The Journal of Chemical Physics

A combined on-the-fly/interpolation procedure for evaluating energy values needed in molecular simulations

Cite as: J. Chem. Phys. **151**, 174116 (2019); https://doi.org/10.1063/1.5124469
Submitted: 14 August 2019 . Accepted: 14 October 2019 . Published Online: 07 November 2019

Konstantin Karandashev, and Jiří Vaníček

The Journal of Chemical Physics ARTICLE scitation.org/journal/jcp

A combined on-the-fly/interpolation procedure for evaluating energy values needed in molecular simulations

Cite as: J. Chem. Phys. 151, 174116 (2019); doi: 10.1063/1.5124469
Submitted: 14 August 2019 • Accepted: 14 October 2019 •
Published Online: 7 November 2019

Konstantin Karandashev[a] and Jiří Vaníček[b]

AFFILIATIONS

Laboratory of Theoretical Physical Chemistry, Institut des Sciences et Ingénierie Chimiques, Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland

[a]Electronic mail: konstantin.karandashev@alumni.epfl.ch
[b]Electronic mail: jiri.vanicek@epfl.ch

ABSTRACT

We propose an algorithm for molecular dynamics or Monte Carlo simulations that uses an interpolation procedure to estimate potential energy values from energies and gradients evaluated previously at points of a simplicial mesh. We chose an interpolation procedure that is exact for harmonic systems and considered two possible mesh types: Delaunay triangulation and an alternative anisotropic triangulation designed to improve performance in anharmonic systems. The mesh is generated and updated on the fly during the simulation. The procedure is tested on two-dimensional quartic oscillators and on the path integral Monte Carlo evaluation of the HCN/DCN equilibrium isotope effect.

I. INTRODUCTION

Accurate evaluation of the Born-Oppenheimer potential energy surface of a molecular system is essential for predicting its dynamical and equilibrium properties. Numerous advances in the algorithms used for the problem[1,2] combined with increasing computational power available to researchers have made it possible to combine on-the-fly *ab initio* evaluation of the potential energy even with path integral[3–5] or semiclassical[6–11] dynamics algorithms. Unfortunately, such approaches are still computationally expensive, and for long simulations requiring a very large number of potential energy values in the same region of configuration space, it is reasonable to instead generate a mesh of points at which accurate *ab initio* calculations are made and then fit a function to reproduce their potential energy values or some other quantities that become bottlenecks of the calculation, such as Hessians of the potential energy.[12,13] For that purpose, a multitude of methods have been proposed, from the modified Shepard interpolation[14–17] to more sophisticated approaches,[18] including those based on interpolating moving least squares,[19,20] Gaussian process regression,[21,22] and neural networks.[23–26]

We aimed for a procedure that would interpolate energies from stored data evaluated at points of a simplicial mesh and that would be comparable to the Shepard interpolation in terms of simplicity and generality. To that end, we investigated interpolation from points of the mesh that constitute a simplex containing the point of interest, an approach already applied to some lower-dimensional systems.[27,28] Compared to the Shepard interpolation, the downside of this approach is the necessity to generate a triangulation for the mesh, whose size grows very fast with the number of dimensions,[29] but the upside is the logarithmic scaling of the interpolation procedure with the number of mesh points as well as an extra order of accuracy for a given number of derivatives available at the mesh points. In comparison to the method of Ref. 28, the main differences in the approach presented here are an alternative triangulation of the mesh and a different choice of the interpolant, along with a procedure for updating the mesh during the simulation.

The theory behind the proposed algorithm is explained in Sec. II, while Sec. III presents numerical tests for model anharmonic potentials and for the HCN/DCN equilibrium isotope effect. While we focus on classical Monte Carlo and path integral Monte Carlo applications, similar interpolation procedures can also be used with molecular dynamics or path integral molecular dynamics methods.

II. THEORY

Running a Monte Carlo simulation requires knowledge of the potential energy function $V(\mathbf{r})$, where $\mathbf{r}$ is the D-dimensional vector of system's internal coordinates. Let us assume that we can access a number of previously stored points together with their potential energy and gradient values as well as a triangulation of their mesh; we want to use that information to estimate the potential energy value at a new point $\tilde{\mathbf{r}}$. If the mesh currently contains fewer than $D+1$ points, $V(\tilde{\mathbf{r}})$ is evaluated exactly and $\tilde{\mathbf{r}}$ is added to the mesh, which will be triangulated (in the only possible way) once $D+1$ points have been added. If the mesh has already been triangulated, the following algorithm is used for estimating $V(\tilde{\mathbf{r}})$:

1. Find the simplex $\tilde{\mathcal{S}}$ containing $\tilde{\mathbf{r}}$ or verify that $\tilde{\mathbf{r}}$ lies outside the convex hull $\mathcal{C}_{\text{mesh}}$ of all mesh points.
2. If $\tilde{\mathcal{S}}$ was found, calculate the value of the interpolant $\tilde{V}(\tilde{\mathbf{r}})$ and estimate whether the interpolation error $|\tilde{V}(\tilde{\mathbf{r}}) - V(\tilde{\mathbf{r}})|$ is below a predefined threshold.
3. If $\tilde{\mathbf{r}} \notin \mathcal{C}_{\text{mesh}}$ or if $V(\tilde{\mathbf{r}})$ cannot be estimated with sufficient accuracy, add more points to the mesh to allow for an accurate estimate of $V(\tilde{\mathbf{r}})$.

We will discuss each part of the algorithm separately in Subsections II A–II C.

A. Interpolation procedure and reliability estimate

Suppose $\tilde{\mathbf{r}}$ is inside simplex $\tilde{\mathcal{S}}$ with vertices $\mathbf{r}^j_{\tilde{\mathcal{S}}}$ $(j = 1, \ldots, D+1)$ and we want to estimate $V(\tilde{\mathbf{r}})$ based on the values of the energy and its gradient at the $D+1$ points $\mathbf{r}^j_{\tilde{\mathcal{S}}}$. Previously, Clough-Tocher interpolants[30,31] were used for the problem in up to three dimensions;[27,28] these interpolation schemes are exact for cubic potentials and have derivatives that are continuous up to the second order, but they have two disadvantages: they use Hessians, whose evaluation increases enormously the cost of an *ab initio* calculation, and their generalization to higher-dimensional systems is not straightforward. Perpendicular interpolation[32] is another powerful approach which, for an arbitrary number of dimensions and an arbitrary number of derivatives q available for all vertices, produces an interpolant that is exact for a polynomial of order $q+1$ and that has q continuous derivatives; however, it scales exponentially with dimensionality D, making potential applications to higher-dimensional systems problematic. In this work, we used an interpolant that exhibits a better scaling with D at the cost of having discontinuous derivatives. (If this is a problem, interpolants of Ref. 32 should be used instead.) To define this interpolant, we introduce barycentric coordinates λ_j $(j = 1, \ldots, D+1)$ of $\tilde{\mathbf{r}}$, which are defined by the $D+1$ equations

$$\sum_{j=1}^{D+1} \lambda_j \mathbf{r}^j_{\tilde{\mathcal{S}}} = \tilde{\mathbf{r}}, \qquad \sum_{j=1}^{D+1} \lambda_j = 1. \tag{1}$$

