,17] The parameter Θ is defined as the degree of trigonal distortion of the coordination geometry from an octahedron towards a trigonal prism. The Θ parameter is the sum of the deviation of 24 unique torsional angles between the ligand atoms on opposite triangular faces of the octahedron viewed along the pseudo-threefold axis (θ_i) from 60°.[18,20] The Θ parameter was popularized almost two decades ago by Marchivie *et al.* who demonstrated that the change in Θ between the HS and LS states is correlated with the limiting temperature of photo-inscription, T(LIESST). In all cases for a perfect octahedron, the values of ζ, Σ, and Θ are zero.

Previous studies on iron SCO complexes indicate that while ζ and Σ are consistently reported, within the limit mentioned above on the initial experimental data, it appears that for Θ no universally agreed upon method is used leading to data in the literature being almost unusable for investigation of structure–properties relationships. This inhomogeneity of calculation arises from the structure being distorted in such a way that the symmetry is lowered, and thus the medium plane of the two opposite faces can be hard to determine directly. Consequently, it is challenging to find a clear strategy to calculate Θ while adhering to the above definition. A universally agreed method is crucial so that comparisons can be made across the increasingly diverse structural SCO database. To address this, we propose a new method to find the 4 optimal pairs of faces and use orthogonal vector projection to compute the unique dihedral angles (θ_i) on the triangular faces, leading to a mathematically robust (Θ) parameter.

To analyze crystal structures computer programs such as Mercury,[21] SHAPE,[22,23] SHELXL,[24] OLEX2,[25] and ChemCraft[26] are often used. While they are useful for the particular purpose for which they were designed, none of these programs is able to calculate octahedral distortion parameters. In this paper, we present a new program called the Octahedral Distortion Calculator (OctaDist) for automatically calculating ζ, Σ, and Θ. The standard features of OctaDist are:

- Extraction of molecular information from many standard chemical file formats
- Computation of octahedral distortion parameters
- Structural analysis tools
- 2D and 3D molecular visualizations
- Built-in scripting language
- Python application programming interface (API)
- Consistency between the graphical user interface (GUI) and the command line interface (CLI).

Our primary purpose in developing OctaDist is to help and support researchers in determining the degree of distortion at an octahedral metal center. While our focus is on spin crossover complexes, the program can be used in other octahedral complexes such as catalysts, and will, we hope, be of use to a wide range of inorganic chemists.

Computational details

As ζ and Σ have a clear mathematical basis and solution, we pay special attention to Θ. To determine Θ it is necessary to select a pair of medium planes that contain the central metal atom first. However, due to the lower symmetry of real metal complexes, it can be difficult for a program to determine the best fitting planes from the 6 ligand atoms. To solve this problem, we propose a method that directly defines the four projection planes for finding the pairs of triangular faces of the octahedron using an orthogonal vector projection.

First, we determine the 8 faces of the octahedral structure. As can be seen from Fig. 1, we orthogonally project the central metal atom and opposite ligand atoms onto the reference face. The new location of the ligand atoms on the new face is called a projected ligand atom. Then, the 6 unique torsional angles (θ_i) are computed as the angle between V_1 and V_2, where V_1 and V_2 are vectors from a projected metal centre atom to a reference ligand atom and to a projected ligand atom, respectively. Examples of the graphical representation of the orthogonal projection and twisting triangular faces can also be seen in Fig. 1.

In the case of a standard octahedral structure, the 3 ligand atoms on the reference face and the 3 atoms on the other opposite face are located alternately. However, in more distorted structures the ligand atoms on the triangular faces may not be located alternately. It follows that the sign of the θ_i angles need to be considered to accurately calculate the Θ parameter. We use the cross-product method to calculate the clockwise (CW)/counterclockwise (CCW) angle between two successive vectors to calculate the unique torsional angle. Computing the angle by means of a cross-product provides us both the value and sign of the angles simultaneously. This means the torsional angle (θ_i) can be either negative or positive depending on the direction of the vector to vector with respect to the normal axis. This is shown in Fig. 2, where for projec-

