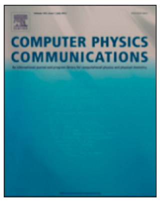
Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc



pyHMA: A VASP post-processor for precise measurement of crystalline anharmonic properties using harmonically mapped averaging[☆,☆☆]

Sabry G. Moustafa, Apoorva Purohit, Andrew J. Schultz, David A. Kofke[*]

Department of Chemical and Biological Engineering, University at Buffalo, The State University of New York, Buffalo, NY 14260-4200, USA

ARTICLE INFO

Article history:
Received 14 February 2020
Received in revised form 8 July 2020
Accepted 29 July 2020
Available online 25 August 2020

Keywords:
Mapped averaging
Harmonically mapped averaging
Anharmonic properties
DFT
VASP
ab initio molecular dynamics
Crystalline systems

ABSTRACT

We introduce a new Python package (pyHMA) that interfaces with VASP to compute (classical) anharmonic properties of crystalline systems by post-processing data from NVT Born–Oppenheimer *ab initio* molecular dynamics (AIMD) simulation. It is based on the recently developed harmonically mapped averaging (HMA) method, which leverages the analytically known harmonic behavior to reformulate the direct/conventional ensemble averages in order to significantly improve precision, for a given CPU time. The package consists of two stages: reading AIMD data from vasprun.xml file(s) and then computing anharmonic properties. While the first stage is MD package-dependent, the second one is universal, given that it receives data in the required format. To demonstrate the usage of pyHMA, we compute anharmonic energy and pressure of aluminum fcc crystal at high pressure ($\approx$ 115 GPa) and up to 4000 K (near melting). We further compute anharmonic free energy as a function of temperature, using thermodynamic integration of the HMA anharmonic energy. Although pyHMA currently interfaces with VASP to compute HMA anharmonic energy and pressure, it is moduled in such a way to allow for interfacing with other codes (e.g., LAMMPS) by adding a new reader and can compute other HMA anharmonic properties (e.g., heat capacity) by adding a new method, once relevant data are available.

Program summary
Program title: pyHMA
CPC Library link to program files: http://dx.doi.org/10.17632/bzgfk52msk.1
Licensing provisions: MPL-2.0
Programming language: Python 3.7
Nature of problem: Theormodynamic properties (e.g., energy, pressure, and heat capacity) of crystalline systems can be decomposed into: lattice (or, property at 0 K), quasiharmonic, and anharmonic contributions. Although the first two are feasible to compute using only a few single-point density functional theory (DFT) calculations, measuring anharmonic contribution requires running *ab initio* molecular dynamics (AIMD) simulation, which is computationally very demanding using direct ensemble averaging.
Solution method: In pyHMA, we are adopting the harmonically mapped averaging (HMA) technique that provides order(s) of magnitude higher precision, in comparison to direct/conventional (Conv) averaging. The package works as a post-processor to VASP AIMD output to provide very precise (and accurate) estimate of anharmonic properties, for a given DFT model, with application to energy and pressure (at this time).
Additional comments: The term *anharmonicity* is commonly used in literature to qualitatively describe a system with no equilibrium configuration at 0 K (i.e., imaginary frequencies); in other words, it refers to a “non-harmonic" potential-energy surface. Here, however, we define *anharmonic contribution* of some property X as the residual in excess of the harmonic approximation; $X_{\rm ah} \equiv X - (X_{\rm lat} + X_{\rm qh})$. Therefore, this specific definition is meaningless if the system does not have equilibrium lattice configuration at 0 K. For this reason, pyHMA checks forces on the first configuration to make sure the system has an equilibrium configuration (i.e., zero forces). In addition, when using pyHMA to measure anharmonic free energy using thermodynamic integration from 0 K (Sec. 3.3), only ground-state DFT must be used; using finite-temperature DFT (i.e., Fermi–Dirac smearing; ISMEAR=-1 and SIGMA=$k_{\rm B}T$), as often done

☆ The review of this paper was arranged by Prof. Hazel Andrew.
☆☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).
* Corresponding author.
E-mail address: kofke@buffalo.edu (D.A. Kofke).

https://doi.org/10.1016/j.cpc.2020.107554

with metals, cannot be used as the PES in this case is temperature-dependent, which is not accounted for in the integration. This contribution, however, can still be included using free-energy perturbation methods as described elsewhere [1]. On the other hand, for properties that do not require temperature integration (e.g., energy and pressure), pyHMA reads the electronic free-energy surface (F), rather than the ground-state energy (E0); hence, electronic contribution is accounted for.

1. Introduction

The harmonic approximation is a very effective starting point for estimating the properties of crystalline systems. In it, the potential-energy surface (PES) is estimated via a second-order series in nuclear displacements, yielding a Hamiltonian that is tractable for evaluation of dynamic and thermodynamic behavior [1]. This leads naturally to the following decomposition of a thermodynamic property X:

$$X = X_{\mathrm{lat}} + X_{\mathrm{qh}} + X_{\mathrm{ah}} \tag{1}$$

Here, lat indicates the static lattice contribution to the property (when all atoms are at their lattice sites); qh is the quasiharmonic contribution (given via the approximation just described); and ah indicates the anharmonic contribution, which exactly corrects the harmonic approximation. The lattice and quasiharmonic contributions are readily evaluated using standard total-energy methods, calculation of the Hessian for the perfect lattice, and some linear algebra. In contrast, *ab initio* molecular dynamics (AIMD) simulation is required to compute the anharmonic contribution, and as a consequence it is often neglected completely due to the high computational cost. Note that this decomposition is valid whether ground- (0 K) or excited-state (Fermi–Dirac statistics) is used; however, especial treatment is needed with the latter when computing anharmonic free energy (see Section 3.3).

The harmonic treatment can fail (X_{ah} is not defined) for systems that are unstable at $T = 0$ K; i.e., having phonon modes with imaginary frequencies. In other cases, the approach becomes inaccurate (X_{ah} is significant) when large displacements from the minimum-energy lattice configuration are relevant, such as occurs at high temperatures, or for systems having soft degrees of freedom (often seen in molecular crystals). A number of approximate methods have been developed to estimate the properties while capturing these anharmonic effects. Examples include self-consistent phonon theory (SCPT) [2–5], vibrational self-consistent field (VSCF) [6–9], VCSF with vibrational configuration-interaction (VCI) [10–12], self-consistent *ab initio* lattice dynamics (SCAILD) [13], stochastic self-consistent harmonic approximation (SSCHA) [14], temperature-dependent effective potential (TDEP) [15–17], and variational methods [18]. In two recent papers [19,20], some of these methods, and others, have been examined for their effectiveness in estimating the anharmonic contributions to the free energy and other properties.

