The Journal of Open Source Software

IFermi: A python library for Fermi surface generation and analysis

Alex M Ganose[*1], **Amy Searle**[†2, 3, 4], **Anubhav Jain**[1], **and Sinéad M Griffin**[2, 3]

1 Energy Technologies Area, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA **2** Materials Science Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA **3** Molecular Foundry, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA **4** Clarendon Laboratory, Department of Physics, University of Oxford, OX1 3PU, UK

DOI: 10.21105/joss.03089

Software

- Review
- Repository
- Archive

Editor: Daniel S. Katz

Reviewers:

- @arosen93
- @lucydot

Submitted: 04 March 2021
Published: 17 March 2021

Summary

The Fermi surface is an important tool for understanding the electronic, optical, and magnetic properties of metals and doped semiconductors (Dugdale, 2016). It defines the surface in reciprocal space that divides unoccupied and occupied states at zero temperature. The topology of the Fermi surface impacts a variety of quantum phenomena including superconductivity, topological insulation, and ferromagnetism, and it can be used to predict the complex behaviour of systems without requiring more detailed computations. For example: (i) large nested Fermi sheets are a characteristic of charge density ordering (Lomer, 1962); (ii) the size and position of Fermi pockets are indicators of high-performance thermoelectrics (Park et al., 2020); and (iii) the average group velocities across the Fermi surface control the sensitivity of materials for dark matter detection (Inzani et al., 2021). IFermi is a Python library for the generation, analysis, and visualisation of Fermi surfaces that can facilitate sophisticated analyses of Fermi surface properties.

Statement of need

Many tools already exist for the generation of Fermi surfaces from *ab initio* band structure calculations. For example, several electronic structure codes such as CASTEP (Segall et al., 2002) and QuantumATK (Smidstrup et al., 2020) include integrated tools for obtaining Fermi surfaces. Furthermore, software such as FermiSurfer (Kawamura, 2019), pyprocar (Herath et al., 2020), BoltzTraP2 (Madsen et al., 2018), and XCrysDen (Kokalj, 1999) interface with common density functional theory packages and can plot Fermi surfaces from their outputs. All such packages, however, are only designed to visualise Fermi surfaces and do not expose any application programming interfaces (APIs) for analysing and manipulating Fermi surfaces as objects. In IFermi, we address these limitations by developing a fully-featured Python library for representing and processing Fermi surfaces and Fermi slices. We also implement a novel algorithm for determining the dimensionality of Fermi surfaces using the connectivity across periodic boundaries and the Euler characteristic of the isosurface mesh. IFermi, therefore, enables the programmatic characterisation of Fermi surfaces and can be used as a foundational library for investigating complex Fermi surface properties such as nesting.

[*] equal contribution
[†] equal contribution

IFermi

IFermi is a Python 3.6+ library and set of command-line tools for the generation, analysis, and visualisation of Fermi surfaces and Fermi slices. The goal of the library is to provide fully-featured `FermiSurface` and `FermiSlice` objects that allow for easy manipulation and programmatic analysis. The main features of the package include:

- The Fourier interpolation of electronic band structures onto dense k-point meshes required to obtain high resolution Fermi surfaces.
- The extraction of Fermi surfaces and Fermi slices from electronic band structures.
- A rich API for representing and manipulating Fermi surface objects.
- The projection of arbitrary properties onto Fermi surfaces and Fermi slices.
- Algorithms to calculate Fermi surface properties, including dimensionality, orientation, and averaged projections such as Fermi velocities.
- Interactive visualisation of Fermi surfaces and slices, and their projections such as spin-texture, with support for mayavi (Ramachandran & Varoquaux, 2011), plotly (Plotly Technologies Inc., 2021) and matplotlib (Hunter, 2007). Examples of the graphics produced by IFermi are presented in Figure 1.

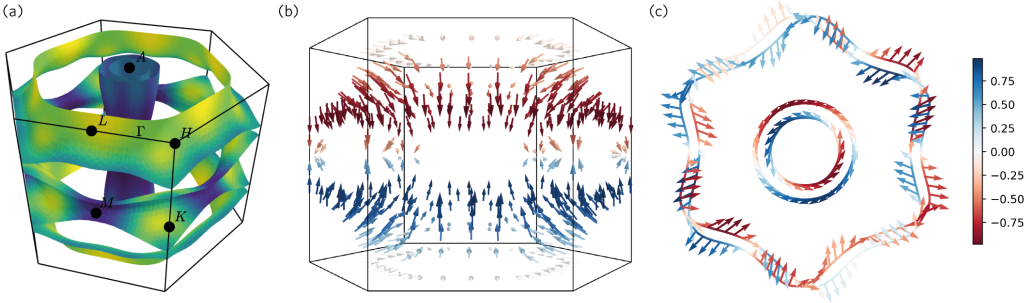

Figure 1: Examples of Fermi surfaces and two-dimensional slices produced by IFermi. Fermi surface of MgB_2 with group velocity projections shown by (a) the isosurface color and (b) arrows colored by the scalar projection onto the [0 0 1] axis. (c) Spin texture of BiSb indicating Rashba splitting.

