Evaluating variability with atomistic simulations: the effect of potential and calculation methodology on the modeling of lattice and elastic constants

Lucas M. Hale, Zachary T. Trautt, Chandler A. Becker

Abstract

Atomistic simulations using classical interatomic potentials are powerful investigative tools linking atomic structures to dynamic properties and behaviors. It is well known that different interatomic potentials produce different results, thus making it necessary to characterize potentials based on how they predict basic properties. Doing so makes it possible to compare existing interatomic models in order to select those best suited for specific use cases, and to identify any limitations of the models that may lead to unrealistic responses. While the methods for obtaining many of these properties are often thought of as simple calculations, there are many underlying aspects that can lead to variability in the reported property values. For instance, multiple methods may exist for computing the same property and values may be sensitive to certain simulation parameters. Here, we introduce a new high-throughput computational framework that encodes various simulation methodologies as Python calculation scripts. Three distinct methods for evaluating the lattice and elastic constants of bulk crystal structures are implemented and used to evaluate the properties across 120 interatomic potentials, 18 crystal prototypes, and all possible combinations of unique lattice site and elemental model pairings. Analysis of the results reveals which potentials and crystal prototypes are sensitive to the calculation methods and parameters, and it assists with the verification of potentials, methods, and molecular dynamics software. The results, calculation scripts, and computational infrastructure are self-contained and openly available to support researchers in performing meaningful simulations.

1. Introduction

Calculations and simulations using so-called classical interatomic potentials occupy a unique place in materials research. Classical molecular dynamics (MD) and Monte Carlo (MC) simulations are considerably less computationally expensive than comparable quantum-based calculations, and will likely remain so as the classical methods typically scale with the number of

atoms, while density functional theory (DFT), the most used quantum-based calculation method, scales with the number of atoms cubed. However, this computational efficiency comes at the cost of accuracy and adaptability in what the classical potentials can realistically represent. In practice, this tradeoff between computation and accuracy means that classical atomistic simulations are best suited for revealing how atomic-level structures and interactions influence the complex, dynamic behaviors and reactions of collections of atoms, molecules, and crystalline defects.

Appropriately performing atomistic simulations and reporting meaningful results requires understanding the strengths and limitations associated with the empirical potentials used. Most classical potentials have functional forms that provide good representations for certain atomic bonds and poor representations of others. Developers fit the parameters of the models to capture specific materials properties, atomic configurations, and energy barriers that they consider most important. The accuracy of these fitted properties depends on the accuracy of the underlying reference information, and the weights that the potential creators placed on capturing each property. The resulting potentials can only be expected to give realistic predictions for simulations that operate within their fitted phase space. However, scientifically interesting simulations often involve complex, dynamically evolving configurations that may explore conditions outside the fitted regime. Characterizing simulation results as realistic or artificial requires an understanding of how the potential behaves under all explored conditions.

Further complications arise from the fact that there are often numerous interatomic potentials available for a given material system. For example, the National Institute of Standards and Technology (NIST) Interatomic Potentials Repository [1] currently lists entries for 22 potentials capable of simulating nickel. These models encompass multiple functional forms, and were fit with different applications and properties in mind. Some even were fit solely for compounds and provide poor representations of elemental nickel. Anyone planning on performing simulations of nickel would need to be able to determine which, if any, of the existing potentials is most likely to provide realistic predictions for their area of study.

The nature of atomistic simulations makes it difficult, if not impossible, to evaluate the quality of potentials based on their mathematical forms alone. Instead, it is only through adequate characterization and consideration of the potentials' predicted properties that one can realistically compare different potentials [2, 3], and to properly analyze simulation results. There

have been numerous comparison studies of potentials published in the literature, often accompanying the release of a new potential [4-8]. While useful, the static nature of the publications limits the comparisons to only a handful of currently existing models. Additionally, there is no consistency across the works in terms of the properties considered, and commonly reported properties may have been evaluated using different methodologies. It is also not uncommon for the description of the methodologies to be lacking or non-existent for many of the so-called basic properties. Clear methodology is important in model verification as it allows for consistent tests and can help identify the source of variations in reported values across different works.

To address this need for clear, repeatable atomistic property evaluations, Trautt, et al. [9] outlined a framework for performing calculations across different interatomic potentials, and demonstrated its application to producing generalized stacking fault maps for face centered cubic (fcc) metals. The most notable aspect of that work was that it outlined a means in which the calculations could be performed in a high-throughput manner, while keeping the calculation's methodology transparent to the user.

This paper introduces a new Python-based high-throughput calculation framework which builds upon the principles introduced in [9]. The new framework features a modular design better suited for creating and running a wide variety of calculations in high-throughput. An emphasis is placed on constructing calculation methods that are easy to use, exist in a concise and sharable format, and fully document and describe the underlying methodology. Additionally, results are produced in a structured format that is human and machine readable allowing for the data to be easily shared either as individual files or as part of a database.

The new framework is utilized to compare the relaxed lattice and elastic constants using different computational methods. Calculations are performed across eighteen different crystal prototypes, one hundred twenty interatomic potentials and three different relaxation methods. Analysis of the results proves to be valuable in assessing not only the strengths and limitations of the calculation methods and interatomic models, but also assists in verification of the underlying algorithms and code.

2. Computational Methods

All calculations performed for this work are encoded as Python scripts within the iprPy computational framework (accessible at https://github.com/usnistgov/iprPy). The iprPy framework is focused on the design and creation of open and transparent calculation methodologies that can easily be integrated into high-throughput workflows. Each calculation script represents the full methodology of a property evaluation by setting up and performing one or more Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) molecular dynamics [10, 11] simulations, followed by processing the simulation results to obtain values of interest. Simple input files provide all variable parameters to each calculation, and the results are exported as structured eXtensible Markup Language (XML)- or JavaScript Object Notation (JSON)-based data models. The high-throughput tools of the iprPy framework were used to prepare and run all calculations presented here. More details of the design of the iprPy framework can be found in Appendix A.

Table 1: List of the crystal prototypes used in this study.

ID	Strukturbericht	Prototype Composition	Common Name
A1--Cu--fcc	A1	Cu	Face-centered cubic (fcc)
A2--W--bcc	A2	W	Body-centered cubic (bcc)
A3'--alpha-La--double-hcp	A3'	α-La	Double hexagonal close-packed (double hcp)
A3--Mg--hcp	A3	Mg	Hexagonal close-packed (hcp)
A4--C--dc	A4	Cu	Diamond cubic (dc)
A5--beta-Sn	A5	β-Sn	
A6--In--bct	A6	In	Body-centered tetragonal (bct)
A7--alpha-As	A7	α-As	
A15--beta-W	A15	β-W	
Ah--alpha-Po--sc	Ah	α-Po	Simple cubic (sc)
B1--NaCl--rock-salt	B1	NaCl	Rock salt
B2--CsCl	B2	CsCl	
B3--ZnS--cubic-zinc-blende	B3	ZnS	Cubic zinc blende
C1--CaF2--fluorite	C1	CaF_2	Fluorite
D0_3--BiF3	$D0_3$	BiF_3	
L1_0--AuCu	$L1_0$	AuCu	
L1_2--AuCu3	$L1_2$	$AuCu_3$	
L2_1--AlCu2Mn--heusler	$L2_1$	$AlCu_2Mn$	Heusler

The calculations were performed across one hundred twenty interatomic potentials, eighteen crystal prototypes, and all pairing combinations of each potential's elemental models to each prototype's unique lattice sites. All potentials used here are hosted on the NIST Interatomic Potentials Repository [1] in formats compatible with LAMMPS. Table I lists the crystal prototypes used. For this work, the prototypes are referred to using a unique identifier that combines the Strukturbericht symbol, prototype composition, and common name. Of the crystal prototypes investigated, nine are for elemental structures and nine are for binary or ternary compounds.

