Gaussian split Ewald: A fast Ewald mesh method for molecular simulation

Yibing Shan,[a] John L. Klepeis, Michael P. Eastwood, Ron O. Dror, and David E. Shaw
D. E. Shaw Research and Development, New York, New York 10036

(Received 26 August 2004; accepted 2 November 2004; published online 13 January 2005)

Gaussian split Ewald (GSE) is a versatile Ewald mesh method that is fast and accurate when used with both real-space and *k*-space Poisson solvers. While real-space methods are known to be asymptotically superior to *k*-space methods in terms of both computational cost and parallelization efficiency, *k*-space methods such as smooth particle-mesh Ewald (SPME) have thus far remained dominant because they have been more efficient than existing real-space methods for simulations of typical systems in the size range of current practical interest. Real-space GSE, however, is approximately a factor of 2 faster than previously described real-space Ewald methods for the level of force accuracy typically required in biomolecular simulations, and is competitive with leading *k*-space methods even for systems of moderate size. Alternatively, GSE may be combined with a *k*-space Poisson solver, providing a conveniently tunable *k*-space method that performs comparably to SPME. The GSE method follows naturally from a uniform framework that we introduce to concisely describe the differences between existing Ewald mesh methods. © *2005 American Institute of Physics.* [DOI: 10.1063/1.1839571]

I. INTRODUCTION

The bottleneck in classical molecular dynamics (MD) simulations is well known to be the evaluation of the nonbonded interactions, of which the slowly decaying electrostatic interactions between partial atomic charges (and ions) comprise a substantial part.[1] In the case of explicit solvent with periodic boundary conditions, which we consider here, a natural choice is the Ewald method.[2,3] The Ewald method divides contributions to force and energy into two main terms. The first, known as the direct summation, may be evaluated safely as a pairwise sum in real space with a distance cut off. The second can be evaluated by solving the Poisson equation for a smoothed charge distribution that is continuous, unlike the original point charge distribution. This term is traditionally evaluated in reciprocal (Fourier) space and is therefore generally known as the reciprocal summation. We will instead refer to it as the smooth summation, because some of the methods we will discuss solve it in real space rather than reciprocal space.

The introduction of Ewald mesh methods by Hockney and Eastwood[4] and their further development (principally by Darden and co-workers[5,6]) enabled the power of the fast fourier transform (FFT) algorithm to be applied to the smooth sum, leading to an $O(N \ln N)$ method, where N is the number of particles in the system. This development, together with ever-mounting evidence of the inadequacies of simple cutoff methods,[7–10] led to a shift in methodology such that FFT-based (*k*-space) Ewald techniques are now the most widely used methods for computing the electrostatic interactions in those biomolecular simulations in which the solvent is explicitly modeled.

Because of the desire to reach longer time scales in MD applications, researchers have continued to investigate alternative methods for long-range electrostatics that promise to further reduce computational cost or to allow more effective parallelization on large computer clusters.[11–16] Among the most promising of these are Ewald mesh methods that avoid use of the Fourier domain by solving the Poisson equation in real space using a multigrid approach.[13] While the FFT requires $O(N \ln N)$ computation, multigrid requires $O(N)$ computation. More importantly, interprocessor communication costs for parallelized multigrid scale asymptotically as $O(N^{2/3})$, while the communication costs of a parallelized multidimensional FFT scale as $O(N)$. When running on a cluster, these differences in communication requirements can prove decisive for performance.

Despite their superior asymptotic performance, existing multigrid methods have proven slower than FFT-based methods for the moderately sized systems most often studied by explicit-atom MD simulation (10^4–10^5 atoms). Perhaps surprisingly, this difference is mostly not a result of inefficiencies in solving the Poisson equation in real space. Indeed, it is the superior parallelizability of the calculations to solve the Poisson equation in a multigrid algorithm that ultimately leads to the attractiveness of real-space methods. However, an extra cost derives from what might seem to be trivial interpolation-type operations: spreading charge to the mesh before solving the Poisson equation and using the electrostatic potential on the mesh to calculate the forces on the atoms after solving the Poisson equation. In current real-space approaches, such as lattice Gaussian multigrid (LGM),[13] these operations require each grid point to interact with a large number of surrounding mesh points, consequently requiring a large number of computations per atom.

Investigators have used a diffusion method in the second stage of a two-stage charge spreading scheme to reduce the cost of charge spreading for real-space methods, but this speed comes at the cost of reduced accuracy. For the accuracy required in a typical biomolecular simulation, these ap-

[a]Electronic mail: yibing.shan@deshaw.com

proaches have been found to be less efficient than LGM.[12,13]

In contrast, leading FFT-based methods, such as SPME,[6] are able to use *k*-space (Fourier domain) correction terms to achieve the required level of accuracy while performing the mesh interpolation operations at a significantly lower cost. Existing FFT-based Ewald methods [including particle-particle particle-mesh (P3M),[4] particle-mesh Ewald (PME),[5] SPME,[6] and fast fourier Poisson (FFP)[17]] differ primarily in how they move between the quantities defined at atom positions (charge and force) and those defined on the mesh (charge and electrostatic potential) and in how these operations affect the Fourier-domain solution of the Poisson equation. A detailed numerical analysis of several Ewald mesh methods is available.[18,19]

This paper presents a simple framework in which the connections between the various real space as well as FFT-based Ewald mesh methods are especially clear. This intuitively leads to a versatile Ewald mesh method that efficiently adjusts the computational burden among the steps involved in calculating energies and forces of the smooth sum. We call this method Gaussian split Ewald (GSE). The charge spreading and force computation steps of GSE may be combined with either a *k*-space (FFT-based) or real-space (e.g., multigrid) solution of the Poisson equation; we refer to the resulting methods as *k*-GSE and *r*-GSE, respectively. *k*-GSE is conceptually simple and more easily tunable than existing FFT-based methods. More importantly, *r*-GSE reduces the computational burden of the charge spreading and force calculation steps for a real-space approach by half when compared to the best existing real-space methods, while maintaining equivalent levels of accuracy. This makes the computational burden of the charge spreading and force calculation steps in the GSE approach approximately equivalent to that of the most efficient FFT-based approaches.

To compare the performance of *r*-GSE and leading FFT-based methods, it is thus necessary to estimate the efficiency of both real-space multigrid and *k*-space Poisson solvers. On a typical computational cluster of 64 processors, estimates given in Appendix A suggest that *r*-GSE with a multigrid solver outperforms the most efficient FFT-based methods for systems larger than 30 000 atoms, a common system size for present-day biomolecular simulations. The analysis also shows that *r*-GSE performance scales more effectively as system size grows. For a 100 000-atom system (on a computational cluster of 64 processors), *r*-GSE is approximately four times more efficient than FFT-based methods. A recent study of parallelized solvers for the Poisson equation provides actual performance measurements for a multigrid solver and an FFT-based approach; the results closely match our estimates of processor utilization efficiency for a 100 000-atom system.[20]

In the following section we describe both the original Ewald method and existing Ewald mesh methods. We then describe our approach, the GSE method, which follows naturally from this general framework. The subsequent section compares the performance of real-space and FFT-based GSE with alternative real-space and FFT-based Ewald methods.

II. METHODS

We consider a cubic system of side length L containing N point charges $\{q_i\}$ at positions $\{\mathbf{r}_i\}$. The system is assumed to be neutral, i.e., $\Sigma_{i=1}^{N} q_i = 0$. Periodic boundary conditions are imposed in the standard way such that copies of the original system (which we term boxes, the original being the "primary box") tile space in a cubic lattice so that particle i and its images lie at $\mathbf{r}_i + \mathbf{n}L$, where $\mathbf{n} = (n_x, n_y, n_z)$, with $n_a = 0, \pm 1, \pm 2, ..., \pm 8$ $(a = x,\ y,\ or\ z)$. The assumption of cubic boxes is merely made to simplify the exposition; what follows may be straightforwardly extended to other box geometries. The total electrostatic energy per box in units where $4\pi\epsilon_0 = 1$ is

$$E = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}}}' \frac{q_i q_j}{|\mathbf{r}_i - \mathbf{r}_j - \mathbf{n}L|}. \tag{1}$$

The prime in the summation indicates that terms which simultaneously satisfy $\mathbf{n} = \mathbf{0}$ and $i = j$ are omitted from the sum. The most straightforward method of evaluating Eq. (1) and the corresponding forces is a straight cutoff method, i.e., to simply ignore terms in the sum with $|\mathbf{r}_i - \mathbf{r}_j - \mathbf{n}L|$ greater than some cutoff distance. Unfortunately, because of the slow decay of the Coulomb potential, this has been shown to lead to artifacts in dynamical simulations, at least for cutoff radii small enough to allow rapid computation of the sum. The best known alternative method for evaluating Eq. (1) is the Ewald method, which we now briefly outline.

A. Ewald sum

We give a brief sketch of the Ewald method. As is well known, the summation of Eq. (1) is only conditionally convergent. So, strictly speaking, we should start by defining exactly what we mean by it—for example, by including a convergence factor. However, since the rigorous derivation is available elsewhere[3] and is not central to this paper, we give a concise and transparent description while bearing in mind that it is not fully rigorous.

The summation of Eq. (1) may be equivalently written as

$$E = \frac{1}{2} \sum_{i=1}^{N} \phi_i(\mathbf{r}_i) q_i, \tag{2}$$

where $\phi_i(\mathbf{r}_i)$ is the electrostatic potential at $\mathbf{r}_i$ due to all charges except the charge at $\mathbf{r}_i$ itself. This electrostatic potential may be written as

$$\phi_i(\mathbf{r}_i) = \int \frac{\rho(\mathbf{r}) - \rho_i(\mathbf{r})}{|\mathbf{r} - \mathbf{r}_i|} d^3\mathbf{r}, \tag{3}$$

where $\rho(\mathbf{r}) = \Sigma_{i=1}^{N} \Sigma_{\mathbf{n}} q_i \delta(\mathbf{r} - \mathbf{r}_i - \mathbf{n}L)$ represents the charge distribution and $\rho_i(\mathbf{r}) = q_i \delta(\mathbf{r} - \mathbf{r}_i)$ is introduced to avoid including self-interactions.

