

matador: a Python library for analysing, curating and performing high-throughput density-functional theory calculations

Matthew L. Evans¹ and Andrew J. Morris²

¹ Theory of Condensed Matter Group, Cavendish Laboratory, University of Cambridge, J. J. Thomson Avenue, Cambridge, CB3 0HE, U.K. ² School of Metallurgy and Materials, University of Birmingham, Edgbaston, Birmingham, B15 2TT, U.K.

DOI: [10.21105/joss.02563](https://doi.org/10.21105/joss.02563)

Software

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

Editor: [Jeff Gostick](#) ↗

Reviewers:

- [@mkhorton](#)
- [@srmnitc](#)

Submitted: 27 July 2020

Published: 27 October 2020

License

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

Summary

The properties of materials depend heavily on their atomistic structure; knowledge of the possible stable atomic configurations that define a material is required to understand the performance of many technologically and ecologically relevant devices, such as those used for energy storage (A. F. Harper et al., 2020; Marbella et al., 2018). First-principles crystal structure prediction (CSP) is the art of finding these stable configurations using only quantum mechanics (A. F. Harper, Evans, et al., 2020). Density-functional theory (DFT) is a ubiquitous theoretical framework for finding approximate solutions to quantum mechanics; calculations using a modern DFT package are sufficiently robust and accurate that insight into real materials can be readily obtained. The computationally intensive work is performed by well-established, low-level software packages, such as CASTEP (Clark et al., 2005) or Quantum Espresso (Giannozzi et al., 2009), which are able to make use of modern high-performance computers. In order to use these codes easily, reliably and reproducibly, many high-level libraries have been developed to create, curate and manipulate the calculations from these low-level workhorses; `matador` is one such framework.

Statement of need

The purpose of `matador` is fourfold:

- to promote the use of local databases and high-throughput workflows to increase the reproducibility of the computational results,
- to perform reliable analysis of the stability, structure and properties of materials derived from calculations,
- to provide tools to create customisable, publication-quality plots of phase diagrams, spectral properties and electrochemistry,
- to make the above functionality available to those with limited programming experience.

matador

`matador` is a Python 3.6+ library and set of command-line tools for performing and analysing high-throughput DFT calculations using the CASTEP (Clark et al., 2005) and Quantum

Espresso (Giannozzi et al., 2009) packages. It is well-tested and fully-documented at [ReadTheDocs](#), and comes with several tutorials and examples. The package is available on PyPI under the name `matador-db`. As with many projects, `matador` is built on top of the scientific Python ecosystem of NumPy (Harris et al., 2020), SciPy (Virtanen et al., 2020) and matplotlib (Hunter, 2007).

`matador` has been developed with high-throughput CSP in mind and has found use in the application of CSP to energy storage materials (Harper et al., 2020; Marbella et al., 2018); in this use case, a single compositional phase diagram can consist of tens of thousands of structural relaxation calculations. This package is aimed at users of CASTEP or Quantum Espresso who are comfortable with the command-line, yet maybe lack the Python knowledge required to start from scratch with more sophisticated packages. There are many mature packages that provide overlapping functionality with `matador`, the most widespread of which being the Atomic Simulation Environment (ASE) (Larsen et al., 2017) and pymatgen (Ong et al., 2013). A translation layer to and from the structure representation of both of these packages is provided, such that analysis can be reused and combined.

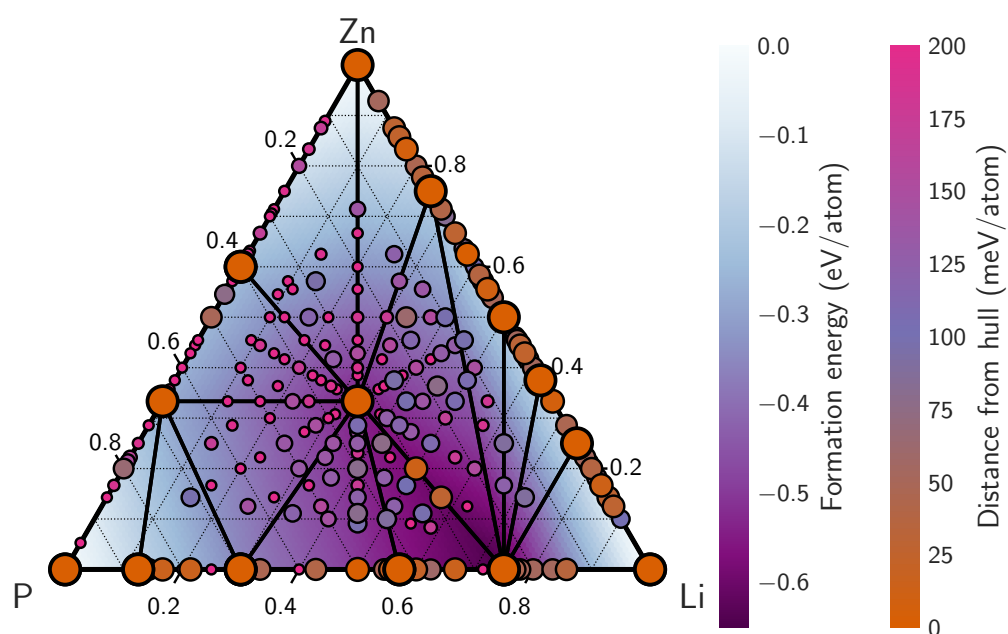


Figure 1: Li-Zn-P ternary phase diagram created with `matador`, plot generated with matplotlib (Hunter, 2007) and python-ternary (Harper & others, 2019).