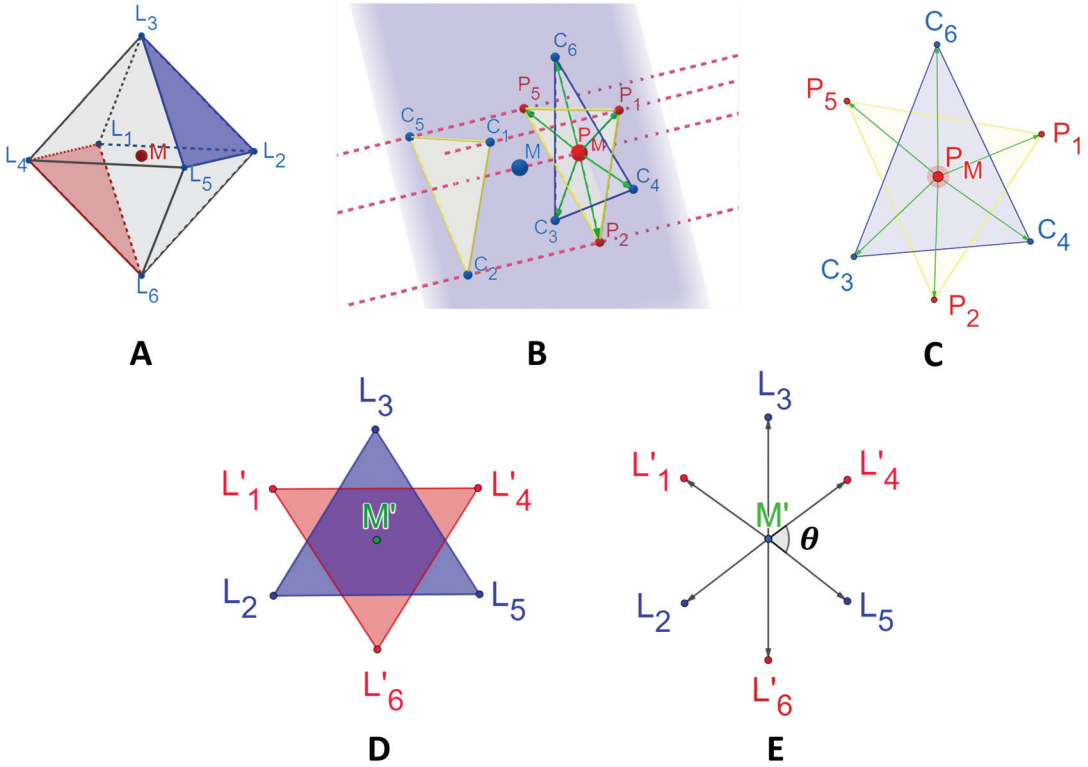

Fig. 1 (A) A pair of projection planes defined by the face of the octahedron: front and back faces are highlighted in blue and red, respectively. (B) Orthogonal projection of triangular face on reference face. (C) A 2D view of projection and reference planes created from (B) showing distortion of the octahedron. (D) Picture of a pair of triangular faces viewed along the pseudo 3-fold axis. (E) View of one of the unique torsional angles (θ_i) between the two vectors of twisting triangular faces.

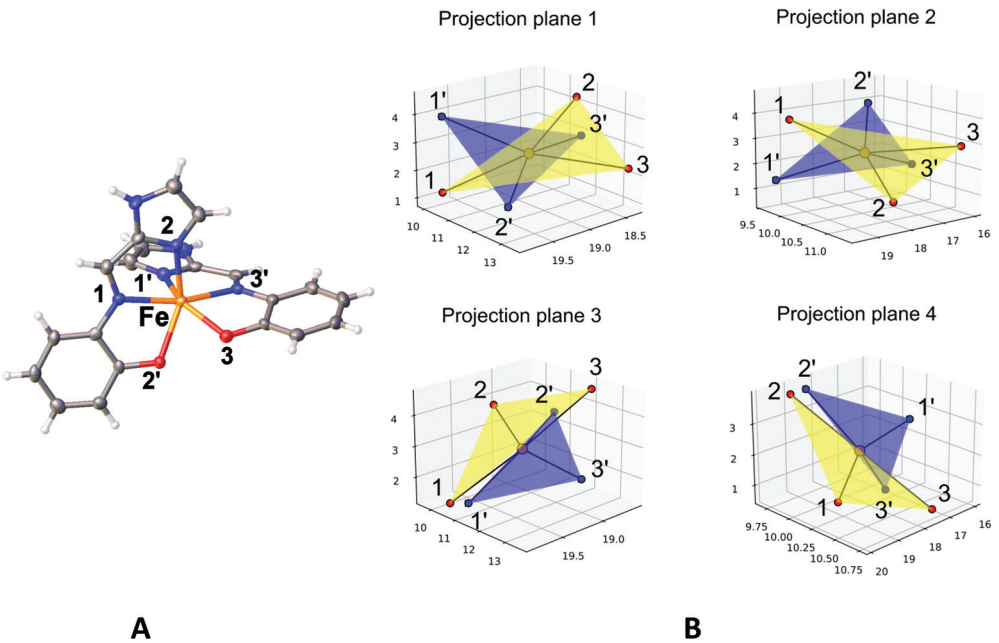

Fig. 2 (A) View of the highly distorted peripheral Fe(II) center in the cluster $[Fe_4(2\text{-Himap})_6][NO_3]_3$ (2-Himap = 2-((1*H*-imidazol-2-yl)methylene-amino)phenolate) with the labels matching those in projection plane 1.[27] (B) Projected twisting triangular planes in $[Fe_4(2\text{-Himap})_6][NO_3]_3$. Ligand atoms on the reference and opposite planes are highlighted in red and blue, respectively.

tion planes 1–2 the θ_i angles are positive, while for planes 3 and 4 the angle between red atom no. 1 and blue atom no. 1′, no. 2 and 2′ for 4, is negative.

Additionally, since the symmetry of real complexes is lower than a normal octahedron, the faces are not necessarily parallel and the projection onto one face is not equivalent to the projection onto the reverse face. To take this into account, both projections are calculated for each of the four directions of projection and the mean value of the distortion parameter is calculated.

Distortion of the Fe polyhedron in SCO complexes

To test the accuracy of the proposed algorithm we have used OctaDist to calculate Θ for a range of iron-based SCO systems with the results shown in Tables S1 and S2 in ESI.† Fig. 3 shows the correlation plot between Θ computed by OctaDist and that taken from previous works.[28–30] In general, the results from OctaDist mirror those previously reported. As the exact method used to calculate previously reported Θ values is not explicitly stated in the papers, some difference is expected. In cases where considerable differences are observed, this seems to stem from the fact that these Θ values were calculated using drawn planes, with centroids and then measuring the torsion angle between the donor atom–centroid–centroid–donor atom. This procedure ignores the position of the metal atom which can be very far away from the two centroids of the octahedral faces and we believe artificially lowers Θ. Another difference comes from the choice of the projecting plane. As mentioned above, because the symmetry is lower than in a normal octahedron, the faces are not necessarily parallel and the projection onto one face is not equivalent to the projection onto the reverse face. This can lead to significant deviations depending on the chosen projection face if the mean value of both is not calculated. In (rare) systems where the metal ligand bond lengths are similar over the whole coordination sphere, the impact is small but when these are significantly different, as in Fe(III) SCO systems and $[Fe(NCS)_2(diimine)_2]$ complexes, the difference is greater. This effect also explains why the correlation between the calculated and published values is better for the complexes in the LS state where the metal–ligand bond lengths are more homogenous. The deviation between the calculated values using OctaDist and the Θ values shown here illustrate clearly the need to standardize the calculation of this parameter. Indeed, several different research groups have attempted to find trends between SCO features and such structural distortion parameters. For instance, the limiting temperature of photo-inscription is suspected to be linked to this trigonal distortion but its generalization to a large number of complexes of different families requires a unified method of calculation to be used.