The Ewald decomposition may be considered to arise from adding and subtracting a screening charge distribution to the numerator of Eq. (3). Specifically, we define the screening charge distribution to be $-\rho^{\sigma}(\mathbf{r})$, where

$$\rho^{\sigma}(\mathbf{r}) = \sum_{j=1}^{N} \sum_{\mathbf{n}} q_j \left(\frac{1}{2\pi\sigma^2} \right)^{3/2} \exp(-|\mathbf{r} - \mathbf{r}_j - \mathbf{n}L|^2 / 2\sigma^2). \tag{4}$$

We refer to $\rho^\sigma(\mathbf{r})$ as the ''smooth charge distribution'' since it is simply the original charge distribution with the point charges smoothed into Gaussian charge distributions of standard deviation σ. Shapes other than Gaussians have been investigated for the screening charge distribution,[21,22] but we stick with the Gaussian form.

In the Ewald decomposition, the total energy is split into three terms,

$$E=\frac{1}{2}\sum_{i,j=1}^{N}{}'\sum_{\mathbf{n}}\frac{q_iq_j\,\mathrm{erfc}(|\mathbf{r}_i-\mathbf{r}_j-\mathbf{n}L|/\sqrt{2}\sigma)}{|\mathbf{r}_i-\mathbf{r}_j-\mathbf{n}L|}+\frac{1}{2}\sum_{i=1}^{N}\phi^\sigma(\mathbf{r}_i)q_i-\frac{\Sigma_{i=1}^{N}q_i^2}{\sqrt{2\pi\sigma^2}},\tag{5}$$

where the first term is the direct summation, which arises from adding the screening charge distribution to the original charge distribution. Since $\mathrm{erfc}(x)$ decays rapidly with x, terms with $|\mathbf{r}_i-\mathbf{r}_j-\mathbf{n}L|$ larger than some cutoff value (to be determined by σ and the accuracy requirement) may be safely omitted from the sum. The second term is the smooth summation, with its potential,

$$\phi^\sigma(\mathbf{r}_i)=\int\frac{\rho^\sigma(\mathbf{r})}{|\mathbf{r}_i-\mathbf{r}|}d^3\mathbf{r},\tag{6}$$

arising from the smooth charge distribution. Finally, the last term (self energy term) is needed to cancel self-interactions present in the second term.

We have chosen to denote the long-range potential $\phi^\sigma(\mathbf{r})$ to emphasize its connection to the smooth charge distribution $\rho^\sigma(\mathbf{r})$ through the Poisson equation $\nabla^2\phi^\sigma=-4\pi\rho^\sigma$, the formal solution of which is simply Eq. (6). Since $\rho^\sigma(\mathbf{r})$ is smoothly varying for all $\mathbf{r}$, it can be accurately represented with a finite number of wave-number terms in k space, or on a grid in real space. The actual value of the grid spacing (or number of k-space terms) will depend on both the overall accuracy required and the value of σ. For fixed accuracy requirements, larger values of σ require larger cutoffs in the direct sum, but allow $\rho^\sigma(\mathbf{r})$ to be represented on a coarser grid (or with fewer k-space terms). Tuning σ therefore allows weight to be shifted between the direct and smooth summations, and σ is normally chosen to minimize the cost of the total computation. How the long-range contribution to the potential and the corresponding contributions to the energy and forces may be calculated is the subject of subsequent sections.

B. Reformulation of smooth summation with convolutions

In this section, we explicitly formulate the smooth summation in terms of convolutions. While this is mathematically very straightforward, it allows us to establish some relations (and a notation) that will prove useful in subsequent sections where mesh operations are also considered. We denote a convolution by the symbol $\otimes$. If $A\otimes B=C$, where A, B, and C are all real-space functions, then by the convolution theorem $\tilde{A}\tilde{B}=\tilde{C}$, where we use the tilde to denote k-space functions (note that in order to simplify notation we do not explicitly give arguments for these functions). For example, with $k=|\mathbf{k}|$ and $r=|\mathbf{r}|$, we note that the k-space representations of the Green function $\gamma(\mathbf{r})=1/r$ and the normalized Gaussian $G^\sigma(\mathbf{r})=(2\pi\sigma^2)^{-3/2}\exp(-r^2/2\sigma^2)$ are $\tilde{\gamma}=4\pi/k^2$ and $\tilde{G}^\sigma=\exp(-\sigma^2k^2/2)$, respectively. These functions are used frequently below.

To denote Fourier coefficients of periodic quantities (i.e., quantities that are identical in every box such as the charge density) we will use a subscript wave vector rather than an argument, e.g., $\tilde{A}_k=1/V\int_V A(\mathbf{r})e^{-i\mathbf{k}\cdot\mathbf{r}}d^3\mathbf{r}$, where $\int_V d^3\mathbf{r}$ denotes an integral over the primary box of volume $V=L^3$. The structure factor (the Fourier representation of the original point charge density) is a characteristic example of such a quantity:

$$\tilde{\rho}_k=\frac{1}{V}\sum_{j=1}^{N}q_je^{-i\mathbf{k}\cdot\mathbf{r}}.\tag{7}$$

It is also convenient to define the operation $\cdot$ between two periodic quantities to be

$$A\cdot B=\int_V A(\mathbf{r})B(\mathbf{r})d^3\mathbf{r}=V\sum_{\mathbf{k}}\tilde{A}_{\mathbf{k}}\tilde{B}_{-\mathbf{k}},\tag{8}$$

where $\mathbf{k}=2\pi\mathbf{n}/L$ and the second equality is simply Parseval's theorem. From these definitions, if B is an even function (like γ or G^σ) and A and C are periodic (like ρ or ρ_σ), it can be easily shown that

$$A\cdot(B\otimes C)=(A\otimes B)\cdot C.\tag{9}$$

We can now write the smooth summation using the notation defined above. First, it is clear that the smooth charge distribution is simply the original point charge distribution convolved with a normalized Gaussian of standard deviation σ,

$$\rho^\sigma=\rho\otimes G^\sigma.$$

The long-range contribution to the electrostatic potential is [from Eq. (6)] the charge density ρ^σ convolved with the Green function $\gamma=1/r$,

$$\phi^\sigma=\rho^\sigma\otimes\gamma=\rho\otimes G^\sigma\otimes\gamma,\tag{10}$$

which is simply the formal solution to the Poisson equation for the smooth charge distribution.

The smooth summation contribution to the energy is given by

$$E^\sigma=\tfrac{1}{2}\rho\cdot\phi^\sigma=\tfrac{1}{2}\rho\cdot(\rho\otimes G^\sigma\otimes\gamma).\tag{11}$$

It is useful to note that if we choose two quantities σ_A and σ_B such that $\sigma_A^2+\sigma_B^2=\sigma^2$ (and thus $G^{\sigma_A}\otimes G^{\sigma_B}=G^\sigma$), then from Eq. (9) we may equivalently rewrite the energy as

$$E^\sigma=\tfrac{1}{2}(\rho\otimes G^{\sigma_A})\cdot(\rho\otimes G^{\sigma_B}\otimes\gamma)=\tfrac{1}{2}\rho^{\sigma_A}\cdot(\rho^{\sigma_B}\otimes\gamma)=\tfrac{1}{2}\rho^{\sigma_A}\cdot\phi^{\sigma_B}.\tag{12}$$

As written in Eq. (11), the smooth sum energy is usually thought of as the energy of the point charges interacting with the potential field attributable to (minus) the screening charge distribution. Equation (12) shows that this is equivalent to the interaction of Gaussian distributed charges with the electrostatic potential due to a screening charge distribu-

tion built from more narrowly distributed charges. While the constraint $\sigma_A^2+\sigma_B^2=\sigma^2$ means that both σ_A and σ_B cannot be chosen independently, clearly the energy is independent of the ratio σ_A/σ_B which may be chosen freely. As described later, the choice $\sigma_A/\sigma_B=1$ is the basis of the FFP method. Note that the smooth summation contribution to the force on particle i may also be written in terms of ϕ^{σ_B} as

$$\mathbf{F}_i^{\sigma}=-\nabla_{\mathbf{r}_i}E^{\sigma}=-\nabla_{\mathbf{r}_i}\rho^{\sigma_A}\cdot\phi^{\sigma_B}, \tag{13}$$

where again we have used Eq. (9).

The convenience of the compact notation will become apparent when we additionally consider mesh operations in the following section, since these additional operations (i.e., charge spreading to mesh and potential interpolation from mesh) may also be written as convolutions. The ability to rewrite the energy in different forms, in a similar way to those described above, will then allow computational weight to be shifted between different convolutions and give a tunable parameter (similar to σ_A/σ_B) which may be varied to minimize the total computational burden.

C. Ewald mesh methods

Ewald mesh methods attempt to reduce the computational cost of the Ewald method [which is otherwise an $O(N^{3/2})$ method][3] by employing relatively fast grid-based operations to given an overall $O(N)$ or $O(N\ln N)$ method. There are two classes of mesh-based Ewald methods. Real-space mesh methods take advantage of fast finite-difference Poisson solvers to solve the Poisson equation in real space. The other class, FFT-based methods, solves the Poisson equation in k-space (like the classical Ewald method) but uses FFTs to transform a charge density to k-space and inverse FFTs to transform the electrostatic potential back to real space. Both approaches are explained in more detail below, but first we briefly describe two operations common to both: going from an off-mesh to on-mesh charge distribution (charge spreading), and conversely going from a potential defined on-mesh to the forces on the off-mesh particles. Each of these operations, which are sometimes called the discretization and interpolation steps, may also be written as a convolution. Optimal choices of the convolution kernels are crucial for the efficiency and accuracy of an Ewald mesh method.

