Multilevel summation with B-spline interpolation for pairwise interactions in molecular dynamics simulations

David J. Hardy, Matthew A. Wolff, Jianlin Xia, Klaus Schulten, and Robert D. Skeel

Citation: The Journal of Chemical Physics **144**, 114112 (2016); doi: 10.1063/1.4943868
View online: http://dx.doi.org/10.1063/1.4943868
View Table of Contents: http://scitation.aip.org/content/aip/journal/jcp/144/11?ver=pdfcov
Published by the AIP Publishing

Multilevel summation with B-spline interpolation for pairwise interactions in molecular dynamics simulations

David J. Hardy,[1,a)] Matthew A. Wolff,[2] Jianlin Xia,[3] Klaus Schulten,[1] and Robert D. Skeel[2]
[1]*Beckman Institute, University of Illinois, 405 North Mathews Avenue, Urbana, Illinois 61801, USA*
[2]*Department of Computer Science, Purdue University, 305 North University Street, West Lafayette, Indiana 47907, USA*
[3]*Department of Mathematics, Purdue University, 150 North University Street, West Lafayette, Indiana 47907, USA*

(Received 8 January 2016; accepted 1 March 2016; published online 21 March 2016)

The multilevel summation method for calculating electrostatic interactions in molecular dynamics simulations constructs an approximation to a pairwise interaction kernel and its gradient, which can be evaluated at a cost that scales linearly with the number of atoms. The method smoothly splits the kernel into a sum of partial kernels of increasing range and decreasing variability with the longer-range parts interpolated from grids of increasing coarseness. Multilevel summation is especially appropriate in the context of dynamics and minimization, because it can produce continuous gradients. This article explores the use of B-splines to increase the accuracy of the multilevel summation method (for nonperiodic boundaries) without incurring additional computation other than a preprocessing step (whose cost also scales linearly). To obtain accurate results efficiently involves technical difficulties, which are overcome by a novel preprocessing algorithm. Numerical experiments demonstrate that the resulting method offers substantial improvements in accuracy and that its performance is competitive with an implementation of the fast multipole method in general and markedly better for Hamiltonian formulations of molecular dynamics. The improvement is great enough to establish multilevel summation as a serious contender for calculating pairwise interactions in molecular dynamics simulations. In particular, the method appears to be uniquely capable for molecular dynamics in two situations, nonperiodic boundary conditions and massively parallel computation, where the fast Fourier transform employed in the particle–mesh Ewald method falls short. [http://dx.doi.org/10.1063/1.4943868]

I. INTRODUCTION

The calculation of pairwise interactions among a large set of particles is vital to many simulations of physical phenomena. This calculation is done either directly or using fast methods. For doing fast N-body calculations, there are two common approaches: The first is to use hierarchical clustering methods (HCMs) such as the fast multipole method (FMM) and tree methods. The second, especially for periodic boundaries, is to use methods based on the fast Fourier transform (FFT), such as the (smooth) particle–mesh Ewald (PME)[1] and particle–particle particle–mesh (P^3M)[2] methods. However, many of these computations can be done more quickly using a relatively obscure algorithm known as the multilevel summation method (MSM).[3] The MSM is a simple, yet flexible, linear time algorithm based on multiscale piecewise polynomial interpolation of the interaction kernel and well suited for modern computer architectures, due to its use of moderately large grid stencils. Indeed, multilevel summation might be expected to outperform other methods for important classes of problems and to be a formidable competitor in other situations. In particular, the method appears to be uniquely capable for molecular dynamics in two situations, nonperiodic systems and massively parallel computation, where the FFT falls short.[4]

The calculation of pairwise interactions and the solution of elliptic partial equations are the time-limiting steps of applications that consume vast amounts of CPU cycles. Molecular dynamics (MD), in particular, can require months of computer time; hence, the extraordinary efforts to maximize performance, such as Desmond,[5] OpenMM,[6] NAMD,[7] and GROMACS.[8] The work presented here is motivated by problems in computational molecular biophysics. Of course, there are many other applications, including atomic level and coarser-grained simulation of all types of materials, particle methods for fluid dynamics, astrophysics, the Coulomb term in Hartree-Fock and density functional theory, integral transforms, and partial differential equations having explicit solutions involving integrals.[9]

Though the MSM will benefit many applications, it is uniquely qualified for molecular simulations involving nonperiodic boundary conditions, including solvent boundary potentials[10,11] and the modeling of implicit (i.e., continuum) solvent.

The multilevel summation method was introduced for integral transforms in 1990.[3] It was later applied to

a)Author to whom correspondence should be addressed. Electronic mail: dhardy@illinois.edu

particle monopoles and dipoles in 2D,[12] C^1 kernels for particles in 3D,[13] eigenvalues,[14] generalized Born potentials,[15] interseismic stress interactions,[16] Madelung constants of ionic crystals,[17] and dispersion interactions.[18] The MSM has been shown to have good parallel scalability compared to PME,[19] it is an option in the molecular simulators NAMD[4] and LAMMPS,[20] and it has been used for a GPU implementation of the electrostatic potential calculation[21] in the molecular visualization and analysis program VMD.[22] One study[21] produces a speedup of 26.4, over the use of a CPU alone, for calculating a map of the electrostatic potential of a 1.5×10^6 atom system. A recent article[23] examines various implementation issues, including error estimation. This article as well as Ref. 4 involving the use of MSM in NAMD and the thesis (Ref. 24, Section 6.2) demonstrates that the MSM handles in a straightforward manner periodic boundary conditions in 1, 2, or 3 coordinates, including general parallelepipeds.

A concern of previous studies[4,13,23] of the multilevel summation method is its accuracy as a function of computational effort. The principal contribution of the present article is to address this shortcoming, by showing how to implement B-splines for nonperiodic boundaries and how their use makes the multilevel summation method an exceptionally efficient algorithm for MD. B-spline approximation has proved effective for the popular algorithm (S)PME[1] for Coulomb interactions with periodic boundary conditions. However, for nonperiodic boundary conditions, there is currently no satisfactory algorithm. B-splines have several advantages over the C^1 piecewise polynomials used previously: In addition to higher regularity, B-splines provide one order of accuracy higher for the same work and provide nested function spaces for nested grids, making prolongation operations exact. As a consequence of the nesting property, reduced grid extensions are possible. The disadvantage of a B-spline is that it is nonzero at several grid points, which necessitates a preprocessing step to determine coefficients. B-spline coefficients for an interpolant of a given function can be obtained by convolving point values of that function with a special sequence that is derived from the B-spline itself.[25] Given in the present article is a complete algorithm, which takes care of three difficulties. The first two are (i) generating the special sequence automatically and (ii) reducing the interpolation in 6 variables of the kernel to interpolation in 3 variables. The third is reducing the preprocessing time, for those situations where this matters and doing so with minimal loss of accuracy. This is accomplished by *quasi-interpolation*, which maintains the desired order of accuracy at the expense of collocation (exactly matching values of the given function). Presented here is a novel algorithm that provides a stable way to do quasi-interpolation with arbitrarily small collocation error. Numerical experiments indicate that the proposed MSM implementation is competitive with a modern C implementation of the FMM,[26] and it is *several times faster* if the fast multipole parameters are chosen to avoid energy drift (which is consistent with previous results[13]). A secondary contribution of this article is some insight into the basic structure of the MSM and other N-body methods.

A. Specification of task

Considered here are Coulomb interactions in 3 dimensions. Let $\mathbf{r}_i$ denote the position of particle i, and let q_i be its partial charge. The task is to compute sums of the form

$$U(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N) = \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \notin \chi(i)}}^{N} q_i q_j k(\mathbf{r}_i, \mathbf{r}_j), \qquad k(\mathbf{r}, \mathbf{r}') = \frac{1}{|\mathbf{r} - \mathbf{r}'|} \tag{1}$$

as well as derivatives of such sums. Here $\chi(i)$ consists of the indices of those particles that are excluded: $j = i$ and in the case of molecular dynamics simulations also values of j corresponding to atoms covalently bonded to atom i or to another atom that is covalently bonded to atom i. Generally, there is also a constant prefactor, which is omitted here. Kernels other than $k(\mathbf{r}, \mathbf{r}') = |\mathbf{r} - \mathbf{r}'|^{-1}$ are possible, as are extensions to dipoles, etc. It is computationally advantageous to have $k(\mathbf{r}, \mathbf{r}') = \kappa(\mathbf{r} - \mathbf{r}')$. (For multilevel summation, it substantially reduces memory requirements.)

B. Basics of multilevel summation methods

The idea of the multilevel summation method is to create a multilevel separable approximation of the form

$$k(\mathbf{r}, \mathbf{r}') \approx k_0(\mathbf{r}, \mathbf{r}') + \tilde{k}_{1+}(\mathbf{r}, \mathbf{r}'), \qquad \tilde{k}_{1+}(\mathbf{r}, \mathbf{r}') = \sum_{l=1}^{L} \sum_{\mathbf{m}} \sum_{\mathbf{n}} \varphi^l_{\mathbf{m}}(\mathbf{r}) \mathcal{K}^l_{\mathbf{m},\mathbf{n}} \varphi^l_{\mathbf{n}}(\mathbf{r}'). \tag{2}$$

The superscripts l index nested grid levels and the subscripts $\mathbf{m}$, $\mathbf{n}$ index grid points. The finest grid has a grid size h chosen so that it has $O(N)$ grid points. The first term $k_0(\mathbf{r}, \mathbf{r}')$ is a direct calculation with a cutoff a that is a small multiple of h. The basis functions $\varphi^l_{\mathbf{m}}(\mathbf{r})$ have local support. In particular, for any value of $\mathbf{r}$, there are at most p^3 values of $\mathbf{m}$ for which $\varphi^l_{\mathbf{m}}(\mathbf{r})$ is nonzero, where $p - 1$ is the degree of the piecewise polynomial. Also, for any level $l < L$ and any $\mathbf{m}$, there are only about $\frac{4}{3}\pi(2a/h)^3$ values of $\mathbf{n}$ for which $\mathcal{K}^l_{\mathbf{m},\mathbf{n}}$ is nonzero, e.g., 3239 if $a/h = 4.6$.

The utility of approximation (2) is realized when applied to sum (1) with a large number of particles N. If the distribution of particles is roughly uniform, as it is for condensed matter, then the cost of the direct calculation using k_0 is $O(N)$. The calculation for grid level l can be done as follows:

$$\sum_{\mathbf{m}} \sum_{\mathbf{n}} \mathcal{K}^l_{\mathbf{m},\mathbf{n}} q^l_{\mathbf{m}} q^l_{\mathbf{n}},$$

where

$$q^l_{\mathbf{m}} = \sum_i q_i \varphi^l_{\mathbf{m}}(\mathbf{r}_i). \tag{3}$$

The number of nonzeros in $\varphi^1_{\mathbf{m}}(\mathbf{r}_i)$ for each i is p^3, yielding an operation count of $O(N)$ for q^1. The two-scale relation for B-splines enables a nested calculation of the $q^l_{\mathbf{m}}$, $l > 1$. It has the form

$$q^l_{\mathbf{m}} = \sum_{\mathbf{n}} (\mathcal{I}^l_{l-1})_{\mathbf{m},\mathbf{n}} q^{l-1}_{\mathbf{n}}, \tag{4}$$

where the sum over $\mathbf{n}$ has $(p+1)^3$ nonzero terms. There are $O(8^{1-l}N)$ elements in q^l, so the operation count for all levels l is $O(N)$. The sum over $\mathbf{m}$ in Eq. (3) has $O(8^{1-l}N)$ terms, and the sum over $\mathbf{n}$ has about $\frac{4}{3}\pi(2a/h)^3$ terms—for levels $l < L$. If L is chosen so that the coarsest grid has about $N^{1/2}$ points, the double sum for $l = L$ will have about N terms. The total operation count is thus $O(N)$. If forces are to be computed, the symmetry of the energy expression cannot be exploited and the computation is not quite as straightforward; see Section III A. Also discussed there is the treatment of excluded interactions.

C. Outline

Sections II A–II E consider approximation issues. The multilevel approximation, Eq. (2), is based on a splitting of the kernel $k(\mathbf{r},\mathbf{r}')$. Section II A shows that the splitting proposed in Ref. 13 is optimal in a certain theoretical sense. Construction of the B-spline coefficients $\mathcal{K}^l_{\mathbf{m},\mathbf{n}}$ is the topic of Sections II B and II C. Section II D presents theoretical results on the accuracy of B-spline interpolation, as well as an experimental comparison of the accuracy of B-splines to that of C^1 piecewise polynomials. Section II E shows how the two-scale relation for B-splines can be used to nest interpolation operations for the different levels, which does not hold exactly for C^1 piecewise polynomials.

Sections III A and III B consider algorithmic issues. Section III A gives the structure of the algorithm. Section III B compares the performance of an MSM implementation to that of an FMM.

Section IV A presents a detailed comparison between multilevel summation and alternative methods.

D. Conclusion

The multilevel summation method has two components (a softener for the interaction kernel and an interpolation scheme) and three parameters (grid size, ratio of cutoff to grid size, and order of interpolation). It combines the best features of HCMs and FFT-based 2-level methods, making it a strong candidate as the method of choice for molecular biophysics and structural biology. It shares with hierarchical clustering methods a geometry-based hierarchical structure resulting in calculations that are more parallelizable and have an essentially $O(N)$ operation count. It shares with FFT-based kernel-splitting methods their relative simplicity and the property of computing an interaction kernel having any degree of continuity. The use of B-splines reduces the error of multilevel summation by an order of magnitude compared to previously used C^1 interpolants. Coupled with innovative quasi-interpolation techniques, this significantly improves the performance of multilevel summation, as shown by numerical experiments and a partial error analysis. The availability of a fast method for nonperiodic boundaries encourages the development and use of models that do not require periodicity.