Examining a wider range of compounds, we have also calculated the distortion parameter Θ with OctaDist for all the Fe–N_6 complexes found in the Cambridge Structural Database (CSD) database[31] (search October 2020) *versus* the generalized coordinate obtained from the SHAPE software.[32] The generalized coordinate is the angular path fraction between two different shapes along the minimal distortion interconversion pathway. This parameter is zero when the structure is coincident with the first ideal shape, and 100 when it is at the end of the path, having another ideal shape. It can be viewed as the degree of distortion between two reference shapes. It is meaningful only for structures that fall close to the minimal distortion interconversion pathway. Among the 7627 six coordinate geometries found, some correspond to the same crystal structure as distinct Fe–N_6 moieties can be found in the same crystal structure. The value obtained for Θ is between 0 and 1170° which is the maximum value of Θ for a perfect trigonal

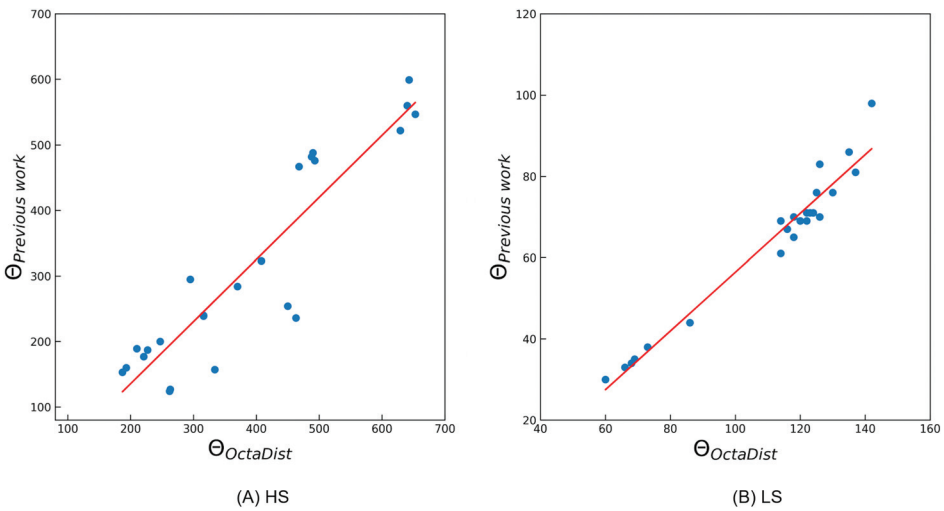

Fig. 3 Correlation between published and standardized Θ values of selected series of Fe complexes for (A) high-spin (HS) and (B) low-spin (LS) states.

prismatic geometry. The general trend is quasi linear meaning that Θ is another way to measure the distortion from perfect octahedral geometry to trigonal prismatic (Fig. 4). Note also that Θ can be calculated for all complexes while the generalized coordinate is meaningful only for structures that do not significantly deviate from the interconversion pathway.

It is interesting to note that the evolution is not exactly the same in the "octahedral" part and in the "trigonal prismatic" part of the chart, Θ increases more rapidly in the second part. Furthermore, there is greater deviation from the mean value of the trend for the octahedral geometry for which this parameter is better adapted. The spreading of the curve especially for the medium values of Θ ($\Theta \approx 200°$) corresponds to distortions of a different nature (angular, bond lengths) that can be easily detected using OctaDist because they generally correspond to very distinct behaviour of angular and bond lengths distortions but are difficult to evaluate concurrently with other software. For example, the same values of the generalized coordinate of SHAPE[33] are obtained for significantly different Θ values (generalized coordinate around 20 in Fig. 4). In this case high Θ values correspond to low ζ values (high angular distortion but low bond length distortion) and low Θ values correspond to high ζ values (low angular distortion but high bond length distortion). Using OctaDist, we are able to separate these diverse contributions.

A powerful feature of OctaDist is the ability to run calculations using a batch of several files that can be easily extracted from the CSD database. To illustrate this point, a CSD search has been performed to find and separate all Fe–N_6 complexes involving monodentate, two bidentate and two tridentate ligands. The results have been extracted from the CSD database using the CSD coordinate file export tool. The resulting file was then used to create the corresponding XYZ files that can be read by OctaDist. Fig. 5 gathers the results obtained for each type of complex showing the different Θ values *versus* the mean Fe–N distances.

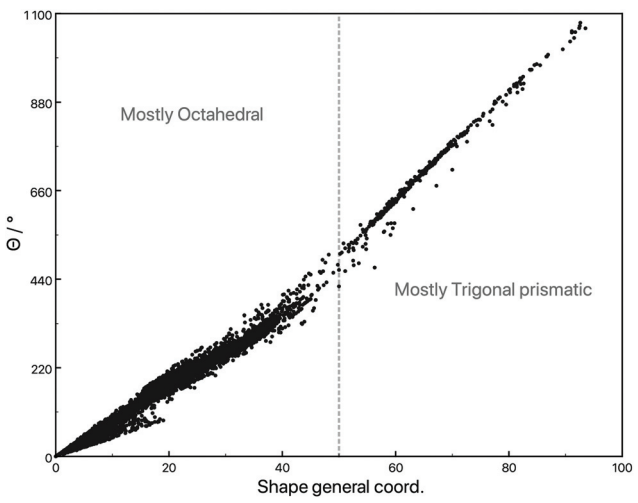

Fig. 4 Evolution of Θ calculated with OctaDist *versus* the generalized coordinate between the perfect octahedron and the trigonal prismatic geometry calculated with SHAPE for complexes that do not deviate more that 20% from the interconversion pathway.

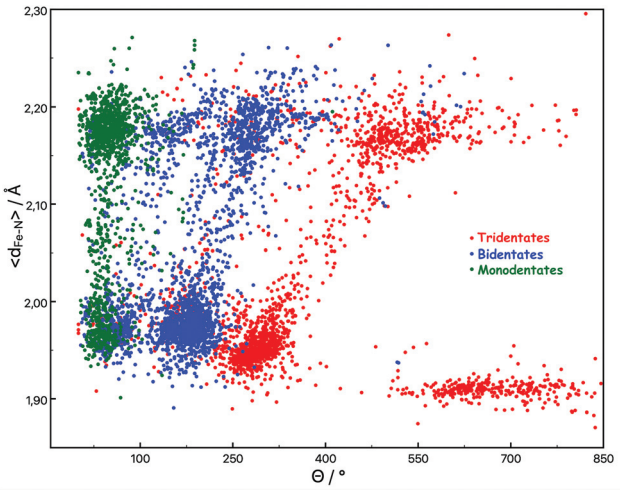

Fig. 5 Representation of the mean Fe–N distance *versus* Θ calculated with OctaDist for a series of 6546 Fe–N_6 complexes with ligands of different denticity extracted from the CSD. Complexes with only monodentate ligands are drawn as green circles, complexes containing 2 bidentate ligands are drawn as blue circles and complexes containing 2 tridentate ligands are drawn as red circles.

