Facilitating the selection and creation of accurate interatomic potentials with robust tools and characterization

Zachary T. Trautt

Materials Measurement Sciences Division, National Institute of Standards and Technology, Gaithersburg, MD, 20899 USA

E-mail: `zachary.trautt@nist.gov`

Francesca Tavazza

Materials Science and Engineering Division, National Institute of Standards and Technology, Gaithersburg, MD, 20899 USA

Chandler A. Becker

Materials Science and Engineering Division, National Institute of Standards and Technology, Gaithersburg, MD, 20899 USA

Abstract. The Materials Genome Initiative seeks to significantly decrease the cost and time of development and integration of new materials. Within the domain of atomistic simulations, several roadblocks stand in the way of reaching this goal. While the NIST Interatomic Potentials Repository hosts numerous interatomic potentials (force fields), researchers cannot immediately determine the best choice(s) for their use case. Researchers developing new potentials, specifically those in restricted environments, lack a comprehensive portfolio of efficient tools capable of calculating and archiving the properties of their potentials. This paper elucidates one solution to these problems, which uses Python-based scripts that are suitable for rapid property evaluation and human knowledge transfer. Calculation results are visible on the repository website, which reduces the time required to select an interatomic potential for a specific use case. Furthermore, property evaluation scripts are being integrated with modern platforms to improve discoverability and access of materials property data. To demonstrate these scripts and features, we will discuss the automation of stacking fault energy calculations and their application to additional elements. While the calculation methodology was developed previously, we are using it here as a case study in simulation automation and property calculations. We demonstrate how the use of Python scripts allows for rapid calculation in a more easily managed way where the calculations can be modified, and the results presented in user-friendly and concise ways. Additionally, the methods can be incorporated into other efforts, such as openKIM.

1. Introduction

A central goal of the Materials Genome Initiative (MGI) is to reduce the cost and time of development of new materials by fifty percent [1, 2, 3]. If this goal is to be achieved, improvements must be made at each stage of development, as a new material moves from discovery to commercial deployment. During materials discovery, classical atomistic simulation techniques, such as molecular dynamics and Monte Carlo, serve an important role. While quantum mechanical methods, such as density functional theory, offer greater accuracy, the approximation made by using a classical interatomic potential significantly increases the time and length scales accessible within a given computational constraint.

Since 2008, the Interatomic Potentials Repository (IPR) project at the National Institute of Standards and Technology (NIST) has served as a location for developer-approved interatomic potentials and related files to help researchers obtain interatomic models and judge their quality and applicability [4, 5]. Posted files have been in a variety of formats of the developers' choosing. Many of those interatomic potentials have now been incorporated into the openKIM repository [6, 7, 8], as well as other efforts . While the IPR will continue to provide an archive location, primarily for potentials not compatible with other projects (e.g., openKIM), some of these other projects are now sufficiently developed for the NIST IPR to link to them a a primary source and focus on larger questions related to simulation reproducibility and confidence, particularly related to classical atomistic simulations. The IPR will continue to provide potentials, as appropriate and requested, as well as links to locations such as openKIM, Web Force-Field (WebFF) [9], developer websites, and other relevant items as a service to our customers.

In support of this wider objective, the IPR is focusing more on the development of property calculation methodologies and self-contained scripts to be used for archiving research, and to be an access point for locating multiple interatomic potentials and force fields (archived through the IPR or elsewhere). Additionally, while frameworks such as the KIM project allow for automated comparisons for potentials and tests in the system, it is still necessary for researchers to develop the property calculation methodologies and contribute them to the project. The approach outlined here can be used to facilitate that development.

While progress has been made in recent years related to archiving potentials and generating databases of classical molecular (atomistic) simulation results, there is still a need for the development of self-contained simulation frameworks capable of running exclusively on local resources or via connections to external resources (e.g., repositories of interatomic potentials such as openKIM). Also needed are examples of fully documented calculations that can be used to understand how a calculation was done and be re-run by a user or adapted to a new simulation. Additionally, any scripts and tools need to fit within a larger ecosystem of tools for integrated computational materials engineering (ICME) and the Materials Genome Initiative (MGI). All of these considerations are important for documenting the research process, transferring knowledge between people, and improving the ability to re-run simulations without guessing what parameters or methods were originally used. They are also important considerations in the MGI and will become increasingly relevant with the push for open data and research.

In seeming contradiction to the open data movement, we are particularly focusing on tools that can be contained within local (or controlled-access) resources due to privacy or other considerations. This is because much MGI work is done in corporate or otherwise restricted environments and because researchers should have the discretion to determine how and when their data will be made publicly available, subject to policies and funding considerations. However, the ability to use tools in this way does not require anyone to do so, nor should it be construed as encouragement.

Any framework to support automated simulation management and property calculations needs to be flexible and capable of making use of high-performance computing resources. It should allow for automation and be easily adaptable to other simulation conditions or calculations. One application of this approach is in the development of interatomic potentials prior to release. The scripts used in this framework could also be adapted for broader research than just testing of interatomic potentials, or for method development prior to wider public release.

The vision described above has been implemented using Python-based tools for performing property calculations within the NIST IPR. As a case study, we will focus on the automation and augmentation of previously considered stacking fault energy calculations of nickel [10]. Stacking fault energy is an important property to consider if a researcher wishes to investigate fracture mechanics or dislocation phenomena within an atomistic simulation. We will demonstrate how the new scripts can easily be adapted to other elements and crystal structures. They can also be modified to treat alloys. We will explain how our tools allow easy calculation using many more potentials and materials, as well as easy adaptation from fcc to bcc, hcp and other crystal types, and additional stacking fault orientations. All simulations described here were conducted using the Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) [11, 12].

The study described in Ref. [10] contained more than ten thousand distinct molecular dynamics simulations that were managed by bash scripts and processed with linux utilities. While this approach worked well for us at the time, and enabled us to reproduce any of those calculations locally, it did not enable us to easily give the simulations to other researchers or incorporate them as part of a test calculation suite. Nor, based on our experience, did researchers given the simulation files necessarily understand how to use them if they were not already LAMMPS users. The new method allows for easier incorporation into research workflows and reproduction of previous work. It is important to note that we are also collaborating with others to incorporate these scripts and methodologies into other frameworks. For example, we are working to implement the generalized planar fault energy calculations in the openKIM system. We will also note that, consistent with MGI goals, the properties primarily of interest in this work are those used by engineers as part of the design process. This complements more fundamental tests such as barrier heights for atomic motion.

For the purposes of doing comparative simulations, it is often necessary to run many independent calculations with very similar initial conditions. Some of these calculations can also be sufficiently large or long that they should be done in a high performance computing environment. Thus our choice of simulation infrastructure must also accommodate that, which is also discussed in the following sections.

2. Documented research and integrated computational tools

As described in the previous section, we are rapidly computing a number of properties for many, if not most, interatomic potentials within the NIST IPR. It is also easy to run the scripts against potentials housed elsewhere, as long as they are available for download. This effort will generate a considerable amount of materials property data. This data will be presented as tables and figures on the NIST IPR website. Furthermore, data management and exchange are critical aspects of the MGI at NIST. Within this effort, we have focused on both local data archiving as well as discoverability and accessibility of atomistic simulation data. The Python scripts and data supporting this specific publication may be accessed online [13].

We selected Python because it is open source, actively maintained and developed, and does not have a required license fee. This may allow testing algorithms to be easily incorporated within other projects using Python [14] and Matlab [15]. Python scripts can be easily used with other tools like sumatra for version management. They could also be incorporated into sophisticated workflow management systems, but they do not need to be.

While Python scripts do not necessarily capture considerations such as environment or versions of the simulation codes being run, they are one means to document the steps, inputs, and outputs involved in a calculation and present that information to another user. This is particularly true if the scripts do not make extensive use of libraries and utilities outside of the core language. Additionally, the use of the scripts improves transparency by not being reliant on compiled code that can not be examined by the user (unless a researcher chooses to make such adaptations).

2.1. Simulation data archiving

As an initial step in archiving calculation results and supporting metadata in a self-documenting and portable way, we have chosen to archive both calculation data and simulation management data within the HDF5 format using PyTables [16, 17], and the H5MD format appears promising for efficiently accessing and archiving simulation output in the future [18]. We chose the HDF5 format for local storage because it has few limits on allowable data types (such as a tensor), it has few limits on data attributes (such as a physical unit), and the addition of new attributes or data does not generally break existing code for accessing data. Furthermore, HDF5 libraries are broadly available to users of java, C, C++, Fortran, MATLAB, Mathematica, and Python. It also comes packaged with a variety of graphical and command line utilities for accessing research data.