Four different calculation scripts were used for this work, referred to as E_vs_r_scan, LAMMPS_ELASTIC, refine_structure, and dynamic_relax. Initial rough estimates for the lattice parameters of stable crystal structures were first obtained using the E_vs_r_scan calculation. The structures corresponding to energy minima identified by the E_vs_r_scan calculation were then used as initial guesses for stable crystal structures that were passed on to the other calculation scripts, with each one providing some means of relaxing the initial guess to a more optimal configuration. The LAMMPS_ELASTIC, and refine_structure calculations also provide an estimate of the relaxed structure's elastic constants.

The E_vs_r_scan calculation evaluates a crystal's cohesive energy, E_{coh}, as a function of shortest atomic nearest neighbor distance, r_0. An atomic system for a hypothetical crystal structure is generated by filling in the unique lattice sites of a crystal prototype with a potential's elements. The dimensions of the system are uniformly scaled relative to r_0. For this calculation, the ratio of the lattice constants is held fixed at an ideal value (e.g., c/a = 1.663 for hexagonal close-packed). For each value of r_0, the cohesive energy is evaluated using LAMMPS without atomic relaxations. The analysis here used 100 steps of r_0 over the range $0.2\ \text{nm} \leq r_0 \leq 0.6\ \text{nm}$. This range was found to be optimum for the potentials included in this investigation as most of the potentials are for metallic and semiconductor systems.

The refine_structure calculation statically calculates the ideal lattice constants and elastic constants at a specified pressure. The underlying algorithm works by having LAMMPS evaluate the pressure for a system as given, and at small positive and negative strains without any atomic relaxation. The full elastic stiffness tensor, C_{ij}, is calculated from the change in pressures with respect to the change in strains. Assuming linear elasticity, the pressure of the unstrained system

and the elastic compliance tensor, $S_{ij} = C_{ij}^{-1}$, are used to guess a new box size, and the system is uniformly scaled accordingly. This process is then repeated until the lattice constants converge.

The LAMMPS_ELASTIC calculation script is a wrapper around the ELASTIC example simulation distributed with the LAMMPS code in its Examples folder. The ELASTIC example script is a well-known, open source resource for LAMMPS users to statically calculate the elastic constants for a crystal. In the ELASTIC example, the system is first relaxed using an energy minimization of atomic coordinates coupled with box dimension adjustments towards zero pressure (i.e., minimize plus fix box_relax commands). The elastic stiffness tensor is then evaluated using the virial pressure of the relaxed system, and at fixed small strain values from the relaxed configuration. For the small strain states, each was subjected to an energy minimization of the atomic coordinates with fixed box dimensions prior to measuring the virial pressure.

The LAMMPS_ELASTIC Python script in iprPy extends the capabilities of the original script by providing the same input parameter interface used by the other iprPy calculations. Doing so makes it possible to run the calculations in high-throughput across the interatomic potentials and initial crystal structure guesses. During testing, it was revealed that the relaxation step of the underlying method may not fully relax crystal structures that are far from a minimum energy state. To address this issue, the iprPy LAMMPS_ELASTIC calculation was updated such that it iteratively runs the underlying method until the lattice constants converge. Results are shown here for both the original single-run version and the newer iterative version to demonstrate the limitations with the original. For clarity, the updated version is referred throughout this paper as “iterative LAMMPS_ELASTIC”.

The dynamic_relax calculation script runs a molecular dynamics simulation at a specified temperature and pressure, allowing the system to evolve over time. Relaxations are performed using nph integration with a Langevin thermostat set for 0 K. This integration scheme adjusts the box dimensions to the given pressure, while dampening out the kinetic energy of the atoms. Relaxed cell parameters are obtained by averaging the instantaneous system dimensions at each timestep after an equilibration time.

The calculation scripts and supporting files are integrated into iprPy, and the descriptions provided here are consistent with iprPy versions 0.6 and 0.7.1. The primary difference between the two versions is that the iterative LAMMPS_ELASTIC became standard in version 0.7.1,

although an option is included that allows the calculations to run without iteration. Simulations were performed on different computing resources using LAMMPS versions 2016-03-11, 2016-09-02, 2016-11-17, and 2017-01-26. XML records for all calculations performed and an archival version of the codebase including calculation scripts, high-throughput tools, and analysis tools consistent with this work can be accessed from the NIST Material Measurement Laboratory data repository server (https://materialsdata.nist.gov/).

3. Results and Discussion

3.1 Initial scan

Figure 1 shows some example plots of cohesive energy versus interatomic spacing as obtained from the E_vs_r_scan calculation method. Out of the 5570 E_vs_r_scan calculations performed, 98 found no energy minima within the scanned range, 4416 found one minimum, and 1056 found multiple minima resulting in a total of 7427 possible crystal structures being identified. It should be noted that this calculation method by itself is insufficient for determining the stability of the possible structures and that many will likely be revealed as unstable upon further refinement. The identification of multiple minima with some of the calculations may seem concerning, but does not necessarily mean that a given potential is of questionable quality. The multiple minima may correspond to unstable configurations, or configurations only accessible under extreme conditions far from the ideal equilibrium.

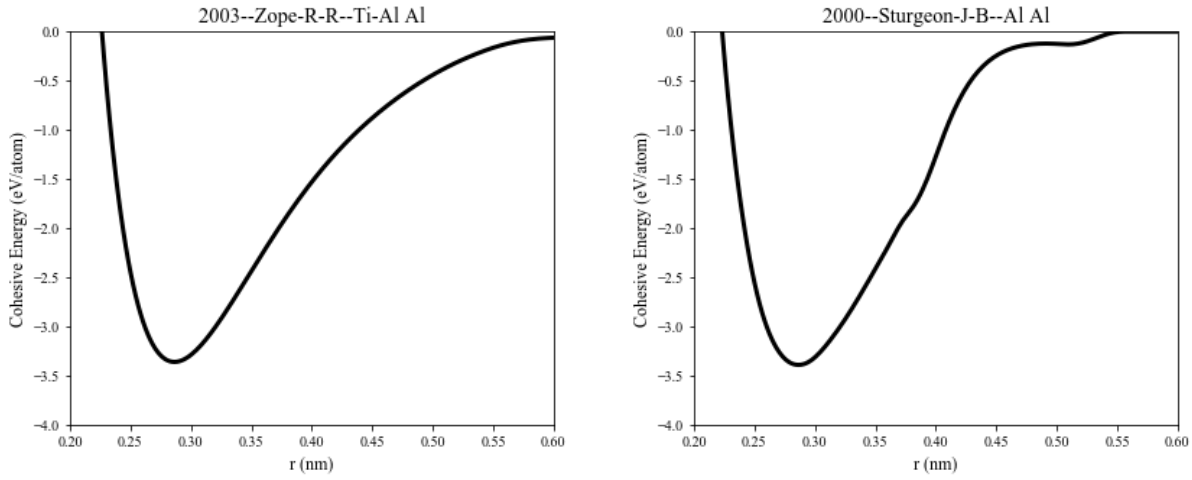

Figure 1: Cohesive energy versus ideal interatomic spacing of A1—Cu—fcc crystal structure for elemental aluminum using (A) 2003--Zope-R-R--Al [6] and (B) 2000--Sturgeon-J-B--Al [12] interatomic potentials. For (B), note the shallow secondary minima around r = 0.51 nm.