While approximate methods are useful and play an important role in the study of crystalline systems, there are many situations where one wants to apply methods that have, for a given molecular model, no inherent inaccuracy and that can yield properties with low uncertainty. Such methods require sampling of nuclear configurations by molecular dynamics or Monte Carlo simulation, and collection of appropriate averages over the sampled coordinates. Sampling such a large number of configurations can be very expensive, particularly for *ab initio* potentials; hence, it is valuable to have approaches that can complete this task as efficiently as possible.

In this regard, we recently proposed and demonstrated the "mapped-averaging" framework [21,22] that allows approximate theoretical results derived from statistical mechanics to be reintroduced into the underlying formalism, yielding reformulated ensemble averages that are rigorous (contain no approximation) and allow direct evaluation of the correction to the theory by molecular simulation. To the extent that the theory is accurate, this correction will be small, and hence measured with small uncertainty. Thus, by using these reformulated ensemble averages, accurate and precise values of thermodynamic properties can be obtained while using less computational effort, sometimes far less.

When applied to the simulation of crystalline systems, we refer to this method as harmonically mapped averaging (HMA), as it involves transformation/mapping of coordinates leveraging the known harmonic behavior [21]. An HMA reformulated ensemble average removes the known harmonic behavior from the conventional ensemble average (which we label Conv), yielding a direct measurement of the anharmonic contribution X_{ah} without contamination by noise produced by the already known harmonic behavior. In previous applications [21–25], we have obtained property values to a given precision with at least one to two orders of magnitude less computational effort (relative to the Conv averaging), depending on the property and the thermodynamic state.

Additionally, we have made observations that relate to the accuracy (as distinct from precision) of the calculated properties. Here, accuracy is in reference to the true thermodynamic behavior for a given molecular model (i.e., apart from the question of whether the molecular model is an accurate representation of the physical system being modeled). First, anharmonic contributions X_{ah} to properties are intrinsically (regardless the method of measurement) much less sensitive to certain parameters of the simulation, in comparison to the full property value. Specifically, for a given molecular model, a given accuracy in X_{ah} may be obtained using a smaller system size (fewer atoms), and/or a shorter potential truncation radius, relative to what is required to obtain the same accuracy in X itself [21,23,26,27]. These benefits can be realized most effectively by having the HMA framework available to provide X_{ah} directly, i.e., without requiring it be obtained by subtracting $X_{\mathrm{lat}} + X_{\mathrm{qh}}$ from (noisy) full averages. Further, we have observed additional advantages in X_{ah} specific to its calculation via HMA (in comparison to Conv), including the ability to use a larger molecular dynamics time step with less loss of accuracy, and faster equilibration and decorrelation of properties [21,23,25].

Some other benefits of focusing on X_{ah} and computing it via HMA, connecting in particular to modeling with electronic density functional theory (DFT), are described in Section 3.

The observations given above pertain to the class of systems we have studied to date. These are mainly lattices of monatomic molecules with no rotational or internal degrees of freedom. The efficiencies we observe may extend to more complex systems, but this determination awaits further study.

Importantly, HMA does not require any alteration in how sampling is performed during the simulation, so it may be used with standard Monte Carlo or molecular dynamics methods; implementation requires only recording and processing of appropriate data calculated on the sampled configurations. To aid with this, we present in this paper a Python package, called pyHMA, that

Table 1
Conventional and HMA expressions for ensemble averages of anharmonic energy and pressure for crystalline systems. The number density is denoted by ρ, and the virial and quasiharmonic pressures are defined as $\langle P_{\mathrm{vir}}\rangle \equiv -\langle \partial U/\partial V\rangle$ and $P_{\mathrm{qh}} \equiv -\partial A_{\mathrm{qh}}/\partial V$, respectively.

Anharmonic property	Conventional (Conv)	Harmonically Mapped Averaging (HMA)
Energy, U_{ah}	$\langle U\rangle - \frac{3}{2}(N-1)k_{\mathrm{B}}T - U_{\mathrm{lat}}$	$\left\langle U + \frac{1}{2}\mathbf{F}\cdot\varDelta\mathbf{r}\right\rangle - U_{\mathrm{lat}}$
Pressure, P_{ah}	$\rho k_{\mathrm{B}}T + \langle P_{\mathrm{vir}}\rangle - P_{\mathrm{qh}} - P_{\mathrm{lat}}$	$\left\langle P_{\mathrm{vir}} + \frac{P_{\mathrm{qh}}-\rho k_{\mathrm{B}}T}{3(N-1)k_{\mathrm{B}}T}\mathbf{F}\cdot\varDelta\mathbf{r}\right\rangle - P_{\mathrm{lat}}$

reads the VASP vasprun.xml files obtained from the canonical AIMD simulations, and evaluates the thermodynamic properties like pressure and energy with according to the HMA framework. It is fast, and straightforward to apply, allowing VASP users to readily exploit the benefits offered by HMA. We hope this development opens up new possibilities for first-principles modeling of crystalline systems.

The next section presents the details of the reformulated ensemble averages and how that are implemented in pyHMA. To demonstrate the efficiency of pyHMA, Section 3 presents calculations of the thermodynamic energy, pressure and free energy of aluminum using VASP AIMD simulations. It is worth emphasizing that because nuclear motion is treated using Newtonian dynamics, the measured properties do not include quantum effects (e.g., zero-point energy). We compare the computational efficiency of the results obtained using HMA and conventional methods. The last section provides a summary as well as the conclusions.

2. Methodology

2.1. Harmonically mapped averaging (HMA) method

In the Conv approach to computing anharmonic properties of crystals, the anharmonic energy U_{ah} and pressure P_{ah} are evaluated by simply subtracting the lattice and quasiharmonic contributions from the full ensemble average (indicated by $\langle\ldots\rangle$),

$$U_{\mathrm{ah}}^{\mathrm{Conv}} = \langle U\rangle - U_{\mathrm{lat}} - \frac{3}{2}(N-1)k_{\mathrm{B}}T \text{ , and} \quad (2)$$

$$P_{\mathrm{ah}}^{\mathrm{Conv}} = \langle P\rangle - P_{\mathrm{lat}} - P_{\mathrm{qh}}, \quad (3)$$

where $\frac{3}{2}(N-1)k_{\mathrm{B}}T$ is the harmonic average energy, with T the set temperature and k_{B} the Boltzmann constant, and P_{qh} is given by $P_{\mathrm{qh}} = -\partial A_{\mathrm{qh}}/\partial V$, where A_{qh} is the quasiharmonic free energy (given by Eq. (5)).

