A non-periodic particle mesh Ewald method for radially symmetric kernels in free space

Dennis M. Elking[a]

[a]*Fielddyne, L.L.C.*

Abstract

The FFT-based smooth particle mesh Ewald (PME) method is extended to non-periodic charge systems interacting via a radially symmetric kernel $f(r)$. The proposed non-periodic PME (NPME) method begins by splitting the kernel $f(r)$ into a short-range component $f_s(r)$ and a smooth long-range component $f_l(r)$. A Fourier extension for $f_l(r)$ is computed numerically using discrete Fourier transform interpolation, enabling efficient treatment of anisotropic rectangular charge volume and offering additional flexibility in the choice of kernel splitting. A derivative-matched (DM) splitting is introduced for general radially symmetric kernels $f(r)$, improving computational performance over traditional Ewald splitting methods. An optimized grid storage algorithm for NPME is proposed, reducing total grid memory by a factor of four. The NPME algorithm is implemented in a C++ library, `npme`, which supports both pre-defined kernels (e.g. $1/r, r^\alpha, \exp(ik_0 r)/r$) and user-defined kernels via C++ classes. `npme` is benchmarked and compared to `fmm3D` on test systems in computational chemistry and computational electromagnetics. As a practical application, NPME is combined with Method of Moments (MoM) to form a hybrid MoM-NPME algorithm for calculating the radar cross section (RCS) of a perfect electric conductor (PEC). The MoM–NPME method is used to compute the bistatic RCS of a 1-meter PEC sphere at 37.8 GHz and the monostatic RCS of the NASA almond at 75 GHz.

PROGRAM SUMMARY

Program Title: `npme`

CPC Library link to program files: (to be added by Technical Editor)

*Corresponding author.

E-mail address: delking@fielddyne.com

Developer's repository link: `https://github.com/ElkingD/npme`
Licensing provisions: Apache 2.0
Programming language: C++
Supplementary material: A supplementary material containing additional technical details and results is provided.
Nature of problem: `npme` computes the potential and its gradient of N non-periodic charges contained within a 3D rectangular volume interacting through a radially symmetric kernel. The charges, kernel, potential, and its gradient can be real or complex. Pre-defined kernels are given by $1/r, r^\alpha, \exp(ik_0 r)/r$ where α is real and k_0 is complex. In addition, users can implement their own radially symmetric kernel with C++ classes.
Solution method: `npme` computes the potential and its gradient using the non-periodic particle mesh Ewald (NPME) method proposed here in $O(N \log N)$ operations. `npme` is optimized for variable charge and static geometry inputs using vector intrinsic functions and OpenMP.
Additional comments including restrictions and unusual features: `npme` requires the Intel® C/C++ compiler with Math Kernel Library.

---

1. Introduction

Many computational problems in physical science and engineering can be studied by calculating classical potentials ϕ_i of N interacting charges q_i with positions $\mathbf{r}_i$ given by

$$\phi_i = \sum_{j \neq i} f(r_{ij}) q_j \tag{1}$$

where $r_{ij} \equiv |\mathbf{r}_i - \mathbf{r}_j|$ and $f(r)$ is a kernel function for pairwise interactions. Some example kernels include: $1/r$ for electrostatic problems, $1/r^6$ for molecular dispersion problems, and $\exp(ik_0 r)/r$ for computational electromagnetic problems. The direct evaluation of Eq. (1) requires $O(N^2)$ operations. Algorithms, such as the fast multipole method (FMM) [1, 2], have been developed to reduce this cost to $O(N)$ operations. For periodic problems, the particle mesh Ewald (PME) method [3, 4] has $O(N \log N)$ complexity and is widely used in molecular dynamics simulations with the $1/r$ and $1/r^6$ kernels. A method of computing Eq. (1) is proposed in this work by applying PME to non-periodic charge systems interacting through radially symmetric kernels $f(r)$ in free space. Unlike previous

methods, the long-range kernel contribution is represented as a Fourier extension computed numerically with discrete Fourier interpolation (DFT). This numerical approach eliminates the need for analytical Fourier integrals, leads to efficient treatment of anisotropic rectangular volumes, and introduces additional flexibility to the choice of kernel splitting.

An important class of modern algorithms for efficiently calculating the sum in Eq. (1) is based on the Ewald summation [5] method for calculating the electrostatic energy of a periodic crystal represented by point charges. The Ewald electrostatic energy is expressed as the sum of a short-range contribution and a long-range contribution computed with a Fourier transform. However, the computational cost of Ewald summation with an optimized cutoff radius, which balances real space and Fourier space costs, scales [6] as $O(N^{3/2})$. Hockney and Eastwood [7] proposed the particle-particle-particle mesh (P^3M) method as a faster alternative with $O(N \log N)$ complexity in which the charge density is interpolated on a grid and long-range interactions are computed by a convolution with the fast Fourier transform (FFT). P^3M has been applied to multiple types of problems, including molecular dynamics [8, 9, 10, 11] and cosmological simulations [12, 13].

Based on the work of Hockney and Eastwood, Darden [3, 4] developed the particle mesh Ewald (PME) method, in which the Fourier transform of charge density is interpolated on a grid using Lagrange polynomials [3] and later improved with B-spline polynomials [4]. Smooth PME [4] was applied to $f(r) = 1/r^p$ kernels for $p \geq 1$ and later generalized to higher order multipoles [14] and Gaussian charge density [15, 16]. Early studies on the P^3M and PME methods have demonstrated improved performance over FMM for uniformly distributed periodic systems [17, 18].

Smooth PME is widely used in many molecular dynamics codes [19, 20, 21, 22, 23, 24]. Recent PME methodological advances in molecular dynamics include higher order multipole spherical tensor derivatives [25, 26], analytic Hessians [27], compression strategies [28], many body dispersion effects [29], PME-FMM hybrid approaches [30], and application to hybrid quantum mechanical/molecular mechanical systems [31, 32]. Beyond molecular dynamics, smooth PME has also been applied to problems involving the Stokes equation [33] with periodic boundary conditions and to electrostatic interactions in free space for spherical charge clusters [34].

In addition to the P^3M and PME methods, other fast Ewald approaches include the fast Fourier Poisson (FFP) method [35, 26], the Gaussian split

Ewald method [36], and the Spectral Ewald (SE) method [37, 38, 39, 40]. The FFP and SE methods use Gaussian functions rather than polynomial interpolation to distribute the charge density on a grid. SE has also been applied to the Stokes potential in free space [38], leveraging FFT convolution techniques for truncated Green's functions [41, 42].

The u-series method [43] provides an alternative kernel splitting for the $f(r) = 1/r$ potential, with improved computational performance for molecular dynamics. The random batch Ewald (RBE) method [44, 45] is a stochastic variant of Ewald summation that reduces computational cost in molecular dynamics simulations by randomly sampling particle pairs within each batch. Other recent developments include the use of nonequispaced fast Fourier transforms (NFFT) for Ewald summation [46, 47], FFT-based methods for computing the electromagnetic Green's function in rectangular cavities [48], and mixed periodicity methods [47, 39, 40]. FFT-based electrostatic methods have also been extended to gas-phase electronic structure calculations [34, 49].

Many non-periodic fast Fourier methods are based on the convolution of a spherically truncated kernel with a Gaussian function [49, 41, 38, 50]. However, this approach would preclude kernels with higher order singularities, such as $f(r) = 1/r^6$. In addition, the spherical nature of the truncated kernel approach is limited to cubic volumes [34, 41, 38, 50]. To address this limitation, Greengard et al. [42] proposed a numerical method for calculating the Fourier transform of a spatially truncated kernel within an anisotropic rectangular volume.

This work builds on the smooth PME framework and applies it to non-periodic charge systems by replacing analytic Fourier transforms with a numerically computed Fourier extension. This approach avoids the limitations of prior methods and supports flexible kernel choices and kernel splittings. In particular, a "derivative-matched" (DM) scheme is introduced that improves performance, especially for the $f(r) = 1/r$ case. The method also supports efficient computation in both cubic and anisotropic volumes and forms the basis of the NPME algorithm detailed below.

The proposed non-periodic PME (NPME) method begins with a kernel splitting of $f(r)$ into short-range and long-range components

$$f(r) = f_s(r) + f_l(r) \tag{2}$$

where $f_s(r) \approx 0$ for large r and $f_l(r)$ is a smooth function of $\mathbf{r} \equiv (x, y, z)$. As shown in Section 2.1, substituting Eq. (2) into Eq. (1) leads to the Ewald

expression for potential, which consists of a short-range direct sum, long-range reciprocal sum, and a self-correction term. For example, the original Ewald splitting of $f(r) = 1/r$ is given by

$$\frac{1}{r} = \frac{\operatorname{erfc}(\beta r)}{r} + \frac{\operatorname{erf}(\beta r)}{r} \tag{3}$$

where $f_s(r) = \operatorname{erfc}(\beta r)/r$, $f_l(r) = \operatorname{erf}(\beta r)/r$, and β is an Ewald parameter controlling the distance in which $f_s(r) \approx 0$.

For non-periodic systems, a Fourier extension of $f_l(r)$ is expressed by

$$f_l(r) = \sum_{\mathbf{k}} \widetilde{f_l}(\mathbf{k}) \exp(i\mathbf{k}\cdot\mathbf{r}) \tag{4}$$

and computed using a technique described by Boyd [51] and generalized to three dimensions with DFT interpolation. The main requirement for rapid convergence is that $f_l(r)$ be smooth in $\mathbf{r} \equiv (x, y, z)$. As the Fourier extension coefficients $\widetilde{f_l}(\mathbf{k})$ are computed numerically, difficult analytic integrals are avoided, enabling efficient treatment of rectangular volumes and allowing greater flexibility in both the kernel and its splitting. As discussed below, the "derivative-matched" (DM) scheme leads to improved performance for $f(r) = 1/r$. In addition, an optimized grid storage algorithm for NPME is proposed resulting in a fourfold savings in total grid memory.

Given a radially symmetric kernel $f(r)$, the DM splitting defines the short-range $f_{\mathrm{DM},s}(r)$ and smooth long-range $f_{\mathrm{DM},l}(r)$ kernel components by

$$f_{\mathrm{DM},s}(r) \equiv \begin{cases} f(r) - \sum_{p=0}^{N_{\mathrm{der}}} a_p r^{2p} & r \le R_{\mathrm{dir}} \\ 0 & r > R_{\mathrm{dir}} \end{cases} \tag{5}$$

$$f_{\mathrm{DM},l}(r) \equiv \begin{cases} \sum_{p=0}^{N_{\mathrm{der}}} a_p r^{2p} & r \le R_{\mathrm{dir}} \\ f(r) & r > R_{\mathrm{dir}} \end{cases} \tag{6}$$

where R_{dir} is the direct sum distance cutoff controlling the range of $f_{\mathrm{DM},s}(r)$. The polynomial $\sum_p a_p r^{2p}$ is an even function of r and a smooth function of $\mathbf{r} \equiv (x, y, z)$. The coefficients a_p are chosen such that $f_{\mathrm{DM},l}(r)$ and its partial derivatives with respect to x, y, z are continuous up to order N_{der}. Compared to original Ewald splitting for $f(r) = 1/r$ in Eq. (3), this formulation improves performance by enabling faster evaluation of $f_{\mathrm{DM},s}(r)$ by 2x and reducing the grid size required for the Fourier extension by a factor of 2–3x.

The NPME method is implemented in a C++ application and library named `npme`. `npme` is optimized using vector intrinsic functions [52] and parallelized with OpenMP [53]. `npme` is implemented for the following pre-defined kernels: $f(r) = 1/r, r^{\alpha}, \exp(ik_0 r)/r$, where α is real and k_0 is complex. Users can also define custom radially symmetric kernels by providing C++ function classes, which can be used with general-purpose `npme` library routines. The current version of `npme` is optimized for static geometry with variable charges inputs.

`npme` is developed on randomized particle systems and benchmarked on example static geometry systems from computational chemistry and computational electromagnetics. For the electrostatic $f(r) = 1/r$ and dispersion $f(r) = 1/r^6$ kernels, `npme` is tested on non-periodic spherical water droplets and non-periodic elongated water tubes constructed with TIP3P [54] force field parameters obtained from example input files in the Tinker [22] molecular modeling package. For the Helmholtz $f(r) = \exp(ik_0 r)/r$ kernel, `npme` is benchmarked on systems with charges distributed over the surfaces of a sphere and NASA almond [55]. These benchmarks are representative of configurations used in a Method of Moments (MoM) [56, 57] implementation written by the author for calculating the radar cross sections (RCS) of a perfect electrical conductor (PEC). Performance is evaluated by comparing computational times for the $f(r) = 1/r$ and $f(r) = \exp(ik_0 r)/r$ kernels against results obtained by the Laplace and Helmholtz optimized routines of `fmm3D` [58, 59, 60, 61, 62].

The `npme` library for the $f(r) = \exp(ik_0 r)/r$ kernel is integrated into a MoM [56, 57] implementation to accelerate RCS computations involving PECs. For RCS problems of homogeneous materials, MoM [56] transforms a boundary value problem into a matrix equation by discretizing the object's surface and enforcing boundary conditions. For the PEC case, an incident electromagnetic wave induces a current on the object's surface. The induced surface current generates a scattered field outside the object, while the total fields inside the object are zero. The unknown surface current density is determined from boundary conditions for the tangential components of the electric and magnetic fields imposed by Maxwell's equations. When the MoM matrix equation is solved iteratively [63, 64], the matrix–vector product becomes the dominant computational cost. A variety of fast algorithms have been developed to accelerate this step, including the Helmholtz-based fast multipole method (FMM) [2, 65, 66, 67, 68, 69, 70, 71, 72, 73], FFT-accelerated conjugate gra-

dient methods [74, 75, 76, 77], and the Adaptive Integral Method (AIM) [78, 79, 80, 81, 82, 83]. These citations are representative and not intended to be exhaustive.

In this work, NPME is used to accelerate the MoM matrix-vector product by calculating electromagnetic fields arising from discretizing the surface current distribution. A technical description of the MoM formulation [84, 85, 86, 87, 88] and the hybrid MoM–NPME algorithm is provided in Section 4.1. The MoM–NPME method is used to compute the bistatic RCS of a 1-meter PEC sphere at 37.8 GHz and the monostatic RCS of the NASA almond [55] at 75 GHz. The bistatic RCS of the sphere is compared to analytic results computed with the Mie series solution [89]. The NASA almond results are validated by calculating the RCS at lower frequencies and comparing to recent measured data [90].

