Implementation of harmonically mapped averaging in LAMMPS, and effect of potential truncation on anharmonic properties

Cite as: J. Chem. Phys. 152, 014107 (2020); https://doi.org/10.1063/1.5129942
Submitted: 02 October 2019. Accepted: 13 December 2019. Published Online: 06 January 2020

Apoorva Purohit, Andrew J. Schultz, and David A. Kofke

COLLECTIONS

This paper was selected as Featured

ARTICLES YOU MAY BE INTERESTED IN

On the temperature dependence of liquid structure
The Journal of Chemical Physics 152, 011102 (2020); https://doi.org/10.1063/1.5135932

Analytical classical density functionals from an equation learning network
The Journal of Chemical Physics 152, 021102 (2020); https://doi.org/10.1063/1.5135919

Ion pair free energy surface as a probe of ionic liquid structure
The Journal of Chemical Physics 152, 014103 (2020); https://doi.org/10.1063/1.5128693
Implementation of harmonically mapped averaging in LAMMPS, and effect of potential truncation on anharmonic properties

Apoorva Purohit, Andrew J. Schultz, and David A. Kofke

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

Electronic mail: kofke@buffalo.edu

ABSTRACT
Implementation of the harmonically mapped averaging (HMA) framework in the Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) is presented for on-the-fly computations of the energy, pressure, and heat capacity of crystalline systems during canonical molecular dynamics simulations. HMA has a low central processing unit and storage requirements and is straightforward to use. As a case study, the properties of the Lennard-Jones and embedded-atom model (parameterized for nickel) crystals are computed. The results demonstrate the higher efficiency of the new class compared to the inbuilt LAMMPS classes for calculating these properties. However, HMA loses its effectiveness in systems where diffusion occurs in the crystal, and an example is presented to allow this behavior to be recognized. In addition to its improved precision, HMA is less affected by small errors introduced by having a larger time step in molecular dynamics simulations. We also present an analysis of the effect of potential truncation on anharmonic properties, and show that artifacts of truncation on the HMA averages can be eliminated simply by shifting the potential energy to zero at the truncation radius. Full properties can be obtained by adding easily computed values for the lattice and harmonic properties using the untruncated potential.

I. INTRODUCTION

Mapped averaging is a general framework that enables an approximate theory or reference model for the behavior of a molecular system to be applied to derive new ensemble averages for the system’s thermodynamic properties. Expressions derived this way have the form of a (often analytically expressed) term that yields the property according to the approximate theory, plus an additional term that forms an exact correction to the theory. This additional term is given as an ensemble average, and often it may be evaluated much more easily than the full ensemble average of the property. In this manner, mapped averaging can yield a property value of a given precision with much less computational effort, with no loss of accuracy.

Harmonically mapped averaging (HMA) is the application of mapped averaging to crystalline systems, with a harmonic model forming the approximate reference. Mapped averaging has had its greatest effectiveness in this application, largely because harmonic behavior is a very good starting point for describing the crystals. HMA provides a direct measurement of the anharmonic contributions to the properties, and because the harmonic behavior is treated exactly and separately, the anharmonic contributions can be measured with great precision with little effort. In addition to offering improved precision, it has been shown that properties given via HMA: (a) are less susceptible to finite-size effects; (b) are not as sensitive to truncation of the potential; (c) provide more statistically independent configurations for a given number of samples; (d) are faster to equilibrate from a nonequilibrium initial condition; and (e) are less sensitive to the time step in molecular dynamics (MD) simulations.

Mapped averaging has no effect on how sampling is performed, and mapped averages for multiple properties may be collected at.
The primary applications of HMA so far are for the determination of the basic thermal and volumetric properties (energy, pressure, heat capacity, etc.) of monatomic crystals and for the calculation of the free energy by thermodynamic integration of data obtained this way. HMA provides a significant benefit here, because with conventional averaging, the uncertainty in the anharmonic contribution at low temperature is the largest source of error in the free energy, even at melting. Efforts are in progress toward application to molecular crystals, as well as other properties such as elastic constants and density distributions. Mapped averaging is basically a method for calculation of free-energy derivatives, so any property that can be expressed as such is amenable to formulation as a mapped average. Transport properties in particular appear not to fall in this category.

The primary obstacles to the broader use of mapped averaging, and HMA especially, are (1) awareness in the community of the benefits it provides; and (2) the availability of codes that implement it. Of course, progress in remedying the latter can help to alleviate the former, and to this end, we are endeavoring to implement HMA in several popular molecular simulation codes. Presently, we report our progress in completing this development in LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator).

