Matrix-free Brownian dynamics simulation technique for semidilute polymeric solutions

Amir Saadat and Bamin Khomami*

Material Research and Innovative Laboratory, Department of Chemical and Biomolecular Engineering, University of Tennessee, Knoxville, Tennessee 37996-2200, USA

(Received 18 April 2015; published 29 September 2015)

Evaluating the concentration dependence of static and dynamic properties of macromolecules in semidilute polymer solutions requires accurate calculation of long-range hydrodynamic interactions (HI) and short range excluded volume (EV) forces. In conventional Brownian dynamics simulations (BDS), computation of HI necessitates construction of a dense diffusion tensor commonly performed via Ewald summation. Krylov subspace techniques allow efficient decomposition of this tensor [computational cost scales as $O(N^2)$, where N is the total number of beads in bead-spring representation of macromolecules in a simulation box] and computation of Brownian displacements in the box. In this paper, a matrix-free approach for calculation of HI is implemented which leads to $O(N \log N)$ scaling of computational expense. The fidelity of the algorithm is demonstrated by evaluating the asymptotic value of center-of-mass diffusivity of polymer molecules at very low concentrations and their radius of gyration scaling as a function of number of beads for dilute and semidilute solutions (with concentrations up to 5 times the overlap concentration). In turn, a favorable comparison between our results and the blob theory is shown.

DOI: 10.1103/PhysRevE.92.033307 PACS number(s): 47.57.Ng, 83.10.Mj, 83.80.Rs

I. INTRODUCTION

The effect of concentration on equilibrium and dynamic properties of polymer solutions has been observed experimentally even at very low concentrations [1–3]. Accurate determination of the aforementioned properties of the polymer solutions near or above c^* (the concentration at which chains begin to partially overlap at equilibrium), i.e., the semidilute regime, is of great importance to the polymer physics as well as polymer processing communities. To this end, development of high-fidelity and computationally efficient simulation techniques for this class of fluids is important both from a scientific perspective and in industrial applications.

Brownian dynamics simulation (BDS) is a mesoscopic simulation technique that has been extensively used to study the equilibrium and nonequilibrium properties of solutions in a broad range of solute concentration, e.g., dilute and semidilute suspension of particles with simple or complex shapes [4–6] or synthetic and biological polymeric solutions [7–12]. In BDS, the influence of the solvent on the mesoscale solutes is implicitly considered via random Brownian and hydrodynamic drag forces.

In polymeric solutions, hydrodynamic interaction (HI) is present which results in the perturbation of the velocity field around a polymer segment due to the movement of all segments of the same chain (intrachain interaction) and the segments of other chains (interchain interaction). Simulations of large multichain systems are predominantly performed for a homogeneous system in an unbounded domain, which necessitates the application of a periodic boundary condition. Excluded volume (EV) is another important interaction that needs to be considered. This interaction is short range, in the sense that the interaction is restricted to the entities involved in the cutoff radius. For polymer solutions, the EV effect arises due to the fact that polymer segments cannot physically overlap and consequently there is a repulsive force between different parts of molecules [13]. Clearly, this could indirectly introduce an attractive interaction between the polymer and the solvent.

For dilute polymer solutions in the presence of HI, the most time consuming operation in performing BDS is the decomposition of the diffusion tensor [12]. The straightforward approach for the decomposition is the Cholesky factorization, which for a bead-spring chain with N_b beads requires $O(N_b^3)$ operations [4]. The Chebyshev polynomial and the Krylov subspace based techniques are two elegant alternatives which can reduce the number of operations to $O(N_b^{2.25})$ and $O(N_b^2)$, respectively. Recent comparison of the three aforementioned techniques has revealed that the Krylov subspace approach is the best choice, for all $N_b \gtrsim 10$, when the diffusion tensor is updated at each time step [12]. In semidilute polymer solutions, N_c chains are considered in a periodic box, so the number of interacting beads is $N = N_c N_b$. Due to the long-range nature of HI, each bead interacts not only with the beads inside the primary simulation box, but also with particles in all periodic replicas (images) of the primary box. This sum is known to be slowly and/or conditionally convergent [14,15]. Similar to electrostatic interactions (which is also an example of long-range interaction), Ewald summation can be employed to split the original sum into two exponentially decaying sums in real and reciprocal spaces [14–16]. In this approach, one can distribute the computational load between the real and reciprocal space sums by tuning a parameter α, such that one of the computations scales as $O(N)$ and the other scales as $O(N^2)$ [15]. Hence, in a straightforward implementation of the Ewald sum, the construction of the diffusion matrix requires $O(N^2)$ operations. This procedure followed by the decomposition which also scales as $O(N^2)$ (if the Krylov subspace method is used) are the most cost prohibitive procedures in simulating polymer solutions with concentrations well above c^*. Similar arguments apply for colloidal suspensions which contain N particles in the simulation box.

*Author to whom correspondence should be addressed: bkhomami@utk.edu

To mitigate the high computational cost of simulating long-range electrostatic interactions, approximation methods such as particle mesh Ewald (PME) [17,18] and/or particle-particle particle-mesh (P^3M) [19,20] have been successfully introduced. Smooth particle mesh Ewald (SPME) is another variant of PME which has improved accuracy due to utilization of high-order B-splines instead of the Lagrange function in PME, in the grid value assignment and interpolation parts of the algorithm. In this approach, small subset of interacting particles is treated in real space while the main load (a large number of particles) is transferred to the reciprocal space sum where fast Fourier transform (FFT) is employed to accelerate the computations. Efficient implementation of this method leads to scaling of $O(N \log N)$.

Guckel [21] applied the PME method developed for electrostatic interactions to hydrodynamic interactions of rigid particles in Stokes flow. Subsequently, Brady and co-workers introduced accelerated Stokesian dynamics (ASD) [22,23] which benefits from PME algorithm in the calculation of the far-field mobility matrix. Saintillan *et al.* [6] combined SPME and ASD algorithms to simulate long-range HI of sedimenting fibers. For polymer solutions confined between two parallel walls, Hernández-Ortiz *et al.* [24,25] introduced a "general geometry Ewald-like" BDS method that scales as $O(N^{1.25} \log N)$. In the context of BDS, Jain *et al.* [15] used another class of optimization which is based on the original Ewald summation but with an optimal value of α, such that computational expense is evenly distributed between real and reciprocal space sums. Jain *et al.* also used the Chebyshev polynomial approximation for calculating the decomposition of the diffusion tensor and were able to obtain a computational time that scales as $O(N^{1.8})$.

Recently, Liu and Chow [26] introduced the "matrix-free" approach in the context of BDS of colloidal suspensions. In their approach, the computational burden of the real space sum is substantially reduced by considering the interaction of particles within a small cutoff radius. This causes the real part of the diffusion matrix to be highly sparse; hence, it can be efficiently computed. To compute the decomposition of diffusion tensor, Liu and Chow used the Krylov subspace method (Note that, in a matrix-free algorithm, the direct decomposition of diffusion matrix, e.g., the Cholesky factorization, cannot be used; see Sec. II C). On the other hand, the reciprocal part is calculated using highly efficient FFT routines. The method is capable of simulating systems with as many as 500 000 particles.

While the Ewald summation for HI in ASD is based on the Oseen-Burgers tensor [22], Beenakker's Ewald sum [14] which is widely used in BDS is based on the Rotne-Prager-Yamakawa (RPY) tensor [15,16,26,27]. The RPY tensor consists of two branches: namely a far-field solution and the regularization of the singularity which occurs for interbead distances of less than the diameter of a bead (see Sec. II A). The Beenakker solution only considers the far-field part of the RPY tensor. This solution works well only when the beads do not overlap, which is the case if a strong enough EV potential is utilized. However, there should be a correction to the original Beenakker formula for the simulation of the systems where the overlap of the beads is permitted, e.g., θ solutions. Zhou and Chen [27] and Jain *et al.* [15] resolved this issue by appropriately taking into account the second branch of the RPY tensor in the case of an overlap.

In this work, we start by formulating the stochastic differential equation (SDE) such that it can be used in both Euler-Murayama as well as semi-implicit predictor-corrector schemes [8,12,28]. Then, we adopt a matrix-free algorithm for simulating semidilute polymer solutions where an optimized version of the Krylov subspace approach recently developed for calculating Brownian displacements [12,29] is implemented. Compared to the original implementation of a matrix-free approach for colloidal suspensions [26], our implementation has the extended capability of correctly accounting for the overlap between the beads, which is particularly important for polymer solutions in θ solvent or slightly better-than-θ solvent. Also in this work, the EV potential is incorporated using the soft Gaussian potential, which has been extensively used in predicting the behavior of macromolecules in slightly better-than-θ solvent and good solvent [12,30,31]. Overall, our algorithm has several improved features when compared to those of Stoltz *et al.* [16] and Jain *et al.* [15]: (i) it uses highly efficient libraries for sparse matrix vector multiplication (math kernel library or MKL) to calculate the real space contribution of the diffusion tensor; (ii) FFT calculations are performed using the efficient MKL routines; (iii) fine-grained level parallelization for shared memory platforms using OpenMP has been added.