A summary of this work is given as follows: Section 2 describes the NPME method, including an overview of NPME (2.1), the Fourier extension procedure (2.2), kernel splitting properties (2.3), and an efficient grid storage algorithm (2.4). Section 3 presents `npme` benchmark results, including implementation details (3.1), a summary of computational methods (3.2), results for the $f(r) = 1/r, 1/r^6$ kernels (3.3), and results for the $f(r) = \exp(ik_0 r)/r$ kernel (3.4). Section 4 presents the RCS results computed with the MoM-NPME method, including an algorithm description and results for the PEC sphere and NASA almond. Conclusions are summarized in Section 5.

Appendices A - E are summarized as follows: Appendices A and B provide mathematical background on discrete Fourier transforms and B-spline properties, respectively. Appendix C contains a brief summary of the smooth PME formulation used in NPME, which closely follows the original periodic case described by Darden [4]. Appendix D presents the McMurchie-Davidson recursion [91], which is used to derive the DM splitting polynomial coefficients in Section 2.3. Appendix E provides a mathematical result required by the efficient grid storage algorithm introduced in Section 2.4.

The Supporting Information contains additional technical details and results. Section 2 explicitly lists the DM polynomial coefficients a_n for the $f(r) = 1/r$ kernel at $R_{dir} = 1.0$ for various orders $N_{der} = 3 - 8$. Section 3 provides an analysis of NPME errors. Section 4 presents additional RCS results for the NASA almond, computed by MoM–NPME and compared to recent measured data [90]. Section 5 introduces a reciprocal sum error

model (RSEM) using ideas borrowed from Refs. [92, 93]. The RSEM is mainly used as a tool inside the `npme` library for estimating grid size of predefined kernels and B-spline orders. Alternatively, users can also manually specify grid size. Section 6 presents a comparison of parallel efficiencies between `npme` and `fmm3D` for the $f(r) = 1/r$ kernel. Lastly, Section 7 briefly describes application of NPME to non-radially symmetric kernels.

2. Non-periodic particle mesh Ewald (NPME)

2.1. Overview of PME for non-periodic systems

Consider a rectangular box with dimension (X, Y, Z) centered at the origin containing N charges q_i with coordinates $\mathbf{r}_i$. The charges interact via a kernel function $f(r)$, which is split into short-range $f_s(r)$ and long-range $f_l(r)$ contributions. Substituting the kernel splitting in Eq. (2) into the expression for potential in Eq. (1) leads to the Ewald expression for potential ϕ_i

$$\phi_i = \phi_{i,dir} + \phi_{i,rec} + \phi_{i,self} \tag{7}$$

where

$$\phi_{i,dir} \equiv \sum_{j \neq i} f_s(r_{ij}) q_j \tag{8}$$

.

$$\phi_{i,rec} \equiv \sum_{j} f_l(r_{ij}) q_j \tag{9}$$

$$\phi_{i,self} \equiv -q_i \lim_{r \to 0} f_l(r) \tag{10}$$

The direct sum $\phi_{i,dir}$ and reciprocal sum $\phi_{i,rec}$ account for short-range and long-range interactions, respectively. The self-term $\phi_{i,self}$ corrects for including the $i = j$ term in the reciprocal sum.

The primary kernel splitting parameter used in this work is the direct space cutoff distance R_{dir}, in which $f_s(r) \approx 0$ and $f_l(r) \approx f(r)$ for $r > R_{\text{dir}}$. original. The direct sum then simplifies to

$$\phi_{i,dir} = \sum_{r_{ij} < R_{\text{dir}}} f_s(r_{ij}) q_j \tag{11}$$

For the $f(r) = 1/r$ kernel with original Ewald splitting in Eq. (3), the Ewald parameter β can be interchanged with R_{dir} by solving $\text{erfc}(\beta R_{\text{dir}})/R_{\text{dir}} = \text{tol}$ for some tolerance, e.g. $\text{tol} = 10^{-5}$. Unlike the periodic case, the zero total charge condition ($\sum_i q_i = 0$) is not required in the non-periodic case.

Consider the argument $f_l(r_{ij})q_j$ in the reciprocal sum. Note that $r_{ij} \equiv |\mathbf{r}_i - \mathbf{r}_j|$ is a coordinate vector within a rectangular volume centered at the origin with twice the lengths of the original volume, i.e. $(2X, 2Y, 2Z)$. In the following section, a procedure is described for computing a Fourier extension of $f_l(r)$ in Eq. (4) for $\mathbf{r}$ contained in the rectangular volume $(2X, 2Y, 2Z)$. A key assumption for constructing a Fourier extension is that $f_l(r)$ is a smooth function of $\mathbf{r} \equiv (x, y, z)$. In Appendix C, the reciprocal sum $\phi_{i,rec}$ for non-periodic systems is derived by substituting the Fourier extension of $f_l(r)$ into the smooth PME methodology.

The main splitting parameter R_{dir} is selected to minimize total computational cost. A smaller value of R_{dir} leads to a smaller direct sum cost. However, a smaller value of R_{dir} also causes additional oscillations in $f_l(r)$, which leads to a larger grid size and FFT cost.

2.2. Fourier extension of $f_l(r)$

The Fourier extension procedure for a one-dimensional function, as described by Boyd [51], is first briefly summarized and then generalized to three-dimensional functions using DFT interpolation. An efficient algorithm for constructing a Fourier extension of a smooth function $f_l(\mathbf{r}) \equiv f_l(r)$ is described below, where $\mathbf{r}$ is contained within a rectangular volume centered at the origin with dimensions $(2X, 2Y, 2Z)$. Note that radial symmetry of $f_l(\mathbf{r})$ is not required.

The goal is to construct an extended function $f_{l,\text{ext}}(\mathbf{r})$ that is identical to $f_l(\mathbf{r})$ within the rectangular volume of interest $(2X, 2Y, 2Z)$. In addition, $f_{l,\text{ext}}(\mathbf{r})$ is constructed such that both $f_{l,\text{ext}}(\mathbf{r})$ and its derivatives with respect to x, y, z satisfy periodic boundary conditions (p.b.c.) on the surface of a larger extended volume (L_1, L_2, L_3), where $L_1 > 2X$, $L_2 > 2Y$, and $L_3 > 2Z$. For example, $f_{l,\text{ext}}(\mathbf{r})$ satisfies p.b.c. in the x direction if

$$f_{l,\text{ext}}\left(-\frac{L_1}{2}, y, z\right) = f_{l,\text{ext}}\left(\frac{L_1}{2}, y, z\right) \tag{12}$$

for $|y| \leq L_2/2$ and $|z| \leq L_3/2$.

If the function is radially symmetric, i.e. $f_l(\mathbf{r}) = f_l(r)$, then $f_l(r)$ itself satisfies p.b.c. on the surface of the smaller volume $(2X, 2Y, 2Z)$, but its

partial derivatives do not. Consequently, applying Fourier series integration or DFT interpolation directly to $f_l(r)$ in the smaller volume leads to slow convergence of the Fourier extension coefficients $\widetilde{f_l}(\mathbf{k})$ due to Gibbs phenomena [94].

The procedure for calculating a Fourier extension of a smooth one-dimensional function is described by Boyd [51] as follows. The "ramp" function $H(x;a)$ is first defined as

$$H(x;a) \equiv \begin{cases} 0 & x < -1 \\ \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{ax}{\sqrt{1-x^2}}\right)\right) & -1 \le x \le 1 \\ 1 & 1 < x \end{cases} \tag{13}$$

Note that $H(x;a)$ has continuous derivatives of all orders for all x, i.e. $H(x;a)$ is smooth and infinitely flat [51] at the transitions points $x = \pm 1$. Suppose $[-X, X]$ is the interval of interest and $[-\Theta, \Theta]$ is an extended interval, where $\Theta > X$ and $L \equiv 2\Theta$. The "top hat" function is defined over $[-\Theta, \Theta]$ as

$$T(x;a,X,\Theta) \equiv \begin{cases} H\left(\frac{x+X+\Xi}{\Xi};a\right) & -\Theta \le x < -X \\ 1 & -X \le x \le X \\ H\left(\frac{-x+X+\Xi}{\Xi};a\right) & X < x \le \Theta \end{cases} \tag{14}$$

where $\Xi \equiv (\Theta - X)/2$. Θ and a are adjustable parameters with values defined below. Note that $T \equiv 1$ inside the interval of interest$[-X, X]$, but smoothly transitions to zero at the extended boundary defined by $x = \pm\Theta$.

When $T(x;a,X,\Theta)$ is multiplied by a smooth one-dimensional function $f_l(x)$, the resulting extended "non-overlapping" function defined by

$$f_{l,\text{ext,no}}(x) \equiv T(x;a,X,\Theta) f_l(x) \tag{15}$$

is identical to $f_l(x)$ inside $[-X, X]$, but smoothly transitions to zero in the "smoothing zones" given by $[-\Theta, -X]$ and $[X, \Theta]$. By construction, $f_{l,\text{ext,no}}(x)$ is smooth and zero at the boundary $x = \pm\Theta$, i.e. $f_{l,\text{ext,no}}(x)$ and its derivatives satisfy p.b.c. A rapidly converging Fourier extension of $f_l(x)$ can be obtained over the $[-\Theta, \Theta]$ interval by applying Fourier series integration or DFT interpolation to $f_{l,\text{ext,no}}(x)$. Note the smoothing zones of periodic replicas of $f_{l,\text{ext,no}}(x)$ do not overlap with one another.

As described above, forcing $f_l(x)$ to zero at the extended boundary is a method of imposing p.b.c. However, a more efficient method of imposing

p.b.c. is to allow the smoothing zones of adjacent periodic replicas to overlap with one another. The smoothing zone length is effectively doubled in size without increasing the period length $L = 2\Theta$. The net effect is a more rapidly converging Fourier extension. Letting $S(x) \equiv T(x; a, X, 2\Theta - X)$, the extended overlapping function is defined [51] by

$$f_{l,\text{ext,ov}}(x) \equiv \begin{cases} S(x)f_l(x) + S(x+2\Theta)f_l(x+2\Theta) & -\Theta \le x < -X \\ f_l(x) & -X \le x \le X \\ S(x)f_l(x) + S(x-2\Theta)f_l(x-2\Theta) & X < x \le \Theta \end{cases} \tag{16}$$

The Fourier extension coefficients $\widetilde{f_l}(k)$ in the one-dimensional case are obtained by taking the DFT of $f_{l,\text{ext,ov}}(x)$ evaluated on a grid.

Extending Eq. (16) to more than one dimension is straightforward but results in a lengthy expression. Instead, the above procedure is generalized to the three-dimensional case in the form of a practical algorithm for computing $f_{l,\text{ext,ov}}(\mathbf{r})$ on a grid as follows. Suppose the grid sizes (N_1, N_2, N_3) are even positive integers. Define a one dimensional DFT array $x_1[N_1]$ in the x direction by

$$x_1[p] \equiv \frac{L_1}{N_1} \begin{cases} p & 0 \le p \le \frac{N_1}{2} - 1 \\ p - N_1 & \frac{N_1}{2} \le p \le N_1 - 1 \end{cases} \tag{17}$$

for $0 \le p \le N_1 - 1$ and where

$$L_1 \equiv 2\Theta_1 \tag{18}$$

Define a second array $x_2[N_1]$ by

$$x_2[p] \equiv \begin{cases} x_1[p] + 2\Theta_1 & -\Theta_1 \le x_1[p] < -X \\ 0 & -X \le x_1[p] \le X \\ x_1[p] - 2\Theta_1 & X < x_1[p] \le \Theta_1 \end{cases} \tag{19}$$

The top hat arrays $T_{x,1}[N_1]$ and $T_{x,2}[N_1]$ are defined by

$$T_{x,1}[p] \equiv T(x_1[p]; a, X, 2\Theta_1 - X) \tag{20}$$

$$T_{x,2}[p] \equiv \begin{cases} T(x_2[p]; a, X, 2\Theta_1 - X) & -\Theta_1 \le x_1[p] < -X \\ 0 & -X \le x_1[p] \le X \\ T(x_2[p]; a, X, 2\Theta_1 - X) & X < x_1[p] \le \Theta_1 \end{cases} \tag{21}$$

Arrays for the y and z directions are defined in a similar manner. A point on the DFT grid is defined by $(x_1[p], y_1[q], z_1[r])$ where $0 \leq q \leq N_2 - 1$ and $0 \leq r \leq N_3 - 1$. The value of $f_{l,\text{ext,ov}}(\mathbf{r})$ at this grid point can be computed as

$$f_{l,\text{ext,ov}}[p,q,r] \equiv \sum_{i,j,k=1}^{2} T_{x,i}[p]T_{y,j}[r]T_{z,k}[s]f_l\left(x_i[p], y_j[q], z_k[r]\right) \tag{22}$$

The Fourier extension coefficients $\widetilde{f_l}(\mathbf{k})$ in Eq. (4) are obtained by applying a DFT to $f_{l,\text{ext,ov}}[p,q,r]$. The two adjustable Fourier extension parameters are taken to be $a \equiv 4.0$ and

$$\begin{aligned} \Theta_1 &\equiv X + R_{\text{dir}} \\ \Theta_2 &\equiv Y + R_{\text{dir}} \\ \Theta_3 &\equiv Z + R_{\text{dir}} \end{aligned} \tag{23}$$

and were determined by minimizing the reciprocal sum error of several randomized particle computations.

An efficient grid storage algorithm is described below in Section 2.4. To apply this algorithm, a correction due to the B-spline support must be made to the rectangular volume containing the point charges. Let (X_0, Y_0, Z_0) be the dimensions of the smallest rectangular volume enclosing the point charges. Starting from assumptions described in Section 2.4, an expression for the $X \geq X_0$ correction is derived in Appendix E with analogous results for the y and z directions.

2.3. Kernel splitting

2.3.1. Properties

Methods for splitting the kernel $f(r)$ into short $f_s(r)$ and long-range $f_l(r)$ contributions are described as follows. The kernel splitting should satisfy the following properties:

1. $f_s(r)$ can contain a singularity, but $f_s(r) \approx 0$ for $r > R_{\text{dir}}$
2. $f_l(r) \approx f(r)$ for $r > R_{\text{dir}}$
3. $f_l(r)$ is an even smooth function of $\mathbf{r} \equiv (x, y, z)$

Property 1 is required for efficient evaluation of the direct sum, while property 2 is the complement of property 1. Property 3 is required for rapid convergence of Fourier extension coefficients.