3.2 Lattice constant estimates

Twelve different relaxation calculations were performed per possible crystal structure identified from the E_vs_r_scan calculations: one dynamic_relax calculation, five of each of refine_structure and LAMMPS_ELASTIC corresponding to small strain values of 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7} and 10^{-8}, and one iterative LAMMPS_ELASTIC with a small strain value of 10^{-8}. Despite there being more than ten times the number of static calculations as dynamic calculations, performing the dynamic calculations required noticeably more computational hours than performing the static calculations.

Not all of the relaxation calculations finished successfully. For the dynamic_relax method, 124 calculations issued LAMMPS simulation errors consistent with highly unfavorable configurations being explored, resulting in unstable simulations. With LAMMPS_ELASTIC, roughly 30 calculations per strain range issued errors indicative of extremely unstable configurations being explored. The iterative LAMMPS_ELASTIC additionally had around 350 errors associated with a failure of the lattice constants to converge after 100 cycles. The refine_structure method had the most errors by far with between 556 and 832 errors per strain range. Out of the refine_structure errors, roughly half were consistent with a failure to converge the lattice parameters, and the other half being unable to compute the elastic compliance at a given state due to the measured elastic stiffness matrix being singular and non-invertible. Closer examination showed that most of the refine_structure errors were associated with crystal structures likely to be unstable with the explored potentials, e.g. nearly three-quarters of the α-As calculations failed.

Analyzing the lattice parameter results across all the methods, potentials and crystal structures poses a challenging endeavor. For this work, it was realized that the simplest means to identify possible issues is to compare the results obtained with the calculation methods for a specific crystal structure and potential to each other. Any noticeable disagreements in the relaxed lattice constants and cohesive energy values would indicate which calculations warrant a closer examination. For every crystal structure and composition, a plot was made comparing the measured values versus the interatomic potential using every initial configuration, method and strain range. These plots provide a convenient means to compare the predicted properties across both methods and potentials, and assist in discovering trends.

Comparisons of values for a given crystal structure across the different potentials reveals some interesting trends and behaviors (Figures 2-4). With elemental crystals that are expected to

be stable, the measured lattice constants across the potentials mostly fall into unimodal or bimodal distributions, where the bimodal distributions arise from differing reference values used in fitting the potentials. Figure 2 shows the computed lattice constants for body-centered cubic iron, where most potentials predict lattice constants near either the 0 K experimental value of 0.2855 nm or the 300 K experimental value of 0.2866 nm. Observed outliers are associated with potentials where the element was designed only for use in alloys and compounds, potentials where the focus of the fits was on properties other than the lattice constants, and initial states corresponding to alternate energy minima identified along the cohesive energy versus interatomic spacing plots. For the three largest outliers in Figure 2, the 2011—Bonny-G—Fe-Ni-Cr [13] and 2013--Henriksson-K-O-E--Fe-C [14] primarily designed for fcc steels rather than pure bcc iron, and 2012—Proville-L—Fe [15] placed less of a fitting weight on the lattice constants to better capture dislocation properties.

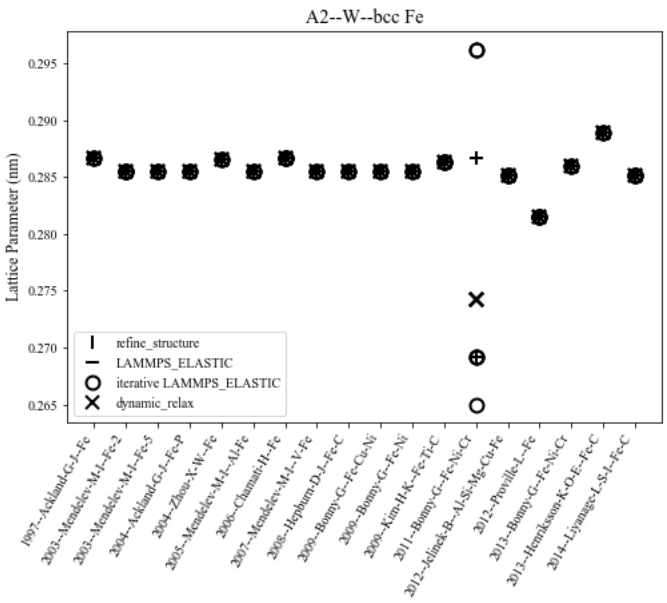

Figure 2: Lattice constant predictions for bcc Fe. Most potentials predict lattice constants near either the 0 K experimental value of 0.2855 nm or the 300 K experimental value of 0.2866 nm. For outliers, 2011—Bonny-G—Fe-Ni-Cr [13] and 2013--Henriksson-K-O-E--Fe-C [14] were primarily designed for fcc steels not bcc iron, and 2012—Proville-L—Fe [15] placed less of a fitting weight on the lattice constants to better capture dislocation properties.

By far, most of the observed disagreements between the methods are associated with the dynamic_relax and iterative LAMMPS_ELASTIC calculations producing lower energy configurations than the refine_structure and original LAMMPS_ELASTIC calculations. Figure 3 shows results for AlNi in the unstable rock salt structure, where the final structures remain cubic

for refine_structure and LAMMPS_ELASTIC, but not for dynamic_relax and iterative LAMMPS_ELASTIC. The reason for this is that the relaxation algorithms for dynamic_relax and LAMMPS_ELASTIC often retain the symmetry conditions of the original system box. If two or three of the dimensions are the same size, the algorithms apply equal adjustments along those box dimensions. The iterative LAMMPS_ELASTIC appears to avoid this issue for many structures, such as the AlNi rock salt structure in Figure 3 where the final relaxed configurations across all potentials are non-cubic.

It should be noted that the general inability of refine_structure and LAMMPS_ELASTIC to break the symmetry of the system may be preferable for certain investigations. They allow for the relative energies of ideal structures to be measured regardless of whether the structure is stable or unstable. This can be useful for comparing the predictions from classical potentials with predictions from quantum potentials where full relaxations may be challenging to obtain.