II. SIMULATION ALGORITHM

A. Governing equations

The dynamics of a macromolecule in semidilute polymer solutions can be expressed using a coarse grained bead-spring model [15,16,27]. In this micromechanical model, the linear flexible polymer molecule with N_K independent statistical Kuhn steps is discretized using N_b identical beads, which resemble the centers of hydrodynamic resistance, connected by $N_b - 1$ springs, which account for the entropic force between the neighboring beads. For simulation of a semidilute polymer solution, there are N_c bead-spring chains in the simulation box. The box is assumed to have sides with equal length L; i.e., $V = L^3$. Therefore, the concentration of beads in the box is given by $c = \frac{N}{V}$, where $N = N_c N_b$. In what follows, we use the convention that $\nu, \mu = 1, \ldots, N$, $\beta = 1, \ldots, N_b$, $\gamma = 1, \ldots, N_b - 1$, $i = 1, \ldots, N_c$, and $q, s = 1,2,3$. The configurational state of the system can be specified by the position vector of all beads $\boldsymbol{r}_\nu$, or equivalently using the connector vectors of all springs in the simulation box $\boldsymbol{Q}_{i,\gamma}$ and the center of mass of all chains $\boldsymbol{r}_{c,i}$. As it was shown in earlier studies of bead-spring models [7,8,12], the Itô stochastic differential equation of motion which describes the time evolution of beads in the bead-spring model can be nondimensionalized using the the time scale $\lambda_H = \zeta/4H$ and length scale $l_H = \sqrt{k_B T/H}$, where ζ is the bead friction coefficient, which relates the hydrodynamic radius of the bead a_b to the solvent viscosity η_s through the Stokes relation, i.e., $a_b = \zeta/6\pi\eta_s$. H is Hooke's spring constant, k_B is the Boltzmann constant, and T is the absolute temperature. As we are dealing with the equilibrium properties of semidilute polymeric solutions in this paper, the terms regarding the flow

are not present in the SDE. Furthermore, since we employ the RPY HI tensor [8], the nondimensionalized SDE does not contain the spatial derivative of the diffusion tensor and can be represented as follows (note that the variables expressed in the rest of the paper are dimensionless unless otherwise stated):

$$\mathrm{d}\tilde{\boldsymbol{Q}} \equiv \tilde{\tilde{\boldsymbol{B}}} \cdot \mathrm{d}\boldsymbol{r} = \tilde{\tilde{\boldsymbol{B}}} \cdot \left[\frac{1}{4} \overbrace{\boldsymbol{D} \cdot \boldsymbol{F}^{\phi}}^{\boldsymbol{u}} \mathrm{d}t + \frac{1}{\sqrt{2}} \boldsymbol{C} \cdot \mathrm{d}\boldsymbol{W} \right], \quad (1)$$

where t denotes time, $\tilde{\boldsymbol{Q}}$ is a block column vector consisting of N_c block vectors $\boldsymbol{Q}_i$, each of which contains $N_b - 1$ connector vectors between the adjacent beads of an individual chain i, i.e., $\boldsymbol{Q}_{i,\gamma} = \boldsymbol{r}_{\nu+1} - \boldsymbol{r}_{\nu}$ and $\nu = (i-1)N_b + \gamma$. $\tilde{\tilde{\boldsymbol{B}}}$ is a diagonal square block matrix with dimensions N_c, where each of its diagonal elements is $\bar{\boldsymbol{B}}$, which is the transformation matrix used by Bird *et al.* [7] to convert position vectors to connector vectors and is defined as

$$\bar{\boldsymbol{B}} = \begin{bmatrix} -\boldsymbol{\delta} & \boldsymbol{\delta} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{\delta} & \boldsymbol{\delta} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & -\boldsymbol{\delta} & \boldsymbol{\delta} \end{bmatrix}_{N_b - 1 \times N_b} \quad (2)$$

with $\boldsymbol{\delta}$ being the 3×3 unit tensor. The diffusion matrix $\boldsymbol{D}$ is also a block matrix which contains $N \times N$ blocks and, due to the long-range nature of HI, each block involves the sum of the corresponding RPY HI tensor $\boldsymbol{\Upsilon}_{\nu\mu}$ over infinite periodic images. $\boldsymbol{\Upsilon}_{\nu\mu}$ between the beads ν and μ is written as

$$\boldsymbol{\Upsilon}_{\nu\mu} = \begin{cases} \boldsymbol{\delta}, & \nu = \mu, \quad (3a) \\ \frac{3\sqrt{\pi}h^*}{4r_{\nu\mu}} \left[\left(1 + \frac{2\pi h^{*2}}{3r_{\nu\mu}^2}\right)\boldsymbol{\delta} + \left(1 - \frac{2\pi h^{*2}}{r_{\nu\mu}^2}\right) \frac{\boldsymbol{r}_{\nu\mu}\boldsymbol{r}_{\nu\mu}}{r_{\nu\mu}^2} \right], & \nu \neq \mu,\ r_{\nu\mu} \geqslant 2\sqrt{\pi}h^*, \quad (3b) \\ \left[\left(1 - \frac{9r_{\nu\mu}}{32\sqrt{\pi}h^*}\right)\boldsymbol{\delta} + \frac{3}{32\sqrt{\pi}h^*} \frac{\boldsymbol{r}_{\nu\mu}\boldsymbol{r}_{\nu\mu}}{r_{\nu\mu}} \right], & \nu \neq \mu,\ r_{\nu\mu} < 2\sqrt{\pi}h^*, \quad (3c) \end{cases}$$

where $r_{\nu\mu} = |\boldsymbol{r}_{\nu\mu}|$ and $\boldsymbol{r}_{\nu\mu} = \boldsymbol{r}_{\nu} - \boldsymbol{r}_{\mu}$. h^* is the hydrodynamic interaction parameter which is related to the dimensionless hydrodynamic radius: $h^* = (1/\sqrt{\pi})a$ and $a = \frac{a_b}{l_H}$. Note that $\boldsymbol{F}^{\phi} = \boldsymbol{F}^{S} + \boldsymbol{F}^{EV}$ is the block force vector defined based on all conservative interactions, i.e., $\boldsymbol{F}_{\nu}^{S}$ which is the net spring force and the $\boldsymbol{F}_{\nu}^{EV}$, the total EV force on a bead due to the interaction with all other beads. $\boldsymbol{F}_{\nu}^{S}$ is obtained from the tension in the neighboring springs $\boldsymbol{F}_{i,\gamma}^{c}$ and $\boldsymbol{F}_{i,\gamma-1}^{c}$ with $\boldsymbol{F}_{i,\gamma}^{c}$ defined based on the connector vector between the adjacent beads, $\boldsymbol{Q}_{i,\gamma}$, and the corresponding force law. For instance, the finitely extensible nonlinear elastic (FENE) force law, which refers to the Warner approximation to the inverse Langevin function, is written as

$$\boldsymbol{F}_{i,\gamma}^{c} = \frac{\boldsymbol{Q}_{i,\gamma}}{1 - Q_{i,\gamma}^2/b} \quad (4)$$

where $Q_{i,\gamma} = |\boldsymbol{Q}_{i,\gamma}|$ and $\sqrt{b}$ is the maximum dimensionless length of polymer springs. Other forms of force law can be found elsewhere [12]. $\boldsymbol{F}_{\nu}^{EV}$ is calculated using the soft Gaussian potential introduced by Prakash and Öttinger [30]:

$$\boldsymbol{F}_{\nu}^{EV} = \begin{cases} \sum_{\mu \neq \nu} \left(\frac{z^*}{d^{*5}}\right) \boldsymbol{r}_{\nu\mu} \exp\left(-\frac{r_{\nu\mu}^2}{2d^{*2}}\right), & r_{\nu\mu} \leqslant r_{c,F}, \\ 0, & r_{\nu\mu} > r_{c,F}, \end{cases} \quad (5)$$

where the parameter z^* is an indication of EV potential strength and d^* specifies the spatial range of the potential. The solvent quality can be shown to scale with the square root of molecular weight, i.e., $z = \frac{z^*}{\chi(b)^3}\sqrt{N_b}$. In this equation, χ is a function of b due to finite extensibility of the chain, where in the limit of a Hookean spring $\chi = 1$. $\chi(b)$ can be directly obtained for a given force extension behavior [31], and for the case of the FENE force law it is $\chi^2 = \frac{b}{b+5}$. The effective radius for EV interaction is specified with $r_{c,F}$. Note that the conservative spring force on the beads is obtained by using the relation $\boldsymbol{F}^{S} = -\tilde{\tilde{\boldsymbol{B}}}^{T} \cdot \boldsymbol{F}^{c}$. Also, the term $\boldsymbol{D} \cdot \boldsymbol{F}^{\phi}$ can be interpreted as a vector containing the velocity $\boldsymbol{u}$ of all beads.