For radially symmetric kernels, it is not sufficient that $f_l(r)$ is a smooth function of r. For example, both $f_l(r) \equiv r$ and $f_l(r) \equiv \text{erf}(r)$ are smooth functions of r, but not of $\mathbf{r} \equiv (x, y, z)$ as the partial derivatives of $r = \sqrt{x^2 + y^2 + z^2}$ do not exist at the origin. Consequently, the Fourier extensions of both examples converge very slowly with grid size. On the other hand, both $f_l(r) \equiv r^2$ and $f_l(r) \equiv \text{erf}(r)/r$ are even and smooth functions of r with rapidly converging Fourier extensions. It follows that if $f_l(r)$ is a smooth radially symmetric and even function of r, then $f_l(r)$ is a smooth function of $\mathbf{r} \equiv (x, y, z)$.

2.3.2. Derivative-matched (DM) splitting

In order to obtain a rapidly converging Fourier extension, it is found that $f_l(r)$ need only be smooth up to a sufficiently high order. For example, the Fourier extension of $f_l(r) = r^3$, which has continuous first derivatives at the origin, converges more quickly with grid size than $f_l(r) = r$, which is continuous only. This observation motivates the derivative-matched (DM) splitting for radially symmetric kernels $f(r)$ as defined by Eqs. 5 and 6. The $N_{\text{der}} + 1$ coefficients are chosen such that $f_{\text{DM},l}(r)$ and its partial derivatives up to order N_{der} are continuous at $r = R_{\text{dir}}$. This condition is ensured by:

$$\left\{\left(\frac{1}{r}\frac{d}{dr}\right)^n \sum_{p=0}^{N_{\text{der}}} a_p r^{2p}\right\}\Bigg|_{r=R_{\text{dir}}} = \left\{\left(\frac{1}{r}\frac{d}{dr}\right)^n f(r)\right\}\Bigg|_{r=R_{\text{dir}}} \tag{24}$$

for $n = 0, 1, ..N_{\text{der}}$. A proof of this statement follows from the McMurchie-Davidson recursion [91] for calculating x, y, z partial derivatives of a radially symmetric function, which is briefly summarized in Appendix D. Note that Eq. (24) is a set of $N_{\text{der}} + 1$ linear equations with an upper triangular matrix for the coefficients a_p with solution

$$a_{N-p} = \frac{1}{C_{N-p,N-p}}\left(f^{(N-p)}(R_{\text{dir}}) - \sum_{q=0}^{p-1} C_{N-p,N-q} a_{N-q}\right) \tag{25}$$

which can solved recursively for $p = 0, 1, ..N_{\text{der}}$ where

$$f^{(n)}(r) \equiv \left(\frac{1}{r}\frac{d}{dr}\right)^n f(r) \tag{26}$$

$$C_{p,q} \equiv 2^p \frac{q!}{(q-p)!} R_{\text{dir}}^{2(q-p)} \tag{27}$$

The DM splitting method can be applied to radially symmetric kernels $f(r)$, which have radial derivatives at $r = R_{\text{dir}}$ and are smooth for $r \geq R_{\text{dir}}$.

A recursion for calculating the radial derivatives of the $f(r) = r^{\alpha}$ kernel for real α is given by

$$f^{(n)}(r) = \frac{\alpha - 2n + 2}{r^2} f^{(n-1)}(r) \tag{28}$$

The radial derivatives for the $f(r) = \exp(ik_0 r)/r$ kernel for complex k_0 can be expressed in terms of spherical Hankel functions of the first kind [94]. The recursion for spherical Hankel functions can be transformed into the following recursion for $f^{(n)}(r)$

$$f^{(n)} = -\frac{2n-1}{r^2} f^{(n-1)} - \left(\frac{k_0}{r}\right)^2 f^{(n-2)} \tag{29}$$

which is valid for $k_0 \to 0$.

For the static kernel $f(r) = r^{\alpha}$, the coefficients a_n as a function of R_{dir} follow a dimensional analysis relationship given by

$$a_n(R_{\text{dir}}) = a_n(R_{\text{dir}} = 1.0) R_{\text{dir}}^{\alpha - 2n} \tag{30}$$

for a fixed order N_{der}. For example, the coefficients for $f(r) = 1/r$ ($\alpha = -1.0$) with $N_{\text{der}} = 4$ and $R_{\text{dir}} = 1.0$ are given in Table 1. The corresponding coefficients for arbitrary R_{dir} can be found by scaling the coefficients in Table 1 by $1/R_{\text{dir}}^{2n+1}$. For other polynomial orders $N_{\text{der}} = 3 - 8$, the coefficients for the $f(r) = 1/r$ kernel at $R_{\text{dir}} = 1.0$ are provided in Section 2 of the Supporting Information. In addition, `npme` prints the DM coefficients in a .log file by default. The calculation of the DM polynomial is numerically stable: while the coefficients a_n may become arbitrarily large for $R_{\text{dir}} < 1.0$ or small for $R_{\text{dir}} > 1.0$, the product $a_n r^{2n}$ remains $O(r^{\alpha})$ for $r \sim R_{\text{dir}}$.

2.3.3. DM splitting reciprocal sum performance

Consider the $f(r) = 1/r$ kernel case. Note the direct sum truncation error for DM splitting is exactly zero, as $f_{\text{DM},s}(r) \equiv 0$ for $r > R_{\text{dir}}$. Therefore, the total error for DM splitting is equal to the reciprocal sum error. In contrast, original Ewald splitting has a non-zero direct sum truncation error. In order to fairly compare the reciprocal sum errors for both methods, the direct sum truncation error should be less than or equal to the reciprocal sum

n	a_n
0	2.460937500
1	-3.281250000
2	2.953125000
3	-1.406250000
4	0.2734375000

Table 1: DM polynomial coefficients for the $f(r) = 1/r$ kernel with $N_{\text{der}} = 4$ at $R_{\text{dir}} = 1.0$.

error for original Ewald splitting. A numerical test comparing reciprocal sum errors of DM and original Ewald splitting is described below.

For the $f(r) = 1/r$ kernel, the reciprocal sum error for DM splitting is compared to that of the original Ewald splitting for a system of 10^5 random particles contained within a cube with length $X_0 = 100.0$ and direct sum cutoff radius $R_{\text{dir}} = 5.0$. For B-spline orders $n = 4, 6, 8, 16$ and target tolerances tol $= 10^{-4}, 10^{-5}, ..10^{-13}$, grid sizes and N_{der} parameters are determined using a model for the reciprocal sum error, as described in the Supporting Information. In Figure 1, the reciprocal sum error for DM splitting is plotted as a function of grid size. Using the same grid sizes and B-spline orders, the reciprocal sum errors for the original Ewald splitting method is also plotted with β values determined by solving $\text{erfc}(\beta R_{\text{dir}})/R_{\text{dir}} = \text{tol}$ for each target tolerance. In most cases, the reciprocal sum errors are substantially lower for the DM splitting as compared to the original method. For the $n = 16$ case, the errors are approximately equal at lower precision ($> 10^{-7}$), but at higher precision ($< 10^{-8}$), the DM method is substantially more accurate. For a given target tolerance, the DM splitting method typically results in an error reduction of up to $10x$, or equivalently, a $2 - 3x$ reduction in required grid size.

Note that $f_{\text{original},l}(r) = \text{erf}(\beta r)/r$ is infinitely smooth in $\mathbf{r}$, while $f_{\text{DM},l}(r)$ is smooth only up to order N_{der}. One might initially assume that $f_{\text{original},l}(r)$ should have smaller reciprocal sum errors when compared to $f_{\text{DM},l}(r)$. This would indeed be true if β were held fixed. However, for fixed R_{dir}, β must increase with grid size or target precision in order to ensure a fair comparison of total error between both splitting methods. As β increases, $f_{\text{original},l}(r)$ becomes more oscillatory, which reduces the convergence rate of its Fourier extension.

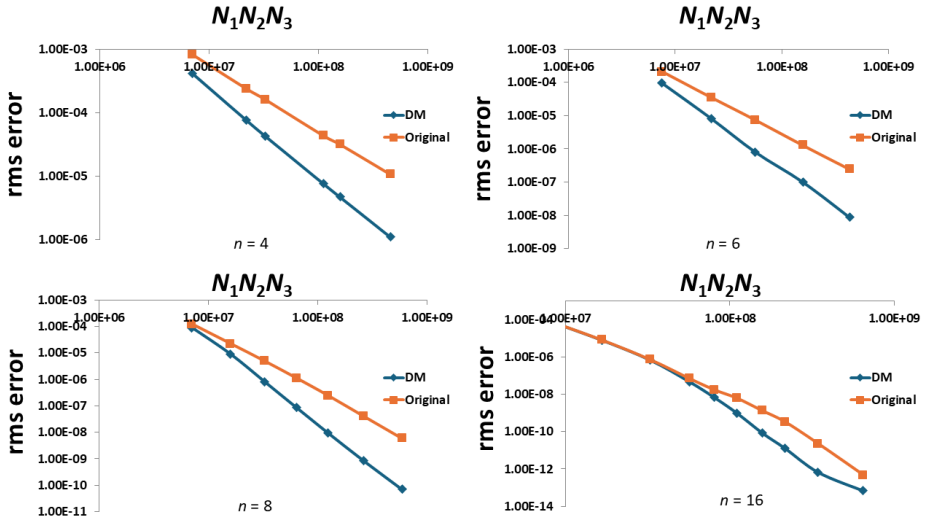

Figure 1: Rms error in reciprocal sum potential as a function of number of grid points $N_1N_2N_3$ comparing DM and original Ewald splitting results for the $f(r) = \frac{1}{r}$ kernel with various B-spline orders: $n = 4$ (top left), 6 (top right), 8 (bottom left), and 16 (bottom right).

2.3.4. DM splitting direct sum performance

DM splitting also exhibits improved direct sum computational properties. For the $f(r) = 1/r$ kernel, computing the short-range kernel $f_{\text{original},s}(r) = \text{erfc}(\beta r)/r$ and its gradient for the original Ewald splitting requires evaluating real transcendental and algebraic functions: `sqrt(x)`, `erfc(x)`, and `exp(x)`. In contrast, computing $f_{\text{DM},s}(r)$ requires only `sqrt(x)` and a polynomial of order N_{der}. The computational cost of evaluating $f_{\text{original},s}(r)$ exceeds that of $f_{\text{DM},s}(r)$ if both routines are optimized using vector intrinsic functions. Specifically, the relative cost of computing $f_{\text{original},s}(r)$ compared to $f_{\text{DM},s}(r)$ is approximately: $2.9x$ for $N_{\text{der}} = 4$, $2.5x$ for $N_{\text{der}} = 8$, and $1.6x$ for $N_{\text{der}} = 16$. As shown in Section 3.1, the direct sum computational time for the original Ewald method is approximately $2x$ longer than that of the DM approach for the $f(r) = 1/r$ kernel with $N_{\text{der}} = 8$.

The DM splitting for the complex kernel $f(r) = \exp(ik_0r)/r$ also has significant performance advantages. Note that the evaluation of $f(r) = \exp(ik_0r)/r$ for complex k_0 can be separated into real and imaginary parts, each expressed in terms of elementary real transcendental and algebraic functions: `sqrt(x)`, `cos(x)`, `sin(x)`, and `exp(x)`. Thus, the real and imaginary parts for $f_{\text{DM},s}(r)$ can be evaluated separately with a polynomial and

elementary real transcendental and algebraic functions. This result is significant because high-performance vectorized math libraries are available for elementary real transcendental and algebraic functions [52], but are usually not available for complex variable transcendental functions.

2.4. Efficient grid storage algorithm

The full DFT grid used for the reciprocal sum has total size $N_1N_2N_3$. For radially symmetric kernels, storage can be substantially reduced by approximately $4x$ using the following algorithm. Recall the particles are contained within a rectangular volume with dimensions (X_0, Y_0, Z_0) and the Fourier extension for $f_l(r)$ must be computed within a larger volume $(2X, 2Y, 2Z)$, where $X \geq X_0$, $Y \geq Y_0$, and $Z \geq Z_0$ are the B-spline corrected lengths derived in Appendix E. As described in more detail below, the Fourier extension coefficients of a radially symmetric kernel $f_l(r)$ also satisfy radial symmetry and can be stored on grids of approximate size $N_1N_2N_3/8$.

The NPME reciprocal sum algorithm is formulated such that all necessary quantities can be computed on grids with size $N_1N_2N_3/8$. To demonstrate this, first define the following index conventions. For each dimension $i = 1, 2, 3$, define a "full grid" with size N_i and indexes $p_i = -N_i/2, -N_i/2 + 1, ..N_i/2 - 1$. Certain quantities are non-zero only on a "small grid" with size $N_i/2$ and indexes $p_i = -N_i/4, -N_i/4 + 1, ..N_i/4 - 1$. The grid sizes N_i are chosen to be multiples of 4, such that $N_i/4$ is always an integer. Note that the small grid has a total size $N_1N_2N_3/8$ in three dimensions.

In the first step of the NPME reciprocal sum algorithm, the real-space charge interpolation array $Q_{p_1p_2p_3}$ in Eq. (C.4) is evaluated. Since the particles are contained within a smaller volume, most entries for $Q_{p_1p_2p_3}$ are zero on the full grid. Therefore, $Q_{p_1p_2p_3}$ can be stored in the small grid. Similarly, in the last steps of the NPME reciprocal sum algorithm for calculating potential in Eq. (C.7) and potential gradient in Eq. (C.8), $C^i_{p_1p_2p_3}$ in Eq. (B.11) is non-zero on the small grid for the same reason. Although both $\theta_{p_1p_2p_3}$ in Eq. (C.6) and $\widetilde{\theta}_{p_1p_2p_3}$ are non-zero in the full grid, only the values in the small grid are needed. Thus, $\widetilde{\theta}_{p_1p_2p_3}$ can be stored in the small grid.

Although the intermediate NPME reciprocal sum steps must be performed on the full grid, the required computations can be performed independently on eight small grids. The algorithm is described as follows.

After calculating and storing $Q_{p_1p_2p_3}$ on a small grid, the DFT of $\widetilde{Q}_{m_1m_2m_3}$ on the full grid is given by

$$\widetilde{Q}_{m_1m_2m_3} = \frac{1}{N_1N_2N_3} \sum_{p_1=-\frac{N_1}{4}}^{\frac{N_1}{4}-1} \sum_{p_2=-\frac{N_2}{4}}^{\frac{N_2}{4}-1} \sum_{p_3=-\frac{N_3}{4}}^{\frac{N_3}{4}-1} Q_{p_1p_2p_3} \omega_1^{-m_1p_1} \omega_2^{-m_2p_2} \omega_3^{-m_3p_3} \tag{31}$$

where $\omega_i \equiv \exp(2\pi i/N_i)$. Note the summation indexes p_i are restricted to small grid values, while the m_i indexes range span the full grid.