The main drawback with HDF5 is the lack of concurrent write from independent processes, which we resolved by using a single engine IPython cluster to manage read/write operations, as described in Appendix A.1. A traditional database would be a logical solution to concurrent write, but a traditional database does not offer the same level of support in different programming languages, flexibility, and ease of use as HDF5 for complex scientific data. For this reason, we conclude that the benefits of HDF5 outweigh its limitations, which may be temporary. Our use of HDF5 does not preclude other users from modifying the scripts to utilize alternative formats and platforms.

2.2. Material property data discovery and access

Data exchange is central to the mission of MGI at NIST. As a more integrated alternative to HDF5 files, we are adopting the Materials Data Curation System [19], which was developed in the NIST Information Technology Laboratory as a result of the MGI, and recently became available for download [20]. Unlike a traditional database design, the Materials Data Curation System uses a document database model for data storage, which allows for considerable flexibility in the structuring of the data. We are currently developing data models to store data pertaining to interatomic potentials, material properties derived from atomistic simulations, and reference data. The Materials Data Curation System includes a representational state transfer application programming interface (RESTful API), which will allow automated access to interatomic potentials and material property data stored at NIST. This will be discussed in a future publication.

3. Development and distribution of property calculation methods

A challenge associated with the transfer and reuse of a computational tool is the difficulty associated with understanding and modifying the underlying scientific methodology. Although the scripts described here lower the barriers to adoption of the MGI approach, the addition of automation and data management code may obscure the underlying physical methodology. Therefore, we are distributing IPython Notebooks that can be used to understand the calculations. The IPython Notebook contains a minimalist version of the calculation, which can be easily executed or modified. Complementing that minimalist code is rich text explanation and formatted equations, creating a self-enclosed exercise. The notebooks also provide examples of how automation and workflow tools improve reproducibility and knowledge transfer, both very important aspects of the MGI. In Figure 1, we show three segments of an IPython Notebook that computes the stacking fault energies as described in Ref. [10]. The top segment shows the formatted text capability of the notebook, the middle section shows a LAMMPS input script, where LAMMPS is executed as a Python subprocess, and the bottom section shows a plot defined from data generated by the atomistic simulation. The notebook can be exported as HTML, displayed on the IPR website, and viewed with a standard web browser. The user may also download the notebook, meet dependencies (Python, LAMMPS), edit the path (directory) to their LAMMPS executable, and "run all" cells to reproduce the calculation and figure. We note that this approach can also be used to fully document an entire research project, and notebooks used as examples can be teaching tools that can be modified and expanded for this purpose.

3.1. Calculations outside of intended usage

As discussed in Refs. [10, 4], interatomic potentials are often fit for specific applications. However, users employ these interatomic potentials for calculations of a significantly different nature. Thus, it is as important to understand how well these interatomic potentials behave for unintended use as for intended use. Therefore, interatomic potentials are sometimes tested for unintended applications. For example, within this work we are testing the elemental parts of a potentials that may have been strongly (or only) fit to the properties of a compound. We reiterate here, as elsewhere, that potentials used for unintended applications should not be considered "*bad*" if they do not compare favorably with reference data. Rather, they may be the best option available for the *intended* application(s) and users must exercise their best judgement and be prepared to defend their choices. Our intent is to present users with data from a number of interatomic potentials so they can make educated decisions about which potentials meet their needs. Furthermore, because we are providing the property calculation tools, we hope to make it easier for developers to test these properties during the fitting process so they know *before releasing a potential* how well it will compare against other models and reference data.

3.2. Support for High Performance Computing

The primary focus of this section is the development of high throughput property calculation scripts. These scripts are designed to be compatible with potentials from many locations, such as the Knowledgebase of Interatomic Models (openKIM) [6, 7, 8] and Web Force-Field (WebFF) [9], and for use on a researcher's available computational platform. A major goal of this work is to make the scripts as self-explanatory and simple as possible to use, without omitting desired or necessary functionality. This approach facilitates the sharing of methods (knowledge transfer), reproducibility, and archiving of simulation studies. This last point is particularly significant with the growing expectation that publicly funded results be open and machine readable [21].

Studies comparing different interatomic potentials may make use of local high performance computing resources. A material property may be computed by executing multiple simulations, each of which is intrinsically parallel. For example, one of the many ways to compute melting temperature involves simulating a solid-liquid interface at many temperatures near the melting temperature, where each simulation is large enough that it is beneficial to utilize parallelism. Therefore, we have designed our scripts to efficiently distribute parallel tasks. The Python script that was used to perform the calculations in this manuscript are available from NIST [13]. This script was executed via the command line at every stage of calculation. We fully document the use of the script in Appendix A, but we summarize it here.

First, for a single calculation (or a series of calculations), the script is used in "preparation" mode to store simulation input parameters in files devoted to job and task data. This is described in Appendix A.2. At the time of publication, an internal format was used to list a majority of the potentials in the NIST IPR. However, we are currently developing an open JSON format for potential parameters, which will allow our tools to compute material properties for any LAMMPS pair style, for both the single potential case or the many potential case. Furthermore, we have simplified the use of the scripts by building in default parameters for the most common use case. However, very little is shielded from the interested user. A significant number of calculation parameters, such as a fault plane or the fault directions are available for the user to adjust. Furthermore, the user can easily review or adjust LAMMPS input script commands used within the Python script. Next, many identical instances of the script are sent to the queueing system in "execute" mode. A given Python script running on the computing resource will request input parameters, run LAMMPS in serial or parallel mode (depending on batch and queue parameters), and store simulation results in the task data file. In this paradigm, tasks are completed in a just-in-time manner. Furthermore, by decoupling the parameters of the task execution script from parameters of the individual scientific task, it becomes straightforward to throttle the number of worker jobs or integrate the script within a traditional workflow management system. This is further described in Appendix A.3. Finally, the script contains two- and three-dimensional plotting modes, creating Figures 2-6 and Figure 7 respectively. This is described in Appendix A.4.

4. Case Study: Automation of Generalized Planar Fault Energy Simulations

As an example of how this improved methodology can be used, we will now revisit stacking fault energy calculations done previously [10]. We demonstrate the use of the scripts on Ag, Au, Cu, and Al, in addition to Ni. We show how results are generated by the scripts and presented to users on the IPR website to assist in characterization and comparison of results generated with different interatomic potentials. We also describe additional functionality to identify and label extrema and saddle points on the calculated generalized planar fault energy surface. We note that, while additional potentials will be examined with time, the old versions of scripts will be archived so that users can examine the evolution (if any) of a calculation methodology.

We began by integrating the LAMMPS input scripts used previously into a set of Python scripts and applied them to the pure elements listed in Table 1. By themselves, these scripts can be used to recreate the simulations previously executed. However, the nature of our investigation is to do many different simulations that use similar conditions and different interatomic potentials. In that case, reproducing the simulations by hand would be tedious, even if each simulation only required running a Python script. In order to deal with this, a more sophisticated automation framework was developed, which is described in Section 3.2.

The stacking fault calculation methodology was described in Ref. [10], but we summarize it here. We note that size studies, boundary condition selection, and other simulation conditions were examined previously. It is important that factors such as these be considered when using or adapting the input scripts, particularly when exploring a new class of materials or phenomena. However, since all details are available to users through the archived scripts, the users are able to examine parameters and make judgements about the quality of the simulations. Additionally, sensitivity analysis and similar types of studies are easily automated and documented in an approach such as described here.