$\boldsymbol{C}$ is the coefficient matrix which satisfies the equation

$$\boldsymbol{D} = \boldsymbol{C} \cdot \boldsymbol{C}^{T}. \quad (6)$$

The solution to Eq. (6) is not unique. Cholesky decomposition is a straightforward approach to determine $\boldsymbol{C}$; however as its computational expense scales with $O(N^3)$, polynomial approximation of $\boldsymbol{D}^{1/2}$, e.g., the Chebyshev or the Krylov based techniques are the two best alternatives [12]. Since the diffusion tensor is symmetric, the Lanczos approach is used to construct the orthonormal basis for the Krylov subspace. In the rest of the paper, "Lanczos" is used to designate the Krylov subspace technique. $\boldsymbol{W}$ is a $3N$ dimensional Wiener process [8], defined for all beads in the simulation box.

The evolution equation for the position of center of mass of the chain i, i.e., $\boldsymbol{r}_{c,i} = (1/N_b)\sum_{\beta=1}^{N_b} \boldsymbol{r}_{i,\beta}$, is written as

$$r_{c,i;n+1}^{q} = r_{c,i;n}^{q} + \frac{1}{N_b} \left[\frac{1}{4} \sum_{\beta=1}^{N_b} u_{\{i,\beta\};n}^{q} \Delta t + \frac{1}{\sqrt{2}} \sum_{\beta=1}^{N_b} \Delta S_{\{i,\beta\};n}^{q} \right], \quad (7)$$

where $\{q = 1,2,3\}$ refers to the three Cartesian coordinates and $\Delta \boldsymbol{S}_n = \boldsymbol{C}_n \cdot \Delta \boldsymbol{W}_n$ is the Brownian displacement after n time steps.

The numerical integration of the governing differential equation [Eq. (1)] can be performed by either the Euler-Murayama or a semi-implicit predictor-corrector scheme [12]. We have implemented both integration methods. In the predictor-corrector scheme, as $\boldsymbol{F}^{\phi}$ gets updated in the corrector steps it is necessary to calculate $\boldsymbol{D} \cdot \boldsymbol{F}^{\phi}$, which in the case of large N is a very expensive procedure. On the other hand, at equilibrium where $10^{-3} \lesssim \Delta t \leqslant 10^{-2}$, the Euler-Murayama method results in sufficient accuracy. Hence, this method is used in the computations reported in this paper. However, in nonequilibrium simulations, the difference between the appropriate time step for the Euler-Murayama and the

semi-implicit predictor-corrector can be around three orders of magnitude, particularly at high flow strength [32]. This issue, namely, the efficiency of the predictor-corrector versus the Euler-Murayama method for nonequilibrium simulations, will be addressed in a future communication.

It should also be noted that the conversion of $\tilde{\boldsymbol{Q}}$ to $\boldsymbol{R}$, where $\boldsymbol{R}_{i,\beta} = \boldsymbol{r}_{i,\beta} - \boldsymbol{r}_{c,i}$, is achieved with the help of the equation $\boldsymbol{R} = \tilde{\boldsymbol{\mathcal{B}}} \cdot \tilde{\boldsymbol{Q}}$, and the matrix $\tilde{\boldsymbol{\mathcal{B}}}$ is an $N_c \times N_c$ matrix with only diagonal nonzero blocks $\boldsymbol{\mathcal{B}}$ that are defined as

$$\boldsymbol{\mathcal{B}}_{\beta\gamma} = \begin{cases} \frac{\gamma}{N_b}\boldsymbol{\delta}, & \gamma < \beta, \\ -\left[1 - \frac{\gamma}{N_b}\right]\boldsymbol{\delta}, & \gamma \geqslant \beta, \end{cases} \tag{8}$$

B. Ewald representation of an infinite sum

It was stated in the Sec. II A that $\boldsymbol{D}_{\nu\mu}$ contains the hydrodynamic interaction of bead ν with beads μ not only in the simulation box (primary image) but also in all other replicas of the primary image which span the entire space. It is known that HI is long range in nature, i.e., it scales with $1/r$. Therefore, to overcome the convergence issue of the infinite sum, Beenakker [14] used the idea of Ewald summation to split the sum into two exponentially decaying sums, one in real space and the other in reciprocal (or Fourier) space:

$$\boldsymbol{D}_{\nu\mu} = \boldsymbol{D}^{\text{self}}_{\nu\mu} + \boldsymbol{D}^{\text{real}}_{\nu\mu} + \boldsymbol{D}^{\text{recip}}_{\nu\mu}, \tag{9}$$

where the first term is the correction due to self interaction and the last two terms are the contributions of the real and the reciprocal space summations, respectively. Each term is written as

$$\boldsymbol{D}^{\text{self}}_{\nu\mu} = \left(1 - \frac{6}{\sqrt{\pi}}\alpha a + \frac{40}{3\sqrt{\pi}}\alpha^3 a^3\right)\delta_{\nu\mu}\boldsymbol{\delta}, \tag{10a}$$

$$\boldsymbol{D}^{\text{real}}_{\nu\mu} = {\sum_{\boldsymbol{n}\in Z^3}}' \boldsymbol{M}^{(1)}_{\alpha}(\boldsymbol{r}_{\nu\mu,\boldsymbol{n}}), \tag{10b}$$

$$\boldsymbol{D}^{\text{recip}}_{\nu\mu} = \frac{1}{V}\sum_{\boldsymbol{k}_\lambda \neq 0} \exp(-i\boldsymbol{k}_\lambda \cdot \boldsymbol{r}_{\mu\nu})\boldsymbol{M}^{(2)}_{\alpha}(\boldsymbol{k}_\lambda). \tag{10c}$$

Here, the parameter α controls the relative computational load between the reciprocal and the real space and hence their convergence rate. If large values of α are selected, the real space sum converges faster than its reciprocal counterpart. $\delta_{\nu\mu}$ is the Kronecker delta. The vector $\boldsymbol{n} = (n_x, n_y, n_z)$ with integer components specifies all images including the primary image ($\boldsymbol{n} = 0$). However, as indicated by the prime on the summation over $\boldsymbol{n}$, $\boldsymbol{n} = 0$ is omitted for $\nu = \mu$. $\boldsymbol{M}^{(1)}_{\alpha}$ is a 3×3 matrix which is a function of the vector connecting bead μ to beads ν in different images, i.e., $\boldsymbol{r}_{\nu\mu,\boldsymbol{n}} = \boldsymbol{r}_\nu - \boldsymbol{r}_\mu + \boldsymbol{n}L$. $\boldsymbol{M}^{(1)}_{\alpha}$ is written as

$$\boldsymbol{M}^{(1)}_{\alpha}(\boldsymbol{r}) = \left[C_1 \operatorname{erfc}(\alpha r) + C_2\frac{\exp(-\alpha^2 r^2)}{\sqrt{\pi}}\right]\boldsymbol{\delta} + \left[C_3 \operatorname{erfc}(\alpha r) + C_4\frac{\exp(-\alpha^2 r^2)}{\sqrt{\pi}}\right]\frac{\boldsymbol{r}\boldsymbol{r}}{r^2}, \tag{11}$$

where $r = |\boldsymbol{r}|$, erfc( ) denotes the complementary error function, and the $C_1, \ldots C_4$ parameters are defined as follows:

$$\begin{aligned} C_1 &= \frac{3a}{4r} + \frac{a^3}{2r^3}, \\ C_2 &= 4\alpha^7 a^3 r^4 + 3\alpha^3 a r^2 - 20\alpha^5 a^3 r^2 - \frac{9}{2}\alpha a + 14\alpha^3 a^3 + \frac{\alpha a^3}{r^2}, \\ C_3 &= \frac{3a}{4r} - \frac{3a^3}{2r^3}, \\ C_4 &= -4\alpha^7 a^3 r^4 - 3\alpha^3 a r^2 + 16\alpha^5 a^3 r^2 + \frac{3}{2}\alpha a - 2\alpha^3 a^3 - \frac{3\alpha a^3}{r^2}. \end{aligned} \tag{12}$$