The quantity $\widetilde{Q}_{m_1m_2m_3}$ on the full grid can be computed independently as eight separate smaller grid contributions corresponding to even/odd terms in each dimension. For example, it follows directly from Eq. (31) that the even-odd-even terms can be computed as the following small grid DFT

$$\widetilde{Q}_{2m_1,2m_2+1,2m_3} = \frac{1}{8} \underset{\text{small}}{\text{DFT}} \left\{ Q_{p_1p_2p_3} \omega_2^{-p_2} \right\} \tag{32}$$

Each of the eight small grid contributions to $\widetilde{Q}_{m_1m_2m_3}$ are used to compute corresponding contributions to $\theta_{p_1p_2p_3}$ with Eq. (C.6). The DFT of each contribution to $\theta_{p_1p_2p_3}$ is computed on a small grid, truncated, and added to a main small grid array for $\widetilde{\theta}_{p_1p_2p_3}$, which is then used to compute potential in Eq. (C.7) and potential gradient in Eq. (C.8). The entire procedure uses small grids only and is summarized in Algorithm 1.

For radially symmetric kernels, the Fourier extension of $f_l(r)$ satisfies the symmetry

$$f_{l,ext}(\pm x_p, \pm y_q, \pm z_r) = f_{l,ext}(x_p, y_q, z_r) \tag{33}$$

where $x_p = pL_1/N_1$ for $p = -N_1/2, -N_1/2+1, ..N_1/2-1$ with similar definitions for y_q and z_r. As the Fourier extension coefficients $\widetilde{f_{l,ext}}(\mathbf{k})$ also satisfy the same symmetry, $\widetilde{f_{l,ext}}(\mathbf{k})$ can be stored on a reduced grid of size $(N_1/2+1)(N_2/2+1)(N_3/2+1)$. Note the three dimensional DFT of $f_{l,\text{ext}}(\mathbf{r})$ on the full grid can be computed by several one dimensional DFT's in order to explicitly avoid storing the full grid array. The algorithm for this step is straightforward and any additional minor overhead is mitigated by the fact that this step need only be performed once during setup.

An algorithm involving full grids, each with size $N_1N_2N_3$, only requires two arrays: one for $\widetilde{f_{l,ext}}(\mathbf{k})$ and one for $Q_{p_1p_2p_3}$, which can be overwritten

Algorithm 1 Efficient Grid Storage

1. Let one array with label A have size $(N_1/2+1)(N_2/2+1)(N_3/2+1)$ $\approx N_1N_2N_3/8$ and three other arrays B, C, D have size $N_1N_2N_3/8$
2. compute $\widetilde{f_l}(\mathbf{k})$ once and store in A
3. compute $Q_{p_1p_2p_3}$ and store in B
4. Zero array C representing $\widetilde{\theta}_{p_1p_2p_3}$
5. For each of the 8 even/odd contributions
   (a) Copy array $Q_{p_1p_2p_3}$ into array D and multiply by $\omega_i^{-p_i}$ for any odd contribution
   (b) Take the DFT of array D and multiply by 1/8
   (c) Scale array D by $\widetilde{f_l}(\mathbf{k})$ and other scaling factors in Eq. (C.6)
   (d) Take the DFT of array D, truncate to small grid values, and add to array C
6. compute the reciprocal sum potential Eq. (C.7) and potential gradient Eq. (C.8) using array C.

with $\widetilde{\theta}_{p_1p_2p_3}$. Therefore, the total storage for a full grid algorithm is $2N_1N_2N_3$. In comparison, the efficient grid storage algorithm requires 4 arrays of approximate size $N_1N_2N_3/8$, which leads to a memory savings of $\approx 4x$.

Recall that the particles are contained within a rectangular volume with minimum dimensions (X_0, Y_0, Z_0). In order for the efficient grid algorithm described above to be valid, the lengths of the rectangular box may require a small correction due to the B-spline support extending outside the smallest box containing the particles. The $X_0 \to X$ correction is derived in Appendix E and is given by

$$X = \max\left\{X_0, X_0\frac{N_1}{N_1-2(n+1)} - \frac{\delta}{2}\frac{N_1-4(n+1)}{N_1-2(n+1)}\right\} \tag{34}$$

where $\delta \equiv \Theta - X = R_{\text{dir}}$ is the smoothing zone length, n is the B-spline order, and N_1 is the grid size in the x direction. Similar results hold for the y and z directions. In addition, the particle positions are translated such that $\max(x_i) = X/2, \max(y_i) = Y/2$, and $\max(z_i) = Z/2$.

3. NPME benchmark systems

3.1. Implementation details

The NPME algorithm is implemented in a C/C++ library called `npme`, which also includes optimized $O(N^2)$ direct pairwise computations for the potential and its gradient. The library uses double-precision arithmetic, is parallelized with OpenMP, and is explicitly vectorized with AVX and AVX-512 intrinsics [52]. FFTs, matrix operations, and random number generation are performed with the Intel Math Kernel Library (MKL) [95].

The library includes a setup function that handles memory allocation, Fourier extension coefficient computation, table generation for the direct and reciprocal sums, and internal optimizations to reduce runtime at the expense of longer setup time. The setup typically costs $3 - 4x$ more than a single potential or gradient evaluation, and has not been tuned for speed. The current implementation is optimized for rigid geometries with variable charge inputs.

`npme` supports derivative-matched (DM) splitting for the kernels $f(r) = 1/r, r^\alpha$, and $\exp(ik_0 r)/r$, where α is real and k_0 is complex. The $f(r) = 1/r$ kernel is also implemented with the original Ewald splitting for comparison. Users can define custom radially symmetric kernels by implementing C++ classes that satisfy the splitting conditions described in Section 2.3.1. Each class defines the kernel and its derivatives for array inputs `x[K], y[K], z[K]` and outputs `f[K], dfdx[K], dfdy[K], dfdz[K]` for any positive integer `K`. For improved performance, vectorized versions of these functions can be defined for `K` a multiple of 4 (AVX) or 8 (AVX-512). Additional details are provided in the `npme` user manual.

The library distinguishes between real and complex kernel types. Inputs include particle coordinates and keyword parameters. For real kernels, the potential routine accepts N real-valued charges and returns real-valued potential and its gradient outputs. For complex kernels, both input charges and output potential and its gradient values are complex.

The direct sum algorithm partitions particles into cells of size R_{dir}/n, where $n = 1, 2, 3$, or 4, and computes n-nearest neighbor cell-cell interactions. To reduce communication overhead, cells are grouped into clusters; each thread processes one cluster–cluster interaction, which includes all corresponding cell–cell interactions. Cell and cluster data are stored in compact tables, and particle indices are permuted so that particles in the same cell are stored contiguously. Only starting indices and counts are

Tolerance	B-spline order n
$10^{-3} - 10^{-4}$	4 or 6
$10^{-5} - 10^{-6}$	6 or 8
$10^{-7} - 10^{-10}$	8 or 16
$< 10^{-10}$	16

Table 2: Suggested B-spline orders for various target tolerances based on empirical accuracy and runtime tradeoffs

stored. The permutation is done in-place using "k-cycle decompositions", a technique from elementary symmetric group theory [96]. For large systems (e.g., $\approx 10^9$ particles), this strategy saves substantial memory-over 64 GB for a single copy of the input/output arrays.

The reciprocal sum consists of two main steps: (1) computing the $Q_{p_1p_2p_3}$ array, and (2) evaluating the potential and gradient from the $\widetilde{\theta}_{p_1p_2p_3}$ array. Each step is block-partitioned along grid dimensions n_1, n_2, n_3 to improve data locality. The block sizes are chosen so that $N_i/2$ is divisible by n_i, where N_i is the FFT grid size along direction i. Threads process one block at a time and access only the particles associated with that block. FFT grid sizes N_i must be divisible by 4. B-spline orders n can range from 2 to 40 (even), with the constraint $n < N_i/4$ for $i = 1, 2, 3$.

The user must specify R_{dir} and B-spline order n, both of which strongly affect performance. Suggested B-spline orders for various error tolerances are listed in Table 2. While larger n increases reciprocal sum cost $O(Nn^3)$, it reduces grid size and storage requirements, as shown in Supporting Information Section 2.

Grid and block sizes can be estimated automatically using the reciprocal sum error model (RSEM) described in Supporting Information Section 4, for predefined kernels and supported spline orders. Alternatively, users may manually specify these values for any valid kernel and B-spline combination.

3.2. Computational details

All computations were performed on a dual-socket 2×16-core Xeon 4216 system (2.1/3.2 GHz), with 32 physical cores, 64 threads, and 512 GB of DDR4 memory. `npme` was compiled with Intel C/C++ compiler version 19.1.1.217 [95]. For the Laplace kernel $f(r) = 1/r$ and Helmholtz kernel $f(r) = \exp(ik_0r)/r$, computation times are compared to those obtained with

`fmm3D` [58, 59, 60, 61, 62]. `fmm3D` was compiled with GNU Fortran [97] using the "`FAST_KER=ON`" option.

The total `fmm3D` time includes the potential and gradient computation, and presumably a setup cost, as the C interface does not allow separate timing of these components. For both `npme` and `fmm3D`, input/output overhead (reading charges and coordinates from file) is excluded.

The `npme` setup cost is generally negligible in applications with static geometries. It is performed once during initialization and is not optimized. For example, in the bistatic RCS simulation of a sphere in Section 4, the setup is called once, while the main evaluation routine is called 316 times.

3.3. Results for $f(r) = 1/r$ and $f(r) = 1/r^6$

3.3.1. Water droplets and water tubes

The $f(r) = 1/r$ and $f(r) = 1/r^6$ kernels in `npme` are used to compute intermolecular electrostatic and dispersion energies for non-periodic molecular systems. Each atom is assigned a charge, and the electrostatic potential and its gradient are computed for each atom. The long-range intermolecular dispersion energy U_{disp} is modeled as

$$U_{disp} = -\sum_{i<j} \frac{C_{ij}}{r_{ij}^6} \tag{35}$$

where $C_{ij} \geq 0$ is a dispersion coefficient between atoms i and j. C_{ij} is computed as a function of "atom type" parameters C_{ii} and C_{ij} using combination rules [98]. If a geometric combination rule is used, i.e. $C_{ij} = \sqrt{C_{ii}C_{jj}}$, then U_{disp} and its gradient can be computed from the potential and its gradient arising from "dispersion charges" $q_i \equiv \sqrt{C_{ii}}$ interacting via a $f(r) = 1/r^6$ kernel.

The molecular test systems studied here include water droplets and water tubes, illustrated [99] in Figure 3. These systems are generated from a periodic cubic unit cell (length 24.662Å) containing 500 water molecules. The unit cell, taken from the "waterbox.xyz" example input file in the Tinker molecular modeling distribution [22], is replicated and translated in three dimensions to form a finite truncated crystal. A water droplet is constructed by selecting only those molecules whose centers of mass fall within a given droplet radius. Droplet radii range from 29.0Å to 1337.0Å, corresponding to systems of approximately $N \approx 10^4$ to $N \approx 10^9$ atoms. The internal structure of the droplet is preserved from the periodic unit

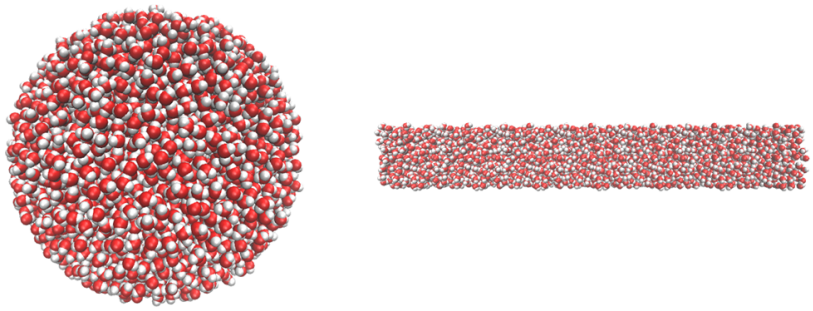

Figure 2: Water droplet and water tube structures used in testing, visualized using VMD [99]

cell, i.e. non-periodic molecular dynamics or geometry optimization is not performed. The potential and its gradient are computed on these static geometries. Water tubes are generated similarly, but by replicating the unit cell in one dimension only. Tube dimensions are $(X, 24.662, 24.662)$, with X ranging from 172Å to $1.64 \cdot 10^6$Å, corresponding to $N \equiv 10^4$ to $N \equiv 10^8$ atoms.

Atomic charges and dispersion parameters are taken from the TIP3P water model [54]. For the electrostatic kernel $f(r) = 1/r$, oxygen and hydrogen atoms are assigned charges of $q_O = -0.834e$ and $q_H = +0.417e$, respectively. The oxygen–oxygen dispersion parameter is $C_{OO} = 595.0$ kcal/mol/Å^6, while C_{OH} and C_{HH} are zero in TIP3P.

3.3.2. Electrostatic and molecular dispersion computation times

The `npme` library is benchmarked on the water droplet and tube systems using the electrostatic $f(r) = 1/r$ and dispersion $f(r) = 1/r^6$ kernels with DM splitting. Figure 3 shows the time required to compute the NPME potential and its gradient as a function of the number of charges N for both electrostatic (top left) and dispersion (top right) kernels. Computations are performed using B-spline order $n = 8$ and a target precision of 10^{-5}. For comparison, pairwise summation results using $O(N^2)$ scaling are also shown. NPME outperforms direct summation for $N \gtrsim 2 \cdot 10^4$. Grid storage (GB) is also shown in Figure 3 for the electrostatic (bottom left) and dispersion (bottom right) kernels. Timings for both kernels are similar, while the dispersion kernel requires 1.5–2x less memory. For the electrostatic kernel, optimal R_{dir} values range from 11 Å at $N \approx 10^4$ to 13 Å at $N \approx 10^9$. For the dispersion kernel, R_{dir} ranges from 9–11 Å. The largest system, a

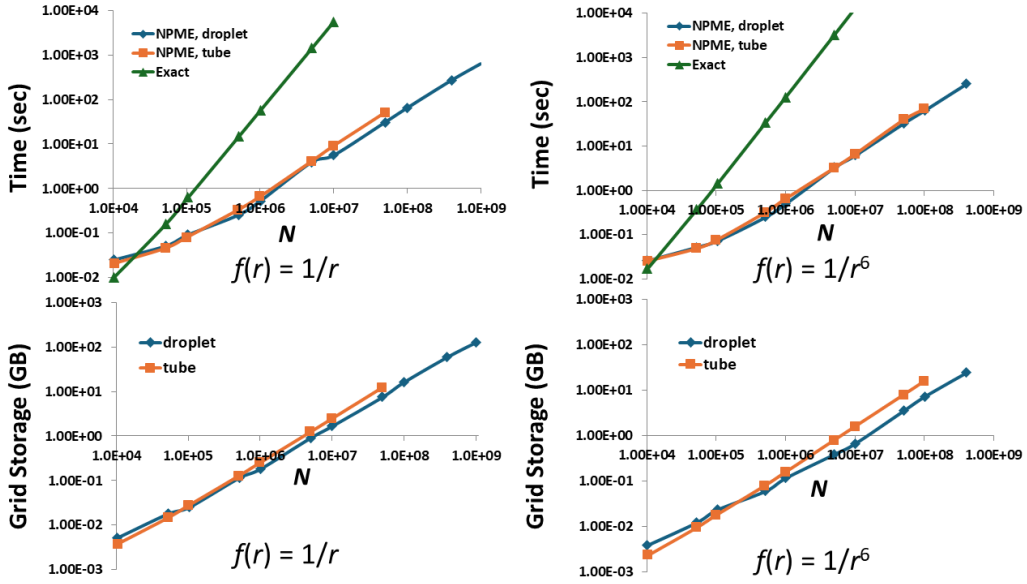

Figure 3: Water droplets and water tubes computation time (top) and grid storage (bottom) for the electrostatic (left) and dispersion (right) kernels at the 10^{-5} precision level

.

water droplet with $\approx 10^9$ charges, required 632 seconds and 126 GB of grid memory.