There is a clear inefficiency in stochastically measuring the full energy $\langle U\rangle$ and pressure $\langle P\rangle$ when only the anharmonic contribution is of interest, as an analytical expression for the dominant quasiharmonic contribution is already known (but still fluctuates in contributing to the average). HMA is available to remedy this issue [21,22]. From it, for example, the anharmonic energy is given directly by,

$$U_{\mathrm{ah}}^{\mathrm{HMA}} = \langle U + \frac{1}{2}\mathbf{F}\cdot\varDelta\mathbf{r}\rangle - U_{\mathrm{lat}} \quad (4)$$

where the vector $\mathbf{F}$ represents the forces on all atoms, and $\varDelta\mathbf{r}$ are the displacements of each atom from its lattice (equilibrium) site. To illustrate that this expression represents the anharmonic contribution, let us consider a perfectly harmonic system. In this model, $-\frac{1}{2}\mathbf{F}\cdot\varDelta\mathbf{r}$ equals $U - U_{\mathrm{lat}}$ for each configuration, and the quantity being averaged vanishes; hence, this expression represents the anharmonic contribution to the energy. A similar HMA expression for anharmonic pressure exists and is given in Table 1. Note that the virial pressure at each configuration P_{vir} (defined as $-\partial U/\partial V$) is usually implemented in standard MD codes (e.g., VASP and LAMMPS). More details on these HMA expressions can be found elsewhere [21].

2.2. pyHMA package

Like the conventional average, the HMA method provides only a reformulated ensemble average and does not affect the sampling algorithm (e.g., MD or MC); therefore, ensemble averages can be obtained through post-processing of MD outputs. We implemented the HMA method in a form of a Python package (pyHMA) that interfaces with the widely used VASP DFT code. The package reads (e.g., configurations and energy) data from vasprun.xml XML output file(s) to compute anharmonic energy and pressure, using both Conv and HMA formulations. In addition to reading VASP data, pyHMA is moduled in such a way to read data from other MD codes (e.g., LAMMPS) by just adding a new "reader" as we show below. Similarly, in addition to energy and pressure, the code is readily extended to measure other properties (e.g., isochoric heat capacity, C_V [21]), if necessary data (viz, instantaneous Hessian matrix) are available.

The pyHMA package can be downloaded from the development version on GitHub (https://github.com/etomica/mapped-averaging), or from a release source package on the Python Package Index (PyPI) manager (https://pypi.org/project/pyhma/). The package can be installed from PyPI using the pip command (Python 3.x):

```
$  pip install pyhma
```

Detailed documentation on the HMA theory and pyHMA package is hosted on Read the Docs website (https://pyhmadocs.readthedocs.io/).

Fig. 1 shows the overall structure of the pyHMA package, while Listing 1 shows detailed implementation of these steps interactively using Python interpreter. The code consists of two stages: reading MD data from vasprun.xml file(s) and then using the output to compute anharmonic energy and pressure (using Conv and HMA methods). In the first stage, the read(['vasprun.xml']) function (exists in vasp_reader.py module) parses the vasprun.xml file to extract the following data, and saves them to a dictionary (data): box_row_vecs (box edge row vectors, in Å), num_atoms (total number of atoms), volume_atom (volume per atom, in Å^3/atom), basis (atomic fractional positions of first configuration), position (instantaneous atomic fractional positions), force (instantaneous atomic forces, in eV/Å), energy (instantaneous ground-state potential energy E0 for ISMEAR$\neq$ −1, or electronic free energy F for ISMEAR=-1, in eV/atom), pressure (instantaneous pressure, in GPa), pressure_ig (ideal-gas pressure, in GPa), timestep (MD timestep, in fs), and temperature (set temperature, in K). The function handles a sequence of vasprun.xml files from the same simulation; for example, read(['vasprun-1.xml','vasprun-2.xml']). In this case, the output configuration from the first run (CONTCAR) must be the input (POSCAR) for the second run. It is worth emphasizing that we use LXML parser with the capability to recover broken XML files (lxml.etree.XMLParser(recover=True)). Therefore, the read() function can handle incomplete vasprun.xml file(s) generated from interrupted AIMD runs (by the user, or due to some time constraint).

The read() function takes the following optional arguments as well: force_tol, raw_files, fermi_dirac, and verbose. The value of force_tol (default is 0.001 eV/Å) is the maximum magnitude of the force allowed on any atom of the initial configuration in the first vasprun.xml file, which is supposed to be the equilibrium one (i.e., zero forces). However, if read() function detects force(s) larger than force_tol, it will be interrupted and prints a warning with list of atoms having large initial forces. The second argument, raw_files (default is False), directs pyHMA to generate the following files that it reads from

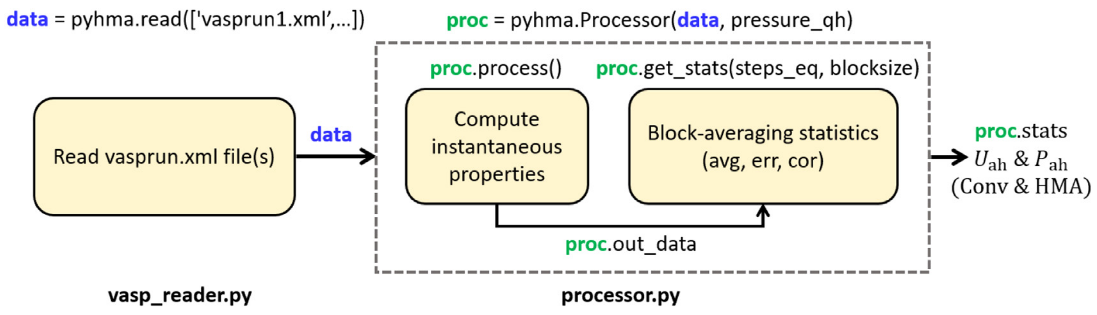

data = pyhma.read(['vasprun1.xml',...])
proc = pyhma.Processor(data, pressure_qh)
proc.process()
proc.get_stats(steps_eq, blocksize)
Read vasprun.xml file(s)
data
Compute instantaneous properties
Block-averaging statistics (avg, err, cor)
proc.stats
U_{ah} & P_{ah}
(Conv & HMA)
proc.out_data
vasp_reader.py
processor.py

