




cifkit: A Python package for coordination geometry and atomic site analysis

Sangjoon Lee ¹¶ and Anton O. Oliynyk ^{2,3}

1 Department of Applied Physics and Applied Mathematics, Columbia University, New York, NY 10027, USA **2** Department of Chemistry, Hunter College, City University of New York, New York, NY 10065, USA **3** Ph.D. Program in Chemistry, The Graduate Center of the City University of New York, New York, NY 10016, USA ¶ Corresponding author

DOI: [10.21105/joss.07205](https://doi.org/10.21105/joss.07205)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Rachel Kurchin](#)  

Reviewers:

- [@espottesmith](#)
- [@ml-evs](#)
- [@lancekavalsky](#)

Submitted: 30 August 2024

Published: 13 November 2024

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

cifkit provides higher-level functions and properties for coordination geometry and atomic site analysis from .cif files, which are standard file formats for storing crystallographic data such as atomic fractional coordinates, symmetry operations, and unit cell dimensions. Designed for functionalities demanded by experimental synthesists, cifkit has been used as a backend for Python applications that automate crystal structure analysis, enabling the extraction of physics-based features crucial for understanding geometric configurations and identifying irregularities. cifkit offers functions such as plotting a coordination geometry-based polyhedron from each site, calculating bond fractions, determining atomic mixing information, and sorting .cif files based on a set of attributes.

Statement of need

In solid-state chemistry and materials science, the Crystallographic Information File (CIF) ([Hall et al., 1991](#)) is the primary file format for storing and distributing crystal structure information. Open-source Python packages for reading, editing, and creating CIF files include Python Materials Genomics (pymatgen) ([Ong et al., 2013](#)) and the Atomic Simulation Environment (ASE) ([Larsen et al., 2017](#)). Pymatgen offers advanced functionalities such as generating electronic structure properties, phase diagrams, and implementing coordination environment identification through ChemEnv ([Waroquiers et al., 2020](#)). ASE provides a comprehensive suite of tools for generating and running atomistic simulations.

cifkit distinguishes itself from existing libraries by offering higher-level functions and variables that allow solid-state synthesists to obtain intuitive and measurable properties of interest. It facilitates the visualization of coordination geometry from each site using four coordination determination methods and extracts physics-based features like volume and packing efficiency, which are crucial for structural analysis in machine learning tasks. Moreover, cifkit extracts atomic mixing information at the bond pair level, tasks that would otherwise require extensive manual effort using GUI-based tools like VESTA, Diamond, and CrystalMaker. These functions can be further developed on-demand, as demonstrated by cifkit's ability to extract coordination geometry information based on four coordination number determination methods for a newly discovered phase ([Tyvanchuk et al., 2024](#)).

cifkit further enhances its utility by providing functions for sorting, preprocessing, and analyzing the distribution of underlying CIF files. It systematically addresses common issues in CIF files from databases, such as incorrect loop values and missing fractional coordinates, by standardizing and filtering out ill-formatted files. The package also preprocesses atomic site labels, transforming labels such as 'M1' into 'Fe1' in files with atomic mixing for enhanced

visualization and pattern matching. Beyond preprocessing, *cifkit* offers functionalities to copy, move, and sort files based on attributes such as coordination numbers, space groups, unit cells, and shortest distances. It excels in visualizing and cataloging CIF files, organizing them by supercell size, tags, coordination numbers, elements, and atomic mixing.

Examples

cifkit is designed to minimize reliance on API documentation for users with limited programming experience and no background in computational materials science or chemistry. By simplifying user interactions while maintaining robust functionality, *cifkit* enables a broader range of scientists to leverage computational tools for complex tasks—such as extracting geometry-based polyhedra descriptors from atomic sites. As shown in Figure 1, *cifkit* provides a higher-level function to visualize the atomic site coordination geometry from a single .cif file using the *Cif* object. It also provides an overview of multiple .cif files through the *CifEnsemble* object. The full installation process can be executed via a Jupyter notebook, accessible through the Google Colab URL provided in the official documentation.