Equilibrium bulk lattice constants a_0 were uniquely determined for each interatomic potential. Once those values were known, simulation blocks having dimensions of 1 x 1 x 6 rotated unit cells were created with $[0\,\bar{1}\,1]$ aligned with the x-axis, $[2\,\bar{1}\,\bar{1}]$ aligned with the y-axis, and $[1\,1\,1]$ aligned with the z-axis. Treated as completely independent LAMMPS simulations, the top half of the simulation block was rigidly displaced with respect to the bottom half in fractional units of the repeat length from 0.0 to 0.5 in increments of 0.01, where the repeat length in x and y are $a_0\sqrt{2}$ and $a_0\sqrt{6}$ respectively. Within each independent simulation a conjugate gradient minimization was performed prior to displacement to allow for surface relaxation. The total potential energy following this relaxation is recorded as $E_{\text{unfaulted}}$. The displacement then occurs and a second energy minimization is performed to allow for relaxation at the planar defect. The total potential energy following this relaxation is recorded as E_{faulted}. The planar fault energy is then computed as:

$$E_{\text{GPF}} = (E_{\text{faulted}} - E_{\text{unfaulted}})/A, \tag{1}$$

where A is the area of the fault. Results presented within this paper illustrate the information that is also presented on the IPR website as well as the capability of the Python property evaluation scripts. Figures 2-6 compare the stacking fault energies of $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ faults for pure nickel, aluminum, copper, silver, and gold interatomic potentials. All results are compared to a density functional theory (DFT) calculation, which was performed via the same methodology described in previous work [10]. All DFT calculations were performed using DMol^3 [22, 23], where the wavefunction is expanded in a double-zeta atom-centered basis set with a real-space cutoff of 0.4 nm. A generalized gradient approximation is used for the exchange-correlation functional [24], and a hardness conserving semilocal pseudopotential is utilized [25]. In the future, we plan to make comparison to additional reference values, including other DFT calculations [26], classical simulations via molecular statics [27], or experimental values

[28, 29, 30, 31].

Consistent with the previous investigation [10], most of the interatomic potentials produced the expected double peak energy profiles. The magnitude of the lower peak is known as the unstable stacking fault energy and the local minimum between the two peaks is known as the stable, or intrinsic, stacking fault energy. The unstable fault energy acts as an activation barrier that must be overcome during the creation of a stable stacking fault. Because it is energetically unfavorable, the higher peak in Figures 2-6 is not directly applicable to stacking fault phenomena. However, a few of the interatomic potentials produced magnitudes of the largest peak that were comparable to the unstable fault energy, which may increase the likelihood of slip in that direction. This observation highlights the importance of computing the entire fault energy surface and recalling that a given interatomic potential may have been designed for a different application, as discussed in Section 3.1.

While Figures 2-6 allow a researcher to compare interatomic potentials, it does not capture the complete stacking fault energy calculation, as shown in Figure 7. In order to make the calculation sufficiently general, we have included functionality to identify maxima, minima, and saddle points within a fault energy surface of sufficiently small discretization. For the fcc $\{1\,1\,1\}$ fault energy surface in Figure 7, these values map to the stable (local minima of 145.9 mJ/m^2) and unstable (saddle point of 167.3 mJ/m^2) stacking fault energies.

The figures presented within this paper are representative of the figures that appear on the NIST IPR website. Various sections of the website will be devoted to the comparison of single element and multicomponent material properties, similar to Figures 2-6. Furthermore, each interatomic potential and force field will have its own webpage, which will show a extensive summary of materials properties and include figures similar to Figure 7. These figures, and others like them, should decrease the time required to select an appropriate potential for a given use case.

5. Discussion and Outlook

Interatomic potentials have enjoyed significant development, and they are addressing more questions in more materials. With this growth comes a tremendous opportunity to begin to realize the vision of the MGI. To this end, we have taken steps to reduce the cost and time required to select and use an interatomic potential. However, many challenges remain. While we have begun to fully automate property calculations, we find it is increasingly important to adopt higher-level automation, which can execute and manage the various property calculation scripts in harmony with competing applications. While we have begun to present material property figures on the IPR website, interactive figures would convey more information. For example, clicking on a stacking fault curve in Figures 2-6 could relay additional data and metadata to the user, such as potential type (e.g., eam/fs) or a direct comparison to reference value(s). While in Figures 2-6 we have adopted shared symbols based on a numerical threshold, the original publications state unambiguously when portions of an interatomic potential are reused. Capturing, curating and presenting inheritance metadata can also help the user select an appropriate potential for their use case.

Throughout this paper, we have placed a specific focus on developing tools, which can be used on local HPC resources independent of other internal or external resources. As we continue to expand our portfolio of property evaluation tools, we will maintain our philosophy that users should have access to tools that fit their needs and environments. For example, while we are designing our tools to deposit material property data in the Materials Data Curation System, this is a non-default option. We believe that allowing reuse on either the component or the system level will maximize the likelihood of realizing MGI goals. Finally, we are fully embracing the open data movement via the Materials Data Curation System. In collaboration with those performing a range of experimental measurements, we are developing modular data models and metadata conventions, which we will discuss in a future publication. It is our goal that the data models developed for properties considered within the IPR can be easily extended to other properties, and therefore broadly improving the discoverability and accessibility of materials data.

6. Disclaimer

No approval or endorsement of any commercial product by NIST is intended or implied. Certain commercial software systems are identified in this paper to facilitate understanding. Such identification does not imply that these software systems are necessarily the best available for the purpose.

[1] James A. Warren and Ronald F. Boisvert. Building the materials innovation infrastructure: Data and standards. Technical Report NIST IR 7898, National Institute of Standards and Technology, Gaithersburg, MD, November 2012.

[2] `http://nist.gov/mgi/`

[3] `http://www.whitehouse.gov/mgi`

[4] Chandler A. Becker, Francesca Tavazza, Zachary T. Trautt, and Robert A. Buarque de Macedo. Considerations for choosing and using force fields and interatomic potentials in materials science and engineering. *Current Opinion in Solid State and Materials Science*, 17(6):277–283, December 2013.

[5] `http://www.ctcms.nist.gov/potentials/`

[6] Ellad B. Tadmor, Ryan S. Elliott, Simon R. Phillpot, and Susan B. Sinnott. NSF cyberinfrastructures: A new paradigm for advancing materials simulation. *Current Opinion in Solid State and Materials Science*, 17(6):298–304, December 2013.

[7] EB Tadmor, RS Elliott, JP Sethna, RE Miller, and CA Becker. The potential of atomistic simulations and the knowledgebase of interatomic models. *JOM Journal of the Minerals, Metals and Materials Society*, 63(7):17–17, 2011.

[8] `https://openkim.org`

[9] `http://www.nist.gov/mml/msed/polymers/webff.cfm`

[10] C.A. Becker, F. Tavazza, and L.E. Levine. Implications of the choice of interatomic potential on calculated planar faults and surface properties in nickel. *Philosophical Magazine*, 91(27):3578–3597, September 2011.

[11] Steve Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117(1):1–19, March 1995.

[12] `http://lammps.sandia.gov/`

[13] `http://hdl.handle.net/11256/121`

[14] Bohumir Jelinek, Kiran Solanki, John F Peters, and Sergio D Felicelli. Investigating robustness of interatomic potentials with universal interface. In *Journal of Physics: Conference Series*, volume 402, page 012006. IOP Publishing, 2012.

[15] `https://icme.hpc.msstate.edu/mediawiki/index.php/LAMMPS_tutorials`

[16] The HDF Group. Hierarchical Data Format, version 5, 1997-NNNN. `http://www.hdfgroup.org/HDF5/`

[17] Francesc Alted, Ivan Vilata, et al. PyTables: Hierarchical datasets in Python, 2002–.

[18] Pierre de Buyl, Peter H. Colberg, and Felix Höfling. H5md: A structured, efficient, and portable file format for molecular data. *Computer Physics Communications*, 185(6):1546–1553, June 2014.

[19] `https://mgi.nist.gov/materials-data-curation-system`

[20] `https://github.com/usnistgov/MDCS`

[21] Executive Order 13642, 78 FR 11739, 2013.

[22] B. Delley. An all-electron numerical method for solving the local density functional for polyatomic molecules. *The Journal of Chemical Physics*, 92(1):508, 1990.

[23] B. Delley. From molecules to solids with the DMol[sup 3] approach. *The Journal of Chemical Physics*, 113(18):7756, 2000.

[24] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. *Physical review letters*, 77(18):3865, 1996.

[25] B. Delley. Hardness conserving semilocal pseudopotentials. *Physical Review B*, 66(15), October 2002.

[26] Xiao-Zhi Wu, Rui Wang, Shao-Feng Wang, and Qun-Yi Wei. Ab initio calculations of generalized-stacking-fault energy surfaces and surface energies for FCC metals. *Applied Surface Science*, 256(21):6345–6349, August 2010.