Fig. 1. Overall structure of pyHMA package, showing its two stages. In the first stage (vasp_reader.py module), pyHMA extracts relevant MD information from vasprun.xml files(s) and saves them to a data dictionary, with the following keys: box_row_vecs (Å), num_atoms, volume_atom (Å^3/atom), basis (fractional), position (fractional), force (eV/Å), energy (eV/atom), pressure (GPa), pressure_ig (GPa), timestep (fs), and temperature (K). In the second stage (processor.py module), a Processor instance (proc) of the Processor class is created using the data dict and a user-specified quasiharmonic pressure (pressure_qh). The instantaneous (Conv and HMA) anharmonic energy and pressure are then computed by calling proc.process() method, which saves the output into a proc.out_data list. Finally, using block-averaging technique, ensemble statistics (averages, stochastic uncertainties, and adjacent block correlations) are computed by calling the proc.get_stats() method.

XML file(s): poscar_eq.dat (initial POSCAR), posfor.dat (instantaneous atomic positions (fractional) and force (eV/atom)), energy.dat (instantaneous ground-state potential energy E0 for ISMEAR≠ −1, or electronic free energy F for ISMEAR=−1, in eV/atom), pressure.dat (instantaneous pressure, GPa). The third argument, fermi_dirac (default is False), directs pyHMA to read the electronic free-energy surface (F), instead of the ground-state energy (E0), when using AIMD simulation with Fermi–Dirac statistics (ISMEAR=−1 and SIGMA= $k_B T$) to compute free-energy-derivative properties (e.g., energy and pressure). It is worth noting that this simulation cannot be used directly to measure anharmonic free energy using thermodynamic integration (see Section 3.3). The last argument, verbose (default is False), makes pyHMA print (to the console) the positions and forces of the initial configuration to verify the data pyHMA read. Line 5 of Listing 1 shows the full usage of the read() function.

In the second stage of pyHMA, the data are analyzed to compute anharmonic energy and pressure. This is accomplished, first, by creating an object (proc) of the Processor class using the data dict in addition to the quasiharmonic pressure (pressure_qh, in GPa) at the given T, for HMA pressure calculations. The units of anharmonic energy can be set to meV/atom, instead of the pyHMA's eV/atom default, by passing meV=True to the Processor constructor — see line 10 of Listing 1. The instantaneous, Conv and HMA, anharmonic properties are then obtained by calling proc.process() method (see line 13 of Listing 1). The method takes two optional arguments, steps_tot and verbose; the former is the number of steps to be used (default is steps found in vasprun.xml files) and the latter is to direct pyHMA to print simulation details while running. The output is saved to a 2D array (proc.out_data) of length equal to the number of MD steps and contains four columns: Conv and HMA anharmonic energies and pressures. The method also generates energy_ah.out and pressure_ah.out output files for the anharmonic energy (eV/atom; or meV/atom if meV=True) and pressure (GPa), respectively. Each file contains three columns; time (in fs), Conv, and HMA estimates of the property.

Finally, using block averaging technique, statistics (average, uncertainty, and correlation) of ensemble averages are obtained by calling proc.get_stats() method. The method takes two required (steps_eq and blocksize) and one optional (verbose) arguments. The steps_eq argument sets the number of equilibration steps, blocksize sets the number of steps in each block, and verbose (default is False) prints to the console information about the steps used in the block averaging technique. The method returns the statistics output in a form of a dictionary (stats) of four entries: Conv and HMA anharmonic energies (e_ah_conv and e_ah_hma) and pressures (p_ah_conv and p_ah_hma), each with three elements of average (avg), uncertainty (err), and adjacent blocks correlation (cor). The output can be presented in a more user-friendly format by using proc.print_stats() method, which yields the output shown after line 43 of Listing 1.

Table 2
Summary of arguments used by pyHMA's methods and script. See Listings 1 and 2 for usage.

Required	Optional	Default
pressure_qh (GPa)	force_tol	0.001 eV/Å
steps_eq	steps_tot	Steps in vasprun.xml
blocksize	raw_files	False
vasprun.xml file(s)	meV	False
	verbose	False
	fermi_dirac	False

Both stages can be invoked directly from the command line using a script named pyhma, which has the same options to those used with the interactive usage (except for the usage of -r and -v short options). Listing 2 shows the usage of the script, with application to the same system used in Listing 1. Table 2 provides a list of all arguments used by the pyHMA package.

It is worth emphasizing that the correlation should be as small as possible (less than ≈ 0.2) to ensure accurate estimate of uncertainty. Although increasing the block size length (blocksize) reduces the correlations, the number of blocks should be large enough ($\gtrsim$ 50) to yield meaningful statistics.