The effect of varying the B-spline order and precision is examined for the electrostatic kernel applied to water droplets with $N \approx 10^4$–10^7 charges. On average, B-spline order $n = 6$ reduced computation time by ≈10% but increased grid storage by ≈50% compared to $n = 8$ at a 10^{-5} precision level. For fixed B-spline order $n = 8$, increasing precision from 10^{-5} to 10^{-8} led to a ≈2x increase in computation time and a ≈5.2x increase in grid storage. At higher precision levels ($< 10^{-8}$), grid storage can be significantly reduced by increasing the B-spline order from $n = 8$ to $n = 16$. Additional results on precision, grid size, and B-spline order are presented in Section 2 of the Supporting Information.

The direct sum computational times for the electrostatic kernel using DM splitting are compared to those of the original Ewald splitting. For a fixed $R_{\text{dir}} = 9$Å and $N_{\text{der}} = 8$, the timings are reported in Table 3 for water droplets with $N \approx 10^4$ to 10^8 atoms or charges. Both routines are optimized using vector intrinsic functions with similar implementation details. On average, the original Ewald direct sum is approximately $2x$ slower than

N	Original	DM
10,206	1.12E-02	5.63E-03
104,718	5.36E-02	2.17E-02
1,007,976	4.18E-01	2.20E-01
10,006,569	4.40E+00	2.27E+00
100,316,031	4.62E+01	2.46E+01

Table 3: Water droplet direct sum times (sec) for both original Ewald and DM short-range kernels as a function of the number of charges or atoms N at the 10^{-5} precision level

.

N	`npme` (total)	`npme` (compute only)	`fmm3D` (total)
10^4	0.0346	0.025	0.048
10^5	0.467	0.091	0.235
10^6	3.245	0.525	1.70
10^7	30.0	5.56	18.36

Table 4: Comparison of compute times (seconds) for `npme` and `fmm3D` on spherical water droplets with N particles. The `npme` times are reported both with (total) and without (compute only) the one-time setup cost. All potential and its gradient calculations were performed using 64 threads and a precision level of 10^{-5}.

the DM version, consistent with the relative computational costs of $f_{\mathrm{DM},s}(r)$ and $f_{\mathrm{original},s}(r)$ discussed in Section 2.3.

The `npme` computation times for the electrostatic kernel are compared to those obtained with the Laplace routine of `fmm3D` for water droplets at the 10^{-5} precision level and shown in Table 4. On average, the `fmm3D` total runtimes are $2.8x$ longer than the `npme` computation time for the electrostatic kernel. If the `npme` setup cost is included in the total runtime, the average total `npme` time becomes approximately $2.0x$ longer than that of `fmm3D`.

The parallel efficiency of `fmm3D` is consistently higher than that of `npme` across all tested system sizes (see Supporting Information, Table 9). For instance, at $N = 10^6$, `npme` achieves 62% efficiency compared to 70% for `fmm3D`; at $N = 10^7$, the efficiencies are 58% and 76%, respectively. While `npme` shows somewhat lower scalability on shared-memory systems, this is partly due to the use of older OpenMP scheduling strategies and a lack of fine-grained control over memory locality. Although FFT grids are allocated using aligned memory, more advanced memory placement

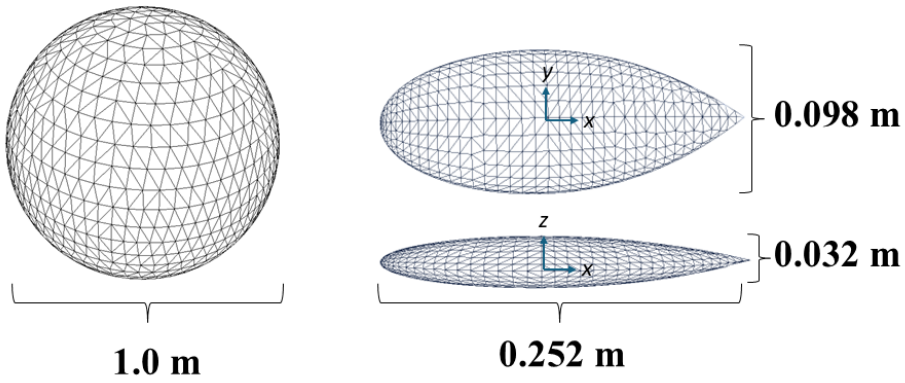

Figure 4: Sphere (left) and NASA almond (right), visualized using Meshlab [100]

and thread binding features available in newer OpenMP versions were not used in the current implementation. Despite this, `npme` achieves favorable compute times for large-scale problems due to its compact memory layout and efficient kernel decomposition.

3.4. Results for $f(r) = \exp(ik_0 r)/r$

The `npme` library implementation for the $f(r) = \exp(ik_0)/r$ kernel is benchmarked by computing the potential and its gradient for surface charges distributed on a sphere and the NASA almond [55] objects, illustrated [100] in Figure 4. The object surfaces are discretized into triangular facets. Charges are distributed on the surface using a three points per triangle quadrature grid [85], so the total number of charges is $N = 3N_T$, where N_T is the number of triangles. Each charge is assigned a complex random number with magnitude $q_{mag} \equiv \bar{l}/15$, where $\bar{l}$ is an average triangle length. The value for q_{mag} is chosen such that the average magnitude of potential is approximately equal to one, i.e. $|\bar{\phi}_i| \approx 1.0$, in order that the relative and absolute errors are approximately equal. The wavelength $\lambda_0 = 2\pi/k_0$ is related to the average triangle length by $\lambda_0 = 20\bar{l}$.

Figure 5 shows the `npme` computational times (left) and grid storage (right) as a function of N. The computations are performed at a precision level of 10^{-5} using B-spline order $n = 8$. For comparison, the computational times for exact potential and gradient evaluation using $O(N^2)$ pairwise summation are also shown.

The `npme` computational times are compared to total runtimes for the Helmholtz routine of the `fmm3D` code at the 10^{-5} precision level. A detailed

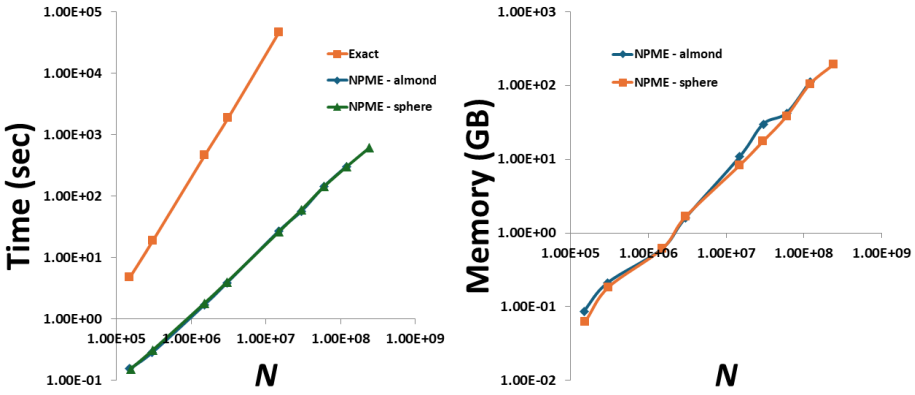

Figure 5: Computation time (left) and grid storage (right) for NASA almond and sphere at the 10^{-5} precision level.

N	`npme` (total)	`npme` (compute only)	`fmm3D` (total)
$3.0 \cdot 10^5$	2.2	0.29	4.0
$3.0 \cdot 10^6$	16.0	3.9	38.0
$3.0 \cdot 10^7$	258.0	57.0	391.0

Table 5: Comparison of compute times (seconds) for `npme` and `fmm3D` on the NASA almond geometry with N particles. The `npme` times are reported both with (total) and without (compute only) the one-time setup cost. All potential and its gradient calculations were performed using 64 threads and a precision level of 10^{-5}.

comparison for the NASA almond geometry with $N \approx 3.0 \cdot 10^5, 3.0 \cdot 10^6, 3.0 \cdot 10^7$ charges is shown in Table 5. On average, the `fmm3D` total runtimes are approximately $10x$ longer than `npme` computational times, although this ratio decreases for increasing system size. If the `npme` setup time is added to the computation time to form the total time, the `fmm3D` total runtime is $1.8x$ longer on average than `npme`.

4. Application: radar cross section

The `npme` library for the $f(r) = \exp(ik_0 r)/r$ kernel is integrated with the Method of Moments (MoM) [56] implementation for computing radar cross sections (RCS) of a perfect electric conductor (PEC). A brief overview of MoM is described here, while a more complete treatment can be found in [56, 57].

An electromagnetic plane wave with electric field $\mathbf{E}_{\text{inc}}(\mathbf{r})$, magnetic field $\mathbf{H}_{\text{inc}}(\mathbf{r})$, and implicit time dependence $\exp(-i\omega t)$ is incident on the outer surface of a solid conducting object. The incident wave induces an electric current on the surface, which generates scattered $\mathbf{E}_{sc}(\mathbf{r})$ and $\mathbf{H}_{sc}(\mathbf{r})$ fields outside the object. The fields are zero inside the conducting object, and the unknown surface current and scattered fields are determined by applying tangential component boundary conditions for the total electric and magnetic fields, as derived from Maxwell's equations.

The unknown electric surface current $\mathbf{J}(\mathbf{r})$ is expanded in the Rao-Wilton-Glisson (RWG) basis [84] $\mathbf{g}_m(\mathbf{r})$ by

$$\mathbf{J}(\mathbf{r}) = \sum_m c_m \mathbf{g}_m(\mathbf{r}) \tag{36}$$

The RWG basis functions $\mathbf{g}_m(\mathbf{r})$ are defined over pairs of adjacent triangles that share an edge. The surface current satisfies the equation of continuity

$$\nabla_S \cdot \mathbf{J}(\mathbf{r}) - i\omega\sigma(\mathbf{r}) = 0 \tag{37}$$

where ∇_S is the surface divergence operator [88] and $\sigma(\mathbf{r})$ is the scalar surface charge density.

The electric surface current coefficients are determined by solving the Combined Field Integral Equation (CFIE) [87]

$$\sum_m Z_{nm} c_m = a_n \tag{38}$$

where

$$a_n \equiv -\frac{1}{2} \oiint g_n(\mathbf{r}) \cdot (\mathbf{E}_{\text{inc}}(\mathbf{r}) + \hat{\mathbf{n}} \times \mathbf{H}_{\text{inc}}(\mathbf{r})) \tag{39}$$

$$Z_{nm} \equiv \frac{1}{2} \oiint \mathbf{g}_n(\mathbf{r}) \cdot \left(i\frac{\eta}{k} \mathbf{D}(\mathbf{g}_m) + \hat{\mathbf{n}} \times \mathbf{K}(\mathbf{g}_m) \right) \tag{40}$$

where $\eta \equiv \sqrt{\mu_0/\epsilon_0}$ is the impedance of free space, $k = \omega\sqrt{\mu_0\epsilon_0}$, and the $\mathbf{D}(\mathbf{X})$ and $\mathbf{K}(\mathbf{X})$ integral operators [86] are defined by

$$D(\mathbf{X}) \equiv k^2 \oiint G(R)\mathbf{X}(\mathbf{r}')dS' + \nabla \oiint G(R)\nabla'_S \cdot \mathbf{X}(\mathbf{r}')dS' \tag{41}$$

$$\mathbf{K}(\mathbf{X}) \equiv \oiint \nabla G(R) \times \mathbf{X}(r')dS' + \frac{1}{2}\hat{\mathbf{n}}(\mathbf{r}) \times \mathbf{X}(\mathbf{r}) \tag{42}$$

and $G(R) \equiv \exp(ikR)/(4\pi R)$ is the Helmholtz Green's function in three dimensions. CFIE is the standard integral equation method for solid conducting objects as the solutions are accurate and the Z matrix is well-conditioned [87] at finite frequencies.

The CFIE equations are solved iteratively for $\mathbf{c}$ using the Generalized Minimal Residual Method (GMRES) [63] algorithm to a residual norm of 10^{-5} with a sparse approximate inverse (SAI) preconditioner $\mathbf{M}$. Starting from the preconditioned initial guess $\mathbf{c} = \mathbf{M} \cdot \mathbf{a}$, matrix-vector products of the form $\mathbf{y} = \mathbf{Z} \cdot \mathbf{c}$ are computed and used to make refined estimates for $\mathbf{c}$. For small distance separations between basis functions $\mathbf{g}_n(\mathbf{r})$ and $\mathbf{g}_m(\mathbf{r})$, Z_{nm} is computed accurately using the $(1/R, R)$ singularity extraction technique [86] with triangle sub-partitioning for increased precision. For large distance separations, Z_{nm} is computed numerically with a three point per triangle Gaussian quadrature grid [85].