Firstly, we observe that all complexes separate into two different groups related to mean Fe–N distances of *ca.* 1.95 Å and 2.20 Å. These two groups correspond clearly to LS and HS complexes, respectively. The different kind of complexes are clearly separated into 3 groups that correspond to different octahedron distortions; these three groups match almost perfectly the monodentate, bidentate and tridentate type of complexes. It can be concluded that the monodentate complexes are the least distorted ($\Theta \approx 50°$ to 75° in the LS and HS state, respectively), followed by the bidentate ones ($\Theta \approx 175°$ to 275° in the LS and HS state, respectively) and the most distorted correspond to the tridentate complexes ($\Theta \approx 300°$ to 500° in the LS and HS state, respectively). A fourth group corresponding to highly distorted complexes ($\Theta > 550°$) mainly in the LS state and may correspond to non-octahedral systems probably closer to the trigonal prismatic geometry (see Fig. 4). HS complexes with a trigonal prismatic geometry are found for higher values of Θ but are not shown in Fig. 5.

This analysis reveals several clear trends that can also be extracted from Fig. 5: (i) the octahedral distortion is clearly related to the denticity: the higher the denticity, the greater the distortion. (ii) As expected, complexes in the HS state have a more distorted coordination sphere than those in the LS state, but (iii) the difference between the HS and LS distortion is much more pronounced for polydentate complexes, following directly the denticity ($\Theta_{HS} - \Theta_{LS} \approx 25°$ for monodentate, 100° for bidentate and 200° for tridentate). Such an evolution of the distortion of the coordination sphere of iron complexes with denticity has been assumed for a while but, to the best of our knowledge, has never been proved on such a large structural database so far. This example clearly demonstrates the

richness of the perspectives offered by OctaDist, which makes it possible to process very large datasets.

Architecture and implementation

Software structure

OctaDist is written entirely in the object-oriented Python 3 programming language.[34] It has a simple architecture and is designed to have an easy-to-use application programming interface (API). The architecture includes range modules, functions, and commands as discussed below. It is composed of a number of built-in modules (libraries) having different functions for a specific task. The function is implemented using a special class constructor allowing us to access its attributes and to initiate a new object (variable) in the class by an inheritance mechanism. To keep all related independent functions in OctaDist together, all classes and submodules are dynamically connected to a main module where the GUI widgets are built. This makes OctaDist compact and avoids inconsistency when working with several functions simultaneously. Fig. 6 graphically shows the simple architecture of OctaDist in which a data processing pipeline is used. Users can simply compute the distortion parameters following the functional steps highlighted in green. In addition, OctaDist allows the user to adjust the value of the parameters used in the calculation as needed. The default settings and values of those parameters can be readily restored.

The native GUI of OctaDist is shown in Fig. 7. This reports the imported data such as Cartesian coordinate of molecules, computed results, displays the structure of molecules and allows the user to navigate through a console menu bar to open and edit input files and save results. Molecular information is encapsulated as a Class. Atomic symbols are intrinsically stored in a list, while the atomic coordinates are represented by an array managed and manipulated by the NumPy package.[35]

The alternative to the GUI is the command line interface (CLI). The CLI allows the user to perform system commands: bulk operations on files and folders in a quick and easy way because the cache overhead of updating the interface with each step is eliminated. To support the CLI, a name-based direct object import registration is used and is implemented by entry points, which are available in the setuptools package. Fig. S1 in the ESI† shows an example of the normal usage of the CLI of OctaDist.

All modules in OctaDist are tested using a doctest package in the standard Python test suite. The doctest tests OctaDist by running the examples embedded in the docstrings in the documentation and verifying that they produce the expected results. Before creating a major release of and distributing OctaDist to the community, the preliminary OctaDist package was tested using the Travis CI continuous integration service. Travis CI simulates the environment and OS and performs various operations on the code. The configuration setting for Travis CI can be customized in .tarvis.yml file. The simple

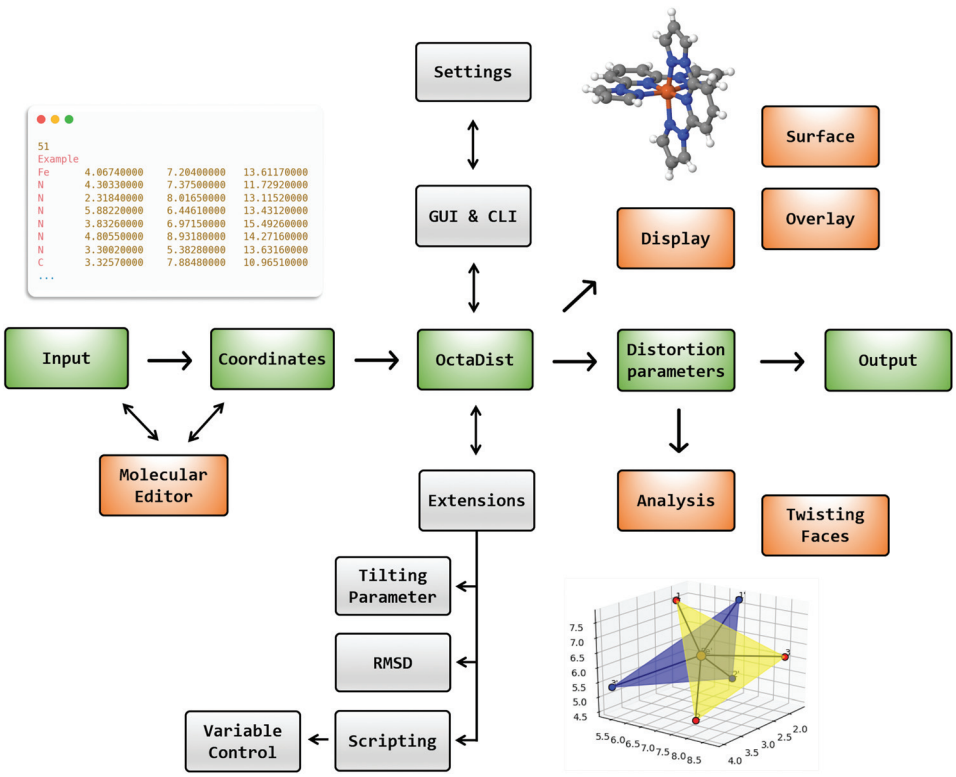

Fig. 6 Simple flowchart showing the architecture of OctaDist. The main processes in OctaDist are represented by the green line.