For charge spreading, an on-mesh charge distribution ρ^S is created by convolving the point charge distribution with a charge spreading function S,

$$\begin{aligned}\rho^S(\mathbf{r}_m)=S\otimes\rho&=\int S(\mathbf{r}_m-\mathbf{r})\rho(\mathbf{r})d^3\mathbf{r}\\&=\sum_{i=1}^{N}\sum_{\mathbf{n}}q_iS(\mathbf{r}_m-\mathbf{r}-\mathbf{n}L)\\&\approx\sum_{i=1}^{N}q_iS(\mathbf{r}_{im}^{\min}).\end{aligned} \tag{14}$$

Here (and below) we use m as an index for mesh points that lie within the primary box. There are $N_m=(L/h)^3$ mesh points per box in a simple cubic array with a grid spacing of h between each point and its nearest neighbors. The label $\mathbf{n}$ runs over all boxes and $\mathbf{r}_{im}^{\min}$ is the vector from mesh point m (at $\mathbf{r}_m$ in the primary box) to the nearest image of particle i. The final equality (approximate) of Eq. (14) follows because the charge spreading function is chosen to falloff rapidly to zero with distance, so each mesh point only receives a contribution from, at most, one image of any particular particle [strictly speaking, this will be an approximation in cases where $S(\mathbf{r})$ merely decays rapidly and does not identically vanish beyond some distance]. The set of mesh points at which $S(\mathbf{r})$ is nonzero (and cannot reasonably be approximated by zero) is called the support of $S(\mathbf{r})$. The number of mesh points in this set is denoted by N_S. The size of the support partially determines the computational expense of the charge spreading step: for each charge i, $S(\mathbf{r}_{im}^{\min})$ must be evaluated at N_S mesh points. Since charge spreading (together with the analogous force calculation described below) turns out to comprise a majority of the computation, it would be desirable for $S(\mathbf{r})$ to have small support. Unfortunately, using very small supports leads to inaccuracies due to the effects of mesh discretization, so an appropriate balance must be found. Note that the object of charge spreading is not to try and reproduce something as close to the original point charge distribution as possible, nor even to try and reproduce ρ^{σ}. The object of charge spreading is merely to transfer the charges to a well-behaved on-mesh representation, so that further manipulations can rapidly lead to accurate energies and forces for the smooth summation.

The force computation step is almost identical to charge spreading, though perhaps less intuitively obvious. The force is a result of a convolution of an on-mesh potential $\phi^{(m)}$ and a second (vector-valued) function $\nabla_{\mathbf{r}}T$, where the scalar function T is even and is often chosen to be the same as the charge spreading function S. To see why this is so, we note that the potential $\phi^{(m)}$ is not typically a mesh-based representation of ϕ^{σ}. Instead, it is arranged (as explained below) that $\phi^{(m)}$ is the potential for which some prechosen interpolation function T satisfies $\phi^{\sigma}=T\otimes\phi^{(m)}$. This is quite analogous to the observation that the on-mesh charge distribution arising from the charge spreading step does not have to be a mesh-based approximation of either ρ^{σ} or ρ as long as the final forces and energies are correctly calculated. From Eqs. (10) and (12) we find

$$E^{\sigma}=\tfrac{1}{2}\rho\cdot(\phi^{(m)}\otimes T)=\tfrac{1}{2}(\rho\otimes T)\cdot\phi^{(m)}=\tfrac{1}{2}\rho^T\cdot\phi^{(m)}, \tag{15}$$

where $\rho^T=\rho\otimes T$. If the energy is calculated in real space (as is necessarily the case for real-space methods) then Eq. (15) can be interpreted as either using the convolution with T to take the off-mesh charges to the on-mesh clectrostatic potential or, equivalently, using the convolution to take the on-mesh potential to the off-mesh charges. Differentiating Eq. (15) with respect to position of particle i gives the following expression for force $\mathbf{F}_i^{\sigma}=-\nabla_{\mathbf{r}_i}E^{\sigma}=-\nabla_{\mathbf{r}_i}\rho^T\cdot\phi^{(m)}$. Straightforward manipulation shows that the force at particle i is

$$\mathbf{F}_i^{\sigma}(\mathbf{r}_i)=q_i\mathbf{F}^{\sigma}(\mathbf{r}_i), \tag{16}$$

where we have defined $\mathbf{F}^{\sigma}=(\nabla_{\mathbf{r}}T)\otimes\phi^{(m)}$. In practice, and analogously to Eq. (14), the force will be evaluated as a mesh convolution

$$\mathbf{F}_i^{\sigma}(\mathbf{r}_i) \approx q_i h^3 \sum_{m(\mathbf{r}_i,T)} \phi^{(m)}(\mathbf{r}_m) \nabla_r T(\mathbf{r}_{im}^{\min}), \tag{17}$$

where $m(\mathbf{r}_i,T)$ represents the set of support points for $T(\mathbf{r}_{im}^{\min})$.

We now outline the general procedure for real-space methods:

(i) Generate a mesh-based charge density using $\rho^S = \rho \otimes S$.

(ii) Iteratively solve the Poisson equation $\nabla^2 \phi^{(m)} = -4\pi\rho^S$ on the real-space mesh using a discrete representation of the Laplacian operator ∇^2. Algorithms such as multigrid or SOR (successive over-relaxation)[23] are applicable here. This yields the on-mesh potential $\phi^{(m)} = \gamma \otimes \rho^S$.

(iii) Calculate the forces according to Eq. (17). If the electrostatic potential at the particle positions is required, it may be calculated from $\phi^{\sigma} = \phi^{(m)} \otimes T$, and the energy follows from $E^{\sigma} = \frac{1}{2}\rho \cdot \phi^{\sigma}$.

Substituting the expression for $\phi^{(m)}$ into the equation for the electrostatic potential (at the particle positions) gives $\phi^{\sigma} = \rho \otimes S \otimes \gamma \otimes T$. From Eq. (10) we also know that $\phi^{\sigma} = \rho \otimes G^{\sigma} \otimes \gamma$, which implies that S and T should be chosen so that $S \otimes T = G^{\sigma}$. A natural choice is $S = T = G^{\sigma/\sqrt{2}}$, which has the advantage (in the case where ϕ^{σ} is not required) that it removes the need to perform a convolution for the energy, since it can also be obtained from $E^{\sigma} = \frac{1}{2}\rho^S \cdot \phi^{(m)}$. This choice corresponds to the FFP method, originally an FFT Ewald method but later combined with a real-space multigrid Poisson solver in the lattice Gaussian multigrid (LGM) approach of Sagui and Darden. Currently, LGM appears to be the most rigorous and accurate real-space Ewald method.

The typical procedure of FFT Ewald mesh methods is as follows:

(i) Generate a mesh-based charge density using $\rho^S = \rho \otimes S$.

(ii) Perform the FFT $\rho^S \rightarrow \tilde{\rho}^S$.

(iii) Solve the modified Poisson equation. The modifications to the Poisson equation, which we describe below, are represented as a change to the Green function. We denote the modified Green function as γ_{mod}. The on-mesh potential can then be described as $\phi^{(m)} = \gamma_{\mathrm{mod}} \otimes \rho^S$, which is equivalent to $\tilde{\phi}_{\mathbf{k}}^{(m)} = \tilde{\gamma}_{\mathrm{mod}}(\mathbf{k}) \tilde{\rho}_{\mathbf{k}}^S$ in k space.

(iv) Calculate the energy

$$E^{\sigma} = \tfrac{1}{2}\rho \cdot (T \otimes \phi^{(m)}) = \frac{V}{2} {\sum_{\mathbf{k}}}' \frac{\tilde{T}(\mathbf{k})}{\tilde{S}(\mathbf{k})} \tilde{\gamma}_{\mathrm{mod}}(\mathbf{k}) |\tilde{\rho}_{\mathbf{k}}^S|^2.$$

(As is common practice, we omit the $\mathbf{k}=\mathbf{0}$ term as indicated by the prime on the sum. A more rigorous analysis[4] shows how the $\mathbf{k}=\mathbf{0}$ term depends on the boundary conditions at infinity, with its omission corresponding to ''tin-foil boundary conditions.'')

(v) Perform the inverse FFT $\tilde{\phi}^{(m)} \rightarrow \phi^{(m)}$.

(vi) Calculate the forces according to Eq. (17). If the electrostatic potential at the particle positions is required it may be calculated from $\phi^{\sigma} = \phi^{(m)} \otimes T$.

Writing out all the contributions to the electrostatic potential explicitly (i.e., substituting the expression for $\phi^{(m)}$ into the equation for the electrostatic potential at the particle positions), we find

$$\phi^{\sigma} = \rho \otimes S \otimes \gamma_{\mathrm{mod}} \otimes T, \tag{18}$$

and since [from Eq. (10)] $\phi^{\sigma} = \rho \otimes G^{\sigma} \otimes \gamma$, we see that S, T, and γ_{mod} should ideally satisfy the following relationship:

$$S \otimes \gamma_{\mathrm{mod}} \otimes T = G^{\sigma} \otimes \gamma. \tag{19}$$

One possibility, which corresponds to the original FFP method, is to have $S = T = G^{\sigma/\sqrt{2}}$ with $\gamma_{\mathrm{mod}} = \gamma$. However, selecting $\gamma_{\mathrm{mod}} = \gamma$ demands that $S \otimes T = G^{\sigma}$. Thus, the variances of S and T must sum to σ^2, just as in the real-space method. Instead, it is more efficient to perform an additional convolution in k space to reduce the amount of spreading in real space. As noted above, the additional k-space convolution may be absorbed into a modified Green function $\gamma_{\mathrm{mod}} = U \otimes \gamma$. U varies in different mesh methods. The obvious inclination is to minimize the support as much as possible by using low-order B-spline functions. However, higher-order B-splines are smoother functions and lead to smaller errors, due to the discrete operations. Because of these competing effects, the optimum B-spline function typically extends over more than the nearest mesh points, but its support is much less than that of $G^{\sigma/\sqrt{2}}$. Note that the ability to modify the Green function is an important advantage of k-space approaches.

