Rapid Sampling of Stochastic Displacements in Brownian Dynamics Simulations

Andrew M. Fiore,[1] Florencio Balboa Usabiaga,[2] Aleksandar Donev,[2] and James W. Swan[1, a)]

[1)]*Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

[2)]*Courant Institute of Mathematical Sciences, New York University, New York, NY 10012*

We present a new method for sampling stochastic displacements in Brownian Dynamics (BD) simulations of colloidal scale particles. The method relies on a new formulation for Ewald summation of the Rotne-Prager-Yamakawa (RPY) tensor, which guarantees that the real-space and wave-space contributions to the tensor are independently symmetric and positive-definite for all possible particle configurations. Brownian displacements are drawn from a superposition of two independent samples: a wave-space (far-field or long-ranged) contribution, computed using techniques from fluctuating hydrodynamics and non-uniform Fast Fourier Transforms; and a real-space (near-field or short-ranged) correction, computed using a Krylov subspace method. The combined computational complexity of drawing these two independent samples scales linearly with the number of particles. The proposed method circumvents the super-linear scaling exhibited by all known iterative sampling methods applied directly to the RPY tensor that results from the power law growth of the condition number of tensor with the number of particles. For geometrically dense microstructures (fractal dimension equal three), the performance is independent of volume fraction, while for tenuous microstructures (fractal dimension less than three), such as gels and polymer solutions, the performance improves with decreasing volume fraction. This is in stark contrast with other related linear-scaling methods such as the Force Coupling Method and the Fluctuating Immersed Boundary method, for which performance degrades with decreasing volume fraction. Calculations for hard sphere dispersions and colloidal gels are illustrated and used to explore the role of microstructure on performance of the algorithm. In practice, the logarithmic part of the predicted scaling is not observed and the algorithm scales linearly for up to 4×10^6 particles, obtaining speed ups of over an order of magnitude over existing iterative methods, and making the cost of computing Brownian displacements comparable the cost of computing deterministic displacements in BD simulations. A high-performance implementation employing non-uniform fast Fourier transforms implemented on graphics processing units and integrated with the software package HOOMD-blue is used for benchmarking.

Keywords: Colloids, Brownian Motion, Hydrodynamics, Coarse Graining

I. INTRODUCTION

Complex fluids composed of colloidal particles or dissolved polymers (herein referred to simply as particles) are common to many industrial applications. The dynamics of these materials are determined by the structure and interactions of the constituent elements. Micron and sub-micron sized constituents interact generically through the solvent via hydrodynamic interactions (HI). HI are critical to describing large scale collective motions abundant in diffusing[1], aggregating[2,3] and sedimenting dispersions[4]. For the slow viscous flows characteristic of colloidal motion, the HIs are determined by the viscous response of the fluid to motion of the immersed particles, as described by the Stokes equations. To account for Brownian motion of suspended particles, a fluctuating stress acting on the fluid must be incorporated into its momentum balance. On time scales that are long relative to the momentum relaxation time of the fluid, this stochastic stress has an equivalent representation as instantaneously correlated stochastic forces acting on directly on the particles. This suggests two different approaches for modeling stochastic motion of colloids. In one approach, the motion of the solvent is represented implicitly. The particle motion induced by hydrodynamic forces is constructed from multipolar expansions of the fundamental solutions of the Stokes equations, and stochastic forces acting directly on the particles are used to determine the particle trajectories. In an alternative approach, the motion of the solvent is determined explicitly either via discretization and solution of the Stokes equations around the particles, or by representing the solvent as a discrete collection of momentum carrying quasi-particles. For these explicit solvent techniques, the stochastic stress in the fluid domain or stochastic forces acting on the suspended particles and quasi-particles represent thermal fluctuations that satisfy fluctuation dissipation balance.

The most common implicit solvent method is Brownian Dynamics (BD) with hydrodynamic interactions (BD-HI), which approximates fluid-mediated particle interactions with the Rotne-Prager-Yamakawa (RPY) tensor[5] and explicitly integrates out inertia by taking the overdamped limit. Hydrodynamic interactions (HI) between particles described by the RPY tensor represent particles as a point force plus a quadrupole. Higher multipole

a)Electronic mail: jswan@mit.edu

moments (e.g. torques and stresslets) can be accounted for with a systematic multipole expansion, as done in Stokesian Dynamics[6]. Particle motion is linearly coupled to these multipole moments by the mobility tensor, $\mathcal{M}$. Fluctuation dissipation requires the Brownian displacements to have zero mean and variance proportional to $\mathcal{M}$, necessitating the computation of the action of $\mathcal{M}^{1/2}$ on a vector. Modern methods can compute the product of $\mathcal{M}$ with a vector in $\mathcal{O}(N)$ time, where N is the number of particles being modeled.[7] The long-standing approach to multiplying $\mathcal{M}^{1/2}$ by a vector has been to use either Cholesky decomposition or the iterative approximation of Fixman[8]. An iterative scheme with better error control has recently been developed[9], but all known iterative schemes have superlinear scaling of computational complexity with number of particles[10,11] because the condition number of $\mathcal{M}$ increases as a power law in N. This leads to superlinear scaling for stochastic simulations using implicit solvent techniques of $\mathcal{O}\left(N^{1.25-1.50} \log N\right)$, restricting dynamic simulations to systems of no more than a few thousand particles. A technique with linear scaling is clearly needed to perform multiscale simulations of structured materials. We describe here a novel linear-scaling algorithm for calculating Brownian motion in hydrodynamically interacting systems.

Explicit solvent methods which include fluid inertia, the most common of which are Lattice Boltzmann (LB)[12–14], Stochastic Rotation Dynamics (SRD)[15], and Dissipative Particle Dynamics (DPD)[16,17], are commonly cited as having linear scaling in the number of particles, even for stochastic simulations. While the operation count of explicit solvent algorithms is linear in the number of particles, these algorithms are limited by the need to resolve inertial degrees of freedom in the solvent, which results in a significant loss of efficiency on approach to overdamped systems. To approximately maintain overdamped dynamics at all simulated length scales, the dimensionless numbers relating process time scales to the rate of vorticity diffusion over the system size must also be controlled. For example, in SRD careful analysis has been performed to determine the relationship between simulation parameters and dimensionless groups such as the Schmidt, Reynolds, and Mach Numbers[18]. Controlling these variables over all length scales is especially important for flow and self-assembly in structured systems, where particles can form aggregates or percolating networks that approach the system size. The Reynolds number can be written as $Re = \ell^2/\tau\nu$ where ℓ is the relevant length scale, ν is the kinematic viscosity of the fluid, and τ is the flow time scale. An overdamped simulation of colloids, nanoparticles, or polymers should be performed so that $L^2/\tau\nu < Re_c$ where Re_c is the maximum allowable Reynolds number and L is the system size. The simulation time step Δt must be proportional to $\tau_I = a^2/\nu$ to resolve vorticity diffusion, where $a \sim \left(L^3\phi/N\right)^{1/3}$ is the particle size and ϕ is the volume fraction of particles in the system. Therefore, the number of time steps needed to resolve the flow time scale is $\tau/\tau_I \sim \phi^{-2/3} Re_c^{-1} N^{2/3}$.

Each time step requires $\mathcal{O}(N)$ operations, and thus the overall computational complexity of explicit solvent techniques with control over vorticity diffusion at all length scales is $\mathcal{O}\left(N^{5/3}\right)$. Another way of thinking about this scaling is that $\mathcal{O}(V) \sim N$ operations are required per time step to solve the Stokes equations, but $L^2/\nu \sim N^{2/3}$ time steps required for vorticity to diffuse throughout the system volume, leading to the $\mathcal{O}\left(N^{5/3}\right)$ scaling.

A class of "explicit solvent" methods directly tackles the overdamped limit by solving the *steady* Stokes equations[19–22]. This approach avoids using an analytic form of the mobility tensor, which can be an advantage for non-periodic domains[19]. Furthermore, the Brownian displacements needed for stochastic simulations can be generated with little to no extra cost, as demonstrated with the development of the Fluctuating Immersed Boundary (FIB) method,[19] and later incorporated into the fluctuating Force Coupling Method (FCM)[20,22] (which also includes higher-order stresslet corrections in the computation of the hydrodynamic interactions). This is accomplished by using fluctuating hydrodynamics, which in this context amounts to adding a stochastic forcing in the Stokes equations in the form of a stochastic stress tensor, rather than adding "Brownian forces" to the particles as in more traditional approaches. Our Positively-Split-Ewald (PSE) method builds on this key idea to compute the far-field (long-ranged) contribution to the hydrodynamic interactions in a periodic domain, by using Fast Fourier Transforms (FFTs) to efficiently solve the fluctuating Stokes equation.[19,20]

Both the FCM and FIB methods, while $\mathcal{O}(N)$, share a number of important shortcomings. The first of these is that in FCM/FIB one cannot independently control the form of the *near-field* hydrodynamic interactions between particles. Specifically, an RPY-like hydrodynamic mobility is obtained at large distances, but at short distances the hydrodynamic interaction tensor is different from RPY, in a way that depends on the specific form of the kernel (envelope) function used to transfer forces from the particles to the fluid. While one can argue that the RPY tensor is only accurate in the far field anyway, it is troublesome from a numerical methods perspective to change the physics of the model in order to make the numerical computations easier. The second important shortcoming is that methods such as FCM/FIB require a grid whose spacing is on the order of the particle size or smaller. This means that at low densities, where the RPY approximation is most accurate, these methods require large grids that eventually cannot even fit in the computer memory. FIB is better than FCM in this respect, since it allows for very compact kernels (3 or 4 grid points only in each direction) through the use of Peskin's specially-designed immersed boundary kernels.[23] However, a notable shortcoming of FIB, and in fact of all particle-mesh Ewald approaches as well, is that the accuracy of the computations cannot be controlled independently and some degree of lack of translational invariance must be accepted and then corrected for.[19]

Our PSE method overcomes the super-linear scaling of BD-like methods while bypassing the shortcomings of FCM/FIB-like methods and generates Brownian displacements whose variance is proportional to the exact, periodic RPY tensor. The PSE method represents the RPY tensor as a sum of two symmetric positive definite operators. We call this "positive splitting" because both parts of the RPY tensor are independendently positive-definite for all particle configurations. One operator is spatially local and easily evaluated via an exact analytical formula as in BD-like methods. The other operator is non-local and easily evaluated via FCM/FIB-like methods. The accuracy of the non-local operator is controlled by building on the recently-developed Spectral Ewald (SE) method,[24] which decouples numerical errors in interpolation from numerical errors resulting from discretization and solution of the steady Stokes equations. Samples of the Brownian displacement are drawn from a sum of two independent displacements whose variances are proportional to the local and the global parts of the RPY tensor, respectively. Local contributions are computed using the recently-proposed iterative Lanczos method for evaluating the action of the square root of a matrix.[9] Non-local contributions are computing using an approach inspired by fluctuating hydrodynamics. We show that this method is $\mathcal{O}(N)$ and that the accuracy of the various numerical approximations is completely decoupled. We can therefore optimize the algorithm to maximize efficiency, *without* sacrificing accuracy, which is controlled independently by a chosen relative error tolerance. In this sense the discretization, interpolation, and iterative approximations used in the PSE method are merely computational tools that do not affect the physics (RPY mobility) to within a prescribed tolerance.