3. Example application

3.1. Model and computational details

We modeled fcc aluminum crystal at high pressure (with P_{lat} = 114.4 GPa) and up to 4000 K (near-melting temperature [28]) as a test case for using the pyHMA package. All computations are performed using the DFT method as implemented in the Vienna *ab initio* simulation package (VASP, version 5.3.5) [29]. The projector-augmented wave (PAW) potential approach [30] is used, with 3 electrons treated as valence, and the exchange–correlation functional is described by the Perdew–Burke–Ernzerhof (PBE) [31] generalized-gradient approximation. To ensure accuracy of 1 meV/atom and 0.1 GPa for the anharmonic energy and pressure, respectively, we use a plane-wave cutoff energy of 300 eV and a Γ-centered Monkhorst–Pack k-points mesh of size 4 × 4 × 4. First-order Methfessel–Paxton smearing method is used, with σ = 0.2 eV.

```
>>> import pyhma
>>> # Read 'vasprun.xml' files and save them to 'data' dictionary.
>>> # Optional args defaults: force_tol=0.001 (eV/A), raw_files=False, fermi_dirac=False, and
    verbose=False.
>>> # raw_files=True makes: poscar_eq.dat, posfor.dat, energy.dat, and pressure.dat.
>>> data = pyhma.read(['vasprun-1.xml', 'vasprun-2.xml'], force_tol=0.002, raw_files=True,
    fermi_dirac=False, verbose=False)
>>> # data.keys() = dict_keys(['box_row_vecs', 'num_atoms', 'volume_atom', 'basis', 'position',
    'force', 'energy', 'pressure', 'pressure_ig', 'timestep', 'temperature', 'ismear'])
>>> # Create Processor object to analyze the data.
>>> # Optional arg default: meV=False (i.e., energy is given in eV).
>>> # 'pressure_qh' is a required quasiharmonic pressure (GPa) for HMA pressure.
>>> proc = pyhma.Processor(data, pressure_qh=4.94525, meV=True)
>>> # Compute instantaneous anharmonic energy and pressure (using Conv and HMA methods).
>>> # Optional args defaults: verbose=False and steps_tot = MD steps from 'vasprun.xml'.
>>> proc.process(steps_tot=10000, verbose=True)

Simulation data
===============
 Ground-state DFT calculations (E0) using ISMEAR=1
 Set temperature        (K): 1000.00000
 Volume          (A^3/atom):   10.00000
 MD timestep           (fs):    2.00000
 Lattice energy  (eV/atom):   -2.21324
 Harmonic energy (eV/atom):    0.12522
 Lattice pressure     (GPa):  114.44281
 Harmonic pressure    (GPa):    4.94525

 Found 11036  total MD steps
 Using 10000  user-set MD steps

 Computing instantaneous properties ...

>>> # Perform block averaging statistics on the production steps (steps_tot - steps_eq).
>>> # Optional arg default: verbose=False
>>> stats = proc.get_stats(steps_eq=1000, blocksize=90, verbose=True)

Block averaging statistics
==========================
 9000 production steps (after 1000 equilibration steps)
 100 blocks (blocksize = 90  steps)

 Computing statistics ...

>>> # Print anharmonic energy and pressure (using Conv and HMA methods).
>>> proc.print_stats(stats)

 e_ah_conv (meV/atom):    2.10911 +/- 1.1e+00    cor: 0.35
 e_ah_hma  (meV/atom):    0.42650 +/- 4.3e-02    cor: 0.11
 p_ah_conv      (GPa):    0.01371 +/- 3.1e-02    cor: 0.36
 p_ah_hma       (GPa):   -0.03419 +/- 4.1e-03    cor: 0.26
```

Listing 1: Using pyHMA in the Python interpreter, with application to aluminum fcc crystal at high pressure ($P_{\text{lat}} = 114.4$ GPa; corresponds to volume of 10 Å^3/atom) and temperature (1000 K). The VASP AIMD simulation consists of two consecutive runs, with the first one starts from the equilibrium fcc configuration. The total MD length used is 10, 000 steps, including 2000 steps for equilibration, with timestep of $\Delta t = 2.0$ fs. The quasiharmonic pressure input (pressure_qh) needed for HMA pressure is computed using Phonopy package (see Section 3.1).

AIMD simulations are carried out in the standard NVT canonical ensemble, with the temperature controlled using a Langevin thermostat [32] of friction coefficient of 10 ps^{-1}. The use of this specific thermostat (rather than Andersen or Nosé–Hoover) allows for using a relatively large MD integrator time step ($\Delta t = 2.0$ fs) without loss of accuracy (for the given DFT model), as we showed recently [33]. A single isochore of volume 10.0 Å^3/atom is considered, with temperature varying from 500 to 4000 K, in 500 K intervals. The simulation box is made of 32-atoms supercell, created from 2 × 2 × 2 fcc conventional unit cells. Clearly, finite-size effects can be large at such small system; however, our focus here is not absolute accuracy (i.e., "true" estimate) *per se*, but to demonstrate the usage of pyHMA for a given model. All simulations run for 10^4 steps (20 ps), with the data collection starting after 10^3 steps of equilibration. The stochastic uncertainty in ensemble averages is estimated using the block-averaging technique. To ensure small correlations (less than ≈ 0.2), we use 100 blocks, each having block size of 90 steps. Uncertainties are based on 68% confidence limits.

```
$ # Usage:
$ # pyhma --pressure_qh=qh pressure (GPa) --steps_eq=equilib. steps --blocksize=block size
$ #      [--steps_tot=used steps] [--force_tol=force tolerance] [--raw_files|-r]
$ #      [--fermi_dirac] [--meV] [--verbose|-v] vasprun-1.xml vasprun-2.xml ...
$ pyhma --pressure_qh=4.94525 --steps_eq=1000 --steps_tot=10000 --blocksize=90
        -r --meV vasprun-1.xml  vasprun-2.xml
e_ah_conv (meV/atom):    2.10911 +/- 1.1e+00    cor: 0.35
e_ah_hma  (meV/atom):    0.42650 +/- 4.3e-02    cor: 0.11
p_ah_conv      (GPa):    0.01371 +/- 3.1e-02    cor: 0.36
p_ah_hma       (GPa):   -0.03419 +/- 4.1e-03    cor: 0.26

```

Listing 2: Using pyHMA from the command-line through the pyhma script, applied to the same system as in Listing 1. The command-line options are given in square brackets, with the following default values: steps_tot=MD steps detected and force_tol=0.001 (eV/atom). In addition, the following flags direct pyHMA to do the following: raw_file generates raw data files (poscar_eq.dat, posfor.dat, energy.dat, and pressure.dat), fermi_dirac uses electronic free-energy surface F (default is reading ground-state PES; E0), uses meV units (default is eV), and verbose prints simulation details while running

The quasiharmonic pressure $P_{\rm qh}$ (Table 1) is computed from $P_{\rm qh} = -\partial A_{\rm qh}/\partial V$, where $A_{\rm qh}$ is the classical free energy (apart from the lattice contribution) using the quasiharmonic approximation [1,34],

$$A_{\rm qh}\,(T, V) = k_{\rm B}T \sum_{j=1}^{3(N-1)} \ln\left(\frac{\hbar\omega_j}{k_{\rm B}T}\right) - k_{\rm B}T\ln(N^{1/2}V), \tag{5}$$

where $\hbar \equiv h/2\pi$, with h being Planck's constant, N is the number of atoms, V is the box volume, and ω_j are the phonon frequencies, obtained as the eigenvalues of the dynamical Hessian matrix of the minimized potential energy. The summation is carried out over the $3(N-1)$ degrees of freedom and the last term accounts for the center of mass motion (vanishes in the thermodynamic limit). The phonon calculations are conducted using the finite-displacement (frozen-phonon) method, as implemented in Phonopy [35] (version 2.4.2), with VASP used to compute forces. We obtained the volume derivative of the first term in Eq. (5) by performing lattice dynamics calculations on 21 different volumes (from 9.0 to 11.0, with interval of 0.1 Å^3/atom), then applying a fourth-order polynomial fit, from which the derivative is obtained analytically. The low CPU cost of the quasiharmonic calculations allowed us to do such a (relatively) large number of points; hence, we get an accurate estimate at the desired density with low CPU effort, in comparison to AIMD simulations. The trivial derivative of the center-of-mass term is given analytically by $k_{\rm B}T/V$. At the volume of interest (10 Å^3/atom), the total derivative of $A_{\rm qh}$ yields a quasiharmonic pressure of the form: $P_{\rm qh}(\text{GPa}) = 0.00494525\ T(\text{K})$.