The interpolant we propose is defined in terms of "partial" interpolants $\tilde{V}^j$,

$$\tilde{V}^j(\tilde{\mathbf{r}}) = V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) + \frac{1}{2}\left[\nabla V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) + \sum_{j'=1}^{D+1} \lambda_{j'} \nabla V(\mathbf{r}^{j'}_{\tilde{\mathcal{S}}})\right] \cdot (\tilde{\mathbf{r}} - \mathbf{r}^j_{\tilde{\mathcal{S}}}), \tag{2}$$

all of which are exact for quadratic potentials. One way to combine them into a single interpolant symmetric with respect to vertex permutations is

$$\sum_{j=1}^{D+1} \lambda_j \tilde{V}^j(\tilde{\mathbf{r}}) = \sum_{j=1}^{D+1} \lambda_j \left[V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) + \frac{1}{2}\nabla V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) \cdot (\tilde{\mathbf{r}} - \mathbf{r}^j_{\tilde{\mathcal{S}}})\right] + \frac{1}{2}\left[\sum_{j'=1}^{D+1} \lambda_{j'} \nabla V(\mathbf{r}^{j'}_{\tilde{\mathcal{S}}})\right] \cdot \left[\sum_{j=1}^{D+1} \lambda_j (\tilde{\mathbf{r}} - \mathbf{r}^j_{\tilde{\mathcal{S}}})\right] \tag{3}$$

$$= \sum_{j=1}^{D+1} \lambda_j \left[V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) + \frac{1}{2}\nabla V(\mathbf{r}^j_{\tilde{\mathcal{S}}}) \cdot (\tilde{\mathbf{r}} - \mathbf{r}^j_{\tilde{\mathcal{S}}})\right], \tag{4}$$

which is an interpolant proposed in Ref. 33 (based on Refs. 34 and 35). The term on the second line is zero because the second factor is zero. This combination of $\tilde{V}^j$, however, would not reproduce the potential energy gradient at the vertices, which is a waste since each $\tilde{V}^j$ reproduces the gradient at vertex j. An alternative expression that does reproduce gradients at all vertices is

$$\tilde{V}(\tilde{\mathbf{r}}) = \frac{\sum_{j=1}^{D+1} \lambda_j^2 \tilde{V}^j(\tilde{\mathbf{r}})}{\sum_{j=1}^{D+1} \lambda_j^2}. \tag{5}$$

It is impossible to get a reliable estimate of the interpolation error without any knowledge of the third derivatives of $V(\mathbf{r})$ in the simplex, and application of Bayesian approaches as in the Shepard interpolation[36] is complicated by $\tilde{V}^j(\tilde{\mathbf{r}})$ containing data from all vertices of the simplex at once. One exception is the one-dimensional case, where defining

$$\delta V(\tilde{\mathbf{r}}) = \max_{j=1,\ldots,D+1} |\tilde{V}(\tilde{\mathbf{r}}) - \tilde{V}^j(\tilde{\mathbf{r}})| \tag{6}$$

(with $D = 1$) yields an exact estimate $|\tilde{V}(\tilde{\mathbf{r}}) - V(\tilde{\mathbf{r}})| \leq \delta V(\tilde{\mathbf{r}})$. The estimate seems to perform qualitatively correctly for a large number of higher-dimensional potentials as well, so we decided to deem the interpolation result reliable if $\delta V(\tilde{\mathbf{r}})$ were below some predetermined threshold $\delta V_{\max}$. This makes $\delta V_{\max}$ a parameter whose only relation to the interpolation error is that both are proportional to the magnitude of the third derivatives in a small enough simplex. The lack of a more precise relation between the two quantities forced us to estimate the *exact* interpolation error for a small number of randomly chosen interpolation results, determining whether the chosen $\delta V_{\max}$ is adequate. For small enough simplices, the leading contributions to both the interpolation error and $\delta V(\tilde{\mathbf{r}})$ depend linearly on the tensor of third derivatives of potential energy, so one possible rule of thumb for fixing an unacceptable interpolation error would be a proportional decrease in $\delta V_{\max}$, e.g., halving

$\delta V_{\max}$ if root mean square error (RMSE) of interpolation should be halved.

B. Updating the mesh and its triangulation

If we either find $\tilde{\mathbf{r}}$ to be outside the convex hull $\mathcal{C}_{\text{mesh}}$ or that $\delta V \geq \delta V_{\max}$, we add a carefully chosen point $\mathbf{r}_{\text{add}}$ to the mesh and update the triangulation. Before describing the algorithm, let us introduce several definitions. First, the boundary of $\mathcal{C}_{\text{mesh}}$ is a set of faces referred to as $\boldsymbol{\mathcal{F}}_{\text{mesh}}$. Second, we will often use the signed distance $n_{\mathcal{F}}(\mathbf{r})$ from the plane containing the face $\mathcal{F} \in \boldsymbol{\mathcal{F}}_{\text{mesh}}$, with the sign defined to be non-negative at the mesh points. Finally, both systems that will be considered in Sec. III have certain restrictions on the values $\mathbf{r}$ can take: in the symmetric two-dimensional quartic oscillator, the symmetry makes it possible to consider values of coordinates in only one quadrant (e.g., both x and y non-negative), while in HCN, our choice of internal coordinates implies that two of them only take non-negative values. We thus consider a situation where $\mathbf{r}$ needs to satisfy one or more linear constraints of the form

$$\mathbf{v}_i \cdot \mathbf{r} - c_i \geq 0, \tag{7}$$

where i is an index of the constraint, c_i is a scalar constant, and $\mathbf{v}_i$ is a vector constant.

When $\tilde{\mathbf{r}}$ was inside $\mathcal{C}_{\text{mesh}}$, we found that adding a point $\mathbf{r}_{\text{add}} = \tilde{\mathbf{r}}$ to the mesh worked well enough. However, when $\tilde{\mathbf{r}}$ was outside $\mathcal{C}_{\text{mesh}}$, we "pushed" $\mathbf{r}_{\text{add}}$ further out, i.e., chose $\mathbf{r}_{\text{add}}$ further away from $\mathcal{C}_{\text{mesh}}$ than $\tilde{\mathbf{r}}$, primarily to avoid creating nearly singular simplices. The procedure, referred to as "outward push," works as follows:

1. Find $\mathcal{F}_{\min} \in \boldsymbol{\mathcal{F}}_{\text{mesh}}$ that minimizes $n_{\mathcal{F}}(\tilde{\mathbf{r}})$.
2. Set $\mathbf{r}_{\text{add}} = \tilde{\mathbf{r}} - c_{\text{push}} \nabla n_{\mathcal{F}_{\min}}$, where c_{push} is some constant.
3. If, for some i, $\mathbf{v}_i \cdot \mathbf{r}_{\text{add}} - c_i < 0$, then replace $\mathbf{r}_{\text{add}}$ with

$$\mathbf{r}_{\text{add}} - \frac{\mathbf{v}_i}{|\mathbf{v}_i|^2}(\mathbf{v}_i \cdot \mathbf{r}_{\text{add}} - c_i). \tag{8}$$

Step 3 is designed to move the mesh point (pushed away in steps 1–2) onto one of the "constraining surfaces" defined by

$$\mathbf{v}_i \cdot \mathbf{r} - c_i = 0, \tag{9}$$

instead of rejecting a move that would violate the constraint. Several such rejections would lead to a mesh that would approach infinitely close to one of the constraining surfaces during the simulation, as discussed in Subsection III A.