[27] Jonathan A. Zimmerman, Huajian Gao, and Farid F. Abraham. Generalized stacking fault energies for embedded atom FCC metals. *Modelling and Simulation in Materials Science and*

Engineering, 8(2):103, 2000.

[28] John Price Hirth and Jens Lothe. *Theory of dislocations.* Krieger Pub. Co, Malabar, FL, 2nd ed edition, 1992.

[29] Lawrence Eugene Murr. Interfacial phenomena in metals and alloys. 1975.

[30] KH Westmacott and RL Peck. A rationalization of secondary defect structures in aluminium-based alloys. *Philosophical Magazine*, 23(183):611–622, 1971.

[31] CB Carter and ILF Ray. On the stacking-fault energies of copper alloys. *Philosophical magazine*, 35(1):189–200, 1977.

[32] Xiang-Yang Liu, Furio Ercolessi, and James B Adams. Aluminium interatomic potential from density functional theory calculations with improved stacking fault energy. *Modelling and Simulation in Materials Science and Engineering*, 12(4):665–670, July 2004.

[33] G.P. Purja Pun and Y. Mishin. Development of an interatomic potential for the ni-al system. *Philosophical Magazine*, 89(34-36):3245–3267, December 2009.

[34] M.I. Mendelev, M.J. Kramer, C.A. Becker, and M. Asta. Analysis of semi-empirical interatomic potentials appropriate for simulation of crystalline and liquid al and cu. *Philosophical Magazine*, 88(12):1723–1750, April 2008.

[35] Y. Mishin, M. Mehl, and D. Papaconstantopoulos. Embedded-atom potential for b2-NiAl. *Physical Review B*, 65(22), June 2002.

[36] Available at http://www.ctcms.nist.gov/potentials/.

[37] M. I. Mendelev, D. J. Sordelet, and M. J. Kramer. Using atomistic computer simulations to analyze x-ray diffraction data from metallic glasses. *Journal of Applied Physics*, 102(4):043501, 2007.

[38] Xiang-Yang Liu, P.P. Ohotnicky, J.B. Adams, C.Lane Rohrer, and R.W. Hyland. Anisotropic surface segregation in al?mg alloys. *Surface Science*, 373(2-3):357–370, March 1997.

[39] M.I. Mendelev, M.J. Kramer, R.T. Ott, D.J. Sordelet, D. Yagodin, and P. Popel. Development of suitable interatomic potentials for simulation of liquid and amorphous cu–zr alloys. *Philosophical Magazine*, 89(11):967–987, April 2009.

[40] A. Landa, P. Wynblatt, D.J. Siegel, J.B. Adams, O.N. Mryasov, and X.-Y. Liu. Development of glue-type potentials for the al–pb system: phase diagram calculation. *Acta Materialia*, 48(8):1753–1761, May 2000.

[41] Y. Mishin. Atomistic modeling of the γ and γ'-phases of the ni–al system. *Acta Materialia*, 52(6):1451–1467, April 2004.

[42] Y Mishin, D Farkas, MJ Mehl, and DA Papaconstantopoulos. Interatomic potentials for monoatomic metals from experimental data and ab initio calculations. *Physical Review B*, 59(5):3393, 1999.

[43] G. J. Ackland, G. Tichy, V. Vitek, and M. W. Finnis. Simple N -body potentials for the noble metals and nickel. *Philosophical Magazine A*, 56(6):735–756, December 1987.

[44] G Bonny, N Castin, and D Terentyev. Interatomic potential for studying ageing under irradiation in stainless steels: the FeNiCr model alloy. *Modelling and Simulation in Materials Science and Engineering*, 21(8):085004, December 2013.

[45] P L Williams, Y Mishin, and J C Hamilton. An embedded-atom potential for the cu–ag system. *Modelling and Simulation in Materials Science and Engineering*, 14(5):817–833, July 2006.

[46] M.I. Mendelev, M.J. Kramer, S.G. Hao, K.M. Ho, and C.Z. Wang. Development of interatomic potentials appropriate for simulation of liquid and glass properties of NiZr $_2$ alloy. *Philosophical Magazine*, 92(35):4454–4469, December 2012.

[47] Y. Mishin, M. Mehl, D. Papaconstantopoulos, A. Voter, and J. Kress. Structural stability and lattice defects in copper: Ab initio, tight-binding, and embedded-atom calculations. *Physical Review B*, 63(22), May 2001.

[48] J J Hoyt, J W Garvin, E B Webb, and Mark Asta. An embedded atom method interatomic potential for the cu pb system. *Modelling and Simulation in Materials Science and Engineering*, 11(3):287–299, May 2003.

[49] MI Mendelev, M Asta, MJ Rahman, and JJ Hoyt. Development of interatomic potentials appropriate for simulation of solid–liquid interface properties in al–mg alloys. *Philosophical Magazine*, 89(34-36):3269–3285, 2009.

[50] X. Zhou, R. Johnson, and H. Wadley. Misfit-energy-increasing dislocations in vapor-deposited CoFe/NiFe multilayers. *Physical Review B*, 69(14), April 2004.

[51] J E Angelo, N R Moody, and M I Baskes. Trapping of hydrogen to lattice defects in nickel. *Modelling and Simulation in Materials Science and Engineering*, 3(3):289–307, May 1995.

[52] Henry H. Wu and Dallas R. Trinkle. Cu/ag EAM potential optimized for heteroepitaxial diffusion from ab initio data. *Computational Materials Science*, 47(2):577–583, December 2009.

[53] Jess Sturgeon and Brian Laird. Adjusting the melting point of a model system via gibbs-duhem integration: Application to a model of aluminum. *Physical Review B*, 62(22):14720–14727, December 2000.

[54] G Bonny, R C Pasianot, and L Malerba. Fe–ni many-body potential for metallurgical applications. *Modelling and Simulation in Materials Science and Engineering*, 17(2):025010, March 2009.

[55] M.I. Mendelev, D.J. Srolovitz, G.J. Ackland, and S. Han. Effect of fe segregation on the migration of a non-symmetric $\sigma 5$ tilt grain boundary in al. *Journal of Materials Research*, 20(01):208–218, January 2005.

[56] Rajendra Zope and Y. Mishin. Interatomic potentials for atomistic simulations of the ti-al system. *Physical Review B*, 68(2), July 2003.

[57] J. B. Adams, S. M. Foiles, and W. G. Wolfer. Self-diffusion and impurity diffusion of fee metals using the five-frequency model and the embedded atom method. *Journal of Materials Research*, 4:102–112, 1989.

[58] J M Winey, Alison Kubota, and Y M Gupta. A thermodynamic approach to determine accurate potentials for molecular dynamics simulations: thermoelastic response of aluminum. *Modelling and Simulation in Materials Science and Engineering*, 17(5):055004, July 2009.

[59] Gregory Grochola, Salvy P Russo, and Ian K Snook. On fitting a gold embedded atom method potential using the force matching method. *The Journal of chemical physics*, 123(20):204719, 2005.

Al
Al_LiuX-Y_2004 [32]
Ni-Al_MishinY_2002 [35]
Mg-Al_LiuX-Y_1997 [38]
Al-Pb_LandaA_2000 [40]
Al-Co_PurjaPunGP_2013 [36]
Ni-Al-Co_PurjaPunGP_2013 [36]
Ni-Al_PurjaPunGP_2009 [33]
Al_MishinY_1999 [42]
Al-Mg_MendelevMI_2009 [49]
Al_MendelevMI_2008 [34]
Al_SturgeonJB_2000 [53]
Al-Fe_MendelevMI_2005 [55]
Al_ZopeRR_2003 [56]
Ni-Al_MishinY_2004 [41]
Ti-Al_ZopeRR_2003 [56]
Al_WineyJM_2009 [58]
Al_ZhouXW_2004 [50]
Ni-Al-H_AngeloJE_1995 [51]