Once the anharmonic contribution is available from pyHMA, the absolute property can be obtained by adding to it the lattice and quasiharmonic contributions,

$$U = U^*_{\rm lat} + \frac{3}{2}(N-1)k_{\rm B}T + U_{\rm ah}, \text{ and} \tag{6}$$

$$P = P^*_{\rm lat} + P^*_{\rm qh} + P_{\rm ah}. \tag{7}$$

In principle, for a given level of DFT theory (exchange–correlation, pseudopotential, etc.), all components should be computed using the same level of "DFT quality" (e.g., k-points and energy cutoff). However, we observed here (and earlier [23]) that the convergence rate of properties with respect to DFT parameters depends on the contribution under investigation (i.e., lattice, quasiharmonic, or anharmonic). More specifically, we found (for monatomic atoms) that the anharmonic contribution requires the least rigorous DFT convergence parameters to reach the same accuracy as the other contributions (results not shown), which may be attributed to cancellation of errors. Therefore, for a given target accuracy (say, 1 meV/atom and 0.1 GPa), we have used different DFT settings for each component of the absolute energy and pressure. Both lattice and quasiharmonic contributions (not the focus of this work) can be computed with arbitrary accuracy, because only few single-point DFT energy calculations are required — this is denoted in Eqs. (6)–(7) by asterisk superscripts. In addition, only a single unit cell is needed for the lattice calculations due to the independence of this component on the system size.

It is worth emphasizing here that $U_{\rm lat}$, $P_{\rm lat}$, and $P_{\rm qh}$ inputs needed for the anharmonic calculations (see Table 1) must be obtained using the same setting (e.g., system size, DFT parameters, etc.) as used with AIMD simulations in order to ensure having the same PES. Conversely, since $U^*_{\rm lat}$, $P^*_{\rm lat}$, and $P^*_{\rm qh}$ are computed at different level of theory, they should be used only for computing full, absolute properties (i.e., by adding to the anharmonic-property averages output by pyHMA), and not as inputs for the pyHMA calculations, as this will result in inconsistent and inaccurate anharmonic results.

In the next two subsections we present anharmonic results for aluminum obtained using pyHMA, to show the effectiveness of the HMA method relative to standard (Conv) averaging approach. Since we are interested in presenting only anharmonic properties in this work, the other lattice and quasiharmonic contributions needed for absolute values are not given here.

3.2. Results: Anharmonic energy and pressure

Fig. 2 depicts the temperature dependence of the anharmonic energy (a) and pressure (b), up to 4000 K, using both Conv and HMA formulations. The first observation is that HMA method provides more precise estimates, compared to the Conv approach, without loss of accuracy. Some of the Conv data points are not statistically consistent with HMA, which can be attributed to the relatively large timestep used here ($\Delta t = 2.0$ fs). As we showed earlier, while HMA can handle such a timestep, Conv technique introduces uncontrolled inaccuracies [25]. Moreover, as expected for anharmonic behavior, the leading term for the temperature variation should be T^2 (i.e., linear term is absent). Accordingly, we fit the anharmonic energy and pressure (Fig. 2) using a polynomial function (blue line) in the form $c_1T^2 + c_2T^3 + c_3T^4$.

To quantify the extent of precision improvement of HMA compared to Conv, we use the recently introduced difficulty ratio metric, $D_{\rm Conv}/D_{\rm HMA}$; where the difficulty D is defined by $D \equiv \sigma t^{1/2}$, where σ is the stochastic uncertainty and t is the CPU time [36]. This quantity is invariant with the simulation length, so results from different runs can be compared. However, for our case, the CPU times are the same from Conv and HMA, so the difficulty

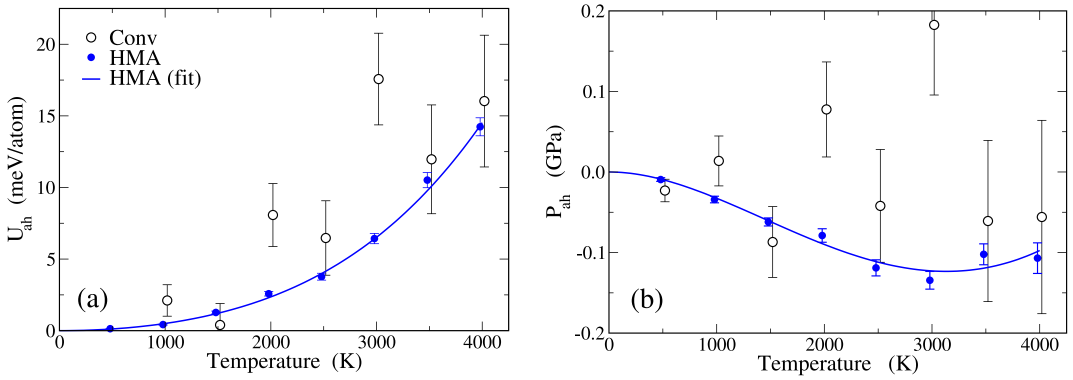


Fig. 2. Temperature dependence of anharmonic energy (a) and pressure (b), both using Conv and HMA approaches. The blue lines are polynomial fits in the form $c_1T^2 + c_2T^3 + c_3T^4$, to ensure a leading quadratic behavior. All error bars are based on 68% confidence limits, here and throughout this work. For clarity, data points are shifted slightly to the right (Conv) and to the left (HMA).