The manuscript is organized as follows. In section IIA, we develop the PSE method by deriving a new Fourier space representation of the RPY tensor in a periodic geometry. Then, we utilize Ewald summation to deconstruct this tensor into local and non-local parts. In section IIB, we demonstrate how samples of the Brownian displacement may be drawn independently from two normal distributions with variance proportional to the local and non-local parts of the RPY tensor. In section IIC, we discuss optimization of algorithm performance while controlling the numerical error in all parts of the proposed algorithm. In Section III, we present various numerical tests which characterize the overall algorithm performance and compare that performance to other standard algorithms as well as analytically derived performance estimates.

II. POSITIVELY SPLIT EWALD APPROACH TO BROWNIAN DYNAMICS

A. The Mobility Tensor

The primary difficulty in performing dynamic simulations of hydrodynamically interacting particles is to calculate the velocity imparted to each particle due to forces acting on the other particles. In colloidal systems, the dynamics are overdamped, so the fluid motion is described by the Stokes equations. The fluid-mediated coupling among particles is represented by the symmetric, positive-definite mobility tensor $\mathcal{M}$. In a very common approach, $\mathcal{M}$ is constructed using a multipole expansion of the force density on particle surfaces to determine the flow field produced by the particles. Particle motion is inferred from Faxén laws that dictate how a flow field exerts stresses on the particle surfaces. Evaluating the multipole expansion requires knowledge of the Green's function for Stokes flow subject to known boundary conditions.

Periodic systems are common in simulation, and the appropriate Green's function is given by Hasimoto[25]

$$\mathbf{J}\left(\mathbf{x}, \mathbf{y}\right) = \frac{1}{\eta V} \sum_{\mathbf{k} \neq 0} e^{i\mathbf{k} \cdot (\mathbf{x} - \mathbf{y})} \frac{1}{k^2} \left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}\right), \qquad (1)$$

where $\mathbf{J}\left(\mathbf{x}, \mathbf{y}\right) \cdot \mathbf{F}$ is the fluid velocity at point $\mathbf{x}$ propagated by a force $\mathbf{F}$ located at point $\mathbf{y}$ and its periodic images on a lattice, $\mathbf{I}$ is the identity tensor, $\mathbf{k}$ are drawn from the set of reciprocal space lattice vectors excluding the zero vector, η is the fluid viscosity, and V is the periodic cell volume. The sum over periodic images in Equation (1) is slowly converging in most cases and divergent for $\mathbf{x} = \mathbf{y}$. For use in practical computation the summation process can be accelerated by using an Ewald method. A splitting function, $H(k, \xi) = \left(1 + \frac{k^2}{4\xi^2}\right) e^{-k^2/4\xi^2}$, introduced by Hasimoto[25] is used to decompose the sum into real space and wave space parts

$$\begin{aligned} \mathbf{J}\left(\mathbf{x}, \mathbf{y}\right) = \frac{1}{4\pi\eta} \sum_{\mathbf{R}} &\left[\frac{\xi}{\pi^{1/2}} \phi_{-1/2}\left(\xi^2 r^2\right) \mathbf{I} \right. \\ &\left. - \frac{\xi^3 r^2}{\pi^{1/2}} \phi_{1/2}\left(\xi^2 r^2\right) \left(\mathbf{I} - \hat{\mathbf{r}}\hat{\mathbf{r}}\right)\right] \\ &+ \frac{1}{\eta V} \sum_{\mathbf{k} \neq 0} e^{i\mathbf{k} \cdot (\mathbf{x} - \mathbf{y})} \frac{1}{k^2} H\left(k, \xi\right) \left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}\right), \quad (2) \end{aligned}$$

where $\mathbf{R}$ is the set of real space lattice vectors, $\mathbf{r} = \mathbf{x} - \mathbf{y} + \mathbf{R}$, and ϕ_ν are the incomplete gamma functions. The error ϵ associated with truncating the sums at finite radii r_{cut} and k_{cut} decays exponentially with a rate controlled by the splitting parameter, ξ. The real space and wave space truncation errors decay as $\epsilon_R \sim e^{-\xi^2 r_{\text{cut}}^2}$ and $\epsilon_W \sim e^{-k_{\text{cut}}^2/4\xi^2}$, respectively.

Assuming rigid spherical particles of equal radius a, a multipole expansion using this Green's function along

with Faxén's first formula yields a periodic form of the well-known Rotne-Prager (RP) tensor:[5,26]

$$\mathbf{M}_{\text{RP}}^{ij} = \begin{cases} \frac{1}{6\pi\eta a}, & i = j \\ \left[\left(1 + \frac{a^2}{6}\nabla_{\mathbf{x}}^2\right)\left(1 + \frac{a^2}{6}\nabla_{\mathbf{y}}^2\right)\mathbf{J}(\mathbf{x},\mathbf{y})\right]_{\substack{\mathbf{x}=\mathbf{x}_i, \\ \mathbf{y}=\mathbf{x}_j}}, & i \neq j \end{cases} \tag{3}$$

where $\mathbf{x}_i$ is the location of the center of particle i, and $\mathbf{U}_i = \mathbf{M}_{\text{RP}}^{ij} \cdot \mathbf{F}_j$ is the velocity of particle i induced by the force $\mathbf{F}_j$ on particle j. However, it is known that the operator (3) is symmetric positive definite (SPD) only when particles are not closer than the sum of their hydrodynamic radii. Additional regularizing corrections to maintain positivity are needed for overlapping configurations, which can occur in e.g. simulations of coarse-grained polymer chains or polymer-coated colloids. The common form used in BD-HI simulations is the Rotne-Prager-Yamakawa (RPY) tensor, which is SPD for all particle configurations, and has the form of an isotropic tensor for an unbounded domain:[5]

$$\mathbf{M}_{\text{RPY}}^{ij} = \frac{1}{6\pi\eta a} \begin{cases} \left(\frac{3a}{4r} + \frac{a^3}{2r^3}\right)\boldsymbol{I} + \left(\frac{3a}{4r} - \frac{3a^3}{2r^3}\right)\hat{\mathbf{r}}\hat{\mathbf{r}}, & r > 2a \\ \left(1 - \frac{9r}{32a}\right)\boldsymbol{I} + \left(\frac{3r}{32a}\right)\hat{\mathbf{r}}\hat{\mathbf{r}}, & r \leq 2a \end{cases}, \tag{4}$$

where $\mathbf{r} = \mathbf{x}_i - \mathbf{x}_j$. For periodic domains, a compact form of the RPY tensor must be derived.

Starting with the periodic Green's function, it is possible to derive a relationship for the mobility that is positive definite for all particle separations with no special regularization required, and whose constituent parts (the real space and wave space sums) are always independently positive-definite. This approach is based on the integral formulations of the the Stokes equations and integral representations of the Faxén laws for rigid bodies:

$$\mathbf{u}(\mathbf{x}) = \int_S dS(\mathbf{y})\,\mathbf{J}(\mathbf{x},\mathbf{y}) \cdot \mathbf{f}(\mathbf{y}), \tag{5}$$

$$\mathbf{U} = \frac{1}{4\pi a^2} \int_S dS(\mathbf{x})\,\mathbf{u}'(\mathbf{x}), \tag{6}$$

where $\mathbf{u}$ is the velocity field produced by a force density $\mathbf{f}$ on the particle surface S, and $\mathbf{U}$ is the velocity at which a particle moves in response to a flow field $\mathbf{u}'$ on its surface, induced by the other particles. Combining equations (5) and (6) for spheres of equal sizes, and keeping only the first moment in the multipole expansion of the force density, $\mathbf{f} = \mathbf{F}/4\pi a^2$, gives a velocity-force mobility relation between two particles,

$$\mathbf{U}_i = \frac{1}{4\pi a^2} \int_{S_i} dS_i(\mathbf{x})\,\frac{1}{4\pi a^2} \int_{S_j} dS_j(\mathbf{y})\,\mathbf{J}(\mathbf{x},\mathbf{y}) \cdot \mathbf{F}_j, \tag{7}$$

where $\mathbf{x}$ is on the surface of particle i and $\mathbf{y}$ is on the surface of particle j. As discussed in Refs. 27 and 28, this representation is identical to (3) for non-overlapping particles, but also gives an SPD tensor for overlapping particles as well, and can be used to generalize the RPY

tensor to other geometries. Using (1) and evaluating the integrals gives the pair mobility for spherical particles with equal finite size,

$$\mathbf{M}_{ij} = \frac{1}{\eta V} \sum_{\mathbf{k}\neq 0} e^{i\mathbf{k}\cdot(\mathbf{x}_i - \mathbf{x}_j)} \frac{1}{k^2}\left(\frac{\sin ka}{ka}\right)^2\left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}}\right). \tag{8}$$

This mobility describes the hydrodynamic interaction between two finite spheres and is positive definite and unconditionally convergent for all particle separations. The $\text{sinc}(ka) = (\sin ka)/ka$ terms are shape factors that account for the way in which spheres propagate flows into the fluid. Performing the calculations in (3) on the Hasimoto Green's function (1), produces a shape factor that is instead $\left(1 - k^2 a^2/6\right)^2$, which matches the Taylor expansion of $\text{sinc}(ka)$ up to second order in k. Higher order terms in the expansion of $\text{sinc}(ka)$ vanish after integration over the wave vectors due to the biharmonic nature of the Stokes flow. Equation (8) appears to be a very simple yet previously unknown Fourier space representation of the RPY tensor that will be the basis of our positive Ewald splitting; if one replaces the sum by an integral over all $\mathbf{k}$, (8) gives the standard free-space RPY tensor (4).

1. Positively Split Ewald RPY tensor

The first component required to perform BD-HI efficiently is to have a fast method for multiplying the mobility matrix by a vector. For the RPY tensor in an unbounded domain, a Fast Multipole Method (FMM) has been developed by rewriting the real-space representation of the far-field RPY tensor as a combination of several scalar (charge) monopole and dipole FMM evaluations.[7] In this approach the near-field RPY correction is simply incorporated in the near-field sum in the FMM. For point Stokeslets (monopoles), which do not include the quadrupolar far-field correction of the RPY tensor, an alternative fast Spectral Ewald method based on using the FFT has been developed by Lindbo and Tornberg[24] by building on the non-uniform FFT (NUFFT) method.[29] This method was recently incorporated into a Stokesian Dynamics (SD) method for Brownian polydisperse sphere suspensions by Wang and Brady[26]. The SD mobility includes the RP tensor, but the approach proposed by Wang and Brady[26] applies the differential formulation of the Faxén laws, which yields a non-positive splitting even for well-separated particles. Therefore, inclusion of Brownian motion is not straightforward. Here we propose a positive Ewald splitting of the RPY tensor that allows us to use the Spectral Ewald approach to efficiently and accurately evaluate the far-field contribution and rapidly generate Brownian increments.