In Eq. (10c) $\boldsymbol{k}_\lambda = \frac{2\pi}{L}\boldsymbol{l}$ where $\boldsymbol{l} \in Z^3$, and $\boldsymbol{M}^{(2)}_{\alpha}$ is written as

$$\boldsymbol{M}^{(2)}_{\alpha} = m^{(2)}_{\alpha}\left(\boldsymbol{\delta} - \frac{\boldsymbol{k}_\lambda \boldsymbol{k}_\lambda}{k^2}\right), \tag{13}$$

where $k = |\boldsymbol{k}_\lambda|$ and $m^{(2)}_{\alpha}$ is defined as

$$m^{(2)}_{\alpha} = \left(a - \frac{a^3 k^2}{3}\right)\left(1 + \frac{k^2}{4\alpha^2} + \frac{k^4}{8\alpha^4}\right)\frac{6\pi}{k^2}\exp\left[-\frac{k^2}{4\alpha^2}\right]. \tag{14}$$

Equation (10b) implicitly assumes no overlap between the beads, since Beenakker [14] exploited Eq. (3b) to derive the Ewald sum of the RPY formulation for the HI tensor. To remove this assumption, Zhou and Chen [27] introduced a correction which is applied by adding Eq. (3c) to the real part of the diffusion tensor for a pair of beads and subsequently subtracting Eq. (3b) when there is an overlap between the two beads. This is particularly important for the primary image and all the images in its vicinity (27 replicas in total). This approach is equivalent to adding a 3×3 matrix $\boldsymbol{M}^*$ to the right-hand side (RHS) of Eq. (10b) in the case of an overlap [15]:

$$\boldsymbol{D}^{\text{real}}_{\nu\mu} = {\sum_{\boldsymbol{n}\in Z^3}}' \boldsymbol{M}^{(1)}_{\alpha}(\boldsymbol{r}_{\nu\mu,\boldsymbol{n}}) + \boldsymbol{M}^*(\boldsymbol{r}_{\nu\mu,\boldsymbol{n}})(1 - \delta_{\nu\mu}), \tag{15}$$

where $\boldsymbol{M}^*$, which is included only for $\nu \neq \mu$, is defined as

$$\boldsymbol{M}^* = \begin{cases} 0, & r \geqslant 2a, \\ \left[1 - \frac{9r}{32a} - \frac{3a}{4r} - \frac{a^3}{2r^3}\right]\boldsymbol{\delta} + \left[\frac{3r}{32a} - \frac{3a}{4r} + \frac{3a^3}{2r^3}\right]\frac{\boldsymbol{r}\boldsymbol{r}}{r^2}, & r < 2a. \end{cases} \tag{16}$$

C. SPME representation of an infinite sum

In this section, we present the matrix-free approach that is employed for the simulation of semidilute bead-spring polymer solutions. This approach uses the smooth particle mesh Ewald method for the calculation of reciprocal space sum based on the original SPME method for electrostatic interactions by Essmann *et al.* [18]. PME based techniques for dealing with long range HI were also considered previously [6,22,26]. Our matrix-free approach is very similar to the one implemented by Liu and Chow [26] for simulation of suspension of Brownian particles, and hence we try to follow similar nomenclatures and notations throughout this section.

Primarily, we are looking to obtain $\boldsymbol{u}_\nu = (\boldsymbol{D}\cdot\boldsymbol{F}^\phi)_\nu = \sum_\mu \boldsymbol{D}_{\nu\mu}\cdot\boldsymbol{F}^\phi_\mu$. The RPY Ewald operator on the force $\boldsymbol{F}^\phi$ can be expressed as

$$\boldsymbol{D}\cdot\boldsymbol{F}^\phi = \overbrace{\boldsymbol{D}^{\text{real}}\cdot\boldsymbol{F}^\phi}^{\boldsymbol{u}^{\text{real}}} + \overbrace{\boldsymbol{D}^{\text{recip}}\cdot\boldsymbol{F}^\phi}^{\boldsymbol{u}^{\text{recip}}} + \overbrace{\boldsymbol{D}^{\text{self}}\cdot\boldsymbol{F}^\phi}^{\boldsymbol{u}^{\text{self}}}, \tag{17}$$

where the three terms on the RHS are related to the real, reciprocal, and self parts of the RPY interaction tensor. The reciprocal term can be written as

$$\boldsymbol{u}^{\text{recip}}_\nu = \sum_{\boldsymbol{k}_\lambda\neq 0}\sum_{\mu=1}\exp(-i\boldsymbol{k}_\lambda\cdot\boldsymbol{r}_{\mu\nu})\boldsymbol{M}^{(2)}_\alpha(\boldsymbol{k}_\lambda)\cdot\boldsymbol{F}^\phi_\mu. \tag{18}$$

After some minor manipulations, Eq. (18) can be rewritten as

$$\boldsymbol{u}^{\text{recip}}_\nu = \sum_{\boldsymbol{m}\neq 0,\mu}\exp(2\pi i\boldsymbol{m}\cdot\boldsymbol{r}_\nu)\boldsymbol{M}^{(2)}_\alpha(2\pi\boldsymbol{m}') \times\exp(-2\pi i\boldsymbol{m}\cdot\boldsymbol{r}_\mu)\boldsymbol{F}^\phi_\mu, \tag{19}$$

where $\boldsymbol{m}$, the reciprocal lattice vector, is defined by $\boldsymbol{m} = m_1a_1^* + m_2a_2^* + m_3a_3^*$, where a_q^* parameters are the conjugate reciprocal vectors defined based on a_s parameters which are the unit vectors that form the edges of the simulation box. The two aforementioned vectors are related as $a_q^*\cdot a_s = \delta_{qs}$. For a box with dimensions $L\times L\times L$, $\boldsymbol{m} = \frac{1}{L}(m_1,m_2,m_3)$, where the m_q range is $0,\ldots,K-1$. Note that the periodic feature of complex exponentials was used to map the range of $\boldsymbol{k}_\lambda$ to those of mesh grid points. Also, $\boldsymbol{k}_\lambda = 2\pi\boldsymbol{m}'$ and $\boldsymbol{m}' = m_1'a_1^* + m_2'a_2^* + m_3'a_3^*$, where m_q' is defined as

$$m_q' = \begin{cases} m_q, & 0\leqslant m_q\leqslant K/2,\\ m_q - K, & \text{otherwise.}\end{cases} \tag{20}$$

The term $\sum_\mu \exp(-2\pi i\boldsymbol{m}\cdot\boldsymbol{r}_\mu)\boldsymbol{F}^\phi_\mu$ can be interpreted as the discrete Fourier transform of the forces $\boldsymbol{F}^\phi_\mu$, i.e., $\tilde{\boldsymbol{F}}(\boldsymbol{m})$:

$$\boldsymbol{u}^{\text{recip}}_\nu = \sum_{\boldsymbol{m}\neq 0}\exp(2\pi i\boldsymbol{m}\cdot\boldsymbol{r}_\nu)\boldsymbol{M}^{(2)}_\alpha(2\pi\boldsymbol{m}')\cdot\tilde{\boldsymbol{F}}(\boldsymbol{m}); \tag{21}$$

here the summation over $\boldsymbol{m}$ can be regarded as the inverse Fourier transform.

1. Spreading the forces

As it was shown above,

$$\tilde{\boldsymbol{F}}(\boldsymbol{m}) = \sum_\mu\left\{\prod_q\exp\left(-2\pi i\frac{m_q}{K}\xi_{\mu,q}\right)\right\}\boldsymbol{F}^\phi_\mu; \tag{22}$$

here $\xi_{\mu,q} = K/Lr_{\mu,q}$. To take advantage of the fast Fourier transform (FFT), the force on the nonequally spaced particles cannot be directly used. Instead, the forces have to be spread onto a regular mesh, which is the primary task of PME based methods. This can also be achieved by interpolating a complex exponential on the regular grid defined earlier using the properties of Euler exponential splines [18]:

$$\exp\left(-2\pi i\frac{m_q}{K}\xi_{\mu,q}\right) \approx b_q^*(m_q)\sum_{k=-\infty}^{+\infty}M_p(\xi_{\mu,q}-k) \times\exp\left(-2\pi i\frac{m_q}{K}k\right). \tag{23}$$

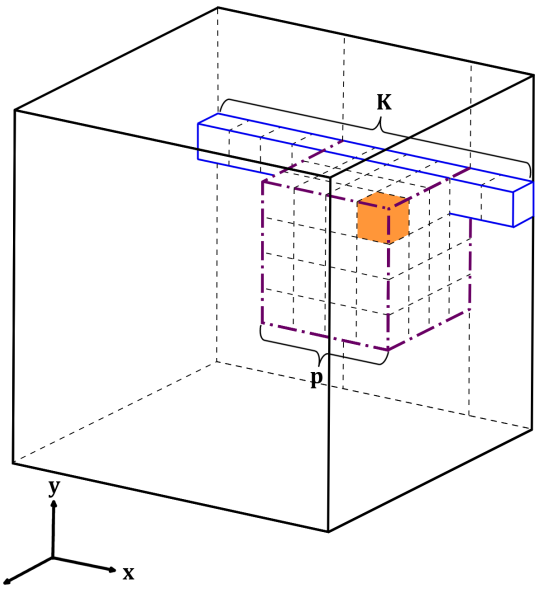

FIG. 1. (Color online) Spreading the force on a regular mesh, where the cells represent the grid points. The filled cell is the nearest point to the particle and the cells bounded by the dash-dotted line are p^3 grid points at which the force on the particle is distributed.