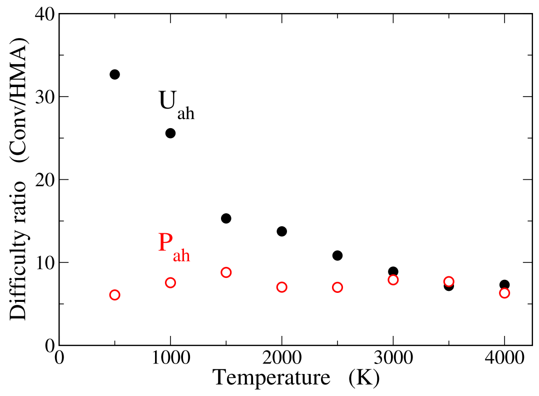


Fig. 3. Temperature dependence of the $D_{\text{Conv}}/D_{\text{HMA}}$ difficulty ratio of measuring anharmonic energy and pressure, where $D \equiv t^{1/2}\sigma$ (see text).

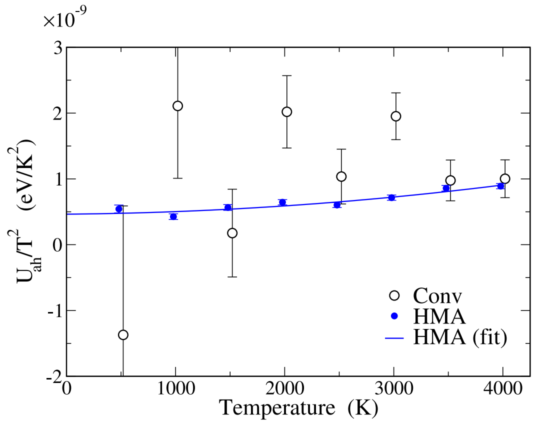


Fig. 4. Temperature dependence of the thermodynamic-integration integrand (Eq. (8)), both using Conv and HMA methods. The blue line is a second-order polynomial fit of the HMA data, which is given by (in eV/K^2): $4.6395 \times 10^{-10} + 1.4877 \times 10^{-14}T + 2.4021 \times 10^{-17}T^2$. For clarity, data points are shifted slightly to the right (Conv) and to the left (HMA).

ratio is simply the ratio of uncertainties. The square of this ratio is the Conv to HMA relative CPU time required to obtain a result of a given precision. Fig. 3 shows the temperature dependence of the difficulty ratio for measuring anharmonic energy and pressure. For anharmonic energy, the HMA improvement increases with decreasing temperature to about 33× at 500 K (i.e. ≈ 10^3 CPU speed-up). On the other hand, the HMA improvement of anharmonic pressure is nearly constant, ≈ 7× (i.e. ≈ 50× CPU speed-up).

3.3. Results: Anharmonic free energy

As an application to the precise HMA anharmonic energy obtained from pyHMA, we consider computing anharmonic free energy A_{ah} along an isochore (fixed volume) via thermodynamic integration in temperature, which is computationally demanding when using standard (Conv) averaging. The A_{ah} free-energy expression is given by [21],

$$A_{\text{ah}}(T) = -T \int_0^T \frac{U_{\text{ah}}(T')}{T'^2} dT'. \tag{8}$$

Because the leading term of U_{ah} is quadratic in T, the integrand $U_{\text{ah}}(T')/T'^2$ has a finite intercept at $T \to 0$. It is important to emphasize that the thermodynamic integration in temperature (HMA and Conv) is valid only for a temperature-independent PES (i.e., ground-state DFT; E0); hence, it cannot be used with finite-temperature DFT models (e.g., using Fermi–Dirac statistics; ISMEAR=-1). Fig. 4 shows the variation of the integrand with temperature, using both Conv and HMA methods, computed from the data given in Fig. 2(a). The variation from HMA is significantly smoother than Conv, which makes it the more efficient choice to compute anharmonic free energies. To do this, we fit the HMA integrand data using a second-order polynomial function (blue line), then analytically integrate the fitting function using Eq. (8) to get $A_{\text{ah}}(T)$. Fig. 5 depicts the temperature dependence of the anharmonic free energy using HMA energies, which also has a leading quadratic term in T, as can be inferred from Eq. (8). The dashed lines represent stochastic uncertainty bounds (obtained from propagation of errors in the fit), which is less than our 1 meV tolerance target.

Once the anharmonic contribution is available from pyHMA, the absolute Helmholtz free-energy property (again, not the focus of this work) can be obtained by adding to it the lattice and quasiharmonic contributions,

$$A = U^*_{\text{lat}} + A^*_{\text{qh}} + A_{\text{ah}}, \tag{9}$$

where A^*_{qh} is the quasiharmonic free energy as given by Eq. (5), with the asterisk indicating the possibility of using a different DFT parameters than those used with the AIMD simulation.

4. Conclusions

We have developed the pyHMA package, which allows VASP users to exploit the benefits offered by the HMA method for

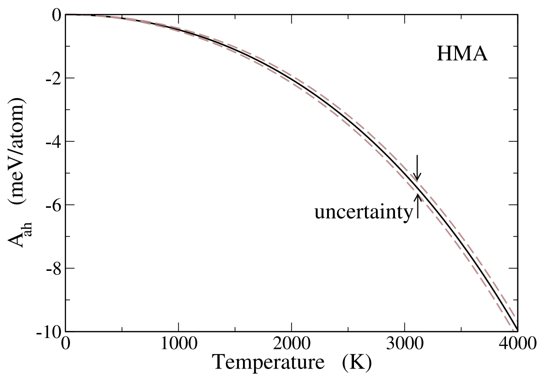


Fig. 5. Temperature dependence of the anharmonic free energy using HMA method. This is obtained from Eq. (8), with the integrand given by the fitting function in Fig. 4. The dashed lines represent stochastic uncertainty bounds as obtained from propagation of errors in the fit.

measuring anharmonic properties of harmonically stable crystals (i.e., no imaginary frequencies). Advantages include high precision (hence, speed-up) and accuracy, relative to the conventional approach, for a given choice of DFT settings (e.g., exchange–correlation functional).

In particular, the package calculates the anharmonic energy and pressure of crystalline systems by post-processing the `vasprun.xml` files obtained from canonical AIMD simulations. As a case study, the thermodynamic properties of aluminum at high pressure and up to near-melting temperature were computed; the computational savings from HMA were many orders of magnitude relative to conventional averaging. Also, the conventionally overwhelming computational cost of free-energy calculations was significantly reduced.

Extension of `pyHMA` to permit calculation of properties given as the second derivative of the free energy (e.g., heat capacity, elastic constants) would require the capability in `VASP` to compute and manipulate the Hessian matrix given the second derivatives of the energy with respect to atom coordinates. Such a capability has been implemented in LAMMPS [27].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the U.S. National Science Foundation, grant OAC-1739145.