Applying the Ewald sum splitting of Hasimoto[25] to decompose Equation (8) as $\mathbf{M}_{ij} = \mathbf{M}_{ij}^{(w)} + \mathbf{M}_{ij}^{(r)}$ into a long-ranged wave-space part $\mathbf{M}_{ij}^{(w)}$, and a short-ranged

real-space part $\mathbf{M}_{ij}^{(r)}$, and analytically performing the integral for the inverse Fourier transform to the short-ranged part, we get

$$\mathbf{M}_{ij}^{(w)} = \frac{1}{\eta V} \sum_{\mathbf{k} \neq 0} e^{i\mathbf{k} \cdot (\mathbf{x} - \mathbf{y})} \frac{\text{sinc}^2 (ka)}{k^2} H(k, \xi) \left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}} \right),$$
(9)

$$\mathbf{M}_{ij}^{(r)} = F(r, \xi) \left(\mathbf{I} - \hat{\mathbf{r}}\hat{\mathbf{r}} \right) + G(r, \xi) \, \hat{\mathbf{r}}\hat{\mathbf{r}},$$ (10)

where

$$H(k, \xi) = \left(1 + \frac{k^2}{4\xi^2} \right) e^{-k^2/4\xi^2},$$ (11)

and $F(r, \xi)$ and $G(r, \xi)$ are scalar functions given by

$$F(r, \xi) = \frac{1}{\eta} \frac{1}{2\pi^2} \int\limits_0^\infty dk \, (1 - H(k, \xi)) \, \text{sinc}^2 (ka)$$

$$\times \left(\frac{kr \cos kr + (k^2 r^2 - 1) \sin kr}{k^3 r^3} \right),$$ (12)

$$G(r, \xi) = \frac{1}{\eta} \frac{1}{\pi^2} \int\limits_0^\infty dk \, (1 - H(k, \xi)) \, \text{sinc}^2 (ka)$$

$$\times \left(\frac{\sin kr - kr \cos kr}{k^3 r^3} \right).$$ (13)

The integrals in equations (12) and (13) have complicated closed-form solutions, which are given in Appendix A.

These pairwise mobility functions can be easily applied to many-body simulations by summing the pairwise interactions:

$$\mathbf{U}_i = \sum_j \left(\mathbf{M}_{ij}^{(w)} + \mathbf{M}_{ij}^{(r)} \right) \cdot \mathbf{F}_j.$$ (14)

This set of sums can be written as the grand mobility tensor, $\mathcal{M}$

$$\mathbf{U} = \mathcal{M} \cdot \mathbf{F},$$ (15)

where $\mathbf{U}$ and $\mathbf{F}$ are vectors containing the velocity and force on all particles, respectively, and the blocks of $\mathcal{M}$ correspond to $\mathbf{M}_{ij}$. Like the pairwise mobility, the grand mobility has real space and wave space components, $\mathcal{M} = \mathcal{M}^{(w)} + \mathcal{M}^{(r)}$. $\mathcal{M}$ is symmetric and positive definite for all particle configurations. Using our approach, $\mathcal{M}^{(w)}$ and $\mathcal{M}^{(r)}$ are also guaranteed to independently be positive definite for all particle configurations and values of the splitting parameter, as demonstrated in Appendix B.

The shape factor $s^2(ka) = \text{sinc}^2(ka)$ and the splitting factor $H(k, \xi)$ collectively control whether a given representation of the RPY tensor can be positively split; both factors must be non-negative to guarantee positivity, which is required for the sampling method described

in the next section. This is illustrated by the Ewald sum of the RPY pair mobility tensor

$$\mathbf{M}_{\text{RPY}}^{ij} = \sum_{\mathbf{k} \neq 0} s^2(ka) \, (1 - H(k, \xi)) \, \mathbf{P}_k$$

$$+ \sum_{\mathbf{k} \neq 0} s^2(ka) \, H(k, \xi) \, \mathbf{P}_k,$$ (16)

where $\mathbf{P}_k = e^{i\mathbf{k} \cdot (\mathbf{x}_i - \mathbf{x}_j)} k^{-2} \left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}} \right)$, and the first sum is the wave space representation of the real space sum. Both $s^2(ka) \, (1 - H(k/\xi))$ and $s^2(ka) \, H(k/\xi)$ must be non-negative if each sum is to be SPD. Since $s^2(ka)$ and $H(k/\xi)$ are chosen independently, both $s^2(ka) \geq 0$ and $0 \leq H(k/\xi) \leq 1$ are required to guarantee a positive splitting for all values of k and ξ, i.e. guarantee that both the real space and wave space components of the mobility tensor are independently positive-definite. Beenakker's formulation of the Ewald sum of the RP tensor uses a differential representation of the tensor, shown in (3), and does not include overlap corrections[30]. The wave space representation of the shape factor associated with this formulation is $s_B^2(ka) = 1 - \frac{a^2 k^2}{3}$, which clearly is not positive when $a^2 k^2 > 3$. Therefore, RPY formulations based on this shape factor, even those that account for overlap corrections, cannot ensure a positive splitting with *any* splitting factor. In contrast, the shape factor used in this work is always non-negative. The two prevalent splitting factors are due to Hasimoto[25], $H_H(k/\xi) = \left(1 + \frac{k^2}{4\xi^2} \right) e^{-k^2/4\xi^2}$, used in this work, and Beenakker, $H_B(k/\xi) = \left(1 + \frac{k^2}{4\xi^2} + \frac{k^4}{8\xi^4} \right) e^{-k^2/4\xi^2}$, which has been used used in prior works[28,30]. Of these two, only the Hasimoto factor satisfies the condition that $0 \leq H(k/\xi) \leq 1$. Any choice of $H(k/\xi)$ such that $0 \leq H(k/\xi) \leq 1$ can be used in principle. The Hasimoto function is convenient because it results in exponentially decaying real-space sums. Additionally, any positive shape factor, e.g. the Gaussian kernels employed in the FCM technique[20], can be used. The $\text{sinc}^2(ka)$ shape factor derived in this work yields unconditionally convergent sums that *exactly* match the Stokes drag and RPY tensor for both overlapping and non-overlapping particles, without needing to explicitly consider overlap corrections.

B. Brownian Dynamics with Fast Stochastic Sampling

In Brownian dynamics simulations, particle trajectories evolve according to the overdamped Langevin equation

$$d\mathbf{x} = \mathcal{M} \cdot \mathbf{F} \, dt + \sqrt{2k_B T} \, \mathcal{M}^{1/2} \cdot d\mathbf{W}(t) + k_B T \, (\nabla \cdot \mathcal{M}) \, dt,$$ (17)

where $d\mathbf{x}$ is the set of particle displacements, k_B is Boltzmann's constant, T is temperature, $\mathbf{F}$ are forces applied to the particles, and $\mathbf{W}(t)$ is a collection of independent

standard Wiener processes. Here $\mathcal{M}^{1/2}$ denotes a "square root" of the mobility and can be any matrix that satisfies $\mathcal{M}^{1/2} \left(\mathcal{M}^{1/2} \right)^{\dagger} = \mathcal{M}$, where dagger denotes an adjoint. For translationally-invariant systems, including periodic domains, the divergence of the mobility appearing in the final stochastic drift term vanishes, $\nabla \cdot \mathcal{M} = 0$, and it is not necessary to use special methods in order to capture this term;[19,22] one can use the Euler-Maruyama method for temporal integration. Applying the action of the square root of the mobility is the most time-consuming calculation in a simulation. Iterative methods using e.g. Chebyshev polynomials[8] or a Lanczos process[9] require $N^{0.25} - N^{0.5}$ evaluations of the mobility to evaluate the square root. The number of iterations required has a power law dependence on the condition number of $\mathcal{M}$, which is a monotonically increasing function of N. The total operation count of this approach is $N^{1.25-1.5} \log N$ for accelerated algorithms such as particle-mesh Ewald[31] or Spectral Ewald[24].

We propose an alternative approach utilizing our novel formulation of the mobility to draw independent samples from the real space and wave space parts of the mobility,

$$\mathcal{M}^{1/2} \cdot d\mathbf{W}(t) \stackrel{d}{=} \mathcal{M}_R^{1/2} \cdot d\mathbf{W}_1(t) + \mathcal{M}_W^{1/2} \cdot d\mathbf{W}_2(t), \quad (18)$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are two independent Wiener processes. The two separate samples can be efficiently approximated by leveraging the structure of $\mathcal{M}_R$ and $\mathcal{M}_W$.

1. Real Space Contribution

The action of $\mathcal{M}_R^{1/2}$ can be readily approximated by iterative schemes such as those proposed by Fixman[8] or Chow and Saad[9] because of the short-ranged nature of the kernels F and G given in Eqs.(12)-(13) for *finite* $\xi > 0$. For a sufficiently small real space cutoff radius (sufficiently large ξ), $\mathcal{M}_R$ contains only local interactions and therefore has a compact eigenspectrum. Figure 1 shows the condition number κ of $\mathcal{M}_R$ as a function of ξ for varying N. The condition number is independent of N at a fixed cutoff radius, therefore the number of iterations is also asymptotically independent of N. The limit $\xi a \to 0$ represents the condition number of the full RPY mobility, $\mathcal{M}$, which contains long-ranged hydrodynamic interactions decaying as the inverse power of the distance, and becomes increasingly ill-conditioned for increasing N.

2. Wave Space Contribution

The wave space component of the mobility can be written as a sequence of linear operations:

$$\mathcal{M}_w = \mathbf{D}^{-1} \cdot \mathbf{B} \cdot \mathbf{D}, \quad (19)$$

where the diagonal matrix $\mathbf{B}$ maps the wave space representation of forces to velocities, and $\mathbf{D}$ is the non-uniform
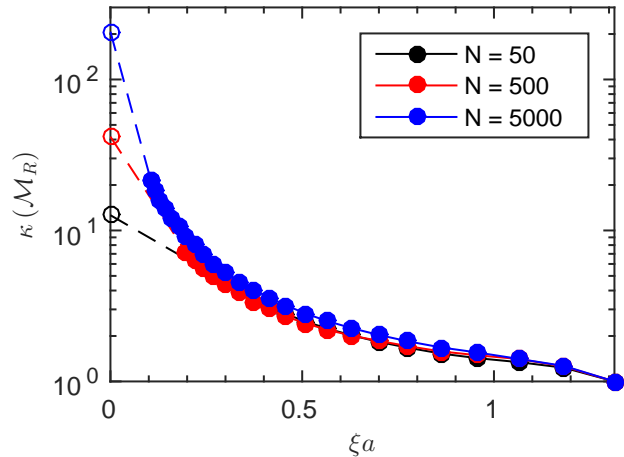

FIG. 1. Condition number of the real space mobility tensor $\mathcal{M}_R$ as a function of ξa for varying N.