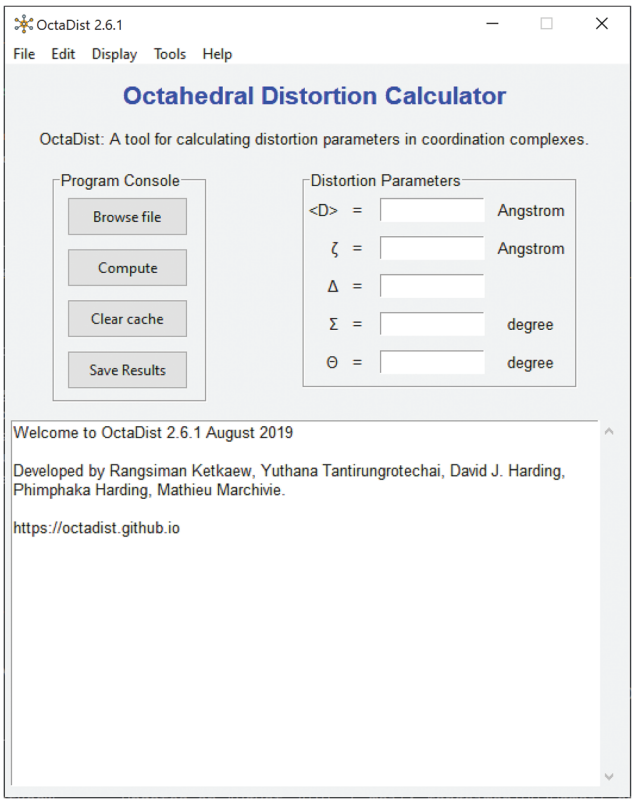

Fig. 7 Graphical user interface of OctaDist.

structure of OctaDist allows users and developers to easily modify the source code to meet their own scientific needs and workflow.

Compilation and installation

A stable release of OctaDist is available to download on the Python Package Index (PyPI) library at https://pypi.org/project/octadist/. It can be installed on most operating systems using the standard Python package manager called pip. The user can install OctaDist and other package dependencies *via* commands, such as

python -m pip install -U octadist

This will install OctaDist and make it available as a script on the command line for the local user but does not write data to the system directories. The user can also build a standalone executable (binary) to run OctaDist without having Python on the operating system. As the OctaDist modules require other external packages and libraries, it is necessary to build OctaDist and all its dependencies into a single standalone executable. A pre-built binary of OctaDist using PyInstaller is available at the official website of the program (https://octadist.github.io).[36] In addition, OctaDist is available at the Anaconda package repository (https://anaconda.org/rangsiman/octadist). The user can use conda, a tool for managing and deploying applications and packages for Anaconda, to install OctaDist *via* command:

conda install -c rangsiman octadist

OctaDist can be executed directly without any intermediate compilation to the machine code for GUI and CLI *via* commands:

octadist

and

octadist_cli –input molecule.xyz

System requirements

OctaDist can operate on most modern operating systems both 32-bit and 64-bit systems. The recommended minimum screen resolution is 1024 × 768 pixels. For the best experience, the device should meet the following requirements:

- Windows 7/8/8.1/10
- Linux (kernel version 2.6.27 or newer)
- MacOS (version 10.8 or newer)

Python interpreter (version 3.6 or newer), NumPy, SciPy, Matplotlib, and rmsd packages need to be installed on the system for the CLI. Note that for users of Linux and RHEL an X server (https://www.x.org) is required for the GUI environment.

Functionalities

At this time, OctaDist comes with many built-in modules and functions listed in alphabetical order in Table 1. Each module holds a function for a specific operation. The modules can be grouped based on the type of task as follows:

Importing data. The molecule module supports a number of standard chemical file formats, such as XYZ, XMol, Mol, Mol2, and PDB, as well as the output from many computational chemistry programs including Gaussian,[37] Q-Chem,[38] ORCA,[39] and NWChem.[40] Once the input file has been loaded into OctaDist, an extract_coord associated with an extract_octa parsers will extract the atomic symbols and coordinates of the octahedral structure from the molecular input. Atomic symbols and coordinates will then be stored in a dictionary, which is a special data container in Python, as key-value pairs. In addition, OctaDist also allows users to modify the raw input file using the edit tool at any time if the molecular data is incorrectly processed.

Table 1 List of the modules in the OctaDist package

Module	Description
Calc	Computes distortion parameters
Draw	Displays molecular structures
Elements	Element properties, such as atomic radii
Linear	Built-in mathematical functions
Main	Main body of the program
Molecule	Manipulates atomic coordinates
Octadist_cli	Command line interface module
Octadist_gui	Graphical user interface module
Plane	Manipulates the plane
Plot	Plots graph and chart
Popup	Error, warning, and info messages
Projection	2D and 3D vector projections
Scripting	Interactive code console
Structure	All data about the structure
Tools	Third-party libraries
Util	Frequently-used utility functions

Calculating and reporting parameters. The functions related to calculating the distortion parameters are enclosed in a CalcDistortion class written in a calc module. In order to prevent confusion, the present section describes the functions used for calculating the octahedral distortion parameters. Firstly, OctaDist calls calc_zeta, which calculates ζ. This function first calls the calc_d_bond and calc_d_mean functions to determine the 6 unique metal–ligand bond lengths and average value. Then the results will be parsed to calc_zeta to calculate ζ. Secondly, a calc_sigma function will then call the calc_bond_angle function to determine the 12 *cis* bond angles around the central metal atom to calculate Σ. Thirdly, the calc_theta uses two functions: one is responsible for sorting and finding the 8 appropriate faces of the octahedron, the value derived from this calculation is an average one. Once the calculation is done, OctaDist will report the results on the text widget. These results can also be saved as external text-based files such as a TXT file.