LAMMPS is a classical molecular dynamics (MD) code, widely used for studying crystalline (and noncrystalline) systems. It has interatomic potentials for metals, semiconductors, biomolecules, polymers, and coarse-grained or mesoscopic systems, and it can be easily used to model particles at the atomic, meso, or continuum scales. It can run on single processors or in parallel and is distributed as open-source code under the terms of the GPL. Its sustainability is further ensured by the large community of users and developers, which continues to grow steadily.

We have implemented basic HMA averages in the LAMMPS C++ code. A new class, ComputeHMA, has been created for this purpose, which performs on-the-fly computations of the thermodynamic energy, pressure, and heat capacity, and specifically their anharmonic contributions. This development will allow the practitioners of MD to conveniently exploit the tremendous benefits of HMA and thereby advance the entire field. Moreover, it opens up a path for extending HMA capabilities to the calculation of other thermodynamic properties.

In the following sections, we review the key formulas involved in HMA, and then we discuss how we have implemented them in LAMMPS. We provide demonstrations of their use, show how they perform in relation to potential truncation, and describe how they behave when failing due to the presence of diffusion in the simulated system.

II. HARMONICALLY MAPPED AVERAGING FORMULAS

The key formulas for the calculation of the energy, pressure, and heat capacity are presented in Table I. These expressions capture the primary elements involved in the calculation of harmonically mapped averages, and their implementation can serve as a prototype for other properties that might be introduced to LAMMPS in the future. The formula for the anharmonic contribution to the average configurational energy, for example, is

\[ U^\text{sh} = \left( U + \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r} \right) - U^{\text{lat}}. \]  

The full configurational energy is obtained by adding (or just not subtracting as indicated) the lattice contribution \( U^{\text{lat}} \) (which is a constant obtained separately via a lattice sum for the perfect crystal), and the quasiharmonic contribution, which is (for a 3-dimensional system with fixed center of mass)

\[ U^{\text{qh}} = \frac{1}{2} (N - 1) k_B T, \]

where \( N \) is the number of atoms, \( k_B \) is the Boltzmann’s constant, and \( T \) is the temperature. The angle brackets in Eq. (1) represent an ensemble average, and within them the term \( \mathbf{F} \cdot \Delta \mathbf{r} \) is the dot product of two vectors of length \( 3N \) (one entry for each atomic coordinate): \( \mathbf{F} \) represents the force on each atom, and \( \Delta \mathbf{r} \) is the displacement of each atom from its respective lattice site. For a perfectly harmonic system, the configurational energy \( U \) equals \( -(1/2) \mathbf{F} \cdot \Delta \mathbf{r} \), and the quantity being averaged is identically zero (apart from the lattice energy); hence, to the extent it is nonzero, it represents the anharmonic contribution to the energy.

Similar to the configurational energy, the HMA ensemble average for pressure separates the quasiharmonic pressure \( (P^{\text{qh}}) \) from the residual average containing the conventional virial pressure \( (P^{\text{vir}}) \). The HMA expression is given in terms of \( P^{\text{vir}} \), which is already implemented in LAMMPS. This is fortunate because there are

<table>
<thead>
<tr>
<th>Property</th>
<th>Conventional average</th>
<th>Harmonically mapped average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy, ( U )</td>
<td>( \langle U \rangle - \frac{1}{2} (N - 1) k_B T - U^{\text{lat}} )</td>
<td>( \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r} ) - ( U^{\text{lat}} )</td>
</tr>
<tr>
<td>Pressure, ( P )</td>
<td>( \rho k_B T + (P^{\text{vir}}) - P^{\text{ph}} - P^{\text{lat}} )</td>
<td>( P^{\text{vir}} + \frac{P^{\text{ph}} - k_B T}{2(N - 1) k_B T} \mathbf{F} \cdot \Delta \mathbf{r} ) - ( P^{\text{lat}} )</td>
</tr>
<tr>
<td>Heat capacity, ( C_V / k_B )</td>
<td>( \text{Var} \left[ \frac{U}{k_B T} \right] ) - ( \frac{1}{2} (N - 1) )</td>
<td>( \text{Var} \left[ \frac{U + \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r}}{k_B T} \right] ) - ( \frac{1}{4k_B T} \mathbf{F} \cdot \Delta \mathbf{r} \cdot \mathbf{\Phi} \cdot \Delta \mathbf{r} )</td>
</tr>
</tbody>
</table>
subtleties regarding this term and periodic boundary conditions when applied to multibody potentials. These issues are already handled in LAMMPS, so the HMA expression can be applied to multibody forms without additional difficulty. The HMA expression also requires that a value be specified for \( P^{\Phi} \). The LAMMPS implementation takes a user-specified value (see below) for computing the ensemble average shown in the table, and adds the same user-specified \( P^{\Phi} \) to the result, when obtaining the total pressure \( P \). The value specified for \( P^{\Phi} \) does not have to be exactly the true quasiharmonic pressure, and in principle the choice of this quantity does not affect the value computed for the total pressure. However, the uncertainty of the result will be minimized to the extent that the user-specified \( P^{\Phi} \) accurately represents the quasiharmonic contribution to the pressure. An appropriate estimate can be obtained from packages for lattice dynamics (e.g., phonopy\textsuperscript{17}).