Discrete Fourier transform (NUDFT) operator. The application of $\mathbf{D}$ converts the point forces applied on the particles to Fourier space. The application of $\mathbf{B}$ multiplies each Fourier component by the *non-negative* factor (see (9))

$$\frac{1}{\eta V} \frac{\text{sinc}^2(ka)}{k^2} H(k, \xi),$$

and projects onto the space of divergence free velocity fields via the projector $\left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}} \right)$. Lastly, the inverse NUDFT $\mathbf{D}^{-1}$ converts back to real space by evaluating the velocities at the positions of the particles. The matrix $\mathbf{B}$ is block-diagonal and positive semi-definite and therefore has a trivial analytical square root. Using the unitary property of the Fourier transform allows $\mathcal{M}_w$ to be re-expressed as

$$\mathcal{M}_w = \left(\mathbf{D}^{\dagger} \cdot \mathbf{B}^{1/2} \right) \cdot \left(\mathbf{D}^{\dagger} \cdot \mathbf{B}^{1/2} \right)^{\dagger}. \quad (20)$$

This shows that the wave space Brownian displacement can be calculated with a single matrix multiplication

$$\mathcal{M}_w^{1/2} \cdot d\mathbf{W}_2(t) \equiv \mathbf{D}^{\dagger} \cdot \mathbf{B}^{1/2} \cdot d\mathbf{W}_2(t). \quad (21)$$

The random vector $d\mathbf{W}_2(t)$ is complex valued because it is a random variable in wave space, unlike $d\mathbf{W}_1(t)$ which is real valued. Some care must be exercised to ensure the proper conjugacy conditions, $d\mathbf{W}_2(-\mathbf{k}) = d\mathbf{W}_2^{\dagger}(\mathbf{k})$, and obtain a real-valued particle velocity; details of this can be found in the article describing the fluctuating FCM method.[20]

In our work, following the approach taken for point Stokeslets in the Spectral Ewald method[24], we use the non-uniform FFT (NUFFT) method of Greengard and Lee[29] to obtain the action of $\mathbf{D}^{\dagger}$. The only change is to add the additional sinc factors in (9) to give the RPY

regularization of the Oseen tensor, i.e., replacing the inverse Laplacian k^{-2} with $\mathrm{sinc}^2\left(ka\right)/k^2$. The Spectral Ewald approach is also described in detail and compared to alternative particle-mesh Ewald approaches in Ref. 26. Here we only give a very brief description. The basic idea is to transfer information between the particle positions and a regular grid (to be used for the FFTs) using a Gaussian kernel truncated at m standard deviations, which is sufficiently resolved by the grid to obtain a desired error bound (number of digits of accuracy). The spreading of forces from the particles to the grid, and its adjoint operation, interpolation of velocities from the grid to the particles, is accomplished by directly evaluating the Gaussian spreading kernel on a subgrid of P^3 grid points around each particle, where P is a sufficiently large integer for a given error tolerance.[24] In the Spectral Ewald approach we have $\mathbf{D} = \mathcal{F} \cdot \mathbf{S}$ so $\mathcal{M}_w = \mathbf{S}^\dagger \cdot \mathcal{F}^{-1} \cdot \mathbf{B} \cdot \mathcal{F} \cdot \mathbf{S}$, where $\mathcal{F}$ represents the FFT. Here $\mathbf{S}$ spreads particle forces to the uniform grid and its adjoint $\mathbf{S}^\dagger = \sigma\,\mathbf{S}^T$ is the interpolation operator, where σ is the volume of a grid cell. This gives the desired

$$\mathcal{M}_w^{1/2} \equiv \sigma^{1/2}\,\mathbf{S}^T \cdot \mathcal{F}^\dagger \cdot \mathbf{B}^{1/2},$$

which can be applied to a vector of complex-valued Gaussian variates efficiently by a combination of rescaling, inverse Fourier transform, and interpolation operations. While we have used the Spectral Ewald approach to calculate the Ewald sum because of its spectral accuracy, in principle any algorithm for evaluating the wave space sum, e.g. Particle-Mesh Ewald, could be used. The above approach to the stochastic calculation is very similar to the one used in the FIB[19] and fluctuating FCM[20] methods. Although presented here from an algebraic perspective, the method has a very natural physical interpretation based on fluctuating hydrodynamics. Specifically, it can be seen as generating long-range correlated particle velocities (stochastic displacements) by applying a fluctuating stress to the fluid and then interpolating the resulting random velocity field at the positions of the particles.[32] As in FCM/FIB our method employs a smoothly varying kernel to represent the force density applied by the particles in the fluid. FCM uses Gaussian kernels, while FIB uses compactly supported kernels ($P = 3, 4,$ or 6) developed for the immersed boundary method.[23,33] The present work uses the sinc Fourier shape factors convoluted with Gaussians as the representation of the force distribution. In the FIB approach, unlike both FCM and our PSE method, the steady Stokes equations are solved using multigrid algorithms instead of FFTs; this makes the method easy to generalize to non-periodic domains. For a periodic system with only monopole terms included, the Brownian increment generation in FCM is analogous to the current approach with the real space sum neglected. For non-overlapping particles, PSE is equivalent to FCM if one chooses a value of ξ at which the error in truncating the real space sum at $r_{\mathrm{cut}} = 2a$ is less than the desired tolerance. The constraint to operate at larger ξ and the ability to balance
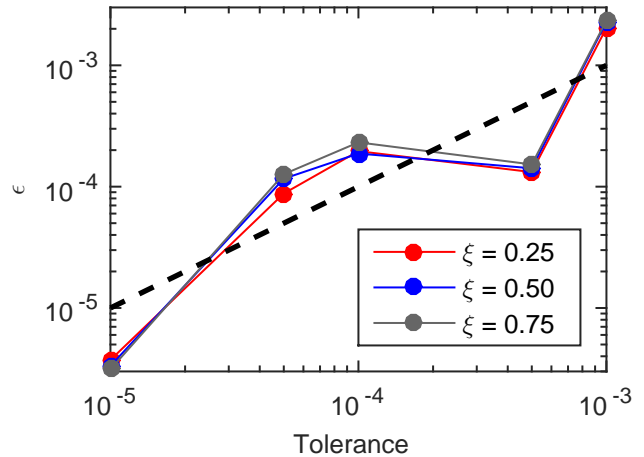

FIG. 2. Relative error in the mobility calculation as a function of desired tolerance for different ξ. The error is defined as $\epsilon = \|\mathbf{U} - \mathbf{U}_{\mathrm{exact}}\|/\|\mathbf{U}_{\mathrm{exact}}\|$, where the exact solution was computed using an error tolerance of 10^{-10}. The dashed line indicates equality between the computed error and the specified error tolerance. Discrete steps (integer values) are used for the support size in the NUFFT quadrature operations, which causes the bump at tolerances near 10^{-4}.

calculations between real space and wave space means that the PSE approach is potentially superior to FCM for periodic simulations of Brownian motion. Specifically, the PSE operator splitting provides flexibility to optimize performance across a broad range of volume fractions and microstructures guided by Equations (23) and (24).

C. Optimizing Performance

The algorithm's performance can be easily optimized when using the Spectral Ewald approach[24] because the sources of numerical error are decoupled and carefully controlled. There are three independent contributions to the total error in the Spectral Ewald approximation of $\mathcal{M}$: truncation of the real space sum at finite radius r_{cut}, truncation of the wave space sum at finite wave vector k_{cut}, and quadrature. The real space and wave space truncation errors are bounded by $\epsilon_R \sim e^{-\xi^2 r_{\mathrm{cut}}^2}$ and $\epsilon_W \sim e^{-k_{\mathrm{cut}}^2/4\xi^2}$. The quadrature error is bounded by $\epsilon_q \sim e^{-\pi^2 P^2/2m^2} + \mathrm{erfc}\left(m/\sqrt{2}\right)$, where m is the width of the Spectral Ewald Gaussian used in the NUFFT, and P is the number of grid points in each particle's support per spatial dimension. All errors decays exponentially, so the method is spectrally accurate. Furthermore, all the errors are independent, so the bounds for each term can be used to separately choose the operating parameters r_{cut}, k_{cut}, m, and P. Figure 2 shows the error in the sedimentation velocity for a random suspension of 5000 hard spheres ($\phi = 0.2$) as a function of the desired tolerance. The error is well-controlled and convergent.

At a fixed error threshold, $r_{\text{cut}} \sim \xi^{-1}$ and $k_{\text{cut}} \sim \xi$. The operation count for the real space sum of the mobility calculation is $n_R \sim C_R\, n_{\text{neigh}}\, N$. The average number of neighbors within a radius of r_{cut} of each particle is $n_{\text{neigh}} \sim \phi\, r_{\text{cut}}^{d_f} \sim \phi\, \xi^{-d_f}$ in a dispersion with volume fraction ϕ and fractal dimension d_f, so $n_R \sim C_R\, \phi\, \xi^{d_f}\, N$. C_R is an implementation-specific constant that is independent of N, ξ, and ϕ to within hardware specific effects such as atomic conflicts, memory bandwidth saturation, etc. Drawing the real space random samples requires n_{iter} iterations to compute $\mathcal{M}_R^{1/2} \cdot d\mathbf{W}_1$ to within a given tolerance, so the operation count is $n_R' \sim n_{\text{iter}}\, n_R = n_{\text{iter}}\, C_R\, \phi\, \xi^{-d_f}\, N$. Because the condition number of $\mathcal{M}_r$ is bounded for finite ξ, n_{iter} is independent of N.

The operation count to evaluate the wave space sum or draw wave space random samples is $n_W \sim C_W\, (\xi^3\, \phi\, N)\, \log(\xi^3\, \phi\, N) + C_Q\, NP^3$. The Fourier transform cost is controlled by $\xi\, \phi^{-1/3}\, N^{1/3}$, the number of Fourier modes per spatial dimension. The quantity $C_Q\, NP^3$ represents the operation count associated with the Spectral Ewald particle-to-grid and grid-to-particle quadrature operations. Like C_R, C_W and C_Q are implementation-specific and independent of ξ, ϕ, and N to within hardware effects. The total operation count is $n = n_R' + n_W$,

$$n \sim C_R\, n_{\text{iter}}\, N\, \phi\, \xi^{-d_f} + C_W\, (\xi^3\, \phi^{-1}\, N)\, \log(\xi^3\, \phi^{-1}\, N) \\ + C_Q\, NP^3. \tag{22}$$

This expression makes it apparent that work can be freely partitioned between wave space and real space, with the total work optimized at a particular value of the splitting parameter, ξ^*. This optimal value is found by setting the derivative Equation (22) with respect to ξ equal to zero. The solution is formally given by product logarithms, but can be simplified in the asymptotic limit of large N

$$\xi^* \sim \left(\frac{3C_W \log N}{C_R\, d_f\, n_{\text{iter}}\, \phi^2} \right)^{\frac{-1}{d_f+3}}. \tag{23}$$

To leading order, the operation count associated with ξ^* is

$$n^* \sim C_R\, n_{\text{iter}} \left(\frac{3C_W}{d_f\, C_R\, n_{\text{iter}}} \right)^{\frac{d_f}{d_f+3}} \phi^{\frac{3-d_f}{3+d_f}}\, N\, (\log N)^{\frac{d_f}{d_f+3}} \\ + C_q\, NP^3. \tag{24}$$

For three-dimensional simulations, $1 \leq d_f \leq 3$, so the timing scales between $N\, (\log N)^{1/4}$ and $N\, (\log N)^{1/2}$. These estimates are nearly linear functions of the system size, and in practice the logarithmic factors are not observed.