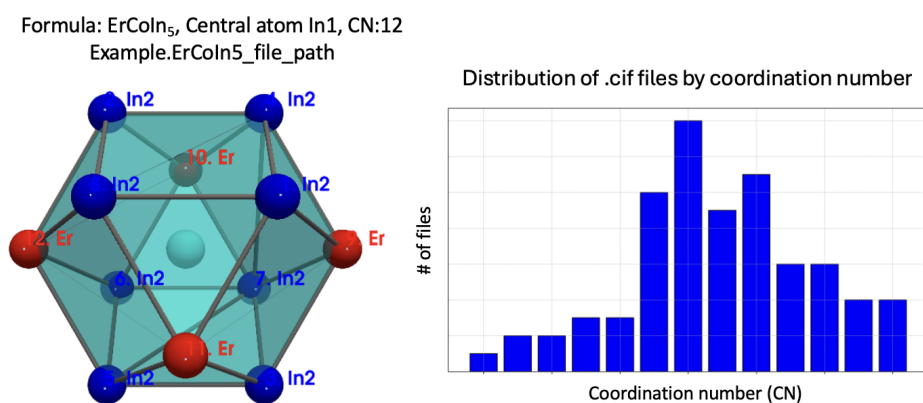


Figure 1: Atomic site coordination geometry from a single .cif file (left) and distribution of coordination numbers obtained from an ensemble of .cif files (right)

```
>>> from cifkit import Cif, Example

# Initialize with the .cif file path
>>> cif = Cif(Example.Er10Co9In20_file_path)

# Plot polyhedron from the site element of In1
>>> cif.plot_polyhedron("In1")

# Atomic mixing information
>>> cif.site_mixing_type # full occupancy, full_occupancy_atomic_mixing, etc.

# Determine coordination numbers based on four methods:
>>> cif.CN_max_gap_per_site
{
  "In1": {
    "dist_by_shortest_dist": {"max_gap": 0.306, "CN": 14},
    "dist_by_CIF_radius_sum": {"max_gap": 0.39, "CN": 14},
    "dist_by_CIF_radius_refined_sum": {"max_gap": 0.341, "CN": 12},
    "dist_by_Pauling_radius_sum": {"max_gap": 0.398, "CN": 14},
  },
}
```

```
...
    "Rh2": {
        "dist_by_shortest_dist": {"max_gap": 0.31, "CN": 9},
        "dist_by_CIF_radius_sum": {"max_gap": 0.324, "CN": 9},
        "dist_by_CIF_radius_refined_sum": {"max_gap": 0.397, "CN": 9},
        "dist_by_Pauling_radius_sum": {"max_gap": 0.380, "CN": 9},
    },
}
```

For processing a large number of .cif files, you may use CifEnsemble:

```
>>> from cifkit import CifEnsemble, Example

# Initialize with the folder path containing .cif files
>>> ensemble = CifEnsemble(Example.ErCoIn_big_folder_path)

# Filter .cif by formula(s)
>>> ensemble.filter_by_formulas(["LaRu2Ge2"])

# Filter .cif by site mixing type(s)
>>> ensemble.filter_by_site_mixing_types(["deficiency_without_atomic_mixing"])

# Filter .cif by coordination number(s)
>>> ensemble.filter_by_CN_min_dist_method_containing([14])
```

Applications

cifkit has been used for research conducted at academic and national laboratories for crystal structure analysis and machine learning studies. CIF Bond Analyzer (CBA) utilizes cifkit to extract coordination geometry information for a newly discovered phase (Tyvanchuk et al., 2024). The Structure Analysis/Featurizer (SAF) employs cifkit to construct and extract physics-based geometric features for binary and ternary compounds (Jaffal et al., 2024). Furthermore, geometric features generated with cifkit are being incorporated into a follow-up study on thermoelectric materials (Barua et al., 2024), building upon the compositional properties explored in (Lee et al., 2024).