The functions M_p are the cardinal B-splines of order p (piecewise polynomials of degree $p-1$):

$$M_2(x) = \begin{cases}1-|x-1|, & 0\leqslant x\leqslant 2,\\ 0, & x<0,\ x>2,\end{cases}$$
$$M_p(x) = \frac{x}{p-1}M_{p-1}(x) + \frac{p-x}{p-1}M_{p-1}(x-1), \quad p>2. \tag{24}$$

The M_p functions spread the forces for the particles near the boundaries based on the corresponding periodic grids inside the simulation box. $b_q^*(m_q)$ is the complex conjugate of $b_q(m_q)$ which is given by $b_q(m_q) = \frac{\exp[2\pi i(p-1)m_q/K]}{\sum_{k=0}^{p-2}M_p(k+1)\exp(2\pi i m_q k/K)}$. The distribution of the forces on the nearby grids is depicted in Fig. 1.

2. Forward Fourier transform (3D FFT)

Now Eq. (22) is approximated as

$$\tilde{\boldsymbol{F}}(\boldsymbol{m}) \approx \hat{\boldsymbol{F}}(\boldsymbol{m}) = \left\{\prod_q b_q^*(m_q)\right\}\mathscr{F}[\boldsymbol{F}^{\phi,g}](m_1,m_2,m_3), \tag{25}$$

where $\boldsymbol{F}^{\phi,g}$ are the forces on the regular grid, i.e., $\boldsymbol{F}^{\phi,g} = \sum_\mu\{\prod_q M_p(\xi_{\mu,q}-k_q)\}\boldsymbol{F}^\phi_\mu$, and $\mathscr{F}[\cdot]$ is the forward FFT operator which is defined as

$$\mathscr{F}[\boldsymbol{F}^{\phi,g}](m_1,m_2,m_3) = \sum_{k_1,k_2,k_3=0}^{K-1}\left\{\prod_q\exp\left[-2\pi i\frac{m_qk_q}{K}\right]\right\} \times\boldsymbol{F}^{\phi,g}(k_1,k_2,k_3). \tag{26}$$

3. Backward Fourier transform (3D IFFT)

Now back to Eq. (21), again we use the properties of Euler exponentials but this time for $\exp(2\pi i \boldsymbol{m}\cdot\boldsymbol{r}_v)$:

$$\boldsymbol{u}_v^{recip} = \sum_{k_1,k_2,k_3=0}^{K-1} \left\{\prod_q M_p(\xi_{v,q}-k_q)\right\} \sum_{\boldsymbol{m}\neq 0} \left\{\prod_q b_q(m_q) b_q^*(m_q)\right\} \boldsymbol{M}_\alpha^{(2)}(2\pi\boldsymbol{m}')\cdot\mathscr{F}[\boldsymbol{F}^{\phi,g}](m_1,m_2,m_3) \left\{\prod_q \exp\left[2\pi i \frac{m_q k_q}{K}\right]\right\}. \tag{27}$$

If we define the influence function $\boldsymbol{I}(m_1,m_2,m_3)$, which is a 3×3 matrix at each of $K\times K\times K$ grid points, as $\boldsymbol{I}(m_1,m_2,m_3) = \{\prod_q |b_q(m_q)|^2\}\boldsymbol{M}_\alpha^{(2)}(2\pi\boldsymbol{m}')$, then

$$\boldsymbol{u}_v^{\text{recip}} = \sum_{k_1,k_2,k_3=0}^{K-1} \left\{\prod_q M_p(\xi_{v,q}-k_q)\right\} \mathscr{F}^{-1}[\boldsymbol{I}\cdot\mathscr{F}[\boldsymbol{F}^{\phi,g}]], \tag{28}$$

where

$$\mathscr{F}^{-1}[\boldsymbol{I}\cdot\mathscr{F}[\boldsymbol{F}^{\phi,g}]](k_1,k_2,k_3) = \sum_{\boldsymbol{m}\neq 0} (\boldsymbol{I}\cdot\mathscr{F}[\boldsymbol{F}^{\phi,g}])(m_1,m_2,m_3) \left\{\prod_q \exp\left[2\pi i \frac{m_q k_q}{K}\right]\right\}.$$

4. Interpolation of the velocities to particle positions

Equation (28) is interpreted as the back interpolation of the velocities on the grid, i.e., $\boldsymbol{u}^{recip,g}(k_1,k_2,k_3) = \mathscr{F}^{-1}[\boldsymbol{I}\cdot\mathscr{F}[\boldsymbol{F}^{\phi,g}]](k_1,k_2,k_3)$, to the positions of the particles, which is done with the same functions used for the spreading task.

D. Accuracy of Ewald summation

The parameters which influence the accuracy of the Ewald summation technique are identified as Ewald parameter α, which changes the distribution of load between real and reciprocal space sums, real space cutoff radius $r_{c,D}$, and $k_{\max}$, a parameter which defines the accuracy of the reciprocal space summation. As shown below, these three parameters are related. In fact, only $r_{c,D}$ and $k_{\max}$ are required to be optimized to ensure the accuracy of the Ewald summation.

Fincham [33] proposed a method to choose the optimal values of these parameters for electrostatic interactions. Similar discussions were made by Jain *et al.* [15] for the hydrodynamic interactions. Following the arguments made by Fincham [33] and Jain *et al.* [15] and based on Eqs. (11) and (14), the convergence of the real and reciprocal space sums is determined by $\exp(-\alpha^2 r_{c,D}^2)$ and $\exp(-\frac{k^2}{4\alpha^2})$, respectively. M_{exp} is defined such that $\exp(-M_{\text{exp}}^2)$ becomes negligibly small, therefore

$$\alpha = \frac{M_{\text{exp}}}{r_{c,D}}, \quad k_{\max} = \frac{2M_{\text{exp}}^2}{r_{c,D}}. \tag{29}$$

Hence, the accuracies of the real and reciprocal space sums and therefore the Ewald summation are tuned using $r_{c,D}$ and M_{exp}.

III. OPTIMAL FEATURES OF THE MATRIX-FREE ALGORITHM

This section summarizes the main distinguishing features of the matrix-free method as compared to the original Ewald summation as well as the straightforward implementations of PME based algorithms. As the term "matrix-free" implies, all instances of matrix variables are avoided in the numerical calculations. Specifically, $\boldsymbol{D}^{\text{recip}}\cdot\boldsymbol{F}^{\phi}$ is directly calculated using the SPME technique without explicitly calculating $\boldsymbol{D}^{\text{recip}}$. $\boldsymbol{D}^{\text{self}}$ is diagonal and therefore is calculated efficiently without storing the matrix. $\boldsymbol{D}^{\text{real}}$ is, in general, a dense matrix which is made sparse in the matrix-free approach (as will be described in detail in Sec. III A). $\tilde{\tilde{\boldsymbol{B}}}$ and $\tilde{\boldsymbol{\mathcal{B}}}$ are both highly sparse and are treated efficiently using optimized MKL routines for sparse matrices.

A. Sparsification of $\boldsymbol{D}^{\text{real}}$

As pointed out earlier in the paper, $\boldsymbol{D}^{\text{real}}$ has to be sparse in the matrix-free approach. To this end, $r_{c,D}$ is made very small such that each bead interacts with only a few surrounding beads. The sparse $\boldsymbol{D}^{\text{real}}$ is then constructed using the combination of the cell linked list and the Verlet list [34,35]. As the elements $\boldsymbol{D}_{v\mu}^{\text{real}}$ are 3×3 tensors, the sparse variant of $\boldsymbol{D}^{\text{real}}$ is stored based on the MKL's block compressed sparse row format [26]. Furthermore, as $\boldsymbol{D}^{\text{real}}$ is symmetric, MKL's sparse matrix vector (SpMV) operations for symmetric matrices are used in the calculation of $\boldsymbol{D}^{\text{real}}\cdot\boldsymbol{F}^{\phi}$ and SpMV operations which are involved in the Lanczos algorithm.