II. THEORY

A. Splitting the kernel

Multilevel summation is based on separation of length scales and interpolation from grids for all but the shortest length scale. The separation of scales is effected by splitting the kernel into a sum of partial kernels of increasing range and increasing length scale. In particular, an $(L+1)$-level summation method uses

$$k(\mathbf{r},\mathbf{r}') = k_0(\mathbf{r},\mathbf{r}') + k_1(\mathbf{r},\mathbf{r}') + \cdots + k_L(\mathbf{r},\mathbf{r}'), \tag{5}$$

where the short-range part k_0 is calculated directly, and the other parts are interpolated as functions of $\mathbf{r}$, $\mathbf{r}'$ from pairs of identical 3-dimensional grids of increasing coarseness. In this way, the problem is reduced to calculating interactions between nearby particles at level 0 and between nearby grid points at higher levels. Typically, the terms of the split kernel would have ranges $a, 2a, \ldots, 2^L a, +\infty$, respectively, where a is a cutoff parameter. Because the range and the grid size are both doubling at each level, the number of interactions per grid point is the same at each level. For a kernel depending only on distance, $k(\mathbf{r},\mathbf{r}') = g(|\mathbf{r}-\mathbf{r}'|)$, one can define a splitting

$$g(r) = g_0(r) + g_1(r) + \cdots + g_L(r), \tag{6}$$

as illustrated in Figure 1, and define $k_l(\mathbf{r},\mathbf{r}') = g_l(|\mathbf{r}-\mathbf{r}'|)$. This approach to scale separation is proposed in Ref. 27; the original approach[3] is to use a single smoothed kernel and perform scale separation on the (approximate) discretized smoothed kernel.

The splitting of the $1/r$ kernel for the MSM is defined by a cutoff distance a and a dimensionless softening function $\gamma(\rho)$, defined to be $1/\rho$ for $\rho \geq 1$ and to have bounded higher derivatives for $\rho \leq 1$. Specific formulas for γ are derived below. Multilevel splitting is neatly expressed as

$$\frac{1}{\rho} = \gamma_0(\rho) + \frac{1}{2}\gamma_1\left(\frac{1}{2}\rho\right) + \cdots + \frac{1}{2^L}\gamma_L\left(\frac{1}{2^L}\rho\right),$$

where

$$\gamma_0(\rho) = (1/\rho) - \gamma(\rho), \qquad \gamma_l(\rho) = 2\gamma(2\rho) - \gamma(\rho),$$
$$l = 1,2,\ldots,L-1, \qquad \gamma_L(\rho) = 2\gamma(2\rho).$$

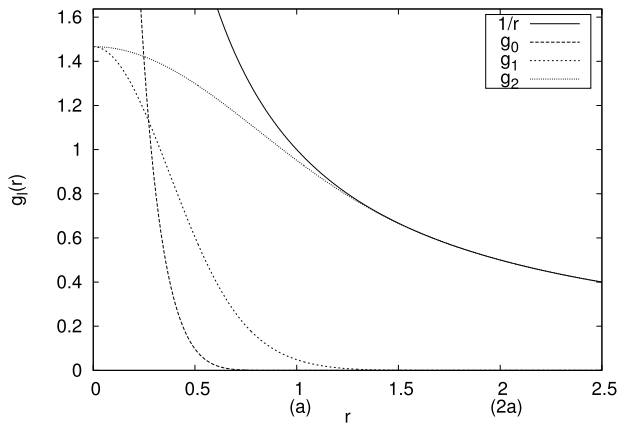

FIG. 1. A sextic even-powers splitting of the $1/r$ kernel as a sum $g(r) = g_0(r) + g_1(r) + g_2(r)$.

From this, define

$$g_l(r) = \frac{1}{a_l}\gamma_l(\frac{r}{a_l}), \quad l = 0,1,\ldots,L,$$

where $a_l = 2^l a$. Kernels $\gamma_l(\rho)$, $0 \le l \le L-1$ cut off at $\rho = 1$, and accordingly, the first L kernels $k_0, k_1, \ldots, k_{L-1}$ are zero beyond cutoff distances $a, 2a, \ldots, 2^L a$, respectively, but k_L retains the infinitely long tail of $1/r$ for $r \ge 2^L a$. Because the kernels $k_1, k_2, \ldots, k_L$ lack the singularity at zero and have higher derivatives of decreasing magnitude, they can be well approximated on grids of spacing $h, 2h, \ldots, 2^{L-1}h$, respectively, for an appropriate choice of h.

The accuracy of interpolants of order p is known to be dependent on the magnitude of the pth derivative of the function being interpolated. In particular (Ref. 24, Sec. 3.1.2), what matters for accuracy of the interpolant and its gradient is

$$M_p = \|(\partial^p/\partial u^p)\zeta\|_\infty$$

and

$$M_p' = \|(\partial/\partial v)(\partial^{p-1}/\partial u^{p-1})\zeta\|_\infty,$$

where

$$\zeta(u,v,w) = \gamma(\sqrt{u^2+v^2+w^2}).$$

This requires ζ to be C^{p-1}. Assuming p is even, it can be shown that C^{p-1} continuity implies that $(d^k/d\rho^k)\gamma(1) = (-1)^k k!$, $k = 0,1,\ldots,p-1$, and, by expanding $\gamma(|u|)$ in a Maclaurin series for each of $u \le 0$ and $u \ge 0$, that $(d^k/d\rho^k)\gamma(0) = 0, k = 1,3,\ldots,p-1$. As a heuristic, minimize $\int_0^1((d^p/d\rho^p)\gamma(\rho))^2 d\rho$ for the function $\gamma(\rho)$. Applying the calculus of variations and integrating by parts yields additional conditions

$$(d^k/d\rho^k)\gamma(0) = 0, \quad k = p+1, p+3, \ldots, 2p-1$$

and

$$(d^{2p}/d\rho^{2p})\gamma(\rho) \equiv 0.$$

The result is a softener defined in terms of even powers.

The even-powered softening functions are obtained for $\rho \le 1$ by the Taylor expansion of $\rho^{-1} = s^{-1/2}$ about $s = 1$,

$$s^{-1/2} = 1 - \frac{1}{2}(s-1) + \frac{3}{8}(s-1)^2 - \frac{5}{16}(s-1)^3 + \cdots.$$

Truncate this expansion so that $s^{-1/2} = \tau_p(s) + O((s-1)^p)$ where $\tau_p(s)$ is a polynomial of degree $p-1$. Hence, $\Delta(s) = s^{-1/2} - \tau_p(s)$ and its first $p-1$ derivatives vanish at $s = 1$, as do those of $\Delta(\rho^2) = \rho^{-1} - \tau_p(\rho^2)$. Therefore, the even-powered softening functions defined by

$$\gamma_p(\rho) = \begin{cases} \tau_p(\rho^2), & \text{for } 0 \le \rho \le 1, \\ 1/\rho, & \text{for } \rho \ge 1 \end{cases}$$

satisfy all the conditions given above. This construction is equivalent to that of Ref. 27, Eq. (13).

B. Spline interpolation

Let $f(x)$ be a bounded function defined for all real x, and consider the problem of interpolating it using splines with knots from a uniform grid $x_m = mh$, $m = 0, \pm1, \pm2, \ldots$. For basis functions, consider the use of B-splines of fixed degree $p-1$ where p is even, which are advantageous due to the minimal number of grid cells on which they are nonzero.

1. B-splines

Construction of a B-spline can be done by means of a recurrence (Ref. 1, Eq. (4.1) and Ref. 28, Thm. 4.3(viii)). Let Q_1 be the indicator function for the half-open interval $[0,1[$. The recurrence defining the B-spline Q_k of degree $k-1$ is

$$Q_k(u) = \frac{u}{k-1}Q_{k-1}(u) + \frac{k-u}{k-1}Q_{k-1}(u-1). \tag{7}$$

For interpolation, use the centered B-spline $\Phi(u) = Q_p(u + p/2)$ of degree $p-1$ as an unscaled basis function. It has local support $[-p/2, p/2]$ consisting of just p grid cells along the u axis.

A Taylor expansion for each piece of the B-spline $\Phi(u)$ can be precomputed using the recurrence in Eq. (7) and the relation (Ref. 1, Eq. (4.2) and Ref. 28, Thm. 4.3(vii))

$$(d/du)Q_k(u) = Q_{k-1}(u) - Q_{k-1}(u-1).$$

2. Interpolation in one dimension

By design, the interpolant of $f(x)$ has the form

$$\tilde{f}(x) = \sum_n \hat{f}_n \varphi_n(x),$$

where $\varphi_n(x) = \Phi(x/h - n)$. For any particular value of x, only p terms of the sum are nonzero. The coefficients $\hat{f}_n$ are chosen so that $\tilde{f}(x_m) = f(x_m)$ at all grid points x_m.

The problem of determining the interpolant $\tilde{f}(x)$ reduces to that of finding the interpolant $\Psi(u)$ that satisfies

$$\Psi(u) = \begin{cases} 1, & u = 0, \\ 0, & u = \pm1, \pm2, \ldots. \end{cases}$$

sometimes called a "fundamental" spline. Assuming there is a solution

$$\Psi(u) = \sum_m \omega_m \Phi(u-m), \tag{8}$$

the interpolant of $f(x)$ is given by

$$\tilde{f}(x) = \sum_n f(nh)\Psi(x/h - n) = \sum_m \hat{f}_m \Phi(x/h - m),$$

where

$$\hat{f}_m = \sum_n \omega_{m-n} f(nh). \tag{9}$$

It is shown in Schoenberg[25] (p. 37) and Chui[28] (p. 110) that there exists unique coefficients ω_m in Eq. (8) if $\Psi(u)$ is required to be bounded. In the limit as the degree $p \to \infty$, $\Psi(u)$ becomes the sinc function $\sin \pi u/(\pi u)$. Shown in Figure 2 are plots of the cubic, quintic, and septic fundamental splines and the sinc function.

Placing this into context, the evaluation of the interpolant requires computation at three levels:

1. Prepreprocessing: compute universal values ω_m; compute Taylor expansions for the p pieces of $\Phi(u)$ using recurrences.

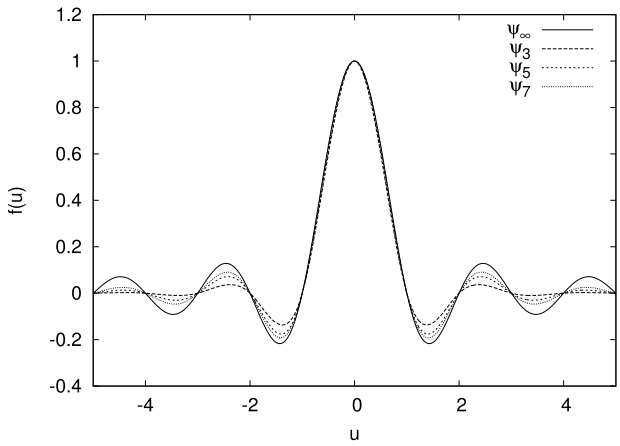

FIG. 2. The cubic, quintic, and septic fundamental splines and the sinc function.

2. Preprocessing: compute coefficients $\hat{f}_m$.
3. Processing: calculate values $\tilde{f}(x)$ (and derivatives) using Horner's rule.

To obtain formulas for prepreprocessing, it is convenient to work with discrete operators that act on sequences. With $\hat{f}$ denoting the sequence with terms $\hat{f}_n$ and f^h denoting that with terms $f^h_n = f(nh)$, one can write

$$f^h = \mathcal{B}\hat{f},$$

where

$$\mathcal{B} = \sum_{n=1-p/2}^{(p/2)-1} \Phi(-n)E^n$$

and E is the forward shift operator $(E\hat{f})_n = \hat{f}_{n+1}$. Therefore, it follows from $\hat{f} = \mathcal{B}^{-1}f^h$ and Eq. (9) that

$$\mathcal{B}^{-1} = \sum_n \omega_n E^{-n}. \tag{10}$$

Appendix A provides an algorithm for computing the coefficients ω_n.

3. Interpolation for convolution kernels in one dimension

Consider the interpolation of $F(x,x') = f(x-x')$ from values on a 2-dimensional grid with spacing h using basis functions $\varphi_m(x)\varphi_n(x')$

$$\tilde{F}(x,x') = \sum_m \sum_n \hat{F}_{mn}\varphi_m(x)\varphi_n(x').$$

With $\hat{F}$ denoting the sequence with terms $\hat{F}_{mn}$ and F^h denoting that with terms $F^h_{mn} = F(mh,nh)$, one can write

$$F^h = \mathcal{B}_1\mathcal{B}_2\hat{F},$$

where $\mathcal{B}_1$ operates on the first index and $\mathcal{B}_2$ on the second. Let f^h denote the 1-dimensional sequence with $f^h_k = f(kh)$. To relate F^h to f^h, write

$$F^h = \mathcal{T}f^h,$$

where $\mathcal{T}$ is the operator mapping a 1-dimensional sequence g to a 2-dimensional sequence $\mathcal{T}g$ defined by $(\mathcal{T}g)_{mn} = g_{m-n}$.