Once $\mathbf{r}_{\text{add}}$ is chosen, we need to update the triangulation of the mesh. Here, we only present the main ideas of the employed algorithms; the details are in the Appendix. The starting point of this work was using a Delaunay triangulation[37] following its previous successful applications.[27,28] There exist several algorithms[29] that update a Delaunay triangulation at a cost that does not increase with the number of simplices. The approach outlined in this subsection uses Lawson flips[37] defined using "parabolic lifting"[38] instead of the more conventional empty circumsphere test.[37,38] One considers sets of $D + 2$ points, which can be triangulated at most in two different ways.[37] Whenever such a set is already triangulated using simplex array $\boldsymbol{\mathcal{S}}$ and an alternative triangulation $\boldsymbol{\mathcal{S}}'$ is available, one compares the values $G(\boldsymbol{\mathcal{S}})$ and $G(\boldsymbol{\mathcal{S}}')$, where the function G is defined as

$$G(\boldsymbol{\mathcal{S}}) = \sum_{\mathcal{S} \in \boldsymbol{\mathcal{S}}} g(\mathcal{S}) v(\mathcal{S}) \tag{10}$$

and where $v(\mathcal{S})$ is the volume of simplex $\mathcal{S}$. The choice of the cost function $g(\mathcal{S})$ that yields the Delaunay triangulation is

$$g_{\text{Delaunay}}(\mathcal{S}) = \sum_{j=1}^{D+1} |\mathbf{r}_{\mathcal{S}}^{j}|^2. \tag{11}$$

If $G_{\text{Delaunay}}(\boldsymbol{\mathcal{S}}) > G_{\text{Delaunay}}(\boldsymbol{\mathcal{S}}')$, which is equivalent to $\boldsymbol{\mathcal{S}}$ failing the empty circumsphere test used to define the Delaunay triangulation,[37,38] the simplices of $\boldsymbol{\mathcal{S}}$ are replaced with those of $\boldsymbol{\mathcal{S}}'$.

One performs Lawson flips until they fail to change the triangulation regardless of the initial $\boldsymbol{\mathcal{S}}$. Since each Lawson flip decreases $G_{\text{Delaunay}}(\boldsymbol{\mathcal{S}}_{\text{mesh}})$, where $\boldsymbol{\mathcal{S}}_{\text{mesh}}$ is the array of all simplices in $\mathcal{C}_{\text{mesh}}$, the algorithm is bound to stop at a certain point, and it can be proven[37] that the resulting final triangulation is unique to the mesh. It can also be shown that $G(\boldsymbol{\mathcal{S}}) \neq G(\boldsymbol{\mathcal{S}}')$ for the two triangulations of $D + 2$ points unless the points lie on a sphere or in a hyperplane; treatment of these singular cases is discussed in the Appendix.

The expression for $g_{\text{Delaunay}}(\mathcal{S})$ underlines one problem with the Delaunay triangulation: it treats all dimensions equivalently, necessitating a choice of internal coordinates that makes the properties of $V(\mathbf{r})$ approximately isotropic, which tends to be nontrivial. A Bayesian approach to bypassing the problem for the Shepard interpolation is discussed in Ref. 36, while for simplex interpolation, one can use higher-order derivatives to define a Riemannian metric[39] that can then be used to construct the triangulation optimal for the current interpolation procedure.[40–42] Unfortunately, for quadratic interpolation, the latter option would involve calculating the third derivatives of the potential, which is rather expensive; therefore, we instead used Lawson flips with a modified $g(\mathcal{S})$. Obviously, the procedure still stops at a certain triangulation regardless of the choice of $g(\mathcal{S})$, even though we will not be able to guarantee the triangulation's uniqueness without restrictions on the potential $V(\mathbf{r})$. The anisotropic $g(\mathcal{S})$ proposed in this work is

$$g_{\text{anisotr}}(\mathcal{S}) = \max_{j',j''=1,\ldots,D+1} \left| V(\mathbf{r}_{\mathcal{S}}^{j'}) - V(\mathbf{r}_{\mathcal{S}}^{j''}) - \frac{[\nabla V(\mathbf{r}_{\mathcal{S}}^{j'}) + \nabla V(\mathbf{r}_{\mathcal{S}}^{j''})] \cdot (\mathbf{r}_{\mathcal{S}}^{j'} - \mathbf{r}_{\mathcal{S}}^{j''})}{2} \right|, \tag{12}$$

which is a qualitative estimate of the upper bound for interpolation error in a given simplex; unlike g_{Delaunay}, g_{anisotr} is invariant with respect to linear transformations of coordinates. To avoid entering infinite loops for cases when $G(\boldsymbol{\mathcal{S}}) = G(\boldsymbol{\mathcal{S}}')$, we modify the flipping criterion to be $G(\boldsymbol{\mathcal{S}}) - G(\boldsymbol{\mathcal{S}}') > \delta G_{\min}$, where $\delta G_{\min}$ is a small predefined parameter.

g_{anisotr} should be applicable for any simplex interpolant that uses potential and its gradient and is exact for quadratic potentials. To understand why quadratic interpolation is special in this context, consider interpolating from $D + 1$ vertices with q derivatives available, which, in general, can yield an interpolant exact for polynomials up to degree $q + 1$. In the special case of $D = 1$, q polynomials of degree $q + 1$ can be constructed from $2(q + 1)$ parameters available; for $q = 1$, g_{anisotr} arises naturally as the degree to which the two cubic

polynomials disagree. The $q = 1$ case is special because for larger values of q and more than two polynomials of degree $(q + 1)$, several analogs of g_{anisotr} are possible, while for $q = 0$, the (linear) interpolant is uniquely defined, making it impossible to define a similar g_{anisotr}.

From now on, the triangulation that results from using g_{anisotr} with Lawson flips will be referred to as "anisotropic triangulation."

C. Search for the simplex

The last task is finding the simplex $\tilde{\mathcal{S}}$ that contains $\tilde{\mathbf{r}}$. We used *stochastic walk*[43] that iteratively updates $\tilde{\mathcal{S}}$ from an initial guess $\tilde{\mathcal{S}}_{\text{init}}$ by calculating barycentric coordinates λ_j [Eq. (1)], then terminating the search if all λ_j are positive or if a negative λ_j corresponds to a face that is also a face of $\mathcal{C}_{\text{mesh}}$, and otherwise obtaining the next $\mathcal{S}$ as the simplex across a randomly chosen face corresponding to a negative λ_j. The procedure is guaranteed to find $\tilde{\mathcal{S}}$ regardless of the triangulation used.[43] In molecular dynamics simulations or other calculations where configuration space coordinates are changed incrementally, the simplex that contained the previous simulation point would be a suitable candidate for $\tilde{\mathcal{S}}_{\text{init}}$. Unfortunately, this is not the case for good Monte Carlo simulations, which change the coordinates significantly in a single step of the random walk. As a result, we used k-trees[44] for generating an approximation $\mathbf{r}_{\text{close}}$ for the mesh point closest to $\tilde{\mathbf{r}}$; once $\mathbf{r}_{\text{close}}$ is found, we randomly choose a simplex $\tilde{\mathcal{S}}_{\text{init}}$ that has $\mathbf{r}_{\text{close}}$ as its vertex. The computational cost of finding $\mathbf{r}_{\text{close}}$ scales logarithmically with the number of points in the mesh, and it is reasonable to assume that the computational cost of subsequent choosing of $\tilde{\mathcal{S}}_{\text{init}}$ and locating $\tilde{\mathcal{S}}$ is approximately constant for large enough numbers of mesh points. Since the computational cost of simplex interpolation based on a known simplex does not depend on the number of mesh points, the total cost of our interpolation procedure scales logarithmically with the number of mesh points, as was mentioned in the Introduction.