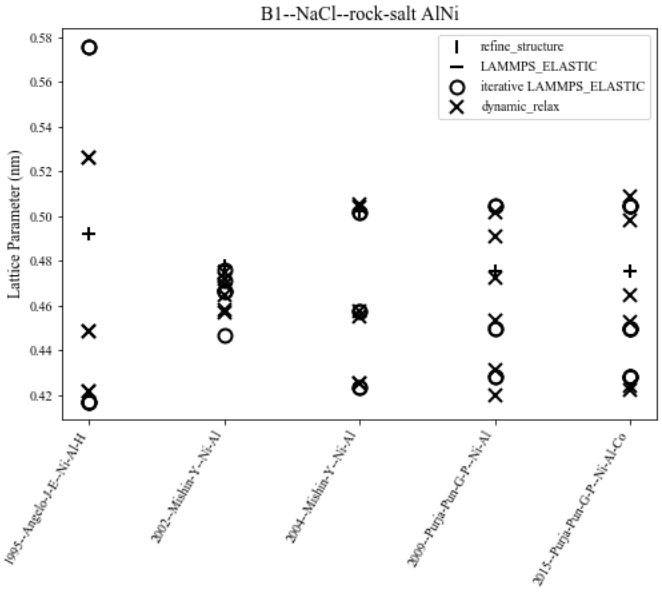

Figure 3: Predictions of B1 AlNi crystal structures. For all potentials, the refine_structure and LAMMPS_ELASTIC methods predict the same stable cubic structures (horizontal and vertical lines form a plus). The dynamic_relax and iterative LAMMPS_ELASTIC methods reveal all potentials relax the structure to non-cubic configurations (multiple 'x' and 'o' markers).

While the iterative LAMMPS_ELASTIC method does a better job at identifying unstable structures than the other static methods, it is still unable to capture some relaxations seen with the dynamic_relax calculations. An example of this is in Figure 2 for the 2011--Bonny-G--Fe-Ni-Cr potential. Two possible body-centered cubic configurations were identified with lattice constants around 0.269 nm and 0.287 nm. With iterative LAMMPS_ELASTIC, the smaller

configuration remains bcc, while the larger relaxes into a tetragonal structure. In contrast, the dynamic_relax method transforms both initial configurations into a distorted bcc structure with lattice constants of 0.274 nm.

Other disagreements are observed that are specific to the static method used. The original LAMMPS_ELASTIC method is seen to have trouble with systems where the initial ideal b/a and c/a ratios are far from the relaxed ratios. Most notably, this is observed with hcp structures of Al and Ti, and with nearly all the bct structures (Figure 4). This is associated with the underlying minimization algorithm, which the LAMMPS documentation notes "is not a mathematically well-defined minimization problem" [16]. It can have issues if the initial dimensions are far from the fully relaxed dimensions. The iterative LAMMPS_ELASTIC corrects this issue and gives lattice constants consistent with the refine_structure and dynamic_relax methods for these structures.

Interesting observations are also made for the refine_structure method. A disproportionately high number of errors (roughly 1/5) are issued for calculations using the α-As prototype. While the other methods do not issue errors, they do show the α-As crystals to be unstable, which is to be expected as the prototype is covalent in nature and most potentials included here lack angular-dependent terms.

There is also observed scatter in the reported lattice constant values across the strain ranges of the refine_structure calculations for the Alkali metals of the 2016--Nichol-A [17] family of potentials. A closer examination revealed issues with the implementations tested for those potentials. The authors were subsequently contacted and replacement versions were added to the Interatomic Potential Repository.

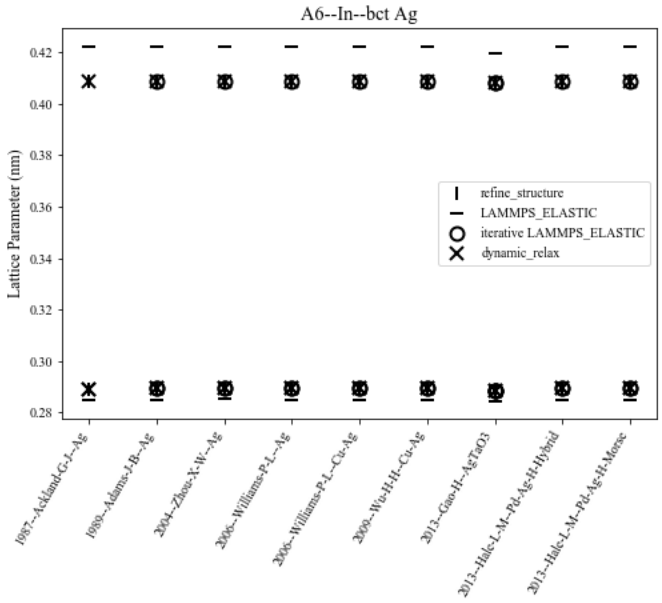

Figure 4: Predictions of bct silver. The initial unrelaxed prototype used the ideal c/a ratio of 1.5. With refine_structure, iterative LAMMPS_ELASTIC, and dynamic_relax, the c/a ratio relaxes to 1.4142 corresponding to the fcc structure. In contrast, the original LAMMPS_ELASTIC gives intermediate c/a ratios indicating it has trouble fully relaxing the box dimensions.

3.3 Elastic constants estimates

Some potentials were observed to have computed elastic constants that were sensitive to the strain range used, most notably the 1987--Ackland-G-J [18], and 2004--Zhou-X-W [19] families of potentials (Figure 5). With these potentials, the scatter is related to the fact that the third derivatives of the potentials' functions are not continuous (i.e., the potentials are not C^4 continuous) near the equilibrium configurations. The small strain method of evaluating the elastic constants is sensitive to these third-order discontinuities, and will give different predictions if the strains sample configurations above, below, or straddling the discontinuities. It should be noted that this sensitivity would likely not be an issue for finite temperature dynamic simulations as the explored configurations of a dynamic simulation average and smooth over the static noise and discontinuities.

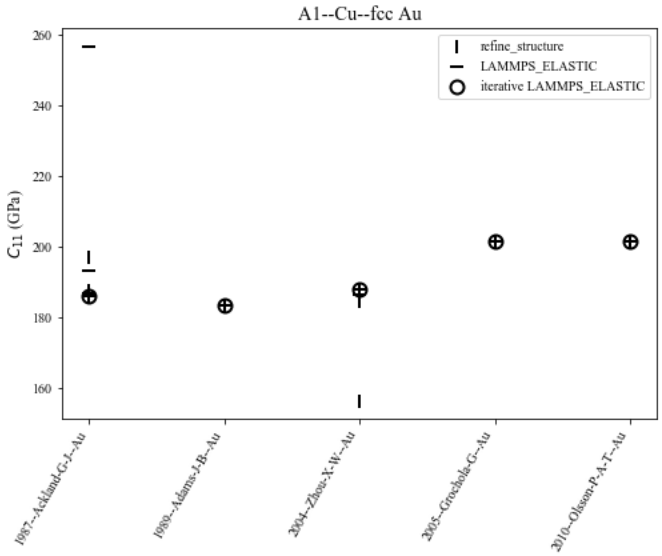

Figure 5: Predicted C_{11} elastic constants for fcc gold. Note that scatter is observed in the values for both the 1987--Ackland-G-J--Au and 2004--Zhou-X-W--Au potentials.