TABLE I. Tabulation of ω'_m for $p-1$ vs. m, $m = 0, 1, \ldots, 12$ with $\mu = \infty$.

m	Cubic	Quintic	Septic	Nonic	Undecic
0	3.464	12.379	51.971	241.384	1190.122
1	−1.732	−9.377	−45.671	−225.114	−1140.060
2	0.679	5.809	34.575	189.064	1014.420
3	−0.240	−3.266	−24.022	−147.928	−853.182
4	0.080	1.735	15.825	110.306	688.291
5	−0.026	−0.889	−10.061	−79.525	−538.377
6	0.008	0.444	6.237	55.945	411.423
7	−0.002	−0.217	−3.794	−38.635	−308.820
8	0.001	0.105	2.275	26.301	228.557
9	0.000	−0.050	−1.348	−17.700	−167.246
10	0.000	0.024	0.792	11.801	121.250
11	0.000	−0.011	−0.461	−7.807	−87.226
12	0.000	0.005	0.267	5.131	62.340

It is straightforward to show

$$\mathcal{B}_1\mathcal{T}g = \mathcal{T}\mathcal{B}g, \quad \text{and} \quad \mathcal{B}_2\mathcal{T}g = \mathcal{T}\mathcal{B}g,$$

where the second equality uses the relation $\Phi(-m) = \Phi(m)$. Therefore, $\hat{F} = \mathcal{B}_2^{-1}\mathcal{B}_1^{-1}F^h = \mathcal{B}_2^{-1}\mathcal{B}_1^{-1}\mathcal{T}f^h = \mathcal{B}_2^{-1}\mathcal{T}\mathcal{B}^{-1}f^h = \mathcal{T}\mathcal{B}^{-2}f^h$, and the B-spline interpolant of $F(x,x') = f(x-x')$ is given by

$$\tilde{F}(x,x') = \sum_m \sum_n (\mathcal{B}^{-2}f^h)_{m-n}\varphi_m(x)\varphi_n(x'). \tag{11}$$

Coefficients ω'_m in the expansion

$$\mathcal{B}^{-2} = \sum_n \omega'_n E^{-n}$$

are given in Table I.

4. Quasi-interpolation

In practice, an expansion of $\mathcal{B}^{-2}$ in negative and positive powers of E must be truncated at some point. Due to slow convergence, it is not always possible to amortize the preprocessing cost of obtaining adequate accuracy. Following the suggestion of Chui[28] (Section 4.5) the truncation is done in such a way that the property of being exact for polynomials of degree $p-1$ is preserved, thus maintaining pth order accuracy. At the same time, it is desirable to have control on the approximation error at the grid points. The expansion of Chui[28] (Eq. (4.5.14)) cannot be used, because the norm of the operator is not bounded uniformly with respect to the number of terms.

To overcome this limitation, one proceeds as follows: The central difference operator δ satisfies $\delta^2 = E - 2 + E^{-1}$, and it is not hard to see, using the B-spline symmetry,

$$\mathcal{B} = \Phi(0) + \sum_{m=1}^{(p/2)-1} \Phi(m)(E^m + E^{-m}),$$

that $\mathcal{B}$ can be expressed as a polynomial of degree $(p/2)-1$ in δ^2

$$\mathcal{B} = B_{p/2}(\delta^2).$$

A formula for the coefficients of $B_{p/2}$ is derived in Appendix B from a formula in Ref. 25. As a consequence of the formulation

in central differences, $\mathcal{B}^{-2}$ can be expressed as

$$\mathcal{B}^{-2} = 1 + b_1\delta^2 + \cdots + b_{(p/2)-1}\delta^{p-2} + \delta^p \sum_m c_m E^m, \tag{12}$$

where $c_{-m} = c_m$. For example,

$$\mathcal{B}^{-2} = \begin{cases} 1 - \frac{1}{3}\delta^2 + O(\delta^4), & p = 4, \\ 1 - \frac{1}{2}\delta^2 + \frac{41}{240}\delta^4 + O(\delta^6), & p = 6. \end{cases}$$

The truncation of $\mathcal{B}^{-2}$ given by

$$\mathcal{A} = 1 + b_1\delta^2 + \cdots + b_{(p/2)-1}\delta^{p-2} + \delta^p \sum_{m=-\mu}^{\mu} c_m E^m$$

is exact for polynomials of degree $\leq p-1$ and has an interpolation accuracy at grid points that is determined by the adjustable parameter μ. For quasi-interpolation, Eq. (11) becomes

$$\tilde{F}(x,x') = \sum_m \sum_n (\mathcal{A} f^h)_{m-n} \varphi_m(x) \varphi_n(x'). \tag{13}$$

For implementation, one expresses this using $\mathcal{A} = \sum_{m=-\mu-p/2}^{\mu+p/2} \omega'_{\mu,m} E^m$. Note that $\omega'_{\mu,m} = \omega'_m$ for $|m| \leq \mu - p/2$. Details on computing coefficients for $\mathcal{A}$ are provided in Appendix C.

Note that (S)PME[1] does not perform exact interpolation; rather, it interpolates after applying a low-pass filter (i.e., truncating a Fourier series).

5. Quasi-interpolation for convolution kernels in three dimensions

Interpolation from values on a 3-dimensional grid with spacing h can be represented as a linear combination of basis functions $\varphi_{\mathbf{m}}(\mathbf{r}) = \varphi_{m_x}(r_x)\varphi_{m_y}(r_y)\varphi_{m_z}(r_z)$, where $\mathbf{m}$ indexes the points of the grid. Let $\tilde{F}(\mathbf{r},\mathbf{r}')$ be a spline (quasi-)interpolant of the kernel $F(\mathbf{r},\mathbf{r}') = f(\mathbf{r}-\mathbf{r}')$ at points $(\mathbf{r},\mathbf{r}') = (h\mathbf{m}, h\mathbf{n})$ for all integer vectors $\mathbf{m}$, $\mathbf{n}$. Extending Eq. (13) to 3 dimensions gives

$$\tilde{F}(\mathbf{r},\mathbf{r}') = \sum_{\mathbf{m}} \sum_{\mathbf{n}} (\mathcal{A}_x \mathcal{A}_y \mathcal{A}_z f^h)_{\mathbf{m}-\mathbf{n}} \varphi_{\mathbf{m}}(\mathbf{r}) \varphi_{\mathbf{n}}(\mathbf{r}'), \tag{14}$$

where $(f^h)_{\mathbf{k}} = f(h\mathbf{k})$.

C. Kernel approximations

Consider now the construction of the approximation given in Eq. (2). This employs nested grids indexed from 1 through L with the lth grid having grid size $h_l = 2^{l-1}h$ and basis functions $\varphi^l_{\mathbf{m}}(\mathbf{r}) = \varphi^l_{m_x}(r_x)\varphi^l_{m_y}(r_y)\varphi^l_{m_z}(r_z)$ where $\varphi^l_n(x) = \Phi(x/h_l - n)$. In this section, the grids are taken to be of infinite extent; Section III A determines actual finite limits.

Let $\tilde{k}_l(\mathbf{r},\mathbf{r}')$ be a spline (quasi-)interpolant of the kernel $k_l(\mathbf{r},\mathbf{r}') = \kappa_l(\mathbf{r}-\mathbf{r}') = g_l(|\mathbf{r}-\mathbf{r}'|)$ at points $(\mathbf{r},\mathbf{r}') = (h_l\mathbf{m}, h_l\mathbf{n})$ for all integer vectors $\mathbf{m}$, $\mathbf{n}$. Applying Eq. (14) gives

$$\tilde{k}_l(\mathbf{r},\mathbf{r}') = \sum_{\mathbf{m}} \sum_{\mathbf{n}} \hat{k}^l_{\mathbf{m},\mathbf{n}} \varphi^l_{\mathbf{m}}(\mathbf{r}) \varphi^l_{\mathbf{n}}(\mathbf{r}'), \tag{15}$$

where

$$\hat{k}^l_{\mathbf{m},\mathbf{n}} = (\mathcal{A}_x \mathcal{A}_y \mathcal{A}_z \kappa^l)_{\mathbf{m}-\mathbf{n}} \quad \text{and} \quad \kappa^l_{\mathbf{k}} = \kappa_l(h_l\mathbf{k}).$$

Note that

$$\kappa^l_{\mathbf{k}} = \kappa_l(h_l\mathbf{k}) = g_l(|h_l\mathbf{k}|) = \frac{1}{a_l}\gamma_l\left(\frac{h_l}{a_l}|\mathbf{k}|\right) = \frac{2^{-l}}{a}\gamma_l\left(\frac{h}{2a}|\mathbf{k}|\right) = \frac{2^{-l}}{a}\gamma^l_{\mathbf{k}},$$

where

$$\gamma^l_{\mathbf{k}} = \gamma_l\left(\frac{h}{2a}|\mathbf{k}|\right). \tag{16}$$

Hence,

$$\hat{k}^l_{\mathbf{m},\mathbf{n}} = \frac{2^{-l}}{a} K^l_{\mathbf{m}-\mathbf{n}},$$

where

$$K^l = \mathcal{A}_x \mathcal{A}_y \mathcal{A}_z \gamma^l. \tag{17}$$

This enables one to write Eq. (15) as

$$\tilde{k}_l(\mathbf{r},\mathbf{r}') = \sum_{\mathbf{m}} \sum_{\mathbf{n}} \frac{2^{-l}}{a} K^l_{\mathbf{m}-\mathbf{n}} \varphi^l_{\mathbf{m}}(\mathbf{r}) \varphi^l_{\mathbf{n}}(\mathbf{r}'). \tag{18}$$

If the spline degree $p-1$ exceeds 1, the spline interpolants $\tilde{k}_l(\mathbf{r},\mathbf{r}')$, $l = 1, 2, \ldots, L-1$, have (generally) nonzero coefficients $K^l_{\mathbf{m}-\mathbf{n}}$ for the entire domain even though the range of k_l is limited. Nonzero values of $\tilde{k}_l(\mathbf{r},\mathbf{r}')$ beyond the range are purely interpolation error. So, for example, one might include only those basis functions $\varphi^l_{\mathbf{m}}(\mathbf{r})\varphi^l_{\mathbf{n}}(\mathbf{r}')$ whose support intersects that of the (finite-range) kernel $k_l(\mathbf{r},\mathbf{r}')$. Because the kernel has spherical support and the basis functions have cubic support, it can be shown that the multi-index difference $\mathbf{m}-\mathbf{n}$ would be included only if $h_l\mathbf{m} - h_l\mathbf{n} = \mathbf{r}_c + \mathbf{r}_s$ for some $|\mathbf{r}_c|_\infty < ph$ and $|\mathbf{r}_s| < a_l$, where $|\cdot|_\infty$ denotes the maximum norm for vectors. Then the difference $h_l\mathbf{m} - h_l\mathbf{n}$ would lie in a region that is cubic but with rounded edges and corners. However, including all these terms would not only be a bit complicated but also costly (compared to C^1 interpolation). Moreover, the more distant pairs of basis functions make very small contributions. As a compromise, in Eq. (18), we use only those $K^l_{\mathbf{m}-\mathbf{n}}$ for which $|h_l\mathbf{m} - h_l\mathbf{n}|_\infty < a_l$. The resulting cubic stencil is still more costly than the spherical stencil used by C^1 interpolation. However, timings for the large water sphere of Section II D 2 show only a small increase in setup time for a cubic rather than a spherical stencil and no increase in the time to do an energy/force calculation. With this choice, the approximation becomes

$$k_l(\mathbf{r},\mathbf{r}') \approx \sum_{\mathbf{m}} \sum_{\mathbf{n}} \varphi^l_{\mathbf{m}}(\mathbf{r}) \mathcal{K}^l_{\mathbf{m},\mathbf{n}} \varphi^l_{\mathbf{n}}(\mathbf{r}'), \tag{19}$$

where

$$\mathcal{K}^l_{\mathbf{m},\mathbf{n}} = \begin{cases} (2^{-l}/a) K^l_{\mathbf{m}-\mathbf{n}}, & l = L \text{ or } |\mathbf{m}-\mathbf{n}|_\infty < 2\alpha, \\ 0, & \text{otherwise}, \end{cases} \tag{20}$$

where

$$\alpha = a/h.$$

In conclusion, one approximates $k_l(\mathbf{r},\mathbf{r}')$ as given by Eq. (19) and substitutes into Eq. (5) to obtain the approximation to $k(\mathbf{r},\mathbf{r}')$ given by Eq. (2).

1. Preprocessing

The grid stencils $K_{\mathbf{k}}^l$ can be precomputed using Eqs. (17) and (16). For $l < L$, as indicated in Eq. (20), values of $K_{\mathbf{k}}^l$ are needed only for $|\mathbf{k}|_\infty < 2\alpha$, and values of $\gamma_{\mathbf{k}}^l$ are nonzero only for $|\mathbf{k}| < 2\alpha$. Hence, to compute these $K_{\mathbf{k}}^l$, values of $\omega_{\mathbf{k}}'$ are needed only for $|\mathbf{k}|_\infty < 4\alpha$, so using $\mu \geq 4\alpha + p/2$ suffices for exact interpolation. Moreover, for levels $l < L$, $\gamma_{\mathbf{k}}^l$ and therefore $K_{\mathbf{k}}^l$ are independent of l.