Notably, first-derivative properties (meaning properties that are given as the first derivative of the free energy; here, the energy and the pressure) express the HMA as an average that involves the force on each atom. LAMMPS is primarily an MD code, and forces are needed to propagate the trajectory, so this information is available to compute the HMA average for any molecular model already implemented in LAMMPS. In contrast, for second-derivative properties (e.g., the isochoric heat capacity \( C_V \)) HMA averages involve the Hessian of the intermolecular potential, i.e., the matrix of the second derivatives of the energy with respect to the individual atom coordinates (labeled \( \Phi \) in the \( C_V \) formula in Table I). This capability was not previously available in LAMMPS (except perhaps by numerical differentiation, which is prohibitively expensive for repeated use in a molecular simulation); therefore, we had to introduce new methods in LAMMPS designed to provide this information when needed (without, of course, impacting the performance for the vast majority of applications where the Hessian is not needed). We have implemented the code for the Hessian only for the Lennard-Jones (LJ) model, so if it is desired to compute a second-derivative property via HMA, it may be necessary to perform additional coding to implement the Hessian for the potential of interest (if it has not been done previously). We outline how this is done in the discussion that follows further below. First, we describe how properties are computed in LAMMPS, assuming that all the force and (if needed) Hessian capabilities are in place for the model being simulated.

### III. IMPLEMENTATION OF HMA IN LAMMPS

#### A. New class: ComputeHMA

We implement HMA by defining a new class, ComputeHMA, which is a subclass of the LAMMPS Compute class. In LAMMPS, Compute provides one of the primary means for adding new functionality. It is invoked after each time step, or if desired, after multiple steps as specified by the LAMMPS thermo command.

ComputeHMA has been designed to efficiently calculate the energy and pressure and their anharmonic contributions for the case of monatomic crystals using interatomic potentials available in LAMMPS that can describe crystalline solids. The reformulated HMA ensemble averages have been implemented in this class. The code has been parallelized so that one or many processors may be used. Quantities calculated by the compute are instantaneous values, meaning they are calculated from the positions and forces of atoms on the current timestep.

The class is easy to set up and use because it follows the native format of LAMMPS input files. It can yield pressure, energy, and/or heat capacity (as specified by the user) in a single simulation, has low central processing unit (CPU) and storage requirements, and is straightforward to use. A limitation of this class is that it will provide superior results than LAMMPS inbuilt classes only for crystalline systems without defects or where the defects are not moving significantly. Also, the class will work with all thermostats except the LAMMPS Langevin thermostat, as the inbuilt class for implementing the Langevin thermostat modifies the forces on particles, which interferes with HMA calculation.

In order to use this class, the user must first specify the compute in the input file, formatted as follows:

\[
\text{computeID all hma\_fixID\{anharmonic\} \{u\}\{p\}\{harmonic\}\{cv\}}
\]

The first term, compute, indicates that the line specifies a LAMMPS compute. The second argument represents the ID of the compute as created in the LAMMPS input script. The next argument specifies that the HMA properties for all particles need to be computed. Then, the hma style name specifies that HMA is to be performed. fixID should specify the ID of the canonical simulation thermostat—computeHMA extracts the set temperature for the simulation using this argument and uses it for calculating the reformulated ensemble averages. Any or all of the remaining arguments may be included (as indicated by the braces, which are not part of the command). anharmonic specifies that the user requires the evaluation of the anharmonic contributions to the properties; this feature is discussed further in Sec. V. u indicates that the intermolecular average energy is to be computed. The calculation of the pressure is indicated by \( p \), and requires that an estimate of the harmonic pressure (in excess of the lattice pressure, using the unit system specified elsewhere in the input file) be supplied as shown in the command; as stated above, the precision of the result is maximized if the input is close to the harmonic pressure as given by lattice dynamics calculations. Finally, cv specifies that the heat capacity is computed.

When ComputeHMA is defined in the LAMMPS input script as shown above, it stores the positions of all the particles in the form of an array using the “fix store” style, these lattice positions are later used to calculate \( \Delta r \), the displacement from lattice sites as required in the mapped averages. Thus, the compute must be defined just after the minimization of the energy, when forces on all the particles are zero.