Overview of functionality

There are two ways of working with `matador`, either from the command-line interface (CLI) or through the Python library directly, with some features that are unique to each. The functionality of `matador` can be broadly split into three categories:

1. *Creation and curation of databases of the results of first-principles calculations.*

`matador` allows for the creation of MongoDB databases of CASTEP (6.0+) geometry optimizations from the command-line, using `matador import`. Only calculations that are deemed “successful”, and that have fully-specified inputs are stored, with errors displayed for the rest. The resulting database can be queried with `matador query`, either with Python or through

the powerful CLI. The results can be filtered for structural “uniqueness” and written to one of several supported file types and exported for use in other frameworks, such as ASE or pymatgen. Prototyping of structures directly from the database is achieved using `matador swaps`, which uses the same interface as `matador query` to return structure files with “swapped” elements (Marbella et al., 2018).

2. High-throughput calculations and automated workflows.

The `run3` executable bundled with `matador` allows for high-throughput calculations to be performed with little setup and no programming knowledge. Specialised support for CASTEP and the post-processing tool OptaDOS (Morris, Nicholls, Pickard, & Yates, 2014; Nicholls, Morris, Pickard, & Yates, 2012) is provided to perform high-throughput geometry optimisations, orbital-projected band structures and densities of states, phonon calculations and elastic properties, however `run3` can also be used to run generic MPI programs concurrently on a set of structures. Sensible defaults for these workflows are provided by leveraging the open-source SeeK-path (Hinuma, Pizzi, Kumagai, Oba, & Tanaka, 2017) and `spglib` (Togo & Tanaka, 2018) libraries. The bundled `dispersion` script and associated library functionality allows for the creation of publication-quality spectral and vibrational property plots, in a similar fashion to the `sumo` package (Ganose, Jackson, & Scanlon, 2018). The `matador.compute` module behind `run3` also powers the `ilustrado` genetic algorithm code (Evans, 2020).

3. Stability and structural analysis (with an emphasis on battery materials).

The construction of reliable compositional phase diagrams requires several independent calculations to be performed on different atomic configurations with a compatible set of external parameters. These can be generated from a database query using `matador hull`, which allows the user to filter between different sets of calculations, and, where relevant, `matador voltage` can provide the electrochemical properties of that same phase diagram. Structural fingerprints implemented include pair distribution functions, powder X-ray diffraction patterns, and periodic crystal bond graphs. As more calculations are performed, changes to phase diagrams stored in the local database can be tracked with `matador hulldiff`. Phase diagrams can also be constructed from multiple energy values per structure, for example to show the effects of finite temperature (Harper et al., 2020), or in the specific case of ensemble-based exchange-correlation functionals like the Bayesian Error Estimate Functional (BEEF) (Mortensen et al., 2005). An example of a ternary phase diagram is shown Figure 1.

Acknowledgements

We acknowledge all the contributors, users and testers of this package, primarily Angela Harper, James Darby, Jordan Dorrell and Matthew Cliffe. M.E. would like to acknowledge the EPSRC Centre for Doctoral Training in Computational Methods for Materials Science for funding under grant number EP/L015552/1. A.J.M. acknowledges funding from EPSRC (EP/P003532/1). The authors acknowledge networking support via the EPSRC Collaborative Computational Projects, CCP9 (EP/M022595/1) and CCP-NC (EP/T026642/1). Much of the development and testing was performed on the Cambridge Service for Data Driven Discovery (CSD3) operated by the University of Cambridge Research Computing Service (<http://www.csd3.cam.ac.uk/>), provided by Dell EMC and Intel using Tier-2 funding from the Engineering and Physical Sciences Research Council, and DiRAC funding from the Science and Technology Facilities Council (www.dirac.ac.uk).