Visualizing molecular structure. OctaDist can display the 3D structure of a molecule using the DrawComplex class developed in the MoleView package.[41] The DrawComplex class can be customized to show multiple structures simultaneously. OctaDist can also plot relationship graphs between computed Σ and Θ. The draw module in OctaDist allows the user to produce a high-resolution graphic of the molecular structure. It exploits the pyplot module, a collection of MATLAB-like command style functions in the Matplotlib package, offering several clear styles of drawing.[42] Due to its object-oriented nature, it is possible to define and customize the aspect of the display separately. For example, Fig. 8 shows a 3D structure of $[Fe(1\text{-bpp})_2][BF_2]_2$ in the low spin state and the eight faces of the octahedron.[43] Perspective and orthographic views are available, the former renders an image in depth, whereas the latter gives the same scale of atoms. For image preparation, both raster and vector file types such as JPEG, PNG, TIFF, PDF, or SVG are supported.

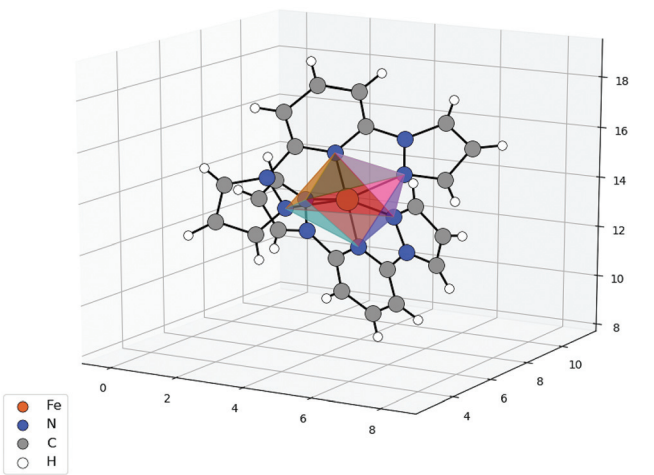

Fig. 8 3D molecular structure in ball and stick style, and the eight octahedral faces of $[Fe(1\text{-bpp})_2][BF_4]_2$.

Usage examples

The command line is a powerful tool for users and developers to find, create, and manipulate files and folders. One can use the command-line interface (CLI) to setup the OctaDist environment as well as the GUI does. This section walks users through the steps for setting up the CLI in OctaDist on a terminal based environment such as a Linux terminal. As the CLI comes in many forms, this tutorial will use the CLI called Bash. The Bash script is shown in Fig. S2 in the ESI† and shows how the CLI associated with OctaDist calculates the distortion parameters. OctaDist can be run on other Python platforms and services that run in the cloud such as Google Colab (https://colab.research.google.com), Jupyter Notebook (https://jupyter.org), and Jupyter Lab (https://jupyterlab.readthedocs.io), see Fig. S3.† These services offer a powerful engine and pre-installed required modules for executing Python code with interactive execution of commands in kernels.

To get started with OctaDist, a complete beginner's tutorial is available at https://octadist.github.io/tutorial. In the guide, users will learn how to install, compile, and use OctaDist. In addition, we provide a playground in which users can try OctaDist in Google Colab's interactive Python notebook. In addition, we also provide a complete guide for developing and implementing third-party functions and modules into OctaDist.

Extensions

Angular Jahn–Teller distortions. Additional tools for structural analysis have been implemented into OctaDist. This includes calculation of an angular Jahn–Teller distortion parameter, α, written in a CalcJahnTeller class in a tools module. The parameter α is the angle between the two 3D least-squared planes to the ligand atoms. The user can define the plane of interest by selecting a collection of atoms (chunk) such that the program can then compute the parameter α. Note that greater accuracy is achieved as more atoms are selected. Fig. 9 shows the two best fit planes to the chunk of the selected ligand atoms of $[FeL_2]^{2+}$ complex, where L = 2,6-di(pyrazol-1-yl)pyridine.[43]

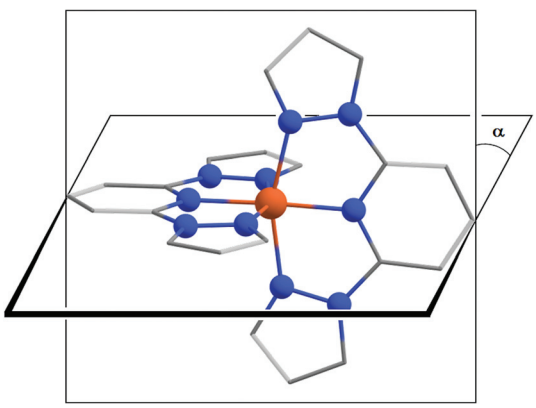

Fig. 9 Jahn–Teller distortion angle in a $[Fe(1\text{-bpp})_2]^{2+}$ complex.

Root-mean-square deviation (RMSD). The second tool is a root-mean-square deviation (RMSD) calculator, which is used to compare two sets of Cartesian coordinates of two complexes and quantify their minimal deviation of atomic positions. It is implemented using the rmsd package developed by Kromann.[44] It makes use of the optimize function in the SciPy package in combination with the Quaternion and the Kabsch algorithms to minimize the squared distance between two identical atomic positions.[45,46]

Scripting language. OctaDist has a built-in versatile scripting language written in the scripting module that allows the user to directly communicate with the program engine permitting variables and commands to be created, customized, and removed without affecting the other functions. Fig. S4 in the ESI† shows an example of the scripting language in assigning a new value of 1.0 to the cutoff_hydrogen variable which stores the hydrogen bond length cut-off.

Documentation

The documentation for the current version of OctaDist has been created from the reStructuredText source and docstring written in the source code of OctaDist. They have been generated to HTML, PDF, and EPUB outputs by Sphinx (https://www.sphinx-doc.org/en/master/). The API documentation for users and developers is automatically versioned and hosted by the Read The Docs system (https://octadist.readthedocs.io). The user manual contains useful information about the program, such as installation, normal usage, and a tutorial on how to use OctaDist.