The ComputeHMA should be called using the LAMMPS command thermo, exactly like invoking the other child classes of Compute. Once invoked, the class can compute thermodynamic properties on-the-fly in a single simulation, so there is no need to store large trajectory files. HMA does not modify sampling, so its use for one property does not interfere with HMA for any other (so conventional averages of the same properties can be recorded at the same time). The positions and forces required for HMA computation are already stored by the LAMMPS inbuilt “Atom” class, so for first-derivative properties, there is no extra computation cost or storage space required (except for one array that stores the lattice coordinates).

An example input script is included in Appendix C.
B. Second-derivative properties and the Hessian

The $3N \times 3N$ Hessian matrix $\Phi$ has components $\phi_{\alpha\beta}$,

$$\phi_{\alpha\beta} = \frac{\partial^2 U}{\partial r_{\alpha} \partial r_{\beta}},$$

(3)

where $i$ and $j$ are atom indices, and $\alpha, \beta = [x, y, z]$ indicate the coordinate directions. For a spherically symmetric, pairwise-additive potential, such that

$$U = \sum_{i=1}^{N} \sum_{j\neq i} u(r_{ij}),$$

(4)

where $r_{ij} = |r_i - r_j|$, then

$$\phi_{\alpha\beta} = \begin{cases} r_{ij}^{-3} \left( r_{ia} - r_{ja} \right) \left( r_{ia} - r_{ja} \right) u'(r_{ij}) - \delta_{\alpha\beta} u'(r_{ij}), & i \neq j \\ - \delta_{\alpha\beta} u'(r_{ij}), & i = j \end{cases}$$

(5)

and the primes indicate derivatives of the pair potential with respect to the radial distance between the interacting atoms. The Hessian for other potential forms may be evaluated on a case-by-case basis.

The calculation of the Hessian using such formulas was not a capability previously included in LAMMPS, so we implemented it as part of this project. We have added a method (single_hessian) to the Pair class, which is similar to the existing single method, but also computes the Hessian matrix for a given pair of atoms. To simplify the implementation of this new method for strictly two-body pair styles [as opposed to many-body styles like embedded-atom model (EAM)], we have also added a helper method (hessian_twobody) in Pair that will construct the Hessian matrix from radial properties and the displacement vector between the atom pair. The steps needed for a developer to enable the calculation of the Hessian for a new potential are outlined in Appendix A.

The HMA expression for a second-derivative property includes a variance term, which is evaluated by combining two ensemble averages. For example, the expression for the HMA heat capacity includes

$$\text{Var} \left[ U + \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r} \right] = \left( \left( U + \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r} \right)^2 \right) - \left( U + \frac{1}{2} \mathbf{F} \cdot \Delta \mathbf{r} \right)^2.$$

(6)

The quantity returned by ComputeHMA includes only the first term on the right-hand side of Eq. (6). The second term is the square of the HMA anharmonic energy, so this average too should be computed if calculation of the heat capacity is being performed. Then, after the simulation is completed, this average can be squared and subtracted from the (incomplete) value returned for the HMA heat capacity to yield the actual result for the HMA $C_V$. This procedure, however, can introduce significant round-off error. To address this, the anharmonic keyword can be passed and/or the output format can be specified with more digits:

```
thermo_modify format float '%22.15e'
```

The anharmonic keyword will instruct the compute to return anharmonic properties rather than the full properties (lattice, harmonic, and anharmonic). When using this keyword, the compute must be first active [it must be included via a thermo_style

custom command, while the atoms are still at their lattice sites (before equilibration)].

IV. EXAMPLE APPLICATIONS

We consider two example applications to demonstrate the use and effectiveness of HMA as implemented in LAMMPS.

A. Lennard-Jones model

MD simulations of Lennard-Jones (LJ) fcc crystals were conducted in the HMA-enabled LAMMPS code, to compare the efficiency of the new class vs the inbuilt conventional-averaging classes. All quantities in this section are given in “LJ units,” such that $\sigma_{\text{LJ}} = 1$, $\epsilon_{\text{LJ}}/k_B = 1$, and $m = 1$, where $\sigma_{\text{LJ}}$ and $\epsilon_{\text{LJ}}$ are the LJ size and energy parameters, respectively, and $m$ is the atom mass. The simulations were all performed using $N = 4000$ atoms at a density $\rho = 1.0$ and temperatures $T$ from 0.1 up to 1.0, which is slightly above the melting temperature (0.9561%). A MD time step of 0.005 was employed, and 200000 MD steps were performed, in addition to 20 000 steps of equilibration. The LJ pair potential was truncated with force shifting at $r_c = 6$.