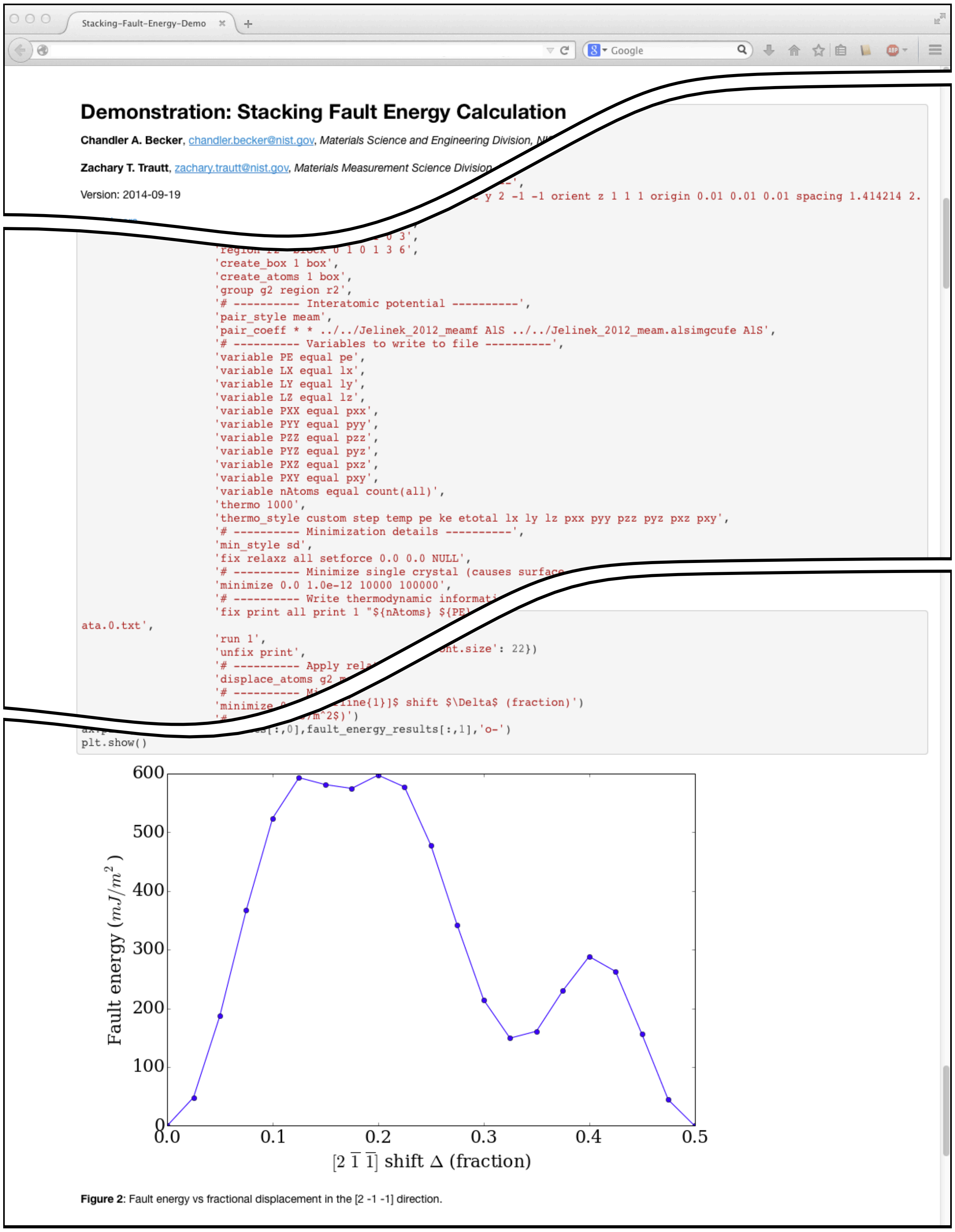
Ni
Ni-Al_PurjaPunGP_2009 [33]
Ni-Al-Co_PurjaPunGP_2013 [36]
Ni-Co_PurjaPunGP_2013 [36]
Ni-Al_MishinY_2004 [41]
Ni_MishinY_1999 [42]
Fe-Ni-Cr_BonnyG_2013 [44]
Ni_MendelevMI_2012 [46]
Ni-Zr_MendelevMI_2012 [46]
Ni_ZhouXW_2004 [50]
Ni-Al-H_AngeloJE_1995 [51]
Fe-Ni_BonnyG_2009 [54]
Ni-Al_MishinY_2002 [35]
Ni_AcklandGJ_1987 [43]

Au
Au_AcklandGJ_1987 [43]
Au_AdamsJB_1989 [57]
Au_GrocholaG_2005 [59]
Au_ZhouXW_2004 [50]

Cu
Cu_MendelevMI_2008 [34]
Cu-Zr_MendelevMI_2007 [37]
Cu-Zr_MendelevMI_2009 [39]
Cu_MendelevMI_2012 [36]
Cu_AcklandGJ_1987 [43]
Cu-Ag_WilliamsPL_2006 [45]
Cu_MishinY_2001 [47]
Cu-Pb_HoytJJ_2003 [48]
Cu_ZhouXW_2004 [50]
Cu-Ag_WuHH_2009 [52]

Ag
Ag_AcklandGJ_1987 [43]
Ag_AdamsJB_1989 [57]
Cu-Ag_WuHH_2009 [52]
Ag_WilliamsPL_2006 [45]
Cu-Ag_WilliamsPL_2006 [45]
Ag_ZhouXW_2004 [50]

Table 1. Identifiers assigned to the interatomic potentials investigated in this study, sorted by element.


Stacking-Fault-Energy-Demo
Google
600
500
400
300
200
100
0
0.0
0.1
0.2
0.3
0.4
0.5
Fault energy (mJ/m^2)
$[2\,\bar{1}\,\bar{1}]$ shift Δ (fraction)

Demonstration: Stacking Fault Energy Calculation

Chandler A. Becker, chandler.becker@nist.gov, *Materials Science and Engineering Division, N*

Zachary T. Trautt, zachary.trautt@nist.gov, *Materials Measurement Science Division*

Version: 2014-09-19

```
                                                                    -',
                                              y 2 -1 -1 orient z 1 1 1 origin 0.01 0.01 0.01 spacing 1.414214 2.
                                                     0 3',
                       'region r2  block 0 1 0 1 3 6',
                       'create_box 1 box',
                       'create_atoms 1 box',
                       'group g2 region r2',
                       '# ---------- Interatomic potential ----------',
                       'pair_style meam',
                       'pair_coeff * * ../../Jelinek_2012_meamf AlS ../../Jelinek_2012_meam.alsimgcufe AlS',
                       '# ---------- Variables to write to file ----------',
                       'variable PE equal pe',
                       'variable LX equal lx',
                       'variable LY equal ly',
                       'variable LZ equal lz',
                       'variable PXX equal pxx',
                       'variable PYY equal pyy',
                       'variable PZZ equal pzz',
                       'variable PYZ equal pyz',
                       'variable PXZ equal pxz',
                       'variable PXY equal pxy',
                       'variable nAtoms equal count(all)',
                       'thermo 1000',
                       'thermo_style custom step temp pe ke etotal lx ly lz pxx pyy pzz pyz pxz pxy',
                       '# ---------- Minimization details ----------',
                       'min_style sd',
                       'fix relaxz all setforce 0.0 0.0 NULL',
                       '# ---------- Minimize single crystal (causes surface
                       'minimize 0.0 1.0e-12 10000 100000',
                       '# ---------- Write thermodynamic informati
                       'fix print all print 1 "${nAtoms} ${PE}
ata.0.txt',
                       'run 1',
                       'unfix print',                          ont.size': 22})
                       '# ---------- Apply rel
                       'displace_atoms g2 m
                       '# ---------- Mi
                       'minimize 0           line{1}]$ shift $\Delta$ (fraction)')
                       '#            /m^2$)')
ax.p                      s[:,0],fault_energy_results[:,1],'o-')
plt.show()
```

Figure 2: Fault energy vs fractional displacement in the [2 -1 -1] direction.

Figure 1. Illustration of three segments of an IPython Notebook, which defines, explains, and executes a simplified stacking fault energy calculation.