Equation (19) provides a simple but important criterion by which choices of S, T, and γ_{mod} can be evaluated. One caveat is that this equation is not strictly satisfied for any Ewald mesh method. There are at least two sources of error to consider: First, even in the continuous limit, in which grid spacing approaches zero, the functions could be chosen such that Eq. (19) is not satisfied. Second, a more subtle effect is attributable to the high frequency k-space cutoff intrinsically linked to the finite mesh size. In other words, unlike the functions S, T, G^{σ}, and γ in Eq. (19), which are introduced in their continuous forms, the accuracy of γ_{mod} is limited by its k-space representation on the mesh. Equation (19) is important because it tells us that the first source of error can be eliminated by appropriately choosing the functions for S, T, and γ_{mod} in the continuous limit. The second source of error may be diminished by choosing S and T with smaller high-frequency components, i.e., smooth and wide functions that generally require larger supports. Consideration of Eq. (19) naturally leads us to suggest a different splitting, which appears particularly promising for real-space calculations. This is Gaussian split Ewald, which we now describe.

D. Gaussian split Ewald method

The Gaussian split Ewald (GSE) method is a Ewald mesh method that can be used both with a real-space and k-space solver of the Poisson equation. Our main motivation in developing GSE was to devise a method of charge spreading and force calculation that would be fast and accurate, as well as compatible with a real-space solver. GSE involves two stages of charge spreading. First, charges are spread to the mesh by convolving with a function with small support. The small support ensures that the operation is rapid. Then, a second convolution is performed to further spread the

charges. In real-space GSE (*r*-GSE), this is a mesh operation with minimal computational burden. In *k*-space GSE (*k*-GSE), the second convolution is conveniently done in *k* space. The final interpolation step to calculate the forces is also performed rapidly by convolving with a function with small support. This general idea of two-stage charge spreading has been used before in diffusion-based methods such as lattice diffusion multigrid (LDM), but it was found to be significantly less accurate than the relatively slow one-stage charge spreading of the FFP type. The key difference in GSE is that Eq. (19) is strictly satisfied by our choice of S, T, and γ_{mod} in their continuous form, in contrast to the *ad hoc* choices of LDM and earlier two-stage charge spreading approaches.[12,13] A secondary but significant difference is in the method used to perform the second real-space charge spreading step, discussed below.

From Eq. (19) we know that with the choice of a standard Green function (i.e., $\gamma_{\mathrm{mod}}=\gamma$), S, and T should satisfy $S\otimes T=G^{\sigma}$. A two-stage charge spreading method splits S into two parts, so by definition $S=S_1\otimes S_2$, and Eq. (19) implies $S_1\otimes S_2\otimes T=G^{\sigma}$. We choose $S_1=T=G^{\sigma_1}$ and $S_2=G^{\sigma_2}$, with the Gaussian variances satisfying

$$2\sigma_1^2+\sigma_2^2=\sigma^2. \tag{20}$$

Substituting the GSE choice of S, T, and γ_{mod} into Eq. (18), we find

$$\phi^{\sigma}=\rho\otimes G^{\sigma_1}\otimes G^{\sigma_2}\otimes\gamma\otimes G^{\sigma_1}, \tag{21}$$

which effectively defines the GSE method. Note that in the particular case $\sigma_2=0$ we recover the FFP choice of $S=T=G^{\sigma/\sqrt{2}}$. In this sense, GSE may be viewed as a generalization of the FFP method. The key advantage of this generalization is that by tuning the ratio of σ_1/σ_2, GSE allows computational load to be shifted away from the bottleneck of the charge spreading and interpolation steps to a second charge spreading step. This second charge spreading step can be made highly efficient by taking advantage of the fact that an on-mesh convolution with a Gaussian is nearly trivial computationally but still satisfies Eq. (19).

r-GSE may be broken down into the following four steps:

(1) *First charge spreading.* $\rho^{\sigma_1}=\rho\otimes G^{\sigma_1}$. Each particle charge is spread on the grid by convolving with G^{σ_1}. From Eq. (14), the computation to be performed is

$$\rho^{\sigma_1}(\mathbf{r}_m)=\frac{1}{(2\pi\sigma_1^2)^{3/2}}\sum_{i=1}^{N} q_i\exp(-|\mathbf{r}_{im}^{\min}|^2/2\sigma_1^2). \tag{22}$$

(2) *Second charge spreading.* $\rho^S=\rho^{\sqrt{\sigma_1^2+\sigma_2^2}}=\rho^{\sigma_1}\otimes G^{\sigma_2}$. A second (and more efficient) on-mesh convolution with G^{σ_2} is performed to complete the charge spreading. To carry out this convolution, one takes advantage of the separability property of Gaussians, which allows the three-dimensional Gaussian convolution to be replaced by three trivial one-dimensional convolutions (one along each dimension). In other words, one convolves each row with a Gaussian (i.e., along the x axis), then convolves each column of the result with a Gaussian (i.e., along the y axis), and finally convolves with a Gaussian along the z axis. This replaces the diffusion method used previously (see Appendix B for details). We show this approach to be more accurate below.

(3) *Solution of Poisson equation on mesh.* $\phi^{(m)}=\rho^S\otimes\gamma$. In *r*-GSE, using the charge density ρ^S obtained in step (2), the Poisson equation is directly solved in real space (e.g., using either SOR or multigrid algorithms).[23] We follow Sagui and Darden in the choice of a Hermitian discrete representation on the finest grid to achieve higher accuracy.[13] Note that the mesh-based potential could equivalently be written $\phi^{(m)}\equiv\phi^{\sqrt{\sigma_1^2+\sigma_2^2}}$.

(4) *Interpolation.* $\phi^{\sigma}=G^{\sigma_1}\otimes\phi^{(m)}$ [and $\mathbf{F}_i^S=q_i(\nabla_{\mathbf{r}}G^{\sigma_1})\otimes\phi^{(m)}$]. With the on-mesh potential $\phi^{(m)}$ computed, the potential at each particle position ($\mathbf{r}_i$) may be obtained by interpolation, using G^{σ_1} as the interpolation kernel,

$$\phi^{\sigma}(\mathbf{r}_i)=\frac{h^3}{(2\pi\sigma_1^2)^{3/2}}\sum_m \exp(-|\mathbf{r}_{im}^{\min}|^2/2\sigma_1^2)\phi^{(m)}(\mathbf{r}_m). \tag{23}$$

The force on particle i is

$$\mathbf{F}_i^{\sigma}=\frac{q_i h^3}{(2\pi)^{3/2}\sigma_1^5}\sum_m \mathbf{r}_{im}^{\min}\exp(-|\mathbf{r}_{im}^{\min}|^2/2\sigma_1^2)\phi^{(m)}(\mathbf{r}_m). \tag{24}$$

Remember that $\mathbf{r}_{im}^{\min}$ is the vector from mesh point m (at $\mathbf{r}_m$ in the primary box) to the nearest image of particle i.

The energy can be calculated during the interpolation step or, alternatively, directly on the mesh (before the interpolation step), provided that the values of ρ^{σ_1} (on the mesh) from step (1) are stored in memory,

$$E^{\sigma}=\frac{1}{2}\rho^{\sigma_1}\cdot\phi^{(m)}=\frac{h^3}{2}\sum_m \rho^{\sigma_1}(\mathbf{r}_m)\phi^{(m)}(\mathbf{r}_m). \tag{25}$$

We note that it is also possible to use Eq. (19) to derive an accurate real-space method which uses B splines for the initial charge spreading and the force calculation steps, with a fast second charge spreading stage similar to that used in *r*-GSE (see Appendix C). Our numerical experiments show that this method is comparable to *r*-GSE in terms of performance, so we do not discuss it further here. Nevertheless, this illustrates the flexibility afforded by the convolution framework in formulating efficient and accurate Ewald mesh methods.

The same principles of splitting may also be used in an FFT approach. In *k*-GSE the second charge spreading step is done as a multiplication in *k*-space (which we write as a modification of the Green function), i.e., $\phi^{\sigma}=\rho\otimes G^{\sigma_1}\otimes(G^{\sigma_2}\otimes\gamma)\otimes G^{\sigma_1}=\rho\otimes G^{\sigma_1}\otimes\gamma_{\mathrm{mod}}\otimes G^{\sigma_1}$. Steps (1) and (4) of *k*-GSE are algorithmically identical to *r*-GSE, while steps (2) and (3) are replaced by FFT-based techniques to solve the Poisson equation in *k*-space. To be consistent with previous notation, the charge density produced after step (1) would be referred to as ρ^S in *k*-GSE [as opposed to *r*-GSE, which denotes the final on-mesh charge density after step (2) as ρ^S]. With this in mind, we refer readers back to steps (ii), (iii), and (v) of the general description of the FFT Ewald mesh method, which would be substituted in place of steps (2) and (3).

TABLE I. Comparison of operations used in various Ewald mesh methods.