For level L, the range of $\mathbf{k}$ depends on the range of the particles. Assume there is a region Ω, e.g., a rectangular box or an ellipsoid, known to contain all particles for every evaluation of the energy and forces. With such a bound, values of $K_{\mathbf{m}-\mathbf{n}}^L$ are needed only for $\mathbf{m}, \mathbf{n} \in \mathcal{M}_L$ where

$$\mathcal{M}_l = \{\mathbf{m} : \varphi_{\mathbf{m}}^l(\mathbf{r}) \neq 0 \text{ for some } \mathbf{r} \in \Omega\}. \tag{21}$$

Experimental evidence (not shown) suggests using $\mu \geq 3p/2$ for computing $K_{\mathbf{k}}^L$.

The values of the stencils $K_{\mathbf{k}}^l$ are invariant under permutations of the multi-index $\mathbf{k}$ and under reflections in the direction of each of the 3 axes, resulting in a 48-fold symmetry, with a commensurate reduction in computational cost.

D. Accuracy

There is, in the thesis,[24] a rigorous error analysis for multilevel approximation using C^1 piecewise polynomials, which provides error bounds in terms of the fundamental method parameters. An important result (Ref. 24, Eq. (3.47)) that transfers to B-splines is that, for the energy, each level contributes an error roughly half that of the previous level. For the force, the factor is one quarter.

The first part of this section compares theoretically the accuracy of different types of piecewise polynomial interpolants. A more complete error analysis, though desirable, is beyond the scope of the present study. The remainder of this section presents results of numerical experiments.

1. Theoretical evidence

A simple examination of results from error analysis for 1 dimension indicates that the accuracy of B-spline interpolation

1. is comparable to centered C^0 interpolation, (used in Ref. 27) and
2. superior to Taylor interpolation (used by the fast multipole method) or Hermite interpolation (investigated in Ref. 24)

for the same computational effort, if the computational effort of evaluating a derivative is assumed to be the same as that for the function. However, the second observation does not necessarily transfer to a six-dimensional interpolation of the kernel.

A routine calculation using classical results shows that centered C^0 piecewise polynomial interpolation of degree $p-1$ has the error bound

$$\frac{1 \cdot 3 \cdot \cdots \cdot (p-1)}{2 \cdot 4 \cdot \cdots \cdot p} \left(\frac{h}{2}\right)^p \|f^{(p)}\|_\infty.$$

For $(p/2)$-fold piecewise Hermite interpolation and for piecewise Taylor interpolation, a good error bound for spacing h is

$$\frac{1}{p!}\left(\frac{h}{2}\right)^p \|f^{(p)}\|_\infty.$$

For B-spline interpolation, M. Reimer[29] (Eqs. (12), (15), (20)) provides the error bound

$$\left(\frac{1 \cdot 3 \cdot \cdots \cdot (p-1)}{2 \cdot 4 \cdot \cdots \cdot p} + \|\mathcal{L}\|\right)\left(\frac{h}{2}\right)^p \|f^{(p)}\|_\infty,$$

where $\mathcal{L}$ is the linear operator mapping a function f to its interpolation error $\tilde{f} - f$. Asymptotically,[30,31]

$$\|\mathcal{L}\| = \frac{2}{\pi}\left(\log p + 2\log\frac{4}{\pi} + \gamma\right) + O\left(\frac{1}{p}\right) \quad \text{as } p \to \infty,$$

where γ is the Euler-Mascheroni constant.

Let h_{eff} denote the "effective" spacing: For 2-point piecewise Hermite interpolation of derivatives of order 0 through $(p/2)-1$, choose spacing $h = (p/2)h_{\text{eff}}$ to keep the work the same, and for piecewise Taylor interpolation, choose spacing $h = ph_{\text{eff}}$ to keep the work the same. Otherwise, take $h = h_{\text{eff}}$. Using Stirling's formula, the error term is $(C_p h_{\text{eff}}/2)^p \|f^{(p)}\|_\infty$ where

$$C_p = \begin{cases} 1 - \dfrac{1}{2p}\log p + O\left(\dfrac{1}{p}\right), & \text{centered,} \\ 1 + \left(\dfrac{4}{\pi} - 1\right)\dfrac{1}{2p}\log p + O\left(\dfrac{1}{p}\right), & \text{B-spline,} \\ \dfrac{1}{2}\mathrm{e}\left(1 - \dfrac{1}{2p}\log p\right) + O\left(\dfrac{1}{p}\right), & \text{Hermite,} \\ \mathrm{e}\left(1 - \dfrac{1}{2p}\log p\right) + O\left(\dfrac{1}{p}\right), & \text{Taylor.} \end{cases}$$

2. Empirical evidence

Results of numerical experiments are presented that compare the accuracy of B-spline to C^1 piecewise polynomial basis functions for various cutoffs a and that examine their order of accuracy. The experiments use C^{p-2} and C^{p-1} softening functions with the C^1 piecewise polynomials and B-splines, respectively.

Specifically, these computations determine the accuracy of forces for an equilibrated sphere of 10 002 water molecules with radius 42 Å. They calculate error in mass-weighted long-range forces relative to the exact long-range forces, where mass-weighted-norm $\sqrt{m_i^{-1}|\mathbf{F}_i^{1+}|^2}$ is applied to each long-range force $\mathbf{F}_i^{1+}$. Dividing by mass gives mass-weighted acceleration, which has the effect of ascribing greater importance to positions of heavy atoms compared to hydrogen. (It is common to use mass-weighted coordinates $\bar{x} = M^{1/2}x$ in computing the RMSD between two structures.[32]) Because interpolation is applied to only the softened kernel $k_{1+} = k_1 + \cdots + k_l$, the calculations compare errors relative to forces arising from the softened kernel, using $k_{1+}(\mathbf{r}, \mathbf{r}')$ in place of $k(\mathbf{r}, \mathbf{r}')$ for energy and forces. A good value for the grid spacing h is 2.5 Å, for which there are approximately N grid points at the finest level (see Section III B 1).

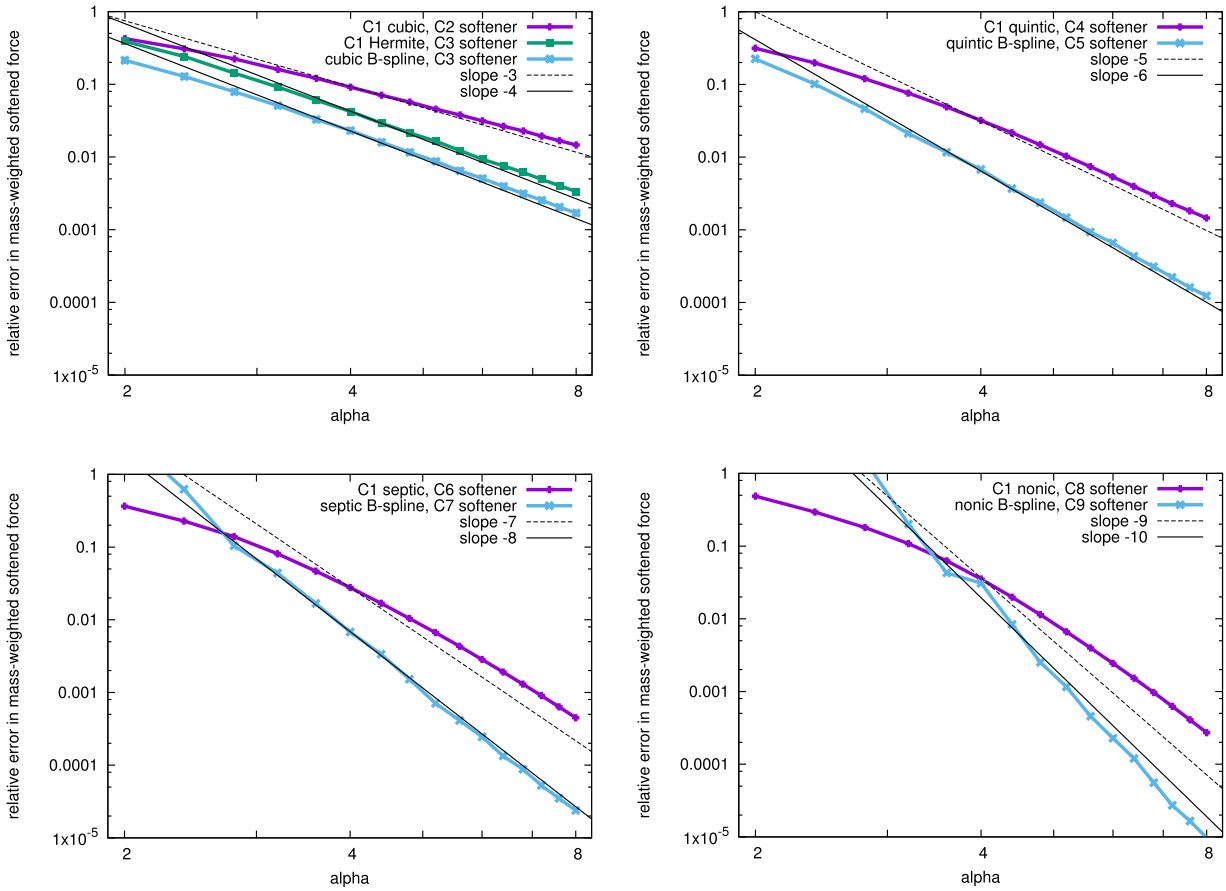

FIG. 3. Comparison between B-spline and C^1 piecewise polynomial basis functions for relative error in mass-weighted force vs. scaled cutoff $\alpha = a/h$. Results are given for piecewise polynomials of degrees 3, 5, 7, and 9.

Figure 3 exhibits the accuracy of the *long-range parts* of forces for B-spline and C^1 basis functions for a range of $\alpha = a/h$, for each of $p = 4, 6, 8, 10$. The cutoff a ranges through values $5, 6, \ldots, 20$ Å, i.e., $2 \le a/h \le 8$. One sees for $p \le 8$ that the B-spline interpolants are an order of magnitude more accurate for equal computational effort. For $p = 10$, accuracy is significantly improved by extending the "radius" of the grid-to-grid stencils $K^l_{\mathbf{k}}$ to $|\mathbf{k}|_\infty < 2\alpha + 1$. Also plotted on top of each graph of numerical data is a straight line indicating the inferred theoretical slope as $\alpha \to \infty$. Two features of these slopes require explanation. (i) For basis functions of degree $p - 1$, it is observed that the order is $p - 1$ for C^1 piecewise polynomials and p for B-splines. These orders of accuracy in the gradients are one greater than what is expected from typical theoretical considerations.[24] This can be explained by the fact that interpolation error vanishes at grid points, and as a consequence, the error in the first derivative changes sign within each grid cell. A sum of many such errors accumulates slowly due to cancellation. (ii) There is a gradually steepening of the slope (the observed order of accuracy) of the graph for the C^1 interpolant, which is expected. However, the opposite is happening for the B-splines. The initial excess error (which is greater for larger p) for B-splines is a consequence of the truncation of the stencil specified by Eq. (20). This conclusion is confirmed by increasing the width of the stencil by 50%.

E. Nesting of spline function spaces

A useful property is that of *nested interpolation*, in which the basis functions of a coarser grid are interpolated exactly at the next finer level. This holds for B-spline interpolation but not for C^1 piecewise polynomial interpolation.

A coarse-grid B-spline can be expressed in terms of fine-grid B-splines using the *two-scale relation*

$$\Phi(u) = \sum_{n=-p/2}^{p/2} J_n \Phi(2u - n),$$

where

$$J_n = 2^{1-p} \binom{p}{(p/2) + |n|}. \tag{22}$$

From this relation, one gets $\varphi^l_m(x) = \sum_n J_n \varphi^{l-1}_{n+2m}(x) = \sum_n J_{n-2m} \varphi^{l-1}_n(x)$. (Note that $J_n = 0$ for $|n| > p/2$.) In three dimensions, this becomes

$$\varphi^l_{\mathbf{m}}(\mathbf{r}) = \sum_{\mathbf{n}} J_{\mathbf{n}-2\mathbf{m}} \varphi^{l-1}_{\mathbf{n}}(\mathbf{r}),$$

where

$$J_{\mathbf{n}} = J_{n_x} J_{n_y} J_{n_x}. \tag{23}$$

Plotted in Figure 4(a) is a coarse-grid cubic B-spline and the 5 fine-grid cubic B-splines that sum up to it.