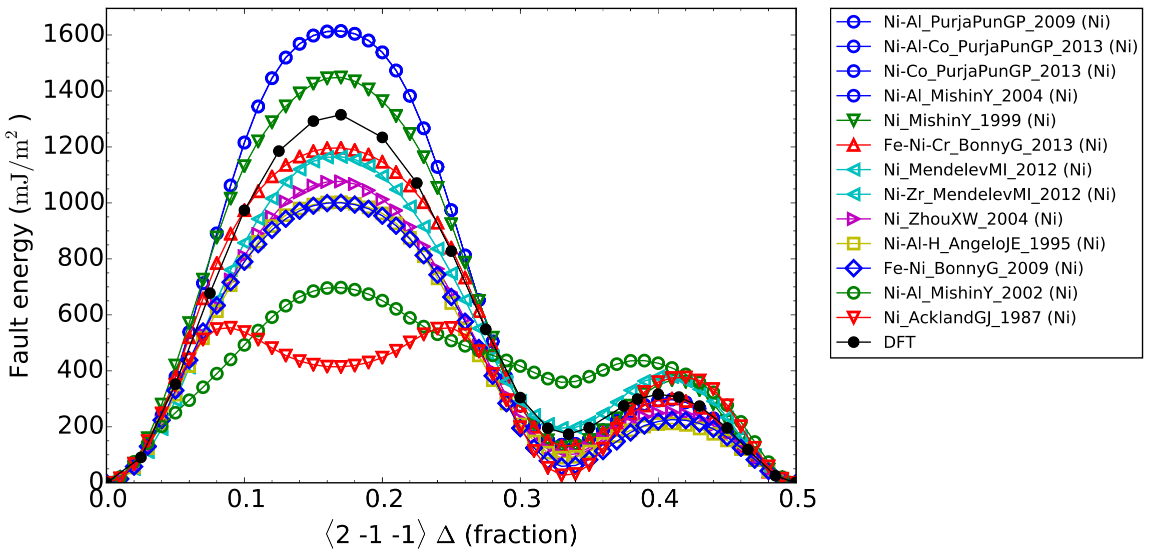


Figure 2. Comparison of nickel $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ fault energies to DFT results. Symbols are shared when the cumulative energy difference is less than 1.0 mJ/m^2.

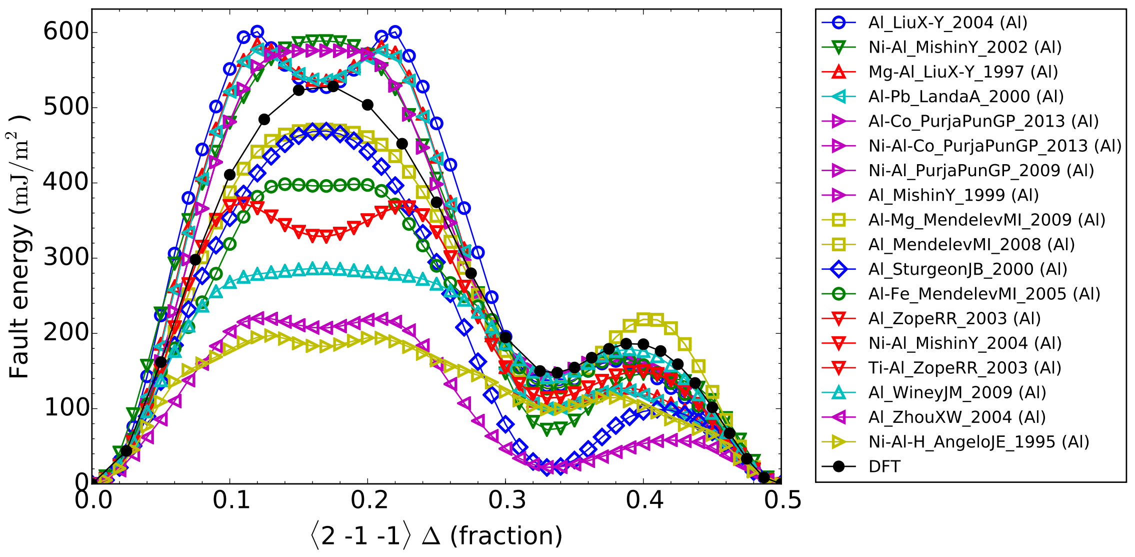


Figure 3. Comparison of aluminum $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ fault energies to DFT results. Symbols are shared when the cumulative energy difference is less than 1.0 mJ/m^2.

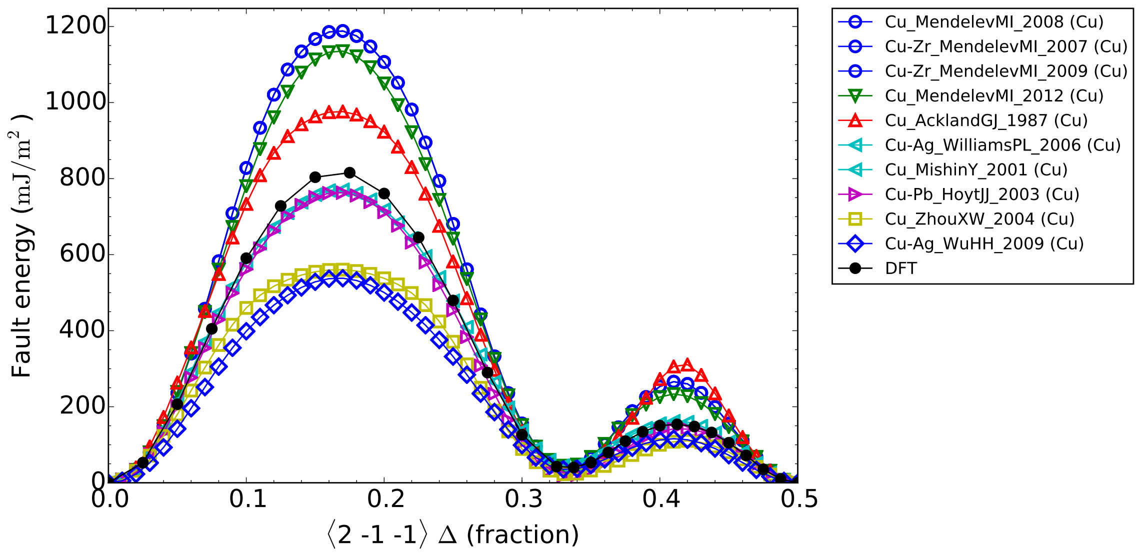


Figure 4. Comparison of copper $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ fault energies to DFT results. Symbols are shared when the cumulative energy difference is less than 1.0 mJ/m^2.

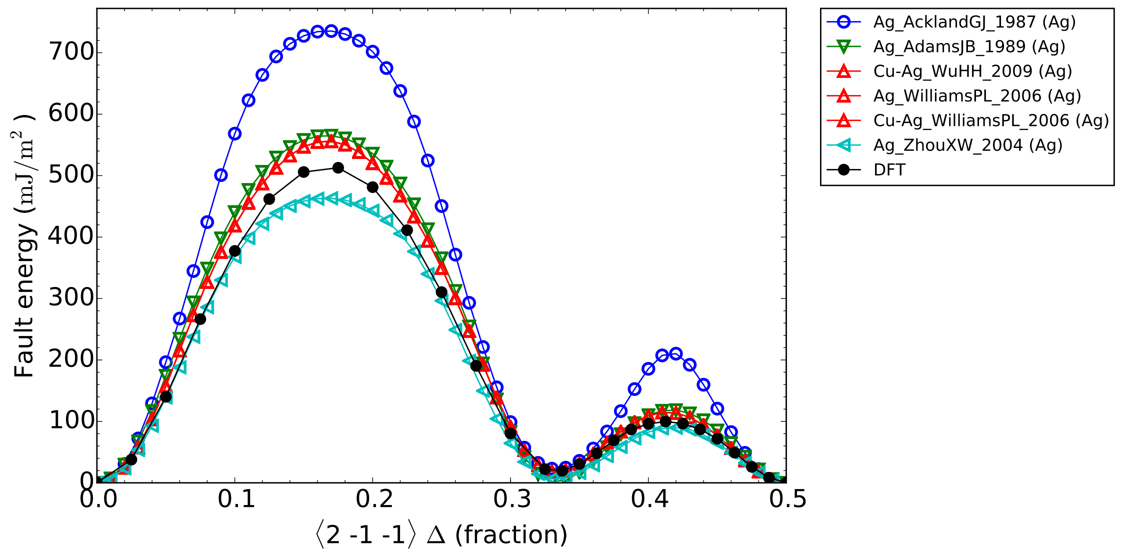


Figure 5. Comparison of silver $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ fault energies to DFT results. Symbols are shared when the cumulative energy difference is less than 1.0 mJ/m^2.