Method	Initial charge spreading (S or S_1)[a] and force interpolation (T)[b] kernels	Second on- mesh charge spreading (S_2)[a]	Green's function (modified)[c] expressed in k-space	How Poisson equation is solved
PME[d]	Lagrange interpolation[e]	None	$\tilde{\gamma}(\mathbf{k})=4\pi/k^2$	Multiplication with $\tilde{\gamma}$ in k- space
SPME[f]	B-spline	None	$\tilde{\gamma}_{\mathrm{mod}}(\mathbf{k})=\dfrac{\tilde{G}^{\sigma}(\mathbf{k})\tilde{\gamma}(\mathbf{k})}{\tilde{S}^2(\mathbf{k})}$	Multiplication with $\tilde{\gamma}_{\mathrm{mod}}$ in k-space
P3M[g]	B-spline	None	$\tilde{\gamma}_{\mathrm{mod}}(\mathbf{k})=\dfrac{-i\mathbf{D}(\mathbf{k})\cdot\mathbf{k}\tilde{G}^{\sigma}(\mathbf{k})\tilde{\gamma}(\mathbf{k})}{\lvert\mathbf{D}(\mathbf{k})\rvert^2\tilde{S}^2(\mathbf{k})}$ [h]	Multiplication with $\tilde{\gamma}_{\mathrm{mod}}$ in k-space
FFP[i]	$G^{\sigma/\sqrt{2}}$	None	$\tilde{\gamma}(\mathbf{k})=4\pi/k^2$	Multiplication with $\tilde{\gamma}$ in k- space
LGM (real-space FFP)[j]	$G^{\sigma/\sqrt{2}}$	None	$\tilde{\gamma}(\mathbf{k})=4\pi/k^2$	SOR or multigrid in real space
LDM[j]	B-spline	Diffusion equation	$\tilde{\gamma}(\mathbf{k})=4\pi/k^2$	SOR or multigrid in real space
k-GSE	G^{σ_1} ($\sigma_1<\sigma/\sqrt{2}$ and $2\sigma_1^2+\sigma_2^2=\sigma^2$)	None	$\tilde{\gamma}_{\mathrm{mod}}(\mathbf{k})=\tilde{G}^{\sigma_2}(\mathbf{k})\tilde{\gamma}(\mathbf{k})$	Multiplication with $\tilde{\gamma}_{\mathrm{mod}}$ in k-space
r-GSE	G^{σ_1} ($\sigma_1<\sigma/\sqrt{2}$ and $2\sigma_1^2+\sigma_2^2=\sigma^2$)	Efficient on-mesh convolution with G^{σ_2} (separable in each dimension)	$\tilde{\gamma}(\mathbf{k})=4\pi/k^2$	SOR or multigrid in real space

[a]S refers to all convolutions needed to give the final on-mesh charge density (prior to solving the Poisson equation). In the r-GSE and LDM methods convolutions with S_1 and S_2 combine to give S.
[b]The charge spreading and force interpolation kernels are chosen to be identical. In r-GSE and LDM, this refers to the initial charge spreading kernel only.
[c]Modified Green functions appear only in k-space methods.
[d]Reference 5.
[e]In PME, forces are calculated by first employing the ($i\mathbf{k}$) difference operator in k-space and then individually converting force components to real space. Therefore, three (instead of one) inverse FFTs are required in this approach.
[f]Reference 6.
[g]Reference 4.
[h]$\mathbf{D}(\mathbf{k})$ is the Fourier transform of the employed finite differentiation operator in P3M (Refs. 18, 19).
[i]Reference 17.
[j]Reference 13.

E. Summary of Ewald mesh methods

We close this section with a simple summary of Ewald mesh methods. Equation (19) provides a convenient framework to compare Ewald mesh methods, since they differ principally only in their choices of S, T, and γ_{mod} and in the way the Poisson equation is solved (in real space or k space). In Table I, these differences are summarized for a number of commonly known Ewald mesh methods, as well as r-GSE and k-GSE.

We note that only r-GSE and LDM make use of the second on-mesh charge spreading. In these two cases, we need both S_1 and S_2 to describe the two charge spreading operations, which combine to give the final on-mesh charge density ρ^S. In all other cases, only one charge spreading kemel S is needed. We also note that the (initial) charge spreading kernel is equivalent to the force interpolation kernel in all methods. Finally, since real-space methods solve the Poisson equation in real space using the standard Green function, modified Green functions only appear in k-space methods.

III. RESULTS AND DISCUSSION

In this section, the performance of GSE is examined and compared to existing methods. As described above, a crucial quantity controlling the tradeoff between the speed and the accuracy of GSE is the ratio of the widths of the Gaussian used in the initial charge spreading (and force calculation) and the Gaussian used in the second charge spreading step (σ_1/σ_2). In this section, rather than directly using the ratio σ_1/σ_2, the related parameter κ is used to quantify the balance between the different Gaussians. κ is defined by

$$\sigma_1=\sigma\sqrt{\kappa},\quad 0\leqslant\kappa\leqslant\tfrac{1}{2}. \tag{26}$$

For a given value of σ, the value of κ also sets the standard deviation of the second Gaussian, according to

$$\sigma_2=\sigma\sqrt{1-2\kappa}. \tag{27}$$

In contrast to the discrete nature of the parameter space for the SPME and P3M interpolation schemes (e.g., in the SPME approach, the B-spline interpolation scheme is typically limited to even-integer orders), the continuous nature of the κ-parameter space can be easily exploited to fine tune the amount of computation needed for desired accuracy.

To analyze both real-space and k-space implementations of the GSE approach, comparisons were made to results obtained using existing mesh-based Ewald methods. Specifically, the LGM, a method developed by Sagui and Darden and based on FFP, was directly compared to r-GSE. The

LGM performs Gaussian spreading according to standard FFP (i.e., $\kappa=1/2$, so that $\sigma_1=\sigma/\sqrt{2}$ and $\sigma_2=0$), while the solution of Poisson's equation is achieved by solving the difference equation in real space. *k*-GSE was compared to both FFP and the widely used SPME method.

Tests were performed for a cubic system of side length 64 Å containing a small protein solvated with SPC water. The protein corresponds to the 20-residue designed protein known as Betanova.[24] The system was neutralized through the addition of three chloride counter ions and subsequently equilibrated using the minimization and molecular dynamics routines of GROMACS.[25,26]

To test the performance of the GSE approaches, input parameters were first established. It is typical to first specify a value for the cutoff used in the direct summation, which can be related to the standard deviation of the screening Gaussian charge distribution via the following relationship:

$$r_\sigma=\lambda_\sigma\sqrt{2}\sigma.$$

For most calculations, a direct summation cutoff of $r_\sigma=9$ Å was used (as this is also an appropriate value for truncation of van der Waals forces), along with a value of $\lambda_\sigma=3$. The specification of λ_σ essentially sets the value of the short-range potential at the cutoff (i.e., for the above value $[\mathrm{erfc}(\lambda_\sigma)]/r_\sigma=\mathrm{erfc}(3)/9=2.5\times10^{-6}$), such that contributions beyond the cutoff can be estimated and truncation errors in the direct summation can be controlled. For $\lambda_\sigma=3$, errors attributable to the direct summation are sufficiently small so as to be appropriate for molecular dynamics (1×10^{-4} to 1×10^{-5} relative rms force errors[27]). By setting the value for this parameter, the standard deviation of the Gaussian-spread charge distribution is set to $\sigma=3/\sqrt{2}$ Å, a value held constant for all subsequent calculations. In an analogous manner, a cutoff is specified for the charge spreading and force calculations

$$r_{\sigma_1}=\lambda_{\sigma_1}\sqrt{2}\sigma_1=\lambda_{\sigma_1}\sqrt{2\kappa}\sigma,$$

where Eq. (26) has been used. Both dimensionless parameters κ and λ_{σ_1} are varied in the following analysis to optimize performance.

As these methods are all mesh based, the calculations also require the specification of grid spacing. In the analysis presented below, this value is always held constant at $h=\frac{4}{3}$ Å, a setting that aims to balance errors arising from the direct and long-range potentials. With grid spacing held constant, one can determine the number N_{σ_1} of mesh-based grid points needed in charge spreading and force calculation (i.e., the "support") through the relationship

$$N_{\sigma_1}\approx\frac{4\pi}{3}\left[\frac{r_{\sigma_1}}{h}\right]^3=\frac{4\pi}{3}\left[\frac{\lambda_{\sigma_1}\sqrt{2\kappa}\sigma}{h}\right]^3=\frac{4\pi}{3}\left[\frac{9}{4}\lambda_{\sigma_1}\sqrt{\kappa}\right]^3. \tag{28}$$

The first two figures illustrate the performance of *r*-GSE as κ, and hence the number of support points is varied. Figure 1 depicts relative rms force error as a function of support for three values of λ_{σ_1}, where relative rms force error is defined as

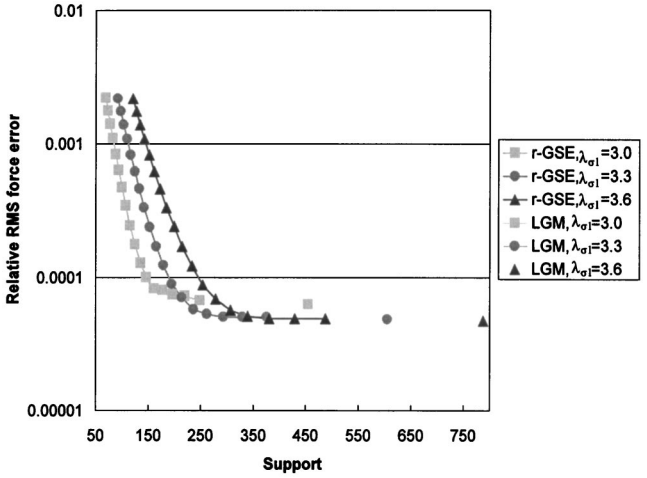

FIG. 1. Relative rms force error vs support (N_{σ_1}). In all cases $\lambda_\sigma=3$. Results are shown for *r*-GSE using three different values for λ_{σ_1} (3.0, 3.3, and 3.6), which correspond to data marked by squares, circles, and triangles, respectively. As the parameter values increase, the cutoffs used in charge spreading and force calculation also increase. The corresponding LGM results, which represent the upper bound on support and the lower bound on relative rms force error, are included as isolated data points (no connecting lines). We note that as *r*-GSE approaches the LGM limit in terms of N_{σ_1}, the second charge spreading may become prohibitively small ($\sigma_2\to0$). Relative rms force error may increase slightly because the second convolution is inadequately represented on the mesh (data not shown). These errors can be minimized by applying a precomputable *k*-space correction, although this is strictly not necessary, since the effect is not important in the regime over which *r*-GSE is most advantageous.