B. Efficient implementation of SPME

Following the original matrix-free approach for HI [26], the $N\times K^3$ transformation matrix P is defined as

$$P(\mu,k_1+k_2K+k_3K^2) = \left\{\prod_{q=1}^{3} M_p(\xi_{\mu,q}-k_q)\right\}; \tag{30}$$

note that P has only p^3 nonzero terms at each row, which means that it is considerably sparse. It is stored according to the well-known compressed sparse row format. In addition, P only depends on the configuration of the system and hence can be computed *a priori* in each time step and reused for all the corresponding SpMV operations involved in the calculation of $\boldsymbol{D}\cdot\boldsymbol{F}^{\phi}$ and the Lanczos decomposition.

Next, the forces on the particles are spread to the regular grid,

$$\left[F_x^{\phi,g}, F_y^{\phi,g}, F_z^{\phi,g}\right] = P^T\cdot\left[F_x^{\phi}, F_y^{\phi}, F_z^{\phi}\right].$$

Then, the forward 3D FFT at all grid points, i.e., $\mathscr{F}[\boldsymbol{F}^{\phi,g}]$, is performed using efficient MKL discrete Fourier transform interfaces. In this regard, an "in-place real-complex" storage scheme is used which is more efficient than the "out-place complex-complex" variant from the memory transactions perspective.

Subsequently, the influence function is applied by calculating $\boldsymbol{I} \cdot \mathscr{F}[\boldsymbol{F}^{\phi,g}]$ again at all grid points. It is obvious from the definition of $\boldsymbol{I}$ in Sec. II C 3 that influence function is not dependent on configuration of the system. Hence, it can be stored once the number of mesh points is known. However, the memory efficient way of storing this function is to store only $\{\prod_q |b_q(m_q)|^2\} m_\alpha^{(2)}$ and calculate $(\boldsymbol{\delta} - \frac{\boldsymbol{k}_\lambda \boldsymbol{k}_\lambda}{k^2})$ on the fly [26]. Furthermore, owing to the inversion symmetry of reciprocal space, the influence function is stored for about half of the grid points [15,26]. This operation is followed by the inverse 3D FFT, where $\mathscr{F}^{-1}[\boldsymbol{I} \cdot \mathscr{F}[\boldsymbol{F}^{\phi,g}]]$ is calculated at the regular grid points.

The last step in SPME is to interpolate the velocities at grid points to the location of the particles,

$$\left[u_x^{\text{recip}}, u_y^{\text{recip}}, u_z^{\text{recip}}\right] = P \cdot \left[u_x^{\text{recip},g}, u_y^{\text{recip},g}, u_z^{\text{recip},g}\right].$$

C. The frequency of updating D

One of the strategies used to increase the speed of the calculation is to update the diffusion tensor after every λ_{RPY} time steps [12,26,32,36]. The appropriate choice of λ_{RPY} was investigated in our recent paper [12] and it was found that $\lambda_{RPY}\Delta t$ is the key parameter which determines the accuracy of the integration. Here, our implementation benefits from this approach. Specifically, in cases where the diffusion tensor changes slowly, $\lambda_{RPY} > 1$ is utilized.

IV. RESULTS AND DISCUSSION

A. Error definition in iterative Lanczos

In order to track the accuracy of the Lanczos approximation method, one has to define the error in estimation of correlated vectors. We use the error expression proposed by Ando *et al.* [36] based on two consecutive iterations in Lanczos algorithm, i.e., $m-1$ and m:

$$E^{(m)} = \frac{||\Delta \boldsymbol{S}^{(m)} - \Delta \boldsymbol{S}^{(m-1)}||_2}{||\Delta \boldsymbol{S}^{(m-1)}||_2}. \tag{31}$$

In case $\lambda_{RPY} \neq 1$, the first column of $\Delta \boldsymbol{S}$ is used to calculate the error. The convergence criteria is met when the value of error falls below a certain threshold. Based on the results presented by Ando *et al.* [36], $E^{(m)} = 0.01$ was selected as an indication of sufficient accuracy to reproduce the results of the Cholesky decomposition within the statistical error.

B. Algorithm verification

To validate the accurate implementation of the Ewald and matrix-free algorithms, the equilibrium properties of the dilute and semidilute polymer solutions are evaluated in θ solvent and good solvent. To this end, the mean-square displacement (MSD) as a function of time τ and long-time diffusivity of center of mass D_{cm} is used to track movement of the chains in a θ solvent:

$$D_{cm} = \lim_{\tau\to\infty} \frac{\text{MSD}(\tau)}{6\tau} = \lim_{\tau\to\infty} \frac{\langle (\boldsymbol{r}_c(t+\tau) - \boldsymbol{r}_c(t))^2 \rangle}{6\tau}, \tag{32}$$

where t is an arbitrary initial time. In case of θ solvent or good solvent, the mean-square end-to-end distance and the mean-square radius of gyration are utilized to ascertain the accuracy of the matrix-free algorithm.

$$\left\langle R_{ee}^2 \right\rangle = \langle (\boldsymbol{r}_{N_b} - \boldsymbol{r}_1)^2 \rangle, \tag{33}$$

$$\left\langle R_g^2 \right\rangle = \frac{1}{N_b} \left\langle \sum_\beta (\boldsymbol{r}_\beta - \boldsymbol{r}_c)^2 \right\rangle. \tag{34}$$

1. *Diffusivity in θ solvent*

To validate the diffusivity of center of mass in semidilute regime, the values of MSD as a function of normalized time and D_{cm} for a system of multichains with $N_b = 20$ at different c/c^* were computed (see Fig. 2).

The overlap concentration is calculated using $c^* = N/(4\pi/3)R_{g,0}^3$ where $R_{g,0}$ is the radius of gyration at infinite

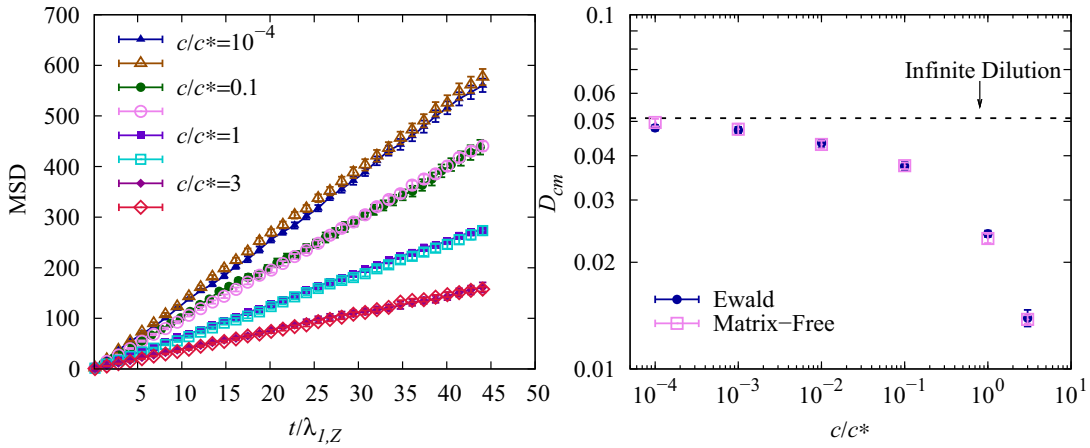

FIG. 2. (Color online) (a) The mean-square displacement and (b) the diffusivity of center of mass in θ solvent obtained using both Ewald and matrix-free techniques. The filled and open symbols represent the results for the Ewald and the matrix-free algorithms, respectively.

TABLE I. The simulation parameters for multichain systems at different concentrations. L is obtained using the analytical radius of gyration of polymers in a θ solvent. The simulation results of end-to-end distance and radius of gyration are given for both the Ewald and the matrix-free algorithms.