The NPME method accelerates matrix-vector products by efficiently computing interactions between spatially distant basis functions. A given current vector $\mathbf{c}$ is transformed into four sets of complex quadrature weighted charges corresponding to $(\mathbf{J}_x, \mathbf{J}_y, \mathbf{J}_z, \sigma)$ for the total set of grid points constructed from the three point per triangle grid. The potential and its gradient are computed for the four sets of point charges with NPME and converted to electric and magnetic fields. The NPME contribution represented by $\mathbf{y}_{npme} \equiv \mathbf{Z}_{npme} \cdot \mathbf{c}$ is computed by contracting the electric and

magnetic fields with $\mathbf{g}_n$ and $\mathbf{g}_n \times \hat{\mathbf{n}}$, respectively. The total matrix-vector product is given by

$$\mathbf{y} = \mathbf{Z}_s \cdot \mathbf{c} + \mathbf{Z}_{npme} \cdot \mathbf{c} \tag{43}$$

where $\mathbf{Z}_s \equiv \mathbf{Z} - \mathbf{Z}_{npme}$ is stored as a sparse matrix correction. Note that contributions to $\mathbf{y}_{npme}$ are accurate if the basis functions are separated by a large distance, while the $\mathbf{Z}_s \cdot \mathbf{c}$ contribution exactly corrects for inaccuracies in $\mathbf{y}_{npme}$ for basis functions separated by a short distance. The resulting MoM-NPME algorithm is a fast method for calculating $\mathbf{y} = \mathbf{Z} \cdot \mathbf{c}$. A hybrid MoM-NPME code is written by the author for calculating the RCS of solid conducting objects.

The surface of each object is discretized with triangular facets as illustrated in Figure 4 for the sphere and the NASA almond. The relationship between the number of triangular facets and wavelength is described in Section 3.4.

The bistatic RCS of a 1-meter diameter sphere is computed at 37.8 GHz, involving 60,064,230 unknowns. Plane waves with propagation vector $\mathbf{k}_i$ in the direction of $-\hat{\mathbf{x}}$ and electric field polarizations in the vertical $\hat{\mathbf{z}}$ and horizontal $\hat{\mathbf{y}}$ directions are incident upon the sphere located at the origin. The vertically and horizontally polarized bistatic RCS is measured in the xy-plane as a function of azimuthal angle ϕ. The bistatic RCS of the PEC sphere computed with MoM-NPME is compared to analytic results calculated by the Mie series [89] and plotted in Figure 6 as a function of ϕ for vertically polarized (top left) and horizontally polarized (top right) incident fields. Due to the large number of oscillations and limited plot resolution, the $165^\circ \leq \phi \leq 175^\circ$ intervals are expanded in order to demonstrate agreement between the MoM-NPME and Mie series results.

The monostatic RCS is computed for the NASA almond with incident frequency 75 GHz and 3,009,720 unknowns. Plane waves with propagation vectors $\mathbf{k}_i$ in the xy-plane are characterized by the azimuthal angle ϕ with electric field polarizations in the vertical and horizontal directions. The vertically and horizontally backscattered monostatic RCS is measured as a function of azimuthal angle ϕ. In order to obtain sufficient resolution, the angles between adjacent $\mathbf{k}_i$ are set to 0.1° resulting in 3602 different incident right-hand side vectors $\mathbf{a}$, which are linearly dependent. Singular value decomposition (SVD) is applied to the right-hand side matrix, reducing the required solutions to 326. The 75 GHZ monostatic RCS of

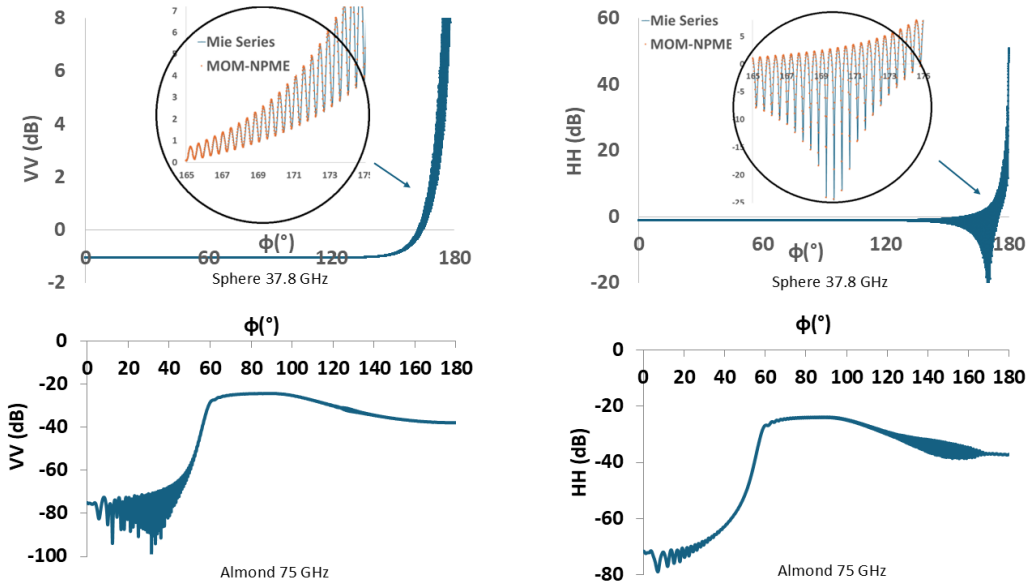

Figure 6: Bistatic rcs of a sphere 37.8 GHz (top) with 60,064,230 unknowns and monostatic rcs of a NASA almond 75 GHz (bottom) with 3,009,720 unknowns.

the NASA almond is also plotted in Figure 6 as a function of ϕ for both vertical (bottom left) and horizontal (bottom right) polarizations. In order to validate the NASA almond results, the monostatic RCS is computed at lower frequencies (3.5, 5.125, 7.0, and 10.25 GHz), and good agreement is observed with recent measured data [90]. See Section 4 of this Supporting Information for additional details.

5. Conclusions

The proposed NPME method is implemented in the C++ library and application code `npme`, currently optimized for static geometries with variable charges. `npme` is benchmarked against the Laplace and Helmholtz routines of `fmm3D` on test systems from computational chemistry and electromagnetics. For static systems, `npme` includes a one-time setup step that takes $3-4x$ the cost of a single computation, though this cost is negligible in typical applications. Excluding setup, `npme` is $2.8x$ and $10x$ faster than `fmm3D` for Laplace and Helmholtz kernels, respectively. Including setup, `npme` is $2.0x$ slower for Laplace and $1.8x$ faster for Helmholtz.

Parallel efficiency for the $f(r) = 1/r$ kernel was assessed on a 32-core

shared-memory system. While `npme` achieves somewhat lower scalability than `fmm3D` - 62% vs. 70% at $N = 10^6$, and 58% vs. 76% at $N = 10^7$ - its total runtimes remain competitive due to efficient kernel decomposition and compact memory layout. A future version may target improved scalability on GPUs using CUDA if sufficient user interest develops.

The `npme` library is also integrated into a Method of Moments (MoM) implementation for calculating the radar cross section (RCS) of perfect electric conductors. The bistatic RCS of a 1-meter sphere at 37.8 GHz with 60,064,230 unknowns is computed using MoM-NPME and shown to agree with Mie series results. The monostatic RCS of the NASA almond is computed at 75 GHz with 3,009,720 unknowns and validated against recent experimental data measured at lower frequencies.

Declaration of generative AI and AI-assisted technologies in the writing process

The original manuscript and accompanying `npme` source code were developed entirely by the author without the use of AI assistance. Generative AI (GPT-4-turbo) was used under the author's supervision to assist with language editing and to improve clarity and presentation of the final manuscript. All scientific content, algorithms, implementation details, and interpretations are the author's original work.

Data Availability

Data files, including RCS results and sample input files, can be downloaded from https://github.com/ElkingD/npme-data.

CRediT authorship contribution statement

Dennis M. Elking: Writing – review and editing, Visualization, Validation, Software, Methodology, Conceptualization

Declaration of competing interest

The author has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

Acknowledgments

The author would like to thank the anonymous reviewers for their comments and suggestions, which improved this manuscript. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Discrete Fourier transform properties

Consider a rectangular volume centered at the origin with period lengths L_1, L_2, L_3. Suppose the volume is partitioned into N_1, N_2, N_3 points for each of the x, y, z directions. A uniform grid in real space is defined within the rectangular volume by

$$\mathbf{r}_n \equiv \left(n_1 \frac{L_1}{N_1}, n_2 \frac{L_2}{N_2}, n_3 \frac{L_3}{N_3} \right) \tag{A.1}$$

where $n_i = -N_i/2, -N_i/2+1, ..N_i/2-1$. A function $f(\mathbf{r})$ evaluated on the grid with values is denoted by $f_{n_1 n_2 n_3} \equiv f(\mathbf{r_n})$. The discrete Fourier transform (DFT) of $f_{n_1 n_2 n_3}$ is given by

$$\widetilde{f}_{m_1 m_2 m_3} \equiv \frac{1}{N_1 N_2 N_3} \sum_{n_1=-\frac{N_1}{2}}^{\frac{N_1}{2}-1} \sum_{n_2=-\frac{N_2}{2}}^{\frac{N_2}{2}-1} \sum_{n_3=-\frac{N_3}{2}}^{\frac{N_3}{2}-1} f_{n_1 n_2 n_3} \omega_1^{-m_1 n_1} \omega_2^{-m_2 n_2} \omega_3^{-m_3 n_3} \tag{A.2}$$

where $m_i = -N_i/2, -N_i/2+1, ..N_i/2-1$ and $\omega_i \equiv \exp(2\pi i/N_i)$. The inverse DFT is given by

$$f_{n_1 n_2 n_3} = \sum_{m_1=-\frac{N_1}{2}}^{\frac{N_1}{2}-1} \sum_{m_2=-\frac{N_2}{2}}^{\frac{N_2}{2}-1} \sum_{m_3=-\frac{N_3}{2}}^{\frac{N_3}{2}-1} \widetilde{f}_{m_1 m_2 m_3} \omega_1^{m_1 n_1} \omega_2^{m_2 n_2} \omega_3^{m_3 n_3} \tag{A.3}$$

The fast Fourier transform (FFT) algorithm is used to efficiently compute both the DFT and inverse DFT. Suppose $\widetilde{f}_{m_1 m_2 m_3}$ and $\widetilde{g}_{m_1 m_2 m_3}$ are DFT's of $f_{n_1 n_2 n_3}$ and $g_{n_1 n_2 n_3}$, respectively. It is straightforward to show [4]

$$\sum_{p_1 p_2 p_3} \widetilde{f}_{p_1 p_2 p_3} g_{p_1 p_2 p_3} = \sum_{p_1 p_2 p_3} f_{p_1 p_2 p_3} \widetilde{g}_{p_1 p_2 p_3} \tag{A.4}$$

The plane wave vector $\mathbf{k}$ is defined by

$$\mathbf{k} \equiv 2\pi \left(\frac{m_1}{L_1}, \frac{m_2}{L_3}, \frac{m_3}{L_3} \right) \tag{A.5}$$

where m_i are integers. Note that Eqs. A.2 and A.3 can be re-expressed as

$$\widetilde{f_{\mathbf{k}}} = \frac{1}{N_1 N_2 N_3} \sum_{\mathbf{n}} f_{\mathbf{n}} \exp\left(-i\mathbf{k}\cdot\mathbf{r_n}\right) \tag{A.6}$$

$$f_{\mathbf{n}} = \sum_{\mathbf{k}} \widetilde{f_{\mathbf{k}}} \exp\left(i\mathbf{k}\cdot\mathbf{r_n}\right) \tag{A.7}$$

Eq. (A.7) is exact for $f_{\mathbf{n}} \equiv f(\mathbf{r_n})$ evaluated on the grid points. For arbitrary $\mathbf{r}$ within the rectangular volume, Eq. (A.7) is approximately true and leads to discrete Fourier transform (DFT) interpolation if $f(\mathbf{r})$ is a smooth periodic function and the grid is sufficiently dense.

$$f(\mathbf{r}) \equiv \sum_{\mathbf{k}} \widetilde{f_{\mathbf{k}}} \exp\left(i\mathbf{k}\cdot\mathbf{r}\right) \tag{A.8}$$

Appendix B. Discrete Fourier transform properties

The B-splines [101] of order n denoted by $B_n(w)$ are real variable functions and defined as

$$B_1(w) \equiv \begin{cases} 1 & 0 \le w < 1 \\ 0 & \text{otherwise} \end{cases} \tag{B.1}$$

$$B_{n+1}(w) \equiv \int_0^1 dv B_n(w-v) \tag{B.2}$$

For $n \ge 2$, $B_n(w)$ is a piece-wise continuous polynomial of order $n-1$. The following properties can be proved by induction

$$\frac{d}{dw} B_n(w) = B_{n-1}(w) - B_{n-1}(w-1) \tag{B.3}$$

$$B_n(w) = \frac{1}{n-1}\left(wB_{n-1}(w) + (n-w)B_{n-1}(w-1)\right) \tag{B.4}$$

In addition, it also possible to show by induction that $B_n(w) = 0$ for $w < 0$ or $w > n$.

A complex exponential can be approximated with B-splines by

$$\exp\left(2\pi i \frac{m}{M} w\right) \equiv \lambda_m \sum_{p=-\infty}^{\infty} \exp\left(-2\pi i \frac{mp}{M}\right) B_n(w+p) \tag{B.5}$$

where $-M/2 \le m < M/2$ are integers, w is a real variable, and

$$\lambda_m \equiv \frac{1}{\sum_{p=1}^{n-1} \exp\left(-2\pi i \frac{mp}{M}\right) B_n(p)} \tag{B.6}$$

As $B_n(w) = 0$ for $w < 0$ or $w > n$, there are at most n non-zero terms in the sum over p in Eq. (B.5). Eq. (B.5) can be re-written as

$$\exp\left(2\pi i \frac{m}{M} w\right) \equiv \lambda_m \sum_{p=-\frac{M}{2}}^{\frac{M}{2}-1} \exp\left(-2\pi i \frac{mp}{M}\right) \sum_{l=-\infty}^{\infty} B_n(w + p + lM) \tag{B.7}$$

The B-spline approximation to the complex exponential can be used to approximate $\exp(i\mathbf{k} \cdot \mathbf{r}_i)$ and its gradient. Let $\mathbf{r}_i \equiv (x_i, y_i, z_i)$ and note that

$$\exp(i\mathbf{k} \cdot \mathbf{r}_i) = \exp\left(2\pi i \left[\frac{m_1}{N_1}\widetilde{x}_i + \frac{m_2}{N_3}\widetilde{y}_i + \frac{m_3}{N_3}\widetilde{z}_i\right]\right) \tag{B.8}$$

where $\mathbf{k}$ is defined in Eq. (A.5), $\widetilde{x}_i \equiv N_1 x_i / L_1$, $\widetilde{y}_i \equiv N_2 y_i / L_2$, and $\widetilde{z}_i \equiv N_3 z_i / L_3$. Substituting Eq. (B.8) into Eq. (B.9) with $m = m_i$ and $M = N_i$ and $w = \widetilde{x}_i, \widetilde{y}_i$ or $\widetilde{z}_i$ results in

$$\exp(i\mathbf{k} \cdot \mathbf{r}_i) = N_1 N_2 N_3 \lambda_{m_1} \lambda_{m_2} \lambda_{m_3} \widetilde{C}^i_{m_1 m_2 m_3} \tag{B.9}$$

where $\widetilde{C}^i_{m_1 m_2 m_3}$ is the DFT of $C^i_{p_1 p_2 p_3}$ defined by

$$C^i_{p_1 p_2 p_3} \equiv \sum_{l_1, l_2, l_3 = -\infty}^{\infty} B_n(\widetilde{x}_i + p_1 + l_1 N_1) B_n(\widetilde{y}_i + p_2 + l_2 N_2) B_n(\widetilde{z}_i + p_3 + l_3 N_3) \tag{B.10}$$

If the constraint $n < N_i/4$ is imposed on the B-spline order n and grid sizes N_i, it can be shown that only the $l_1 = 0$ terms contribute to the sum in Eq. (B.10) by the following argument. Note that $L_1 > 2X$ and $|x_i| \leq X/2$ leads to $|\widetilde{x}_i| < N_1/4$. From the $n < N_i/4$ constraint and the B-spline property: $B_n(w) = 0$ for $w < 0$ or $w > n$, it follows that only the $l_1 = 0$ terms contribute to the sum in Eq. (B.10). By a similar argument, $l_2 = l_3 = 0$ and Eq. (B.10) simplifies to

$$C^i_{p_1 p_2 p_3} = B_n(\widetilde{x}_i + p_1) B_n(\widetilde{y}_i + p_2) B_n(\widetilde{z}_i + p_3) \tag{B.11}$$

The partial derivative of $C^i_{p_1 p_2 p_3}$ with respect to x_i is given by

$$C^{i,x}_{p_1 p_2 p_3} = \frac{N_1}{L_1} B'_n(\widetilde{x}_i + p_1) B_n(\widetilde{y}_i + p_2) B_n(\widetilde{z}_i + p_3) \tag{B.12}$$

which can be computed by the derivative property in Eq. B.3.