Results are presented in Figs. 1 and 2, which address, respectively, the relative precision and accuracy of the conventional vs HMA methods. Figure 1 shows the ratio of the difficulty for the various properties as computed via conventional averaging relative to HMA, with both averages computed in the same simulation run. The difficulty $D$ is defined

$$D = t^{1/2} \sigma,$$

(7)

where $t$ is the amount of CPU time required to obtain an average with uncertainty $\sigma$. This quantity is asymptotically invariant with the amount of sampling, and the ratio provides a measure of the relative efficiency of the two averaging methods. In particular, the square of the difficulty gives the ratio of the amount of CPU required to obtain a result with a given precision. The figure clearly illustrates the efficiency of HMA in obtaining these averages. Depending on temperature, the first-derivative properties are computed 10 (at melting) to 10 000 ($u^m$ at the lowest temperature) times faster by HMA than...
by conventional averaging. As shown in the earlier work, \(^3\) second-derivative properties benefit from HMA even more than the first-derivative properties: in this application, the heat capacity is given 100–10^6 times faster when using HMA.

Figure 2 demonstrates the effect of the MD timestep on the accuracy of conventional and HMA methods. The quantity plotted is \(u_{an} / T^2\), which is of significant interest, as it is the integrand needed to evaluate the free energy as a function of temperature using thermodynamic integration. \(^4\) The averages are computed for three choices of the MD time step \(\Delta t = 0.002, 0.004, \text{and} 0.008\) (again, in LJ units). The HMA results are nearly mutually consistent for all three values of \(\Delta t\) (there are systematic differences, and although they are not due to uncertainties, they are comparable in magnitude to the uncertainties obtained for the amount of sampling used), whereas the conventional averages are statistically in agreement with the other averages only for the smallest \(\Delta t\) (showing that the two averaging methods provide mutually consistent results for sufficiently small \(\Delta t\)). Clearly, the results obtained using HMA are less affected by errors introduced by having a larger time step in MD simulations, whereas conventional averaging is prone to yielding inaccurate results, if the time step is not sufficiently small. This outcome is consistent with previous observations.

Incidentally, Fig. 2 again illustrates the improvement in precision of HMA over conventional averaging. The error bars on the HMA data are much smaller than those from conventional averaging, even though both sets of results are based on the same sample of configurations.

B. Embedded-atom model (EAM)

Next, we demonstrate the performance of HMA in evaluating first-derivative properties for nonpairwise potentials. Specifically, we look at an embedded-atom model (EAM) for nickel crystal containing an extended defect (stacking fault). We have not implemented the Hessian for this model, so we do not examine second-derivative properties such as the heat capacity. The results presented in this section required no additional modification to LAMMPS, beyond that described already for the LJ model. In metals with a face-centered cubic structure, the extended defect observed most often is the intrinsic stacking fault, obtained by removing a single close-packed fcc (111) layer from the perfect fcc matrix. Figure 3 demonstrates the application of HMA for calculating the thermodynamic properties of a nickel crystal containing a stacking fault defect. The EAM potential of Mishin was used\(^{15}\) to model the interatomic interactions. The simulations were conducted at a timestep of 0.5 fs with \(18 \times 10^6\) steps of data collection after \(5 \times 10^6\) steps of equilibration.

In Fig. 3(a), the symbols correspond to the values obtained using the conventional method, and the lines represent the HMA results. There is a small but significant disagreement between the results, and according to Fig. 2 this is most likely due to inaccuracy in the conventional averages—the conventional averaging method would need an even smaller timestep to provide accurate results of \(u_{an} / T^2\) for all temperatures. Figure 3(b) shows the uncertainty ratio of the results obtained using conventional averaging relative to the HMA approach. HMA provides up to 75 times higher precision than conventional averaging (seen at the lowest temperature examined), which corresponds to a 5600-fold reduction in the computational effort; at the highest temperature, the improvement in uncertainty is about a factor of 5, or a factor of 25 in computational effort.

V. EFFECT OF POTENTIAL TRUNCATION

For computational efficiency, many models in LAMMPS use truncation of the potential to reduce the number of neighbor
interactions that must be tracked as the simulation proceeds. It is, therefore, of interest to examine the effect of truncation on the HMA results, as compared to the effect on conventional averages. We assume that the behavior of interest is that of the untruncated potential, and that determining the properties of the truncated potential per se is not the goal. Accordingly, to compute the properties for the full, untruncated potential, we recommend collecting data using the anharmonic keyword, which returns for a property $M$,

$$
\langle M^{ah} \rangle \equiv \langle M - M^{ah} \rangle - M^{lat}.
$$

The HMA formulas yield directly the ensemble average given on the right-hand side of Eq. (8), and the LAMMPS implementation subtracts $M^{lat}$, which is computed by evaluating the property for the atoms at their lattice sites, and thus, it is for the truncated potential. The infinite-system, untruncated value of property $M$ can then be obtained via

$$
M = \langle M^{ah} \rangle + M^{ah}_{\infty} + M^{lat}_{\infty},
$$

where the $\infty$ subscripts indicate that the values are computed (outside of LAMMPS) for the infinite lattice without truncating the potential. We now consider how potential truncation affects $\langle M^{ah} \rangle$ because the approach prescribed by Eq. (9) implies that we are using a value of the anharmonic-contribution average that is also representative of the infinite, untruncated system.