The expression for the pair function term of the 1987--Ackland-G-J potentials changes at the closed-packed interatomic spacing leading to a third-order discontinuity (Figure 6(a)). This results in scatter across the methods and strain ranges for the close-packed prototypes fcc, hcp, and α-La. While other potentials have similar discontinuities in their functional forms, they do not influence the elastic constant measurements performed here since the discontinuities do not correspond to the interatomic spacings of zero pressure equilibrium structures. For the 2004--Zhou-X-W potentials, the underlying functions are C^4 continuous, but the tabulated forms of the functions within the potential files are not and the numerical third derivatives of the potential functions appear wavy (Figure 6(B)). The developer of the potential noted that this is likely due to a precision issue in the code used to generate the potential files, which is planned to be corrected in new implementations.

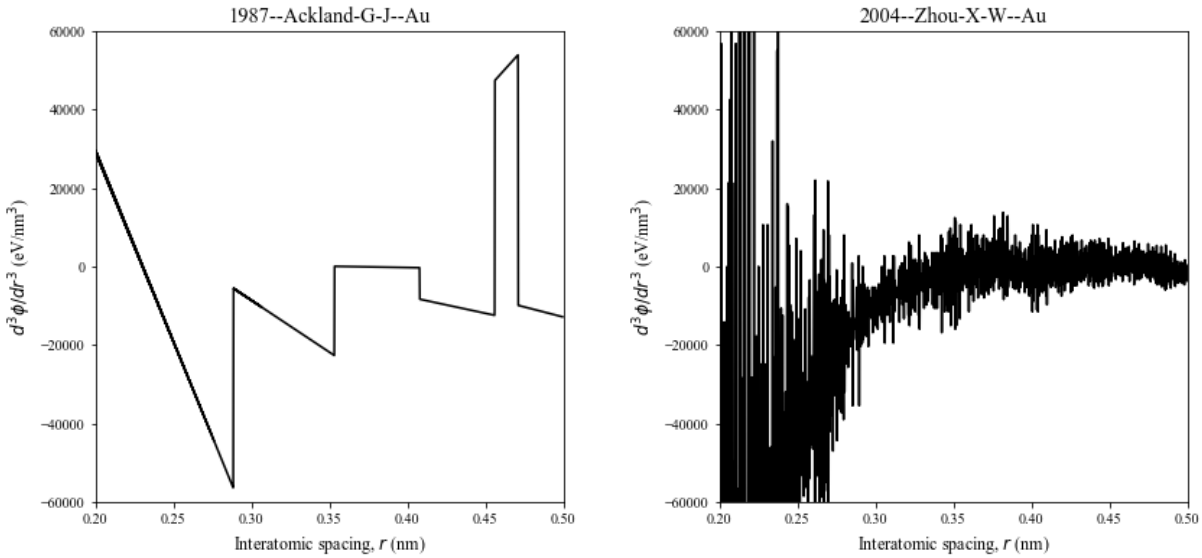

Figure 6: Third derivatives of the EAM pair function term for (A) 1987--Ackland-G-J--Au and (B) 2004--Zhou-X-W--Au potentials showing discontinuities that interfere with static estimates of the elastic constants. With the 1987--Ackland-G-J family of potentials, the functional form is only continuous to the second derivative, and the discontinuity near r = 0.29 nm corresponds to the close-packed interatomic spacing. The functional form of the 2004--Zhou-X-W family of potentials is continuous to the third derivative but the tabulated forms have been numerically transformed resulting in the observed wavy nature.

It should also be mentioned that variation in the elastic constants with different small strain values is possible even with C^4 continuity. The small strain measurements provide estimates for the second-order elastic constants and assume linear elasticity. Large strains may exceed the linear elastic regime of the potential, if it even has one. Conversely, using too small of a strain may introduce error through decreasing numerical precision. To get a quantitative estimate of this effect on the strains sampled here, values of C_{11}, C_{12} and C_{44} for the fcc crystal structure were investigated for all potentials excluding the 2016--Nichol-A, 1987--Ackland-G-J, and 2004--Zhou-X-W families of potentials, and any obvious alternate/unstable configurations. The elastic constants across strains of 10^{-8} to 10^{-5} and both static methods are all within 0.03 GPa of each other, except for the Mg model used by the 1997-- and 1998--Liu-X-Y--Al-Mg [20] potentials where C_{11} varies by roughly 0.1 GPa. Much stronger sensitivities are shown upon reaching a strain of 10^{-4}, where the largest differences in the elastic constants between the 10^{-4} and 10^{-8} strain states are 2.15 GPa for C_{11}, 0.52 GPa for C_{12}, and 0.14 GPa for C_{44}. This indicates that 10^{-4} is too large a strain to consistently predict elastic constants.

Some noticeable disagreements were also observed between the elastic constants specific to certain prototypes and crystal families. With body-centered tetragonal systems (Figure 7), the

elastic stiffness components for the original LAMMPS_ELASTIC differ from refine_structure and iterative LAMMPS_ELASTIC. This is likely due to the original LAMMPS_ELASTIC method not fully relaxing the lattice constants.

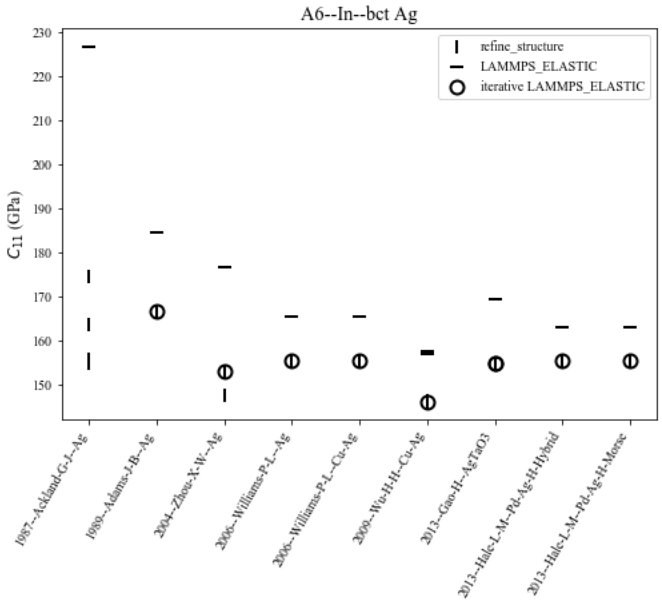

Figure 7: Predicted C_{11} elastic constants for bct silver showing a disagreement between LAMMPS_ELASTIC and the other two methods. Scatter is also observed for the 1987--Ackland-G-J--Ag and 2004--Zhou-X-W--Ag potentials.

In contrast, the C_{11}, C_{12}, and C_{33} components of the hexagonal crystals (Figure 8) and the C_{44} component of the diamond cubic structures show agreement between the two LAMMPS_ELASTIC versions, but not the refine_structure method. Using the silicon model in the 2012--Jelinek-B--Al-Si-Mg-Cu-Fe [21] potential, the diamond cubic C_{44} constant is 77 GPa with LAMMPS_ELASTIC and 250 GPa with refine_structure (experimentally, C_{44} should be around 80 GPa [22]). These results are consistent with previous works showing the importance of internal relaxations on evaluating elastic constants [23-26].