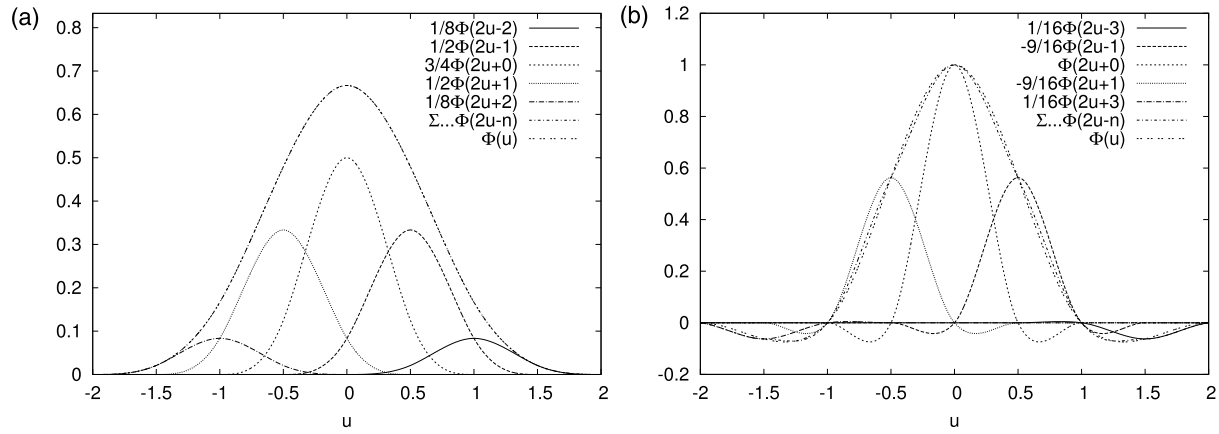

FIG. 4. (a) A coarse-grid cubic B-spline and 5 fine-grid cubic B-splines that sum up to it. (b) A coarse-grid C^1 piecewise cubic and 5 fine-grid C^1 piecewise cubics that interpolate it, but do not sum up exactly.

The C^1 piecewise polynomials used previously[13,24] *do not span* the space of C^1 piecewise polynomials with equidistant knots. For example, the C^1 piecewise cubic $g(u)$ with knots at the integers and values $g(0) = 1$, $g'(0) = 0$ and $g(u) \equiv 0$ for $|u| \geq 1$ cannot be expressed as a linear combination of the C^1 piecewise cubic nodal basis functions. Moreover, nesting does not hold exactly for C^1 (nor C^0) piecewise polynomials. Consider the case of piecewise cubics: A $2h$-grid C^1 piecewise cubic nodal basis function has support $[-4h, 4h]$. It cannot be exactly duplicated by h-grid basis functions at $-2h$, $-h$, 0, h, $2h$. It can be interpolated—with some error—by h-grid basis functions at $-3h$, $-h$, 0, h, $3h$, whose combined support is now $[-5h, 5h]$. To accommodate the expansion in support, C^1 piecewise polynomials require an extension of an *additional* $(p/2) - 1$ grid points (Ref. 24, Section 6.1.1). Plotted in Figure 4(b) is a coarse-grid C^1 piecewise cubic and 5 fine-grid C^1 piecewise cubics that interpolate it, but do not sum up exactly.

III. ALGORITHM AND PERFORMANCE

A. The algorithm

Eqs. (1) and (2) approximate the electrostatic energy as $U(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N) \approx U^0 + U^{1+}$ where

$$U^0 = \frac{1}{2}\sum_i \sum_{j \notin \chi(i)} q_i q_j k_0(\mathbf{r}_i, \mathbf{r}_j) - \frac{1}{2}\sum_i \sum_{j \in \chi(i)} q_i q_j k_{1+}(\mathbf{r}_i, \mathbf{r}_j)$$

and

$$U^{1+} = \frac{1}{2}\sum_i \sum_j q_i q_j \tilde{k}_{1+}(\mathbf{r}_i, \mathbf{r}_j). \tag{24}$$

Substituting Eq. (2) into Eq. (24) gives

$$U^{1+} = \frac{1}{2}\sum_i q_i \sum_l \sum_{\mathbf{m}} \varphi^l_{\mathbf{m}}(\mathbf{r}_i) \sum_{\mathbf{n}} \mathcal{K}^l_{\mathbf{m},\mathbf{n}} \sum_j \varphi^l_{\mathbf{n}}(\mathbf{r}_j) q_j.$$

Taking the negative gradient with respect to $\mathbf{r}_i$, gives the ith long-range force

$$\mathbf{F}^{1+}_i = -q_i \sum_l \sum_{\mathbf{m}} \nabla\varphi^l_{\mathbf{m}}(\mathbf{r}_i) \sum_{\mathbf{n}} \mathcal{K}^l_{\mathbf{m},\mathbf{n}} \sum_j \varphi^l_{\mathbf{n}}(\mathbf{r}_j) q_j.$$

Letting

$$q^l_{\mathbf{n}} = \sum_j \varphi^l_{\mathbf{n}}(\mathbf{r}_j) q_j \tag{25}$$

and

$$E(\mathbf{r}) = \sum_l \sum_{\mathbf{m}} \varphi^l_{\mathbf{m}}(\mathbf{r}) e^l_{\mathbf{m}}, \tag{26}$$

where

$$e^l_{\mathbf{m}} = \sum_{\mathbf{n}} \mathcal{K}^l_{\mathbf{m},\mathbf{n}} q^l_{\mathbf{n}}$$

gives

$$U^{1+} = \frac{1}{2}\sum_i q_i E(\mathbf{r}_i) \quad \text{and} \quad \mathbf{F}^{1+}_i = -q_i \nabla E(\mathbf{r}_i). \tag{27}$$

An algorithm for computing the energy and forces from these expressions is presented and derived in a high level form and later described in greater detail.

1. Overview of algorithm

The structure of the algorithm is represented by Figure 5. Different levels correspond to different sets of points, particle positions at the lowest level and grid points at higher levels. Small circles on the left represent charges and small circles on the right represent electric potentials that accumulate as one

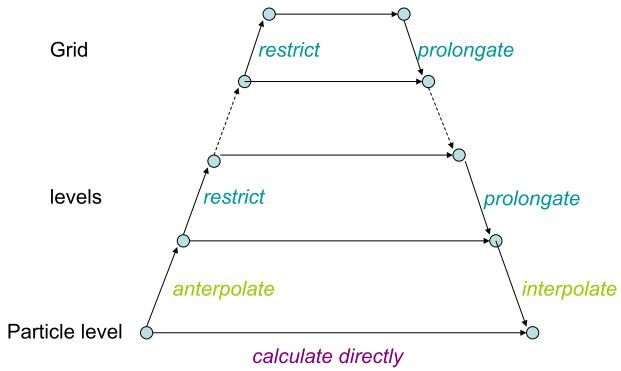

FIG. 5. Diagram of algorithmic steps.

descends. Arrows are linear operators, or matrices, that map one set of values to another. The confluence of two arrows indicates addition of a longer-range contribution to a given grid level or to the particle level. There are six different types of linear operators. Three of them depend on particle positions, for which the matrix elements are computed as needed rather than stored: short-range interactions, interpolation, anterpolation. The other three types of linear operators are convolutions whose stencils can be precomputed: grid interactions, prolongation, and restriction.

The various steps are the following:

1. *Short-range interactions*. Compute U^0 and its gradients.
2. *Anterpolation*. Compute level-1 charges q^1 using
$$q^1_{\mathbf{n}} = \sum_i \varphi^1_{\mathbf{n}}(\mathbf{r}_i) q_i. \tag{28}$$
This mapping of particle charges to charges at grid points is *not* interpolation but rather the adjoint of interpolation. It is not the charges that are being interpolated, but their long-range effect (via interpolation of the interaction kernel).
3. *Restriction*. Compute higher level charges using Eq. (4),
$$q^l = \mathcal{I}^l_{l-1} q^{l-1}, \quad l = 2,3,\ldots,L, \tag{29}$$
where the restriction operator $\mathcal{I}^l_{l-1}$ is defined by
$$(\mathcal{I}^l_{l-1})_{\mathbf{m},\mathbf{n}} = J_{\mathbf{n}-2\mathbf{m}}. \tag{30}$$
4. *Grid to grid mapping* and *prolongation*. Compute accumulated electric potentials for each grid level using the recurrence
$$e^{L+} = \mathcal{K}^L q^L, \tag{31}$$
$$e^{l+} = \mathcal{K}^l q^l + \mathcal{I}^l_{l+1} e^{(l+1)+}, \quad l = L-1, L-2, \ldots, 1, \tag{32}$$
where the prolongation operator $\mathcal{I}^l_{l+1} = (\mathcal{I}^{l+1}_l)^{\mathsf{T}}$.
5. *Interpolation*. Interpolating grid values yields the electric potential
$$E(\mathbf{r}) = \sum_{\mathbf{m}} \varphi^1_{\mathbf{m}}(\mathbf{r}) e^{1+}_{\mathbf{m}}, \tag{33}$$
which is used in formulas (27) for computing energy and forces.

The matrices representing these linear operators are all sparse, except for the operator $\mathcal{K}^L$ representing interactions between all pairs of grid points on the top level grid. The matrices for $\mathcal{K}^l$, $1 \le l \le L-1$, each have $(2\lfloor 2a/h \rfloor + 1)^3$ nonzero elements per row. The matrix that represents interactions k_0 at the particle level has about $\frac{4}{3}\pi(a/h_*)^3$ nonzero elements per row, assuming a particle density of h_*^{-3}. The matrices for I^{l+1}_l, $1 \le l \le L-1$, each have no more than $(p+1)^3$ nonzero elements along each row and column. The matrices implicit in Eqs. (33)/(27), representing interpolation, each have no more than $(p+1)^3$ nonzero elements along each row. Pseudocode for these matrix–vector multiplications is given in Ref. 24, Section 2.3.

a. Exact treatment of exclusions. The foregoing algorithm interpolates the long-range part of excluded interactions, thereby introducing interpolation error $\tilde{k}_{1+}(\mathbf{r}_i,\mathbf{r}_j) - k_{1+}(\mathbf{r}_i,\mathbf{r}_j)$ for nonexistent terms, $j \in \chi(i)$. The algorithm can be augmented to remove the interpolation error of these excluded interactions by doing an additional $O(p^3 NL) = O(p^3 N \log N)$ operations. However, two independent implementations of this augmented algorithm yield results for molecular systems that are less accurate. Presumably, there is a fortuitous cancellation of errors of excluded terms with those of included terms. The numerical experiments performed in this study do not correct the interpolation error due to excluded interactions.

b. Derivation of grid to grid operations. Given here are derivations for Eqs. (29)–(33). To show Eq. (29), use Eq. (30) to write the two-scale relation (23) as
$$\varphi^l_{\mathbf{m}}(\mathbf{r}) = \sum_{\mathbf{n}} (\mathcal{I}^l_{l-1})_{\mathbf{m},\mathbf{n}} \varphi^{l-1}_{\mathbf{n}}(\mathbf{r}). \tag{34}$$
Application of this to Eq. (25) yields the recurrence (4). The next step is to exploit the nesting property to express $E(\mathbf{r})$ in terms of the level 1 basis functions. Applying the two-scale relation to the last two terms of $E(\mathbf{r})$ given by Eq. (26) yields
$$\begin{aligned}\sum_{\mathbf{m}} \varphi^{L-1}_{\mathbf{m}}(\mathbf{r}) e^{L-1}_{\mathbf{m}} + \sum_{\mathbf{m}} \varphi^L_{\mathbf{m}}(\mathbf{r}) e^L_{\mathbf{m}} &= \sum_{\mathbf{m}} \varphi^{L-1}_{\mathbf{m}}(\mathbf{r}) \left(e^{L-1} + \mathcal{I}^{L-1}_L e^L\right)_{\mathbf{m}} \\ &= \sum_{\mathbf{m}} \varphi^{L-1}_{\mathbf{m}}(\mathbf{r}) e^{(L-1)+}_{\mathbf{m}},\end{aligned} \tag{35}$$
where $e^{(L-1)+} = e^{L-1} + \mathcal{I}^{L-1}_L e^L$. Repeated use of this transformation yields Eq. (33) with e^{1+} obtained via the recurrence
$$e^{L+} = e^L, \qquad e^{l+} = e^l + \mathcal{I}^l_{l+1} e^{(l+1)+}, \quad l = L-1, L-2, \ldots, 1.$$

2. Detailed version of algorithm

At the beginning of the algorithm, determine the index sets $\mathcal{M}_l$ defined by Eq. (21).

a. Short-range interactions. The short-range calculation involves looping over all pairs of particles within a given distance a of each other. Avoiding pairs whose separation distance is much beyond the cutoff a is achieved by the spatial hashing of atoms into bins and considering for each atom just those atoms in the surrounding bins.

b. Anterpolation. The anterpolation step loops over all particles, spreading the charge of each particle onto p^3 surrounding grid points. It computes level 1 charges $q^1_{\mathbf{n}}$, $\mathbf{n} \in \mathcal{M}_1$, using Eq. (28): for each particle i, it adds nonzero $\varphi^1_{\mathbf{n}}(\mathbf{r}_i) q_i$ to $q^1_{\mathbf{n}}$.