Appendix C. PME evaluation of $\phi_{i,rec}$ and $\nabla\phi_{i,rec}$

The PME evaluation of the reciprocal sum potential and its gradient is described in detail for periodic systems by Darden [4]. Most of the PME details are the same for periodic and non-periodic systems. For completeness a short summary of the PME method applied to non-periodic systems is given as follows. The Fourier extension in Eq. (4) is substituted into the reciprocal sum in Eq. (7)

$$\phi_{i,rec} = \sum_{\mathbf{k}} \exp(i\mathbf{k}\cdot\mathbf{r}_i)\widetilde{f}_l(\mathbf{k})S^*(\mathbf{k}) \tag{C.1}$$

where $S(\mathbf{k})$ is given by

$$S(\mathbf{k}) \equiv \sum_j \exp(i\mathbf{k}\cdot\mathbf{r}_j)q_j^* \tag{C.2}$$

and $\mathbf{k}$ is given in Eq. (A.5). The PME approximation to $S(\mathbf{k})$ is derived by substituting Eq. (B.9) into Eq. (C.2)

$$S_{PME}(\mathbf{k}) = N_1N_2N_3\lambda_{m_1}\lambda_{m_2}\lambda_{m_3}\widetilde{Q}_{m_1m_2m_3} \tag{C.3}$$

where $\widetilde{Q}_{m_1m_2m_3}$ is the DFT of $Q_{p_1p_2p_3}$ define by

$$Q_{p_1p_2p_3} \equiv \sum_i q_i^* C^i_{p_1p_2p_3} \tag{C.4}$$

and $C^i_{p_1p_2p_3}$ is given in Eq. (B.11). As $C^i_{p_1p_2p_3}$ is non-zero for n^3 terms, the cost of calculating $Q_{p_1p_2p_3}$is proportional to Nn^3 where N is the number of charges. The reciprocal sum potential is found by substituting the PME approximation to $S(\mathbf{k})$ in Eq. (C.3) and the B-spline approximation to $\exp(i\mathbf{k}\cdot\mathbf{r}_i)$ in Eq. (B.9) into Eq. (C.1)

$$\phi_{i,rec} = \sum_{m_1m_2m_3} \widetilde{C}^i_{m_1m_2m_3}\theta_{m_1m_2m_3} \tag{C.5}$$

where $\theta_{m_1m_2m_3}$ is given by

$$\theta_{m_1m_2m_3} \equiv \left|N_1N_2N_3\lambda_{m_1}\lambda_{m_2}\lambda_{m_3}\right|^2 \widetilde{f}_l(\mathbf{k})\widetilde{Q}^*_{m_1m_2m_3} \tag{C.6}$$

The sum in Eq. (C.5) can be transformed with the identity in Eq. (A.4)

$$\phi_{i,rec} = \sum_{m_1m_2m_3} C^i_{m_1m_2m_3}\widetilde{\theta}_{m_1m_2m_3} \tag{C.7}$$

For each i, note that n^3 terms contribute to the sum in Eq. (C.7). The reciprocal sum gradient is found by taking the derivative of Eq. (C.7)

$$\frac{\partial \phi_{i,rec}}{\partial x_i} = \sum_{m_1 m_2 m_3} C^{i,x}_{m_1 m_2 m_3} \widetilde{\theta}_{m_1 m_2 m_3} \tag{C.8}$$

where $C^{i,x}_{m_1 m_2 m_3}$ is given in Eq. (B.12).

Appendix D. McMurchie Davidson recursion

The McMurchie Davidson recursion is a useful and efficient method of calculating x, y, z partial derivatives of a radially symmetric function $f(r)$ given its radial derivatives. Define a gradient tensor of the form

$$T^n_{pqr} \equiv \frac{\partial^{p+q+r}}{\partial x^p \partial y^q \partial z^r} \left(\frac{1}{r} \frac{d}{dr} \right)^n f(r) \tag{D.1}$$

Note the $n = 0$ terms are the quantities of interest. The McMurchie Davidson recursion for the x component is given by

$$T^n_{p+1,q,r} = x T^{n+1}_{p,q,r} + p T^{n+1}_{p-1,q,r} \tag{D.2}$$

and follows from the Leibnitz rule for differentiating a product of two functions. Similar results hold for the y and z components. Starting from the radial derivatives T^n_{000} for $n = 0, 1, ..N_{\text{der}}$, T^0_{pqr} can be computed for all $p + q + r \leq N_{\text{der}}$.

Appendix E. $X_0 \rightarrow X$ correction

The $X_0 \rightarrow X$ correction described in Section 2.4 is introduced in order that the real space charge interpolation array $Q_{p_1 p_2 p_3}$ have non-zero values only for the small grid indexes, $p_i = -N_i/4, -N_i/4 + 1, ..N_i/4 - 1$. Recall $Q_{p_1 p_2 p_3}$ is defined in Eq. (C.4) in terms of the B-spline product $C^i_{p_1 p_2 p_3}$ given in Eq. (B.11). Consider the argument of the B-spline in the x direction of $C^i_{p_1 p_2 p_3}$. Recall $B_n(w) = 0$ for $w < 0$ or $w > n$. Thus, non-zero contributions to $C^i_{p_1 p_2 p_3}$ must satisfy

$$0 < \widetilde{x}_i + p_1 < n \tag{E.1}$$

After substituting $\widetilde{x}_i \equiv N_1 x_i / L_1$, the above inequality can be re-arranged as

$$-\frac{N_1 x_i}{L_1} < p_1 < n - \frac{N_1 x_i}{L_1} \tag{E.2}$$

which must be true for all particles including $x_{\max} \equiv \max_i(x_i)$ and $x_{\min} \equiv \min_i(x_i)$, i.e.

$$-\frac{N_1 x_{\max}}{L_1} < p_1 < n - \frac{N_1 x_{\min}}{L_1} \tag{E.3}$$

In addition, note that $X_0 \equiv x_{\max} - x_{\min}$ is the minimum length of the x component of the rectangular volume containing the particles. The min. and max. values of the grid indexes p_i lead to the following conditions

$$-\frac{N_1}{4} \leq -\frac{N_1 x_{\max}}{L_1} \tag{E.4}$$

$$n - \frac{N_1 x_{\min}}{L_1} \leq \frac{N_1}{4} - 1 \tag{E.5}$$

The first constraint re-arranges to $x_{\max} \leq L_1/4$ and the second condition re-arranges to

$$-\frac{L_1}{4}\left(1 - \frac{4(n+1)}{N_1}\right) \leq x_{\min} \tag{E.6}$$

Now translate the coordinates such that $x_{\max} = X/2$ for some $X \geq X_0$ which leads to the first constraint being satisfied, i.e. $x_{\max} < L_1/4$. Substituting $x_{\min} = \frac{X}{2} - X_0$ into Eq. (F.6) leads to

$$-\frac{L_1}{4}\left(1 - \frac{4(n+1)}{N_1}\right) \leq \frac{X}{2} - X_0 \tag{E.7}$$

After substituting $L_1 \equiv 2(X + \delta)$ where $\delta \equiv \Theta - X > 0$ is the smoothing zone Fourier extension parameter with value $\delta = R_{\text{dir}}$, Eq. (E.7) re-arranges to the following result

$$X \geq X_0 \frac{N_1}{N_1 - 2(n+1)} - \frac{\delta}{2}\frac{N_1 - 4(n+1)}{N_1 - 2(n+1)} \tag{E.8}$$

which leads to the definition for X in Eq. (33). Note the correction is automatically satisfied for $n << N_1$ and is needed only for small systems.

References

[1] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. 73 (1987) 325.

[2] R. Coifman, V. Rokhlin, S. Wandzura, The fast multipole method for the wave equation: a pedestrian prescription, IEEE Antennas Propag 35 (1993) 7.

[3] T. Darden, D. York, L. Pedersen, Particle mesh ewald: An n log(n) method for ewald sums in large systems, J. Chem. Phys. 98 (1993) 10089.

[4] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, L. G. Pedersen, A smooth particle mesh ewald method, J. Chem. Phys. 103 (1995) 1995.

[5] P. P. Ewald, Die berechnung optischer und elektrostatischer gitterpotentiale, Annalen der Physik 369 (1921) 253.

[6] J. W. Perram, H. G. Petersen, S. W. DeLeeuw, An algorithm for the simulation of condensed matter which grows as the 3/2 power of the number of particles, Mol. Phys. 65 (1988) 875.

[7] R. W. Hockney, J. W. Eastwood, Computer Simulation Using Particles, Routledge, 1989.

[8] B. Luty, M. E. Davis, I. Tironi, W. V. Gunsteren, A comparison of particle-particle, particle-mesh and ewald methods for calculating electrostatic interactions in periodic molecular systems, Mol. Sim. 14 (1994) 11.

[9] B. Luty, I. Tironi, W. V. Gunsteren, Lattice-sum methods for calculating electrostatic interactions in molecular simulations, J. Chem. Phys. 103 (1995) 3014.

[10] M. Deserno, C. Holm, How to mesh up ewald sums. i. a theoretical and numerical comparison of various particle mesh routines, J. Chem. Phys. 109 (1998) 7678.

[11] M. Deserno, C. Holm, How to mesh up ewald sums. ii. an accurate error estimate for the particle–particle–particle-mesh algorithm, J. Chem. Phys. 109 (1998) 7694.

[12] G. Efstatiou, J. W. Eastwood, On the clustering of particles in an expanding universe, Mon. Not. R. astr. Soc. 194 (1981) 503.

[13] R. J. Thacker, H. Couchman, A parallel adaptive p3m code withhierarchical particle reordering, Comput. Phys. Commun. 174 (2006) 540.

[14] C. Sagui, L. G. Pedersen, T. A. Darden, Towards an accurate representation of electrostatics in classical force fields: Efficient implementation of multipolar interactions in biomolecular simulations, J. Chem. Phys. 120 (2004) 73.

[15] G. A. Cisneros, J. P. Piquemal, T. A. Darden, Intermolecular electrostatic energies using density fitting, J. Chem. Phys. 123 (2005) 044109.

[16] G. A. Cisneros, J. P. Piquemal, T. A. Darden, Generalization of the gaussian electrostatic model: Extension to arbitrary angular momentum, distributed multipoles, and speedup with reciprocal space methods, J. Chem. Phys. 125 (2006) 184101.

[17] H. G. Petersen, Accuracy and efficiency of the particle mesh ewald method, J. Chem. Phys. 103 (1995) 3668.

[18] E. L. Pollock, J. Glosli, Comments on p3m, fmm, and the ewald method for large periodic coulombic systems, Comput. Phys. Commun. 95 (1996) 93.

[19] I. T. Todorov, W. Smith, K. Trachenko, M. T. Dove, "dl_poly_3: new dimensions in molecular dynamics simulations via massive parallelism", J. Mater. Chem. 16 (2006) 1911.

[20] B. Hess, C. Kutzner, D. van der Spoel, E. Lindahl, Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation, J. Chem. Theory Comput. 4 (2008) 435.

[21] B. R. Brooks, C. L. B. III, A. D. Mackerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. B. 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, M. Karplus, "charmm: The biomolecular simulation program", J. Comp. Chem. 30 (2009) 1545.

[22] J. A. Rackers, Z. Wang, C. Lu, M. L. Laury, L. Lagardère, M. J. Schnieders, J.-P. Piquemal, P. Ren, . . J. W. Ponder, 14, Tinker 8: Software tools for molecular design, J. Chem. Theory Comput. 14 (2018) 5273.

[23] J. C. Phillips, D. J. Hardy, J. D. C. Maia, J. E. Stone, J. V. Ribeiro, R. C. Bernardi, R. Buch, G. Fiorin, J. Henin, W. Jiang, R. McGreevy, M. C. R. Melo, B. K. Radak, R. D. Skeel, A. Singharoy, Y. Wang, B. Roux, A. Aksimentiev, Z. Luthey-Schulten, L. V. Kale, K. Schulten, C. Chipot, E. Tajkhorshid, Scalable molecular dynamics on cpu and gpu architectures with namd, J. Chem. Phys. 153 (2020) 044130.

[24] D. Case, H. Aktulga, K. Belfon, I. Ben-Shalom, J. Berryman, S. Brozell, D. Cerutti, T. Cheatham, III, G. Cisneros, V. Cruzeiro, T. Darden, N. Forouzesh, M. Ghazimirsaeed, G. Giambaşu, T. Giese, M. Gilson, H. Gohlke, A. Goetz, J. Harris, Z. Huang, S. Izadi, S. Izmailov, K. Kasavajhala, M. Kaymak, A. Kovalenko, T. Kurtzman, T. Lee, P. Li, Z. Li, C. Lin, J. Liu, T. Luchko, R. Luo, M. Machado, M. Manathunga, K. Merz, Y. Miao, O. Mikhailovskii, G. Monard, H. Nguyen, K. O'Hearn, A. Onufriev, F. Pan, S. Pantano, A. Rahnamoun, D. Roe, A. Roitberg, C. Sagui, S. Schott-Verdugo, A. Shajan, J. Shen, C. Simmerling, N. Skrynnikov, J. Smith, J. Swails, R. Walker, J. Wang, J. Wang, X. Wu, Y. Wu, Y. Xiong, Y. Xue, D. York, C. Zhao, Q. Zhu, P. Kollman, Amber 2024, University of California, San Francisco. (2024).