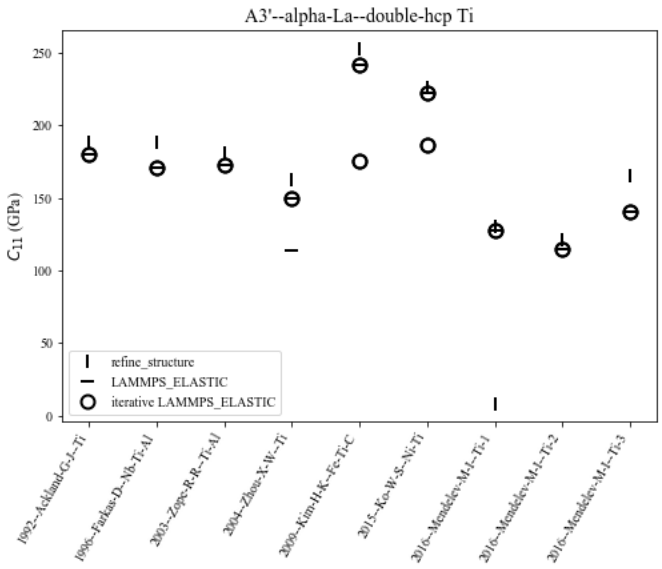

Figure 8: Predictions of C_{11} elastic constants for hcp titanium. Disagreements can be seen between refine_structure and LAMMPS_ELASTIC estimates of C_{11} and C_{12} across all potentials. Scatter and sensitivity to the small strain value can also be seen with the 2004--Zhou-X-W--Ti and 2016--Mendelev_M-I--Ti-1 [27] potentials.

3.4 Calculation methodology discussion

The above results highlight strengths and limitations for the different methodologies. While the refine_structure method provides a quick evaluation of lattice constants for ideal crystal structures, it should not be used to evaluate the elastic constants since it does not perform internal relaxations. The LAMMPS_ELASTIC method does allow for the necessary internal relaxations but can fail to fully relax some configurations. The newer iterative LAMMPS_ELASTIC method overcomes the limitation of the underlying minimization algorithm, and can reveal certain structures to be unstable that the other static methods do not. Finally, the dynamic_relax method provides the most robust test of a structure's stability, but is considerably more computationally expensive and does not provide an estimate of the elastic constants.

One of the useful aspects of the open source framework is that the methods can be improved over time, as was done with the LAMMPS_ELASTIC calculation. An option could be added to both static methods to apply a small perturbation to the initial lattice constant guesses to break any constraining symmetries. LAMMPS_ELASTIC could be further improved by retaining the separate elastic constants estimates for positive and negative strains to check if the

structure is fully relaxed and if there are issues with the potential's derivatives. Finally, dynamic_relax can be improved by performing an energy minimization on the final configuration, or using the final configuration as input to the iterative LAMMPS_ELASTIC calculation. This would remove variations due to dynamic fluctuations and allow for better evaluation of a structure's stability, plus provide elastic constant estimates.

The calculation methods and workflow are observed to be insufficient for robustly identifying certain stable crystal structures. For the prototypes used here, this is most notable for the A6 bct structure. The relaxation calculations will only find one minimum, which may correspond to bcc, fcc, or a stable bct structure. Starting from the ideal c/a=1.5 state, most calculations performed here are observed to relax to fcc. Rigorously identifying any stable bct phases would require an exploration of the bct phase space. This could be accomplished by either adding a calculation specific to investigating Bain transition paths, or expanding E_vs_r_scan to allow multi-dimensional explorations for uncoupled lattice constants.

The number of calculations performed here is probably more than strictly necessary. The high-throughput approach is brute force in that all possible combinations of prototypes, potentials, and element-unique site pairings are explored. Because of prototype symmetries, many of the calculations explored are duplicates (e.g., the order of element-unique site pairings with NaCl doesn't matter, and CsCl with only one element is bcc). The duplicate calculations do, however, provide a verification that the methods and simulation code are behaving as expected. For instance, it was noticed that EAM potentials in the funcfl format (LAMMPS pair_style eam) gave drastically different predictions for identical configurations represented with one atom type versus multiple atom types of the same element. Tests revealed this as being due to a bug introduced in LAMMPS between versions 2016-09-27 and 2016-11-17. This information was passed on to the LAMMPS developers, and the issue should be fixed for version 2017-06-20 and later.

Additionally, many of the crystal structures explored are unstable in the calculations, and rightfully so. Time could be saved by skipping potential-prototype pairings that have incompatible atomic bonding types. For example, EAM potentials lack bond-dependent terms that are necessary for representing the covalent prototypes of diamond cubic, β-Sn, and α-As.

4. Summary and Conclusions

Developing scripts to represent simulation methodologies opens these calculations to many powerful computational and analysis tools. As was demonstrated here, the calculations can be easily implemented and integrated into high-throughput workflows to perform comprehensive investigations across models, configurations, methods and settings. Evaluating predictions across potentials makes it possible to validate whether the models are appropriate for applications of interest by comparing property values to experiments or more robust calculation techniques. Additionally, as was shown here, comparative studies allow for sensitivity analyses of calculation methods and parameters demonstrating method limitations and guiding improvements. Outliers and inconsistent data also assist in verification of the methodology, interatomic models, and simulation software by indicating specific simulations that warrant more detailed investigations.

All the calculation scripts, tools, and results used for this work are openly available. The current version of the iprPy framework, including updated calculation scripts and high-throughput tools, is hosted on GitHub (https://github.com/usnistgov/iprPy). Summaries of property results obtained from these calculations and more are being published on the NIST Interatomic Potentials Repository website (https://www.ctcms.nist.gov/potentials/) for each hosted potential. An archival version of iprPy and the calculation results consistent with this paper can be downloaded from the NIST Material Measurement Laboratory data repository server (https://materialsdata.nist.gov/). The calculation results contained on the data repository include XML records for every calculation performed, and interactive versions of the Figures in this paper for every potential and crystal structure.

Appendix: The iprPy Computational Framework

The iprPy computational framework is focused on the creation and design of open and transparent calculation methodologies that can easily be integrated into high-throughput workflows. It consists of implemented calculation scripts, supporting codebase, and tools and resources for the high-throughput execution of the implemented calculations. All content associated with the iprPy framework is available on GitHub at https://github.com/usnistgov/iprPy/.

A key principle of the framework is to minimize barriers for usage. Python was selected as the primary programming language of the framework due to its open source nature, focus on

clarity of code, and widespread use in the scientific community. The only required software for the core of iprPy to run is Python 2.7, and the list of non-standard Python packages is kept to those that work on any operating system that Python does. All results are generated in an XML/JSON equivalent format making the information directly accessible to both humans and software. Additionally, the supporting codebase provides modular components and common functionalities to help facilitate the rapid development of new calculations.