The evaluation of the B-spline basis functions is required for both this step and the interpolation step, with the latter also requiring a gradient evaluation. For each particle i, one expresses $\mathbf{r}_i = (\mathbf{m} + \mathbf{u}) h_1$ where $\mathbf{m}$ is an integer triple and $0 \le u_\kappa < 1$ for $\kappa = \mathrm{x,y,z}$. Values of $\varphi^1_{\mathbf{m}+\mathbf{k}}(\mathbf{r}_i)$ and $\nabla\varphi^1_{\mathbf{m}+\mathbf{k}}(\mathbf{r}_i)$ are nonzero only for $1 - p/2 \le |\mathbf{k}|_\infty \le p/2$. These values are obtained from
$$\begin{aligned}\varphi^1_{m_\kappa + k}(\mathbf{r}_i) = \Phi(u_\kappa - k) = Q_p(u_\kappa + (p/2) - k), \\ k = 1 - p/2, 2 - p/2, \ldots, p/2,\end{aligned}$$

and its first derivative. To get B-spline values and first derivatives, apply Horner's rule to each piece of the B-spline.

c. Restriction. The restriction step loops over the coarser of two consecutive grids, collecting charge for each point on the coarser grid from nearby points on the finer grid using a fixed stencil of coefficients. Combining Eqs. (29) and (30) gives the following: for $l = 2,3,\ldots,L$, compute higher level grid charges using

$$q_{\mathbf{m}}^{l} = \sum_{\mathbf{n}} J_{\mathbf{n}-2\mathbf{m}} q_{\mathbf{n}}^{l-1}, \quad \text{for } \mathbf{m} \in \mathcal{M}_l,$$

where the sum is over all $\mathbf{n} \in \mathcal{M}_{l-1}$ such that $-p/2 \le |\mathbf{n} - 2\mathbf{m}|_\infty \le p/2$ and where $J_{\mathbf{k}}$ is defined in Eq. (22). The computation is linear in the order p if the collecting is done in one coordinate direction at a time. The calculation involves the use of two intermediate grids with grid spacings that are doubled in one or two directions. The algorithm given in Ref. 24, Table 2.10 shows how to reduce the required intermediate buffer space to just $O(N^{2/3})$.

d. Grid to grid mapping and prolongation. Combining Eqs. (31) and (20) gives the following: compute electric potentials for the top grid level

$$e_{\mathbf{m}}^{L+} = \frac{2^{-L}}{a} \sum_{\mathbf{n}} K_{\mathbf{m}-\mathbf{n}}^{L} q_{\mathbf{n}}^{L}, \quad \mathbf{m} \in \mathcal{M}_L,$$

where the sum is over all $\mathbf{n} \in \mathcal{M}_L$. Combining Eqs. (32) and (20)/(30) gives the following: compute accumulated electric potentials for each lower grid level $l = L-1$, $L-2,\ldots,1$ using the recurrence

$$e_{\mathbf{m}}^{l+} = \frac{2^{-l}}{a} \sum_{\mathbf{n}} K_{\mathbf{m}-\mathbf{n}}^{l} q_{\mathbf{n}}^{l} + \sum_{\mathbf{n}} J_{\mathbf{n}-2\mathbf{m}} e_{\mathbf{n}}^{(l+1)+}, \quad \mathbf{m} \in \mathcal{M}_l.$$

The first sum is over all $\mathbf{n} \in \mathcal{M}_l$ such that $|\mathbf{m} - \mathbf{n}|_\infty < 2\alpha$. The second sum is the prolongation step, which loops over the coarser grid, distributing electric potential from each grid point to nearby points on the fine grid according to some fixed stencil. Specifically, the outer loop is over all $\mathbf{n} \in \mathcal{M}_{l+1}$ such that $-p/2 \le |\mathbf{n} - 2\mathbf{m}|_\infty \le p/2$. Just as for restriction, it is linear in the order p.

e. Interpolation. The interpolation step loops over all particles, and for each particle interpolates the electric potential and electric field from nearby grid points. Specifically, for each particle i, add $\frac{1}{2} q_i \sum_{\mathbf{m}} \varphi_{\mathbf{m}}^{1}(\mathbf{r}_i) e_{\mathbf{m}}^{1+}$ to the energy U, and add $-q_i \sum_{\mathbf{m}} \nabla\varphi_{\mathbf{m}}^{1}(\mathbf{r}_i) e_{\mathbf{m}}^{1+}$ to the ith force F_i. This requires evaluation of p B-spline basis functions along each dimension.

B. Performance analysis

1. Choosing optimal grid size

Ref. 13 analyzes the effect of grid size h and cutoff a on computational cost for a desired error tolerance and a specified order p. The error and cost depend on the ratios h/h_* and $h/a = 1/\alpha$ where $h_* = (\text{volume}(\Omega)/N)^{1/3}$, which is a measure of average distance between nearest neighbors. It is shown that the optimal ratio h/h_* is a value near 1 that is practically independent of the desired accuracy, so it is the ratio h/a that should be varied to control accuracy. This conclusion is confirmed in Ref. 24 with the benefit of the more detailed cost analysis.

To keep the operation count linear in N, it is enough to choose the number of levels L just large enough that the number of grid points at level L does not exceed $\sqrt{N}$. At the same time, it is disadvantageous to reduce the number of grid points below $(2\alpha)^3$, because this would incur a greater operation count than choosing L to be one less.

2. Choosing optimal degree and cutoff

The optimal degree for spline interpolation is determined empirically, using the sphere of 10 002 water molecules from Section II D. CPU time and relative mass-weighted root-mean-square error in the forces $((\sum_{i=1}^{N} m_i^{-1}|\tilde{\mathbf{F}}_i - \mathbf{F}_i|^2)/(\sum_{i=1}^{N} m_i^{-1}|\mathbf{F}_i|^2))^{1/2}$ are computed for orders $p = 4,6,8,10$ and relative cutoffs $\alpha = 2.4, 2.8, \ldots, 8$ (cutoff distances 6 Å $\le a \le$ 20 Å with grid spacing 2.5 Å) and C^{p-1} softener. Results are shown in Figure 6. Based on these results, the following formula is constructed for choosing p for a given α:

$$p = \text{the element of } \{4,6,8\} \text{ nearest } 1.25\alpha + 0.25, \quad (36)$$

i.e., p is chosen to be the middle value 6 for $3.8 < \alpha < 5.4$ and otherwise 4 or 8.

A representative accuracy is, say, a 0.5% error in forces. The effect of a time step of 1 fs is to distort highest frequencies by 1.6%.

3. Comparison to FMM

The efficiency of a C++ implementation of the MSM is compared with a very recent C implementation[26] of the Uniform FMM Laplace Solver of the FMM-Laplace library.[33] (The FMM-Laplace library is built with Intel Cilk disabled to compel execution on a single CPU core.) The timing includes construction of the DAG (directed acyclic graph) as well

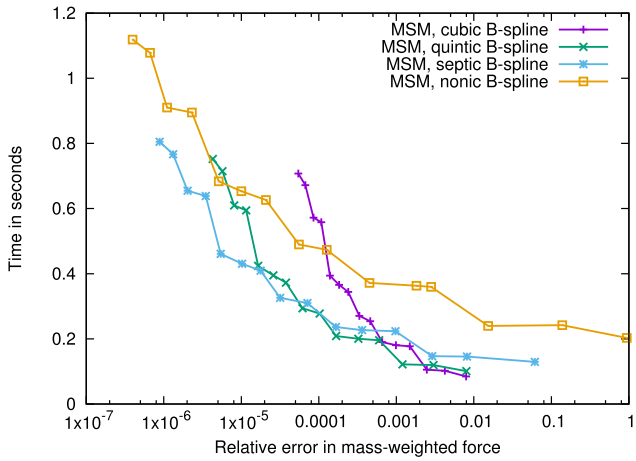

FIG. 6. The MSM performance comparison for B-spline interpolation of degree 3, 5, 7, 9. Each line plot varies the cutoff distance $6,7,\ldots,20$ Å with grid spacing 2.5 Å.

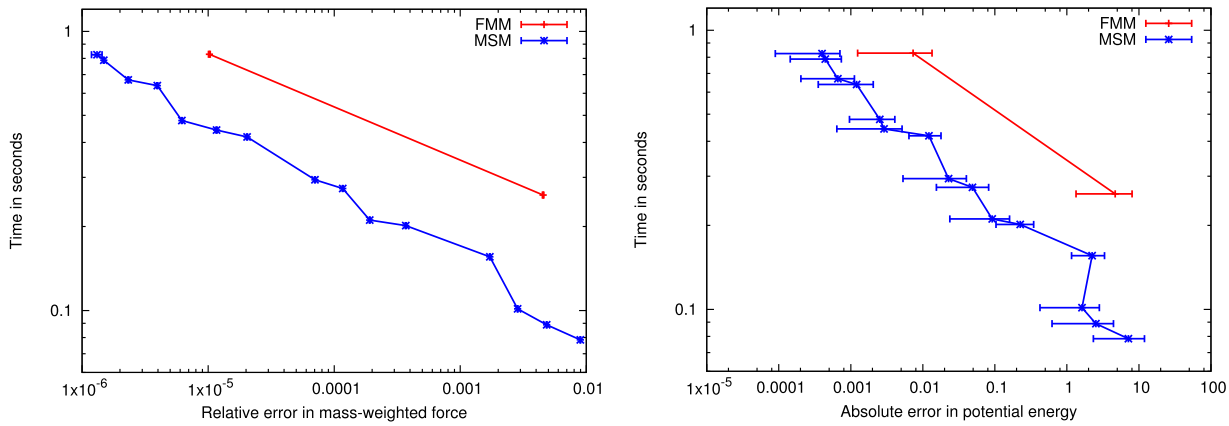

FIG. 7. Comparing efficiency of the MSM with the FMM. The MSM varies the cutoff distance $6, 7, 8, \ldots, 20$ Å with grid spacing 2.5 Å and B-spline order determined from formula (36). The FMM varies the force accuracy from 3 to 6 digits. The left plot shows the relative error in force, and the right plot shows the absolute error in total potential energy. Each point is the average over 100 separated time steps, with error bars showing the standard deviation.

as scaling the position input and force output, which would be part of the solver if used within an MD simulator. Both codes are compiled using the most recent version of the Intel C++ compiler (version 16.0.0) with flags to enable compiler optimization using the AVX2 (Advanced Vector Extensions 2) SIMD (single instruction, multiple data) instruction set. The MSM implementation arranges the grid points into 8-element clusters, so as to be able to make explicit use of the AVX2 instructions, benefiting especially from the supported FMA (fused multiply–add) instructions.[4] Testing was performed using a single core of an Intel Haswell processor (Xeon E5-2680 v3). The grid to grid, restriction, and prolongation calculations are performed in single precision. This is never a problem because the optimal way to do calculations of higher accuracy is to increase the cutoff distance,[13,24] thereby increasing the overall contribution of the short-range part while decreasing the contribution of the long-range part, which is the part being computed in single precision.

The test system is an equilibrated sphere of 10 002 water molecules (30 006 atoms) with radius 42 Å. For MSM, the finest grid spacing is fixed at $h = 2.5$ Å. The calculation and timings are repeated 10 times per data point to suitably "warm up" the hardware and compensate for any other background activities that the workstation might incidentally perform during testing, and the minimum time is reported.

The graph in Figure 7 shows, on the vertical axis, CPU time in seconds plotted against the relative mass-weighted root-mean-square error in the forces and the absolute error in the total potential energy. The MSM lines in the graphs are generated by varying the short-range cutoff distance, where the points correspond to relative cutoffs $\alpha = 2.4, 2.8, 3.2, \ldots, 8$ ($a = 6, 7, 8, \ldots, 20$ Å), with the spline degree determined from the relative cutoff α using formula (36). Also shown are two data points for FMM based on setting the accuracy parameter to either 3 or 6 (digits of accuracy), which are the two possible values provided by the library. For each point, the CPU times and the errors have been averaged over 100 separated time steps, with error bars showing the standard deviation.

Figure 7 shows the MSM to be comparable in efficiency to the FMM. Approximating the force to a relative error of 0.5% should be sufficient for use with MD, provided that the approximation is continuous. However, FMM produces discontinuous forces, for which it is necessary to use high accuracy for stable dynamics,[34] as confirmed below.

4. Stable dynamics

Stability of dynamics is investigated for MSM and FMM for a constant energy MD simulation of the sphere of 10 002 water molecules. The water model is TIP3P with rigid bonds and angles as intended by the CHARMM force field.[35] A quartic restraining force is applied to model the surface tension of a 42 Å water sphere using the parameters suggested by the CHARMM documentation.[36] Simulations are run using NAMD[7] modified to include both the sequential MSM and FMM codes. The Verlet integrator is used with a 1 fs time step. The tested methods include the MSM using both cubic B-spline interpolation with cutoff distance 7 Å ($\alpha = 2.8$) and quintic B-spline interpolation with cutoff distance 12 Å ($\alpha = 4.8$) and the FMM using both the 3-digit (9 term expansion) and 6-digit (18 term expansion) accuracy options.

Figure 8 compares the total energy from two of the simulations for MSM quintic B-spline with that of FMM

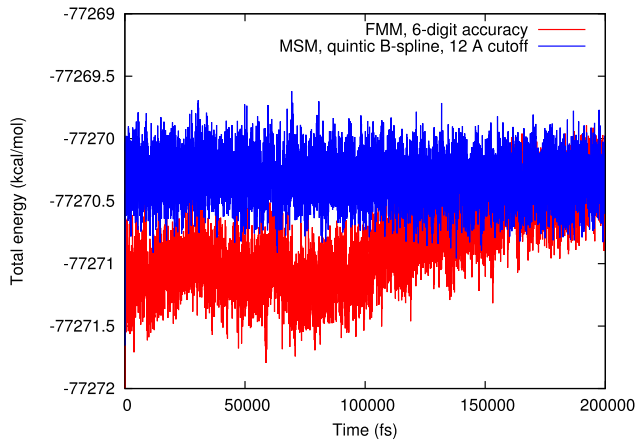

FIG. 8. Total energy for a sphere of 10 002 water molecules for 200 ps constant energy MD simulations comparing the MSM with the FMM.

TABLE II. The average μ and standard deviation σ of the total energy E, kinetic energy T, and potential energy U are compared for four simulations.