III. NUMERICAL TESTS

A. Anharmonic oscillator

A large number of molecular systems are close to harmonic in the most relevant part of their configuration space, so as a model problem, we chose a system of two harmonic vibrational modes with a "very anisotropic" anharmonic perturbation,

$$V(x, y) = x^2 + y^2 + \epsilon_{\text{anharm}}\left[x^4 + (4y)^4\right], \tag{13}$$

where ϵ_{anharm} determines the "anharmonicity" of the potential. In all examples presented here, we ran 2^{24} ($\approx 1.68 \times 10^7$) step Monte Carlo simulations with inverse thermodynamic temperature $\beta = 1$ and $\delta V_{\max} = 3.125 \times 10^{-2}$; the mesh was constructed in $|x|$ and $|y|$ rather than x and y to capitalize on the potential's symmetry. The results are presented in Figs. 1–3, with the RMSEs of interpolation and the number of points added to the mesh during simulations with $\epsilon_{\text{anharm}} = 0$ and $\epsilon_{\text{anharm}} = 0.01$ displayed in Table I.

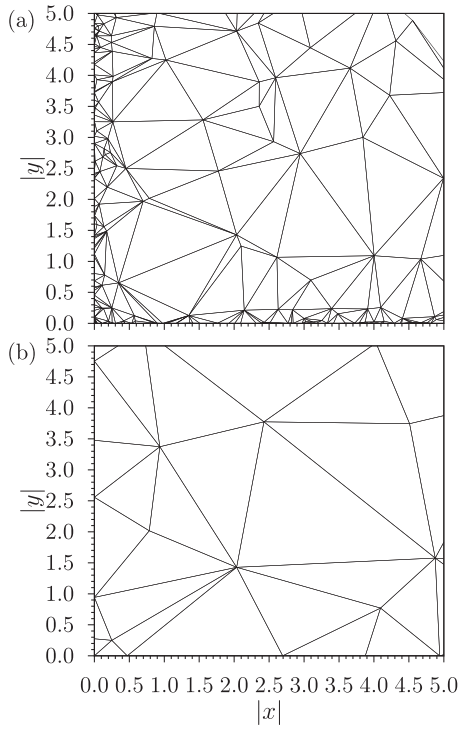

FIG. 1. Mesh triangulations obtained in classical Monte Carlo simulations of a 2D harmonic oscillator without (a) and with (b) the "outward push" procedure.

Figure 1 illustrates the reasoning behind introducing the "outward push" procedure (see Subsection II B) by comparing Delaunay triangulations generated at $\epsilon_{\text{anharm}} = 0$ without [panel (a)] and with [panel (b)] the outward push. In this harmonic case, the interpolation procedure is exact with $\delta V(\tilde{\mathbf{r}}) = 0$. Comparing the two triangulations illustrates how introducing the "outward push" decreases the number of points used and prevents the algorithm from placing many mesh points close to $|x| = 0$ and $|y| = 0$ lines as $\mathcal{C}_{\text{mesh}}$ incrementally approaches them. As seen from Table I, in this situation, the "outward push" procedure leads almost to an order-of-magnitude decrease in the number of mesh points, even though, as illustrated by results for $\epsilon_{\text{anharm}} = 0.01$ in Table I, the improvement is definitely less drastic for anharmonic potentials where simplex size is also determined by the magnitude of $\delta V(\tilde{\mathbf{r}})$.

The motivation behind the anisotropic triangulation introduced in this work is illustrated in Fig. 2, comparing the meshes and interpolation error distributions obtained using two different triangulations in Monte Carlo simulations for $\epsilon_{\text{anharm}} = 0.01$. The simplices obtained with Delaunay and anisotropic triangulations are plotted in panels (a) and (b); switching to the anisotropic triangulation "elongates" triangles along the $|x|$ axis, as it should, judging by the form of the anharmonic part of the potential (13). While in this case, the distribution of interpolation errors is not significantly affected, the number of mesh points added is decreased more than by a factor of two (see Table I).

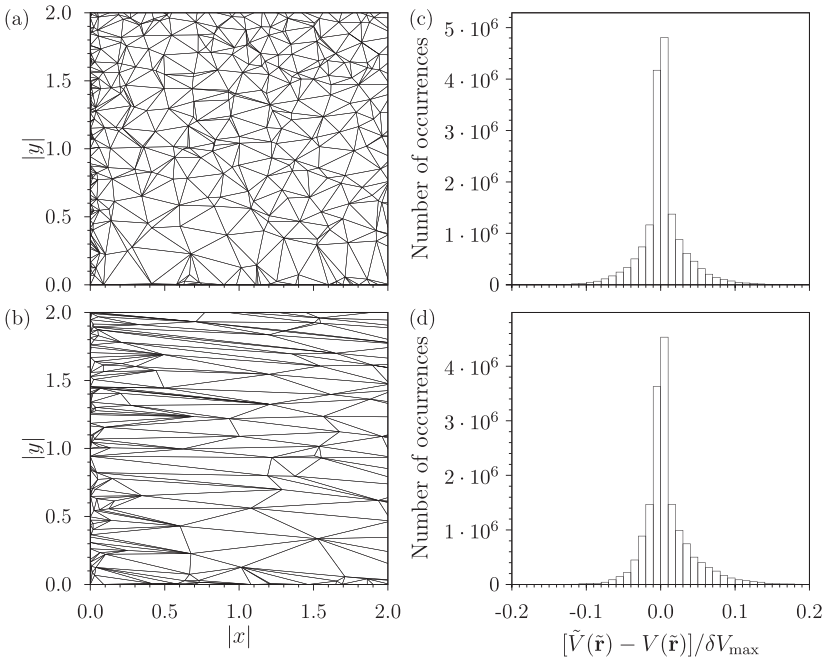

FIG. 2. Mesh triangulations and distributions of the interpolation error obtained in classical Monte Carlo simulations of a 2D quartic oscillator (13) with ϵ_{anharm} = 0.01 using the Delaunay [(a) and (c)] or anisotropic [(b) and (d)] triangulations.

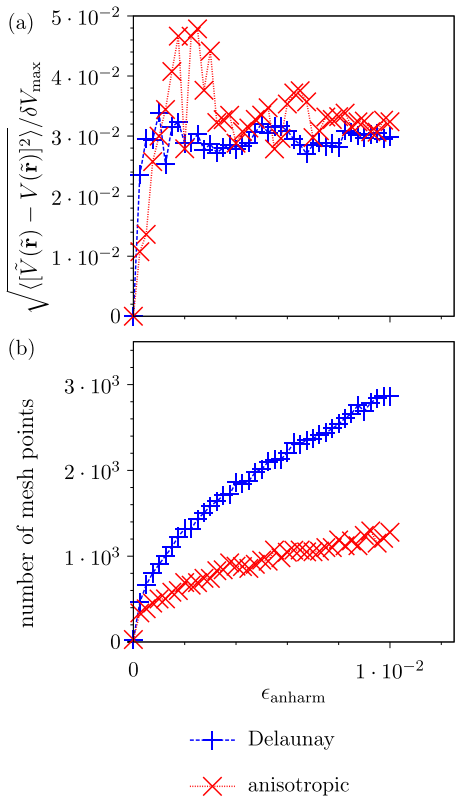

FIG. 3. Comparison of Delaunay and anisotropic triangulations applied in classical Monte Carlo simulations of two-dimensional quartic oscillators (13) with different ϵ_{anharm}. (a) Root mean square errors (RMSEs) of interpolation $\langle[\tilde{V}(\tilde{\mathbf{r}}) - V(\tilde{\mathbf{r}})]^2\rangle^{1/2}$ and (b) number of mesh points required.