			Ewald			Matrix-free		
c/c^*	L	M_{exp}	$r_{c,D}$	$\langle R_{ee}^2 \rangle$	$\langle R_g^2 \rangle$	$r_{c,D}$	$\langle R_{ee}^2 \rangle$	$\langle R_g^2 \rangle$
10^{-4}	297.73	3.5	350	57.1 ± 0.3	10.0 ± 0.04	310	57.5 ± 0.3	10.0 ± 0.04
10^{-3}	138.2	3.5	200	57.2 ± 0.4	10.0 ± 0.05	200	57.3 ± 0.1	9.9 ± 0.01
10^{-2}	64.15	3.75	100	57.6 ± 0.5	10.1 ± 0.06	100	56.4 ± 0.5	9.9 ± 0.06
0.1	29.77	4.25	45	56.5 ± 0.4	9.9 ± 0.05	7	58.0 ± 0.4	10.1 ± 0.05
1	18.76	3.75	20	57.6 ± 0.5	10.1 ± 0.07	5	57.8 ± 0.4	10.1 ± 0.06
3	13	3.75	20	57.1 ± 0.6	10.0 ± 0.08	4	57.1 ± 0.7	10.0 ± 0.07

dilution limit. It is known that in a θ solution (irrespective of concentration regime) $\langle R_g^2 \rangle = \chi^2 \frac{N_b^2 - 1}{2N_b}$ and $\langle R_{ee}^2 \rangle = 3\chi^2(N_b - 1)$ [7,37]. Our simulations for low concentrations, i.e., $c/c^* = 10^{-4}$–10^{-1}, typically consist of an equilibrium run of around 10 dimensionless longest relaxation times, λ_1, followed by a production run of from 25 to 50 relaxation times. As the macromolecules are in a θ solvent, λ_1 is estimated using the Zimm formula, $\lambda_{1,Z} \approx \frac{1.22 N_b^{3/2}}{h^* \pi^2}$. $N_c = 20$ was used in the simulations for low concentrations and hence the final results of MSD and D_{cm} are reported based on averaging over 20–50 independent runs. For higher concentrations, namely $c/c^* = 1$ and 3, $N_c = 50$ with equilibrium and production runs similar to lower concentrations was used. The parameter h^* was chosen to be 0.25 and the time step size is 0.01. The springs are assumed to be Hookean. The parameters specifying the accuracy of Ewald and matrix-free algorithms are given in Table I. The $r_{c,D}$ for the SPME algorithm is chosen such that the diffusion tensor remain semi-positive-definite throughout the course of simulation. Since there is a large degree of overlap between the beads in a θ solution, the minimum cut off radii for very dilute cases were found to be on the order of the box dimension. Subsequently, the real-space part of $\boldsymbol{D} \cdot \boldsymbol{F}^{\phi}$ for the cases where $r_c > 0.42L$ is obtained using dense matrix calculations. For matrix-free simulations, the degree of B-splines is equal to 4.

The asymptotic behavior of D_{cm} at low concentrations is further compared against the value of D_{cm} at infinitely dilute solution at similar conditions using an algortihm recently developed in our group [12]. It is shown in Fig. 2(b) that the value of D_{cm} at very low concentrations is correctly approaching the infinitely dilute solution results. Furthermore, the differences between the Ewald and matrix-free results for both MSD and D_{cm} lie within the statistical error bar [see Figs. 2(a) and 2(b)]. Note that the approach and the parameters are chosen to closely correspond to simulations of Jain *et al.* [15]. Hence, the results are in very good agreement with their findings. Moreover, the radius of gyration and end-to-end distance for all cases studied in this section were in excellent agreement with the theoretical values (see Table I).

2. *Static dimension in good solvent*

In this section, the behavior of multichain systems in good solvents is evaluated both in dilute and semidilute solutions (see Fig. 3). The static size of the chains for low concentrations ($c/c^* < 0.1$) is expected to be equivalent to the dimensions of the chains in infinite dilution [11,16]. Similar to an infinitely dilute solution, the dimensional scaling for multichain systems with low concentration is expected to follow $\langle R_g^2 \rangle \propto N_b^{2\nu}$, where ν is the effective EV exponent. Again the conventional BDS algorithm by Saadat and Khomami [12] is employed to determine the dimension of the macromolecules at $c/c^* = 0$. It was found by Saadat and Khomami [12] that the effective excluded volume exponent based on center-of-mass diffusivity for the bead-spring system with FENE force law and $b = 20$ is $\nu_{cm} = 0.581 \pm 0.01$. Following the same parameter setting in the infinitely dilute case, $z\chi^3 = 1$ is selected for $N_b = 10$ and is increased based on $z\chi^3 \propto \sqrt{N_b}$ for higher number of beads. The broadness of the EV potential is determined based on $d^* = z^{*1/5}$ [38]. The simulations in this section involve an equilibrium run of more than one chain relaxation time and the production run of between 5 to 15 relaxation times. The chain relaxation time here is approximated using $\lambda \approx R_g^2/6D_{cm}$ (see Table II). For the simulations of this section, the degree of B-splines is set as 6. It should be noted that the simulation box size is selected such that $L \geqslant 2R_{ee}$ to prevent any unrealistic interaction of the chain with itself.

At higher concentrations, however, one chain is surrounded by several other chains. Therefore, an individual

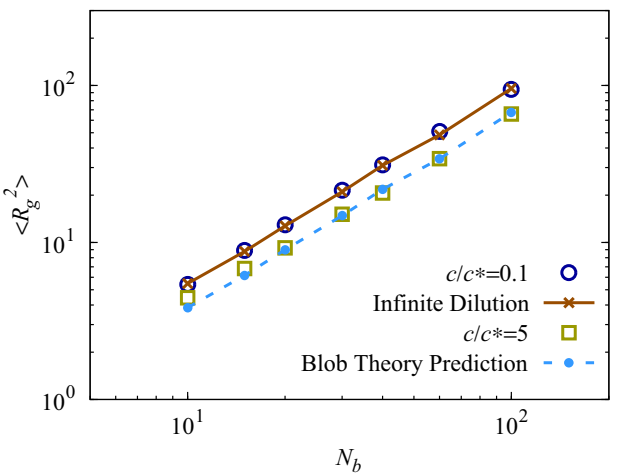

FIG. 3. (Color online) Mean-square radius of gyration as a function of number of beads in a multichain system with $c/c^* = 0.1$ and 5. The error bars are smaller than the symbols.

TABLE II. The parameters of dilute and semidilute multichain systems determined based on the diffusivity, radius of gyration, and end-to-end distance of the chains in infinitely dilute polymer solution.

			$c/c^* = 0.1$			$c/c^* = 5$		
N_b	$D_{cm,0}$	$\langle R^2_{g,0}\rangle$	λ	L (L/R_{ee})	N_c	λ	L (L/R_{ee})	N_c
10	0.074 ± 0.005	5.476 ± 0.2	12.35	28.23 (5)	42	21.55	14.21 (3)	267
20	0.048 ± 0.003	12.75 ± 0.4	44.27	44.04 (5)	45	77.25	22.17 (3)	286
40	0.031 ± 0.002	30.96 ± 1.0	166.45	69.55 (5)	47	290.44	23.34 (2)	88
60	0.026 ± 0.002	48.43 ± 1.6	310.45	87.83 (5)	48	541.70	29.48 (2)	91
100	0.018 ± 0.001	95.64 ± 3.1	871	127.1 (5)	52	1520	42.66 (2)	99

macromolecule experiences the repulsion due to the other chains in its vicinity which results in reduction of its dimensions. The blob theory takes this into consideration when determining the R_g scaling [11,39],

$$R_g/R_{g,0} = (c/c^*)^{-(2\nu-1)/(6\nu-2)}. \tag{35}$$

Here, we use this known theoretical fact to validate our matrix-free algorithm. To this end, the radius of gyration of multichain systems with relative concentration $c/c^* = 5$ is determined. It is clearly seen in Fig. 3 that the results of our BDS algorithm for multichain systems in good solvents and dilute solutions match the infinite dilution case. Although the theoretical scaling based on the blob model holds true for long chains, the simulated static radius of gyration is consistent with the predictions from blob theory for $N_b \gtrsim 20$. Note that the relaxation time in the semidilute regime is approximated based on diffusivity obtained by $D_{cm}/D_{cm,0} = (c/c^*)^{-(1-\nu)/(3\nu-1)}$ [11]. Moreover, as far as the calculation of box dimension is concerned, $R_{ee}/R_{ee,0}$ follows the same scaling as that of the radius of gyration.

C. Computational time scaling

The recent comparison of the BDS algorithm with the lattice Boltzmann molecular dynamics (LB/MD) by Jain *et al.* [15] indicated that the BDS with improved Ewald summation along with the Chebyshev polynomial for decomposition is far more expensive than LB/MD. The matrix-free implementation for polymer solutions developed in this work is expected to substantially reduce the computational cost difference between BDS and other fast mesoscale simulation methods, such as dissipative particle dynamics (DPD) and LB/MD. Furthermore, the total execution time versus the number of beads in the simulation box is anticipated to scale as $O(N \log N)$. To this end, the code is tested on a 16-core Intel Xeon E5-2670 processor by tracking the wall-clock time spent in the simulation of a multichain system with $N_b = 40$. The chains are assumed to be in a good solvent, and hence there is EV interaction between each pair of particles and HI is considered with $h^* = 0.25$. A box with side length equal to 27.8 is considered to ensure $L \geqslant 2R_{ee}$. The Ewald summation parameters are selected as $M_{\text{exp}} = 4.25$ and $r_{c,D} = 11$ and 5 for Ewald and matrix-free methods, respectively. The values of M_{exp} and $r_{c,D}$ for the matrix-free approach imply $K = 63$. The degree of B-splines is chosen to be 6.