Availability

OctaDist is distributed under the GNU General Public License version 3.0, on top of the Python license and the BSD license of Tcl/Tk.[47] The program source codes are maintained on the GitHub version control system (https://github.com/OctaDist). Not only do we work on code and issues there, but we also publish our roadmap and iteration plans. In addition, OctaDist aims to be an extensional choice for developers who need an easy-to-use program for determining molecular distortion. OctaDist can be embedded as a module for further development of other scientific software provided that proper attribution is given.

Conclusions

We present a python-based program OctaDist as a tool for determining octahedral distortion parameters. OctaDist represents an easy and consistent way to calculate octahedral distortion parameters that will serve not only spin crossover researchers, for which it is initially designed, but also a much wider scientific community, including inorganic chemists and biochemists. OctaDist offers scripting language where the user can interactively communicate with the code. We also pave the way for estimating Θ using a vector projection-based method which addresses the problems encountered in highly distorted octahedral systems. We believe that our algorithm is robust and will help contribute to the discovery of reliable and universal structure–properties relationships, notably leading to the design of applicative materials.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

We thank the Thailand Research Fund (BRG6180008) for funding this research. We also thank the Franco-Thai Mobility Programme for traveling costs (PHC-SIAM 40746YH). The authors acknowledge all issues reported and suggestions from users for further development of OctaDist.

Notes and references

1 O. I. Kucheriv, V. V. Oliynyk, V. V. Zagorodnii, V. L. Launets and I. A. Gural'Skiy, *Sci. Rep.*, 2016, **6**, 1–7.
2 C. M. Jureschi, J. Linares, A. Boulmaali, P. R. Dahoo, A. Rotaru and Y. Garcia, *Sensors*, 2016, **16**, 187.
3 T. Houari, E. Cuza, D. Pinkowicz, M. Marchivie, S. Yefsah and S. Triki, *Crystals*, 2018, **8**, 1–14.
4 L. Salmon and L. Catala, *C. R. Chim.*, 2018, **21**, 1230–1269.
5 R. W. Hogue, S. Singh and S. Brooker, *Chem. Soc. Rev.*, 2018, **47**, 7303–7338.
6 C. D. Polyzou and V. Tangoulis, *J. Coord. Chem.*, 2019, **72**, 389–418.
7 G. A. Craig, O. Roubeau and G. Aromí, *Coord. Chem. Rev.*, 2014, **269**, 13–31.
8 N. Bibi, E. G. R. De Arruda, A. Domingo, A. A. Oliveira, C. Galuppo, Q. M. Phung, N. M. Orra, F. Béron, A. Paesano, K. Pierloot and A. L. B. Formiga, *Inorg. Chem.*, 2018, **57**, 14603–14616.
9 P. Gütlich, A. B. Gaspar and Y. Garcia, *Beilstein J. Org. Chem.*, 2013, **9**, 342–391.
10 M. Halcrow, *Crystals*, 2016, **6**, 58.
11 P. Guionneau, M. Marchivie and G. Chastanet, *Chem. – Eur. J.*, 2020, DOI: 10.1002/chem.202002699.
12 G. Chastanet, C. Desplanches, C. Baldé, P. Rosa, M. Marchivie and P. Guionneau, *Chemistry Squared*, 2018, **2**, 2.
13 E. König, G. Ritter and B. Kanellakopulos, *J. Phys. C: Solid State Phys.*, 1974, **7**, 2681–2690.
14 E. König, in *Progress in Inorganic Chemistry*, John Wiley & Sons, Inc., 1987, vol. 35, pp. 527–622.
15 M. Buron-Le Cointe, J. Hébert, C. Baldé, N. Moisan, L. Toupet, P. Guionneau, J. F. Létard, E. Freysz, H. Cailleau and E. Collet, *Phys. Rev. B: Condens. Matter Mater. Phys.*, 2012, **85**, 39–41.
16 M. G. B. Drew, C. J. Harding, V. McKee, G. G. Morgan and J. Nelson, *J. Chem. Soc., Chem. Commun.*, 1995, 1035–1038.