We also checked how the tendencies observed for $\epsilon_{\text{anharm}} = 0.01$ hold for other values of ϵ_{anharm}; Fig. 3 demonstrates the resulting RMSEs of interpolation $\langle[\tilde{V}(\tilde{\mathbf{r}}) - V(\tilde{\mathbf{r}})]^2\rangle^{1/2}$ [panel (a)] and the number of mesh points [panel (b)]. Although changing the triangulation to anisotropic in this system increases slightly the interpolation errors, the decrease in the number of mesh points is much more significant. Also note that the RMSE is always much smaller than $\delta V_{\max}$, a tendency we observed for a wide range of potentials.

Finally, recall that running the Monte Carlo simulations presented here involved $2^{24} + 1 \approx 1.6 \times 10^7$ potential energy evaluations, and instead of exact calculations in each instance, we used mere thousands of mesh points to reproduce these exact calculations with great precision (see Table I). This demonstrates the potential of our method for speeding up practical calculations, a point elaborated further in Subsection III B.

B. HCN/DCN equilibrium isotope effect

In this subsection, we combine our interpolation procedure with the path integral Monte Carlo method,[46,47] which accounts for nuclear quantum effects by replacing each atom of the simulated molecule with P replicas connected by harmonic forces.[48] We combined the path integral Monte Carlo method with the free energy perturbation approach[49] (direct estimators[50]) for isotope fractionation to calculate the HCN/DCN equilibrium isotope effect defined as

$$\text{IE} = \frac{Q_{\text{DCN}}}{Q_{\text{HCN}}}, \tag{14}$$

where Q denotes the partition function. The potential energy surface of HCN was taken from Ref. 51. The interpolation algorithm used three internal coordinates that were defined in terms of atom radius-vectors $\mathbf{r}_{\text{D/H}}$, $\mathbf{r}_{\text{C}}$, and $\mathbf{r}_{\text{N}}$ as follows:

TABLE I. Root mean square errors (RMSEs) $\langle[\tilde{V}(\tilde{\mathbf{r}})-V(\tilde{\mathbf{r}})]^2\rangle^{1/2}$ of interpolation and the number of mesh points generated in Monte Carlo simulations of the quartic oscillator (13) at $\beta = 1$ with different interpolation methods at values of ϵ_{anharm} used in Figs. 1 and 2. The statistical errors of RMSEs of interpolation were estimated with block averaging.[45]

Triangulation	Delaunay				Anisotropic	
use of "outward push"	No		Yes		Yes	
ϵ_{anharm}	RMSE $\times 10^4$ (a.u.)	Mesh points	RMSE $\times 10^4$ (a.u.)	Mesh points	RMSE $\times 10^4$ (a.u.)	Mesh points
0	0	290	0	28	0	28
0.01	9.622 ± 0.003	2975	9.350 ± 0.003	2864	10.120 ± 0.004	1278

$$x_1 = |\mathrm{r_C} - \mathrm{r_N}|, \tag{15}$$

$$x_2 = \frac{(\mathrm{r_{D/H}} - \mathrm{r_C}) \cdot (\mathrm{r_C} - \mathrm{r_N})}{x_1}, \tag{16}$$

$$x_3 = \sqrt{|\mathrm{r_{D/H}} - \mathrm{r_C}|^2 - x_2^2}. \tag{17}$$

It is necessary to use the "outward push" procedure to avoid the mesh approaching infinitely closely the $x_3 = 0$ surface due to the $x_3 \geq 0$ constraint, for reasons discussed in Subsection III A.

1. Numerical details

For each temperature $T = 200, 300, \ldots, 900, 1000$ K, we ran a path integral Monte Carlo simulation of DCN with the isotope effects (14) calculated by averaging the corresponding mass-scaled direct isotope effect estimator. Each Monte Carlo simulation was of 1.25×2^{23} ($\approx 1.05 \times 10^7$) steps, with 20% being displacements of the entire ring polymer as a whole and the other 80% being staging transformation[52,53] movements of one fourth of the ring-polymer. The first 20% of the Monte Carlo simulations were discarded as a warm-up, while during the rest of the simulation, the mass-scaled direct estimator was calculated every 8 Monte Carlo steps (to avoid wasting computational effort on calculating correlated samples); the statistical error of its average was estimated as the root mean square error evaluated with block averaging.[45] The number of replicas P was chosen as 256 and 32 for 200 K and 1000 K, respectively; it was verified with separate calculations that doubling P did not change the isotope effect by more than 1%. For the other temperatures, the P was assigned by linear interpolation of P values as a function of $1/T$. We set $\delta V_{\text{max}} = 10^{-4}$ a.u., and after each successful interpolation, the algorithm had a 10^{-5} probability to carry out an additional exact potential energy calculation in order to estimate the RMSE of interpolation.

2. Results and discussion

In Table II, isotope effects calculated with our interpolation algorithm are compared to benchmark values calculated with the original force field and with harmonic approximation[54–56] values. Interpolation allows reproducing benchmark isotope effect values with an error below 1%; the decent agreement between the harmonic approximation values and the formally exact path integral results is expected considering HCN is a fairly harmonic molecule.

The RMSEs of interpolation and the number of mesh points generated during the simulations are displayed in Table III. If

TABLE II. HCN/DCN isotope effect values calculated with the path integral Monte Carlo method and with the harmonic approximation. The path integral simulations were done using both the original force field (the "benchmark" calculation) and our interpolation algorithm employing either the Delaunay or anisotropic triangulation. P is the number of imaginary-time slices used.

		Path integral calculations			Harmonic
T (K)	P	Delaunay mesh	Anisotropic mesh	Benchmark	approximation
200	256	4.5356 ± 0.0019	4.5379 ± 0.0019	4.5358 ± 0.0018	4.635
300	162	3.1623 ± 0.0008	3.1639 ± 0.0009	3.1623 ± 0.0012	3.228
400	116	2.5028 ± 0.0008	2.5005 ± 0.0009	2.5012 ± 0.0009	2.550
500	88	2.1178 ± 0.0009	2.1166 ± 0.0010	2.1190 ± 0.0007	2.157
600	70	1.8684 ± 0.0007	1.8701 ± 0.0007	1.8696 ± 0.0007	1.903
700	56	1.6985 ± 0.0006	1.6983 ± 0.0005	1.6985 ± 0.0005	1.728
800	46	1.5746 ± 0.0005	1.5751 ± 0.0005	1.5744 ± 0.0005	1.600
900	38	1.4827 ± 0.0005	1.4822 ± 0.0005	1.4824 ± 0.0004	1.505
1000	32	1.4113 ± 0.0004	1.4115 ± 0.0005	1.4108 ± 0.0004	1.431

TABLE III. Root mean square errors (RMSEs) $\langle[\tilde{V}(\tilde{\mathbf{r}})-V(\tilde{\mathbf{r}})]^2\rangle^{1/2}$ of interpolation and the number of mesh points generated for path integral HCN/DCN isotope effect calculations along with the number of exact potential energy surface calculations that were required by the benchmark calculations. The statistical errors of RMSEs of interpolation were estimated with block averaging.[45]