Using these parameters, simulations are performed for about 0.01λ, where λ is obtained using the same procedure outlined in Sec. IV B 2. This corresponds to more than 150 time steps with $\Delta t = 10^{-2}$. The execution time per time step is depicted in Fig. 4 as a function of N. It should be noted that the values of the relative error of the mean square radius of gyration at the end of 0.01λ, calculated for the matrix-free approach compared to the corresponding results of the Ewald algorithm, were less than 5×10^{-4} for all the cases studied in this section.

Clearly the matrix-free implementation with proper choice of the Ewald parameters can result in a total execution time which scales as $O(N \log N)$. As expected, the scaling of execution time for the original Ewald algorithm is as $O(N^2)$, as the algorithm uses a constant $r_{c,D}$ [15].

V. CONCLUSIONS

In this article, a matrix-free approach is presented to enhance the efficiency of the BDS for a large system of macromolecules which are coupled through hydrodynamic interaction and excluded volume forces. The advantages of the matrix-free algorithm over the conventional BDS are due to the fact that all matrices involved in BDS are treated as sparse matrices, which in turn results in considerable speed-up. Moreover, the matrix-free implementation benefits from using the SPME method in construction of the diffusion tensor

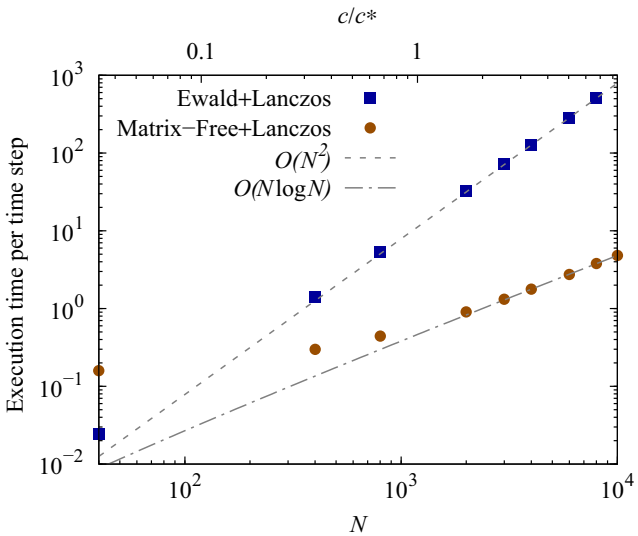

FIG. 4. (Color online) Execution time per time step for Ewald as well as matrix-free algorithms.

along with the Lanczos method for computation of Brownian displacements.

The fidelity and computational efficiency of the matrix-free approach in equilibrium condition is shown by evaluating the mean-square displacement and the averaged diffusivity of the center of mass for a broad range of concentrations in θ solvent as well as the mean-square end-to-end distance and the mean-square radius of gyration for chains with different degrees of fine-graining in a θ solvent or good solvent. The matrix-free results for center-of-mass diffusivity are found to be in excellent agreement with the ones obtained using the original Ewald summation technique. Moreover, the asymptotic values of diffusivity at very low concentrations correctly approach the infinite dilution case. The radius of gyration of chains with different numbers of beads in the dilute regime of concentration was predicted to be consistent with an infinitely dilute system. At higher concentration, namely $c/c^* = 5$, the results of $\langle R_g^2 \rangle$ are in a very good agreement with the values predicted by blob theory [11,39].

Last, it is shown that the computational cost of the matrix-free technique is reduced by more than two orders of magnitude compared to conventional BDS for systems containing more than 10^3 beads. Furthermore, while the execution time for the method based on the Ewald summation and the Lanczos algorithm results in a computational cost scaling of $O(N^2)$, the matrix-free technique improves the scaling to $O(N \log N)$.

ACKNOWLEDGMENTS

This work was supported by NSF Grant No. EPS-1004083. This research was also supported in part by an allocation of advanced computational resources provided by the university of Tennessee and Oak Ridge National Laboratory joint institute for computational sciences.

[1] A. R. Evans, E. S. G. Shaqfeh, and P. L. Frattini, J. Fluid Mech. **281**, 319 (1994).
[2] E. C. Lee, M. J. Solomon, and S. J. Muller, Macromolecules **30**, 7313 (1997).
[3] J. S. Hur, E. S. G. Shaqfeh, H. P. Babcock, D. E. Smith, and S. Chu, J. Rheol. **45**, 421 (2001).
[4] D. L. Ermak and J. A. McCammon, J. Chem. Phys. **69**, 1352 (1978).
[5] M. X. Fernandes and J. G. de la Torre, Biophys. J. **83**, 3039 (2002).
[6] D. Saintillan, E. Darve, and E. S. G. Shaqfeh, Phys. Fluids **17**, 033301 (2005).
[7] R. B. Bird, C. F. Curtiss, R. C. Armstrong, and O. Hassager, *Dynamics of Polymeric Liquids: Vol. 2, Kinetic Theory*, 2nd ed. (Wiley, New York, 1987).
[8] H. C. Öttinger, *Stochastic Processes in Polymeric Fluids* (Springer-Verlag, Berlin, 1996).
[9] N. J. Woo, E. S. G. Shaqfeh, and B. Khomami, J. Rheol. **48**, 299 (2004).
[10] M. Długosz, P. Zieliński, and J. Ttylska, J. Comput. Chem. **32**, 2734 (2011).
[11] A. Jain, B. Dünweg, and J. R. Prakash, Phys. Rev. Lett. **109**, 088302 (2012).
[12] A. Saadat and B. Khomami, J. Chem. Phys. **140**, 184903 (2014).
[13] S. Somani, E. S. G. Shaqfeh, and J. R. Prakash, Macromolecules **43**, 10679 (2010).
[14] C. W. J. Beenakker, J. Chem. Phys. **85**, 1581 (1986).
[15] A. Jain, P. Sunthar, B. Dünweg, and J. R. Prakash, Phys. Rev. E **85**, 066703 (2012).
[16] C. Stoltz, J. J. de Pablo, and M. D. Graham, J. Rheol. **50**, 137 (2006).
[17] T. Darden, D. York, and L. Pedersen, J. Chem. Phys. **98**, 10089 (1993).
[18] U. Essmann, L. Perera, M. L. Berkowitz, T. D. H. Lee, and L. G. Pedersen, J. Chem. Phys. **103**, 8577 (1995).
[19] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles* (McGraw-Hill, New York, 1981).
[20] M. Deserno and C. Holm, J. Chem. Phys. **109**, 7678 (1998).
[21] E. K. Guckel, Ph.D. thesis, University of Illinois at Urbana-Champaign 1999 (unpublished).
[22] A. Sierou and J. F. Brady, J. Fluid Mech. **448**, 115 (2001).
[23] A. J. Banchio and J. F. Brady, J. Chem. Phys. **118**, 10323 (2003).
[24] J. P. Hernández-Ortiz, J. J. de Pablo, and M. D. Graham, J. Chem. Phys. **125**, 164906 (2006).
[25] J. P. Hernández-Ortiz, J. J. de Pablo, and M. D. Graham, Phys. Rev. Lett. **98**, 140602 (2007).
[26] X. Liu and E. Chow, in *27th IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (IEEE, Piscataway, NJ, 2014).
[27] T. Zhou and S. B. Chen, J. Chem. Phys. **124**, 034904 (2006).
[28] M. Somasi, B. Khomami, N. J. Woo, J. S. Hur, and E. S. Shaqfeh, J. Non-Newton. Fluid Mech. **108**, 227 (2002).
[29] T. Ando and J. Skolnick, Proc. Natl. Acad. Sci. USA **107**, 18457 (2010).
[30] J. R. Prakash and H. C. Öttinger, Macromolecules **32**, 2028 (1999).
[31] P. Sunthar and J. R. Prakash, Macromolecules **38**, 617 (2005).
[32] C. C. Hsieh, L. Li, and R. G. Larson, J. Non-Newton. Fluid Mech. **113**, 147 (2003).
[33] D. Fincham, Mol. Simul. **13**, 1 (1994).
[34] G. S. Grest, B. Dünweg, and K. Kremer, Comput. Phys. Commun. **55**, 269 (1989).
[35] M. Allen and D. Tildesley, *Computer Simulations of Liquids* (Oxford Science, London, 1990).
[36] T. Ando, E. Chow, Y. Saad, and J. Skolnick, J. Chem. Phys. **137**, 064106 (2012).
[37] P. Sunthar, D. A. Nguyen, R. Dubbelboer, J. R. Prakash, and T. Sridhar, Macromolecules **38**, 10200 (2005).
[38] K. S. Kumar and J. R. Prakash, Macromolecules **36**, 7842 (2003).
[39] M. Rubinstein and R. H. Colby, *Polymer Physics* (Oxford University Press, Oxford, 2003).