References

[1] M.T. Dove, Introduction to Lattice Dynamics, Cambridge University Press, New York, 2005.
[2] N.R. Werthamer, Phys. Rev. B 1 (1970) 572–581, http://dx.doi.org/10.1103/PhysRevB.1.572.
[3] T. Tadano, S. Tsuneyuki, Phys. Rev. B 92 (2015) 054301, http://dx.doi.org/10.1103/PhysRevB.92.054301.
[4] T. Tadano, S. Tsuneyuki, Phys. Rev. Lett. 120 (2018) 105901, http://dx.doi.org/10.1103/PhysRevLett.120.105901.
[5] F. Zhou, W. Nielson, Y. Xia, V. Ozoliš, Phys. Rev. Lett. 113 (2014) 185501, http://dx.doi.org/10.1103/PhysRevLett.113.185501.
[6] J.M. Bowman, Acc. Chem. Res. 19 (7) (1986) 202–208, http://dx.doi.org/10.1021/ar00127a002.
[7] B. Monserrat, N.D. Drummond, R.J. Needs, Phys. Rev. B 87 (14) (2013) 144302.
[8] E.A. Engel, B. Monserrat, R.J. Needs, Phys. Rev. X 5 (2015) 021033, http://dx.doi.org/10.1103/PhysRevX.5.021033.
[9] J.C.A. Prentice, R.J. Needs, Phys. Rev. Mater. 1 (2017) 023801, http://dx.doi.org/10.1103/PhysRevMaterials.1.023801.
[10] A. Erba, J. Maul, M. Ferrabone, P. Carbonnière, M. Rérat, R. Dovesi, J. Chem. Theory Comput. 15 (6) (2019) 3755–3765, http://dx.doi.org/10.1021/acs.jctc.9b00293.
[11] A. Erba, J. Maul, M. Ferrabone, R. Dovesi, M. Rérat, P. Carbonnière, J. Chem. Theory Comput. 15 (6) (2019) 3766–3777, http://dx.doi.org/10.1021/acs.jctc.9b00294, PMID: 31038948.
[12] P. Carbonnière, A. Erba, F. Richter, R. Dovesi, M. Rerat, J. Chem. Theory Comput. 16 (5) (2020) 3343–3351, http://dx.doi.org/10.1021/acs.jctc.9b01061, PMID: 32275427.
[13] P. Souvatzis, O. Eriksson, M.I. Katsnelson, S.P. Rudin, Phys. Rev. Lett. 100 (2008) 095901, http://dx.doi.org/10.1103/PhysRevLett.100.095901.
[14] I. Errea, M. Calandra, F. Mauri, Phys. Rev. B 89 (2014) 064302, http://dx.doi.org/10.1103/PhysRevB.89.064302.
[15] O. Hellman, I.A. Abrikosov, Phys. Rev. B 88 (14) (2013) 144301.
[16] O. Hellman, P. Steneteg, I.A. Abrikosov, S.I. Simak, Phys. Rev. B 87 (10) (2013) 104111.
[17] O. Hellman, I.A. Abrikosov, S.I. Simak, Phys. Rev. B 84 (18) (2011) 180301.
[18] I. Errea, B. Rousseau, A. Bergara, Phys. Rev. Lett. 106 (2011) 165501, http://dx.doi.org/10.1103/PhysRevLett.106.165501.
[19] P. Korotaev, M. Belov, A. Yanilkin, Comput. Mater. Sci. 150 (2018) 47–53, http://dx.doi.org/10.1016/j.commatsci.2018.03.057.
[20] V. Kapil, E. Engel, M. Rossi, M. Ceriotti, J. Chem. Theory Comput. 15 (11) (2019) 5845–5857, http://dx.doi.org/10.1021/acs.jctc.9b00596, PMID: 31532997.
[21] S.G. Moustafa, A.J. Schultz, D.A. Kofke, Phys. Rev. E 92 (2015) 043303.
[22] A.J. Schultz, S.G. Moustafa, W. Lin, S.J. Weinstein, D.A. Kofke, J. Chem. Theory Comput. 12 (4) (2016) 1491–1498.
[23] S.G. Moustafa, A.J. Schultz, E. Zurek, D.A. Kofke, Phys. Rev. B 96 (2017) 014117, http://dx.doi.org/10.1103/PhysRevB.96.014117.
[24] S.G. Moustafa, A.J. Schultz, D.A. Kofke, J. Chem. Theory Comput. 13 (2) (2017) 825–834.
[25] S.G. Moustafa, A.J. Schultz, D.A. Kofke, J. Chem. Phys. 149 (12) (2018) 124109, http://dx.doi.org/10.1063/1.5043614.
[26] A.J. Schultz, D.A. Kofke, J. Chem. Phys. 149 (20) (2018) 204508, http://dx.doi.org/10.1063/1.5053714.
[27] A. Purohit, A.J. Schultz, D.A. Kofke, J. Chem. Phys. 152 (1) (2020) 014107, http://dx.doi.org/10.1063/1.5129942.
[28] D. Minakov, P. Levashov, V. Fokin, Comput. Mater. Sci. 127 (C) (2017) 42–47.
[29] G. Kresse, J. Furthmüller, Phys. Rev. B 54 (1996) 11169–11186, http://dx.doi.org/10.1103/PhysRevB.54.11169.
[30] P.E. Blöchl, Phys. Rev. B 50 (1994) 17953–17979, http://dx.doi.org/10.1103/PhysRevB.50.17953.
[31] J.P. Perdew, K. Burke, M. Ernzerhof, Phys. Rev. Lett. 77 (1996) 3865–3868, http://dx.doi.org/10.1103/PhysRevLett.77.3865.
[32] M.P. Allen, D.J. Tildesley, Computer Simulation of Liquids, in: Oxford Science Publ, Clarendon Press, 1989.
[33] A. Purohit, A.J. Schultz, S.G. Moustafa, J.R. Errington, D.A. Kofke, Mol. Phys. 116 (21–22) (2018) 3027–3041, http://dx.doi.org/10.1080/00268976.2018.1481542.
[34] T.B. Tan, A.J. Schultz, D.A. Kofke, J. Chem. Phys. 132 (21) (2010) 214103.
[35] A. Togo, I. Tanaka, Scr. Mater. 108 (2015) 1–5.
[36] A.J. Schultz, D.A. Kofke, J. Chem. Theory Comput. 10 (12) (2014) 5229–5234.