To this end, we performed simulations in LAMMPS of the LJ crystal discussed in Sec. IV A, and we present them here using LJ units for all quantities. The simulations used a MD time step of 0.005, and for each conditions studied, we sampled 20,000 steps of equilibration followed by 200,000 steps of production. We gathered data for four values of the truncation radius $r_c$ (3.0, 4.0, 5.0, and 6.0, respectively), and we examined three approaches in performing truncation: simple truncation; truncation with shifting the potential to zero at $r_c$; and truncation with shift of both the energy and force to zero at $r_c$. For each case, results were taken using both conventional averaging and HMA.

The effect of truncation on $\langle u^{ah} \rangle$ (where $u$ is the energy per atom, $U/N$) is summarized in Fig. 4. We plot the behavior in terms of $\langle u^{ah} \rangle/T^2$ (which is finite for $T \to 0$), because (a) this highlights the behavior at low temperature, where some of the key differences are seen (a log scale for the absissa is also used for this purpose); and (b) this quantity is the one needed when performing thermodynamic integration (in $T$) to compute the free energy as a function of temperature.\(^{3,6,7}\)

Figure 4 shows that the conventional and HMA results each converge to a limiting form with increasing $r_c$, but these limits are different for the two methods. The HMA results exhibit a plateau at low temperature at an ordinate value of about $-0.38$—on a linear $T$ scale this would be seen as a simple intercept at this point. In contrast, at low temperature the conventional results appear to be diverging (which is more apparent on a linear $T$ scale). This discrepancy is explained by the much greater sensitivity of the conventional average to the MD time step at low temperature: data in Fig. 4 are obtained using a time step of 0.005, which according to Fig. 2 introduces a noticeable inaccuracy to the conventional averages. Consequently, where the methods are converged with respect to $r_c$, we should consider the HMA results to be more accurate, relative to conventional averages taken under the same conditions.

Turning now to the behavior with respect to the truncation method, we observe that simple truncation has an effect on $\langle u^{ah} \rangle$ that can easily be as large as the value itself, so it cannot be ignored. For both averaging methods, the inaccuracy in $\langle u^{ah} \rangle$ due to truncation becomes comparable to the error bars only for the largest truncation radius ($r_c = 6$). The conventional and HMA results for the same truncation differ from each other, and while we could enumerate several reasons for this, ultimately it does not matter: both provide unreliable estimates of the anharmonic contribution for the infinite untruncated system. The remedy is to work with the shifted or force-shifted forms of truncation, which are both possible in LAMMPS. The figure shows that in these cases, both the conventional and HMA formulations give results that are much less sensitive to $r_c$. Force-shifting is slightly better, as it shows convergence to the limiting form for all temperature even at $r_c = 3$, whereas the simple shift is converged at most temperatures for $r_c = 4$. The consistency of all HMA results with respect to $r_c$ demonstrates that there is no need for any direct correction to $\langle u^{ah} \rangle$ to account for the truncation. The only correction required to offset the shift or force-shift when evaluating the full properties is to ensure that the untruncated, unshifted potential is employed for $u^{lat}$, when adding it in as prescribed by Eq. (9).

Finally, we point out that in all cases the precision of the HMA data—where the uncertainty in many cases is smaller than a line width—greatly exceeds that from conventional averaging, which has noticeable scatter in the data, consistent with their error bars.

The effect of truncation on the anharmonic contribution to the pressure is described by Fig. 5. For each truncation method,
the quasiharmonic pressure $P_{\text{qh}}$ was computed by differentiating a fit of the harmonic free energy as a function of volume (obtained by straightforward lattice-dynamics calculations \cite{footnote2} for the truncated potential). The conventional-method value of $P_{\text{qh}}$ was then obtained by subtracting $P_{\text{vh}}$ and $\frac{\partial U}{\partial V}$ from the pressure computed as usual from the virial, whereas the HMA value was given by the formula in Table I. Simple truncation and truncation+shift give identical results for the pressure, so only the former is shown in the figure.

Figure 5 is similar to the presentation for $\langle \nu_{\text{ah}} \rangle$ given in Fig. 4. We divide by $T^2$ to generate a quantity that has a finite intercept at $T \to 0$ (although $P_{\text{qh}}/T^2$ does not have the same significance as an integrand for free-energy calculations), and we observe significant effect of truncation. The HMA results again show a limiting form with increasing $r_c$. For $r_c = 3$ simple truncation and (to a lesser extent) truncation+shift show significant deviation from the limiting behavior. All other HMA data (including $r_c = 3$ force-shifted) are consistent with the limiting behavior (i.e., are converged with respect to $r_c$).