Acknowledgement

We acknowledge the initial testing done by Nishant Yadav, Siddha Sankalpa Sethi, and Arnab Dutta from the Indian Institute of Technology, Kharagpur. We also thank Emil Jaffal, Danila Shiryaev, and Alex Vtorov from CUNY Hunter College for their testing efforts. We acknowledge Fabian Zills for his recommendations on Python tooling.

We thank the developers of the following dependencies:

- gemmi (Wojdyr, 2022): .cif parsing and space group operations
- matplotlib (Hunter, 2007): visualization of histograms
- numpy (Harris et al., 2020): angle conversion, linear algebra
- pyvista (Sullivan & Kaszynski, 2019): visualization of polyhedra
- scipy (Virtanen et al., 2020): minimization function to refine of CIF radius

References

Barua, N. K., Hall, E., Cheng, Y., Oliynyk, A. O., & Kleinke, H. (2024). Interpretable machine learning model on thermal conductivity using publicly available datasets and our internal

- lab dataset. *Chemistry of Materials*, 36(14), 7089–7100. <https://doi.org/10.1021/acs.chemmater.4c01696>
- Hall, S. R., Allen, F. H., & Brown, I. D. (1991). The crystallographic information file (CIF): A new standard archive file for crystallography. *Acta Crystallographica Section A*, 47(6), 655–685. <https://doi.org/10.1107/S010876739101067X>
- Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Jaffal, E., Lee, S., Shiryayev, D., Vtorov, A., Barua, N., Kleinke, H., & Oliynyk, A. (2024). *Composition and structure analyzer/featurizer for explainable machine-learning models to predict solid state structures*. ChemRxiv. <https://doi.org/10.26434/chemrxiv-2024-rrbhc>
- Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Duřak, M., Friis, J., Groves, M. N., Hammer, B., Hargus, C., Hermes, E. D., Jennings, P. C., Jensen, P. B., Kermode, J., Kitchin, J. R., Kolsbjerg, E. L., Kubal, J., Kaasbjerg, K., Lysgaard, S., ... Jacobsen, K. W. (2017). The atomic simulation environment—a python library for working with atoms. *Journal of Physics: Condensed Matter*, 29(27), 273002. <https://doi.org/10.1088/1361-648X/aa680e>
- Lee, S., Chen, C., Garcia, G., & Oliynyk, A. (2024). Machine learning descriptors in materials chemistry used in multiple experimentally validated studies: Oliynyk elemental property dataset. *Data in Brief*, 53. <https://doi.org/10.1016/j.dib.2024.110178>
- 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). Python materials genomics (pymatgen): A robust, open-source python library for materials analysis. *Computational Materials Science*, 68, 314–319. <https://doi.org/10.1016/j.commatsci.2012.10.028>
- Sullivan, C. B., & Kaszynski, A. A. (2019). PyVista: 3D plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK). *Journal of Open Source Software*, 4(37), 1450. <https://doi.org/10.21105/joss.01450>
- Tyvanchuk, Y., Babizhetskyy, V., Baran, S., Szytuła, A., Smetana, V., Lee, S., Oliynyk, A. O., & Mudring, A.-V. (2024). The crystal and electronic structure of $RE_{23}Co_6.7In_{20.3}$ ($RE = Gd-Tm, Lu$): A new structure type based on intergrowth of AlB_2 - and $CsCl$ -type related slabs. *Journal of Alloys and Compounds*, 976, 173241. <https://doi.org/10.1016/j.jallcom.2023.173241>
- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S. J. van der, Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... Mulbregt, P. van. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3), 261–272. <https://doi.org/10.1038/s41592-019-0686-2>
- Waroquiers, D., George, J., Horton, M., Schenk, S., Persson, K. A., Rignanese, G.-M., Gonze, X., & Hautier, G. (2020). ChemEnv: A fast and robust coordination environment identification tool. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 76(4), 683–695. <https://doi.org/10.1107/S2052520620007994>
- Wojdyr, M. (2022). GEMMI: A library for structural biology. *Journal of Open Source Software*, 7(73), 4200. <https://doi.org/10.21105/joss.04200>