D. Comparison to the Force Coupling Method

The number of terms required to evaluate the wave space sum at a given error tolerance is dependent on the kernel used to represent the particle force distributions. The smallest length scale that needs to be resolved by the wave space sum is that of a single particle. Therefore, the wave space cutoff radius k_{cut} can be estimated by evaluating the error in the self mobility of a single particle relative to the stokes mobility $\mathbf{M}_{\text{stokes}} = (6\pi\eta a)^{-1}\, \mathbf{I}$. Note that in practice a fixed grid spacing $a/3$ is used in the FCM method,[21] deemed to give sufficient accuracy. Given a shape factor for the particle in wave space, $s^2(k)$, the error in the approximate wave space mobility, $\widetilde{\mathbf{M}}_{self}^{(w)}$, is

$$\delta\mathbf{M} = \mathbf{M}_{self}^{(w)} - \widetilde{\mathbf{M}}_{self}^{(w)} = \frac{1}{\eta V} \sum_{k>k_{\text{cut}}} s^2(k)\frac{1}{k^2}\left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}} \right). \tag{25}$$

The sum can be bounded by the integral

$$\delta\mathbf{M} \leq \frac{1}{\eta}\frac{1}{(2\pi)^3} \int_{k>k_{\text{cut}}} d\mathbf{k}\, s^2(k)\frac{1}{k^2}\left(\mathbf{I} - \hat{\mathbf{k}}\hat{\mathbf{k}} \right)$$

$$= \frac{\mathbf{I}}{3\pi^2\eta} \int_{k_{\text{cut}}}^{\infty} dk\, s^2(k). \tag{26}$$

The relative error is given by

$$\epsilon_k = \frac{\|\delta\mathbf{M}\|}{\|\mathbf{M}_{\text{stokes}}\|} \leq \frac{2a}{\pi} \int_{k_{\text{cut}}}^{\infty} dk\, s^2(k). \tag{27}$$

The positively split Ewald kernel is $s^2(k) = \left(\frac{\sin ka}{ka}\right)^2 \left(1 + \frac{k^2}{4\xi^2}\right) e^{-k^2/4\xi^2}$, and the FCM kernel is $s^2(k) = e^{-k^2 a^2/\pi}$. The exponentials in the PSE and FCM kernels are equal when $\xi^2 a^2 = \pi/4$.

Direct comparison of computational performance between FCM and PSE is difficult because of the relative cost of the FFTs and the spreading and interpolations depends strongly on implementation specifics. For example, Yeo and Maxey[21] report that in their implementation the FFT takes only 15% of the total cost, which is otherwise dominated by spreading and interpolation, whereas we find that the FFT is the dominant cost in our implementation. Instead of trying to compare numerical performance directly, we compare here the size of the FFT grid required by the two methods in order to achieve a certain target tolerance. Specifically, we compare the minimum value of k_{cut} required to reach a set error tolerance for the two kernels. In three dimensions, the execution time of the FFT roughly scales as k_{cut}^3, therefore $k_{\text{cut,FCM}}^3/k_{\text{cut,PSE}}^3$ provides an estimate of how much faster the FFT part of the algorithm will perform when using PSE. Figure 3 shows this ratio as a function of error tolerance. For an error tolerance of 10^{-3}, $k_{\text{cut,FCM}}^3/k_{\text{cut,PSE}}^3 \approx 10$ for a typical value of $\xi a = 0.5$ (see Fig. 4), i.e. the FFT grid in the PSE algorithm is ten times smaller than that in FCM for these parameters.

This error estimate can also be used to provide a slightly tighter bound on the error than that presented
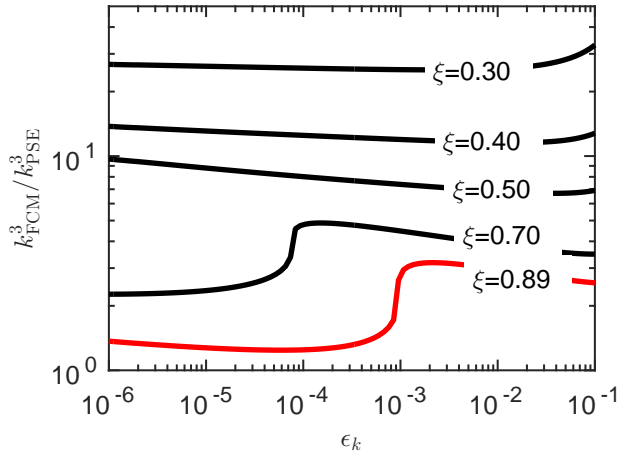
FIG. 3. Relative speedup for the wave space sum using the PSE kernel compared to the FCM kernel as a function of truncation error. The red line corresponds to $\xi^2 = \pi/4a^2$, where the PSE and FCM exponentials have the same decay rate.

by Lindbo and Tornberg[24]. Although there is not an analytical form to Equation (27) for the PSE kernel, an analytical expression does exist for the approximation $s^2(k) \leq \frac{1}{(ka)^2}\left(1 + \frac{k^2}{4\xi^2}\right)e^{-k^2/4\xi^2}$. That error estimate is

$$\epsilon_k \leq \frac{4k_{\text{cut}}^{-1}e^{-k_{\text{cut}}^2/4\xi^2} - \pi^{1/2}\xi^{-1}\text{erfc}(k_{\text{cut}}/2\xi)}{2\pi a}. \quad (28)$$

III. RESULTS AND DISCUSSION

We have implemented our PSE method on GPUs using CUDA in order to take benefit of the massively parallel architecture. The real space Ewald summation is routinely accelerated with linked-cell lists and a specified truncation radius for the interactions. This approach can reduce the cost of the sum from $\mathcal{O}\left(N^2\right)$ to $\mathcal{O}\left(N\right)$. Our implementation of the real space sum uses state-of-the-art cell and neighbor list algorithms supplied in the HOOMD-blue software suite[34,35]. The complicated real space mobility functions are pre-tabulated and stored in textured memory on the GPU, so their evaluation in the mobility multiplication is reduced to a memory fetch. The wave space sum is also truncated, and is accelerated using the non-uniform Fast Fourier transform (NUFFT).[29] Our NUFFT implementation is based on the Spectral Ewald technique of Lindbo and Tornberg[24] and uses the CUDA library CUFFT to efficiently evaluate the FFTs. The spreading operator **S** uses atomic additions and a particle-based (i.e., one block of threads per particle) algorithm to transfer particle forces to the grid. A particle-based algorithm is also used for **S**† to interpolate velocities from the grid to the particles. The atomic operations are very fast so particle-based approaches to
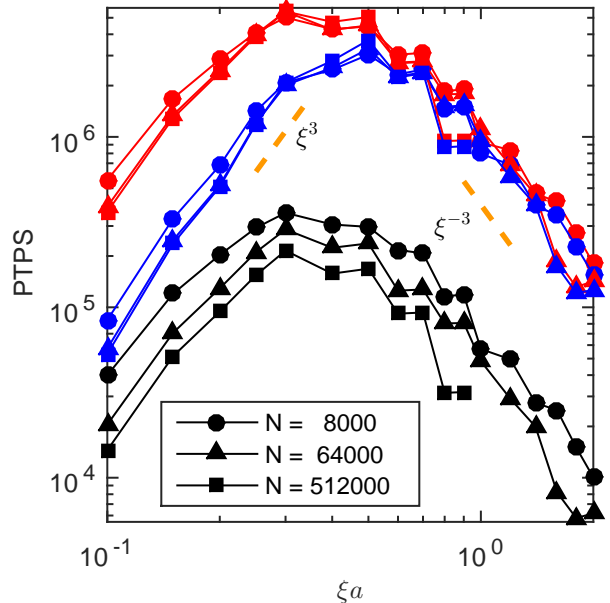

FIG. 4. Timing results for calculations on a random suspension of hard spheres ($\phi = 0.1$). The cost of the computation in terms of particle timesteps per second (PTPS) is given for computing $\mathcal{M} \cdot \mathbf{F}$ (red), Brownian displacements calculation $\mathcal{M}^{1/2} \cdot \mathbf{N}$ using the positively split Ewald sum (blue), and Brownian calculation using the Lanczos method applied to the full mobility (black), where a single matrix-vector product costs the same as the red curve.

spreading outperform grid-based approaches.[36]

The wave space parts of the deterministic and stochastic calculations are combined to minimize the number of FFTs required. Particle forces are spread to the real space grid, and transformed to wave space with one set of FFTs. After projecting the wave space gridded forces to the wave space velocities, the Brownian velocity is added directly to the grid. A set of inverse FFTs produces the real space velocity distribution on the grid, which is then interpolated onto particles. In principle, different values of ξ can be used for the stochastic and deterministic calculations, in which case the FFTs could not be combined and an additional set of FFTs would be required to generate the Brownian displacements.