$$\Delta F=\sqrt{\left(\frac{\Sigma_i\Sigma_{\alpha\in x,y,z}[F_{i,\alpha}-F^*_{i,\alpha}]^2}{\Sigma_i\Sigma_{\alpha\in x,y,z}[F^*_{i,\alpha}]^2}\right)}, \tag{29}$$

with $F^*_{i,\alpha}$ denoting the exact force arising from Coulombic interactions. A larger value for λ_{σ_1} leads to smaller errors in charge conservation by effectively extending the cutoff distance used in charge spreading and force calculations. Relative rms force error is not reduced in going from 3.3 to 3.6, implying that the overall accuracy is essentially limited by the truncation of the direct summation. Note that the LGM results, whose support values give an upper bound, are shown for reference and provide a lower bound on the relative rms force error for the corresponding GSE results. A main advantage of the *r*-GSE approach is that significantly smaller support values are needed to meet specific force accuracy criteria. For example, for a typical value of relative rms force accuracy used during molecular dynamics simulations (e.g., relative rms force accuracy less than 1×10^{-4}), the minimum support is 147 points for *r*-GSE with $\lambda_{\sigma_1}=3.0$. In contrast, the LGM method (for $\lambda_{\sigma_1}=3.0$) requires three times the support. Because the force error plateaus as support increases, *r*-GSE can match the accuracy of LGM using significantly smaller support. For $\lambda_{\sigma_1}=3.3$, LGM requires 606 points, while *r*-GSE achieves its minimum force error with 292 points for essentially the same force accuracy—a more than twofold decrease in the number of the support points.

Different methods can be employed for spreading charge within the context of the *r*-GSE approach. In Fig. 2, the

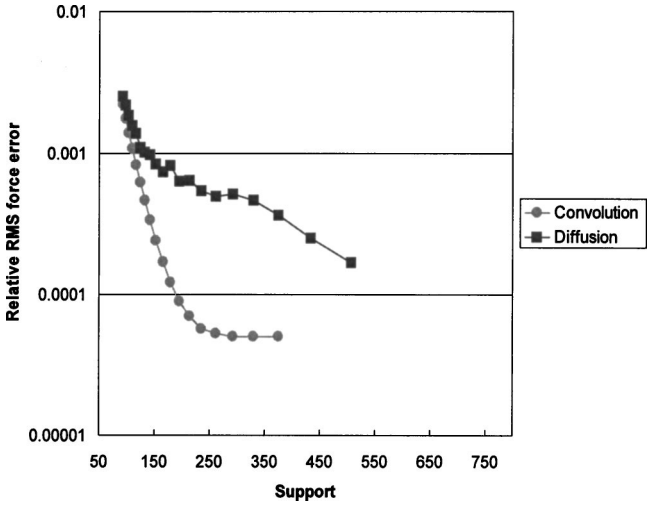

FIG. 2. Relative rms force error vs support (N_{σ_1}). Data points for *r*-GSE and diffusion are indicated by circles and squares, respectively. For *r*-GSE, $\lambda_\sigma=3$ and $\lambda_{\sigma_1}=3.3$, with the second convolution performed on-mesh. Results using the diffusion method for the second convolution are based on the approach first proposed by Beckers *et al.* (Ref. 12) and later adopted by Sagui and Darden (Ref. 13).

results of *r*-GSE are compared to those obtained when a diffusion scheme is substituted for the separable, one-dimensional on-mesh convolutions. In the diffusion scheme, charge is spread on-mesh through numerical integration of the diffusion equation.[12,13] Although the diffusion scheme provides a steady decrease in relative rms force error as support increases, the results are substantially less accurate. These inaccuracies arise from the low-order discrete representation of the diffusion equation.[13]

Figure 3 illustrates relative rms force error versus support for both *r*-GSE and *k*-GSE implementations. Comparison of the *r*-GSE and *k*-GSE results (for the same set of input parameters) clearly indicates that *k*-GSE achieves lower relative rms force errors. The *k*-GSE approach is more accurate, since it does not suffer from the limitations of locally representing the difference equation on the real-space mesh,[4] although the difference is not dramatic. Also shown in this plot are the results for *k*-GSE using a larger value for λ_σ. By increasing the value of this parameter from 3 to 4.5, the direct summation cutoff is extended from 9 to 13.5 Å, thereby virtually eliminating errors arising from truncation of the short-range potential. When $\lambda_\sigma=4.5$, the value of the short-range potential at the cutoff is erfc(4.5)/13.5$=1.5\times10^{-11}$. Although the computational burden associated with the direct summation grows, all other parameters remain unchanged, including the ratio of r_{σ_1} to *h*. The reduction in relative rms force error implies that the overall error was previously limited by direct summation results, so further improvements in the accuracy of long-range calculation are not necessary (for the original combination of direct summation cutoff and standard deviation of the Gaussian-spread charge distribution). The small magnitude of the improvement, less than an order of magnitude, also attests to a reasonable balance between these two sources of error.

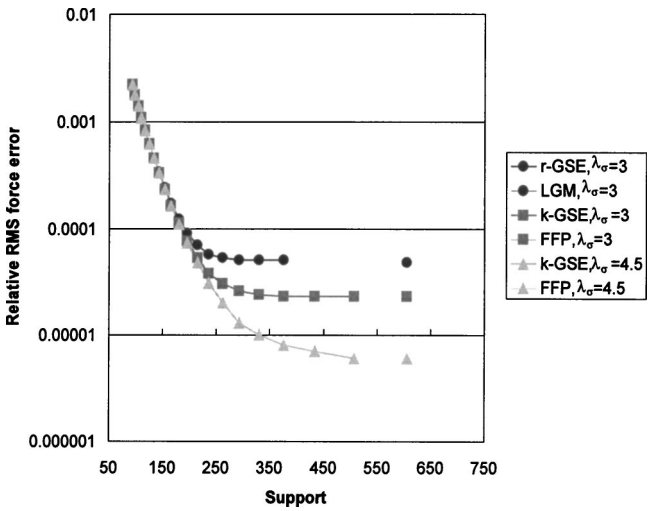

FIG. 3. Relative rms force error vs support (N_{σ_1}). In all cases $\lambda_{\sigma_1}=3.3$. The *r*-GSE results are shown (data marked by circles) for $\lambda_\sigma=3$, while *k*-GSE results are shown for both $\lambda_\sigma=3$ and $\lambda_\sigma=4.5$ (data marked by squares and triangles, respectively). The LGM and FFP results (shown as isolated data points), which correspond to $\sigma_1=\sigma/\sqrt{2}$, serve as an upper bound on the support and lower bound on relative rms force error.

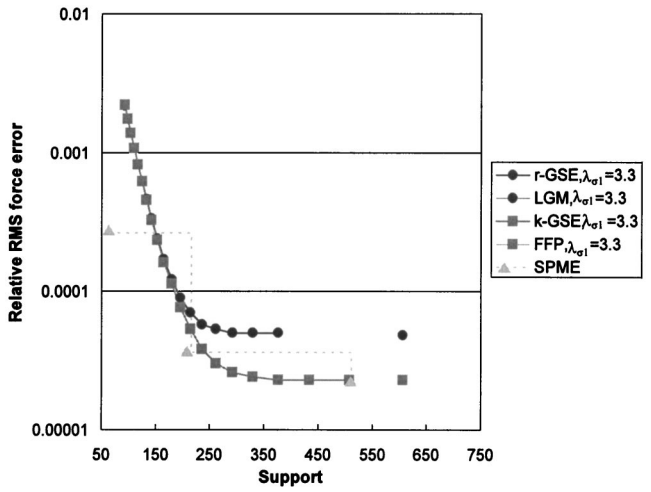

FIG. 4. Relative rms force error vs support (N_{σ_1}). In all cases $\lambda_{\sigma_1}=3.3$ and $\lambda_\sigma=3$. Circles indicate *r*-GSE results, while squares indicate *k*-GSE results. Triangles indicate SPME results for *B*-spline orders of 4, 6, and 8, in order of increasing support.