In addition to the Python library, IFermi includes several command-line tools that can perform common tasks such as calculating Fermi surface dimensionality and Fermi velocities. IFermi uses the pymatgen (Ong et al., 2013) library for parsing first-principles calculation outputs and therefore supports all electronic structure codes supported therein. At the time of writing this comprises Vienna *ab initio* Simulation Package (VASP), ABINIT, and CP2K. IFermi also relies on several open source packages, such as BoltzTraP2 (Madsen et al., 2018) for Fourier interpolation, trimesh (Dawson-Haggerty, n.d.) for processing triangular meshes, and scikit-image (van der Walt et al., 2014) for generating isosurfaces using the marching cubes algorithm developed by Lewiner et al. (2003).

Author Contributions

The library and command-line tools were written by AS and AMG. SMG designed and led the project. The first draft of the manuscript was written by AMG with input from all co-authors.

Conflicts of Interest

There are no conflicts to declare.

Acknowledgements

This work was funded by the DOE Basic Energy Sciences program – the Materials Project – under Grant No. KC23MP. Computational resources were provided by the National Energy Research Scientific Computing Center and the Molecular Foundry, DoE Office of Science User Facilities supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. The work performed at the Molecular Foundry was supported by the Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under the same contract. We would like to acknowledge additional suggestions and code contributions from Matthew Horton and Mariano Forti. We would like to acknowledge many fruitful discussions with Katherine Inzani.

References

Dawson-Haggerty, M. (n.d.). *Trimesh* (Version 3.2.0) [Computer software]. https://trimsh.org/

Dugdale, S. B. (2016). Life on the edge: A beginner's guide to the Fermi surface. *Phys. Scr.*, *91*(5), 053009. https://doi.org/10.1088/0031-8949/91/5/053009

Herath, U., Tavadze, P., He, X., Bousquet, E., Singh, S., Muñoz, F., & Romero, A. H. (2020). PyProcar: A Python library for electronic structure pre/post-processing. *Comput. Phys. Commun.*, *251*, 107080. https://doi.org/10.1016/j.cpc.2019.107080

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Comput. Sci. Eng.*, *9*(3), 90–95. https://doi.org/10.1109/mcse.2007.55

Inzani, K., Faghaninia, A., & Griffin, S. M. (2021). Prediction of tunable spin-orbit gapped materials for dark matter detection. *Phys. Rev. Research*, *3*(1), 013069. https://doi.org/10.1103/PhysRevResearch.3.013069

Kawamura, M. (2019). FermiSurfer: Fermi-surface viewer providing multiple representation schemes. *Comput. Phys. Commun.*, *239*, 197–203. https://doi.org/10.1016/j.cpc.2019.01.017

Kokalj, A. (1999). XCrySDen—a new program for displaying crystalline structures and electron densities. *J. Mol. Graph. Model.*, *17*(3-4), 176–179. https://doi.org/10.1016/s1093-3263(99)00028-5

Lewiner, T., Lopes, H., Vieira, A. W., & Tavares, G. (2003). Efficient Implementation of Marching Cubes' Cases with Topological Guarantees. *J. Graph. Tools*, *8*(2), 1–15. https://doi.org/10.1080/10867651.2003.10487582

Lomer, W. M. (1962). Electronic Structure of Chromium Group Metals. *Proc. Phys. Soc.*, *80*(2), 489–496. https://doi.org/10.1088/0370-1328/80/2/316

Madsen, G. K. H., Carrete, J., & Verstraete, M. J. (2018). BoltzTraP2, a program for interpolating band structures and calculating semi-classical transport coefficients. *Comput. Phys. Commun.*, *231*, 140–145. https://doi.org/10.1016/j.cpc.2018.05.010

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. *Comp. Mater. Sci.*, *68*(C), 314–319. https://doi.org/10.1016/j.commatsci.2012.10.028

Park, J., Xia, Y., Ganose, A. M., Jain, A., & Ozoliņš, V. (2020). High Thermoelectric Performance and Defect Energetics of Multipocketed Full Heusler Compounds. *Phys. Rev. Applied*, *14*(2), 024064. https://doi.org/10.1103/PhysRevApplied.14.024064

Plotly Technologies Inc. (2021). *Collaborative data science*. Plotly Technologies Inc. https://plot.ly

Ramachandran, P., & Varoquaux, G. (2011). Mayavi: 3D Visualization of Scientific Data. *Comput. Sci. Eng.*, *13*(2), 40–51. https://doi.org/10.1109/MCSE.2011.35

Segall, M. D., Lindan, P. J. D., Probert, M. J., Pickard, C. J., Hasnip, P. J., Clark, S. J., & Payne, M. C. (2002). First-principles simulation: Ideas, illustrations and the CASTEP code. *J. Phys.: Condens. Matter*, *14*(11), 2717–2744. https://doi.org/10.1088/0953-8984/14/11/301

Smidstrup, S., Markussen, T., Vancraeyveld, P., Wellendorff, J., Schneider, J., Gunst, T., Verstichel, B., Stradi, D., Khomyakov, P. A., Vej-Hansen, U. G., Lee, M.-E., Chill, S. T., Rasmussen, F., Penazzi, G., Corsetti, F., Ojanperä, A., Jensen, K., Palsgaard, M. L. N., Martinez, U., … Stokbro, K. (2020). QuantumATK: An integrated platform of electronic and atomic-scale modelling tools. *J. Phys.: Condens. Matter*, *32*(1), 015901. https://doi.org/10.1088/1361-648X/ab4007

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., & Yu, T. (2014). Scikit-image: Image processing in Python. *PeerJ*, *2*, e453. https://doi.org/10.7717/peerj.453