17 P. Guionneau, C. Brigouleix, Y. Barrans, A. E. Goeta, J. F. Létard, J. A. K. Howard, J. Gaultier and D. Chasseau, *C. R. l'Acad. des Sci. – Ser. IIC Chem.*, 2001, **4**, 161–171.
18 M. Marchivie, P. Guionneau, J. F. Létard and D. Chasseau, *Acta Crystallogr., Sect. B: Struct. Sci.*, 2005, **61**, 25–28.
19 E. Collet and P. Guionneau, *C. R. Chim.*, 2018, **21**, 1133–1151.
20 H. S. Ren, M. J. Ming, J. Y. Ma and X. Y. Li, *J. Phys. Chem. A*, 2013, **117**, 8017–8025.
21 C. F. MacRae, I. Sovago, S. J. Cottrell, P. T. A. Galek, P. McCabe, E. Pidcock, M. Platings, G. P. Shields, J. S. Stevens, M. Towler and P. A. Wood, *J. Appl. Crystallogr.*, 2020, **53**, 226–235.
22 M. Pinsky and D. Avnir, *Inorg. Chem.*, 1998, **37**, 5575–5582.
23 D. Casanova, J. Cirera, M. Llunell, P. Alemany, D. Avnir and S. Alvarez, *J. Am. Chem. Soc.*, 2004, **126**, 1755–1763.
24 G. M. Sheldrick, *Acta Crystallogr., Sect. C: Struct. Chem.*, 2015, **71**, 3–8.
25 O. V. Dolomanov, L. J. Bourhis, R. J. Gildea, J. A. K. Howard and H. Puschmann, *J. Appl. Crystallogr.*, 2009, **42**, 339–342.
26 ChemCraft, *Graphical Software for Visualization of Quantum Chemistry Computations*, https://www.chemcraftprog.com.
27 D. Sertphon, P. Harding, K. S. Murray, B. Moubaraki, N. F. Chilton, S. Hill, J. Marbey, H. Adams, C. G. Davies, G. N. L. Jameson and D. J. Harding, *Dalton Trans.*, 2018, **47**, 7118–7122.
28 M. A. Halcrow, *Chem. Soc. Rev.*, 2011, **40**, 4119–4142.
29 D. J. Harding, P. Harding and W. Phonsri, *Coord. Chem. Rev.*, 2016, **313**, 38–61.
30 W. Phonsri, P. Harding, L. Liu, S. G. Telfer, K. S. Murray, B. Moubaraki, T. M. Ross, G. N. L. Jameson and D. J. Harding, *Chem. Sci.*, 2017, **8**, 3949–3959.
31 C. R. Groom, I. J. Bruno, M. P. Lightfoot and S. C. Ward, *Acta Crystallogr., Sect. B: Struct. Sci., Cryst. Eng. Mater.*, 2016, **72**, 171–179.
32 S. Alvarez, D. Avnir, M. Llunell and M. Pinsky, *New J. Chem.*, 2002, **26**, 996–1009.
33 M. Llunell, D. Casanova, J. Cirera, P. Alemany and S. Alvarez, *SHAPE: Program for the Stereochemical Analysis of Molecular Fragments by Means of Continuous Shape Measures and Associated Tools*, 2010.
34 G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*, CreateSpace, Scotts Valley, CA, 2009.
35 C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke and T. E. Oliphant, *Nature*, 2020, **585**, 357–362.
36 D. Cortesi, *Pyinstaller*, https://www.pyinstaller.org.
37 M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman and D. J. Fox, *Gaussian16 Revision C.01*, Gaussian Inc., Wallingford CT, 2016.
38 Y. Shao, Z. Gan, E. Epifanovsky, A. T. B. Gilbert, M. Wormit, J. Kussmann, A. W. Lange, A. Behn, J. Deng, X. Feng, D. Ghosh, M. Goldey, P. R. Horn, L. D. Jacobson, I. Kaliman, R. Z. Khaliullin, T. Kuś, A. Landau, J. Liu, E. I. Proynov, Y. M. Rhee, R. M. Richard, M. A. Rohrdanz, R. P. Steele, E. J. Sundstrom, H. L. Woodcock, P. M. Zimmerman, D. Zuev, B. Albrecht, E. Alguire, B. Austin, G. J. O. Beran, Y. A. Bernard, E. Berquist, K. Brandhorst, K. B. Bravaya, S. T. Brown, D. Casanova, C.-M. Chang, Y. Chen, S. H. Chien, K. D. Closser, D. L. Crittenden, M. Diedenhofen, R. A. DiStasio, H. Do, A. D. Dutoi, R. G. Edgar, S. Fatehi, L. Fusti-Molnar, A. Ghysels, A. Golubeva-Zadorozhnaya, J. Gomes, M. W. D. Hanson-Heine, P. H. P. Harbach, A. W. Hauser, E. G. Hohenstein, Z. C. Holden, T.-C. Jagau, H. Ji, B. Kaduk, K. Khistyaev, J. Kim, J. Kim, R. A. King, P. Klunzinger, D. Kosenkov, T. Kowalczyk, C. M. Krauter, K. U. Lao, A. D. Laurent, K. V. Lawler, S. V. Levchenko, C. Y. Lin, F. Liu, E. Livshits, R. C. Lochan, A. Luenser, P. Manohar, S. F. Manzer, S.-P. Mao, N. Mardirossian, A. V. Marenich, S. A. Maurer, N. J. Mayhall, E. Neuscamman, C. M. Oana, R. Olivares-Amaya, D. P. O'Neill, J. A. Parkhill, T. M. Perrine, R. Peverati, A. Prociuk, D. R. Rehn, E. Rosta, N. J. Russ, S. M. Sharada, S. Sharma, D. W. Small, A. Sodt, T. Stein, D. Stück, Y.-C. Su, A. J. W. Thom, T. Tsuchimochi, V. Vanovschi, L. Vogt, O. Vydrov, T. Wang, M. A. Watson, J. Wenzel, A. White, C. F. Williams, J. Yang, S. Yeganeh, S. R. Yost, Z.-Q. You, I. Y. Zhang, X. Zhang, Y. Zhao, B. R. Brooks, G. K. L. Chan, D. M. Chipman, C. J. Cramer, W. A. Goddard, M. S. Gordon, W. J. Hehre, A. Klamt, H. F. Schaefer, M. W. Schmidt, C. D. Sherrill, D. G. Truhlar, A. Warshel, X. Xu, A. Aspuru-Guzik, R. Baer, A. T. Bell, N. A. Besley, J.-D. Chai, A. Dreuw, B. D. Dunietz, T. R. Furlani, S. R. Gwaltney, C.-P. Hsu, Y. Jung, J. Kong, D. S. Lambrecht, W. Liang, C. Ochsenfeld, V. A. Rassolov, L. V. Slipchenko, J. E. Subotnik, T. Van Voorhis, J. M. Herbert, A. I. Krylov, P. M. W. Gill and M. Head-Gordon, *Mol. Phys.*, 2015, **113**, 184–215.
39 F. Neese, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2012, **2**, 73–78.
40 M. Valiev, E. J. Bylaska, N. Govind, K. Kowalski, T. P. Straatsma, H. J. J. Van Dam, D. Wang, J. Nieplocha, E. Apra, T. L. Windus and W. A. De Jong, *Comput. Phys. Commun.*, 2010, **181**, 1477–1489.

41 R. Ketkaew, *MoleView: A fast and lightweight plug-in for 3D molecular visualization*, 2020, https://github.com/moleview/moleview.
42 J. D. Hunter, *Comput. Sci. Eng.*, 2007, **9**, 90–95.
43 J. M. Holland, J. A. McAllister, Z. Lu, C. A. Kilner, M. Thornton-Pett and M. A. Halcrow, *Chem. Commun.*, 2001, 577–578.
44 J. C. Kromann, *Rmsd: Calculate Root-Mean-Square Deviation (RMSD) of Two Molecules Using Rotation*, https://github.com/charnley/rmsd.
45 P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt and SciPy 1.0 Contributors, *Nat. Methods*, 2020, **17**, 261–272.
46 W. Kabsch, *Acta Crystallogr., Sect. A: Cryst. Phys., Diffr., Theor. Gen. Crystallogr.*, 1976, **32**, 922–923.
47 GNU General Public License, http://www.gnu.org/licenses/gpl.html.