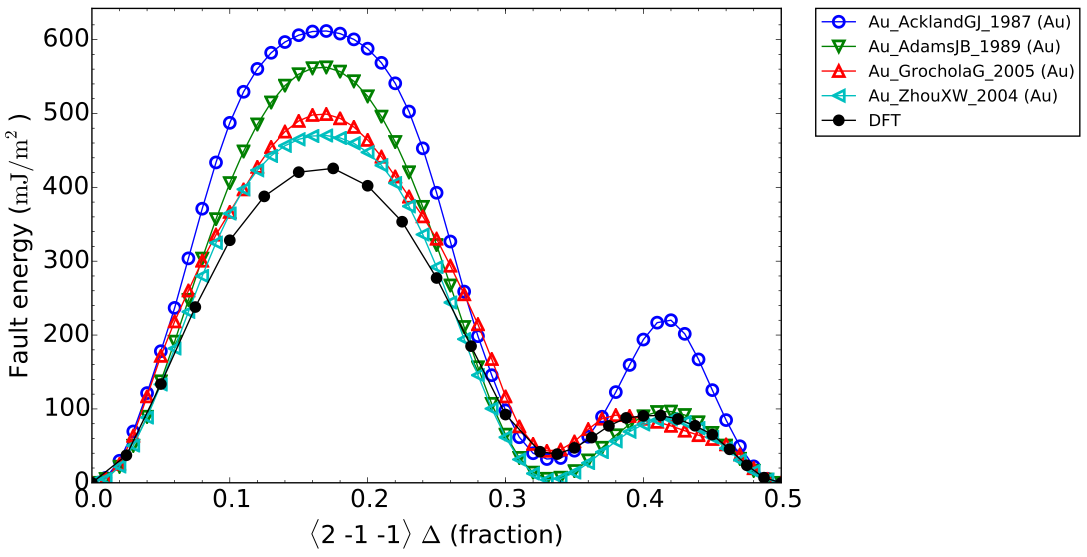


Figure 6. Comparison of gold $\{1\,1\,1\}\langle 2\,\bar{1}\,\bar{1}\rangle$ fault energies to DFT results.

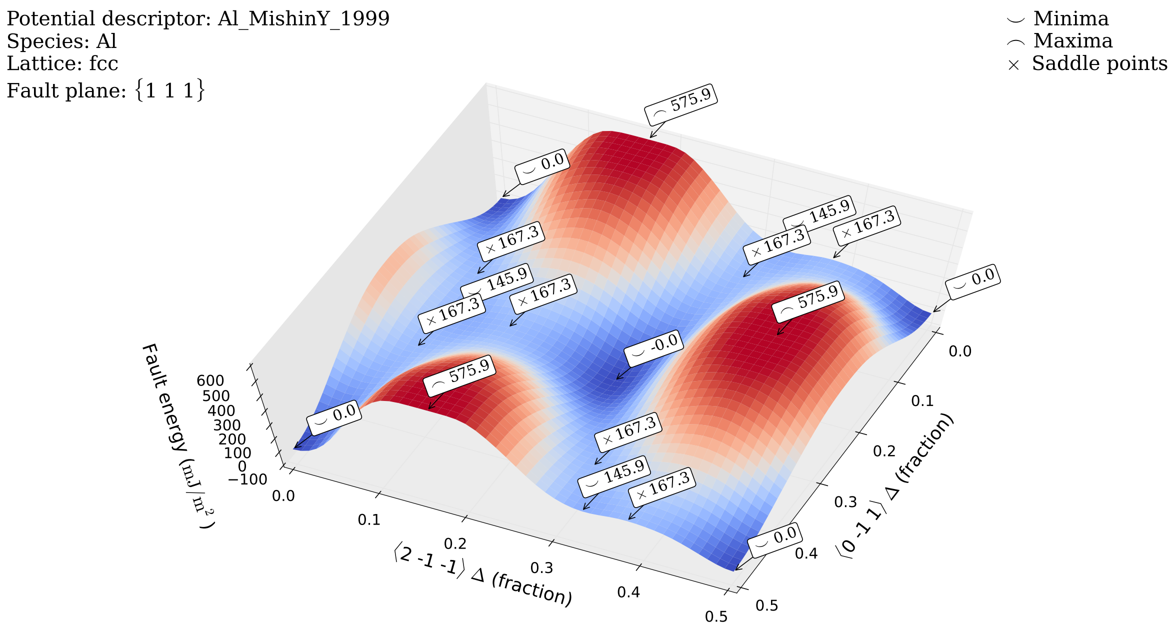


Figure 7. Illustration of a complete fcc $\{1\,1\,1\}$ fault energy surface with local minima, maxima, and saddle points labeled with their respective fault energy. The displacements are shown as a fraction (Δ) of the given lattice vector. These labels are both generally applicable to any fault plane or unit cell, and clearly identify the stable (local minima of 145.9 mJ/m^2) and unstable (saddle point of 167.3 mJ/m^2) stacking fault energies for $\{1\,1\,1\}$ faults in fcc crystals.

Appendix A. Using the High Throughput Python Scripts

The usage of the scripts generally requires three steps: (i) job preparation, (ii) large scale task execution, and (iii) plotting and/or analysis. In the sections that follow, we demonstrate the usage of the planar fault energy script and describe the operations occurring therein. Each Section will reference portions of Figure A1.

Appendix A.1. Starting the IPython cluster

In their current form, the scripts require the use of a single-engine IPython cluster, which exists to queue read and write operations to the HDF5 files. We show a single-engine IPython cluster, composed of an ipcontroller and ipengine, with the bidirectional flow of data represented by colored lines in Figure A1. We use the following bash script to launch a single engine IPython cluster:

```
#!/usr/bin/env bash
ipcontroller --ip=* --profile=pyGPFE &
sleep 3
ipengine --profile=pyGPFE
```

We execute this script within an instance of linux *screen* window manager on the head node. Alternatively, this could also be executed as a serial batch job on a HPC node.

In the code above, the `ipcontroller` command starts the IPython controller, the `--ip=*` option instructs the controller to listen on all interfaces, and the `--profile=pyGPFE` option instructs the controller to use an IPython profile specifically designated for generalized planar fault energy (GPFE) calculations. Then after three seconds via the `sleep` command, an IPython engine is started using the same IPython profile.

Appendix A.2. Calculation preparation

Preparation routines were designed with large numbers of interatomic potentials in mind. While we use these scripts to test the interatomic potentials within the NIST repository, a potential developer might test a large number of candidate potentials. A researcher may run a series of similar calculations to perform parameter studies or calculate for a range of input parameters (e.g., temperature). All calculations presented within this paper can be prepared by running the script five times with the following options:

```
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -s Ni
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -s Al
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -s Cu
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -s Ag
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -s Au
```

This will prepare calculations for all nickel, aluminum, copper, silver, and gold interatomic potentials within the "lammps_potentials_file". The "jobs.h5" file contains newly created job management data. A number of new directories containing calculation

HDF5 files are created as a result of this step. This is illustrated in the "file system" portion of the diagram in Figure A1. No calculation parameters are necessary in the above terminal commands, as an fcc $\{1\,1\,1\}$ fault is the default option. Alternatively, if one wished to compute a hcp $\{0\,0\,0\,1\}$ fault for a single potential, the following command serves as an example:

```
./GPFE.py prepare -J jobs.h5 -P lammps_potentials_file -D . -p
    Al-Mg_MendelevMI_2009 -s Mg -l hcp -X 1 0 0 -Y 0 1 0 -Z 0 0 1 -XS 0.0 1.0
    0.02 -YS 0.0 1.0 0.02
```

In this example, a single element within a binary interatomic potential is tested. The additional options specify the hcp lattice type and the orientation and displacement parameters of the fault plane for the rectangular unit cell constructed within LAMMPS. We note that in this case, two equilibrium lattice constants (a_0 and c_0) are determined prior to fault energy calculations. Full documentation will be made available when the tools are published online.