	μ_E	σ_E	μ_T	σ_T	μ_U	σ_U
FMM 3-digit	−62 147.09	8931.82	23 702.65	3566.13	−85 849.74	5374.94
FMM 6-digit	−77 270.85	0.32	17 998.10	91.27	−95 268.95	91.39
MSM cubic B-spline	−76 229.99	0.18	17 899.09	91.32	−94 129.08	91.43
MSM quintic B-spline	−77 270.31	0.18	17 997.73	91.58	−95 268.04	91.68

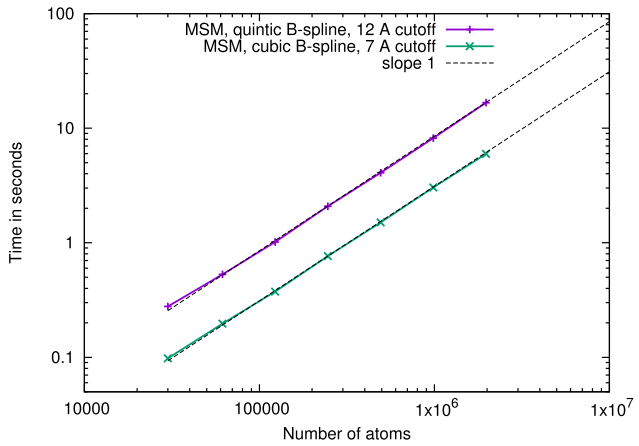

FIG. 9. Linear scaling of the MSM with system size on a single processor.

6-digit accuracy for a 200 ps trajectory. Results of energy averages and standard deviations for four simulations are shown in Table II. Although the cubic B-spline MSM and 3-digit FMM calculations both produce forces within 0.5% accuracy, the MSM is stable, whereas the FMM is not. The 6-digit FMM calculation and the two MSM calculations are considered energy conserving, by the criterion that the standard deviation of total energy is within 20% of the standard deviations of kinetic and potential energies.[37–39] However, the plot reveals less stability with 6-digit FMM than with MSM, and the FMM has larger standard deviation in total energy.

5. Scaling with problem size

Figure 9 shows the MSM code scaling linearly with system size on a single processor. For the experiments, several water spheres are created, each roughly twice the number of water molecules from the previous one, up to almost 2×10^6 atoms. The error is checked against that of the FMM code to ensure that MSM is consistently providing about six digits of accuracy in the energy.

IV. DISCUSSION

A. Comparison to other N-body methods

A fast N-body method constructs *an approximation* that facilitates computation by transferring the problem to a mesh/lattice/grid (still having $O(N)$ points). Fast N-body methods can be classified as 2-level or multilevel. Two-level methods, such as PME and P^{3}M, exploit the translation invariance $k(\mathbf{r},\mathbf{r}') = \kappa(\mathbf{r}-\mathbf{r}')$ of most kernels to perform calculations of mesh–mesh interactions using an FFT. Multilevel methods, such as tree methods, the FMM, and the MSM, typically assume that the kernel $k(\mathbf{r},\mathbf{r}')$ becomes more slowly varying as $|\mathbf{r}-\mathbf{r}'| \to \infty$ and exploit this property to do an approximation, using $O(\log N)$ levels of cells or grids. (Oscillatory kernels can also be handled, e.g., Ref. 40, but with greater complication.) Multilevel methods effect a separation of length scales, in which an increase in the range of the interactions is balanced by a commensurate decrease in the number of interacting cells or grid points. Both classes of methods obtain $O(N \log N)$ operation counts by means of a separable approximation of the kernel. Variants of multilevel methods use nesting to reduce the cost to $O(N)$. An alternative classification is to distinguish "kernel-splitting methods" (KSMs), such as PME, P^{3}M, and MSM, from HCMs, such as tree methods, the FMM, and hierarchical matrix methods.[41] This is discussed below in greater detail. Not discussed here are "indirect" methods based on solving an equivalent elliptic partial differential equation, e.g., the Poisson equation. Such approaches can be attributed to a historical accident, namely, the prior development of fast Poisson solvers.

1. Kernel-splitting methods

Kernel-splitting methods are of two kinds. FFT-based 2-level methods do a single splitting. It is typical to use $\mathrm{erf}(\beta r)$ for the long-range part, where β is a parameter chosen to optimize performance. With this choice, the short-range part decays very rapidly, and its cutoff distance a is set to a small multiple of $1/\beta$. Such methods interpolate the long-range part from a grid in either real or reciprocal space to permit the use of a 3-dimensional FFT. This is an $O(N \log N)$ algorithm on account of its use of the FFT. In contrast, the MSM is an $O(N)$ method that differs from 2-level methods in that it performs the calculation using a set of nested grids of increasing coarseness instead of just a single fine grid. For a 2-level method, the array of basis function coefficients is $\mathcal{K}^{1+} = \mathcal{K}^1$, and $\frac{1}{2}(q^1)^{\mathrm{T}}\mathcal{K}^1 q^1$ is the long-range energy, including excluded terms. Interpolation in real space produces an operator $\mathcal{K}^1$ that is a nonperiodic convolution, so the long-range energy is computable by a 3-dimensional FFT—if the dimension of the matrix is increased 8-fold.[42] Interpolation in reciprocal space is more involved (Ref. 43, Sec. 4.3). We consider the common kernel $k_1(\mathbf{r},\mathbf{r}') = \kappa_1(\mathbf{r}-\mathbf{r}')$. The long-range part is modified so that it decays rapidly beyond the diameter ℓ of the cluster of particles and then the cluster is replicated periodically with lattice spacing $\bar{\ell} > 2\ell$. The result is a kernel $\kappa_{1\mathrm{p}}(\mathbf{r})$ that has a rapidly converging Fourier series with analytically tractable coefficients that are inexpensive to evaluate. This involves making an error comparable to that of truncating the

short-range part at a distance a, if $\bar{\ell}$ exceeds 2ℓ by roughly $2a$. The kernel $\kappa_{1p}(\mathbf{r}-\mathbf{r}')$ is approximated by an *interpolated* truncated Fourier series,[1] giving

$$\frac{1}{2}(q^1)^{\mathsf{T}}\mathcal{K}^1 q^1 = \frac{1}{2}(\overline{Fq^1})^{\mathsf{T}} DFq^1, \tag{37}$$

where F is implemented as a 3-dimensional FFT and D is a diagonal matrix. This is PME, a variant of an older approach, viz., P^{3}M. In this case the domain has been enlarged *more than 8-fold*. Also, here the FFT is used not only to exploit the convolution property but to perform low-pass filtering as well.

The use of an FFT has the advantage that all the approximation errors are incurred on the finest grid; whereas, the use of multiple levels almost doubles the energy error and increases by a factor 4/3 the force error. To compensate for the loss of accuracy in the energy would require an increase in the cutoff a by a factor $2^{1/(p+1)}$. The need for a longer cutoff for the MSM is balanced by the fact that an FFT delivers its best performance only for grid dimensions that are powers of 2. Moreover, multilevel methods have several advantages over FFT-based 2-level methods: (i) For FFT-based methods, there are moderate difficulties and major inefficiencies handling nonperiodic boundaries. Specifically, each nonperiodic direction requires a cell dimension at least double the system dimension, which affects the FFT times.[43] Indeed, full electrostatics is rarely used in the case of nonperiodic boundary conditions due partly to the high cost. (ii) Another drawback is the difficulty of parallelizing the FFT in 3 dimensions. As a result, PME does not scale as easily as the MSM to large numbers of processors.[19] Indeed, the development of massively parallel computers is reviving interest in the use of FMM for MD.[44,45] Even with a single CPU node, the inherently non-local data access patterns of FFTs are less efficient than calculations with the wide 3-dimensional stencils of the direct gridpoint-to-gridpoint interactions of the MSM. (iii) Finally, FFTs cannot exploit situations where adaptive grids might be profitable, such as implicit solvent models; this would be possible with the MSM.

Force approximations obtained from KSMs violate Newton's third law unlike those obtained from HCMs. As a consequence, linear momentum fails to be conserved. Although not serious, this can be inconvenient. The usual remedy is to replace $\mathbf{F}_i$ by $\mathbf{F}_i - (1/N)\sum_j \mathbf{F}_j$, but this yields nonconservative forces and significant energy drift. However, such energy drift is avoided and linear momentum is conserved if the mass-weighted correction $\mathbf{F}_i - (m_i/m_{\text{tot}})\sum_j \mathbf{F}_j$ is used instead.[46]

2. Hierarchical clustering methods

Hierarchical clustering methods employ an oct-tree decomposition of space to partition the set of all particle pairs into pairs of particle clusters where the size of two clusters in a pair increases with separation distance. Interactions at the bottom level between small nearby clusters are computed directly. All other cluster pairs are, by construction, well separated and the interactions between particle pairs from a given pair of clusters are based on a (polynomial) approximation for the kernel $k(\mathbf{r},\mathbf{r}')$ particular to that cluster pair. Truncated Taylor expansions are used in practice. For $|\mathbf{r}-\mathbf{r}'|^{-1}$, each such Taylor polynomial is harmonic and can be expressed in terms of p^2 spherical harmonics, where $p-1$ is the degree of the polynomial, instead of $\frac{1}{6}p^3 + \frac{1}{2}p^2 + \frac{1}{3}p$ monomials. (The coefficients of these expansions are multipoles.) At the most basic level there is a close relationship between the multilevel summation method and hierarchical clustering methods and how they achieve their good scaling as a function of N. In particular, the fast multipole and related algorithms[47,48] have the same structure as the algorithm of Section III A 1. Many of the techniques for one class of methods transfer to the other. But there is one fundamental difference: hierarchical clustering algorithms do not split the interaction kernel—each interaction is present in only one of the terms of the multilevel sum. For more detailed information on HCMs or 2-level methods, Ref. 49 is recommended.

The main advantage of HCMs is that they can exploit special properties of the kernel. In particular, the harmonicity of the $1/r$ kernel can be used to reduce the number of terms in an order 4/6/8 approximation by factors of 1.25/1.56/1.88, respectively. However, these savings apply only to special kernels. Moreover, the disadvantages are significant: (i) An HCM produces an approximation to $k(\mathbf{r},\mathbf{r}')$ that is discontinuous as a function of particle positions $\mathbf{r},\mathbf{r}'$. This feature is intrinsic to HCMs because the shortest range interaction is not a polynomial and cannot be continuously matched to a longer range polynomial approximation. In contrast, KSMs can attain any degree of continuity. Lack of continuity is problematic for dynamics and minimization and disastrous for Hamiltonian dynamics[13,50] (which requires bounded Hessians to conserve total energy). Hence, HCMs may not be usable unless high accuracy is desired. This is the observation of Section III B. By contrast, all of the points plotted for multilevel summation conserve energy well. This is consistent with past experiments[13,43] showing that HCMs perform poorly compared to other methods for MD electrostatics. Indeed, HCMs are seldom used for MD, and a fairly recent review[51] does not mention them. (ii) From an implementation point of view, HCMs are more complicated due to the need for a list of pairs of interacting oct-tree cells, not to mention the (possible) use of special polynomials, such as spherical harmonics, that exploit properties of the kernel. This complexity not only makes it more challenging to utilize the capabilities of new computer architectures but makes it difficult to integrate the method into an application. (iii) To reduce the cost of computing forces, it is beneficial to evaluate slowly varying interactions less often than those that vary most rapidly and a good way to do this is multiple time stepping (or subcycling). Within each (outer) step, each interaction is integrated with multiple substeps of a size that is some fraction of the overall step size. For Hamiltonian systems, a fixed multiplicity must be chosen for each interaction. (Extensive experience indicates that failure to use exactly the same symplectic map for each outer time step results in a secular drift in energy.) For a nonbonded interaction, the appropriate multiplicity will vary, depending on the distance between the two particles. However, kernel-splitting methods provide a partitioning of the interaction, each part of which

can be integrated with its own fixed multiplicity, at the same time exploiting the finite ranges of all but the highest-level part. The FMM does not provide a splitting of the interactions.

ACKNOWLEDGMENTS

The work of D.J.H. and K.S. is supported by NSF Grant No. CHE090957273 and NIH Grant No. 9P41GM104601. That of D.J.H. is also supported by NSF Grant No. CCF08-30582 and a Computational Science and Engineering Fellowship (University of Illinois). The work of M.A.W., J.X., and R.D.S. is supported by NSF Grant No. CHE09-57024. That of R.D.S. is also supported by NSF Grant No. CCF08-30582. Additionally, the authors are grateful to B. Zhang and J. Huang for sharing their FMM code.

APPENDIX A: FUNDAMENTAL SPLINE COEFFICIENTS

To compute the coefficients ω_n of Section II B 2, proceed as follows: Let ψ be the bi-infinite sequence of all zeros except for $\psi_0 = 1$, and let ω be the sequence whose nth term is ω_n. Applying Eq. (10) to ψ gives $\mathcal{B}^{-1}\psi = \mathcal{R}\omega$ where $\mathcal{R}$ is the reversing operator. From this, follows

$$\mathcal{B}\mathcal{R}\omega = \psi,$$

which is an infinite banded system of linear equations. To solve this system, first obtain a Cholesky factorization $\mathcal{B} = \mathcal{G}\mathcal{G}^{\mathsf{T}}$ where $\mathcal{G} = g_0 + g_1E^{-1} + \cdots + g_{(p/2)-1}E^{1-p/2}$. Note that $\mathcal{G}^{\mathsf{T}} = \mathcal{R}\mathcal{G}\mathcal{R}$. To calculate the coefficients, create a semi-infinite banded matrix whose elements on the nth diagonal are $\Phi(n)$ and apply the Cholesky algorithm until there is no change from one row to the row that follows it. This converged row is the generic row of the operator $\mathcal{G}$. Using $\mathcal{G}^{\mathsf{T}} = \mathcal{R}\mathcal{G}\mathcal{R}$, the solution of $\mathcal{B}\mathcal{R}\omega = \psi$ can be broken into two steps:

$$\mathcal{G}\xi = \psi, \quad \mathcal{G}\omega = \mathcal{R}\xi.$$

Choosing $\xi_n = 0$ for $n < 0$, the value of ξ_0 is simply $1/g_0$. Exploiting symmetry $\omega = \mathcal{R}\omega$, solve a system of $p/2$ linear equations to obtain ω_n, $n = 0, 1, \ldots, (p/2) - 1$, and then use forward substitution to obtain ω_n, $n = p/2, (p/2) + 1, \ldots$. This process appears similar to one given in Ref. 52.