VI. EFFECT OF DIFFUSION

The performance of HMA is degraded in the presence of diffusion, whereby an atom occupying one particular lattice site spontaneously jumps to another. The problem is that the variance reduction in HMA is predicated on the idea that an atom moves approximately harmonically about its average location, which is identified as the lattice site it occupied at the beginning of the simulation. In principle, it is possible to accommodate diffusion if the wandering atom is reassigned its lattice site when it is observed to move. It would not be unreasonable to track the atoms and take this movement into account. It is also possible to keep track of atoms and their neighbors, but this method is more computationally intensive. In any case, the variance reduction in HMA is predicated on the idea that an atom moves approximately harmonically about its average location, which is identified as the lattice site it occupied at the beginning of the simulation. In principle, it is possible to accommodate diffusion if the wandering atom is reassigned its lattice site when it is observed to move. It would not be unreasonable to track the atoms and take such action when diffusion is observed, but at present the HMA implementation does not attempt this.

We performed simulations to demonstrate this effect, to better enable the user to recognize the problem if it arises. The simulated systems were prepared with a single vacancy, to enable the site-hopping effect that we aim to demonstrate, and two temperatures 0.6, 0.8, and the much larger scale of the figure makes it difficult to discern a limiting form of $c_v/T$ with respect to increasing $r_c$. For HMA, only $r_c = 3$ simple truncation and (to a lesser extent) truncation+shift show significant deviation from the limiting behavior. All other HMA data (including $r_c = 3$ force-shifted) are consistent with the limiting behavior (i.e., are converged with respect to $r_c$).
The additions to LAMMPS described here now enable the calculation of the anharmonic contributions to the energy and pressure for any model already implemented in LAMMPS, without any additional modifications. The evaluation of second-derivative properties for potentials other than LJ will require programming of the Hessian matrix for the model, whereas evaluation of properties other than the energy, pressure, or heat capacity will require programming of the Hessian for a model makes it immediately be applied to calculate HMA first-derivative properties; (4) any new second-derivative property coded in ComputeHMA can be immediately applied to any potential in LAMMPS, without any additional modifications.

VII. CONCLUSION

The additions to LAMMPS described here now enable the calculation of the anharmonic contributions to the energy and pressure for any model already implemented in LAMMPS, without any additional modifications. The evaluation of second-derivative properties for potentials other than LJ will require programming of the Hessian matrix for the model, whereas evaluation of properties other than the energy, pressure, or heat capacity will require programming of the Hessian for a model makes it immediately be applied to calculate HMA first-derivative properties; (4) any new second-derivative property coded in ComputeHMA can be immediately applied to any potential in LAMMPS, without any additional modifications.

SUPPLEMENTARY MATERIAL

A Python script and sample data are provided in the supplementary material. The script reads a user-prepared extract of the LAMMPS output file and computes ensemble averages (and 68%-confidence uncertainties) based on conventional and HMA formulas. The sample data are used to test that the script functions as intended; details are given in the script header. This script is provided to help others use the HMA methods described in this manuscript. Development of this and other HMA scripts is ongoing, and the latest implementations may be accessed at https://github.com/etomica(mapped-averaging).

ACKNOWLEDGMENTS

This work was supported by the U.S. National Science Foundation (Grant No. OAC-1739145). We are grateful to the LAMMPS development team for their cooperation and guidance as we completed the implementation described in this paper.