	Interpolation procedure				
	Delaunay triangulation		Anisotropic triangulation		Number of exact PES calculations in
T (K)	RMSE $\times 10^6$ (a.u.)	Mesh points	RMSE $\times 10^6$ (a.u.)	Mesh points	benchmark simulation
200	3.13 ± 0.04	3.11×10^4	4.18 ± 0.06	3.15×10^4	1.34×10^9
300	3.06 ± 0.04	2.32×10^4	4.56 ± 0.17	2.30×10^4	8.45×10^8
400	2.98 ± 0.05	2.07×10^4	4.33 ± 0.10	1.84×10^4	6.08×10^8
500	3.09 ± 0.06	1.96×10^4	5.7 ± 0.8	1.80×10^4	4.61×10^8
600	3.04 ± 0.07	1.95×10^4	3.94 ± 0.12	1.90×10^4	3.63×10^8
700	3.18 ± 0.07	1.96×10^4	3.83 ± 0.16	2.65×10^4	2.94×10^8
800	2.94 ± 0.08	1.99×10^4	5.4 ± 0.5	2.47×10^4	2.37×10^8
900	3.20 ± 0.09	2.00×10^4	3.79 ± 0.13	2.12×10^4	1.95×10^8
1000	2.95 ± 0.09	2.09×10^4	3.86 ± 0.18	2.47×10^4	1.68×10^8

we used an expensive *ab initio* procedure for the exact potential, then the speedup due to the interpolation method would equal the ratio of the numbers of interpolated potential energy evaluations and the number of exact potential energy evaluations, which approximately equals to the number of mesh points generated in the simulation. In this 3-dimensional problem, this ratio is always of the order of 10^4, indicating a large potential speedup. As discussed in Subsection II A, we made an additional number of exact potential energy calculations to make sure that the choice of $\delta V_{\max}$ guarantees an adequate interpolation accuracy; however, the number of these additional calculations (approximately the number of potential evaluations during the calculation times 10^{-5}, see Subsection III B 1) was always small compared to the number of mesh points, but still large enough to estimate the RMSE of interpolation with high precision. As was the case for the quartic oscillator, the mean square interpolation error is significantly smaller than $|\delta V_{\max}|^2$. However, because HCN is a very harmonic system, the anisotropic triangulation loses its advantage over the Delaunay triangulation. Both approaches behave similarly and, in fact, the anisotropic triangulation yields slightly higher interpolation errors and generates slightly more mesh points.

TABLE IV. Root mean square errors (RMSEs) $\langle[\tilde{V}(\tilde{\mathbf{r}}) - V(\tilde{\mathbf{r}})]^2\rangle^{1/2}$ of interpolation and the number of additional mesh points generated for path integral HCN/DCN isotope effect calculations that reused the mesh generated during the calculations at 200 K presented in Table III. The statistical errors of RMSEs of interpolation were estimated with block averaging.[45]

	Interpolation procedure			
	Delaunay triangulation		Anisotropic triangulation	
T(K)	RMSE $\times 10^6$ (a.u.)	Additional mesh points	RMSE $\times 10^6$ (a.u.)	Additional mesh points
200	3.13 ± 0.04	0	4.18 ± 0.06	0
300	3.11 ± 0.04	503	4.14 ± 0.08	786
400	3.09 ± 0.05	413	4.15 ± 0.09	713
500	3.09 ± 0.06	747	3.96 ± 0.09	803
600	2.97 ± 0.06	774	3.94 ± 0.10	1102
700	3.15 ± 0.07	1591	4.08 ± 0.15	1635
800	3.03 ± 0.08	2421	3.81 ± 0.12	2467
900	3.20 ± 0.09	2795	3.87 ± 0.15	2357
1000	2.94 ± 0.08	3048	4.02 ± 0.19	3235

We have also investigated how our method performs if the mesh created during one simulation is reused for simulations at other temperatures. One would expect that the best starting point would be the mesh generated during the highest temperature simulation, which—in accordance with the classical Boltzmann distribution—tends to visit a larger region of configuration space during a fixed number of simulation steps. However, in our calculations of HCN isotope effect, we found that using the lowest temperature was preferable.[57] The results are presented in Table IV, which shows clearly that the number of extra mesh points that must be added to the mesh generated during the lowest temperature simulation is relatively small for simulations at all other temperatures.

IV. CONCLUSION

We have proposed an algorithm for interpolating potential energy values from the values of the potential energy and its gradient calculated and stored for points of a mesh generated during a Monte Carlo simulation. The interpolation procedure is exact in harmonic systems, while in anharmonic systems, its accuracy depends on the triangulation procedure chosen for the mesh. For the latter, we considered two choices: the previously used Delaunay triangulation and an anisotropic triangulation designed to decrease the interpolation

error. Both triangulations combined with subsequent interpolation resulted in a very large reduction of potential energy evaluations in comparison with a purely on-the-fly approach. Moreover, we found that for nearly harmonic systems, the two triangulations give similar results, with the Delaunay triangulation demonstrating superior performance in some cases, but for more anharmonic systems, the proposed anisotropic triangulation achieves similar interpolation errors with significantly fewer mesh points. The *ad hoc* procedure used for construction of such anisotropic triangulations may be used to improve the performance of other interpolants,[31–33,35] even though a different definition of $\delta V(\tilde{\mathbf{r}})$ [Eq. (6)] may prove more convenient.

To combine our interpolation algorithm with classical or semiclassical molecular dynamics simulations, one may need to use a different interpolant, as mentioned in Subsection II A; the "outward push" procedure would also need to be extended to points added inside $\mathcal{C}_{\text{mesh}}$ to avoid forming nearly degenerate simplices. By contrast, as mentioned in Subsection II C, searching for the simplex used in interpolation should become even simpler.

It is important to discuss the scaling of our interpolation procedure with respect to two parameters: number of points in the mesh and dimensionality. For the former, adding new points to the mesh is done at a cost that does not depend on the number of points already in the mesh and the cost of finding the simplex used in interpolation scales logarithmically with the number of mesh points; calculating the interpolant costs the same regardless of the number of mesh points. This behavior compares favorably to the Shepard interpolation and Gaussian process regression, which utilize functions whose evaluation cost is proportional to the number of mesh points; triangulating the mesh is a natural way to avoid the issue. However, the cost of storing and updating the triangulation increases dramatically with dimensionality[29] even if one does not take into account the increase in the needed number of mesh points (which also grows quickly with dimensionality, at least for lower dimensions). This problem is likely to be decisive if one wanted to apply our method to systems of dimensionality six and higher (corresponding to molecules with four atoms and more). Yet, in this work, we demonstrated a significant potential speedup achieved by such algorithms in the simulations of two- and three-dimensional systems.

ACKNOWLEDGMENTS

The authors acknowledge the financial support from the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme (Grant Agreement No. 683069—MOLEQULE). They also thank Fabio Albertani for useful discussions.