As shown in Fig. 4, both the *k*-GSE and *r*-GSE results exhibit smooth variation of relative rms force accuracy as a function of support. To emphasize this advantage of the GSE approach, Fig. 4 contrasts these results with those obtained using the widely adopted SPME method. At low accuracies, SPME outperforms the GSE approach. However, due to the discrete nature of the *B*-spline order parameter, the relative rms force error does not vary continuously (the data are connected as a steplike function to help visualize this behavior). Although it may be possible to reduce the size of the discrete ''steps'' by tuning other parameters, such as grid spacing and the standard deviation of the Gaussian-spread charge distribution, optimizing over this parameter space to achieve specific force accuracy with the minimum amount of computation is not a trivial task. As an illustrative example of the benefits of easily tunable support, Fig. 4 shows that to achieve force accuracies below 3.6×10^{-5}, a support of 512

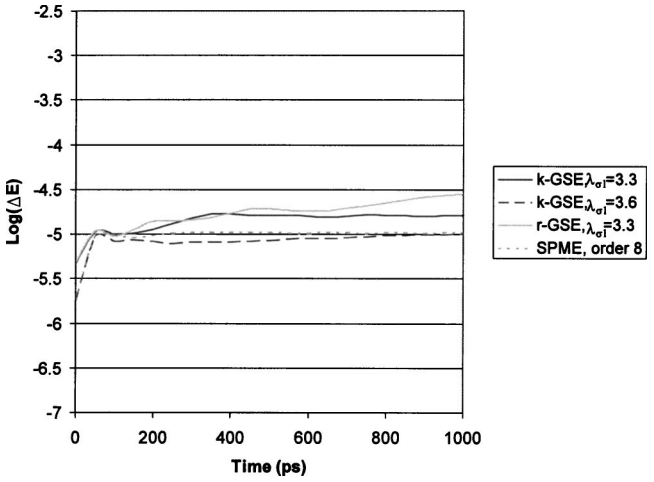

FIG. 5. Energy conservation during the course of 1 ns constant energy simulations. Energy conservation is measured by the logarithm of the total energy fluctuations, with values less than -2.5 indicating acceptable numerical accuracy. A 1 fs time step was used. *r*-GSE and *k*-GSE results are shown as solid and dashed lines, respectively. For comparison, results for SPME (dotted line) are also included.

points is required using SPME, while for *k*-GSE a support with 50% fewer points suffices. This effect is more pronounced over regions in which the force errors plateau.

The preceeding analyses evaluate the performance of GSE in a static sense—that is, by using snapshots taken from a molecular dynamics simulation. To address the issue of how GSE performs in a dynamic sense, one needs to analyze the stability of the molecular dynamics trajectory. These results are shown (for a cubic system of side length 24 Å containing only SPC water molecules) in Fig. 5, which presents a measure of energy conservation during the course of a 1 ns constant energy simulation. The measure chosen is based on the total energy fluctuation

$$\Delta E = \frac{1}{N_t} \sum_{i=1}^{N_t} \left| \frac{E_0 - E_i}{E_0} \right|, \tag{30}$$

where E_0 is the initial energy, N_t is the total number of time steps in time *t*, and E_i is the total energy at step (i). Acceptable numerical accuracy is achieved when $\Delta E \leqslant 0.003$, which corresponds to $\ln(\Delta E) \leqslant -2.5$.[27] As evidenced by Fig. 5, both GSE methods achieve high quality energy conservation, and the results improve slightly as the value of λ_{σ_1} is increased (*k*-GSE results are shown for λ_{σ_1} equal to 3.3 and 3.6). An increase in the value of λ_{σ_1} effectively leads to an increase in the cutoff distance used in charge spreading and force calculation, so a slight improvement is not unexpected. These results are compared to SPME, for which charge is both rigorously conserved during spreading and a continuous function of atom position due to the properties of the *B*-spline interpolation functions. The results for GSE are comparable to those achieved by SPME, an indication that charge conservation is sufficiently maintained and does not affect the level of energy conservation.

IV. CONCLUSIONS

The ability to effectively use real-space Ewald methods to calculate long-range electrostatic interactions would have significant consequences for biomolecular simulation. Asymptotically, real-space methods outperform the widely used FFT-based methods both in terms of computation and communication. However, these benefits have not yet been fully realized due to the greater computational burden associated with the charge spreading and force calculations in real-space methods. The *r*-GSE approach represents a significant advance for real-space methods by reducing the computational burden by a factor of 2 or more when compared to earlier real-space methods. This is practically important, since our estimates suggest that GSE should outperform FFT-based methods for systems larger than $\approx 30\,000$ atoms on a 64-node cluster.

r-GSE achieves its performance gains in several ways. The first is to minimize the size of the support required for charge spreading and force calculations by using a computationally efficient central convolution step. Although this idea is integral to many FFT-based approaches, which perform this second convolution in *k*-space, it has not been efficiently used in the context of earlier real-space methods. Existing methods have either avoided the central convolution entirely (LGM) or have used heuristic diffusion-based schemes to complete the charge spreading. The former scheme is computationally expensive, while the latter leads to unacceptable force accuracies. The GSE method converts the central convolution to a convolution by an on-mesh Gaussian, which can be implemented efficiently and accurately by taking advantage of the separability properties of the Gaussian. This approach follows naturally from a general framework that describes Ewald mesh methods as a series of convolutions.

In addition to its computational efficiency and tunability, the GSE method produces stable dynamics. Computational results clearly show that high-quality energy conservation is maintained. One caveat with regard to real-space Ewald methods in general is that efficient methods for calculating the virial must still be developed if more than the scalar isotropic value is required. For FFT approaches, this computation is trivial.

We note that it is straightforward to extend the GSE methodology to the treatment of multipolar interactions. This seems particularly promising because in the currently available algorithm[28] a rather large support is necessary for the higher-order *B* splines needed when high-order multipoles are to be represented, while the properties of Gaussian functions should provide greater flexibility in spreading multipoles. In addition, GSE should more naturally find applications in electrostatic calculations involving widely used Gaussian basis sets.

Finally, we mention one additional benefit of GSE that stems from the simplicity of the convolution kernels for charge spreading and force interpolation. Since these kernels have a simple Gaussian form, the GSE method lends itself to the development of specialized hardware. This direction holds much promise for extending the applicability of molecular dynamics simulation.

ACKNOWLEDGMENT

The authors would like to thank Celeste Sagui for making her LGM code available.

APPENDIX A: PERFORMANCE ESTIMATE OF MULTIGRID POISSON SOLVERS

In this appendix we estimate the system size at which a multigrid Poisson solver should outperform a *k*-space solver, and therefore give real-space Ewald mesh methods an advantage. On a 64-node cluster, this is found to be about 30 000 atoms. Since the computational burden of charge spreading and force calculation is roughly the same in *r*-GSE and in the best FFT-based approaches, ~30 000 atoms is also the system size where we anticipate that *r*-GSE should begin to outperform the FFT-based approaches.

First, we present the asymptotic scaling properties of both multigrid and FFT-based algorithms. The multigrid algorithm corresponds to the standard iterative V-cycle (as opposed to full multigrid).[29,30] In terms of computational cost, while the FFT scales as $O(N \ln N)$, multigrid scales only as $O(N)$.[29,30] For parallelized versions of these algorithms, if we increase system size while keeping the number of processors fixed, the scaling for interprocessor communication costs (in terms of the total amount of data to be communicated) is as follows:

FFT: $\beta_{\mathrm{FFT}} O(N) + \gamma_{\mathrm{FFT}}$,

Multigrid: $\alpha_{MG} O(\ln N) + \beta_{MG} O(N^{2/3}) + \gamma_{MG}$.

The prefactors α and β reflect the actual latency and bandwidth costs, respectively, associated with each method, while the γ are constants that do not depend on the number of atoms. The $O(N)$ scaling of FFT results from the need to transpose the data in order to process the three-dimensional FFT. In the case of multigrid, the $O(\ln N)$ term relates to the depth of the V-cycle, while the $O(N^{2/3})$ term corresponds to the dependency of the amount of data to be communicated on the surface area of the domain space. Multigrid, therefore, also outperforms FFT in terms of the overall asymptotic scaling of communication cost [$O(N^{2/3})$ versus $O(N)$], and such differences can prove decisive for performance when running on a cluster.

To quantify the actual crossover point between the two methods, one needs to estimate values for α, β, and γ. This is not a trivial task as these values depend on many variables including details of both software and hardware. In particular, we note that efforts (e.g., PMEMD[31,32]) to improve parallelization of FFT-based Ewald mesh methods are continuing. Our approach here is approximate, but based on reasonable estimates for key hardware parameters.

We consider a 64-node commodity cluster with a star topology for the communication network, a typical Beowulf-style (PC cluster) configuration. The network is assumed to have 100 Mbit/s bandwidth (node-to-switch), 20 μs latency (node-to-node), and a total latency for one node that is cumulative and depends on the total number of messages sent to other (distinct) nodes. We assume that we are sending single precision values and that the aggregate bandwidth of the switch is not limiting. Finally, we assume that each compute node achieves 1 GFLOP/s processing power.

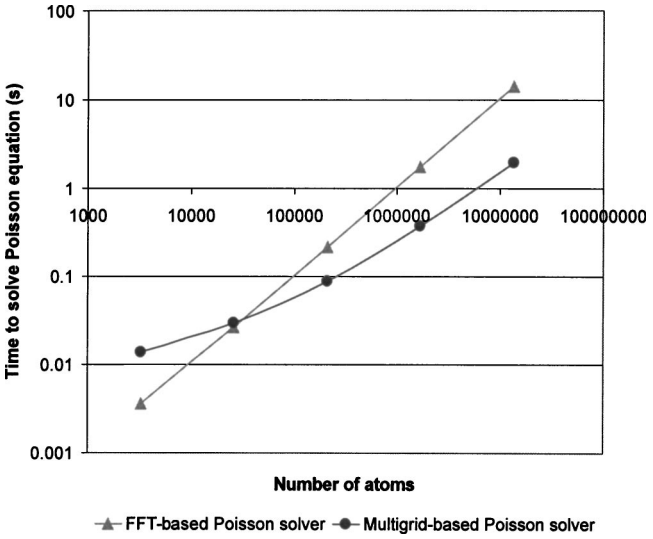

FIG. 6. Numerical estimate of performance of a multigrid Poisson solver as compared to an FFT-based solver. The plot shows (on log-log axes) the time to solve the Poisson equation vs number of atoms on a 64-node cluster. Atom density is taken as 0.1 atoms/Å^3.