Appendix A.3. Simulation task execution

Task execution routines were designed such that an unbounded number of identical scripts can be executed on the user's computational environment, where simulation tasks are automatically completed in a just-in-time fashion. As the Python script completes many tasks, this reduces the total number of jobs submitted to the batch system, which may be very advantageous in some cases. On our linux cluster, we launch a large number of the following batch scripts:

```
#PBS -l nodes=1:ppn=1
#PBS -N GPFEworker
#PBS -q default
./GPFE.py execute -J jobs.h5 -L ~/bin/lammps
```

Alternatively, when performing a single fault energy calculation on a linux workstation, we execute 12 tasks to fully populate available CPUs via:

```
for i in {1..12}; do screen -S GPFEworker -d -m ./GPFE.py execute -J jobs.h5
    -L ~/bin/lammps; done
```

Referring to Figure A1, we show multiple GPFE execution scripts in green. When a GPFE script is first started, it will create a blocking direct view client instance to the IPython cluster, as shown by the green lines. Next, the GPFE script will request an unfinished task via the IPython cluster. Task information, such as simulation parameters, are returned to the GPFE script. Task logic is built into the execution script such that tasks are completed in the following order: (1) compute equilibrium lattice constant(s), (2) perform stacking fault simulations, and (3) identify extrema in the fault energy surface. During an atomistic simulation task, a LAMMPS input script is written to the filesystem, and LAMMPS is executed as a Python subprocess. While we recognize that LAMMPS can be utilized as a library, we find value in having the ability to debug LAMMPS simulations independent of Python automation. This also has the added benefit of shielding no aspect of the calculation from the user.

The execution commands above clearly indicate the distribution of serial tasks. As the stacking fault energy surface calculation involves relatively few atoms, this is the most efficient option. However, this is not the case for other tools currently under development. For example, in our finite temperature elastic constant script currently under development, we launch a large number of the following batch scripts:

```
#PBS -l nodes=1:ppn=8
#PBS -N pyCijWorker
#PBS -q default
./Cij.py execute -J jobs.h5 -R 'mpirun -np 8' -L ~/bin/lammps
```

This automates the execution of parallel simulation tasks, each using 8 CPUs, which may be adjusted heterogeneously. In this example, an additional option of `-R` specifies the parallel execution prefix specific to the user's computational environment.

Appendix A.4. Plotting and analysis

Two plotting options are currently available: (i) two dimensional plotting as is shown in Figures 2-6 and three dimensional plotting as shown in Figure 7. In Section Appendix A.2, we demonstrated preparation of a single hcp calculation in magnesium. After all execution engines have finished, a three dimensional plot can be created with the following command:

```
./GPFE.py plot3D -D
    job_Al-Mg_MendelevMI_2009_11ca8609-b9fc-4acf-82e2-0be01f794632/GPFE.h5 -p
    Figure_9.eps -t 200
```

The three options specify the directory location of the calculation HDF5 file, the file name of the resulting figure, and a threshold over which fault extrema are not plotted. We note that the string following "job_Al-Mg_MendelevMI_2009" is a unique identifier, and it will be different each time the calculation is repeated. The resulting figure is shown in Figure A2. Two dimensional plots can be generated with very similar commands, and full documentation will be made available online.

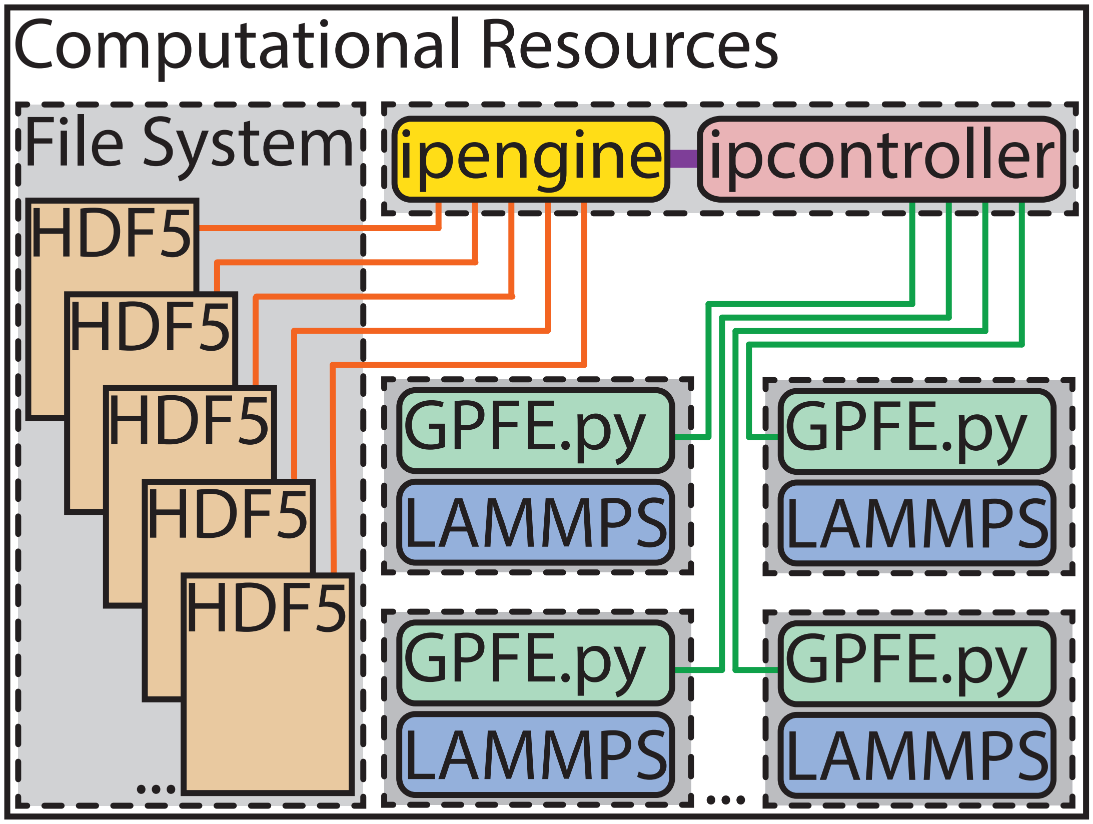


Figure A1. Illustration task distribution and data management paradigm during molecular dynamics simulation task execution as described in Section 3.2. For illustration purposes, a only a small number of HDF5 files and execution scripts is shown.

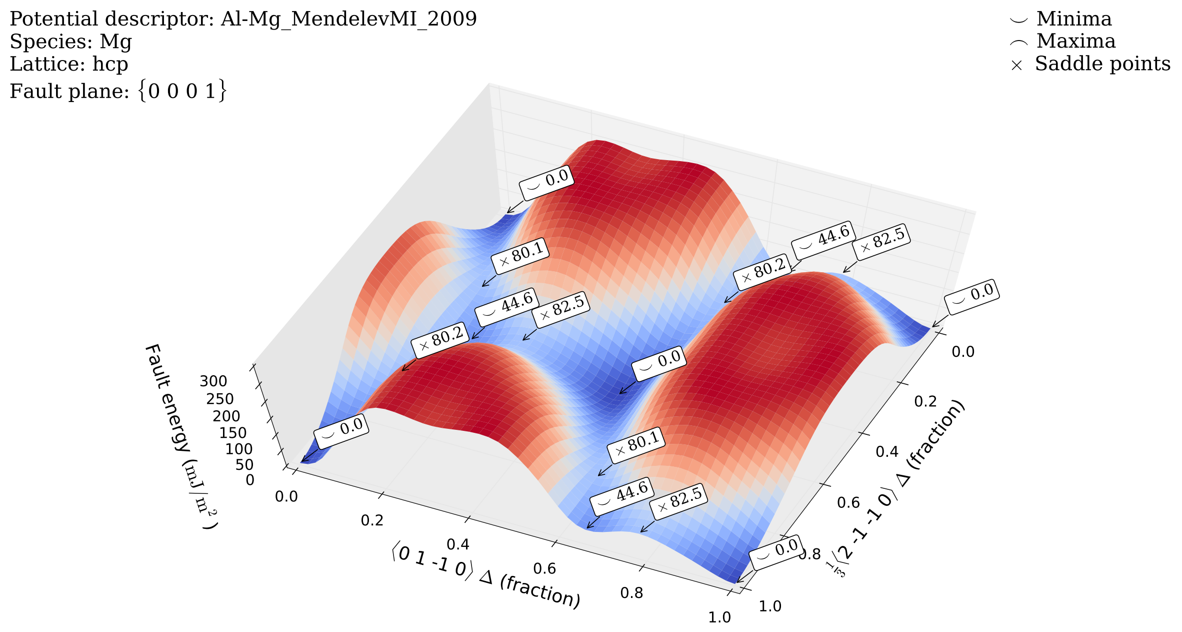


Figure A2. Illustration of a complete hcp {0 0 0 1} fault energy surface with local minima, maxima, and saddle points labeled with their respective fault energy. The displacements are shown as a fraction (Δ) of the given lattice vector.