Calculations form the heart of the iprPy framework and package. Each calculation consists of a "calc.py" calculation script and any other non-variable files that the script accesses. When executed, the calculation script reads in all variable parameters from a simple key-value formatted input parameter file. Upon successful completion, the processed results are saved as either a JSON or XML record allowing for values to be easily read by both humans and computers. Each calculation is designed to be an independent unit of work that can be executed in isolation from any other calculation.

While the framework itself is agnostic to the work that a calculation does, all currently implemented calculations use molecular dynamics simulations. In particular, the atomman Python package (https://github.com/usnistgov/atomman/) is used to create calculation scripts that represent the entire workflow of a property calculation involving LAMMPS [10, 11] simulations. Each of these atomman-based scripts loads or creates an initial atomic configuration, generates a LAMMPS input file from a template by filling in variable terms, runs one or more LAMMPS simulations, and automatically imports the simulation results for post-processing. This design makes it possible for the input and output data to focus on terms that are important to the calculation's purpose instead of the specifics of how any underlying simulations are performed.

Encoding calculation methodologies as Python scripts assists in making the methodologies transparent for both validation and knowledge transfer. A corresponding demonstration Jupyter Notebook is provided for each calculation script that combines working Python code with formatted documentation of the procedure and underlying theory. The code in the Jupyter Notebooks is fully functional; it contains identical underlying functions as the associated calc.py scripts. The code between the two formats only differs in the control of input and output data, with the Notebooks directly receiving and displaying the values as opposed to the calc.py scripts reading from and writing to files.

Using XML/JSON for representing calculation results allows for the associated records to be constructed as modular data models. Instead of defining a completely new schema for each data model from scratch, they can be composed from reusable data types. Each data type represents a complete concept, such as a single value, interatomic potential, or the description of an atomic system. These types can be pieced together like building blocks to define more complex types and concepts, eventually leading to the construction of a full data model schema for a given record style.

There are several advantages to constructing data models in this fashion. First, it allows for new data models to be implemented faster as they can take advantage of the structure and components of existing data models and types. Individual components can also be easily added or modified to define a newer version of a data model if the original version was found to be inadequate. Additionally, this design allows for software to interact with certain components of a data model without requiring that the entire model adhere to a rigid schema. This feature is convenient in that the information contained within a data type can be read from any data model that contains that type. For example, one calculation may compute elastic constants and another use them. If the records for both calculations use the same structured elastic constants data type, then only one function is needed to extract the elastic constants values regardless of the record type.

The design of the calculations and use of XML/JSON records facilitates implementation with high-throughput workflows. Figure 9 shows a schematic overview of how iprPy handles calculations in a high-throughput manner. The scripts “prepare.py”, “runner.py”, and “process.py” allow for a user to interact with records in a database and set up, execute, and analyze calculations. Each calculation is given its own “prepare.py” script which generates calculation instances based on unique combinations of the parameters in the calculation’s input files. Every calculation style has its own unique prepare method to handle the specific combinatorial logic of the calculation’s input parameters. Each prepared instance consists of a folder containing a copy of the calculation script, the completed input file, copies of any other required files, and copies of reference records retrieved from the database. An incomplete record associated with each instance is simultaneously added to the database.

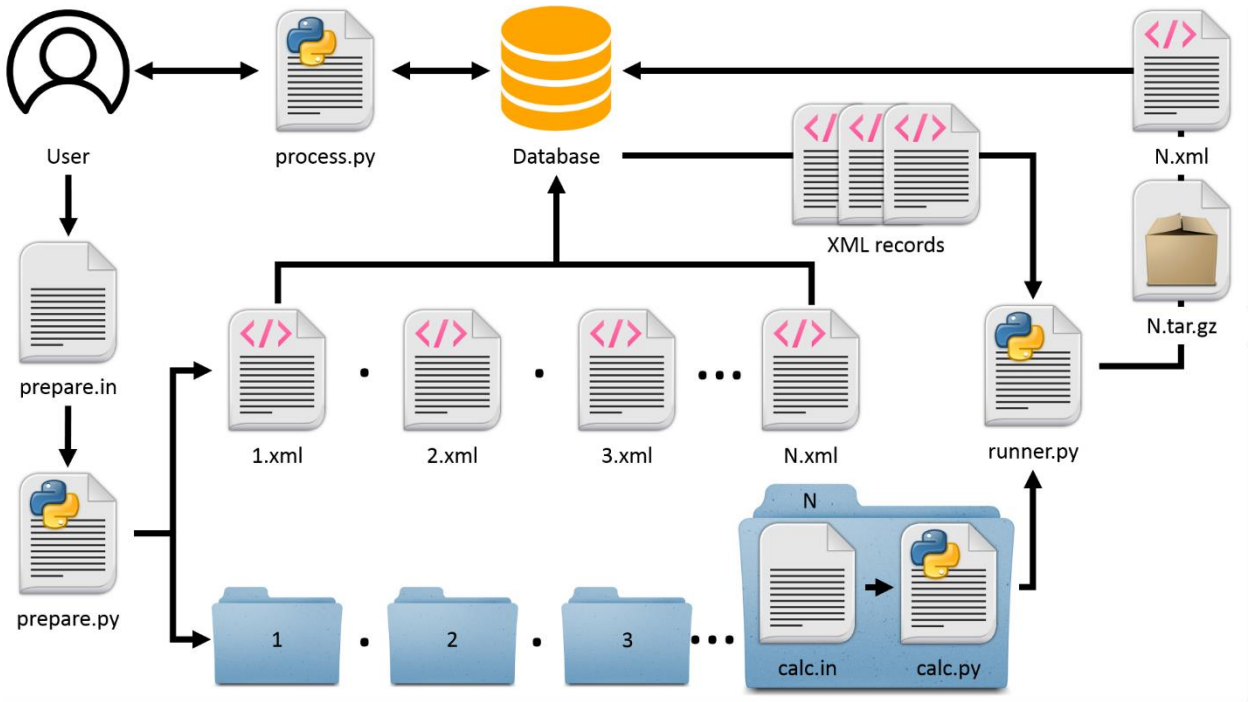

Figure 9: Flow chart for the high-throughput calculation framework. A user creates a prepare.in input file that prepare.py interprets to generate numerous calculation instances and submits a corresponding partial XML record for each instance to the database. Each calculation instance contains a copy of the calculation Python script and an input file for that script. The runner.py script runs each calculation instance using file information in the calculation folder or accessible from the database, and then puts the completed XML record and archived calculation instance into the database. The user can then execute scripts that retrieve and further process the results stored in the database.

The prepared calculations can then be performed in a high-throughput manner using one or more runners by executing the "runner.py" script. Unlike "prepare.py", only one "runner.py" script exists and it operates on all calculations indiscriminately. Each operating runner selects an unfinished calculation instance within a specified directory at random and verifies that no other runners are currently operating on it. The "calc.py" script for that instance is then executed using the associated input parameter file. Upon completion, the corresponding record in the database is updated to a complete record by adding either results or any error message generated by the calculation. The folder in which the calculation instance was performed is compressed into a zipped archive and stored in the database as well.