With these assumptions it is straightforward to determine the perprocessor computation and interprocessor communication costs for parallelized FFT. Estimates are based on uniform 1 Å grid spacing and are simplified by considering only power-of-two length FFTs. The three-dimensional FFT is processed as a series of three one-dimensional FFTs, with each processor handling a distinct set of FFTs during each round. The analysis for multigrid is complicated by the fact that it is an iterative method. Our implementation closely follows that of Sagui and Darden,[13] in that Hermitian and seven-point central finite-difference representations of the Laplacian operator ∇^2 are chosen for the finest and coarsest grids, respectively. The appropriate numbers for total V-cycle iterations and relaxations were determined experimentally (based on satisfying a 1×10^{-4} relative rms force accuracy criterion), with the maximum number of levels in the V-cycle set at 5.

Figure 6 compares the estimated time required to solve the Poisson equation using a multigrid solver to the time required using a *k*-space solver (FFT). The figure indicates that the asymptotic advantage of the multigrid solver should become a practical one for systems larger than 30 000 atoms. The figure also shows that performance of multigrid solvers scales more favorably as system size grows.

APPENDIX B: GAUSSIAN CHARGE SPREADING ON REAL-SPACE MESH

This appendix shows how the second charge spreading can be made very efficient by taking advantage of the separability properties of Gaussians. Consider the second charge spreading convolution, which takes the initial on-mesh charge density ρ^{σ_1} and convolves it with the on-mesh Gaussian G^{σ_2},

$$\rho^S(\mathbf{r}_n) = \frac{1}{(2\pi\sigma_2^2)^{3/2}} \sum_{n=1}^{N_t} \rho^{\sigma_1}(\mathbf{r}_n) \exp(-|\mathbf{r}_{im}^{\min}|^2/2\sigma_2^2).$$

Both m and n are indices for mesh points that lie within the primary box, which contains a total of N_m mesh points.

The following pseudocode outlines how to rapidly evaluate this three-dimensional convolution by three one-dimensional convolutions (one along each dimension). We denote the number of grid points in each dimension as N_x, N_y, and N_z, with $N_m = N_x N_y N_z$. A uniform grid spacing in all dimensions, h, is assumed, and the number of grid points to which the Gaussians spread in each dimension is given by $2n+1$, where $n = \text{ceil}(r_{\sigma_2}/h)$. $\rho_{ijk}^{\sigma_1}$ is the three-dimensional array that stores the initial charge density on the mesh, ρ_{ijk}^{x} and ρ_{ijk}^{xy} are temporary arrays initialized to zero and ρ_{ijk}^{z} is the zero-initialized array that will hold the final result,

```
Loop k from 0 to N_z − 1
Loop j from 0 to N_y − 1
Loop i from 0 to N_x − 1
Loop ii from −n+i to n+i
    G = exp(−|h(i−ii)|^2/2σ_2^2)
    I = mod(ii, N_x)
    ρ^x_Ijk = ρ^x_Ijk + G ρ^σ1_ijk
End loop
End loop
End loop
End loop
Loop k from 0 to N_z − 1
Loop i from 0 to N_x − 1
Loop j from 0 to N_y − 1
Loop jj from −n+j to n+j
    G = exp(−|h(j−jj)|^2/2σ_2^2)
    J = mod(jj, N_y)
    ρ^xy_iJk = ρ^xy_iJk + G ρ^x_ijk
End loop
End loop
End loop
End loop
Loop i from 0 to N_x − 1
Loop j from 0 to N_y − 1
Loop k from 0 to N_z − 1
Loop kk from −n+k to n+k
    G = exp(−|h(k−kk)|^2/2σ_2^2)
    K = mod(kk, N_z)
    ρ^s_ijK = ρ^s_ijK + G ρ^xy_ijk
End loop
End loop
End loop
End loop
```

Note that G can be precomputed and stored in a table. Also note that this algorithm generalizes in a straightforward manner to a case in which grid spacing is different in each dimension.

APPENDIX C: *B*-SPLINE BASED REAL-SPACE EWALD MESH METHOD

In this appendix we use the criterion Eq. (19) to derive a real-space counterpart of the SPME method, where *B*-spline functions are used for the charge spreading and force calculation steps. Numerical tests (not presented) show that this real-space method is roughly equivalent to *r*-GSE in terms of efficiency and accuracy.

The method may be broken down into four steps:

(1) *First charge spreading.* $\rho^B = \rho \otimes B$. Each particle charge is spread to the grid by convolving with a 3D *B*-spline $B(\mathbf{r})$,

$$\rho^B(\mathbf{r}_m) = \sum_{i=1}^{N} q_i B(\mathbf{r}_{im}^{\min}).$$

We note that the first charge spreading and force calculation steps in this method are exactly the same as the equivalent step in SPME.

(2) *Second charge spreading.* $\rho^S = \rho^s \otimes S_2$. A second on-mesh convolution with $S_2 = G^\sigma \otimes B^l \otimes B^l$ is performed, where B^l is an "inverse" *B*-spline function that satisfies $f \otimes B \otimes B^l \equiv f$, where f is an arbitrary function. We note that S_2 may be easily precalculated as the inverse Fourier transform of $\tilde{G}_\sigma / B^2$. Since S_2 (like Gaussian or *B*-spline functions) is separable, the second charge spreading can be performed using cheap one-dimensional convolutions.

(3) *Solution of Poisson equation on mesh.* $\phi^{(m)} = \rho^S \otimes \gamma$.

(4) *Interpolation.* $\phi^\sigma = B \otimes \phi^{(m)}$ [and $\mathbf{F}_i^s = q_i(\nabla_r B) \otimes \phi^{(m)}$]. With the on-mesh potential $\phi^{(m)}$ computed, the potential at each particle position ($\mathbf{r}_i$) may be obtained by interpolation, using B as the interpolation kernel,

$$\phi^\sigma(\mathbf{r}_i) = \sum_m B(\mathbf{r}_{im}^{\min}) \phi^{(m)}(\mathbf{r}_m).$$

The force on particle i is

$$\mathbf{F}_i^\sigma = -q_i \sum_m (\nabla_{\mathbf{r}} B(\mathbf{r}_{im}^{\min})) \phi^{(m)}(\mathbf{r}_m),$$

where $\mathbf{r}_{im}^{\min}$ is the vector from mesh point m (at $\mathbf{r}_m$ in the primary box) to the nearest image of particle i.

It is easy to show that this *B*-spline based method satisfies Eq. (19), our proposed criterion for accurate Ewald mesh methods.

[1] C. Sagui and T. A. Darden, Annu. Rev. Biophys. Biomol. Struct. **28**, 155 (1999).
[2] P. P. Ewald, Ann. Phys. (Leipzig) **64**, 253 (1921).
[3] H. G. Petersen, S. W. Deleeuw, and J. W. Perram, Mol. Phys. **66**, 637 (1989).
[4] R. Hockney and J. Eastwood, *Computer Simulation Using Particles* (Adam Hilger, Bristol, 1988).
[5] T. Darden, D. York, and L. Pedersen, J. Chem. Phys. **98**, 10089 (1993).
[6] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. G. Pedersen, J. Chem. Phys. **103**, 8577 (1995).
[7] C. L. Brooks, B. M. Pettitt, and M. Karplus, J. Chem. Phys. **83**, 5897 (1985).
[8] J. Norberg and L. Nilsson, Biophys. J. **79**, 1537 (2000).
[9] P. Mark and L. Nilsson, J. Comput. Chem. **23**, 1211 (2002).

[10] M. Patra, M. Karttunen, M. T. Hyvonen, E. Falck, P. Lindqvist, and I. Vattulainen, Biophys. J. **84**, 3636 (2003).
[11] L. Greengard and V. Rokhlin, J. Comput. Phys. **73**, 325 (1987).
[12] J. V. L. Beckers, C. P. Lowe, and S. W. De Leeuw, Mol. Simul. **20**, 369 (1998).
[13] C. Sagui and T. Darden, J. Chem. Phys. **114**, 6578 (2001).
[14] R. D. Skeel, I. Tezcan, and D. J. Hardy, J. Comput. Chem. **23**, 673 (2002).
[15] D. Wolf, P. Keblinski, S. R. Phillpot, and J. Eggebrecht, J. Chem. Phys. **110**, 8254 (1999).
[16] A. C. Maggs and V. Rossetto, Phys. Rev. Lett. **88**, 196402 (2002).
[17] D. York and W. T. Yang, J. Chem. Phys. **101**, 3298 (1994).
[18] M. Deserno and C. Holm, J. Chem. Phys. **109**, 7678 (1998).
[19] M. Deserno and C. Holm, J. Chem. Phys. **109**, 7694 (1998).
[20] J. A. Izaguirre and T. Matthey (unpublished).
[21] D. M. Heyes, J. Chem. Phys. **74**, 1924 (1981).
[22] P. F. Batcho, D. A. Case, and T. Schlick, J. Chem. Phys. **115**, 4003 (2001).
[23] L. Collatz, *The Numerical Treatment of Differential Equations* (Springer, Berlin, 1960).
[24] T. Kortemme, M. Ramirez-Alvarado, and L. Serrano, Science **281**, 253 (1998).
[25] H. J. C. Berendsen, D. Vanderspoel, and R. Vandrunen, Comput. Phys. Commun. **91**, 43 (1995).
[26] E. Lindahl, B. Hess, and D. van der Spoel, J. Mol. Model. [Electronic Publication] **7**, 306 (2001).
[27] R. H. Zhou, E. Harder, H. F. Xu, and B. J. Berne, J. Chem. Phys. **115**, 2348 (2001).
[28] C. Sagui, L. G. Pedersen, and T. A. Darden, J. Chem. Phys. **120**, 73 (2004).
[29] A. W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial* (SIAM, Philadelphia, 2000).
[30] W. Hackbush and Trottenburg, *Multigrid Methods* (Springer, Berlin, 1982).
[31] R. E. Duke and L. G. Pedersen, PMEMD3, University of North Carolina, Chapel Hill, 2003.
[32] D. A. Case, D. A. Pearlman, J. W. Caldwell *et al.*, AMBER7, University of California, San Francisco, 2002.