APPENDIX: PSEUDOCODE OF THE TRIANGULATION ALGORITHM

The pseudocode for updating the mesh triangulation given no $D + 1$ points lie in the same hyperplane is outlined in Algorithm 1, which uses the "expand_convex_hull" subroutine described in Algorithm 2. The following notation is used:

- $\mathbf{r}_{\text{add}}$ is the point being added to the mesh.
- We reserve $\mathcal{S}$ for simplex variables (consisting of $D + 1$ points), $\mathcal{F}$ for face variables (consisting of D points), e for edge variables (consisting of $D - 1$ points; for $D = 2$, an edge consists of a single point, but we will still call it "edge" for the

ALGORITHM 1. Updating the triangulation of the mesh once a new point $\mathbf{r}_{\text{add}}$ is added.

if ($\mathbf{r}_{\text{add}}$ is inside a simplex $\mathcal{S} \in \boldsymbol{\mathcal{S}}_{\text{mesh}}$) **then**
  combine each face of $\mathcal{S}$ with $\mathbf{r}_{\text{add}}$ to create an array $\boldsymbol{\mathcal{S}}_{\text{new}}$ of $D + 1$ simplices;
  delete $\mathcal{S}$;
else
  call expand_convex_hull($\mathbf{r}_{\text{add}}$, $\boldsymbol{\mathcal{S}}_{\text{new}}$);
end if
while (array $\boldsymbol{\mathcal{S}}_{\text{new}}$ is not empty) **do**
  choose a random $\mathcal{S}' \in \boldsymbol{\mathcal{S}}_{\text{new}}$;
  delete $\mathcal{S}'$ from $\boldsymbol{\mathcal{S}}_{\text{new}}$;
  for each [$\mathcal{S}''$ that shares a face with $\mathcal{S}'$ (chosen in random order)] **do**
    |∗ Lawson flip ∗|
    form convex hull $\mathcal{C}$ from vertices of $\mathcal{S}''$ and $\mathcal{S}'$;
    attempt to create array $\boldsymbol{\mathcal{S}}_{\text{current}}$ which triangulates $\mathcal{C}$ with currently existing simplices;
    attempt to create array $\boldsymbol{\mathcal{S}}_{\text{flipped}}$ which also triangulates $\mathcal{C}$ and differs from $\boldsymbol{\mathcal{S}}_{\text{current}}$;
    if [both $\boldsymbol{\mathcal{S}}_{\text{current}}$ and $\boldsymbol{\mathcal{S}}_{\text{flipped}}$ exist and $G(\boldsymbol{\mathcal{S}}_{\text{current}}, \boldsymbol{\mathcal{S}}_{\text{flipped}}) > 0$] **then**
      delete all simplices of $\boldsymbol{\mathcal{S}}_{\text{current}}$ from $\boldsymbol{\mathcal{S}}_{\text{mesh}}$ and (where present) $\boldsymbol{\mathcal{S}}_{\text{new}}$;
      add all simplices of $\boldsymbol{\mathcal{S}}_{\text{flipped}}$ to $\boldsymbol{\mathcal{S}}_{\text{mesh}}$ and $\boldsymbol{\mathcal{S}}_{\text{new}}$;
      exit the **for** loop;
    end if
  end for
end while

ALGORITHM 2. Procedure for expanding $\mathcal{C}_{\text{mesh}}$ once a new point $\mathbf{r}_{\text{add}}$ is added outside of it.

```
procedure expand_convex_hull(r_add, S_new)
  find a face F_start such that n_{F_start}(r_add) < 0;
  |* F_conf will consist of all "conflicting" faces F such that n_F(r_add) < 0, e_bound will consist of
edges of F_conf not shared by two faces in the array *|
  call expand_conflict_zone(r_add, F_start, F_conf, e_bound)
  for each (F ∈ F_conf) do
    create a simplex S from r_add and F;
    add S to the current triangulation and S_new;
  end for
  for each (e ∈ e_bound) do
    create a face F from r_add and e;
    add F to F_mesh;
  end for
end procedure
recursive procedure expand_conflict_zone(r_add, F_in, F_conf, e_bound)
  add F_in to F_conf;
  for each (F' sharing an edge e with F_in) do
    if (F' ∈ F_conf) cycle
    if [n_{F'}(r_add) < 0] then
      call expand_conflict_zone(r_add, F', F_conf)
    else
      add e to e_bound;
    end if
  end for
end procedure
```

sake of generality), and bold font for variables that are arrays of simplices, faces, or edges.

- $\mathcal{S}_{\text{mesh}}$ and $\mathcal{F}_{\text{mesh}}$ are the current lists of simplices and faces of the convex.

Algorithm 2 is quite similar to the "conflict zone" procedure for updating a Delaunay triangulation once a new point is added.[29] The latter completes all convex faces with a virtual point at infinity, thus creating "virtual simplices," and constructs a "conflict zone" from simplices that have a "conflict" with the new point by containing it inside their circumsphere (for virtual simplices, this means that the point is in the half-space on the other side of the face's hyperplane than the rest of the mesh points); the conflict zone is then replaced with new simplices. By constraining this algorithm to include in the conflict zone only virtual simplices, one obtains the "expand_convex_hull" subroutine. The concept of "virtual simplices" from Ref. 29 can be used to avoid using a separate routine for expanding the convex hull. In this case, the virtual simplex corresponding to $\mathcal{F}_{\text{start}}$ can be considered the one containing the new point $\mathbf{r}_{\text{add}}$, and the algorithm can proceed directly to Lawson flips with the definition of G [see Eq. (10)] extended to cases when virtual simplices are included into the sum. We still use the "expand_convex_hull" subroutine to make the triangulation computationally cheaper, but we will use the notion of virtual simplices a little later.

As mentioned in Sec. II, it is beneficial to place some mesh points on constraining surfaces (9). In this case, each point of the mesh is assigned a logical variable array whose ith element indicates whether the point lies in the constraining plane with index i. To account for constraints, the following modifications should be made to Algorithms 1 and 2:

1. Each evaluation of $n_{\mathcal{F}}(\mathbf{r}_{\text{add}})$ should be preceded by checking whether a face $\mathcal{F}$ and $\mathbf{r}_{\text{add}}$ lie in the same constraining plane; if this is the case, $n_{\mathcal{F}}(\mathbf{r}_{\text{add}})$ is considered exactly zero. If the face $\mathcal{F}$ does not lie in a single constraining plane in the first place, the logical check should return false.
2. Each time $G(\mathcal{S}_{\text{current}})$ and $G(\mathcal{S}_{\text{flipped}})$ are evaluated, it should be checked whether $\mathcal{S}_{\text{flipped}}$ contains any simplex whose volume is zero (for example, if all of the vertices lie in a single hyperplane). If one of the faces of this simplex has zero area, the flip is considered impossible. Otherwise, faces of such simplices are turned into virtual simplices to be added to $\mathcal{S}_{\text{current}}$ if they are already present in the triangulation and to $\mathcal{S}_{\text{flipped}}$ otherwise. Such virtual simplices are not added to $\mathcal{S}_{\text{new}}$ after a successful Lawson flip and we consider their volume to be zero while evaluating the finalized $G(\mathcal{S}_{\text{current}})$ and $G(\mathcal{S}_{\text{flipped}})$.

REFERENCES

[1] C. Song and T. J. Martínez, J. Chem. Phys. **144**, 174111 (2016).
[2] G. Jeanmairet, S. Sharma, and A. Alavi, J. Chem. Phys. **146**, 044107 (2017).
[3] A. Pérez, M. E. Tuckerman, H. P. Hjalmarson, and O. A. von Lilienfeld, J. Am. Chem. Soc. **132**, 11510 (2010).

