The Journal of Open Source Software

ASIMTools: A lightweight framework for scalable and reproducible atomic simulations

Mgcini Keith Phuthi [1], **Emil Annevelink** [2], **and Venkatasubramanian Viswanathan** [1]

1 University of Michigan, United States **2** Carnegie Mellon University, United States

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Atomic SIMulation Tools (`ASIMTools`) is a lightweight workflow and simulation manager for reproducible atomistic simulations on Unix-based systems. Within the framework, simulations can be transferred across computing environments, DFT codes, interatomic potentials and atomic structures. By using in-built or user-defined python modules (called asimmodules) and utilities, users can run simulation protocols and automatically scale them on slurm based clusters or locally on their console. The core idea is to separate the dependence of the atomistic potential/calculator, the computing environment and the simulation protocol thereby allowing the same simulation to be run with different calculators, atomic structures or on different computers with just a change of one parameter in an input file after initial setup. This is increasingly necessary as benchmarking Machine Learning Interatomic Potentials has become a core part of computational materials science. Input and output files follow a simple standard format, usually yaml, providing a simple interface that also acts as a record of the parameters used in a simulation without having to edit python scripts. The minimal set of requirements means any materials science codes can be incorporated into an ASIMTools workflow in a unified way.

Statement of need

Atomic simulations are a key component of modern day materials science in both academia and industry. However, simulation protocols and workflows used by researchers are typically difficult to transfer to systems using different inputs, codes and computing environments. It often involves rewriting entire scripts in different languages to change from one type of atomistic potential or atomic structure to another. This leads to poor reproducibility and inefficient transfer of code from one researcher to the next. In addition, there exists a zoo of tools and packages for atomic simulation with more being developed every day (Walsh, 2024). There is however no unifying framework that can encompass all these tools without significant software development effort. Significant effort should not be necessary because while the source of the fundamental outputs of atomistic potentials such as energy, forces etc. may differ, simulation protocols built on these outputs should converge towards the most accurate and computationally efficient. ASIMTools focuses on this last aspect by introducing asimmodules which are simply Python functions that act as simulation protocols which have no dependence on a specific atomistic potential or computational environment or atomic structure. Through iteration and community input, these simulation protocols will hopefully converge towards best practice and ensure reproducibility of simulation results.

`ASIMTools` is for users interested in performing atomistic calculations on UNIX-like operating systems and/or on slurm-based High Performance Computing clusters. By defining simulation protocols as "asimmodules", they can be easily added to the library of provided asimmodules

and iterated on. The flexibility of ASIMTools allows integration of any kind of simulation tools such as the heavily used Atomic Simulation Environment (Larsen et al., 2017) pymatgen (Ong et al., 2013), LAMMPS (Thompson et al., 2022) etc. with examples provided. With the asimmodules defined, users only need to provide a set of inputs in the form of yaml files that define the parameters used for each simulation and are therefore a concrete record of used parameters.

State of the Field

There exist a number of popular workflow tools for atomistic simulations such as Aiida (Huber et al., 2020), Fireworks (Jain et al., 2015) and many more. These tools provide frameworks for constructing complex workflows with different underlying principles. Some managers enforce strict rules that ensure that data obeys FAIR principles and emphasizes data provenance and reproducibility. These methods however tend to be fairly large packages with steep learning curves. ASIMTools provides a simple interface as a starting point that can transform any code into ASIMTools compatible code by simply wrapping it in a function that returns a Python dictionary. Any such code can work in ASIMTools and with a few extra steps, the protocol can be made to support an arbitrary calculator and input atomic structure.

In some workflow managers, such as Atomic Simulation Recipes (Gjerding et al., 2021), once workflows are built, it can often be difficult to quickly change and iterate over key parameters such as the choice of atomistic calculator or structure as they are intrinsically built into the code. This is particularly challenging in an age where machine learning models are becoming more popular. Workflows involving machine learning interatomic potentials tend to require the ability to repeat the same calculations on different examples, using different calculators on different hardware iteratively. This is where the value of ASIMTools lies in contrast to more established workflows. ASIMTools is not designed to replace the more powerful workflow managers but rather to supplement them. This is achieved by providing unified inputs that can be easily integrated into, for example, Aiida as Python functions/asimmodules while also being a stand-alone lightweight workflow manager for simpler cases.