The performance of the PSE algorithm is characterized in Figure 4 for a hard sphere dispersion with $\phi = 0.1$ as a function of ξ and N. The quantity reported, particle timesteps per second (PTPS), is the number of particles divided by the average time required to perform one step of the Euler-Maruyama time stepping method using the PSE algorithm, and represents the throughput of the algorithm on a per particle basis. The algorithm has been implemented as a plug-in to the GPU-based molecular dynamics tool kit HOOMD-Blue[34,35]. All timing data were obtained using an NVIDIA Tesla K40 GPU with the Maxwell architecture using a relative error tolerance of 10^{-3} for both the Ewald splitting and the Lanczos it-
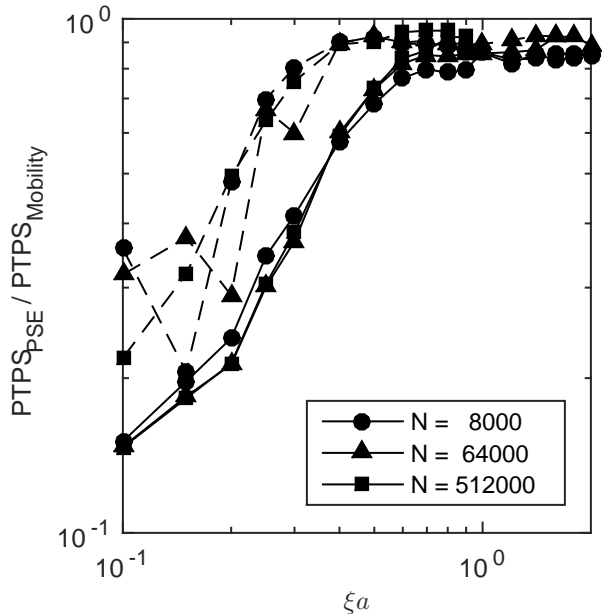
FIG. 5. Speed of the stochastic calculation relative to a single mobility evaluation for a random suspension of hard spheres ($\phi = 0.1$). Solid lines indicate calculations performed with a relative error tolerance of 10^{-3}, dashed lines a relative error of 10^{-5}.
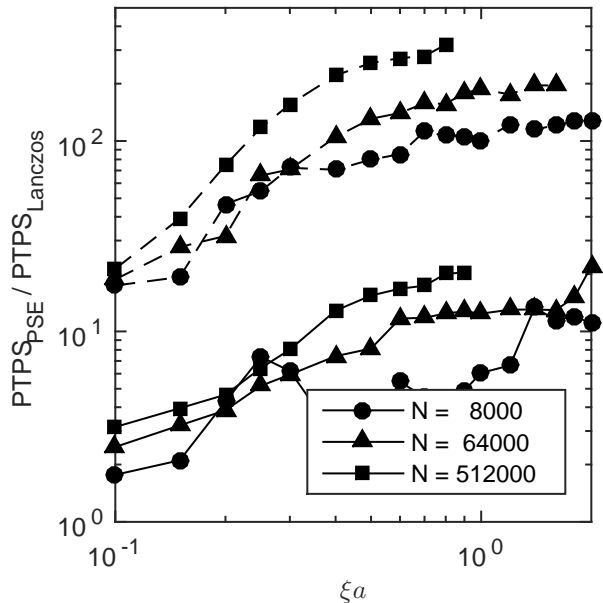

FIG. 6. Speed of the PSE stochastic calculation relative to the iterative Lanczos stochastic calculation for a random suspension of hard spheres ($\phi = 0.1$). Solid lines indicate calculations performed with a relative error tolerance of 10^{-3}, dashed lines a relative error of 10^{-5}.

eration for computing the real space contribution to the Brownian displacements, unless otherwise noted. Drawing a sample of the Brownian displacement using the positively split Ewald kernel (blue lines) is nearly as fast as a mobility calculation $\mathcal{M} \cdot \mathbf{F}$ (red lines) and more than an order of magnitude faster than samples drawn using the Lanczos iterative scheme applied to the full mobility tensor (black lines). The real space and wave space parts of the algorithm scale as expected, as shown by the constant slopes ξ^3 and ξ^{-3} (orange lines).

Figure 4 is an example of the algorithms' performance for a specific configuration. Systematically comparing the relative performance of the algorithms provides more insight into the advantage of the PSE method. The speed of sampling the random displacements $\mathcal{M}^{1/2} \cdot \mathbf{N}$, where $\mathbf{N}$ is a vector of i.i.d. standard Gaussian variables, using the PSE method relative to a single mobility product $\mathcal{M} \cdot \mathbf{F}$ is shown in Figure 5. The number of real space iterations required to sample the Brownian displacement increases weakly as ξ decreases, but is independent of N. The PSE sampling speed relative to the Lanczos iteration sampling speed is shown in Figure 6. The Lanczos approach applied to the full mobility becomes less efficient with increasing system size because the number of iterations required grows with N. Applying the Lanczos approach to draw samples from the the real space part of the mobility mitigates this problem because the real space operator has a condition number that is independent of N. The long range hydrodynamic interactions responsible for poor conditioning of the mobility are ac-

counted for in the wave space part of the mobility. Samples are drawn from this part directly with no iterations required. In the stochastic PSE algorithm, the extra cost associated with the real space Lanczos iterations is compensated by increasing ξ in order to shift work into the wave space part of the calculation. Figure 4 demonstrates this as a difference in the location of the peak for the mobility product (red, $\xi a \approx 0.3$) and the random sampling (blue, $\xi a \approx 0.5$).

Figure 7 shows the performance of the PSE algorithm for 512,000 particles at different volume fractions. The location of the optima shifts to higher ξ as the volume fraction increases, but the value at the optima remains unchanged. This is is further demonstrated in the inset, which shows the optimal performance as a function of system size for each ϕ. It is clear that optimal calculation speed is independent of volume fraction at a given N. At higher ϕ each particle has more interactions at a given cutoff radius, so some work must be shifted to wave space to maintain the optimal speed. The increase in PTPS with N is an artifact of the GPU. For smaller numbers of particles the hardware is not fully utilized. Increasing N occupies more GPU cores and it hides latency times, resulting in the observed throughput increase. As shown in Table I, large enough systems exhibit the expected levelling off and mild, logarithmic decrease in simulation throughput once the number of particles is large enough to fully occupy the hardware.

Figure 8 shows the time to draw the real space and wave space samples for a colloidal gel ($d_f \approx 2$) and a
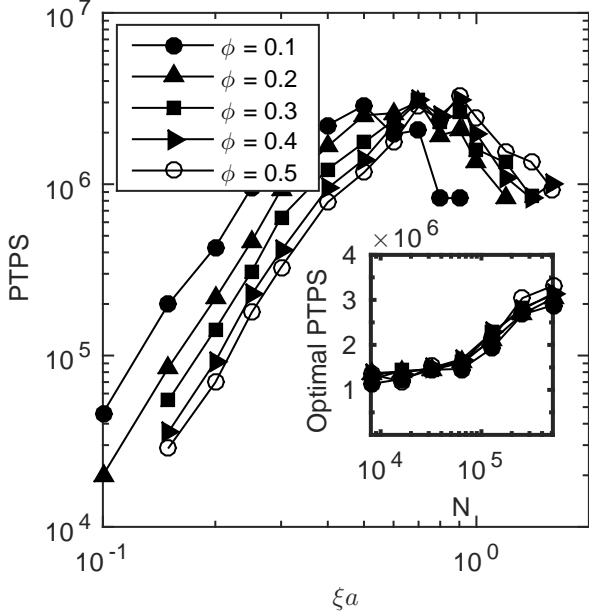
FIG. 7. Performance of the PSE algorithm as a function of ξ for a hard sphere dispersion of 512,000 particles at different volume fractions. The inset shows the optimal calculation performance as a function of N for each volume fraction.

TABLE I. Performance results for calculations of deterministic ($\mathcal{M} \cdot \mathbf{F}$) and stochastic ($\mathcal{M}^{1/2} \cdot \mathbf{N}$) velocities for a random suspension of hard spheres ($\phi = 0.1$) at $\xi a = 0.5$, which is roughly optimal for all N. N is given in thousand of particles, values are given in terms of millions of particle timesteps per second (10^6 PTPS).

$N \cdot 10^{-3}$	8	16	32	64	128	256	512	1024	2048
Deterministic	5.2	5.5	6.1	5.6	5.9	6.2	6.2	5.3	4.8
Stochastic	0.9	1.0	1.4	1.6	2.2	2.8	3.3	3.2	3.0

random hard sphere suspension ($d_f \approx 3$). Both systems have the same particle volume fraction, $\phi = 0.10$, so they each fill the same amount of space, but the space is occupied differently because the gel has a lower fractal dimension than the hard sphere suspension. The real space sum becomes more efficient at lower fractal dimensions due to the decreased number of neighbors within a given radius, as demonstrated by the offset between the real space timing curves. The wave space calculation is largely unaffected by the fractal dimension, so the measured times for the hard sphere and gelled dispersion are nearly equal. In the present implementation, the peak algorithm performance is no different for different particle configurations. The effect of fractal dimension is seen in the real space calculation, but is masked near the optimum. Performance floors associated with auxiliary parts of the algorithm become relevant near the optimal speed. In wave space, the FFTs become nearly as fast as the to-grid and from-grid operations, so the peak performance
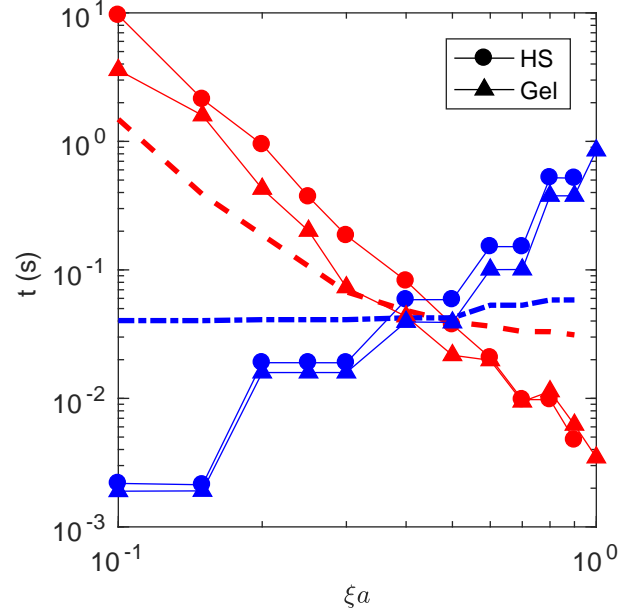

FIG. 8. Execution time for real space (red) and wave space (blue) components of the stochastic calculation for hard sphere and gel suspensions of 512,000 particles at $\phi = 0.1$. For the real space calculation, the solid curves are the time to compute the n_{iter} real space multiplications, and the dashed line is the time required to perform all other calculations associated with the Lanczos iterations. For the wave space calculation, the solid line is the time to compute the FFT/iFFT pair, plus the wave space multiplication and the random number generation. The dashed-dotted line is the sum of the spreading and interpolation time. The total time to evaluate a real and wave space calculation is the sum of the dashed (dashed-dotted) and solid lines.

is set by a balance of all three components. In real space, the evaluation of the near field sums is highly optimized by using the HOOMD linked-list implementation and by pre-tabulating the complicated truncated RPY kernels F and G. The floor is set by the time required to evaluate inner products associated with the Lanczos iterates.

The algorithm is so efficient that further speedup of the wave space sum would need to be achieved by improving the to-grid and from-grid operations. Fast to-grid and from-grid projection is an important computational problem. In particular, these operations have received attention for use in reconstructing MRI images through combinations of source-based and grid-based approaches to associate the non-uniform data with its grid support[37–40]. On GPUs, source-based algorithms reduce to a thread or thread block per particle using atomic operations to add to the grid point. Grid-based approaches, sometimes called gather algorithms, assign a thread or thread block per grid point or region and search for particles within that grid point's vicinity. The appropriate algorithm for a given application depends on the ratio of the source density (in our case, particle density) to the

grid point density. In the extreme case of many source and few grid points, grid-based approach are optimal. In the opposite extreme of many grid points and few sources, source-based approaches are optimal. Both approaches and combinations thereof are reasonable choices for the intermediate case. We have found that in our application the fastest spreading comes from using a block of threads for each particle to assign values to grid points within its support.