Finally, "process.py" scripts can be created to extract information from the records in the database. Different scripts can check on the status of prepared calculations, identify and collect errors issued by the calculations, and collect all results for a given calculation style.

This design is simple, but it provides some useful options. By using modular data models, a record produced by one calculation instance can serve as an input reference record for an instance of another calculation. Child calculations can be prepared based on incomplete parent

records allowing the runners to handle hierarchies solely by checking the completeness of any records contained with a calculation instance. Multiple runners can access and add to the same database, and iterate through prepared calculation instances from the same or different root directories. Runners can be started directly, or submitted to a queueing system, and can be assigned multiple cores. Calculations can be performed heterogeneously, with single core runners operating on easy calculations, and multi-core runners operating on the more computationally expensive.

The core codebase of the iprPy framework is designed to support the rapid implementation of new calculations. While each calculation has a unique set of input parameters, there may be common parameters across similar calculations. For instance, all of the current calculations using atomman and LAMMPS have common parameters for specifying the LAMMPS executable, interatomic potential, initial atomic system generation, and default units to use. Any new similar calculations can reuse these common input parameters and the functions used for interpreting them.

The iprPy framework uses classes to define common interactions across different types of calculations, databases, and records. When an object of one of these classes is identified, it is given a style that specifies the particular actions for each common interaction method. For example, the Database class has methods for accessing a database to add or retrieve records. There are currently two styles, local and curator, that make it possible to interact with either a local directory of XML files or an instance of the Materials Database Curation System in the same manner. The styles are treated modularly and new ones can be added by simply placing the associated code within appropriate folders of the framework.

What the iprPy framework provides is a roadmap for designing high-quality research calculations. Each calculation has clearly defined inputs, is a complete and independent unit of work, and produces refined results in a structured format. Both the scripts and results can be easily shared, used and understood external to the framework itself. The Jupyter Notebook versions of the calculations fully document the calculation methodology making it transparent to users who would want to learn how the calculations work. Finally, the calculation scripts and the data records are designed to be implemented into high-throughput workflow tools and databases.

References

[1] C. Becker, NIST Interatomic Potentials Repository. <http://www.ctcms.nist.gov/potentials>, 2007 2016).
[2] C.A. Becker, F. Tavazza, L.E. Levine, Implications of the choice of interatomic potential on calculated planar faults and surface properties in nickel, Philosophical Magazine 91(27) (2011) 3578-3597.
[3] C.A. Becker, F. Tavazza, Z.T. Trautt, R.A.B. de Macedo, Considerations for choosing and using force fields and interatomic potentials in materials science and engineering, Current Opinion in Solid State & Materials Science 17(6) (2013) 277-283.
[4] G. Grochola, S.P. Russo, I.K. Snook, On fitting a gold embedded atom method potential using the force matching method, J Chem Phys 123(20) (2005).
[5] B.J. Lee, A modified embedded atom method interatomic potential for silicon, Calphad 31(1) (2007) 95-104.
[6] M.I. Mendelev, M.J. Kramer, C.A. Becker, M. Asta, Analysis of semi-empirical interatomic potentials appropriate for simulation of crystalline and liquid Al and Cu, Philosophical Magazine 88(12) (2008) 1723-1750.
[7] A. Budi, D.J. Henry, J.D. Gale, I. Yarovsky, Comparison of embedded atom method potentials for small aluminium cluster simulations, Journal of Physics-Condensed Matter 21(14) (2009).
[8] H.H. Wu, D.R. Trinkle, Cu/Ag EAM potential optimized for heteroepitaxial diffusion from ab initio data, Computational Materials Science 47(2) (2009) 577-583.
[9] Z.T. Trautt, F. Tavazza, C.A. Becker, Facilitating the selection and creation of accurate interatomic potentials with robust tools and characterization, Model Simul Mater Sc 23(7) (2015).
[10] S. Plimpton, Fast Parallel Algorithms for Short-Range Molecular-Dynamics, J Comput Phys 117(1) (1995) 1-19.
[11] LAMMPS Molecular Dynamics Simulator. <http://lammps.sandia.gov>).
[12] J.M. Winey, K. Alison, Y.M. Gupta, Thermodynamic approach to determine accurate potentials for molecular dynamics simulations: thermoelastic response of aluminum, Model Simul Mater Sc 18(2) (2010) 029801.
[13] G. Bonny, N. Castin, D. Terentyev, Interatomic potential for studying ageing under irradiation in stainless steels: the FeNiCr model alloy, Model Simul Mater Sc 21(8) (2013) 085004.
[14] K.O. Henriksson, C. Bjorkas, K. Nordlund, Atomistic simulations of stainless steels: a many-body potential for the Fe-Cr-C system, J Phys Condens Matter 25(44) (2013) 445401.
[15] L. Proville, D. Rodney, M.C. Marinica, Quantum effect on thermally activated glide of dislocations, Nat Mater 11(10) (2012) 845-9.
[16] <http://lammps.sandia.gov/doc/fix_box_relax.html>, (accessed Feb 1, 2018.).
[17] A. Nichol, G.J. Ackland, Property trends in simple metals: An empirical potential approach, Physical Review B 93(18) (2016) 184101.
[18] G.J. Ackland, R. Thetford, An Improved N-Body Semiempirical Model for Body-Centered Cubic Transition-Metals, Philos Mag A 56(1) (1987) 15-30.
[19] X.W. Zhou, R.A. Johnson, H.N.G. Wadley, Misfit-energy-increasing dislocations in vapor-deposited CoFe/NiFe multilayers, Physical Review B 69(14) (2004).
[20] X.Y. Liu, P.P. Ohotnicky, J.B. Adams, C.L. Rohrer, R.W. Hyland, Anisotropic surface segregation in Al-Mg alloys, Surf Sci 373(2-3) (1997) 357-370.

[21] B. Jelinek, S. Groh, M.F. Horstemeyer, J. Houze, S.G. Kim, G.J. Wagner, A. Moitra, M.I. Baskes, Modified embedded atom method potential for Al, Si, Mg, Cu, and Fe alloys, Physical Review B 85(24) (2012).
[22] H.J. McSkimin, W.L. Bond, E. Buehler, G.K. Teal, Measurement of the Elastic Constants of Silicon Single Crystals and Their Thermal Coefficients, Phys Rev 83(5) (1951) 1080-1080.
[23] M.I. Baskes, Modified embedded-atom potentials for cubic materials and impurities, Phys Rev B Condens Matter 46(5) (1992) 2727-2742.
[24] M.I. Baskes, R.A. Johnson, Modified Embedded-Atom Potentials for Hcp Metals, Model Simul Mater Sc 2(1) (1994) 147-163.
[25] R.A. Johnson, Internal relaxation in the HCP lattice, Model Simul Mater Sc 1(5) (1993) 717.
[26] L. Kleinman, Deformation Potentials in Silicon .1. Uniaxial Strain, Phys Rev 128(6) (1962) 2614-2621.
[27] M.I. Mendelev, T.L. Underwood, G.J. Ackland, Development of an interatomic potential for the simulation of defects, plasticity, and phase transformations in titanium, The Journal of Chemical Physics 145(15) (2016) 154102.