Usage To-Date

ASIMTools has been used in the benchmarking Machine Learning Interatomic Potentials (Phuthi, Yao, et al., 2024) and creating a workflow for calculation of vibrational properties of solids calculations (Phuthi, Huang, et al., 2024).

Conclusion and Availability

The ASIMTools package is a powerful tool for building and executing atomic simulation protocols locally and at scale on slurm-based HPC infrastructure. The code is hosted on a public Github repository (https://github.com/BattModels/asimtools) with a number of examples. Asimmodules for common calculations are also implemeted with examples. Interested users are encouraged to submit issues, contact developers and make pull requests, particularly for adding new simulation protocols to the library.

Author Contribution Statement

Conceptualization by Keith Phuthi. Coding and development by Keith Phuthi and Emil Annevelink. Paper writing by Keith Phuthi. Project management by all.

The Journal of Open Source Software

Acknowledgements

We acknowledge feedback from Kian Pu, Lance Kavalsky, Hancheng Zhao and Ziqi Wang.

References

Gjerding, M., Skovhus, T., Rasmussen, A., Bertoldo, F., Larsen, A. H., Mortensen, J. J., & Thygesen, K. S. (2021). Atomic Simulation Recipes: A Python framework and library for automated workflows. *Computational Materials Science*, *199*, 110731. https://doi.org/10.1016/j.commatsci.2021.110731

Huber, S. P., Zoupanos, S., Uhrin, M., Talirz, L., Kahle, L., Häuselmann, R., Gresch, D., Müller, T., Yakutovich, A. V., Andersen, C. W., Ramirez, F. F., Adorf, C. S., Gargiulo, F., Kumbhar, S., Passaro, E., Johnston, C., Merkys, A., Cepellotti, A., Mounet, N., … Pizzi, G. (2020). AiiDA 1.0, a scalable computational infrastructure for automated reproducible workflows and data provenance. *Scientific Data*, *7*(1), 300. https://doi.org/10.1038/s41597-020-00638-4

Jain, A., Ong, S. P., Chen, W., Medasani, B., Qu, X., Kocher, M., Brafman, M., Petretto, G., Rignanese, G.-M., Hautier, G., Gunter, D., & Persson, K. A. (2015). FireWorks: A dynamic workflow system designed for high-throughput applications. *Concurrency and Computation: Practice and Experience*, *27*(17), 5037–5059. https://doi.org/10.1002/cpe.3505

Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Dułak, 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

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

Phuthi, M. K., Huang, Y., Widom, M., & Viswanathan, V. (2024). *Vibrational Entropy and Free Energy of Solid Lithium using Covariance of Atomic Displacements Enabled by Machine Learning*. arXiv. http://arxiv.org/abs/2406.15491

Phuthi, M. K., Yao, A. M., Batzner, S., Musaelian, A., Guan, P., Kozinsky, B., Cubuk, E. D., & Viswanathan, V. (2024). Accurate Surface and Finite-Temperature Bulk Properties of Lithium Metal at Large Scales Using Machine Learning Interaction Potentials. *ACS Omega*, *9*(9), 10904–10912. https://doi.org/10.1021/acsomega.3c10014

Thompson, A. P., Aktulga, H. M., Berger, R., Bolintineanu, D. S., Brown, W. M., Crozier, P. S., Veld, P. J. in 't, Kohlmeyer, A., Moore, S. G., Nguyen, T. D., Shan, R., Stevens, M. J., Tranchida, J., Trott, C., & Plimpton, S. J. (2022). LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Comp. Phys. Comm.*, *271*, 108171. https://doi.org/10.1016/j.cpc.2021.108171

Walsh, A. (2024). Open computational materials science. *Nature Materials*, *23*(1), 16–17. https://doi.org/10.1038/s41563-023-01699-7