The to-grid and from-grid operations can also be accelerated using compactly supported kernels, as in immersed boundary methods[23,33]. These kernels are computationally efficient because a particle's support is localized to only 3, 4, or 6 grid points in each direction. However, compact kernels are not translationally invariant, which becomes more apparent as the support shrinks. More importantly, the accuracy with which the RPY tensor is computed cannot be controlled independently. In the FIB method only the far field RPY behavior is captured up to the quadrupole contributions if a kernel with a suitable positive second moment is used.[19]

Improvements to the spreading operations would lower the wave space floor, accelerating the wave space calculation improving overall efficiency. This highlights one of the most important features of the PSE algorithm: positive splitting of the real and wave space operators provides the ability to shift the optimal parameters based on changes in application or implementation. Improvements in performance of either the real space or wave space part of the algorithm can be translated into an overall speed-up by appropriately varying ξ. For example, the real space calculation could be more aggressively parallelized by using multiple GPUs and a smaller value of ξ. Using M GPUs can reduce the calculation time roughly by a factor of M with efficient domain decomposition algorithms[35], and distributing the inner products and additions required of the Lanczos iteration across multiple GPUs also translates to a factor of M increase in speed. The PSE algorithm can be optimized for both software and hardware implementations, with the overall execution speed approaching that of the fastest component of the calculation.

IV. CONCLUSION

In summary, a new formulation for the RPY mobility tensor has been derived for spherical particles in periodic geometries. When computed by Ewald summation, both the real space and wave space components of this tensor are symmetric, positive definite. Using this fact, a method to rapidly calculate Brownian displacements has been developed that represents samples of the Brownian displacement as independent contributions drawn from the real space and wave space parts of the mobility. The real space displacement is calculated using an iterative scheme to approximate the square root of the real space mobility tensor, while the wave space component

is calculated directly with a single matrix multiplication using FFTs. The new technique, Positively Split Ewald sampling (PSE), is two orders of magnitude faster than the standard Chebyshev polynomial approach of Fixman. Furthermore its performance can be tuned based on suspension microstructure and is consistent across a broad range of volume fractions. PSE is flexible and its performance can be easily optimized for various hardware and software implementations. The algorithm's efficiency enables dynamic simulations of $\mathcal{O}\left(10^6\right)$ particles in reasonable wall times on modern GPU hardware. This technique should find many applications in the polymer and soft matter communities, particularly for large scale simulation of self-assembling materials. An implementation of this algorithm, written as a plug-in for the molecular dynamics suite HOOMD-blue[34,35], has been made freely available as part of the supplementary material.

Generalization of the PSE method is possible. In the present work, only the force-velocity coupling among particles was modeled. However, an extension of the RPY tensor to describe higher order force moments such as torques and stresslets is possible[27,28]. These higher order terms would enter the formulation as a different set of shape factors, $s(ka)$, describing how flows due to higher order force distributions on particle surfaces are propagated by the fluid. Likewise, polydispersity may be incorporated directly by making the shape factor size dependent. Notably, equation (9) cannot be used to efficiently perform the wave space calculations. Instead, the polydisperse shape factors must be incorporated into the spreading and interpolation functions much as is done in FCM[21]. This approach is not taken in the present work with equal sized particles as a slightly larger support for the spreading and interpolation operations is required, which mildly reduces overall performance. A generalization from spherical particles to ellipsoids may also be possible through determination of an appropriate set of shape factors. Encouragingly, an equivalent RPY tensor for ellipsoids in an unbounded geometry is known and can be formally extended to describe the motion and interactions among colloidal particles with high aspect ratios[41]. Finally, extension of the method to non-periodic geometries deserves serious consideration. One possibility for achieving this is through application of the General Geometry Ewald Method, which splits the hydrodynamic interactions into local and non-local parts by a screening procedure similar to Ewald summation[42]. In this approach, a Stokes-like equation subject to non-periodic boundary conditions is used to represent the non-local part of the interaction, while an analytical form of the HI is used for local part. Ensuring positivity of the local and non-local operators while controlling the numerical error is key to making such an approach competitive with FIB, which can also be applied in non-periodic geometries.

As the scale of dynamic simulation of colloidal materials grows, careful consideration must be given to how local errors, which are precisely bounded in the PSE method, propagate into global errors. For example, in

most simulations of colloids, particles are subject to a conservative, pair-wise force. These conservative interactions should yield no net force on interacting particle pairs, but truncation errors in the wave space part of the PSE method applied to the mobility calculation will yield translation of the center of mass of the pair at a rate proportional to prescribed numerical tolerance. A local superposition of such errors, as might occur in a phase separating dispersion, could lead to a sizable net force acting on a condensed region of particles and lead to incorrect prediction of phase separation kinetics. The PSE method is spectrally accurate, however, and a smaller local error tolerance may be chosen to limit global erros over the scale of interest. In particle-mesh Ewald simulations of $\mathcal{O}(10^3)$ colloidal particles, local tolerances on the order of 10^{-3} have been standard[43]. Dynamic simulations of larger colloidal dispersions may necessitate tighter tolerances to limit the accumulation of local errors, which could lead to spurious particle fluxes. Alternatively, it may be possible to develop rescaled forms of the spreading and interpolation operators in the Spectral Ewald approximation to ensure that opposing forces sum to zero in aggregate when discretized. This is left for future work.

V. SUPPLEMENTARY MATERIAL

See supplementary material for a GPU implementation of the PSE algorithm built as a plugin for HOOMD. The supplementary material also includes a sample script to perform a dynamic simulation using the plugin, as well as results for the osmotic pressure of a hard sphere dispersion.

Appendix A: Real Space Scalar Mobility Functions

The scalar functions F and G are defined according to

$$6\pi\eta a\, \mathbf{M}_{ij}^{(r)} = F(r)\,(\mathbf{I} - \hat{\mathbf{r}}\hat{\mathbf{r}}) + G(r)\,\hat{\mathbf{r}}\hat{\mathbf{r}} \qquad \text{(A1)}$$

where

$$
\begin{aligned}
F(r) = {} & f_0 + f_1\, e^{-r^2\xi^2} + f_2\, e^{-(r-2a)^2\xi^2} + f_3\, e^{-(r+2a)^2\xi^2} \\
& + f_4\,\mathrm{erfc}\,(r\xi) + f_5\,\mathrm{erfc}\,((r-2a)\xi) \\
& + f_6\,\mathrm{erfc}\,((r+2a)\xi) \qquad\qquad\qquad\quad \text{(A2)} \\
G(r) = {} & g_0 + g_1\, e^{-r^2\xi^2} + g_2\, e^{-(r-2a)^2\xi^2} + g_3\, e^{-(r+2a)^2\xi^2} \\
& + g_4\,\mathrm{erfc}\,(r\xi) + g_5\,\mathrm{erfc}\,((r-2a)\xi) \\
& + g_6\,\mathrm{erfc}\,((r+2a)\xi) \qquad\qquad\qquad\quad \text{(A3)}
\end{aligned}
$$

Case 1, $r > 2a$

First scalar mobility function:

$$
\begin{aligned}
f_0 &= 0 \\
f_1 &= \frac{18r^2\xi^2 + 3}{64\sqrt{\pi}ar^2\xi^3} \\
f_2 &= \frac{2\xi^2(2a - r)\left(4a^2 + 4ar + 9r^2\right) - 2a - 3r}{128\sqrt{\pi}ar^3\xi^3} \\
f_3 &= \frac{-2\xi^2(2a + r)\left(4a^2 - 4ar + 9r^2\right) + 2a - 3r}{128\sqrt{\pi}ar^3\xi^3} \\
f_4 &= \frac{3 - 36r^4\xi^4}{128ar^3\xi^4} \\
f_5 &= \frac{4\xi^4(r - 2a)^2\left(4a^2 + 4ar + 9r^2\right) - 3}{256ar^3\xi^4} \\
f_6 &= \frac{4\xi^4(2a + r)^2\left(4a^2 - 4ar + 9r^2\right) - 3}{256ar^3\xi^4}
\end{aligned}
$$

Second scalar mobility function:

$$
\begin{aligned}
g_0 &= 0 \\
g_1 &= \frac{6r^2\xi^2 - 3}{32\sqrt{\pi}ar^2\xi^3} \\
g_2 &= \frac{-2\xi^2(r - 2a)^2(2a + 3r) + 2a + 3r}{64\sqrt{\pi}ar^3\xi^3} \\
g_3 &= \frac{2\xi^2(2a + r)^2(2a - 3r) - 2a + 3r}{64\sqrt{\pi}ar^3\xi^3} \\
g_4 &= -\frac{3\left(4r^4\xi^4 + 1\right)}{64ar^3\xi^4} \\
g_5 &= \frac{3 - 4\xi^4(2a - r)^3(2a + 3r)}{128ar^3\xi^4} \\
g_6 &= \frac{3 - 4\xi^4(2a - 3r)(2a + r)^3}{128ar^3\xi^4}
\end{aligned}
$$

Case 2, $r \le 2a$

First scalar mobility function:

$$f_0 = -\frac{(r-2a)^2\left(4a^2+4ar+9r^2\right)}{32ar^3}$$

$$f_1 = \frac{18r^2\xi^2+3}{64\sqrt{\pi}ar^2\xi^3}$$

$$f_2 = \frac{2\xi^2(2a-r)\left(4a^2+4ar+9r^2\right)-2a-3r}{128\sqrt{\pi}ar^3\xi^3}$$

$$f_3 = \frac{-2\xi^2(2a+r)\left(4a^2-4ar+9r^2\right)+2a-3r}{128\sqrt{\pi}ar^3\xi^3}$$

$$f_4 = \frac{3-36r^4\xi^4}{128ar^3\xi^4}$$

$$f_5 = \frac{4\xi^4(r-2a)^2\left(4a^2+4ar+9r^2\right)-3}{256ar^3\xi^4}$$

$$f_6 = \frac{4\xi^4(2a+r)^2\left(4a^2-4ar+9r^2\right)-3}{256ar^3\xi^4}$$

Second scalar mobility function:

$$g_0 = \frac{(2a-r)^3(2a+3r)}{16ar^3}$$

$$g_1 = \frac{6r^2\xi^2-3}{32\sqrt{\pi}ar^2\xi^3}$$

$$g_2 = \frac{-2\xi^2(r-2a)^2(2a+3r)+2a+3r}{64\sqrt{\pi}ar^3\xi^3}$$

$$g_3 = \frac{2\xi^2(2a+r)^2(2a-3r)-2a+3r}{64\sqrt{\pi}ar^3\xi^3}$$

$$g_4 = -\frac{3\left(4r^4\xi^4+1\right)}{64ar^3\xi^4}$$

$$g_5 = \frac{3-4\xi^4(2a-r)^3(2a+3r)}{128ar^3\xi^4}$$

$$g_6 = \frac{3-4\xi^4(2a-3r)(2a+r)^3}{128ar^3\xi^4}$$

Case 3, Self Mobility The real space part of the self mobility of a particle is given by the limit of equation (10) as $r \to 0$, specifically $\mathbf{M}^{(r)} = F(r{\to}0,\xi)\,\mathbf{I}$:

$$6\pi\eta a\,\mathbf{M}_{ii}^{(r)} = \frac{1}{4\pi^{1/2}\xi a}\left(1-e^{-4a^2\xi^2}+4\pi^{1/2}a\xi\,\mathrm{erfc}\,(2a\xi)\right)\mathbf{I}. \tag{A4}$$