[25] A. C. Simmonett, F. C. P. IV, I. H. F. Schaefer, B. R. Brooks, An efficient algorithm for multipole energies and derivatives based on spherical harmonics and extensions to particle mesh ewald, J. Chem. Phys. 140 (2014) 1841014.

[26] T. J. Giese, M. T. Panteva, H. Chen, D. M. York, Multipolar ewald methods, 1: Theory, accuracy, and performance, J. Chem. Theory Comput. 11 (2015) 436.

[27] A. C. Simmonett, B. R. Brooks, Analytical hessians for ewald and particle mesh ewald electrostatics, J. Chem. Phys. 154 (2021) 104101.

[28] A. C. Simmonett, B. R. Brooks, A compression strategy for particle mesh ewald theory, J. Chem. Phys. 154 (2021) 054112.

[29] P. P. Poier, L. Lagardère, J.-P. Piquemal, Smooth particle mesh ewald-integrated stochastic lanczos many-body dispersion algorithm, J. Chem. Phys. 159 (2023) 154109.

[30] I. Chollet, L. Lagardère, J.-P. Piquemal, Ankh: A generalized o(n) interpolated ewald strategy for molecular dynamics simulations, J. Chem. Theory Comput. 19 (2023) 2887.

[31] T. J. Giese, M. T. Panteva, H. Chen, D. M. York, Multipolar ewald methods, 2: Applications using a quantum mechanical force field, J. Chem. Theory. Comput. 11 (2014) 451.

[32] T. J. Giese, D. M. York, Ambient-potential composite ewald method for ab initio quantum mechanical/molecular mechanical molecular dynamics simulation, J. Chem. Theory Comput. 12 (2016) 2611.

[33] D. Saintillan, E. Darve, E. S. G. Shaqfeh, A smooth particle-mesh ewald algorithm for stokes suspension simulations: The sedimentation of fibers, Phys. Fluids 17 (2005) 033301.

[34] G. J. Martyna, M. E. Tuckerman, A reciprocal space based method for treating long range interactions in ab initio and force-field-based calculations in clusters, J. Chem. Phys. 110 (1999) 2810.

[35] D. York, W. Yang, The fast fourier poisson method for calculating ewald sums, J. Chem. Phys. 101 (1994) 3298.

[36] Y. Shan, J. L. Klepeis, M. P. Eastwood, R. O. Dror, D. E. Shaw, Gaussian split ewald: A fast ewald mesh method for molecular simulation, J. Chem. Phys. 122 (2005) 054101.

[37] D. Lindbo, A.-K. Tornberg, Spectral accuracy in fast ewald-based methods for particle simulations, J. Comput. Phys. 230 (2011) 8744.

[38] L. af Klinteberg, D. S. Shamshirgar, A.-K. Tornberg, Fast ewald summation for free-space stokes potentials, Res Math Sci 4 (2017) 1.

[39] D. S. Shamshirgar, J. Bagge, A.-K. Tornberg, Fast ewald summation for electrostatic potentials with arbitrary periodicity, J. Chem. Phys. 154 (2021) 164109.

[40] J. Bagge, A.-K. Tornberg, Fast ewald summation for stokes flow with arbitrary periodicity, J.Comput. Phys. 493 (2023) 112473.

[41] F. Vico, L. Greengard, M. Ferrando, Fast convolution with free-space green's functions, J. Comput. Phys. 323 (2016) 191.

[42] L. Greengard, S. Jiang, Y. Zhang, The anisotropic truncated kernel method for convolution with free-space green's functions, SIAM J. Sci. Comput. 40 (2018) A3733.

[43] C. Predescu, A. K. Lerer, R. A. Lippert, B. Towles, J. P. Grossman, R. M. Dirks, D. E. Shaw, The u-series: A separable decomposition for electrostatics computation with improved accuracy, J. Chem. Phys. 152 (2020) 084113.

[44] S. Jin, L. Li, Z. Xu, Y. Zhao, A random batch ewald method for particle systems with coulomb interactions, SISC 43 (2021) B937.

[45] J. Liang, P. Tan, Y. Zhao, L. Li, S. Jin, L. Hong, Z. Xu, Superscalability of the random batch ewald method, J. Chem. Phys. 156 (2022) 014114.

[46] M. Pippig, D. Potts, Parallel three-dimensional nonequispaced fast fourier transforms and their application to particle simulation, SIAM J. Sci. Comput. 35 (2013) C411.

[47] F. Nestler, M. Pippig, D. Potts, Fast ewald summation based on nfft with mixed periodicity, J. Comput. Phys. 285 (2015) 280.

[48] M. Gruber, C. Koenen, T. Eibert, A fast fourier transform accelerated ewald summation technique for the vector lectromagnetic rectangular cavity green's function, J. Comput. Phys. 280 (2015) 570.

[49] L. Füsti-Molnár, P. Pulay, The fourier transform coulomb method: Efficient and accurate calculation of the coulomb operator in a gaussian basis, J. Chem. Phys. 117 (2002) 7827–7835.

[50] Y. Yao, J. Capecelatro, An accurate particle-mesh method for simulating charged particles in wall-bounded flows, Powder Technol. 387 (2021) 239–250.

[51] J. P. Boyd, A comparison of numerical algorithms for fourier extension of the first, second, and third kinds, J. Comput. Phys. 178 (2002) 118.

[52] Intel®, intrinsic vector functions. https://software.intel.com/sites/landingpage/intrinsicsguide/ (2025).

[53] R. Chandra, L. Dagum, D. Kohr, R. Menon, D. Maydan, J. McDonald, Parallel programming in OpenMP, Morgan kaufmann, 2001.

[54] W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, . . M. L. Klein, J. Chem. Phys. 79, Comparison of simple potential functions for simulating liquid water, J. Chem. Phys. 79 (1983) 926.

[55] A. Woo, H. Wang, M. Schuh, M. S. and, Em programmer's notebook-benchmark radar targets for the validation of computational electromagnetics programs, IEEE ANT PROPAG M. 35 (1993) 84.

[56] R. F. Harrington, Field Computation by Moment Methods reprint edition, Wiley-IEEE Press, New Jersey, 1993.

[57] W. C. Gibson, The Method of Moments in Electromagnetics, 3rd Edition, Chapman and Hall/CRC, Boca Raton, FL, 2022.

[58] T. Askham, Z. Gimbutas, L. Greengard, L. Lu, J. Magland, D. Malhotra, M. O'Neil, M. Rachh, V. Rokhlin, F. Vico, fmm3d 1.0.0, https://github.com/flatironinstitute/FMM3D (2022).

[59] L. F. Greengard, J. Huang, A new version of the fast multipole method for screened coulomb interactions in three dimensions, J. Comput. Phys. 180 (2002) 642.

[60] L. G. J. Huang, V. Rokhlin, S. Wandzura, Accelerating fast multipole methods for the helmholtz equation at low frequencies. ieee computational science and engineering, Comput. Sci. Eng. 5 (1998) 32.

[61] L. Greengard, V. Rokhlin., A new version of the fast multipole method for the laplace equation in three dimensions, Acta Numer. 6 (1997) 229.

[62] H. Cheng, L. Greengard, V. Rokhlin., A fast adaptive multipole algorithm in three dimensions, J. Comput. Phys. 155 (1999) 468.

[63] Y. Saad, M. H. Schultz, Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Comput. 7 (1986) 856.

[64] Y. Saad, Iterative Methods for Sparse Linear Systems, Society for Industrial and Applied Mathematics, 2003.

[65] C. C. Lu, W. C. Chew, Fast algorithm for solving hybrid integral equations, IEE Proc.-H 140 (1993) 455–460.

[66] J. M. Song, W. C. Chew, Fast multipole method solution using parametric geometry, Microw. Opt. Tech. Lett. 7 (1994) 760–765.

[67] B. Dembart, E. Yip, A 3d fast multipole method for electromagnetics with multiple levels, in: 11th Ann. Rev. Progress Appl. Comput. Electromagn., Vol. 1, Monterey, CA, 1995, pp. 621–628.

[68] J. M. Song, W. C. Chew, Multilevel fast-multipole algorithm for solving combined field integral equations of electromagnetic scattering, Microw. Opt. Tech. Lett. 10 (1995) 14–19.

[69] J. M. Song, C. C. Lu, W. C. Chew, Mlfma for electromagnetic scattering by large complex objects, IEEE Trans. Antennas Propagat. 45 (1997) 1488–1493.

[70] K. Donepudi, J. M. Jin, W. C. Chew, A higher order multilevel fast multipole algorithm for scattering from mixed conducting/dielectric bodies, IEEE Trans. Antennas Propag. 51 (2003) 2814.

[71] T. F. Eibert, C. H. Schmidt, Multilevel fast multipole accelerated inverse equivalent current method employing rao–wilton–glisson discretization of electric and magnetic surface currents, IEEE Trans. Antennas Propag. 57 (2009) 1178.

[72] D. T. Schobert, T. F. Eibert, Fast integral equation solution by multilevel green's function interpolation combined with multilevel fast multipole method, IEEE Trans. Antennas Propag. 60 (2012) 4458.

[73] J. Guan, S. Yan, J. M. Jin, An openmp-cuda implementation of multilevel fast multipole algorithm for electromagnetic simulation on multi-gpu computing systems, IEEE Trans. Antennas Propag. 61 (2013) 3607.

[74] N. N. Bojarski, k-space formulation of the electromagnetic scattering problem, Air Force Avionics Lab. Technical Report AFAL-TR-71-75 (1971).

[75] C. F. Wang, J. M. Jin, Simple and efficient computation of electromagnetic fields in arbitrarily-shaped inhomogeneous dielectric bodies using transpose-free qmr and fft, IEEE Trans. Microwave Theory Tech. 46 (1998) 553–558.

[76] J. M. Jin, J. Chen, H. Gan, W. C. Chew, R. L. Magin, P. J. Dimbylow, Computation of electromagnetic fields for high frequency magnetic resonance imaging applications, Phys. Med. Biol. 41 (1996) 2719–2738.

[77] T. K. Sarkar, E. Arvas, S. M. Rao, Application of fft and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies, IEEE Trans. Antennas Propagat. 34 (1986) 635–640.

[78] E. Bleszynski, M. Bleszynski, T. Jaroszewicz, Aim: Adaptive integral method for solving large scale electromagnetic scattering and radiation problems, Radio Sci. 31 (1996) 1225–1251.

[79] C. F. Wang, F. Ling, J. M. Song, J. M. Jin, Adaptive integral solution of combined field integral equation, Microw. Opt. Tech. Lett. 19 (1998) 321–328.

[80] F. Ling, C. F. Wang, J. M. Jin, An efficient algorithm for analyzing large-scale microstrip structures using adaptive integral method combined with discrete complex image method, IEEE Trans. Microwave Theory Tech. Lett. 19 (1998) 321–328.

[81] F. Ling, C. F. Wang, J. M. Jin, Application of adaptive integral method to scattering and radiation analysis of arbitrarily shaped planar structures, J. Electromag. Waves Appl. 12 (1998) 1021–1038.

[82] Z. Q. Lu, X. An, Fast monostatic radar cross-section computation for perfectly electric conducting targets using low-rank compression and adaptive integral method, IET Microw. Antennas Propag. 8 (2013) 46.

[83] H. Gong, H. Zhang, C. Liu, Y. Liu, X. Wang, T. Shan, An efficient method for analyzing the wideband rcs of multi-scale objects based on saim and cat, in: 2022 International Symposium on Antennas and Propagation (ISAP), 2022, pp. 291–292.

[84] D. Wilton, S. Rao, A. Glisson, D. Schaubert, O. Al-Bundak, C. Butler, Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains, IEEE Trans. Antennas Propag. 32 (1984) 276.

[85] D. A. Dunavant, High degree efficient symmetrical gaussian quadrature rules for the triangle, Int. J. Numer. Methods Eng. 21 (1985) 1129.

[86] P. Yla-Oijala, M. Taskinen, Calculation of cfie impedance matrix elements with rwg and n/spl times/rwg functions, IEEE Trans. Antennas Propag. 51 (2003) 1837.

[87] J. R. Mautz, R. F. Harrington, H-field, e-field, and combined-field solutions for conducting body of revolution, AEU 32 (1978) 157.

[88] H. F. Arnoldus, Conservation of charge at an interface, Opt. Commun. 265 (2006) 52.

[89] J.-M. Jin, Theory and Computation of Electromagnetic Fields, John Wiley and Sons, Inc., New Jersey, 2010.

[90] J. W. Massey, J. T. Kelley, C. Courtney, D. A. Chamulak, A. E. Yılmaz, A benchmark suite for quantifying rcs simulation performance on modern computers, in: Proc. USNC/URSI Rad. Sci. Meet., July 2018, p. 67.

[91] L. E. McMurchie, E. R. Davidson, One- and two-electron integrals over cartesian gaussian functions, J. Comput. Phys. 26 (1978) 218.

[92] J. Kolafa, J. W. Perram, Cutoff errors in the ewald summation formulae for point charge systems, Mol. Sim. 9 (1991) 351.

[93] H. Wang, F. Dommert, C. Holm, Optimizing working parameters of the smooth particle mesh ewald algorithm in terms of accuracy and efficiency, J. Chem. Phys. 133 (2010) 034117.

[94] G. B. Arfken, H. J. Weber, Mathematical Methods for Physicists, 4th Edition, Academic Press, New York, 1995.

[95] Intel®, Math kernel library 11.1 update 1 for linux* (2020).

[96] I. N. Herstein, Abstract Algebra, Wiley, 3rd edition, 1996.

[97] GNU, https://gcc.gnu.org/fortran/ (2020).

[98] A. Stone, The Theory of Intermolecular Forces, Oxford University Press; 2nd edition, 2016.

[99] W. Humphrey, A. Dalke, K. Shulten, Vmd: visual molecular dynamics, J. Molec. Graphics 14 (1996) 33.

[100] P. Cignonif, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia, Meshlab: an open-source mesh processing tool, in: Sixth Eurographics Italian Chapter Conference,, 2008, pp. 129–136.

[101] I. J. Schoenberg, Cardinal Spline Interpolation, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1973.