Clark, S. J., Segall, M. D., Pickard, C. J., Hasnip, P. J., Probert, M. I. J., Refson, K., & Payne, M. C. (2005). First principles methods using CASTEP. *Zeitschrift für Kristallographie - Crystalline Materials*, 220(5/6), 567–570. doi:10.1524/zkri.220.5.567.65075

- Evans, M. L. (2020, June). ml-evs/ilustrado: v0.3. Zenodo. doi:[10.5281/zenodo.3904495](https://doi.org/10.5281/zenodo.3904495)
- Ganose, A. M., Jackson, A. J., & Scanlon, D. O. (2018). Sumo: Command-line tools for plotting and analysis of periodic *ab initio* calculations. *Journal of Open Source Software*, 3(28), 717. doi:[10.21105/joss.00717](https://doi.org/10.21105/joss.00717)
- Giannozzi, P., Baroni, S., Bonini, N., Calandra, M., Car, R., Cavazzoni, C., Ceresoli, D., et al. (2009). QUANTUM ESPRESSO: A modular and open-source software project for quantum simulations of materials. *J. Phys.: Condens. Matter*, 21(39), 395502. doi:[10.1088/0953-8984/21/39/395502](https://doi.org/10.1088/0953-8984/21/39/395502)
- Harper, A. F., Evans, M. L., Darby, J. P., Karasulu, B., Koçer, C. P., Nelson, J. R., & Morris, A. J. (2020). Ab initio Structure Prediction Methods for Battery Materials : A review of recent computational efforts to predict the atomic level structure and bonding in materials for rechargeable batteries. *Johnson Matthey Technology Review*, 64(2), 103–118. doi:[10.1595/205651320X15742491027978](https://doi.org/10.1595/205651320X15742491027978)
- Harper, A. F., Evans, M. L., & Morris, A. J. (2020). Novel Phases of Copper Phosphides from Computational Structure Searches. *Chemistry of Materials*, 32(15). doi:[10.1021/acs.chemmater.0c02054](https://doi.org/10.1021/acs.chemmater.0c02054)
- Harper, M., & others. (2019, April). marcharper/python-ternary: Version 1.0.6. Zenodo. doi:[10.5281/zenodo.2628066](https://doi.org/10.5281/zenodo.2628066)
- Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., et al. (2020). Array programming with NumPy. *Nature*, 585, 357–362. doi:[10.1038/s41586-020-2649-2](https://doi.org/10.1038/s41586-020-2649-2)
- Hinuma, Y., Pizzi, G., Kumagai, Y., Oba, F., & Tanaka, I. (2017). Band structure diagram paths based on crystallography. *Computational Materials Science*, 128, 140–184. doi:[10.1016/j.commatsci.2016.10.015](https://doi.org/10.1016/j.commatsci.2016.10.015)
- Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science Engineering*, 9(3), 90–95. doi:[10.1109/MCSE.2007.55](https://doi.org/10.1109/MCSE.2007.55)
- Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Duřak, M., Friis, J., et al. (2017). The atomic simulation environment—a Python library for working with atoms. *J. Phys.: Condens. Matter*, 29(27), 273002. doi:[10.1088/1361-648x/aa680e](https://doi.org/10.1088/1361-648x/aa680e)
- Marbella, L. E., Evans, M. L., Groh, M. F., Nelson, J., Griffith, K. J., Morris, A. J., & Grey, C. P. (2018). Sodiation and Desodiation via Helical Phosphorus Intermediates in High-Capacity Anodes for Sodium-Ion Batteries. *Journal of the American Chemical Society*, 140(25), 7994–8004. doi:[10.1021/jacs.8b04183](https://doi.org/10.1021/jacs.8b04183)
- Morris, A. J., Nicholls, R. J., Pickard, C. J., & Yates, J. R. (2014). OptaDOS: A tool for obtaining density of states, core-level and optical spectra from electronic structure codes. *Computer Physics Communications*, 185(5), 1477–1485. doi:[10.1016/j.cpc.2014.02.013](https://doi.org/10.1016/j.cpc.2014.02.013)
- Mortensen, J. J., Kaasbjerg, K., Frederiksen, S. L., Nørskov, J. K., Sethna, J. P., & Jacobsen, K. W. (2005). Bayesian Error Estimation in Density-Functional Theory. *Physical Review Letters*, 95(21). doi:[10.1103/PhysRevLett.95.216401](https://doi.org/10.1103/PhysRevLett.95.216401)
- Nicholls, R. J., Morris, A. J., Pickard, C. J., & Yates, J. R. (2012). OptaDOS - a new tool for EELS calculations. *J. Phys.: Conf. Ser.*, 371, 012062. doi:[10.1088/1742-6596/371/1/012062](https://doi.org/10.1088/1742-6596/371/1/012062)
- Ong, S. P., Richards, W. D., Jain, A., Hautier, G., Kocher, M., Cholia, S., Gunter, D., et al. (2013). Python Materials Genomics (pymatgen): A robust, open-source python library for materials analysis. *Computational Materials Science*, 68, 314–319. doi:[10.1016/j.commatsci.2012.10.028](https://doi.org/10.1016/j.commatsci.2012.10.028)
- Togo, A., & Tanaka, I. (2018). Spglib: A software library for crystal symmetry search. *arXiv:1808.01590 [cond-mat]*. Retrieved from <http://arxiv.org/abs/1808.01590>

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., et al. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3), 261–272. doi:[10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2)