Appendix B: Proof of Positive-Definiteness

Here we follow the proof presented by Cichocki et al.[44] for the positive-definiteness of a tensor defined by the quadratic form

$$\langle \mathbf{g} \,|\, \mathbf{J} \,|\, \mathbf{g}\rangle \equiv \int d\mathbf{x} \int d\mathbf{y}\, \mathbf{g}^*(\mathbf{x}) \cdot \mathbf{J}(\mathbf{x},\mathbf{y}) \cdot \mathbf{g}(\mathbf{y}) \tag{B1}$$

where an asterisk denotes the complex conjugate and the integrals in $\mathbf{x}$ and $\mathbf{y}$ are over all space, and Cichocki et al.

define

$$\mathbf{g}(\mathbf{x}) \equiv \sum_i \mathbf{w}_i(\mathbf{x}) \cdot \mathbf{d}_i \tag{B2}$$

where $\mathbf{w}_i$ is a tensor such that $\mathbf{w}_i \cdot \mathbf{F}_i$ is the force density on the surface of particle i and $\mathbf{d}_i$ is an arbitrary vector.[28] In this work, $\mathbf{w}_i(\mathbf{x}) = \frac{1}{4\pi a^2}\delta\left(\|\mathbf{x}-\mathbf{x}_i\|-a\right)\mathbf{I}$, where $\mathbf{x}_i$ is the location of the center of particle i, and the pair mobility $\mathbf{M}_{ij}$ is

$$\mathbf{M}_{ij} = \langle \mathbf{w}_i \,|\, \mathbf{J} \,|\, \mathbf{w}_j\rangle. \tag{B3}$$

Because the Green's function $\mathbf{J}$ is positive-definite, and its quadratic form can be related to $\mathbf{M}_{ij}$, $\mathcal{M}$ can be shown to be positive-definite as well:

$$0 \le \langle \mathbf{g} \,|\, \mathbf{J} \,|\, \mathbf{g}\rangle = \sum_{i,j} \mathbf{d}_i^* \cdot \mathbf{M}_{ij} \cdot \mathbf{d}_j. \tag{B4}$$

The wave space representation of $\mathbf{J}$ is $\mathbf{J} = \sum_{\mathbf{k}\neq\mathbf{0}}\mathbf{J_k}$, where $0 \le \langle \mathbf{g} \,|\, \mathbf{J_k} \,|\, \mathbf{g}\rangle$. In this representation, the mobility is

$$\mathbf{M}_{ij} = \langle \mathbf{w}_i \,|\, \sum_{\mathbf{k}\neq\mathbf{0}} \mathbf{J_k} \,|\, \mathbf{w}_j\rangle = \sum_{\mathbf{k}\neq\mathbf{0}} \langle \mathbf{w}_i \,|\, \mathbf{J_k} \,|\, \mathbf{w}_j\rangle, \tag{B5}$$

which can be decomposed into separate sums with the splitting function (homotopy) $H(k)$ to yield

$$\mathbf{M}_{ij} = \sum_{\mathbf{k}\neq\mathbf{0}} \langle \mathbf{w}_i \,|\, (1-H(k))\,\mathbf{J_k} \,|\, \mathbf{w}_j\rangle + \sum_{\mathbf{k}\neq\mathbf{0}} \langle \mathbf{w}_i \,|\, H(k)\,\mathbf{J_k} \,|\, \mathbf{w}_j\rangle, \tag{B6}$$

where the first term is the real space component of the mobility and the second term is the wave space component. It follows from $0 \le H \le 1$ and $0 \le \langle \mathbf{g} \,|\, \mathbf{J_k} \,|\, \mathbf{g}\rangle$ that

$$0 \le \sum_{\mathbf{k}\neq\mathbf{0}} \langle \mathbf{g} \,|\, (1-H(k))\,\mathbf{J_k} \,|\, \mathbf{g}\rangle = \sum_{i,k} \mathbf{d}_i^* \cdot \mathbf{M}_{ij}^{(r)} \cdot \mathbf{d}_j \tag{B7}$$

$$0 \le \sum_{\mathbf{k}\neq\mathbf{0}} \langle \mathbf{g} \,|\, H(k)\,\mathbf{J_k} \,|\, \mathbf{g}\rangle = \sum_{i,j} \mathbf{d}_i^* \cdot \mathbf{M}_{ij}^{(w)} \cdot \mathbf{d}_j \tag{B8}$$

which completes the proof that $\mathbf{M}_{ij}^{(r)}$ and $\mathbf{M}_{ij}^{(w)}$ are independently positive-definite. Note that any choice of H such that $0 \le H \le 1$ will produce a positive splitting, provided that each $\mathbf{J_k}$ is positive definite.

[1] J. W. Swan and J. F. Brady, The Journal of chemical physics **135**, 014701 (2011).

[2] Z. Varga, G. Wang, and J. Swan, Soft Matter **11**, 9009 (2015).

[3] Z. Varga and J. Swan, Soft Matter **12**, 7670 (2016).

[4] A. J. Ladd, Physical review letters **76**, 1392 (1996).

[5] J. Rotne and S. Prager, The Journal of Chemical Physics **50**, 4831 (1969).

[6] J. F. Brady and G. Bossis, Annual review of fluid mechanics **20**, 111 (1988).

[7] Z. Liang, Z. Gimbutas, L. Greengard, J. Huang, and S. Jiang, Journal of Computational Physics **234**, 133 (2013).

[8] M. Fixman, Macromolecules **19**, 1204 (1986).

[9] E. Chow and Y. Saad, SIAM Journal on Scientific Computing **36**, A588 (2014).

[10] T. Ando, E. Chow, Y. Saad, and J. Skolnick, The Journal of Chemical Physics **137**, 064106 (2012), http://dx.doi.org/10.10

[11]T. Geyer and U. Winter, The Journal of Chemical Physics **130**, 114905 (2009), http://dx.doi.org/10.1063/1.3089668.

[12]A. J. Ladd, M. E. Colvin, and D. Frenkel, Physical review letters **60**, 975 (1988).

[13]A. J. Ladd, Physical Review Letters **70**, 1339 (1993).

[14]A. Ladd and R. Verberg, Journal of Statistical Physics **104**, 1191 (2001).

[15]G. Gompper, T. Ihle, D. Kroll, and R. Winkler, in *Advanced computer simulation approaches for soft matter sciences III* (Springer, 2009) pp. 1–87.

[16]P. Hoogerbrugge and J. Koelman, EPL (Europhysics Letters) **19**, 155 (1992).

[17]P. B. Warren, Current opinion in colloid & interface science **3**, 620 (1998).

[18]J. T. Padding and A. A. Louis, Phys. Rev. E **74**, 031402 (2006).

[19]S. Delong, F. B. Usabiaga, R. Delgado-Buscalioni, B. E. Griffith, and A. Donev, The Journal of Chemical Physics **140**, 134110 (2014), http://dx.doi.org/10.1063/1.4869866.

[20]E. E. Keaveny, Journal of Computational Physics **269**, 61 (2014).

[21]K. Yeo and M. R. Maxey, Journal of computational physics **229**, 2401 (2010).

[22]B. Delmotte and E. E. Keaveny, The Journal of chemical physics **143**, 244109 (2015).

[23]C. S. Peskin, Acta numerica **11**, 479 (2002).

[24]D. Lindbo and A.-K. Tornberg, Journal of Computational Physics **229**, 8994 (2010).

[25]H. Hasimoto, Journal of Fluid Mechanics **5**, 317 (1959).

[26]M. Wang and J. F. Brady, Journal of Computational Physics **306**, 443 (2016).

[27]E. Wajnryb, K. A. Mizerski, P. J. Zuk, and P. Szymczak, Journal of Fluid Mechanics **731**, R3 (2013).

[28]K. A. Mizerski, E. Wajnryb, P. J. Zuk, and P. Szymczak, The Journal of chemical physics **140**, 184103 (2014).

[29]L. Greengard and J. Lee, SIAM Review **46**, 443 (2004).

[30]C. W. J. Beenakker, The Journal of Chemical Physics **85**, 1581 (1986).

[31]T. Darden, D. York, and L. Pedersen, The Journal of Chemical Physics **98**, 10089 (1993).

[32]A. Donev and E. Vanden-Eijnden, J. Chem. Phys. **140**, 234115 (2014).

[33]Y. Bao, J. Kaye, and C. S. Peskin, Journal of Computational Physics **316**, 139 (2016).

[34]J. A. Anderson, C. D. Lorenz, and A. Travesset, Journal of Computational Physics **227**, 5342 (2008).

[35]J. Glaser, T. D. Nguyen, J. A. Anderson, P. Lui, F. Spiga, J. A. Millan, D. C. Morse, and S. C. Glotzer, Computer Physics Communications **192**, 97 (2015).

[36]X. Guo, X. Liu, P. Xu, Z. Du, and E. Chow, Procedia Computer Science **51**, 120 (2015).

[37]G. Stantchev, W. Dorland, and N. Gumerov, Journal of Parallel and Distributed Computing **68**, 1339 (2008), general-Purpose Processing using Graphics Processing Units.

[38]J. L. Holtrop, X.-L. Wu, F. Lam, M. Fu, J. P. Haldar, W. mei W. Hwu, Z.-P. Liang, and B. P. Sutton, Journal of Parallel and Distributed Computing **73**, 686 (2013).

[39]S. Koziel, L. Leifsson, M. Lees, V. V.Krzhizhanovskaya, J. Dongarra, P. M. Sloot, X. Guo, X. Liu, P. Xu, Z. Du, and E. Chow, Procedia Computer Science **51**, 120 (2015).

[40]T. S. Sørensen, T. Schaeffter, K. Ø. Noe, and M. S. Hansen, IEEE Transactions on Medical Imaging **27**, 538 (2008).

[41]I. L. Claeys and J. F. Brady, Journal of Fluid Mechanics **251**, 411 (1993).

[42]J. P. Hernández-Ortiz, J. J. de Pablo, and M. D. Graham, Phys. Rev. Lett. **98**, 140602 (2007).

[43]M. Wang and J. F. Brady, Journal of Computational Physics **306**, 443 (2016).

[44]B. Cichocki, R. Jones, R. Kutteh, and E. Wajnryb, The Journal of Chemical Physics **112**, 2548 (2000).