APPENDIX B: COEFFICIENTS OF THE BLURRING OPERATOR

It is shown below, using Ref. 25, Eq. (1.6) on p. 22, that the degree $(p/2) - 1$ polynomial $B_{p/2}$ of Section II B 4 is obtained from the following Maclaurin expansion:

$$\frac{s\sinh(sz)}{s^2 + 2(1 - \cosh(sz))} = \sum_{\substack{p=2 \\ p \text{ even}}}^{\infty} B_{p/2}(s^2)z^{p-1}. \tag{B1}$$

In particular,

$$\begin{aligned}\frac{s\sinh(sz)}{s^2 + 2(1 - \cosh(sz))} &= z\left(1 + \frac{s^2}{6}z^2 + \frac{s^4}{120}z^4 + O(z^6)\right)\left(1 - z^2 - \frac{s^2}{12}z^4 + O(z^6)\right)^{-1} \\ &= z\left(1 + \frac{s^2}{6}z^2 + \frac{s^4}{120}z^4 + O(z^6)\right)\left(1 + z^2 + (1 + \frac{s^2}{12})z^4 + O(z^6)\right) \\ &= z + \left(1 + \frac{s^2}{6}\right)z^3 + \left(1 + \frac{s^2}{4} + \frac{s^4}{120}\right)z^5 + O(z^7).\end{aligned}$$

1. Derivation of Eq. (B1)

From Eqs. (1.6) and (1.7) of Ref. 25, p. 22,

$$\frac{t-1}{t - \mathrm{e}^z} = 1 + \sum_{n=1}^{\infty}\frac{z^n}{(t-1)^n}\sum_{j=0}^{n-1}Q_{n+1}(j+1)t^j.$$

Taking the odd part w.r.t. z gives

$$\begin{aligned}\frac{(t-1)\sinh z}{t^2 + 1 - 2t\cosh z} &= \sum_{\substack{p=2 \\ p \text{ even}}}^{\infty}\frac{z^{p-1}}{(t-1)^{p-1}}\sum_{j=0}^{p-2}Q_p(j+1)t^j \\ &= \sum_{\substack{p=2 \\ p \text{ even}}}^{\infty}\frac{z^{p-1}}{(t-1)^{p-1}} \\ &\quad \times \sum_{m=1-p/2}^{(p/2)-1}Q_p((p/2)+m)t^{m+(p/2)-1}.\end{aligned}$$

Multiplying by $t/(t-1)$ gives

$$\begin{aligned}\frac{\sinh z}{t + t^{-1} - 2\cosh z} &= \sum_{\substack{p=2 \\ p \text{ even}}}^{\infty}\frac{z^{p-1}}{(t - 2 + t^{-1})^{p/2}} \\ &\quad \times \sum_{m=1-p/2}^{(p/2)-1}Q_p((p/2)+m)t^m.\end{aligned}$$

Let $t = 1 + s\sqrt{1 + s^2/4} + s^2/2$. The second summation can be shown to be a polynomial of degree $(p/2) - 1$ in s^2, denoted by $B_{p/2}(s^2)$. Also, $t - 2 + t^{-1} = s^2$. Replacing z by sz yields the stated result.

APPENDIX C: COEFFICIENTS OF THE ANTI-BLURRING OPERATOR

To get the expansion (12) of Section II B 4, write

$$B(\delta^2)^2(b_0 + b_1\delta^2 + \cdots + b_{(p/2)-1}\delta^{p-2}) = 1 - \delta^p C(\delta^2), \tag{C1}$$

where C is a polynomial of degree $p-3$. For example, for $p=4$,

$$\mathcal{B}^2(1-\frac{1}{3}\delta^2) = 1-\delta^4(\frac{1}{12}+\frac{1}{108}\delta^2)$$

and for $p=6$,

$$\mathcal{B}^2(1-\frac{1}{2}\delta^2+\frac{41}{240}\delta^4) = 1-\delta^6(-\frac{1}{20}-\frac{221}{19\,200}\delta^2-\frac{13}{19\,200}\delta^4-\frac{41}{3\,456\,000}\delta^6).$$

Comparing (C1) with (12), one sees that

$$\mathcal{B}^2\sum_m c_m E^m = C(\delta^2).$$

This is an infinite system of linear equations to solve for the coefficients c_m similar to that for the ω_m but with $2(p-3)$ additional nonzero values on the right-hand side and with $\mathcal{B}^2$ in place of $\mathcal{B}$. The symmetry of $\Phi(n)$ implies that $\mathcal{R}\mathcal{B}\mathcal{R}=\mathcal{B}$. Hence,

$$\mathcal{G}\mathcal{G}^{\mathsf{T}} = \mathcal{R}\mathcal{G}\mathcal{R}\mathcal{R}\mathcal{G}^{\mathsf{T}}\mathcal{R} = \mathcal{G}^{\mathsf{T}}\mathcal{G},$$

whence

$$\mathcal{B}^2 = \mathcal{G}_2\mathcal{G}_2^{\mathsf{T}},$$

where

$$\mathcal{G}_2 = \mathcal{G}^2.$$

The algorithm from Appendix A can be applied by solving $\mathcal{G}_2\xi' = C(\delta^2)\psi$ and $\mathcal{G}_2 c = \mathcal{R}\xi'$ where c is the symmetric sequence of unknowns c_n, taking care to calculate ξ'_n, $3-p \le n \le 0$.

[1]U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. Pederson, J. Chem. Phys. **103**, 8577 (1995).
[2]R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles* (McGraw-Hill, New York, 1981).
[3]A. Brandt and A. A. Lubrecht, J. Comput. Phys. **90**, 348 (1990).
[4]D. J. Hardy, Z. Wu, J. C. Phillips, J. E. Stone, R. D. Skeel, and K. Schulten, J. Chem. Theory Comput. **11**, 766 (2014).
[5]K. J. Bowers, E. Chow, H. Xu, R. O. Dror, M. P. Eastwood, B. A. Gregersen, J. L. Klepeis, I. Kolossváry, M. A. Moraes, F. D. Sacerdoti, J. K. Salmon, Y. Shan, and D. E. Shaw, in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing (SC06)* (ACM Press, New York, 2006).
[6]M. S. Friedrichs, P. Eastman, V. Vaidyanathan, M. Houston, S. LeGrand, A. L. Beberg, D. L. Ensign, C. M. Bruns, and V. S. Pande, J. Comput. Chem. **30**, 864 (2009).
[7]J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kalé, and K. Schulten, J. Comput. Chem. **26**, 1781 (2005).
[8]B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl, J. Chem. Theory Comput. **4**, 435 (2008).
[9]L. Greengard and V. Rokhlin, Acta Numer. **6**, 229 (1997).
[10]D. Beglov and B. Roux, J. Chem. Phys. **100**, 9050 (1994).
[11]W. Im, S. Berneche, and B. Roux, J. Chem. Phys. **114**, 2924 (2001).
[12]B. Sandak, J. Comput. Chem **22**, 717 (2001).
[13]R. D. Skeel, I. Tezcan, and D. J. Hardy, J. Comput. Chem. **23**, 673 (2002).
[14]O. E. Livne and A. Brandt, SIAM J. Matrix Anal. Appl. **24**, 439 (2002).
[15]M. S. Lee, J. F. R. Salsbury, and M. A. Olson, J. Comput. Chem. **25**, 1967 (2004).
[16]S. Shin, G. Zöller, M. Holschneider, and S. Reich, Comput. Geosci. **37**, 1075 (2011).
[17]I. Suwan, A. Brandt, and V. Ilyin, J. Math. Stat. **8**, 361 (2012).
[18]D. Tameling, P. Springer, P. Bientinesi, and A. E. Ismail, J. Chem. Phys. **140**, 024105 (2014).
[19]J. A. Izaguirre, S. S. Hampton, and T. Matthey, J. Parallel Distrib. Comput. **65**, 949 (2005).
[20]S. Plimpton, J. Comput. Phys. **117**, 1 (1995).
[21]D. J. Hardy, J. E. Stone, and K. Schulten, Parallel Comput. **35**, 164 (2009).
[22]W. Humphrey, A. Dalke, and K. Schulten, J. Mol. Graphics **14**, 33 (1996).
[23]S. G. Moore and P. S. Crozier, J. Chem. Phys. **140**, 234112 (2014).
[24]D. J. Hardy, "Multilevel summation for the fast evaluation of forces for the simulation of biomolecules," Ph.D. thesis, University of Illinois at Urbana-Champaign (2006), http://hdl.handle.net/2142/11173.
[25]I. J. Schoenberg, *Cardinal Spline Interpolation*, CBMS-NSF Regional Conference Series in Applied Mathematics Vol. 12 (Society for Industrial and Applied Mathematics, Philadelphia, 1973).
[26]B. Zhang, personal communication (2015).
[27]A. Brandt and C. Venner, SIAM J. Sci. Comput. **19**, 468 (1998).
[28]C. K. Chui, *An Introduction to Wavelets* (Academic Press, 1992).
[29]M. Reimer, Numer. Math. **44**, 417 (1984).
[30]F. Richards, J. Approx. Theory **14**, 83 (1975).
[31]G. Meinardus, J. Approx. Theory **16**, 289 (1976).
[32]D. R. Roe, RMSD analysis in CPPTRAJ, 2014, http://ambermd.org/tutorials/analysis/tutorial1/.
[33]J. Huang, J. Jia, B. Zhang, B.-Z. Lu, and X. Cheng, FMMLAP-uni: Uniform FMM Laplace solver, 2010, available online from http://fastmultipole.org/Main/FMMSuite/.
[34]T. C. Bishop, R. D. Skeel, and K. Schulten, J. Comput. Chem. **18**, 1785 (1997).
[35]B. R. Brooks, C. L. Brooks III, A. D. MacKerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V. Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, and M. Karplus, J. Comput. Chem. **30**, 1545 (2009).
[36]R. Venable, CHARMM documentation for c37b1, 2012, http://www.charmm.org/charmm/documentation/by-version/c37b1/params/doc/mmfp/.
[37]W. F. van Gunsteren and H. J. C. Berendsen, Mol. Phys. **34**, 1311 (1977).
[38]H. J. C. Berendsen and W. F. van Gunsteren, *Molecular Liquids* (Springer, 1984), pp. 475–500.
[39]H. J. C. Berendsen and W. F. van Gunsteren, "Molecular dynamics simulation of statistical mechanical systems," in *Proceedings of the International School of Physics, "Enrico Fermi,"* edited by G. C. Ciccotti and W. G. Hoover (North-Holland, Amsterdam, 1986), Vol. 97, pp. 43–65.
[40]A. Brandt, Comput. Phys. Commun. **65**, 24 (1991).
[41]W. Hackbusch, *Hierarchische Matrizen: Algorithmen und Analysis* (Springer, 2009).
[42]A. Neelov, S. A. Ghasemi, and S. Goedecker, J. Chem. Phys. **127**, 024109 (2007).
[43]E. L. Pollock and J. Glosli, Comput. Phys. Commun. **95**, 93 (1996).
[44]Y. Andoh, N. Yoshii, K. Fujimoto, K. Mizutani, H. Kojima, A. Yamada, S. Okazaki, K. Kawaguchi, H. Nagao, K. Iwahashi, F. Mizutani, K. Minami, S. ichi Ichikawa, H. Komatsu, S. Ishizuki, Y. Takeda, and M. Fukushima, J. Chem. Theory Comput. **9**, 3201 (2013).
[45]Y. Ohno, R. Yokota, H. Koyama, G. Morimoto, A. Hasegawa, G. Masumoto, N. Okimoto, Y. Hirano, H. Ibeid, T. Narumi, and M. Taiji, Comput. Phys. Commun. **185**, 2575 (2014).
[46]R. D. Skeel, D. J. Hardy, and J. C. Phillips, J. Comput. Phys. **225**, 1 (2007).
[47]L. Greengard and V. Rokhlin, J. Comput. Phys. **73**, 325 (1987).
[48]Z.-H. Duan and R. Krasny, J. Comput. Chem. **22**, 184 (2001).
[49]M. Griebel, S. Knapek, and G. Zumbusch, *Numerical Simulation in Molecular Dynamics* (Springer, Berlin, Heidelberg, 2007).
[50]J. J. Biesiadecki and R. D. Skeel, J. Comput. Phys. **109**, 318 (1993).
[51]P. Koehl, Curr. Opin. Struct. Biol. **16**, 142 (2006).
[52]M.-J. Lai, Math. Comput. **63**, 689 (1994).