[4]P. Gasparotto, A. A. Hassanali, and M. Ceriotti, J. Chem. Theory Comput. **12**, 1953 (2016).
[5]R. P. de Tudela and D. Marx, Phys. Rev. Lett. **119**, 223001 (2017).
[6]G. A. Worth, M. A. Robb, and I. Burghardt, Faraday Discuss. **127**, 307 (2004).
[7]R. Ianconescu, J. Tatchen, and E. Pollak, J. Chem. Phys. **139**, 154311 (2013).
[8]B. F. E. Curchod and T. J. Martínez, Chem. Rev. **118**, 3305 (2018).
[9]K. E. Spinlove, G. W. Richings, M. A. Robb, and G. A. Worth, Faraday Discuss. **212**, 191 (2018).
[10]A. Patoz, T. Begušić, and J. Vaníček, J. Phys. Chem. Lett. **9**, 2367 (2018).
[11]M. Micciarelli, F. Gabas, R. Conte, and M. Ceotto, J. Chem. Phys. **150**, 184113 (2019).
[12]G. Laude, D. Calderini, D. P. Tew, and J. O. Richardson, Faraday Discuss. **212**, 237 (2018).
[13]R. Conte, F. Gabas, G. Botti, Y. Zhuang, and M. Ceotto, J. Chem. Phys. **150**, 244118 (2019).
[14]J. Ischtwan and M. A. Collins, J. Chem. Phys. **100**, 8080 (1994).
[15]T. J. Frankcombe, M. A. Collins, and G. A. Worth, Chem. Phys. Lett. **489**, 242 (2010).
[16]C. Evenhuis and T. J. Martínez, J. Chem. Phys. **135**, 224110 (2011).
[17]C. W. Kim and Y. M. Rhee, J. Chem. Theory Comput. **12**, 5235 (2016).
[18]X. Huang, B. J. Braams, and J. M. Bowman, J. Chem. Phys. **122**, 044308 (2005).
[19]Y. Guo, A. Kawano, D. L. Thompson, A. F. Wagner, and M. Minkoff, J. Chem. Phys. **121**, 5091 (2004).
[20]R. Dawes, X.-G. Wang, A. W. Jasper, and T. Carrington, Jr., J. Chem. Phys. **133**, 134304 (2010).
[21]J. P. Alborzpour, D. P. Tew, and S. Habershon, J. Chem. Phys. **145**, 174112 (2016).
[22]G. W. Richings and S. Habershon, Chem. Phys. Lett. **683**, 228 (2017).
[23]T. B. Blank, S. D. Brown, A. W. Calhoun, and D. J. Doren, J. Chem. Phys. **103**, 4129 (1995).
[24]S. Manzhos, X. Wang, R. Dawes, and T. Carrington, Jr., J. Phys. Chem. A **110**, 5295 (2006).
[25]M. Malshe, L. M. Raff, M. Hagan, S. Bukkapatnam, and R. Komanduri, J. Chem. Phys. **132**, 204103 (2010).
[26]M. Gastegger, J. Behler, and P. Marquetand, Chem. Sci. **8**, 6924 (2017).
[27]M. R. Salazar and R. L. Bell, J. Comput. Chem. **19**, 1431 (1998).
[28]M. R. Salazar, Chem. Phys. Lett. **359**, 460 (2002).
[29]S. Hornus and J. D. Boissonnat, "An efficient implementation of Delaunay triangulations in medium dimensions," Research Report RR-6743, INRIA, 2008.
[30]R. W. Clough and J. L. Tocher, in *Proceedings of the Conference on Matrix Methods in Structural Mechanics* (Air Force Flight Dynamics Laboratory, 1966), pp. 515–545.
[31]P. Alfeld, Comput. Aided Geom. Des. **1**, 169 (1984).
[32]P. Alfeld, SIAM J. Numer. Anal. **22**, 95 (1985).
[33]F. Dell'Accio, F. D. Tommaso, O. Nouisser, and B. Zerroudi, Appl. Numer. Math. **126**, 78 (2018).
[34]H. Xuli, J. Approximations Theory **124**, 242 (2003).
[35]A. Guessab, O. Nouisser, and G. Schmeisser, J. Comput. Appl. Math. **196**, 162 (2006).
[36]R. P. A. Bettens and M. A. Collins, J. Chem. Phys. **111**, 816 (1999).
[37]C. L. Lawson, Comput. Aided Geom. Des. **3**, 231 (1986).
[38]H. Edelsbrunner, Acta Numer. **9**, 133 (2000).
[39]J. M. Mirebeau, Constr. Approx. **32**, 339 (2010).
[40]F. J. Bossen and P. Heckbert, in *Proceedings of the 5th International Meshing Roundtable* (Sandia National Laboratories, 1996), p. 115.
[41]K. Shimada, A. Yamada, and T. Itoh, Int. J. Comput. Geom. Appl. **10**, 417 (2000).
[42]J. D. Boissonnat, M. Rouxel-Labbé, and M. Wintraecken, in *33rd International Symposium on Computational Geometry (SoCG 2017), Leibniz International Proceedings in Informatics (LIPIcs)*, edited by B. Aronov and M. J. Katz (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2017), Vol. 77, pp. 19:1–19:16.
[43]O. Devillers, S. Pion, and M. Teillaud, Int. J. Found. Comput. Sci. **13**, 181 (2002).
[44]J. L. Bentley, Commun. ACM **18**, 509 (1975).
[45]H. Flyvbjerg and H. G. Petersen, J. Chem. Phys. **91**, 461 (1989).
[46]M. F. Herman, E. J. Bruskin, and B. J. Berne, J. Chem. Phys. **76**, 5150 (1982).
[47]M. E. Tuckerman, B. J. Berne, G. J. Martyna, and M. L. Klein, J. Chem. Phys. **99**, 2796 (1993).
[48]R. P. Feynman and A. R. Hibbs, *Quantum Mechanics and Path Integrals* (McGraw-Hill, 1965).
[49]A. Pérez and O. A. von Lilienfeld, J. Chem. Theory Comput. **7**, 2358 (2011).
[50]B. Cheng and M. Ceriotti, J. Chem. Phys. **141**, 244112 (2014).
[51]V. Y. Makhnev, A. A. Kyuberis, O. L. Polyansky, I. I. Mizus, J. Tennyson, and N. F. Zobov, J. Mol. Spectrosc. **353**, 40 (2018).
[52]M. Sprik, M. L. Klein, and D. Chandler, Phys. Rev. B **31**, 4234 (1985).
[53]M. Sprik, M. L. Klein, and D. Chandler, Phys. Rev. B **32**, 545 (1985).
[54]H. C. Urey, J. Chem. Soc. **1947**, 562.
[55]M. Wolfsberg, W. A. V. Hook, P. Paneth, and L. P. N. Rebelo, *Isotope Effects in the Chemical, Geological and Bio Sciences* (McGraw-Hill, 2010).
[56]M. A. Webb and T. F. Miller III, J. Phys. Chem. A **118**, 467 (2014).
[57]Table III shows that the largest number of mesh points was generated during the lowest temperature simulation. Because the spacing of mesh points does not depend on temperature, this suggests that the mesh generated at the lowest temperature covered the largest region of configuration space and, therefore, should be used as a starting mesh. This counterintuitive observation can be explained as follows: The lower the temperature, the greater the quantum delocalization of the ring polymer and the greater the explored region of configuration space. Although this delocalization diminishes at higher temperatures, it is eventually replaced by an ever increasing motion of the center of the ring polymer associated with the classical Boltzmann distribution. However, we cannot see this transition yet in Table III, most likely because the quantum delocalization at lower temperatures is further increased due to the use of mass-scaled direct estimators, which stretch the ring polymer by a factor of $\sqrt{2}$.[50]