APPENDIX A: STEPS TO IMPLEMENT HESSIAN CAPABILITY FOR A POTENTIAL IN LAMMPS

To evaluate second-derivative properties using HMA, it is necessary to compute the Hessian of the potential for an arbitrary configuration. This is accomplished using analytic formulas for the second derivatives of the potential with respect to the atomic coordinates. Hence, to perform such calculations for a given potential, these formulas must have been coded into LAMMPS. At the time of this writing, we have completed this only for the PairLJSmoothLinear potential, which codes for a truncated and force-shifted Lennard-Jones model. Developers and users who wish to apply HMA to compute second-derivative properties for other models can implement the Hessian calculation via the following steps.
- Implement the method `single_hessian` in the potential class. This method performs the same operations as `single` (i.e., computes force and energy of a single pairwise interaction between two atoms), but in addition it computes the 6 second derivatives defined by Eq. (3). Upon return, `d2u` contains the six elements giving the second derivative with respect to the difference in the `ij` positions, for coordinate pairs `xx, xy, xz, yy, yz, and zz`, respectively. An example is shown here:

```c++
// The arguments are the same as for single, but with the addition of d2u.
double PairLJSmoothLinear::single_hessian(int /i*/ , int /j*/ , int itype,
int jtype, double rsq, double delr[3], double /*factor_coul*/ , double
factor_lj, double &fforce, double d2u[6])
{
// The code from here to the next comment duplicates the code in single
double r2inv,r6inv,forcelj, philj,r,rinv;
r2inv = 1.0/rsq;
r6inv = r2inv*r2inv*r2inv;
rinv = sqrt(r2inv);
r = sqrt(rsq);
forcelj = r6inv*(lj1[itype][jtype]*r6inv-lj2[itype][jtype]);
forcelj = rinv*forcelj - dljcut[itype][jtype];
fforce = factor_lj*forcelj*rinv;
philj = r6inv*(lj3[itype][jtype]*r6inv-lj4[itype][jtype]);
philj = philj - ljcut[itype][jtype]
+ (r-cut[itype][jtype])*dljcut[itype][jtype];
// The next line is the only line that needs changing for a new pairwise
// spherical potential. d2r is u''(r).
double d2r = factor_lj * r6inv * (13.0*lj1[itype][jtype]*r6inv -
7.0*lj2[itype][jtype])/rsq;
// This is a helper method that fills in d2r for a spherically symmetric
// potential. It may be used without change once d2r is coded.
// fforce (computed above) is -u'(r)/r.
hessian_twobody(fforce, -(fforce + d2r) / rsq, delr, d2u);
// Return statement is the same as used in single.
return factor_lj*philj;
}
```

- Add the following flag method in the potential class to indicate that the Hessian is coded in LAMMPS for this model. This method did not previously exist (which LAMMPS interprets as the Hessian code not being implemented).

```c++
PairLJSmoothLinear::PairLJSmoothLinear(LAMMPS *lmp) : Pair(lmp) {
single_hessian_enable = 1;
}
```

The `hessian_twobody` method is a helper method that computes the unique Hessian elements, given the radial derivatives for a spherically symmetric potential model. It does not need to be reimplemented with each new potential, but we list it here for completeness.
void Pair::hessian_twobody(double fforce, double dfac, double delr[3], double phiTensor[6]) {
    int m = 0;
    for (int k=0; k<3; k++) {
        phiTensor[m] = fforce;
        for (int l=k; l<3; l++) {
            if (l>k) phiTensor[m] = 0;
            phiTensor[m++] += delr[k] * delr[l] * dfac;
        }
    }
}

APPENDIX B: ADDING NEW HMA PROPERTIES

At present, we have implemented HMA capabilities in LAMMPS to compute the average potential energy, the pressure, and the heat capacity. Harmonically mapped averaging is a general method for computing free-energy derivatives, and it is possible to extend LAMMPS to evaluate other thermodynamic properties via HMA. In some cases, formulas for the properties have been derived, and only their implementation is needed; in other cases, it will be necessary to derive the properties first, using the framework presented and demonstrated elsewhere.

New properties can be implemented by editing the ComputeHMA class. The full code listing for the class is too lengthy to present here, but it can be viewed on the LAMMPS github at the following URL: http://github.com/lammps/lammps/blob/master/src/USER-MISC/compute_hma.cpp. The primary method that would require editing is compute_vector, which is where the HMA property formulas are implemented. For second-derivative properties, the Hessian matrix is not computed and stored in its entirety; rather, it is evaluated for each pair and immediately used in the property formulas. Modifications are needed also to the constructor and destructor, similar to the code that is in place for other properties. Note that the field comm_forward should be set to 3 in the constructor if computing a second-derivative property.

APPENDIX C: EXAMPLE INPUT SCRIPT

We provide here an example of a LAMMPS input script that invokes HMA for calculation of the energy, pressure, and heat capacity.

```plaintext
# Initialize geometry and box
Clear
units lj
dimension 3
boundary p p p

# Create atoms; positions of atoms here define lattice-site locations ----
atom_style atomic
atom_modify map array
lattice fcc 1.0
regionbox block 0 10 0 10 0 10 units lattice
create_box1 box
latticefcc 1.0 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 region box

# Define interatomic potential and initialize velocities
pair_style lj/smooth/linear 3
pair_coeff * * 1.0 1.0
mass 1.0
atom_modify sort 0 1
velocity all create 0.1 45678 dist Gaussian

# Declare conventional calculation of energy and pressure
compute u all pe
compute p all pressure NULL pair
```
# Declare HMA calculation of energy, pressure, and heat capacity
compute hma all HMA settemp anharmonic u p 9.551442600614891 cv

# Define integrator.
# User-defined name settemp here must match term in HMA calculation line
timestep 0.005
fix settemp all nvt temp 1.00 1.00 0.5

# Specify and format outputs; [1], [2], [3] specify energy, pressure, cv
thermo_modify format float

# Specs for run length
thermo 500
run 20000
thermo 20
run 200000

REFERENCES